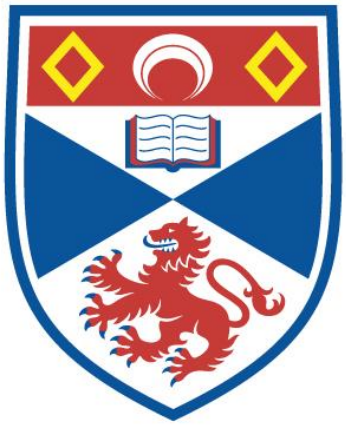
AN APPROACH TO SITUATION RECOGNITION BASED ON LEARNED SEMANTIC MODELS

Graeme Stevenson

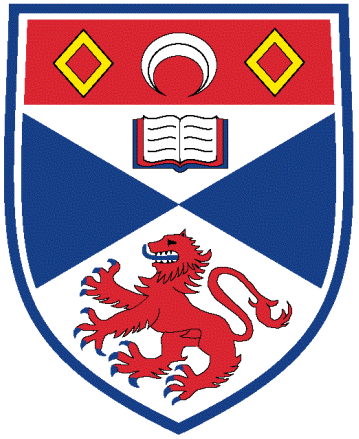
A Thesis Submitted for the Degree of PhD
at the
University of St Andrews



2014

An Approach to Situation Recognition Based on Learned Semantic Models

Graeme Stevenson



The thesis is submitted in partial fulfilment for the degree of
PhD
at the
University of St Andrews

November 2014

DECLARATION

1. Candidate's Declarations

I, Graeme Stevenson, hereby certify that this thesis, which is approximately 56,000 words in length, has been written by me, that it is the record of work carried out by me, or principally by myself in collaboration with others as acknowledged, and that it has not been submitted in any previous application for a higher degree.

I was admitted as a research student in February, 2010 and as a candidate for the degree of Doctor of Philosophy in February, 2010; the higher study for which this is a record was carried out in the University of St Andrews between 2010 and 2014.

Date ..... Signature of candidate .....

2. Supervisor's Declarations

I hereby certify that the candidate has fulfilled the conditions of the Resolution and Regulations appropriate for the degree of Doctor of Philosophy in the University of St Andrews and that the candidate is qualified to submit this thesis in application for that degree.

Date ..... Signature of supervisor .....

3. Permission for Electronic Publication

In submitting this thesis to the University of St Andrews I understand that I am giving permission for it to be made available for use in accordance with the regulations of the University Library for the time being in force, subject to any copyright vested in the work not being affected thereby. I also understand that the title and the abstract will be published, and that a copy of the work may be made and supplied to any bona fide library or research worker, that my thesis will be electronically accessible for personal or research use unless exempt by award of an embargo as requested below, and that the library has the right to migrate my thesis into new electronic forms as required to ensure continued access to the thesis. I have obtained any third-party copyright permissions that may be required in order to allow such access and migration, or have requested the appropriate embargo below.

The following is an agreed request by candidate and supervisor regarding the electronic publication of this thesis:

PRINT COPY
(a) No embargo on print copy

ELECTRONIC COPY
(a) No embargo on electronic copy

Date . . . . . .

Signature of candidate . . . . . .

Signature of supervisor . . . . . .

ABSTRACT

A key enabler of pervasive computing is the ability to drive service delivery through the analysis of *situations*: Semantically meaningful classifications of system state, identified through analysing the readings from sensors attached to the everyday objects that people interact with.

Situation recognition is a mature area of research, with techniques primarily falling into two categories. *Knowledge-based* techniques use inference rules crafted by experts; however often they compensate poorly for sensing peculiarities. *Learning-based* approaches excel at extracting patterns from noisy training data, however their lack of transparency can make it difficult to diagnose errors.

In this thesis we propose a novel hybrid approach to situation recognition that combines both techniques. This offers improvements over each used individually, through not sacrificing the intelligibility of the decision processes that the use of machine learning alone often implies, and through providing better recognition accuracy through robustness to noise typically unattainable when developers use knowledge-based techniques in isolation.

We present an ontology model and reasoning framework that supports the uniform modelling of pervasive environments, and infers additional knowledge from that which is specified, in a principled way. We use this as a basis from which to learn situation recognition models that exhibit comparable performance with more complex machine learning techniques, while retaining intelligibility. Finally, we extend the approach to construct ensemble classifiers with either improved recognition accuracy, intelligibility or both.

To validate our approach, we apply the techniques to real-world data sets collected in smart-office and smart-home environments. We analyse the situation recognition performance and intelligibility of the decision processes, and compare the results to standard machine learning techniques and results published in the literature.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

FOR CLAIRE, THE GIRL WHO WAITED.

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor, Professor Simon Dobson, for giving me the opportunity to undertake a Ph.D, for many fruitful discussions and advice over the past few years, and for giving me the encouragement and motivation to get to this point, which seemed so far away so recently!

I would also like to thank many of my great colleagues from St Andrews and from my earlier time at UCD Dublin, for helping to create engaging research environments that were fun to work in, from which collaborations arose, and where many an interesting idea was discussed over a coffee or a beer. There are too many of you to mention individually, and at this late hour I run the risk of missing someone out. So, I shall simply say - you know who you are. A special mention goes to Dr Juan Ye, with whom I have collaborated on both sides of the Irish Sea, for all her advice and insights, and for making the projects we've worked on together an enjoyable experience.

Part of this work has been supported by the EU FP7 project "SAPERE - Self-aware Pervasive Service Ecosystems" (256873), and I'd like to thank all members of the SAPERE team for the many interesting discussions and collaborations during and beyond the course of the project.

Finally, I'd like to thank my wife, Claire, and the rest of my family for their love and support. I count myself as being incredibly lucky to have such a strong support network. Without you all, I wouldn't have got this far.

STATEMENT OF COLLABORATIONS

I registered as a PhD candidate at University College Dublin in 2008 before transferring to the University of St Andrews in February, 2010. Two of the publications that support this thesis occurred during this initial period: Stevenson et al. (SKDN09), and Stevenson et al. (SYDN10).

The information space model developed in Chapter 3 of this thesis extends from the above work and two subsequent collaborations: Stevenson et al. (SYD10), and Ye et al. (YSD11a). The extensions presented here that go beyond the work reported in these publications are discussed in the introduction to Chapter 3.

The discussion of related work in Chapter 2 references several further collaborations, listed on the following pages. These publications do not directly relate to the body of this thesis, nor are their contributions claimed as part of this work.

LIST OF PUBLICATIONS

- Graeme Stevenson, Stephen Knox, Simon Dobson, and Paddy Nixon. Ontonym: a collection of upper ontologies for developing pervasive systems. In *CIAO '09: Proceedings of the 1st Workshop on Context, Information and Ontologies*, pages 1–8. ACM, 2009

- Graeme Stevenson, Juan Ye, Simon Dobson, and Paddy Nixon. LOC8: A location model and extensible framework for programming with location. *IEEE Pervasive Computing*, 9(1):28 – 37, January 2010

- Graeme Stevenson, Juan Ye, and Simon Dobson. On the impact of the temporal features of sensed data on the development of pervasive systems. In *PMMPS: Proceedings of the International Workshop on Programming Methods for Mobile and Pervasive Systems*, May 2010

- Juan Ye, Graeme Stevenson, Simon Dobson, Michael O'Grady, and Gregory O'Hare. PI: perceiver and interpreter of smart home datasets. In *Proceedings of the 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth 2011)*, May 2011

- Graeme Stevenson and Simon Dobson. Sapphire: Generating Java runtime artefacts from OWL ontologies. In *Proceedings of the Third International Workshop on Ontology-Driven Information Systems Engineering*, June 2011

- Juan Ye, Graeme Stevenson, and Simon Dobson. A top-level ontology for smart environments. *Pervasive and Mobile Computing*, 7(3):359 – 378, 2011. Knowledge-Driven Activity Recognition in Intelligent Environments

- Mirko Viroli, Franco Zambonelli, Graeme Stevenson, and Simon Dobson. From SOA to pervasive service ecosystems: An approach based on Semantic Web technologies. In Javier Cubo and Guadalupe Ortiz, editors, *Adaptive Web Services*

for Modular and Reusable Software Development: Tactics and Solution, chapter 8, pages 207–237. IGI Global, 2012

- Graeme Stevenson, Jose Luis Fernandez-Marquez, Sara Montagna, Alberto Rosi, Juan Ye, Giovanna Di Marzo Serugendo, Mirko Viroli, Simon Dobson, and Akla Esso Tchao. Towards situated awareness in urban networks: A bio-inspired approach. In José Luis Fernandez-Marquez, Sara Montagna, Andrea Omicini, and Franco Zambonelli, editors, *1st International Workshop on Adaptive Service Ecosystems: Natural and Socially Inspired Solutions (ASENSIS 2012)*, SASO 2012, Lyon, France, September

- Graeme Stevenson, Mirko Viroli, Juan Ye, Sara Montagna, and Simon Dobson. Self-organising semantic resource discovery for pervasive systems. In José Luis Fernandez-Marquez, Sara Montagna, Andrea Omicini, and Franco Zambonelli, editors, *1st International Workshop on Adaptive Service Ecosystems: Natural and Socially Inspired Solutions (ASENSIS 2012)*, SASO 2012, Lyon, France, September

- Graeme Stevenson, Gabriella Castelli, Juan Ye, Alberto Rosi, Simon Dobson, and Franco Zambonelli. A bio-chemically inspired approach to awareness in pervasive systems. In *First International Workshop on Sensing and Big Data Mining*, SenseMine '13, November 2013

- Graeme Stevenson, Danilo Pianini, Sara Montagna, Mirko Viroli, Juan Ye, and Simon Dobson. Combining self-organisation, context-awareness and semantic reasoning: the case of resource discovery in opportunistic networks. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, Coimbra, Portugal, March 2013

- Juan Ye and Graeme Stevenson. Semantics-driven multi-user concurrent activity recognition. In Juan Carlos Augusto, Reiner Wichert, Rem Collier, David Keyson, Albert Ali Salah, and Ah-Hwee Tan, editors, *Ambient Intelligence*, volume 8309 of *Lecture Notes in Computer Science*, pages 204–219. Springer International Publishing, 2013

- Juan Ye, Graeme Stevenson, and Simon Dobson. KCAR: A knowledge-driven approach for concurrent activity recognition. *Pervasive and Mobile Computing*, (In Press) 2014

CHAPTER

ONE

INTRODUCTION

A recent survey (MMMD11) predicts that by 2016 over one billion people will own mobile devices, each embedded with powerful sensing, computing, and networking capabilities. Over twenty years ago, Mark Weiser's vision (Wei95) projected this reality to the point where such technology is found not only in phones, but myriad other objects including home furnishings and appliances, cars, clothing, coffee mugs and credit cards to name but a few. Here, such devices opportunistically and spontaneously connect with each other to form dense infrastructures, delivering the correct service to the correct person at the correct time and place by taking advantage of the sensing modalities offered by the surrounding physical and virtual world (VZSD12). This vision established the field commonly referred to as *ubiquitous or pervasive computing*.

In addition to envisioning a future where computational devices surround us, Weiser also described how such technology could enhance our surroundings *invisibly*: people become less aware of technology's presence, and more focused on the task that the technology enables. This perception shift is achieved through reducing the amount of explicit interaction between people and technology and through emphasising inputs via *implicit interaction*.

Consider, for example, a typical home central heating system, where a thermostat monitors the temperature of the environment in order that heating can be automatically regulated. Here, occupant interaction is limited to turning the heating on, off, or setting the desired temperature. Despite the small number of interactions, Weiser's vision leads us to question how even they might be made more implicit. For example, can the heating be switched on automatically when someone is at home, using motion sensors to detect their presence? Can the temperature preferences of individuals be learned to support the automatic tailoring of the environment for its occupants at any given time? Can daily schedules be learned to allow the home to be optimally and energy-efficiently heated before its occupant arrives home? Each of these possibilities affords the heating

system more autonomy, reduces the requirement for human interaction, and, ultimately, allow the technology to *fade into the background* of everyday life.

Achieving this vision is challenging in many ways. In particular, a technology can only be said to fade into the background when its behaviour matches its users' expectations. Difficulties arise when exhibited behaviour does not match a user's mental model of a system's operation. In many domains, the techniques necessary to support applications in understanding the world implicitly are not yet advanced enough to recognise what is happening, or predict what will happen, in an environment with consistently high accuracy.

Approaches to acquire information from the environment, necessary to drive application behaviour, can be found in many application domains. For example:

- In the area of *healthcare and assisted living*, strong emphasis has been placed on recognising "Activities of Daily Living" (KFM+63); a set of tasks, including personal hygiene, dressing and meal preparation, proposed as a means of judging the ability of people to care for themselves. There are many works in this area, that predominantly attempt to identify such tasks through sensors attached to significant objects in the home (TIL04; vKK07; HNS11a; CFPV12; PvdPS10; GPBB12; MKSS13);
- In *childcare*, work has attempted to capture the movements of children (NP13), and details of their interactions with sensor-embedded toys (WAS+12). This has application in a wide variety of areas, for example, implicitly assessing a child's developmental progress, or recognising "stimming" behaviours in autistic children, such that an automatic journal of the child's behaviour can be maintained (MSW+05);
- In *sport*, works have been conducted on recognising significant body poses of athletes (EBMM03; BOP97);
- In *pet care*, projects have recognised the activities and activity level of dogs (WNKL13; LHH+13) as a means of evaluating their health;
- In *travel*, research has been conducted on inferring a person's transportation mode and route of travel (PLFK03); automatically discerning and labelling places of significance (LFK07), and identifying regular movement patterns of individuals (FM11) and crowds (FRMZ11).

Far from being an exclusive list, there are numerous additional areas where infrastructural and wearable sensing platforms are being used to acquire knowledge, including

sign language gesture recognition (NN07), mobile advertising (SPJB07), manufacturing process control (SRO+08), control of video games (Zha12), person-adaptive digital signs (MEBK09), and sustainability research (FPSD09) to name a few.

As a general principle, all *context aware* (Dou04) systems and applications require good awareness of what is happening in their operating environment (be it person-, building-, or process-centric) in order to realise their behaviour effectively. This is achieved through extracting data from sensors embedded in the environment, or worn or carried by people, and using it as a basis for interpreting the state of the world. System behaviour is then driven by automatically recognising state changes over time, allowing implicit interaction to be realised.

From sensor design to application development, many challenges must be met to realise this technological vision. This thesis focuses on one particular sub-component of the process: The modelling and interpretation of sensor data. More precisely, the process by which raw sensor data is interpreted as a *situation*— a symbolic, high-level representation of environment state, such as working, running, eating, or talking, suitable for driving application behaviour (SVL01). In particular, we investigate how, by combining expert knowledge and machine learning, it is possible to develop accurate decision processes for interpreting sensor data that can be made robust to sensor noise while remaining understandable to the human eye.

Next, we outline the main challenges facing researchers working in situation recognition, after which we provide a thesis statement (Section 1.3) and overview the contributions that this thesis makes towards addressing these challenges (Section 1.4). Finally, we conclude with an overview of the structure of this thesis (Section 1.5).

1.1 A Note on Terminology

In the literature, the term *activity recognition* is often used to describe the process of identifying the physical actions of a person (e.g., walking or climbing stairs), or a task they are carrying out (e.g., preparing food or talking). McKeever (McK11) describes situation recognition as having an all-inclusive meaning: Encompassing activities, but also referring to other scenarios (e.g., weather conditions or a person's state of health). In this thesis we use this more general term and meaning. However, in many works, the two terms are used interchangeably. When referring to the literature, we use the term preferred by the papers' authors.

1.2 Situation Recognition Research Challenges

Situation recognition describes the ability of a pervasive system to discern the situation or situations occurring at a particular point in time by interpreting available knowledge. This knowledge may come from a range of sources, both physical (for example, temperature or motion sensors), and virtual (for example, calendar schedule data or web browsing history).

In this section, we overview research challenges in the field of situation recognition. The first two are central to the work described in this thesis: Capturing the structural semantics of knowledge (Section 1.2.1), and making the decision process intelligible (Section 1.2.2). We discuss these in more detail before briefly overviewing other research challenges in the field (Section 1.2.3).

1.2.1 Capturing the Structural Semantics of Knowledge

The use of multimodal sensors produces datasets with heterogeneous features, differing in format, structure, scale, and frequency. Chen et al. (CHN+12) identify that the majority of published work that involves the capture and storage of such sensor information tends to adopt the use of ad-hoc data structures.

Beyond sensor data, many pervasive applications make use of higher-level encodings, such as ontologies, to describe application level concepts, such that sensor data can be mapped to the data model and annotated with additional semantic meaning (RMCM03; CFJ04); for example, indicating that a motion sensor is mounted on a particular door, or that a temperature sensor value corresponds to the concept *warm*. Such an encoding, which provides developers with a symbolic model of their software's operating environment, captures information used to characterise the situation of entities in the environment (their *context* (DA00)) and is commonly referred to as a *context model*.

The capture of expert knowledge can contribute to making inferences that cannot be obtained directly from sensors. For example, the modelling of containment relationships between physical locations supports the inference of a person's located in a larger space when their presence is detected in a space it encloses.

This ability to generalise knowledge can contribute to situation recognition. However, ontological models in the literature are typically ad-hoc, primarily designed to meet the needs of particular applications or types of knowledge (SKDN09).

While standards for representing information about sensors and sensor data are emer-

ging (CBB+12; SHS08), it is not yet the case for these higher-level models. Without their existence, it becomes that much more difficult to share and reuse data and data models for situation recognition across applications, platforms, toolsets and researchers (SKDN09; YSD+12).

1.2.2 Making the Decision Process Intelligible

Intelligibility concerns the ease with which the decision making process of any recognition technique can be understood. This ability is particularly important when there is significant error in recognition accuracy, the cause of which needs to be diagnosed. In such cases, a lack of transparency can present a challenge.

Situation identification techniques are typically classified into two main categories: *Knowledge-based*, where inferences are drawn from rules specified by experts, and *learning-based*, where machine-learning techniques discover relations between sensor-data and situations (YDM12; CHN+12). It is generally the case that, at best, the decisions of learning-based techniques are not as intelligible as those of knowledge-based techniques, and, at worst, opaque. However, machine learning techniques usually outperform knowledge-driven techniques due to their ability to better handle noisy sensor data.

As the field of situation recognition has matured from constrained prototypes to real-world systems, the general research trend has seen a move away from knowledge-based models towards learning-based models. However, the cost of this move has been the loss of ability to scrutinise the decision making processes, which we have already identified as useful.

The challenge is to develop recognition models that harness both the power of learning to attain high accuracy, while retaining the ability to inspect and understand the decision making process.

1.2.3 Other Challenges

Situation recognition faces a number of additional challenges, that, although important, are not the focus of this work.

Unsupervised Learning As described above, situation recognition techniques depend on the availability of high quality training data. While the majority of works require that this training data be labelled with a complete record of the activities that

have taken place, there is a research trend towards the development of models that rely only on partial annotations, or no annotations as all. (HMJ+09; GCTL10; BVMR06; YS13; YSD14a; SR06; SS09). Developing these techniques to the point where their accuracy matches the levels attained when fully annotated training data is available is an ongoing challenge.

Recognition Complexity The majority of work in the literature focuses on single-subject scenarios involving situations that occur largely in sequence. However many situations involve interactions between multiple people, or situations that co-occur or are interleaved (GWW+09). Although some recognition techniques have targeted this area (MBK08; HY08; HNS11a; SZC11; YS13), significant additional research is warranted.

Automatic Error Detection and Correction Once a recognition system is set-up, its performance can degrade for many reasons, including sensor-drift or introduction of noise, sensor-malfunction, or even changes in the way a situation is realised. While some work has been published in this area (FD13), approaches to automatically correct errors or adapt the recognition process in response to drift in sensor readings remain largely unexplored.

Detecting Anomalous Situations Another challenge to be addressed is the detection of unexpected or rarely occurring situation, the nature of which makes them difficult or impossible to train for (CHN+12; YDM12), or situations with duration so short as to make their recognition difficult. Examples include health problems such as a stroke or heart attack, or an event like a home burglary.

Supporting Transfer Learning Training data is required in order to learn a situation recognition model and evaluate it. However, the process of collecting such data is often costly in terms of time and effort required to annotate it with *ground truth*—a description of the situations occurring, along with the time they occur. Consequently, an important target for research in situation recognition is the ability to learn a model in one environment, for which training data is available, and successfully apply that model to an unseen environment for which no, or little, training data exists. This task, known as *transfer learning*, alleviates the need to collect significant amounts (or ideally any) training data from the new environment, and reduces the expertise required to install and initialise the recognition system.

1.3 Thesis Statement

We have described some of the issues facing situation recognition research, focusing on the challenge of capturing sensor data and expert knowledge in a standard model that supports reasoning, sharing, and reuse, and the challenge of making decision processes intelligible to humans, despite the need for high accuracy when handling real-world, noisy sensor data.

To address these challenges, we begin by devising a formal model of concepts, relations, and logical primitives rich enough to support the mapping of sensor data and its abstraction to higher-level knowledge acquired from domain experts. This knowledge serves as the basis for developing *hybrid* situation recognition models, where intelligible, robust situation recognition models are built by harnessing machine-learning capabilities to extract patterns and deal with noise.

The hypothesis of this thesis is that:

1. A top-level ontology with associated reasoning framework can simplify domain and application model development for pervasive environments by providing a uniform modelling approach to infer and relate new knowledge from that which is specified, in a principled way;

2. The semantic constructs of the ontology model provide an expressive basis from which to learn situation recognition models that display comparable performance with more complex machine learning models, while retaining intelligibility; and,

3. The hybrid semantic model and learning-based approach can be further exploited to construct ensemble classifiers with either improved recognition accuracy, intelligibility or both;

Figure 1.1 charts the information flow between the processes to which the hypothesis relates. The left-hand-side of the figure represents the process of constructing the conceptual model of the environment (e.g., people, locations, states of interaction), a task that is manually carried out by a developer. The developer then combines this model with a set of functions to map sensor to the model concepts (Figure 1.1, top) to construct a *context model*, a representation of environment state that relates model concepts in the form of fact-like statements (Figure 1.1, centre), abstracted from the raw sensor data. For example, “Bob is in the kitchen”, or “Alice is using the computer”.

The context model plays roles both as part of the process of learning situation recognition models, and in the subsequent evaluation of the models at runtime. At development

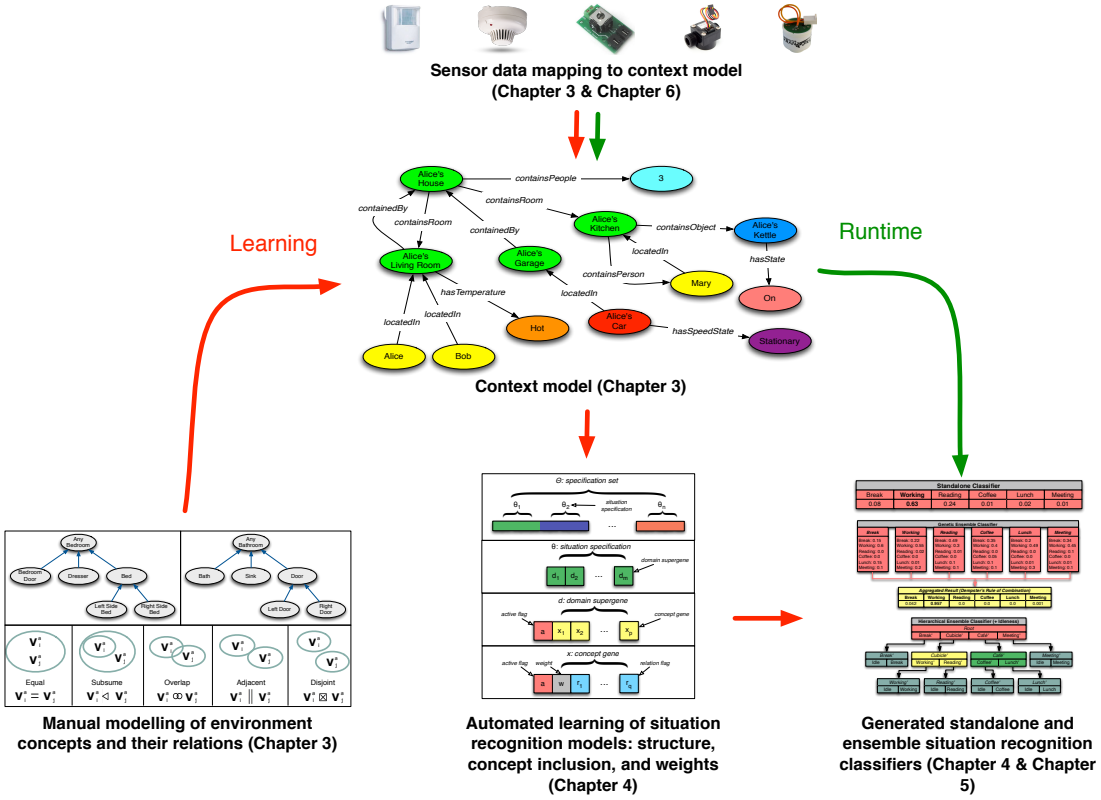


Figure 1.1: The information flow between situation recognition processes, as described in this thesis.

time, the mapping of training data into the context model, as described above, serves as input to the process of generating situation recognition models (Figure 1.1, bottom centre). This is a wholly automated process, driven by the training data and the context model, that determines which statements in the context model are relevant to identifying particular situations, and their relative importance (weights)—the developer has no part to play in their selection. The thesis explores the use of learning to generate both standalone and ensemble models as outputs of the above process. The information flow during the learning phase is indicated by the red arrows in Figure 1.1.

At runtime, the sensor data, which is mapped into the context model in exactly the same way, is evaluated directly against the learned models (Figure 1.1, bottom right) to identify the situation occurring in the environment. The information flow in this phase is indicated by the green arrows in Figure 1.1.

1.4 Thesis Contributions

This thesis presents a novel hybrid approach to sensor-based situation recognition that combines both knowledge- and learning-based techniques. This combination offers

improvements over existing techniques in each separate domain, through not sacrificing the intelligibility of the decision processes that the use of machine learning alone often implies, and through providing better recognition accuracy through robustness to noise typically unavailable when knowledge-based techniques are used in isolation.

The core contributions of this thesis are as follows:

- A review of the significant situation recognition techniques in the literature that have been applied to pervasive environments. The review summarises the findings of existing literature surveys in the area, and focuses on *i)* drawing from the taxonomies of existing surveys to suggest a categorisation for significant techniques in the field; *ii)* extending the characterisation of techniques used by existing surveys; and, *iii)* including literature not covered by existing surveys.
- An analysis of the strengths and weaknesses of Semantic-Web-based technology for modelling pervasive systems, contrasted with alternatives. Based on this, we develop a reusable top-level ontology model that provides a conceptual backbone for developing domain and application ontologies for pervasive environments. The model employs set theory to develop a uniform approach to the modelling of concepts, using a small, core set of semantic properties to relate them. We develop a context model that relates information across modelled domains, and describe how reasoning supports the propagation and transformation of information from its low level definitions to higher level concepts.
- A methodology for applying the semantic constructs of the ontology model to situation recognition, using a genetic algorithm learning-based approach to automatically construct intelligible specification models from training data. We investigate two approaches: The first, a simple model based on weighted contributions of elements of context, and the second based on a decision tree. Both approaches leverage the logical primitives of the ontology-model for inputs, with the genetic algorithm employed as a search heuristic during model construction. We also propose a novel time-sensitive sensor data representation format, beyond that discussed in the literature, to support learning.
- An investigation into two *ensemble* recognition methodologies for improving recognition accuracy and intelligibility that leverage aspects of the chosen semantic and learning approaches: the first, a *genetic ensemble*, adapts the fitness function of the genetic algorithm to generate an ensemble of classifiers that treat different situations preferentially; the second leverages model semantics to group together similar situations in the form of a *semantic hierarchy*, with each level refining the classification performed at the level above.

- A detailed walkthrough of the processes necessary to prepare a dataset for application of the learning techniques, the core aspect of which is the development of ontology concept hierarchies and the mapping of sensor data to those concepts.
- The evaluation of the recognition accuracy and intelligibility of the situation recognition techniques developed in this thesis. This includes evaluation of the standalone and ensemble models, and comparison with standard machine learning techniques and results published in the literature.

1.5 Thesis Overview

Chapter 2 surveys historically significant and state-of-the-art techniques that have been proposed in the area of situation recognition.

Chapter 3 focuses on the modelling of pervasive systems. We overview the features of a selection of modelling technologies from the literature, and discuss their relative merits. Based on this analysis, we use Semantic Web technologies to develop a reusable top-level ontology model that provides a conceptual backbone for developing domain and application ontologies for pervasive environments.

Chapter 4 concerns the automatic learning of situation recognition models from training data. The approaches described leverage the ontology model developed in the previous chapter to construct models that are robust to noise, and whose decision processes remain understandable.

Chapter 5 leverages aspects of the ontology model and learning approach to build ensemble situation recognition models, with the aim of improving recognition accuracy over a single model.

Chapter 6 describes the process of readying a dataset for application of the techniques described in this thesis.

Chapter 7 presents the evaluation of all the techniques proposed in this thesis.

Finally, Chapter 8 draws conclusions from this thesis, and outlines opportunities for future work.

CHAPTER

TWO

BACKGROUND AND RELATED WORK

Situation recognition is a broad and widely studied area. The literature review presented in this chapter focuses primarily on the topic of situation recognition within sensor-instrumented pervasive computing environments, but touching on other areas where relevant.

In Section 2.1 we begin by overviewing existing surveys published in this area, outlining their scope, contributions, and main findings. We then use this as a basis for defining our own contributions in Section 2.2, and define a taxonomy for classifying situation recognition techniques that extends from this prior work. Our taxonomy provides the structure for discussion between Section 2.3 and Section 2.13, before we summarise our findings in Section 2.14.

2.1 Existing Situation Recognition Surveys

Ye et al. (YDM12) review approaches to situation identification found in pervasive computing literature, grouping the techniques into two main categories: *specification-based* techniques that rely on manually constructed expert rules and associated reasoning engines, and *learning-based* techniques that employ machine learning and data mining techniques to classify situations from sensor data in scenarios where it is less feasible to draw on expert knowledge—a high level categorisation we follow here.

Ye et al. find that *i)* there is an opportunity for a hybrid, 'best of both worlds' approach where expert domain knowledge can be augmented by machine learning's capability to extract patterns and handle noise; *ii)* there is a requirement for reasoning to provide secondary-knowledge to applications, such as determining the set of more general situations that are occurring as a consequence of a specific calculation, or determine

the set of conflicting situations that can cannot happen; and, *iii)* there are open research questions surrounding how to detect interleaved situations, situations with multiple subjects, and rarely occurring, unpredictable situations that are difficult to train for, such as heart attacks. Ye et al. also identify a need for systems to provide insights as to how a situation is predicted so that users can form a mental model of how decisions are made.

A number of the findings from surveys on vision-based human activity recognition systems are also relevant (MHK06; FAI+05; TCSU08; Pop10; AR11; WRB11; VA13): *i)* a major challenge for real world systems is robustness in the face of dynamic conditions (noisy sensor data and missing or incomplete information) (TCSU08; AR11; VA13; Pop10); *ii)* techniques need to be robust to variances in how actions are performed (TCSU08); *iii)* it is difficult and time consuming to capture and label training data (VA13) especially in large datasets (Pop10); *iv)* infrequent events are difficult to train for (VA13) (consistent with the finding of Ye et al. (YDM12)); *v)* there is potential bias of algorithms to particular datasets, making findings less generally applicable (Pop10). Standardised, public testbeds are required to compare algorithms and assess progress (TCSU08; WRB11); *vi)* techniques are still limited in the presence of unknown actions; scenes containing multiple persons; and interactions between multiple people (WRB11); and, *vii)* recognition could be improved through integration with other input modalities (TCSU08).

The findings of Chen et al. (CHN+12) focus on emerging trends, namely that: *i)* research into recognising interleaved or concurrent activities and group activities is still in its infancy; *ii)* solutions generally suffer from a lack of interoperability and scalability; it is difficult to generalise solutions to real world use cases; *iii)* the challenge of detecting abnormal activities is still to be addressed, *iv)* difficulties in installing a sensing infrastructure in existing buildings has led to the development of techniques to sensorise the "core infrastructure" of a building: water and waste pipes, heating vents, gas meters, and electrical circuits, with the aim of capturing state changes from which activities can be inferred; *v)* there is a trend towards using ad-hoc data structures for storing sensor information, and a consequent need to develop sensor data models that promote support sharing and reuse.

2.2 Review Structure and Contributions

In this chapter we build upon the existing surveys in the area of situation recognition, through three main contributions:

1. Drawing from the taxonomies of existing surveys, we suggest a categorisation for all significant techniques in the field.

2. We extend the characterisation of existing techniques in terms of implementation effort, scenario complexity, intelligibility, evaluation approach, and knowledge transfer ability. The goal of this analysis is to better identify the distinguishing features of existing techniques and help researchers identify new opportunities for combinations of features yet to be explored.

3. Finally, as sensor-driven situation recognition is an active research field, we provide an updated survey that includes literature not covered by existing surveys.

We structure discussion around our chosen taxonomy: Six core features are selected from discussion topics in the above surveys, which we extend with implementation effort, scenario complexity, intelligibility, evaluation approach, and knowledge transfer ability, a total of eleven, as we focus on key research topics across the breadth of situation recognition research.

Table 2.2 classifies the features of the situation recognition techniques discussed throughout this chapter with respect to each of these categories. The table key is provided in Table 2.1.

We discuss these features in detail in the following sections. ***Environments, situations, and application areas*** discusses the scenarios and phenomena to which situation recognition has been applied. ***Sensors and sensor data*** overviews the different types of single and multi-modal sensor platforms that have been used to observe the world. ***Data segmentation*** discusses techniques to prepare data before situation recognition techniques are applied. ***Supervised and unsupervised learning*** describes the different training requirements of techniques. ***Scenario complexity*** categorises levels of complexity in recognising situations in the real world. ***Recognition techniques*** overviews state-of-the-art machine-learning, knowledge-based, and hybrid situation recognition techniques. ***Intelligibility*** discusses the ability to scrutinise the decision making process of different techniques. ***Engineering effort*** covers the extent of expert knowledge required to setup and use a technique in an environment. ***Uncertainty and noise*** classifies the ability of techniques to operate under the challenge of real-world conditions. ***Evaluation approach*** considers how techniques are evaluated, and, finally, ***Knowledge transfer*** discusses techniques how models learned in one environment can be applied to another.

Feature	Description
Paper	The citation corresponding to the table entry.
Cmp.	The level of complexity inherent in the reasoning tasks: Single subject (SS), Multiple subject (MS), Multiple subject collaborative (MC), or Multiple subject independent (MI). See Section 2.7 for a detailed description of each.
Sensors	The types of sensors employed as part of the research. For example, sensors attached to objects (obj-att), environmental sensors (env. sensors), or worn sensors like accelerometers (worn acc.) or microphones (worn mic.).
Situations	The types of situations the work attempts to recognise. For example, rooms or outdoor locations, household or office activities, or Activities of Daily Living (ADL).
Techniques	The recognition technique applied: For example, Bayesian Network (BN), Support Vector Machine (SVM), Hidden Markov Model variants (*HMM), Conditional Random Field variants (*CRF), Naive Bayes (NB), Decision Tree, Nearest Neighbour (NN), Neural Network, or Dempster Shafer (DS). See Section 2.8 for further details and descriptions.
Learning	The general class of machine learning technique (Supervised (S), Unsupervised (U), or Semi-Supervised (SS)). Techniques that do not use machine learning are marked N/A.
Effort	A rough categorisation of the engineering effort required to use the technique. Either Feature Extraction (FE) or more complex Model Construction (MC).
Intelligible?	Whether the decision processes of the described technique are "white box" or designed to be understood by developers or end users.
Noise/Unc.	Whether or not the technique is designed to explicitly handle noise or uncertainty in data.
Eval. Approach:	The approach used to evaluate the work. Either a custom dataset collected for the purpose (CD), a simulation, publicly available datasets (denoted by reference to the source), or no evaluation (No).
Segmentation	The segmentation technique used to prepare the data for evaluation. The various named techniques are described in Section 2.5.
Transfer	Whether or not the work considers knowledge transfer. Transfer across homes, test subjects and physical position (pos.) of worn sensors are considered.

Table 2.1: The key for the summary table of related work shown in Table 2.2.

Table 2.2: A summary of the surveyed situation recognition techniques.

Paper	Cmp.	Sensors	Situations	Technique	Learning	Effort	Intelligible?	Noise/Unc.	Eval. Approach	Segmentation	Transfer
(CCKM01)	SS	RSSI	Room location	BN	S	FE	No	Yes	CD	N/A	No
(RAMC04)	SS	Obj-att	N/A	BN	N/A	MC	No	Yes	No	N/A	No
(DP04)	N/A	N/A	N/A	BN	N/A	MC	No	Yes	No	N/A	No
(GPZ05)	N/A	N/A	N/A	BN	N/A	MC	No	Yes	No	N/A	No
(TLL05b)	N/A	N/A	N/A	BN	N/A	MC	No	Yes	No	N/A	No
(AE04)	SS	Simulated	Outdoor locations	BN	S	MC	No	Yes	Simulation	N/A	No
(PLFK03)	SS	GPS, bus routes	Transport mode, route	BN	U	MC	No	Yes	CD	N/A	No
(ZHY09)	SS	Obj-att	Household	SVM	S	FE	No	Yes	CD	static	Home
(ADF07)	SS	Motion, Obj-att	Household	BN	S	See Remarks	No	Yes	(ILB$^+$05)	static	No
(CP99)	SS	Worn mic/camera	Location based	HMM	U	FE	No	Yes	CD	static	No
(MSW$^+$05)	SS	Worn acc./mic	Woodwork assembly script	HMM	S	MC	No	Yes	CD	static	Subject
(LCB06)	SS	Worn acc.	Basic physical activities	HMM	S	FE	No	Yes	CD	static	Sub., pos.
(vKNEK08)	SS	Obj-att	Household	HMM/CRF	S	FE	No	Yes	CD	static	No
(RAMC04)	SS	Obj-att	N/A	Fuzzy First-Order Logic	N/A	MC	Yes	Yes	No	N/A	No
(SCSE09)	SS	Motion, Obj-att	ADL	HMM	S	FE	No	Yes	CD	sliding	No
(HRLL08)	SS	Motion, Obj-att	ADL	HMM	S	MC	No	Yes	(ILB$^+$05)	static	No
(WNS06)	MI	Audio, video	Office activities	HHMM	S	MC	No	Yes	CD	static	No
(vKEK10b)	SS	Motion, Obj-att	Kitchen and bathroom	SHMM	S	MC	No	Yes	CD	static	Home
(DBPV05)	SS	Camera tracking	Kitchen	Switching-HMM	S	MC	No	Yes	CD	stream	No
(MBK08)	SS	Simulated profiles	Kitchen	Interleaved-HMM	S	MC	No	Yes	Simulation	unknown	No
(KZSM12)	SS	Motion, Obj-att	Household	MMM	S	MC	No	Yes	(TIL04), (vKEK10a)	stream	No
(BOP97)	SS	Video	Martial art gestures	CHMM	S	MC	No	Yes	CD	static	No
(NN07)	MI	Simulated/vision	Body motion, sign language	CSHMM	S	MC	No	Yes	Sim., subset of (VM03)	stream	No
(WGT$^+$11)	MI	Obj-att	Household	CHMM	S	MC	No	Yes	CD	static	No
(PvdPS10)	SS	mic/cam	Household	HMM	U	FE	No	Yes	CD	stream	No
(KMK$^+$03)	SS	device env. sensors	Environment, noise source	NB	S	FE	No	Yes	CD	static	No
(MBSM04)	SS	device env. sensors	User activity, availability	NB	S	FE	No	Yes	CD	static	No
(TIL04)	SS	Obj-att	ADL	NB	S	FE	No	Yes	CD	dynamic	No
(VVL07)	MC	Positions, velocities	Robot Tag Activities	CRF	S	FE	No	Yes	CD	static	No
(LFK07)	SS	GPS, street-maps	Significant places visited	CRF	U	FE	No	Yes	CD	spatially	No
(NDHC10)	SS	Motion, temp	Household	CRF	S	FE	No	Yes	CD	static	No
(WLYjH07)	SS	Motion, Obj-att	Household	FCRF	S	MC	No	Yes	(ILB$^+$05)	static	No
(HY08)	SS	Various	Various	Skip-Chain CRF	S	MC	No	Yes	(ILB$^+$05), (PFKP05), (CY05)	static	No
(BI04)	SS	Worn acc.	Body-motion-based	Decision Tree	S	FE	Yes	Yes	CD	static	Subject
(MCLC04)	SS	Worn acc.	Body motions	Decision Tree	S	FE	Yes	Yes	CD	static	No
(KNM$^+$06)	SS	Worn acc.	Body motions	Decision Tree	S	FE	Yes	Yes	CD	static	No
(BSLT05)	SS	Worn env. sensors	Office and Household	Decision Tree	S	FE	Yes	Yes	CD	static	No
(LHP$^+$07)	MC	Obj-att, RFID	Household	Decision Tree	S	FE	Yes	Yes	CD	sliding	No
(MRSS06b)	SS	Worn env. sensors	Significant locations	Decision Tree	S	FE	Yes	Yes	CD	static	No
(MYK$^+$10)	SS	Worn env. sensors	ADL	HMM	S	FE	Yes	Yes	CD	static	No
(MKSS13)	SS	Worn magnetic sensor	ADL	HMM	S	FE	Yes	Yes	CD	static	No
(LPLP12)	SS	Worn acc., vital signs	Body motion-based activities	Decision Tree	S	FE	Yes	Yes	CD	static	No
(EBMM03)	SS	Vision-based	Sporting movements	NN	S	FE	No	Yes	CD	static	No
(HGKZ07)	N/A	N/A	N/A	NN	S	FE	No	Yes	CD	static	No

Table 2.2: A summary of the surveyed situation recognition techniques.

Paper	Cmp.	Sensors	Situations	Technique	Learning	Effort	Intelligible?	Noise/Unc.	Eval. Approach	Segmentation	Transfer
(VLKS08)	SS	Env. Sensors	Indoor and outdoor activities	NN	S	FE	No	Yes	CD	static	No
(LHH+13)	SS	Worn acc.	Dog activities	NN	S	FE	No	Yes	CD	static	Subject
(CGF+10)	N/A	Gas flow	Active appliances	NN	S	FE	No	Yes	CD	sliding	No
(NP13)	SS	Worn acc., barometric	Body motions	NN	S	FE	No	Yes	CD	static	No
(BCR09)	MC	3D tracking	Office activities	SVM	U	FE	No	Yes	CD	static	No
(CYW05)	SS	Simulated sensors	Body motions	SVM	S	FE	No	Yes	CD	static	No
(RDML05)	SS	Worn acc.	Body motions	SVM	S	FE	No	Yes	CD	static	Subject
(HBS07)	SS	Worn acc.	ADL	SVM	S	FE	No	Yes	CD	sliding	No
(KGS+08)	SS	Laser rangers	Body motion	SVM	S	FE	No	Yes	CD	static	Subject
(PRK+07)	N/A	Electricity meter	Active appliances	SVM	S	FE	No	Yes	CD	sliding	No
(SS09)	SS	Obj-att, RFID	Household	SVM	SS	FE	No	Yes	(LHP+07)	sliding	No
(GPBB12)	SS	Worn acc., proximity	ADL	SVM	S	FE	No	Yes	CD	sliding	No
(KC14)	SS	Obj-att	ADL	SVM	S	FE	No	Yes	CD	multiple	No
(YT12)	SS	Worn mic.	Sound-based activities	SVM	S	FE	No	Yes	CD	static	Subject
(WAS+12)	SS	Toy-based env. Sensors	Interactions with toys	SVM	S	FE	No	Yes	CD	sliding	Subject
(YWC08)	SS	Worn acc.	Body-motion-based	Neural Network	S	FE	No	Yes	CD	sliding	Subject
(KTH+10)	SS	Worn acc.	Body-motion-based	Neural Network	S	FE	No	Yes	CD	sliding	Subject
(HMJ+09)	SS	Vision-based	House, Loading Dock	Key Object Sequence	U	MC	No	Unknown	CD	stream	No
(YD10)	SS	Obj-att, RFID	Household	Context Lattice	S	MC	Yes	No	(LHP+07), (vKNEK08)	static	No
(RCHSE11)	SS	Obj-att	ADL	Cluster, HMM	U	FE	No	Yes	CD	sliding	No
(FRMZ11)	SS	GPS, street-maps	Location based schedules	Topic Model	U	FE	No	Unknown	CD	location-time	No
(FRMZ11)	MC	Social media updates	Significant locations	Topic Model	U	FE	No	Unknown	CD	location-time	No
(FGP11)	SS	GPS, street-maps	Location based schedules	Topic Model	U	FE	No	Unknown	CD	location-time	No
(CWP12)	SS	Various	ADL	Patterns, LDA	U	FE	No	Yes	See (CWP12)	stream	No
(CFPV12)	SS	Motion, audio, Obj-att	ADL	MLN	S	MC	Partial	Yes	CD	sliding	No
(HNS11a)	SS	RFID	ADL	MLN	S	MC	Partial	Yes	(PFKP05)	sliding	No
(GCTL10)	SS	Obj-att	ADL	Emerging Patterns	U	MC	No	Yes	CD	stream	No
(Lok04b)	N/A	N/A	N/A	First-Order Logic	N/A	MC	Yes	No	N/A	N/A	No
(HI06)	N/A	N/A	N/A	First-Order Logic	N/A	MC	Yes	No	N/A	N/A	No
(BGB06)	N/A	N/A	Household	Action-Description	N/A	MC	Yes	No	N/A	N/A	No
(CNM+08)	N/A	N/A	Household	Event Calculus	N/A	MC	Yes	No	N/A	N/A	No
(DHKZG08)	SS	ECG biosensor	Blood pressure states	Fuzzy Logic	N/A	MC	Yes	Yes	CD	static	No
(WSSY02)	MC	Audio, video	Focus of attention	DS	N/A	MC	Unknown	Yes	CD	static	No
(HNM+09)	SS	Obj-att	Making drinks	DS	N/A	MC	Yes	Yes	N/A	N/A	No
(ZCZG09)	SS	Obj-att	Household	DS	N/A	MC	Unknown	Yes	MIT House Dataset	unknown	No
(MYC+10)	SS	Obj-att	Office and household	DS	N/A	MC	Yes	Yes	CD and (vKNEK08)	static	No
(PG11)	N/A	N/A	Health related	Ontology+Rules	N/A	MC	Yes	No	N/A	N/A	No
(TMKL06)	N/A	N/A	N/A	Ontology+Rules	N/A	MC	Yes	No	N/A	N/A	No
(RB11b)	SS	Simulated env sensors.	Home and workplace	Ontology	N/A	MC	Yes	No	Simulation	N/A	No
(MDEK13)	N/A	N/A	N/A	Ontology+SPARQL	N/A	MC	Yes	No	N/A	N/A	No
(ANH07)	SS	Device env.	Indoor and outdoor	Ontology+Fuzzy Logic	N/A	MC	Yes	Yes	CD	N/A	No
(RB11a)	SS	GP, worn acc.	Body-motion-based	Ontology+MRE	S	MC	No	Yes	CD	static	No
(YS13)	MI	Obj-att	Household	Ontology+PMK	U	MC	No	Yes	(CSE09)	stream	No
(YSD14a)	SS	Obj-att	Household	Ontology+Clustering	U	MC	No	Yes	(ILB+05), and (vKEK10b)	stream	No

2.3 Environments, Situations, and Application Areas

Situation recognition has been applied in both indoor and outdoor settings to a range of recognition areas: from capturing physical phenomena, such as posture, gestures and activities, such as standing, sitting, and walking (LCK+05; LCB06; MSW+05), speaking and singing (YT12), to recognising more complex situations of cooking, cleaning, eating and bathing (TIL04; vKNEK08), and interactions with other individuals (CYW05). Several of these studies, for example, (PvdPS10; GPBB12; MKSS13), focus on Activities of Daily Living (ADL) (KFM+63); designed to serve as a measure of judging the ability of elderly patients to care for themselves. Other studies have focused on recognising travel schedules (PLFK03), the daily routines of individuals (FGP11; FM11), and movement patterns of groups (FRMZ11).

Although situation recognition studies have focused to a large degree on recognising the activities of young and middle-aged adults, studies have also been carried out with specific focus on the activities of the elderly (CYW05; vKK07), babies (NP13), young children (WAS+12), and even dogs (WNKL13; LHH+13). As discussed in the introduction to this thesis, these techniques have application in a vast range of application areas including: healthcare and assisted living, personal fitness, child care, pet care, sport, and travel.

2.4 Sensors and Sensor Data

Different situations lend themselves to recognition by different types of sensors. Physical activities whose interpretation is closely tied to specific motions, can often be identified through video analysis (BOP97; CP99; NN07) or simple accelerometer data (BI04; MCLC04; RDML05; LCB06; KNM+06). More complex recognition is achieved though augmenting such sensing platforms with additional multimodal sensors such as a microphone (KMK+03; MSW+05; WNS06; PvdPS10), light sensor (MYK+10), magnetic sensor (MKSS13), capacitive proximity sensor (GPBB12), or vital sign sensors (LPLP12).

While wearable, or phone-based, sensing platforms are necessary for outdoor situation recognition, indoor solutions often involve instrumenting everyday objects such as cookers, fridges, beds, computers, and toilets with RFID or state change sensors (TIL04; vKK07; MYC+10) as a means of capturing information describing human-object interactions, and infra-red sensors as a means of capturing human mo-

tion (vKEK10a). To lessen instrumentation requirements, work has also been conducted on single-point sensing solutions, attached to, for example, an electricity (PRK+07) or gas meter (CGF+10), with recognition applied to infer appliance use throughout the home.

While the raw output of sensors can often be fed directly as input to a recognition technique, it is often useful to first pre-process data to generate features useful for the recognition process. For example, to generate summary statistics, such as the mean and variance of a signal (CP99; MSW+05), to convert continuous data into categorical data through a strict or fuzzy mapping (HNM+09; MYC+10), to retain information about earlier state transitions (vKNEK08), or to apply a decay function to sensor data that is infrequently asserted (MYC+10; KC14).

2.5 Data Segmentation

After sensor data has been extracted, analysed, and features selected for the recognition process, the next pre-processing step is to translate the training data, usually in the form of a log of continuous sensor readings, into a set of instances that are suitable for learning.

This process, called segmentation, is achieved by partitioning the data log according to some criteria, for which the literature describes a number of strategies. By far the most prevalent are *static (and sliding) window techniques*, which involve splitting data into (overlapping) segments of equal temporal length (typically 30 or 60 seconds) (BVMR06; vKNEK08; YD10), recording the most recent or average sensor value for each sensor during that period. Another strategy involves generating segments containing a fixed number of sensor events (PvdPS10; KC14).

Fixed-window approaches are not suited to all environments. A window may bridge the gap between two situations (thus capturing data from both), or because in some long lived activities (such as sleeping) there may be a long duration between sensors firing (YSD14b). Here, *dynamic window techniques* offer an alternative approach. The technique involves using windows of variable size based on some criteria. For example, the average duration of activities (TIL04; MYC+10), a change in the location associated with consecutively firing sensors (HN09), or the semantic similarity of sensor events (YS13; YSD14a).

2.6 Supervised and Unsupervised Learning

The accuracy of a learned situation recognition model depends highly on the quality and quantity of training data. There are two main approaches to training such models: supervised and unsupervised. Supervised approaches require training data to be collected and labelled with *ground truth* – the situations they represent – before a model can be constructed, while unsupervised approaches attempt to learn a model from data without such labelling; typically by clustering together similar training instances according to some criteria.

Supervised recognition, whether driven through the collection of annotations using a diary (MYC+10), spoken annotations (vKNEK08), or through post-activity analysis (CYW05) are most common. However, unsupervised approaches (HMJ+09; GCTL10; BVMR06; YS13) are increasingly prevalent.

Supervised approaches can be limited by the cost to collect and accurately annotate training data, while unsupervised approaches are typically less accurate and can generates models that do not correspond to the desired set of situations to be recognised. A third class of learning, *semi-supervision*, describes approaches capable of leveraging the presence of some labelled instances within a predominantly unlabelled collection. This, largely unexplored, approach in the context of situation recognition has potential due to the lower cost of asking subjects to record and label small activity samples (SR06), or through supporting recognition with fewer, less precise annotations (SS09).

2.7 Scenario Complexity

Recognition approaches in the literature are targeted towards classification problems of varying levels of complexity. The majority of work has focused on single-subject scenarios, with tasks performed in sequence. However, increasingly, work is investigating approaches to recognising activities that are interleaved, or involve multiple subjects (CHN+12). Gu et al. (GWW+09) provide a rough categorisation of approaches as falling into one of four categories:

Single subject (SS) Activities of a single subject. For example, cooking dinner, followed by taking a bath, followed by sleeping (TIL04; LCB06; vKNEK08; MYC+10).

Multiple subject (MS) Two or more subjects carry out a sequence of activities together. For example, drinking coffee or watching television (HY08; WGT+11).

Multiple subject collaborative (MC) Two or more subjects working together to complete a goal, with each subject working on a different sub-task of the activity. For example, one subject prepares food while another cooks food (GWW+09).

Multiple subject independent (MI) Two or more subjects carry out activities simultaneously, but independently. For example, one subject prepares food while another watches television (YS13; YSD14b).

In their categorisation, Gu et al. assume that the activities of individuals are strictly carried out sequentially. The complexity of each category is then increased by removing this requirement and instead allowing activities to be interleaved or to be performed concurrently. Several of the recognition approaches surveyed focus on this more complex behaviour (MBK08; HY08; HNS11a; SZC11; YS13).

In the context of multiple subject independent activities, Ye and Stevenson (YS13), propose a semantics-based approach to partition sensor data streams in such a way that each partition is treated as describing a single subject performing activities in sequence, allowing recognition to be supported by the main body of techniques developed for single subject activities.

2.8 Situation Recognition Techniques

Situation recognition surveys typically categorise techniques into two high-level categories based on whether they are machine-learning-based or knowledge-based (YDM12; CHN+12)—each subdivided into a set of general techniques and their extensions. The former grouping typically operates on raw or minimally processed data: summary statistics, or conversion from continuous to discrete valued data (CP99; MSW+05; PRK+07; HBS07; BCR09). Their core strength lies in handling noise, and their better capability to discriminate between different classes of situations than can be achieved manually. The latter category relies on the domain knowledge of experts and their ability to manually interpret data and encode the sorts of concepts and relationships that are useful to solving the situation recognition task at hand (CFJ03; RAMC04; Lok04b; MYC+10; YS13). The decision processes of knowledge driven approaches tend to be transparent to developers' eyes, however, they are often less robust to the presence of noise than alternative techniques (YD10).

The move towards a third category of techniques that hybridises aspects of both knowledge and machine-learning techniques is a recent trend emerging from the literature. Such a combination of approaches is evidenced in two ways: the use of machine

learning techniques to mine primitives for activity models (GCTL10), and the use of manually specified knowledge to augment machine-learning approaches through additional inferences (CFPV12) or restrictions (YSD14a). To date, all hybrid examples focus on enriching the knowledge available to the learning algorithm, not on aiding the inspection and interpretation of the resulting decision processes.

We overview notable techniques from the literature, across all three categories, below.

2.8.1 Learning-based Approaches

Bayesian Network A Bayesian network (Hec99; BG07) is a directed acyclic graph that represents the conditional dependencies among a collection of random variables (sensor inputs). Given instances of sensor values, the network calculates the probability that the inputs are indicative of the set of possible hypotheses (situations) under consideration. Each network node is associated with a table that describes the node's conditional probability distribution across all possible combinations of the nodes it depends on. Nodes without dependencies are associated with *a priori* probabilities. Given a set of input values describing the nodes it depends on, a node outputs the probability of the variable, hypothesis, or unknown it represents, with the classification given by the hypothesis with highest probability. Bayesian networks rely on the availability of large amounts of training data with full coverage of the different state combinations that situations can result from.

Castro et al. (CCKM01) infer a device's location through observing the signal strength of visible WiFi access points. The network describes the *a priori* distribution over a set of locations, while the hypothesis nodes describe the probability of observing a discretised RSSI value from each access point given a particular location. Ranganathan et al. (RAMC04) develop a network to capture the causal relationships between sensor observations and system state in their environment, *Gaia* (RHC+02). Abdelsalam et al. (AE04) create a habit-based location-tracking model for people, incorporating information about time, speed, routes, and weather conditions. Using GPS and knowledge of bus routes, Patterson et al. (PLFK03) devise a model to learns a subject's mode of transportation and most likely route. Albinali et al. (ADF07) report on optimising learning from sparse training data by focusing on building a feature profile of an activity from consistent aspects of the data, rather than capturing the whole activity.

Naïve Bayes A simplification of the Bayesian Network, the Naïve Bayes model, captures probabilistic relationships between sets of features and classifications under the assumption that the set of features are mutually independent (Mit97). Although this

assumption is often invalid, Naïve Bayes classifiers are more practical to implement than Bayesian Networks, having a smaller number of parameters that must be estimated and therefore require fewer data to train the classifier, although there is a risk of skewing towards the underlying dependence (WF05). Naïve Bayes classifiers have been shown experimentally to exhibit similar accuracy to Bayesian networks while being orders of magnitude faster (LD05).

Korpipaa et al. (KMK+03), construct a model from the discretised sensor output of a mobile device to detect whether a person is indoors or outdoors, and to determine the source of environmental noise (for example, a car, elevator, or a particular genre of music); significantly higher accuracy is achieved in controlled laboratory conditions than under real world conditions. Mühlenbrock et al. (MBSM04) differentiate subject activities and availability states in an office environment based on data collected from a desktop PC, phone, and mobile PDA, however the scripted evaluation artificially prevented rapid activity switching, and controlled the mapping of availability to specific times. Tapia et al. (TIL04) instrument multiple, real homes with state change sensors. Both a multi-class classifier designed to differentiate all situations, and a set of single class classifiers to enable identification of concurrent situations are investigated.

Hidden Markov Model Markov Models assume that the conditional probability distribution of future states depends only on the current state, not any that proceeded it. A Hidden Markov Model (HMM) models such processes, where only the observations are known, and the states are unobserved (Blu04). Given training data, the prior probabilities of initial states, state transitions, and observations are estimated by computing the maximum likelihood estimates for the probability parameters (BPSW70). Activity recognition then proceeds by selecting states that best explain the observations. HMMs and their variants (AY09) can be expensive in terms of memory requirements, compute time, and often require large amounts of data to train.

Clarkson et al. (CP99) recognise corse-grained situations such as being in the office or bedroom from features extracted from a body-worn microphone and camera. Minnen et al. (MSW+05) use body-worn accelerometers and microphones to uncover patterns of repeated movement that serve as primitives towards detecting higher level activities. Lester et al. (LCK+05; LCB06) perform gesture recognition (for example, walking, sitting, going down stairs) from body-worn sensors. van Kasteren et al. (vKNEK08; KEK11) use a HMM for recognising household situations from object-attached sensors in a home setting. Singla et al. (SCSE09) study the recognition of interleaved *Activities of Daily Living* that can help identify individuals who may have trouble functioning independently at home (KFM+63). Hasan et al. (HRLL08) design a HMM for each

possible situation, wherein sensors are treated as states, and state transition probabilities correspond to the conditional probability of one sensor being activated after another.

Wojek et al. (WNS06) use a Hierarchical-HMM (FST98), where each state may be represented as a (Hierarchical) Hidden Markov Model, to improve recognition of office situations from audio and video features. van Kasteren et al. (vKEK10b) improve on the recognition accuracy of a HMM in the settings of a Kitchen and Bathroom using a Semi-HMM (Yu10), where state transition probabilities additionally depend on the time elapsed since the current state was entered. Duong et al. (DBPV05) combine both approaches into a Switching-HMM tested on recognising kitchen activities. Modayil et al. (MBK08) use an Interleaved-HMM, which records the last object observed for each activity, to improve the recognition of interleaved activities. Kalra et al. (KZSM12) use a two-stage Multi-Markov model; in the first stage, a sensor data stream is evaluated against a Markov Chain for each possible activity. The most probable is selected for the second stage, where a HMM modelling the temporal relations between activities boosts recognition accuracy (by 2%-3%) by acting as an error-correction layer on top of the first stage. Finally, Pijl et al. (PvdPS10) find the standard Baum-Welch trained HMM to outperform a number of variant models in recognising ADL from event sequences generated from a single wall-mounted camera and microphone.

Conditional Random Fields Conditional Random Fields (CRF) resemble HMMs in that they model time steps, each with a hidden and observable variable. However, unlike HMMs, CRFs do not operate under the assumption that observations are independent and instead model the conditional probability of state sequence instead of the joint probability of the states and observations. Thus, their strength is in encoding temporal relationships between states (LMP01).

Vail et al. (VVL07) find CRFs outperform HMMs for simple activity recognition in simulated robot tag. Liao (LFK07) build a CRF model to capture the relationship between GPS traces and a street map, and, from there, activity sequences. This is used to drive the identification and labelling of significant places that a subject visits with over 90% accuracy. Nazerfard et al. (NDHC10) investigate the use of CRFs for activity recognition in a multi-inhabitant smart-home, finding an improvement over HMMs for most activities, although recognition is not person-specific.

As with HMMs, several CRF extensions have been proposed. Wu et al. (WLYjH07) use Factorial-CRFs to support the recognition of multiple concurrent situations, where co-temporal connections are introduced to avoid the increase in complexity resulting from modelling each combination of situations as a new state. van Kasteren et al (vKEK10b) demonstrate that Semi-Markov-CRFs, where state duration is explicitly modelled,

do not provide an increase in recognition accuracy. Hu and Yang (HY08) use Skip-Chain-CRFs to support the identification of concurrent and interleaved situations. The approach leverages long-distance dependencies in observed data in that, given a newly observed input, the model calculates the probability of the observation corresponding to a new situation.

Decision Trees Decision trees take the form of a tree structure, where each node represents a decision taken with respect to the value of a single feature, the outcome determining which branch should be followed. Evaluation of a decision tree begins from a single root node, and continues until a leaf node – representing a classification – is reached (WF05). A decision tree is constructed by recursively selecting a single feature with which to partition the training data. Each split reduces the training data along each path until the point where the remaining data shares a classification. At each step, the attribute providing the most *information gain* – that is, that attribute that most effectively partitions the data – is chosen for the decision.

Bao et al. (BI04), Mathie et al. (MCLC04), Karantonis et al. (KNM+06), Maurer et al. (MRSS06a; MRSS06b), and Logan et al. (LHP+07) find decision trees outperform a suite of alternatives for classifying subject activities from whole-body- and waist-worn accelerometers, wrist worn sensor platforms, and smart-space sensors. Bharatula et al. (BSLT05) also find a decision tree outperforms other algorithms when considering the tradeoff between energy consumption and recognition accuracy on a low-power wearable device, while Lara et al. (LPLP12) augment wearable accelerometer data with vital sign data (heart and respiration rate, breath and ECG amplitude, skin temperature, and posture), improving the recognition of five activities by 3% using an Additive Logistic Regression algorithm (FHT98). Maekawa et al. (MYK+10) collect data from a suite of wrist mounted sensors (including audio, video, accelerometer, and light) and later hand-worn magnetic sensors (MKSS13), to classify a range of activities whose recognition traditionally involves object-attached sensors. Evaluation of several approaches found that classifying raw sensor data individually using a decision tree and then processing the results using a HMM performed best.

Nearest Neighbour Nearest Neighbour algorithms classify unseen data based on its similarity to known training examples. In the standard *k*-Nearest Neighbour (kNN) approach, the *k* training examples whose features are closest to the unclassified data are selected, with a classification assigned based on a majority vote of the selected neighbouring examples (AKA91). Closeness between data points is defined by a heuristic chosen for the classification task: Euclidian and Hamming distance, Jaccard similarity,

and Tanimoto coefficient are typical examples. Nearest Neighbour algorithms can be limited by the computational requirement to search through all training data to classify a single data instance (although this can be optimised (FBF77)), and the challenge in selecting suitable distance metrics and weights to define closeness - especially for high dimensional, category-structured data. Missing or noisy data can have a great impact on nearest-neighbour-based approaches, leading to an incorrect identification of neighbours and consequent misclassification.

Delir Haghighi et al. (HGKZ07) employ kNN to perform situation recognition as part of an outline architecture for processing data streams on resource-constrained devices. Van Laerhoven et al. (VLKS08) sense motion, light, and temperature data from a wrist-worn platform 24 hours a day. Habitual information drawn from the diary data is used to improve recognition in cases where the recognition step does not provide a conclusive result. Cohn et al. (CGF+10) describe a single point sensing approach for inferring the source of gas use, for example, a water heater, pool heater, or fireplace. The audio-based approach uses a microphone to sense the resonance frequency of the gas regulator, and attains 95% accuracy for appliance detection. Nam et al. (NP13) investigate activity recognition for child care, using a waist mounted accelerometer and barometric sensor to detect 11 situations including crawling, climbing, and walking; kNN outperformed a selection of other classification techniques with 96% accuracy. Ladah et al. (LHH+13) investigated activity recognition in dogs by applying KNN to a feature set extracted from a collar worn triaxial accelerometer.

Support Vector Machine Given a set of training examples, Support Vector Machines (SVM) address the challenge of automatically learning the boundaries, or *hyperplanes*, that optimally separate multi-dimensional data points. The term *support vector* refers to the test data instances that, if discarded, would alter the position of the dividing hyperplane. The learning process seeks to optimise the placing of the hyperplane such that it maximises the margin between itself and the support vectors. Where data cannot be separated by a linear function, it is mapped to higher dimensional plane until it is possible to gain linear separation (WF05).

Brdiczka et al. (BCR09) sense a subject's position, velocity, and distance to a table to define a subject's role in an interaction. Combined with further inputs from wall mounted microphone array and subject headset data, situations are discovered and manually labelled by experts (BVMR06). Chen et al (CYW05) simulate yet-to-be-realised sensors likely capable of recognising individual human actions, through an analysis of video and audio footage taken at an elderly care facility. Decision trees are shown to have the highest accuracy under perfect sensing conditions, with SVMs the

most resistant to noise. Ravi et al. (RDML05) and Huhny et al. (HBS07) show SVMs outperform other techniques in classifying activities from body-worn accelerometers. Patel et al. (PRK+07) use a SVM with an electrical noise sensor to classify appliance use from electrical events, where the evaluation shows only a small reduction in recognition accuracy over a six week period, even with a small number of training instances. Stikic and Schiele (SS09) use a modified SVM to support multi-instance learning (ATH03), in which "bags" of training instances are annotated based on whether or not at least one instance of the situation is present in the bag. The goal of the work is to reduce the effort involved in labelling data, and thus supervised recognition as a whole. Zheng et al. (ZHY09) bridge between activities in two domains by using web search to learn confidence weights via a similarity function, which serves as input to a weighted SVM model. Grosse-Puppendahl et al. (GPBB12) enhance wrist-worn accelerometer based detection with a capacitive proximity sensor to indirectly measure the distance and nature of an object within reach, improving the recognition of nine ADLs between 2.4% to 10.7%. Yatani et al. (YT12) describe an acoustic sensor that captures audio features from the subject's throat, with performance in the wild within 10% of that under laboratory conditions. Finally, Westyn et al. (WAS+12) present the detailed design of a collection of smart toys used to characterise the ways in which a child plays. The toys are embedded with a sensor platform that detects motion, sound, and touch, with SVM-based classification found to outperform other techniques.

Other Notable Approaches Yang et al. (YWC08) construct a neural network with inputs from sensor data acquired from a triaxial accelerometer module mounted on a subject's dominant wrist. Summary statistics extracted from the accelerometer data serve as inputs to neural network classifiers, with 95% accuracy for a set of 8 activities. Krassnig et al. (KTH+10) adopt the same approach, showing that gender specific classifiers achieve a higher accuracy than gender non-specific.

Ye and Dobson (YD10) propose a *context lattice* as a formal structure to represent the relationships between semantics implied by low level sensor data and situations. Each lattice node represents a single or compound expression formed from a logical conjunction of context expressions. For any two nodes, there exists a third that represents the greatest logical expression that is entailed by the logical expression of these nodes. The learning process labels each node with the frequency its expression is satisfied by the sensor data and the number of times each situation occurred when the node was active, allowing the probability that an activity is occurring when this node is active to be calculated.

Rashidi et al. (RCHSE11) develop a clustering algorithm to identify sensor event

sequences that appear frequently enough to suggest that they represent an activity that should be tracked and analysed. The algorithm accounts for the fact that events are discontinuous and have varied orders. One limitation is that the designer must specify the number of activities to cluster and model; this presents difficulties in uncontrolled environments.

Ferrari et al. (FM11) and Gatica-Perez (FGP11) use probabilistic Topic Maps to extract commonly recurring patterns of temporal movements of individuals, and urban crowds (FRMZ11) based on location traces and information extracted from social network data respectively. Significant locations are discovered, with transitions between them labeled with the time of day they occur. Latent Dirichlet Allocation (BNJ03) is then applied to cluster the set of labels according to the patterns (or topics) they contain. Chikhaoui et al. (CWP12) apply LDA to activity recognition in a sensor instrumented home. Recognition is presented as a maximum likelihood optimisation problem in which sequences of sensor events are modelled as probability distributions over activities, and activities are, in turn, modelled as probability distributions over sequential patterns.

2.8.2 Knowledge-based Approaches

With these techniques, expert knowledge is employed to model the relationships between sensors and situations, with reasoning applied to the models to interpret sensor data at runtime.

Logic-based Approaches All logic based approaches break down the challenge of situation recognition into two distinct part parts: *i)* modelling the application, environment, scenario, or sensing domain in terms of entities and their relations, and *ii)* combining elements of the model in a modular structure using a logical formalism to describe situations of interest, often in the form of rules and axioms to be interpreted by a reasoner.

Gu et al. (GWPZ04) use propositional logic to support the inference of high-level context information and situations, while Ranganathan (RAMC04) et al., Loke (Lok04b), and Henricksen et al. (HI06) all adopt first-order logic descriptions to support inference on knowledge. Loke (Lok04a) develops a Prolog-like language called LogicCAP around his model that supports operations such as determining the set of situations a subject participates in and determining the most likely situation a subject is engaging in using abductive reasoning. Bouchard et al. (BGB06) propose an approach to identify everyday activities of a smart home inhabitant using plan recognition (Car01), applying

lattice theory and Action-Description logic (Bou05) to identify on-going situations from observing actions and mapping them to the lattice structure. Chen et al. (CNM+08) develop a logic based approach to situation recognition based on Event Calculus (KS86). In Event Calculus, *fluents* describe values that can change over time (triggered by *events*), while *predicates* specify the values of fluents that hold at particular points in time. In Chen's model, events correspond to sensor firings, while fluents describe object state and relations that hold between objects. The strength of this approach is the ability to deduce what will happen if a set of events were to occur, explain how a particular state of the system was arrived at, or suggest actions that will allow a state to be reached. However, extensive expert knowledge is required to model the environment and possible behaviours.

Fuzzy logic provides an approach for reasoning over data that is approximate rather than exact (Zad73). In contrast with two or three valued logic theories, where something can be true, false or unknown, fuzzy logic variables have an associated *degree of truth*, that takes a value between 0 and 1, representing vagueness in knowledge. This value can have different interpretations: A value classification where the boundaries of a concept are hard to define crisply, for example, the temporal concept of *lunch time*; the extent to which it is believed a value truly reflects real world state, for example, a person being located in a particular room; or, a means of expressing a relation between two concepts that cannot be captured in a precise way, for example, a person's preferences for different modes of transport.

Delir Haghighi et al. (DHKZG08) propose a model for fuzzy situation inference based on mapping sensor values to linguistic terms, each associated with a fuzzy membership value based on known sensing inaccuracies, and further weight the contribution of multiple fuzzy terms to calculate the confidence in a situation occurring. Many fuzzy-logic based approaches have been applied in the context of ontology based systems, which we return to later.

Dempster-Shafer Theory Dempster-Shafer theory (DST) (Dem68; Sha76) is a mathematical theory of evidence that provides the ability to combine evidence from different sources to arrive at a degree of belief in a proposition. It is a probability-based approach that has the ability to preserve ignorance, that is, it does not require belief be assigned to all propositions, only to those where it is known. It further supports the allocation of belief to sets of propositions, not only singletons, supporting its specification at varied levels of granularity. We discuss DST in detail in Section 5.1.

Wu et al. (WSSY02) apply the theory to monitoring of people conversing in a meeting room, using an audio and camera sensor to detect the focus-of-attention of a subject.

They introduce the concept of a dynamic discount factor for sensors that changes over time, that relies on ground truth availability. Hong et al. (HNM+09) apply DST to define an evidence based activity model that uses sensor data for activity recognition in a smart home, with examples of making drinks. Hong et al. expand on Wu's work by using evidence propagation to bring evidence up through a hierarchy, so that activities can be recognised, as opposed to just abstracted contexts. Hong's approach describes how belief can be distributed to situations, but does not include a decision stage to determine occurring situations. Zhang et al. (ZCZG09) use DST for reasoning about activities. They propose excluding evidence sources with small belief contributions to both simplify the computational complexity of the model, and as a solution to Zadeh's paradox (Zad86), whereby conflicting evidence sources can give paradoxical results by granting majority belief to a minority opinion. Finally, building on the work of Strat (Str87), McKeever et al. (MYC+10) devise a temporal extension to DST that allows sensor data to be temporally projected. This allows evidence that occurs at a point in time to be projected to a later time, based on the expected duration of a situation as selected using expert knowledge.

DST heavily relies on expert knowledge in that the evidential hierarchy needs to be pre-defined and the uncertainty of sensor data needs to be either provided by experts or derived from sensor manufacturers. When the quality of sensors is not well known or there are a large number of sensors, this approach may suffer from badly defined expert knowledge or may require more knowledge than experts might reasonably be able to provide.

Ontological Reasoning Gruber et al. (Gru95) and Guarino et al. (Gua98) define an ontology as "a formal explicit specification of a shared conceptualization"; "a logical theory accounting for the intended meaning of a formal vocabulary". Put simply, an ontology models a particular facet of the world in terms of its concepts and the relations that exist between them.

Models for representing sensor data have been developed using a number of ontology frameworks (BMPM09; w3c; CHL+09; RKT05). However, the main body of work makes use of Semantic Web technology (BLHL01), principally the Description Logic (BCM+03) based subset of the *Web Ontology Language* (*OWL-DL*) (KPSR+09; GHM+08). OWL's capability to encode complex knowledge – which we discuss in detail in the next chapter – supports the representation of sensor data heterogeneous sources within a uniform model, annotation of the data with its semantics in relation to the system being modelled, reasoning over the data to derive additional knowledge, and querying of the model as a basis for driving system behaviours. This

includes ontologies for encoding sensor datasets (YSD+11b; YSD+12; BMJ+11), or vocabularies describing sensors, devices, people, services, and other pervasive systems concepts (CBB+12; SHS08; YCDN07; RMCM03; SKDN09; CFJ03; CFJ04; LLD07; ZHK10)

Typically, ontology-model-based situation recognition is conducted by augmenting standard reasoning services offered by ontology reasoners with more expressive techniques. For example, Paganelli et al. (PG11) and Toninelli et al. (TMKL06) use standard ontology reasoning to determine class subsumption relationships and model consistency, while developer-supplied production rules, of the form *if* ... *then*, support the inference of application specific knowledge, such as health alerts from monitored body conditions and colocation of people from a shared spatial relationship. Zhang et al (ZHK10) use the Semantic Web Rule Language (HPSB+04) to reason on aspects that cannot be expressed in OWL, such as specifying that a subject is far from home if their current location is over 100 miles from their house. Ye et al. (YSD11a) use production rules to infer complex relationships between domain knowledge and situation concepts (such as overlap and conflict) that go beyond that expressible solely with OWL. Riboni et al. (RB11b) show how the introduction of OWL 2 mitigates, to an extent, the need for an additional rule-based language. Meditskos et al. (MDEK13) propose a general framework for activity recognition that combines OWL 2 with the SPARQL rule language for additional flexibility. Finally, Anagnostopoulos et al. (ANH07) devise a framework for situation estimation based on a fuzzy mapping of sensor values to a hierarchy of concepts. The similarity of specifications defined by an expert is compared with current inputs to determine the most likely situation and the degree of belief that it is occurring.

Beyond rule languages, some of the learning- and knowledge-based approaches to situation recognition described above are backed by OWL environment models. Based on work by Ding (DP04), Gu et al. (GPZ+04; GPZ05) and Truong et al. (TLL05b; TLL05c; TLL05a) propose a set of probability annotations as an extension to OWL (KPSR+09) that support the modelling of prior and conditional probabilities associated with context, and the automatic construction of a Bayesian network from this information. The Bayesian network and probabilistic approaches of Ranganathan et al. (RAMC04), and the Event Calculus approach of Chen et al. (CNM+08) also map to OWL.

2.8.3 Hybrid Approaches

Chahuara et al. (CFPV12) use Markov Logic Networks to recognise daily activities in the home. Inputs to the Markov Logic network take the form of a manually specified

set of first-order logic rules that map discretised sensors values to the activity class they represent, with weights learned for each rule (LD07). The Markov Logic network outperforms both SVM and Naïve Bayes approaches in a scripted experiment. Helaoui et al. (HNS11a; HNS11b) also use Markov Logic to specify temporal relationships between activities to improve the model accuracy. For example, specifying that the activity of setting the table must precede both the activities of eating breakfast and clearing the table (HNS11a). The model is trained to identify the start and end times of activities, allowing interleaved activities to be identified as those that have started, but not ended.

Riboni et al. (RB11a) support statistical reasoning through multi-class ridge estimation (CH92) by using their ontology model to discard situations as valid candidates if the ontology model contains information (for example, a person's location) that conflicts with candidate results. Ye and Stevenson (YS13), and Ye et al. (YSD14b) leverage the semantic similarity of concepts as calculated from their ontological relation to partition streams of sensor events into fragments, each of which corresponds to a single ongoing activity. A technique based on the Pyramid Match Kernel (GD05) is employed to distinguish an unexpected sensor event as either a sensor noise or another concurrent activity. Ye et al. (YSD14a) also propose an unsupervised situation recognition technique that comprises automatic segmentation and clustering of streamed sensor data based on ontological similarity.

Finally, Gu et al. (GCTL10) devise an approach to activity recognition based on mining web-based activity guides (previously proposed by Perkowitz et al. (PPFP04)), for example, instructions for making a cup of tea. For a given set of activities, the approach generates a set of weighted terms for each activity based on the frequency that terms appear in the written instructions. The process further constructs a set of *contrast patterns*, that describe the significant differences between any two activity classifications. An initial laboratory study based on a wrist-worn RFID reader showed a recognition accuracy of 92% for a range of household activities.

2.9 Intelligibility

Intelligibility concerns the ease with which the activity recognition process can be scrutinised and understood by the developer or end user. In pervasive environments where people need to understand the reasoning process, or where there is a significant error, the cause of which needs to be diagnosed, a lack of transparency can present a challenge.

While the distinction between knowledge-based and learning-based recognition techniques is not as clean as describing one as black-box and the other as white-box, is generally true that the decisions of learning-based techniques are not as intelligible as those of knowledge-based techniques at best, and unintelligible at worst. Decision trees (WF05) provide the clearest example of a machine learning technique where the decision making process is most easily inspected. It is not straightforward to pinpoint the limit at which manual inspection of such a learned model becomes impractical, however, their clarity depends largely on the size of the learned tree and the number of nodes it contains. The more that noise is present in training data, the less likely it is that the learned tree will be easily understood.

The decision making processes of knowledge-driven approaches, be they based on propositional (RAMC04; HI06) or fuzzy logic (ANH07; DHKZG08), Dempster Shafer theory (WSSY02; MYC+10), pure ontological reasoning (TMKL06; RB11b), or ontological reasoning plus rules (ZHK10; YSD11a), are, by their nature, easily open to inspection; limited only by the complexity of the manually constructed model. To date, all hybrid approaches (GCTL10; CFPV12; YSD14a) focus on enriching the knowledge available to the learning algorithms, not on aiding the transparency of their decision processes.

2.10 Engineering Effort and Expert Knowledge Requirement

Significant effort is often required to apply or adapt a technique to an individual pervasive environment, although this effort is not uniform across approaches. With learning-driven techniques, developers have the requirement to extract features of sensor data to serve as input to the recognition effort (FE), and perform simple pre-processing of data, such as generating summary statistics for extracted features that might best serve the algorithm. For some learning-driven techniques, for example, Bayesian Networks, expert knowledge is required to define the graph structure (CCKM01) although this structure can also be learned (Mar03). In contrast, knowledge-driven techniques typically require significant effort and expert knowledge to manually construct a model (MC). This can require an expert to specify derivation rules (ZHK10; MDEK13), hierarchies of concepts and relations of interest (TMKL06; PG11), and to define functions to map and abstract sensor data to these manually defined constructs (DHKZG08; MYC+10).

No training data is required in pure knowledge-driven approaches, with expert knowledge used to define the relationships between model concepts and situations (YSD11a),

while learning-based approaches often require a considerable investment in effort to collect the training data necessary to learn a model and its parameters (TSTN06). In supervised approaches, this is complicated by the need to collect accurate ground truth concurrently with (vKNEK08) or after (TIL04) the data collection process. Unsupervised learning approaches provide one way to mitigate the effort required to collect ground truth along with training data (CP99; PLFK03; BVMR06), although some manual intervention is often required, for example, specifying the number of situations represented by the training data (RCHSE11). Knowledge-based and hybrid approaches can also require the addition of expert knowledge describing intra-situation dependencies and sequence restrictions, or direct relationships between situations and sensor data (GCTL10; CFPV12; YSD14a; YSD14b).

It is often assumed that learned models are static, with the introduction of new sensors or new situations requiring that the models be retrained or redesigned. The problem of automatically evolving a situation recognition model without this "reset" step has been largely ignored to date (RCHSE11).

2.11 Uncertainty and Noise

One area where learning-based approaches are generally superior to knowledge-based approaches is their ability to handle uncertain or noisy sensor data. In addition to constructing models that learn the associations between sensor data and situations, some machine learning techniques are capable of accounting for the effect of noise on these associations. For example, Naïve Bayes approaches mitigate the effect of missing values in the decision process by ignoring them when computing probabilities (KZP06). Decision trees can be prone to overfitting when noisy data is present, however, pruning strategies designed to avoid overfitting can mitigate the impact of noisy data on the model. Missing or noisy data can have a great impact on nearest-neighbour-based approaches, leading to an incorrect identification of neighbours and consequent misclassification.

There are two main reasons why knowledge based approaches are often less tolerant of noise than machine learning techniques. The first covers the case where the model has no specific capability for representing or accounting for noise. For example, in early rule-based and pure ontological approaches (Lok04a; TMKL06). The second encompasses the limit to which a domain expert can accurately capture noise and uncertainty within a knowledge-based model, and manually tune the model's parameters to handle its effect. For example, the certainty of sensed information can be discounted using expert

knowledge of the accuracy of the data source (CYL+07), or through a temporal decay function that serves to lessen the contribution of data as time passes (KC14). Both cases can be considered as types of fuzzy membership function, encoding data's degree of membership to a concept. DST can also deal with missing and incomplete knowledge by assigning the contribution of sensor data to set of situations, or directly to the concept of uncertain knowledge (MYCD09b). For particular smart-space datasets, it has been shown that knowledge-driven techniques can achieve recognition accuracy comparable to that of machine learning techniques (McK11).

2.12 Evaluation Approach

Standard practices in evaluating situation recognition techniques have evolved from scripted laboratory sessions (MBSM04; MSW+05) to real world testbeds (TIL04; ILB+05; vKNEK08). However, by and large, the majority of situation recognition techniques surveyed are evaluated using custom datasets (CD), collected by their authors solely to evaluate the technique at hand, and not made publicly available. The lack of a public, standard testbed makes it difficult to directly compare techniques and results, to verify that a technique operates with datasets other than the proprietary data on which it is tested, and means that the human investment in effort and time is often duplicated to collect data for very similar studies (TCSU08; WRB11). Even where publicly available datasets are used, evaluations often only focus on a subset of situations, or use different methodologies.

Examples can be selected from the literature to show where any class of technique outperforms another on a similar recognition challenge. However the reasons why this is the case are typically opaque.

2.13 Knowledge Transfer

Situation recognition techniques typically require substantial amounts of training data to recognise relations between situations and sensors, and to mitigate the presence of noise. While unsupervised learning is one path to lessening this burden, another is *knowledge transfer*, the ability to take a situation recognition model learned in one environment, and apply it to another as means of recognising a known situation, or using it as a starting point to recognise a new situation. Goals of this research area include reducing the effort required to set up and train new situation recognition systems, making situation recognition more versatile and robust, and examining how to effectively reuse existing

knowledge that has been generated (CFK13).

In systems where recognition is based on wearable or mobile sensing platforms, assessing the transfer of knowledge between subjects typically forms part of the standard evaluation, whether adults (BI04; MSW+05; RDML05; LCK+05; YT12), children (WAS+12), or dogs (LHH+13) are involved. However, techniques supporting knowledge transfer in infrastructure-based sensing are less prevalent (vKEK10b; vKEK10a; ZHY09; RC11). Cook et al. (CFK13) survey further research in this area, and outline possible grand challenges for the community.

2.14 Summary

This chapter reviewed the set of significant situation recognition techniques in the literature. We summarised the findings of existing literature surveys, and focused this review on drawing from and extending the taxonomies of existing surveys to suggest a categorisation for significant techniques in the field, and on providing an updated survey that includes literature not covered by existing surveys.

It is clear that there is no "one size fits all" approach to situation recognition. Each of the techniques discussed has advantages and limitations. Further to this, there exists a demonstration where each class of technique outperforms the other on similar reasoning tasks, with little intuition as to why this should be the case. As a consequence, there is a clear need for a situation recognition benchmark, consisting of datasets that will allow techniques to be compared. The large majority of surveyed techniques are evaluated either on proprietary datasets, or on open datasets but using different methodologies. This often makes the comparison of techniques and their results impossible.

We have also identified that in addition to the high level categorisation of knowledge-based and learning-based techniques used in other surveys, there is an emerging trend towards a third category of techniques that hybridises aspects of both. The work described in this thesis falls into this category. We note that the trend away from the use of knowledge-based techniques towards complex learning-based techniques has typically involved the loss of human ability to scrutinise the reasoning process. The work in this thesis focuses on models where this property is retained.

Our particular interest in investigating a hybrid knowledge- and learning-based technique lies in the ability to leverage expert knowledge to define a set of rich modelling primitives for an application domain, while harnessing the strength of learning to discover patterns and mitigate noise. In taking this approach we seek to develop a model that is not only expressive and capable of high recognition accuracy, but also has a

white-box decision process that is intelligible to the human eye, something that no hybrid technique has yet investigated.

In the next chapter we describe the foundations for our knowledge model, using set theory to develop a uniform approach to the modelling of concepts and inter-concept relationships in knowledge domains, that will later provide the basis for our situation recognition models.

CHAPTER

THREE

THE SEMANTIC WEB APPLIED TO THE MODELLING OF PERVASIVE SYSTEMS

In the previous chapter we overviewed the state-of-the-art in activity recognition. In particular, we highlighted the recent trend towards hybrid knowledge- and learning-based techniques that leverage expert knowledge to define a set of rich modelling primitives for an application domain, while harnessing the strength of learning to discover patterns and mitigate noise. Existing hybrid examples focus on enriching the knowledge available to the learning algorithm, not on aiding the inspection and interpretation of its decision making process. In this thesis we seek to address this gap by developing a model that is not only expressive and capable of high recognition accuracy, but also has a white-box decision process that is intelligible to the human eye.

This chapter focuses on the modelling of information in pervasive systems. We begin in Section 3.1 by overviewing a range of modelling technologies, motivating their use for modelling context, and overviewing their benefits and limitations over competing alternatives. Based on this analysis, the remainder of the chapter then focuses on one particular application of Semantic Web technologies: supporting the representation of information that informs the situation recognition process in pervasive environments.

In Sections 3.2 to 3.6 we develop a reusable top-level ontology model that provides a common substrate for developing domain and application ontologies for pervasive environments. The model employs set theory to develop a uniform approach to the modelling of concepts, using a small, core set of semantic properties to relate them. Reasoning supports the correlation of information from its low level definitions to higher level context statements, inferring the relationships between higher level concepts, and managing the uncertainty associated with sensed information automatically.

Section 3.7 presents a detailed worked example that brings together all the concepts discussed in the chapter as they are implemented. A discussion of the merits and limitations of the approach is presented in Section 3.8, before we finally conclude in Section 3.9 with a chapter summary.

Earlier versions of aspects of the work described in this chapter have been published in Stevenson et al. (SKDN09), Stevenson et al. (SYDN10), Stevenson et al. (SYD10) and Ye et al. (YSD11a).

3.1 Modelling Pervasive Systems

The path towards realising Weiser's vision (Wei95) is paved with substantial challenges. The vast expanse of available information about people, places, devices, services, and environmental conditions necessitates the development of sophisticated algorithms to support services in becoming aware of their surroundings: to seek, discover and filter information relevant to their goals, and to aggregate, interpret and reason over such information.

As pervasive computing scenarios usually do not assume the existence of proprietary services through which access to all information is regulated, these responsibilities must be partitioned across the environment's participants, with awareness arising from the interactions between many individuals, each exposing aspects of their own state and perceiving the states of others to further increase their awareness of their surroundings. Such context awareness is a key enabler of intelligent service provision (CCDG05).

The openness of this world view and of such interactions therefore makes paramount the mechanisms by which data is modelled and represented, in order that data or knowledge may straightforwardly be discovered, distributed and unambiguously interpreted by collaborating individuals in a pervasive system.

3.1.1 Candidate Technologies

Context models provide the basis for building pervasive systems by providing developers with a symbolic model of their software's operating environment. Modelling techniques have evolved from simple attribute-value representations of people and places to complex ontological models that provide a means of describing groups of related resources and the relationships between them. Such models can hold rich, detailed descriptions of environments, their contents, associated sensing infrastructure, sensor data, and situations inferred from lower-level data. Here we briefly overview the

advantages and disadvantages of different formulations of context model found in the literature: attribute-value and tuple-space models, entity-relationship models, object-oriented models, and three varieties of markup models: structured, semi-structured, and unstructured.

Attribute-value (ST94; CLK08) and tuple-space context models (JF02; GDL+04), both based on dictionary lookup of values or objects via property names, provide simple techniques for sharing information about entities. The strength of such models lies in their ease of inspection, comprehension, and manipulation, however they lack schema to support validation and verification, and are unable to represent complex structured information at the model level, for example meta-properties (such as the accuracy of a value). While these may be encapsulated within stored objects, they are opaque at the model level. Such approaches are best suited to small, proprietary models of information, where a single party retains overall control of the design and scope of the model.

Entity-relationship-based context models (JS03; HIM05) provide increased capabilities at the modelling level, exposing relationships between concepts in different parts of the model. However, they lack the formality necessary to support model composition. For example, there is no way to automatically determine if lexically identical terms in different parts of the model share the same meaning (e.g., "frequency" might refer to a measurement in Hertz or a period in seconds), or if two different terms are used as synonyms (e.g.,"room" and "space").

Object-oriented context models (that is, models developed using UML (Gro07), or the specific constructs of an object-oriented programming language) in the literature (CMD99; Bar05) leverage programming language support for distributed composition and type based validation. As with entity-relationship models, runtime model composition is difficult without explicit constructs to describe equivalences between independently developed parts of the model. Constraint modelling is typically also limited. For example, with cardinalities, a property may take no value (for example, null), a single value, or many values (a collection type). Although more detailed restrictions and other forms of structural constraints can be enforced by accessor methods, these are "black-boxes" that cannot be reasoned on easily. The intertwining of the data model and the code that manipulates it provides a barrier to ease of model inspection, ease of data exchange, and portability across devices, platforms and programming languages.

Markup Models Markup models support the annotation of values with tags and attributes in order to prescribe their meaning. Broadly speaking, there are three classes of markup model - structured, unstructured, and semi-structured with the literature con-

taining many examples of each (Bro96; DSA01; RC04; CFJ03; RMCM03; GKK+03; HLI04; DNC+07).

Structured models take the form of a hierarchy of tags, where each tag has an associated definition describing what it can legally enclose (other tags, values, and cardinality or type restrictions), while unstructured models provide no underlying schema against which to validate or compose data. For example, use of XML (BPSM+) and JSON (jso13a) can fall into both these categories depending on whether or not they are used in conjunction with their schema languages, XML-Schema (GSMT+08; PGM+08) and JSON-Schema (jso13b).

Semi-structured models retain the ability of structured models to formally describe the concepts and properties in a given domain of discourse, without restricting what a concept (tag) may legally enclose. Semi-structured models are commonly realised in the form of ontology languages such as the Web Ontology Language (OWL) (KPSR+09). The formal basis of this language (where each term and entity is uniquely identifiable by a URI) makes it possible to straightforwardly combine concepts from different ontologies. This allows entities of the same class to be described using different sets of properties depending on what needs to be represented within a given environment — a strength from a modelling perspective if not a programming one.

No silver bullet technology for developing a context model exists. The suitability of each of the above techniques is tightly coupled to the requirements of the application or system being developed: its scope, complexity, size, requirements for distribution, openness, model proprietary, degree of platform heterogeneity, and so on. However, among these alternatives, semi-structured models, specifically ontological models, provide the most flexible option due to the ease with which they can typically express complex information. This view is shared by Strang and Linnhoff-Popien (SLP04).

3.1.2 Semantic Technologies for Modelling Pervasive Systems

As a consequence of the above analysis, in this work we choose to model pervasive system concepts using OWL and the suite of related technologies that fall under the larger banner of *The Semantic Web*—a movement led by the World Wide Web Consortium (W3C) to provide standard data formats and a common framework for encoding and sharing machine processable data across "application, enterprise, and community boundaries". We justify this choice over alternative ontology language frameworks with reference to the availability of open source semantic technologies, the complexity they

handle, and the large, standards-driven communities that maintain them.

In the following, we overview the important role semantic technologies can play in the realisation of pervasive systems from the perspectives of knowledge modelling and representation, inspection and manipulation, discovery and distribution, and support for integrating with legacy data sources and technology. We also touch upon the limitations and common criticisms of the technologies.

3.1.3 Knowledge Modelling

Ontologies support the capture and specification of domain knowledge with its intrinsic semantics through agreed terminology and formal axioms. Based on these, ontologies provide a set of modelling primitives to define classes, individuals, their attributes and their relations, each piece of terminology associated with its intended meaning (Gua98). When an ontological vocabulary models a particular aspect of reality it specifies an *ontological commitment*, that is, the underlying conceptualisation about the intended domain, including domain concepts, attributes, and relations between them. The vocabulary makes data understandable, sharable, and reusable by both humans and machines.

A number of computer science fields, including e-commerce (OWL01; ETB04), information integration (Gui02; VV04; LVB+03), the Semantic Web (GMBM05; SD06), and pervasive computing (SLP04; PVdBW+04; CFJ04; RSK+06; CNM+09) have identified ontologies as a promising technique for knowledge modelling, that addresses data, knowledge, and application heterogeneity.

In the open systems view of pervasive computing, where multi-domain interactions are based on opportunistic encounters between devices, where one may have no prior knowledge of the other's existence or capabilities, successful interactions depend entirely on a service's capability to interpret the structure and semantics of the content that another exposes. Without the use of strong, formal modelling constructs such as ontologies on which to build a shared interpretation of the environment, this becomes an intractable problem.

3.1.4 Knowledge Representation

The Resource Description Framework (RDF) (MM04) provides a formalisation of a directed graph (with nodes representing resources and arcs representing properties). Its semantics are prescribed by two ontology languages: RDF Schema (RDFS) (GB04)

and OWL (KPSR+09); the latter being the more expressive of the two. RDFS provides a basic vocabulary for dividing RDF resources into classes, restricting the classes of resource a property may legally relate, and introduces *subClass* and *subProperty* properties to capture relations between classes and properties at different levels of abstraction. OWL provides a more expressive ontology language by, for example, supporting the expression of functional, transitive, symmetric, and inverse properties. Equivalent properties and classes may be declared, and cardinality restrictions allow constraints to be placed on the legal structure of class members.

For pervasive systems, the benefits of these technologies are a direct result of their formality. RDF's use of URIs to identify concepts and properties, combined with OWL's support for modelling equivalent classes and properties makes determinable whether lexically identical terms in different parts of the model share the same meaning, or if two lexically different terms are homonyms or synonyms. This formality has several advantages: no single authority is responsible for engineering ontologies or producing data; entities may be described by combining concepts from different ontologies; and combining both ontologies and data from multiple sources, for example, in the description of a person or a sensor, is straightforward. Off-the-shelf reasoners, for example, Pellet (SPG+07), can also be used verify the consistency of environment models that are described using published OWL vocabularies.

This feature set directly address the limitations of other candidate technologies identified in the previous section: Complex data structures are supported (a limitation of attribute-value and tuple space models), rich constraints are made explicit (a limitation of object-oriented models), terminology ambiguity is resolved (a limitation of all models except structured and semi-structured markup), and data structures are flexible to support the dynamic integration of descriptions from multiple sources (a limitation of object-oriented, entity-relation, and structured markup models), while retaining the ability to enforce restrictions and validate the model (a limitation of unstructured markup models).

Further to this, the domain-neutrality of RDF supports the vision of a unified data model, under which multiple applications across disparate application domains can execute, each projecting a view onto the model as its needs dictate.

3.1.5 Knowledge Inspection and Manipulation

Technologies for inspecting and updating RDF stores exist in the form of the expressive SPARQL (SH09) and SPARQL Update (GS09) languages. SPARQL supports queries consisting of triple patterns, conjunctions, negations, disjunctions, and optional patterns,

while SPARQL Update supports the conditional insertion and removal of triples from an RDF store.

The desirability of tight integration between a data model and programming language can be observed from the plethora of technologies available that strongly integrate relational and XML models. For example, Hibernate (BK06), Rails (TH06), and Django (FBC08). Several commentators advocate mapping from ontologies to strongly-typed programming languages in order to eliminate a large class of typographic errors at compile time that might only be discovered at runtime or go undetected if a dynamic approach were taken (Gol03; VBRM+07).

Goldman (Gol03) describes two classes of ontology-based applications: *generic*, where only knowledge of the ontology language is required to construct the application, and *ontology-specific*, where an understanding of the semantics or meaning of any particular term inside an ontology is important. Validators and verifiers belong to the first class of application, while context-aware applications (e.g., a program that locates the nearest petrol station to your current location) fall into the second.

Approaches for working with RDF-based data models sit along a spectrum ranging from the ontology agnostic to the domain-specific. The most generic APIs (e.g., the model APIs of *Jena* (McB02) and *NG4J* (BCW05)), provide operations that support manipulation of the data-graph (e.g., the addition and removal of triples or quads). *Tramp* (tra14) binds RDF nodes to Python dictionary structures, allowing indexing by property names, while *SETH* (BH06), supports dynamic schema extension through the writing of inline RDF/OWL snippets in Python code. Increasing the level of abstraction, *The OWL API* (HB11), and *O3L* (Pog09) are libraries based on OWL concepts and semantics that provide reflection-like capabilities for indexing into a data model using classes such as **OWLClass**, **OWLProperty**, and **OWLRestriction**.

Towards the domain-specific end of the spectrum sit tools that map from object-oriented representation to ontology-oriented representation and vice versa. The former category of tools, to which *Jenabean* (jen14), *Sommer* (som14), *Spira* (spi14), and *Empire* (emp14) belong, operate through manual annotation of the fields of Java Bean style objects, from which ontological descriptions are then extracted or persisted to a back-end. While this approach removes the need for developers to construct ontologies natively, these tools realise only a small subset of the RDFS and OWL specifications. Of the latter category, *Sparta* (spa14) binds RDF nodes to domain-typed Python objects and RDF arcs to attributes of those objects, while *ActiveRDF* (ODG+07) provides RDF data inspection capabilities via a Ruby-based virtual API that intercepts method calls and dynamically maps them onto SPARQL queries against a data model. ActiveRDF does not use schema information as part of its mapping process, and while this allows

for great flexibility in terms of the ability to easily manipulate schema and structure of individuals at runtime), this comes as a necessary trade off against the ability to verify application code both in terms of type correctness and typographical errors against an ontology.

There are also tools that generate domain-specific APIs for weakly-typed languages, and strongly-typed programming languages directly from an ontology. *RDFReactor* (Vok06) generates Java libraries for RDFS models that act as a proxy to a triple store back-end at runtime, while *Owl2Java* (owl14) gives OWL ontologies a similar treatment. *OntoJava* (Ebe02) and the work of Kalyanpur et al. (KJ04) (on which Jastor (jas14) is based), perform the conversion of both ontology axioms and rules into executable code. In carrying out this research we developed *Sapphire* (SD11), a tool that builds on the foundations of this final class of tools to provide improved support for OWL's feature set, offering novel features in the form of an extensible type mapping system, support for reification, support for the open-world assumption, and support for the dynamics of multiply-classified OWL individuals.

For pervasive systems, the union of RDF, RDFS, OWL, SPARQL, SPARUL, and any of the object mapping technologies provide a complete framework within which to represent information using terms from the most appropriate vocabularies for a particular environment, or set of applications, and inspect and update their contents using mainly standard tooling.

3.1.6 Knowledge Distribution and Transfer

RDF's data model supports the seamless merging of data from heterogeneous and distributed sources. Both benefits are a direct consequence of RDF's use of URIs to identify resources that provide triples with unambiguous global semantics: An unambiguous statement can be made about any object for which you have a URI. This can be contrasted with, say, traditional database schemata such as entity-relationship models, whose terms and relations have no prescribed semantics, and XML Schema, which is concerned with the hierarchical structure of data elements and not with capturing the relations between data elements. In addition, neither technology is predisposed to easy integration of data adhering to multiple schemata.

The formal, agreed terminology that serves as the foundation of RDF's distributed data model, makes it possible to share and reuse data and knowledge across different components in a system and across different systems (RSK+06; CNM+09).

For pervasive systems, the implication is that a realisation of a data model based on

semantic technology also inherits this predisposition to ease of distribution.

3.1.7 Knowledge Discovery

The diversity of service offerings in a pervasive computing environment gives rise to the need to balance the flexibility of resource descriptions with the consequent difficulty in constructing requests to discover them—a process called matchmaking. A matchmaking algorithm takes a request and a set of resource descriptions as input, and outputs the set of resources that satisfy the request. Many matchmakers adopt simple syntactic schemes; for example, named interfaces or predefined categories (Wal00), or sets of attribute-value pairs to which string and numeric comparisons can be applied (GPVD99). Such approaches are limited by an inability to compare semantically equivalent but syntactically different concepts or handle approximation. As a result, these are insufficient for open, and unstructured worlds, necessitating the investigation of alternative approaches.

In the context of knowledge discovery, OWL's standard classification mechanism can be re-cast as a resource discovery technique—matching ontologically-founded descriptions of resource to requests in the form of template-like OWL class descriptions.

A more expressive approach involves the use of resource and class descriptions as part of a *semantic matching* algorithm—resolving the fuzzy membership relation of the resource to the request to a degree $p, p \in [0:1]$, while accounting for concept similarity. As with a standard OWL class description, a request expresses the intersection of a number of properties, each named concept or existential restriction belonging to a resource. Semantic matching is then conducted by assessing the similarity of the constituent components of the request and resource description. Paolucci et al. (PKPS02) and Li et al. (LH03) introduce ordered degrees of matching to categorise the semantic compatibility between a request and a resource, of which there are five: *exact*, *plugin*, *subsume*, *intersection*, and *disjoint*. Bandara et al. (BPd+08) extend this model with scoring mechanisms to support differentiation of matches falling within the same category by scoring their *semantic distance*, for example, by using a heuristic based on the number of intermediate concepts that relate two terms (SSS07). In Stevenson et al. (SPM+13; SVY+) we use bio-inspired mechanisms to realise semantic-matching-based resource discovery in a distributed pervasive ecosystem.

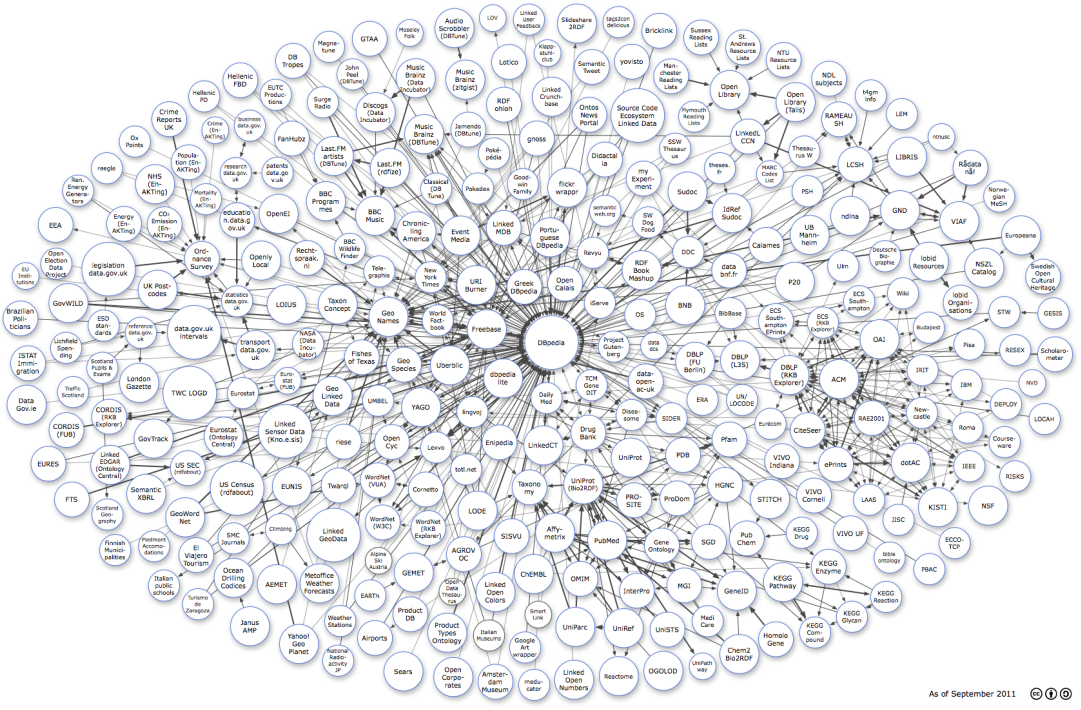


Figure 3.1: The Linking Open Data cloud diagram showing datasets that have been published in Linked Data format, by contributors. Reproduced from `linkeddata.org` under the CC-BY-SA 3.0 Unported License.

3.1.8 Knowledge Integration

A final benefit of adopting RDF as an implementation technology is that an environment model can be built upon, and interlink with, existing ontology-based domain knowledge through the principles of Linked (Open) Data—a set of best practices for exposing, sharing, and connecting pieces of knowledge on the Semantic Web (BHBL09). The premise of Linked Data is that by using URIs to link to data sources, a semi-structured interlinking of ontologies and ontologically-represented data emerges as a web of data that can be navigated and explored. Figure 3.1 illustrates the extent to which existing datasets have been made publicly available through publication in a Linked Data format.

For pervasive systems, Linked Data has two implications. Firstly, it provides a structured means of accessing and integrating data from existing "non-pervasive" data sources with existing RDF representations (or via an RDF wrapper for legacy data sources and technologies). This has potential to bootstrap systems with the knowledge held by myriad social information sources on the web (SSDN09; RDM+11). Secondly, by designing pervasive systems to make use of RDF and OWL, and supplementing it with an appropriate web-facing interface, it provides a means for new data generated within a pervasive system to be straightforwardly linked and exposed to the outside world, and made accessible to web-based applications—for inspection, additional processing, or

archival purposes.

The *Vocabulary of Interlinked Datasets* ontology is one emerging effort to express metadata about RDF datasets, with the goal of providing a bridge between the publishers and consumers of RDF data, for the purposes of data discovery, cataloging, and archiving of datasets (ACHZ11).

3.1.9 Semantic Technology Limitations and Criticisms

While the above discussion provides support for the use of Semantic Web technologies – and it is our view that it is currently the most promising solution available – it is no panacea. In particular there are a number of important features where it is limited, such as lack of native support for representing and reasoning over dynamic knowledge, lack of enterprise-grade tooling, and slow adoption in the developer community.

There is a significant gap between the currently available Semantic Web technologies and the need for native temporal data modelling and reasoning. In addition to "home-brew" strategies that use the standard RDF data model (SYD10), extensions to the model semantics and the SPARQL query languages have been proposed (LLPG05; GHV07; PUS08; TB09; EWK90), however, there is yet to be movement towards the development of standards.

Another issue is that state-of-the-art Semantic Web technologies are designed to handle large volumes of static data, while pervasive systems bring the need to work with highly dynamic data, often with real-time reasoning and query responsiveness requirements (SYD10). A consequence is that the majority of off-the-shelf reasoners do not provide support for deletion of facts after their insertion, requiring all inferences to be re-computed after any operations on the knowledge base. The Pellet reasoner (SPG$^+$07) is one notable exception that provides support for handling knowledge base updates incrementally.

Technologically, the readability and accessibility of serialised RDF data models has been criticised. The 'standard' serialisation, RDF/XML, is verbose, not human friendly, requires RDF's terminology (of resources, properties, literals and blank nodes) to be recast as XML elements and attributes, and requires a graph to be encoded as a tree. However, this is primarily an issue of implementation, rather than a criticism of the model itself. The development of alternative 'pretty' syntaxes (for example N3 (BLC11)), although not yet endorsed as W3C standards, alleviate this concern and are widely understood by tooling.

A more pressing concern is the steep learning that Semantic Technologies may present

to developers, and the commitment required by developers and companies to adopt them. Considerable familiarity with namespace semantics, ontologies, reasoners, and the vocabulary and associated semantics of RDF, RDFS, and OWL are required before development of simple systems can begin. The marketplace for assistive ontology development tools is currently small, with only a few choices available for developing and validating RDF/OWL ontologies. Protégé (NFM00), TopBraid Composer (top), and NeOn Toolkit (neo) are among the most prominent.

The decision to adopt Semantic Web technologies, whether taken by an individual or organisation, comes as a tradeoff between its advantages and risks. The ability to define a structure for knowledge that exactly matches a chosen sub-domain, to describe the richness of this structure, to have it compose cleanly with other such descriptions of complementary sub-domains defined independently – and to be able to exchange all this knowledge with anyone on the web is clearly desirable. However, the current reality is far from this vision. Adopters are buying into a vision with the hope or expectation others will follow suit, without it being clear to what extent that this will be the case. In December 2013, the W3C announced that all Semantic Web Activities are being superseded by the Data Activity, with a "focus on deployment and integration within the broader landscape" (Zai13).

Should it be the case that the Semantic Web is not realised using the currently proposed technology suite, it is likely that any successor technologies will support many of the desirable features of RDF and OWL, and will provide migration paths. For example, recent emphasis on JSON in the web community has given rise to JSON-LD, a light-weight Linked Data format based on JSON that aims to help JSON data interoperate at Web-scale (Lan13). The JSON-LD specification contains the concept of a "context", which provides basic interoperability with the RDF model by allowing object properties in a JSON document to be linked to properties defined in an ontology.

3.1.10 Summary

The suite of Semantic Web technologies has the potential to play an important role in knowledge management of pervasive systems from the perspectives of modelling and representation, inspection and manipulation, discovery and distribution, and support for integrating with legacy data sources and technology. The technology suite is expressive and flexible, although adoption requires mastering a steep learning curve.

Most is to be gained from the Semantic Web at the point where it is fully realised – adopted by the masses – and there is a distinct possibility that this may not come to fruition. However, even if only partially realised, the Semantic Web provides the ability

to capture a set of data and metadata, to reason on the captured information, and to openly exchange said data and model in a domain neutral way. These features improve upon the interoperability and semantic clarity over alternative technologies, supporting the view that developing pervasive systems using Semantic Web technologies remains a promising approach.

3.2 A Semantic Model of Sensed Information and Situations

In the literature, most approaches to designing ontologies for pervasive computing systems focus on the particular needs of applications and systems under development, with little consideration for reuse. This, in many ways, is a natural reflection of the state of the research field. Much work has focused on small lab-based research prototypes, with large, open, pervasive ecosystems (VZSD12) some way from realisation. It is also a reflection of the time investment required to produce high-quality ontologies, and the lack of established standards—although initial attempts at developing standards have been undertaken by groups such as the W3C Semantic Sensor Network Incubator Group (w3c13b) (closed in 2011), and the W3C Provenance Working Group (w3c13a) (closed in 2013).

Giving proper consideration to the shared semantics between applications and across applications domains – the Linked Data vision – can confer many benefits: reducing the need for ontology engineers to reinvent the wheel, therefore lowering required engineering effort; reducing the risk of encoding bias, where data representation choices are tightly coupled to a specific use rather than the general case; and, making easier the sharing of data across application and system boundaries through the use of common terminology and interlinked vocabularies. A rough analogy can be drawn with the art of developing a program, where an experienced software engineer will draw from existing libraries, customising their use and taking advantage of extension points in preference to rewriting the functionality provided by the library from scratch.

For these reasons, it is desirable to build domain ontologies on a substrate of a foundational ontology model, where domain concepts and knowledge are modelled uniformly among different types of information. In this section we propose such a model, using set theory to underpin the description of relations between concepts in an information domain (for example, time, location, mobility, or temperature).

The contribution of this approach is a reusable top-level ontology model that provides a conceptual backbone for developing domain and application ontologies for pervasive

environments, and a basis for using a common set of techniques to reason about knowledge expressed in any domain ontology that extends from the top-level ontology.

The model enables automated reasoning on the underlying correlation across the levels of information, inferring the relationships between higher level concepts. The resultant knowledge can be used, for example, to check knowledge consistency and to aid the process of application design by allowing actions to be triggered by the presence of information at different levels of granularity. While such a model can be used as the basis for any pervasive application, our focus in subsequent chapters is on describing its use as the basis for a situation recognition technique.

Section 3.3 starts by describing a small, core set of semantic relations common to most dimensions of an information space: *equality*, *subsumption*, *disjoint*, *overlapping*, and *adjacent* relationships. These common semantics provide a uniform way to represent and reason on domain knowledge. Ontologies built using them are straightforwardly interpreted by evaluating and comparing these underlying semantics of the concepts and terms they define.

Building upon the context model and its semantics, Section 3.4 describes a context model, that provides a general approach to derive knowledge about *contextual statements*, two-dimensional product spaces of information, in a structured manner. The context model provides a means of making fact-like assertions about the state of the environment through relating concepts. Through reasoning about the relationships between contextual statements, we can infer relationships at a higher level of abstraction, and detect inconsistency between context; that is, when two pieces of context report conflicting states of reality.

Section 3.5 and Section 3.6 then discuss use of the ontology model to represent temporally qualified contextual statements and to manage the propagation of state information associated with contextual statements, with a focus on the features we later use in developing situation recognition techniques.

Aspects of these models draw from work we present in Stevenson et al. (SKDN09), Stevenson et al. (SYDN10), Stevenson et al. (SYD10) and Ye et al. (YSD11a). Discussion relating to the modelling of adjacent concepts and propagation of uncertainty are extensions to this work, while the discussion on activity recognition from Ye et al. (YSD11a) is supplanted by the work in the following chapters.

3.3 Concept model

This section introduces a domain-neutral approach for representing and relating information within a formal ontology model. The model serves as a common basis for reasoning about different types of information based on their shared semantics.

3.3.1 Information Dimensions

Newby et al. define an information space as "the set of concepts and relations among them held by an information system. [An] information space is produced by a set of known procedures, and is changed through intentional manipulation of its content." (New96). Here, we think of an *information system* as termed by Newby, as the sensing capability of a collection of devices, or more generally the sensing capability of an environment such as a room, building, street, or city.

Information spaces, capturing all that can be sensed in an information system, can be subdivided into multiple dimensions of information, each with its own structural and relational characteristics. For example, within the information system of a house, one dimension of information is temperature, represented by a combined numeric value and scale; for example, $23°C$, or $73.4°F$, or a concept with meaning to a person like *hot* or *cold*. Another dimension with multiple representations is location, represented as a point or region within a 2D or 3D coordinate system, or with a symbolic name, like *kitchen*. Time (as an instant) also has multiple representations, for example 2:59pm or 14:59 in 12hr and 24hr clocks.

The set of relevant information dimensions varies with the information system being modelled. For example, a system tracking a mobile entity may be concerned with speed, acceleration, distance travelled, or distance to destination, while a personal health monitoring system may be concerned with heart rate, blood pressure, or respiration.

3.3.2 Abstracting Information

A small number of axiomatic definitions provide a formal basis for modelling dimensions of information. We begin with the concept of *units*—the set of smallest values that can be *measured* in an information dimension. Units are disjoint and exhaustive so as to cover anything representable in the dimension, whether finite or infinite, continuous or discrete. Examples of units for different information dimensions are given in Table 3.1.

In dimensions where multiple unit representations exist – for example, temperature

Dimension	Features	Units
Temperature	Continuous, Finite	Celsius (or Fahrenheit, Kelvin)
Boolean	Discrete, Finite	*True* or *False*
Time	Continuous, Infinite	Millisecond (typically)
Number of Interactions	Discrete, Infinite	Count (Integer)
Location	Continuous, Infinite	Cartesian coordinate system

Table 3.1: Ground value measurements in different information dimensions.

values can be equally well expressed in Celsius, Fahrenheit, or Kelvin – we assume a primary measurement system, to which others are mapped.

While irreducible units of measurement can be used, the ability to *measure* units is a practical consideration that serves to simplify a model. For example, milliseconds typically serves as a good representation for temporal instants, despite being sub-divisible.

Building on unit measurements, each information dimension can have a set of *concept values*, or simply *concepts* – symbolic terms that are human-understandable or useful to the application at hand – to which unit values are mapped. For example, in the temperature dimension ranges of Celsius unit values might be mapped to the concepts *cold*, *warm*, and *hot*; in the location dimension, a region of a cartesian coordinate space may be mapped to a room, building, or city; and values in the speed dimension to *slow* or *fast*.

We formalise this mapping in Definition 1.

Definition 1 (Mapping unit values to concepts)**.**

Let V^g be the unit value set of a dimension of information. A set of concept values V^a are defined via a mapping function $\mu : V^a \to \mathcal{P}(V^g)$, where $\mathcal{P}(V^g)$ is the power set of V^g.

3.3.3 Information Semantics

Using a set-theoretic approach, Definition 2 formalises how the semantics of concept values common to all information dimensions – *equal*, *subsume*, *overlap*, *adjacent*, and *disjoint* – are evaluated from their mappings from unit values. These relations are also illustrated in Figure 3.2.

Definition 2 (Primary semantic relations between concepts)**.**

Given two concepts in the same information dimension: $v_i^a, v_j^a \in V^a$:

- v_i^a equals v_j^a, denoted as $v_i^a = v_j^a$, iff $\mu(v_i^a) = \mu(v_j^a)$;
- v_j^a subsumes v_i^a, denoted as $v_i^a \triangleleft v_j^a$, iff $\mu(v_i^a) \subseteq \mu(v_j^a)$.
- v_i^a overlaps v_j^a, denoted as $v_i^a \infty v_j^a$, iff $\mu(v_i^a) \cap \mu(v_j^a) \neq \varnothing$, $\mu(v_i^a) \nsubseteq \mu(v_j^a)$, and $\mu(v_j^a) \nsubseteq \mu(v_i^a)$.
- v_i^a is adjacent to v_j^a, denoted as $v_i^a \parallel v_j^a$, iff $\mu(v_i^a) \cap \mu(v_j^a) = \varnothing$, and $\exists k \in \mu(v_i^a)$ such that $k + \varepsilon \in \mu(v_j^a)$, where ε is the smallest unit value increment.
- v_i^a is disjoint with v_j^a, denoted as $v_i^a \boxtimes v_j^a$, iff $\mu(v_i^a) \cap \mu(v_j^a) = \varnothing$;

The semantics in Definition 2, allow us to define further semantics through their composition by logical OR ($\vee$) or use of negation ($\neg$), as listed in Table 3.2.

Each of the semantic relations is interpreted in the context of the information dimension it is applied to. Illustrating with temperature, as shown in Figure 3.3, we might define

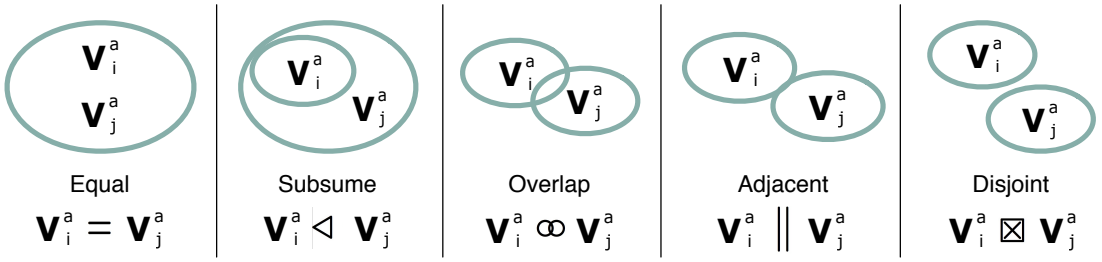


Figure 3.2: Relationships between concepts in an information dimension. Each circle represents the unit value set to which a concept maps.

Symbol	Name	Composition
$\neq$	Not equal	$\neg =$
$\trianglelefteq$	Subsumed by or equal to	$\triangleleft \vee =$
$\triangleright$	Subsumes	$\neg \trianglelefteq$
$\trianglerighteq$	Subsumes or equal to	$\neg \triangleleft$
$\not\infty$	Not overlapping	$\neg \infty$
$\nparallel$	Not adjacent	$\neg \parallel$
$\square$	Not disjoint	$\neg \boxtimes$

Table 3.2: Composed semantic relations between concepts.

four concepts: *freezing*, *cold*, *warm*, and *hot*. In this example, *i) cold* subsumes *freezing*; *ii) freezing* is disjoint with *warm* and *hot*, *iii) hot* is adjacent to *cold*; and, *iv) warm* overlaps both *cold* and *hot*.

The process is similar for other dimensions:

- In a dimension with a boolean property, there are only two concepts, corresponding to *true* and *false*, which are disjoint.
- In the time dimension, concepts such as *afternoon* or *weekend* are mapped to unit value sets containing time instances, which are comparable. For example, the concept *weekend* subsumes the concept *Saturday*.
- In the location dimension, concepts take the form of symbolic locations, with their relationships evaluated by projecting their associated coordinate sets onto a coordinate system and comparing them (SYDN10). For example, the concept *kitchen* may be adjacent to the concept *dining room*. We develop a worked example using location in Section 3.7.

3.3.4 Relation properties

To this point we have defined an approach to inferring basic semantic relations between concepts in an information dimension, solely through a domain expert mapping concepts to sets of unit values. This allows concepts in an information dimension to be compared in a uniform way, regardless of the type information being modelled, be it speed, location, or membership of an organisation.

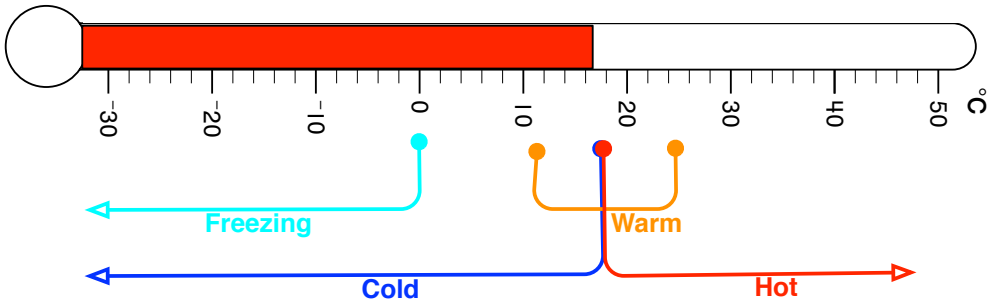


Figure 3.3: An illustration of the semantic relations between concepts in the temperature domain.

In this section we use these semantics to develop a general approach to reasoning over a modelled domain, uncovering the implicit knowledge in any single dimension. The starting point of this derivation process is the axiomatic properties of the five relationships introduced in Definition 2, presented in Definitions 3 – 7.

Definition 3 (Properties of the equals relation)**.**

Equal relationships are reflexive, transitive, and symmetric.

Definition 4 (Properties of the subsume relation)**.**

Subsume relationships are reflexive, transitive, and antisymmetric.

Definition 5 (Properties of the overlap relation)**.**

Overlap relationships are symmetric and irreflexive.

Definition 6 (Properties of the adjacent relation)**.**

Adjacent relationships are symmetric and irreflexive.

Definition 7 (Properties of the disjoint relation)**.**

Disjoint relationships are symmetric and irreflexive.

In addition to the intrinsic properties of the semantic relations, further knowledge can be derived by considering the interactions between multiple relations:

- If two concepts are adjacent, they are necessarily disjoint (directly from Definition 2).
- If two concepts are disjoint, then any pair of abstract values they subsume, one taken from each set, must also be disjoint (Lemma 1);
- If one concept is subsumed by two concepts, neither of which subsumes the other, then the latter two values overlap (Lemma 2).

- The *subsume*, *overlap*, and *disjoint* relationships form an exhaustive and mutually exclusive set of relationships between any two concepts; that is, any two concepts must have one and only one of these three relationships. If two of these relationships are known not to hold, the third must (Lemma 3).

Lemma 1 (Subsumed concepts inherit disjointness)**.**

$\forall v_i^a, v_j^a \in V^a$, if $v_i^a \boxtimes v_j^a$, then $\forall v_{i'}^a \trianglelefteq v_i^a$ and $\forall v_{j'}^a \trianglelefteq v_j^a$, $v_{i'}^a \boxtimes v_{j'}^a$.

Proof. From Definition 2,
$v_i^a \boxtimes v_j^a \Rightarrow \mu(v_i^a) \cap \mu(v_j^a) = \varnothing$ (1),
$v_{i'}^a \trianglelefteq v_i^a \Rightarrow \mu(v_{i'}^a) \subseteq \mu(v_i^a)$ (2),
$v_{j'}^a \trianglelefteq v_j^a \Rightarrow \mu(v_{j'}^a) \subseteq \mu(v_j^a)$ (3),
From (1), (2), (3) $\Rightarrow \mu(v_{i'}^a) \cap \mu(v_{j'}^a) = \varnothing$, and by Definition 2 $\Rightarrow v_{i'}^a \boxtimes v_{j'}^a$. □

Lemma 2 (Concepts sharing a subsumed concept overlap)**.**

For $v_i^a, v_j^a, v_k^a \in V^a$, if $\mu(v_k^a) \neq \varnothing$, $v_k^a \trianglelefteq v_i^a$, $v_k^a \trianglelefteq v_j^a$, $v_i^a \not\trianglelefteq v_j^a$, and $v_j^a \not\trianglelefteq v_i^a$, then $v_i^a \infty v_j^a$.

Proof. From Definition 2,
$v_k^a \trianglelefteq v_i^a \Rightarrow \mu(v_k^a) \subseteq \mu(v_i^a)$ (1),
$v_k^a \trianglelefteq v_j^a \Rightarrow \mu(v_k^a) \subseteq \mu(v_j^a)$ (2),
From (1) and (2) $\Rightarrow \mu(v_k^a) \subseteq \mu(v_i^a) \cap \mu(v_j^a)$ (3).
From (3) and $\mu(v_k^a) \neq \varnothing \Rightarrow \mu(v_i^a) \cap \mu(v_j^a) \neq \varnothing$ (4).
$v_i^a \not\trianglelefteq v_j^a \Rightarrow \mu(v_i^a) \not\subseteq \mu(v_j^a)$ (5),
$v_j^a \not\trianglelefteq v_i^a \Rightarrow \mu(v_j^a) \not\subseteq \mu(v_i^a)$ (6),
From (4), (5), (6), and by Definition 2 $\Rightarrow v_i^a \infty v_j^a$. □

Lemma 3 (Subsume, overlap, and disjoint relations are exhaustive)**.**

For $v_i^a, v_j^a \in V^a$,

(i) if $\neg(v_i^a \infty v_j^a)$, $v_j^a \not\trianglelefteq v_i^a$, and $v_i^a \not\trianglelefteq v_j^a$, then $v_i^a \boxtimes v_j^a$;

(ii) if $\neg(v_i^a \boxtimes v_j^a)$, $v_j^a \not\trianglelefteq v_i^a$, and $v_i^a \not\trianglelefteq v_j^a$, then $v_i^a \infty v_j^a$;

(iii) if $\neg(v_i^a \infty v_j^a)$, $\neg(v_i^a \boxtimes v_j^a)$, and $v_j^a \not\trianglelefteq v_i^a$, then $v_i^a \triangleleft v_j^a$;

(iv) if $\neg(v_i^a \infty v_j^a)$, $\neg(v_i^a \boxtimes v_j^a)$, and $v_i^a \not\trianglelefteq v_j^a$, then $v_j^a \triangleleft v_i^a$.

Definitions 3 – 7 and Lemmata 1 – 3 provide a sound basis for deriving further knowledge from that which is explicitly provided. The main benefit of this approach is that time can be saved in the knowledge specification process. A domain expert need only define the mapping from concepts to the unit value set for an information domain, or specify the primary relationships between concepts directly, in order for these derivation rules to be applicable.

A worked example demonstrating this process is given in Section 3.7.

3.4 Context model

Dourish (Dou04) describes *context* as a "slippery notion"; it is highly application specific, giving rise to definitions that are too restrictive to obtain consensus or too broad to be meaningful. Many have attempted to formalise this notion of context *as it applies to computer systems*. For example, Schilit et al. (SAW94) state:

> "[three] important aspects of context are: where you are, who you are with, and what resources are nearby ... [context] includes lighting, noise level, network connectivity, communication costs, communication bandwidth, and even the social situation; e.g., whether you are with your manager or with a co-worker".

while Dey et al. (DA00) define context more generally as:

> "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 the application themselves".

more broadly again, Yao et al. (YHGY06) and Loke (Lok04b) describe context as:

> "any information acquired from a system or an environment".

This final form most closely resembles the definition of an information space given by Newby et al. introduced in Section 3.3.1.

We define a context model that expands upon the concept model present above by *relating concepts* across dimensions of information. Thus, *Alices's living room* may be *hot*, *her kettle* switched *on*, *her car* may be *stationary*, and *3* people may be in *her house*.

Figure 3.4 illustrates these examples, and other relationships between dimensions of context. Concepts sharing an information domain are denoted by shared colour. Note that often the same knowledge can be expressed in multiple ways. For example, we can equivalently state that *Alice's house* **contains** *Alice's living room*, or *Alice's living room* is **contained by** *Alice's house*.

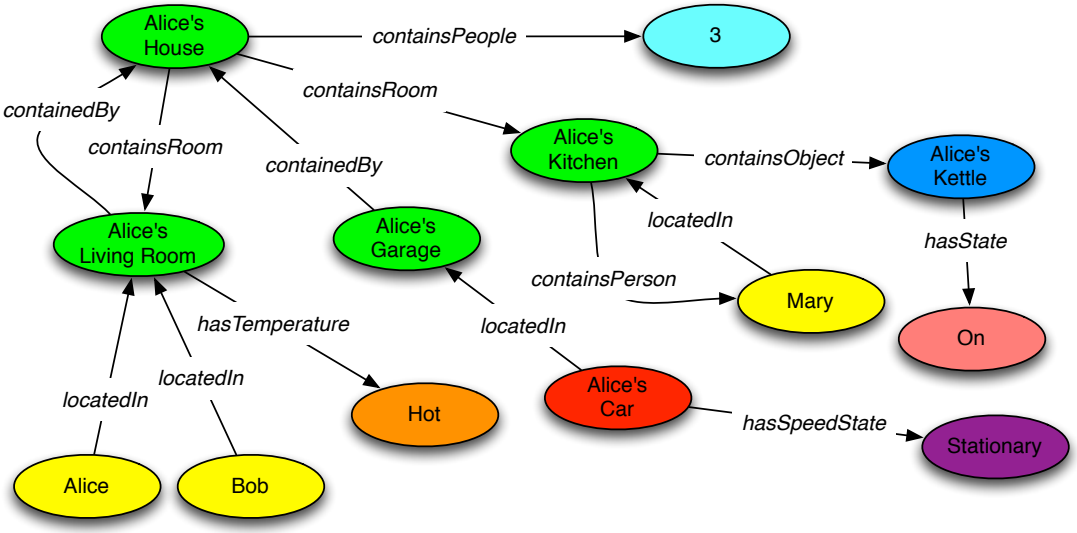


Figure 3.4: An illustration of relationships between information dimensions.

3.4.1 Context statements

Given two information domains, a *context statement* represents a relation between a pair of concepts, mapping to the cartesian square or product set of the information domains.

We represent a context statement as a triple, $[:s, :\mathbf{p}, :o]$, where $:s$ and $:o$, termed the *subject* and *object* respectively, are concepts in two information dimensions, and $:p$, termed the predicate, represents the named relation that exists between the two concepts. This naming scheme and notation is similar to RDF and its markup language *Notation 3* (BLC11), a mapping we explore further in Section 3.7.

We say that a (context) statement holds when the pair of concepts from two information domains are present in the product space of a predicate. For example, from the illustration given in Figure 3.4, the (subject, object) pair $(mary, alice'sKitchen)$ is present in the product space of predicate **locatedIn** and therefore the statement $[:mary, :\mathbf{locatedIn}, :alice'sKitchen]$ holds. Definition 8 formally defines a context statement.

Definition 8 (A context statement)**.**

Let S^a and O^a be two information domains and let predicate p be a relation mapping to their cartesian product $S^a \times O^a$, the set of ordered pairs of concepts (s, o) where $s \in S^a$ and $o \in O^a$.
We represent a context statement as $[:s, :\mathbf{p}, :o]$, where $s \in S^a$ and $o \in O^a$. We say the pair (s, o) entails the statement $[:s, :\mathbf{p}, :o]$ under p.

Further to this, Definition 9 defines four general types of predicate that support the inference of additional statements beyond the statement directly entailed by the presence of a given pair of concepts. Given a *subject-associative* predicate, if the subject of a statement is subsumed by any concepts, the set of corresponding statements with the subject substituted for each of these subsuming concepts also holds. Given an *object-associative* predicate, if the object of a statement is subsumed by any concepts, the set of corresponding statements with the object substituted for each of these subsuming concepts also holds. Given an *all-associative* predicate, if the subject or object are both subsumed by concepts, the set of corresponding statements with all combinations of the subject and object being substituted for their subsuming concepts also holds. Finally, given a *not-associative* predicate, no additional statements are directly inferred.

It is the responsibility of the model designer to define which, if any, of these types apply to a particular predicate and, further, to restrict the domains describing the subject, object or both where the predicate type applies. For example, the predicate **locatedIn** may be defined to be object-associative on the information domain describing locations and their containment relationships. This allows the statement [*:alice*, :**locatedIn**, *:alice's house*] to be inferred from the statement [*:alice*, :**locatedIn**, *:alice'sLounge*]. Similarly, the same predicate could be defined to be subject-associative on the information domain describing residency relationships of people in the house. For example, to support inference of the statement [*:houseResident*, :**locatedIn**, *:alice'sLounge*]. If **locatedIn** is defined as all-associative on these domains, the statement [*:houseResident*, :**locatedIn**, *:alice'sHouse*] can be inferred.

However it would be incorrect to apply a subject-associative **locatedIn** relation to familial relationships (for example, incorrectly inferring the statement [*:mary'sFamily*, :**locatedIn**, *:alice'sKitchen*]). Similarly, the **hasTemperature** relation is not subject-associative on the information domain describing locations and their containment relationships; that is, [*:alice'sKitchen*, :**hasTemperature**, *:hot*] does not support the inference of the statement [*:alice'sHouse*, :**hasTemperature**, *:hot*].

Definition 9 (Predicate types)**.**

Following on from Definition 8, let p^s, p^o, p^a, and p^n be subject-associative, object-associative, all-associative, and not-associative predicates respectively, such that:

- If $s \trianglelefteq s'$, $[:s, :\mathbf{p^s}, :o]$ entails context statement $[:s', :\mathbf{p^s}, :o]$, $\forall s' \in S^a$.
- If $o \trianglelefteq o'$, $[:s, :\mathbf{p^o}, :o]$ entails context statement $[:s, :\mathbf{p^o}, :o']$, $\forall o' \in O^a$.
- If $s \trianglelefteq s'$, $o \trianglelefteq o'$, $[:s, :\mathbf{p^a}, :o]$ entails context statement $[:s', :\mathbf{p^a}, :o']$, $\forall s' \in S^a$, $\forall o' \in O^a$.

p^n describes a predicate where no entailment rule applies.

3.4.2 Relationships between context statements

We build upon the relations we have defined to this point to study the semantic relations between context statements in the same manner as concepts. Returning to the example in Figure 3.3 we can see that the statement [*:edinburgh*, :**hasTemperature**, *:freezing*] is subsumed by the statement [*:edinburgh*, :**hasTemperature**, *:cold*], and is disjoint with the statements [*:edinburgh*, :**hasTemperature**, *:warm*] and [*:edinburgh*, :**hasTemperature**, *:hot*]. Similarly, the statement [*:dublin*, :**hasTemperature**, *:hot*] is disjoint with the statement [*:dublin*, :**hasTemperature**, *:freezing*], is adjacent to the statement [*:dublin*, :**hasTemperature**, *:cold*], and overlaps with the statement [*:dublin*, :**hasTemperature**, *:warm*]. Definition 10 provides these definitions formally.

These relationships can also be inferred directly from the relationships in their corresponding dimensions of information, details of which are relegated to Appendix A.

As with concepts, this provides an approach for deriving information from that which has been explicitly profiled or sensed from the environment. We demonstrate the utility of this approach further via a worked example in Section 3.7.

Definition 10 (Primary semantic relations between context statements)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$ that share the same relation p: $S^a \times O^a$,

- $[s_i, p, o_i]$ equals $[s_j, p, o_j]$, denoted $[s_i, p, o_i] = [s_j, p, o_j]$, iff $\forall (s,o) \in S^a \times O^a$ where (s,o) entails $[s_i, p, o_i]$, $[s_j, p, o_j]$ also holds; and $\forall (s,o) \in S^a \times O^a$ where (s,o) entails $[s_j, p, o_j]$, $[s_i, p, o_i]$ also holds;
- $[s_j, p, o_j]$ subsumes $[s_i, p, o_i]$, denoted $[s_i, p, o_i] \trianglelefteq [s_j, p, o_j]$, iff $\forall (s,o) \in S^a \times O^a$ where (s,o) entails $[s_i, p, o_i]$, $[s_j, p, o_j]$ also holds; and $\exists (s,o) \in S^a \times O^a$ where (s,o) entails $[s_j, p, o_j]$ such that $[s_i, p, o_i]$ does not also hold;
- $[s_i, p, o_i]$ overlaps with $[s_j, p, o_j]$, denoted $[s_i, p, o_i] \bowtie [s_j, p, o_j]$, iff $\exists (s,o), (s',o'), (s'',o'') \in S^a \times O^a$, (s,o) such that both $[s_i, p, o_i]$ and $[s_j, p, o_j]$ hold, (s',o') such that $[s_i, p, o_i]$ holds but $[s_j, p, o_j]$ does not, and (s'',o'') such that $[s_j, p, o_j]$ holds but $[s_i, p, o_i]$ does not;
- $[s_i, p, o_i]$ is adjacent to $[s_j, p, o_j]$, denoted $[s_i, p, o_i] \parallel [s_j, p, o_j]$, iff $\exists (s,o)$ and (s,o') or $(s',o) \in S^a \times O^a$, $s \parallel s'$, $o \parallel o'$ where (s,o) entails $[s_i, p, o_i]$ but not $[s_j, p, o_j]$ while either (s',o) or (s,o') entails $[s_j, p, o_j]$ but not $[s_i, p, o_i]$;
- $[s_i, p, o_i]$ is disjoint with $[s_j, p, o_j]$, denoted $[s_i, p, o_i] \boxtimes [s_j, p, o_j]$, iff $\nexists (s,o) \in S^a \times O^a$ where (s,o) entails both $[s_i, p, o_i]$ and $[s_j, p, o_j]$.

3.5 Temporal Qualification of Statements

Context statements are optionally associated with a time range that denotes the period of time over which the context statement should be interpreted as holding (SYD10), as described by Definition 11. Context statements without associated time ranges are assumed to always hold.

Definition 11 (A temporally qualified context statement)**.**

A temporally qualified context statement is a context statement with an associated time period over which the statement is assumed to hold. We represent it as $[:s, :\mathbf{p}, :o]^{t_s}_{t_e}$, where t_s and t_e respectively denote the start and end instants that the statement $[:s, :\mathbf{p}, :o]$ holds.

- A context statement $[:s, :\mathbf{p}, :o]^{t_s}_{t_e}$ holds if $[:s', :\mathbf{p}, :o']^{t_s}_{t_e}$ holds, and $s' \trianglelefteq s$, $o' \trianglelefteq o$.
- A context statement $[:s, :\mathbf{p}, :o]^{t'_s}_{t'_e}$ holds if $[:s, :\mathbf{p}, :o]^{t_s}_{t_e}$ holds, $t_s \leq t'_s < t_e$, and $t_s < t'_e \leq t_e$.

3.6 Representation and Propagation of State Information

Context models are instantiated by extracting information profiled from people or sampled by physical or virtual sensors (HIM05) and mapping it to the relational structures defined by experts. A standard mapping approach might adopt a boolean logic view of the world, conveying whether statements are true or false at a given point in time. However, it is often desirable to capture uncertainty as part of this model. We achieve this by associating statements with a numeric *confidence value*, as defined in Definition 12.

Definition 12 (An uncertain context statement)**.**

An uncertain context statement is a context statement with an associated degree of truth, representing the confidence that the statement reflects the true state of the sensed or profiled phenomenon. We represent it as $[s, p, o] : c$, where $c \in [0 : 1]$. 0 and 1 correspond to the boolean values of false and true respectively.

Definition 13 (Relations between statement confidence values)**.**

Given two context statements $[s_i, p_i, o_i] : c_i$ and $[s_j, p_j, o_j] : c_j$,

- $c_i = c_j$ if $[s_i, p_i, o_i] = [s_j, p_j, o_j]$;
- $c_j = 1 - c_i$ if predicate $p_j = p_i^{-1}$, is the inverse of p_i.

Definition 13 presents two trivial results that follow from Definition 12, namely, that two statements defined to be equal share the same confidence value, and two statements defined as the inverse of each other (a special case of disjointness) have confidence values that sum to 1.

Of the other relationships presented by Definition 10, only subsumption provides a useful means of associating derived statements with confidence values. The exact nature of this derivation depends on the relation and context being modelled. However three general propagation schemes are intuitively useful: *i)* boolean propagation; *ii)* additive propagation; and *iii)* fuzzy propagation. The remainder of this section gives examples of each.

3.6.1 Boolean State Propagation

Consider a kitchen containing several appliances, for example, a *Kettle*, *Fridge*, and *Stove*, all of which are types of *KitchenAppliance*. We assume that these concepts and relations are expressed using subsumption relationships.

At some point while *Alice* cooks dinner, the following statements may be present in the model:

$[:alice, :\textbf{interactsWith}, :Kettle] : true$
$[:alice, :\textbf{interactsWith}, :Stove] : true$

From which, given the defined subsumption relationships between the concepts and by Definition 10, we can derive:

$[:alice, :\textbf{interactsWith}, :KitchenAppliance] : true$

3.6.2 Additive State Propagation

Consider a simple *building* containing two disjoint floors *floorA*, and *floorB*, and each floor containing two disjoint rooms *A1*, *A2*, *B1*, *B2*. The building contains a tag based positioning system, supporting the placement of tagged people in the building, and the model defines subsumption relationships between the rooms, floor, and building concepts.

Assume that the output of the positioning system, allows us to realise the following three statements about *Bob*.

$[:bob, :\textbf{locatedIn}, :A1] : 0.6$
$[:bob, :\textbf{locatedIn}, :A2] : 0.2$
$[:bob, :\textbf{locatedIn}, :B3] : 0.2$

Given these statements, the defined subsumption relations between the concepts, and by Definition 10, we can add together the confidence values for subsumed concepts to firstly derive:

$[:bob, :\textbf{locatedIn}, :floorA] : 0.8$
$[:bob, :\textbf{locatedIn}, :floorB] : 0.2$

And, finally, by repeating this process we derive:

$[:bob, :\textbf{locatedIn}, :building] : 1$

3.6.3 Fuzzy Propagation

Returning to the previous example, we modify the building layout so that instead of being disjoint, rooms *A1* and *A2* overlap ($A1 \infty A2$). Here, we cannot use additive propagation to assign a confidence value to the statement $[:bob, :\textbf{locatedIn}, :floorA]$, as there is no way of telling whether the 0.2 confidence assigned to the statement $[:bob, :\textbf{locatedIn}, :A2]$ is covered by the 0.6 confidence in the statement $[:bob, :\textbf{locatedIn}, :A1]$.

To address this, we can instead use a fuzzy function (Zad73) – in particular, the fuzzy maximum, defined as $max(c_1, c_2)$ – to derive a confidence value, resulting in the statement:

$[:bob, :\textbf{locatedIn}, :floorA] : 0.6$

Finally, by repeating by using additive propagation to combine the confidences of the disjoint concepts *floorA* and *floorB*, we derive:

$[:bob, :\textbf{locatedIn}, :building] : 0.8$

We discuss the interpretation of a context statement's "state" further in the next chapter, and give an example mapping from raw location sensor data to conceptual room values in Section 6.2.3.

3.7 Mapping to Semantic Web Technologies: An Example

In this section we present an example with a dual purpose: Firstly, to provide a worked example of using the conceptual model, mapping the floor plan of a home to a set of concepts and demonstrating the derivation of information from that which is explicitly specified using the definitions and lemmas above. Secondly, to demonstrate how the conceptual model is realised using Semantic Web technologies, by providing an example vocabulary and a set of reasoning rules that operate over it, augmenting OWL's native axioms to implement the model.

The implementation uses the Named Graphs for Jena (NG4J) software library (BCW05), an extension of the Jena Semantic Web framework (McB02). NG4J supports the modelling of Named Graphs, a concept for identifying statements or groups of statements which, as we will discuss, allow us to easily model temporal quantification and relations between context predicates. Given our need to infer relations between statements, and the lack of a suitable reasoner for doing so, the original implementation presented in Ye et al. (YSD11a) approximated the desired functionality by bolting-on a number of custom functions to the general purpose rule engine feature of Jena.

After publication, we extended the NG4J library to integrate it with the Jess (jes11) rule engine, allowing us to develop a rule engine for OWL where Named Graphs are treated as a first-class feature. This provides us with a cleaner separation of concerns, and an easier, natural way of expressing the rules that realise our reasoning framework, simplifying the process of inspecting and expressing relations between context statements and their associated constraints.

As noted above, these particular tools, and, more generally, the Semantic Web formalism are but one approach to realising the conceptual modelling framework described in this

chapter. Any interested party may realise the same model using other formalisms or software tools of their choosing.

3.7.1 Model Overview

Figure 3.5 depicts the floor plan of a small residential house that we use to demonstrate the conceptual modelling approach described in this chapter. The house consists of a kitchen, dining room, lounge, bathroom, and bedroom split across two floors, with a connecting staircase (excluded from the model). Even for a simple layout such as this, it requires a considerable amount of knowledge engineering effort to manually describe each of the relations between concepts in the building ($O(n^2)$ for n concepts).

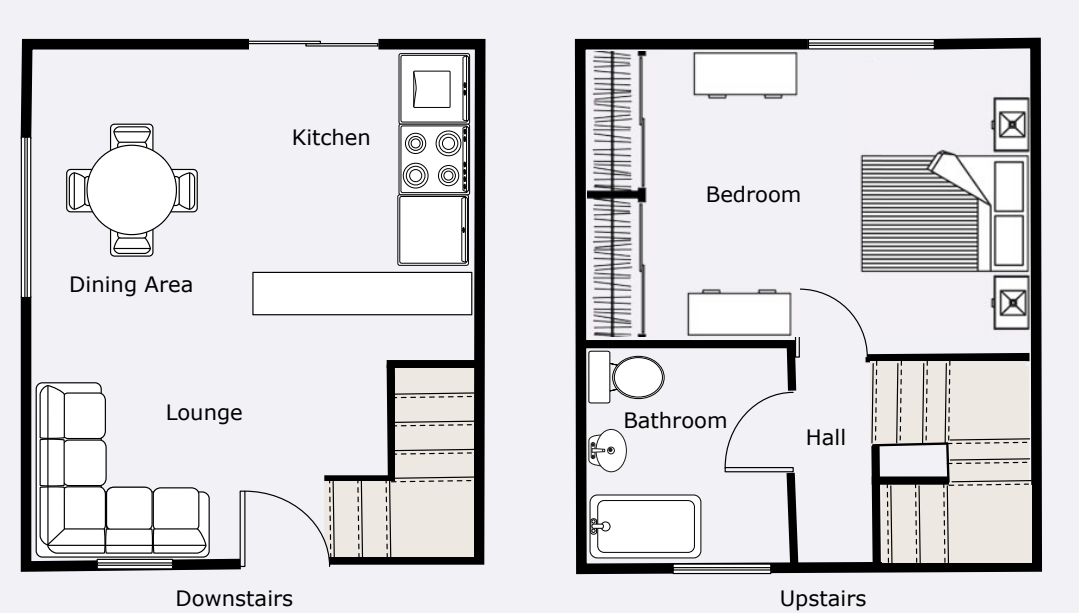


Figure 3.5: An example layout of a one bedroom house

Given the layout shown in Figure 3.5, we now construct a model of the house and illustrate that: *i)* our ontology model allows developers to specify a smaller percentage of the total knowledge, deriving the remainder automatically, and *ii)* the common semantics across contexts can be used to automate the detection of any inconsistent knowledge that is specified.

There are two possible approaches to begin this task: By defining directly a minimal set of relations between concepts, or, alternatively, by defining the mapping from the unit value set to concepts and deriving all relations automatically. We illustrate both approaches below, and then proceed by describing the implementation supporting each.

Approach A: Manually Specifying Relations Table 3.3 lists the set of 12 relations between concepts – consisting only of direct subsumption and adjacency relationships – that must be specified in order that the remaining relations can be inferred automatically. These 25 additional relationships are shown in Table 3.4, along with reference to the Definitions and Lemmata that support their derivation.

Approach B: Mapping from Unit Values Approach A demonstrates that the majority of the relational knowledge between concepts can be inferred from an initially specified 12 relations. These same 12 relations may also be inferred by providing a mapping from unit values to concepts. For simplicity of illustration, consider a 2D cartesian coordinate system projected over the house, from which the regions of space associated with each of the concepts can be defined. This is illustrated in Figure 3.6 (the two floors and house as a whole are not shown).

By defining each of the concepts as the 2D region they represent, Definition 2 provides the means to infer each of the relationships in Table 3.3, and from this the relationships shown in Table 3.4 follow as before.

3.7.2 Implementing the Concept Model

Using OWL, we define properties representing the common semantic relations (and properties derived therefrom) that two concepts in one dimension may share, as described in Section 3.3.1, realised as the *Ontonym domain ontology* (SKDN09).

OWL supports the definition of Reflexive (**owl:ReflexiveProperty**), Irreflex-

No.	Relation
1.	*Upstairs* $\triangleleft$ *Building*
2.	*Downstairs* $\triangleleft$ *Building*
3.	*Hall* $\triangleleft$ *Upstairs*
4.	*Bedroom* $\triangleleft$ *Upstairs*
5.	*Bathroom* $\triangleleft$ *Upstairs*
6.	*Lounge* $\triangleleft$ *Downstairs*
7.	*Kitchen* $\triangleleft$ *Downstairs*
8.	*DiningArea* $\triangleleft$ *Lounge*
9.	*DiningArea* $\triangleleft$ *Kitchen*
10.	*Bathroom* $\parallel$ *Hall*
11.	*Bedroom* $\parallel$ *Hall*
12.	*Upstairs* $\parallel$ *Downstairs*

Table 3.3: The minimal relation set required to derive the complete relational model.

No.	Relation	Derivation Path
13.	*DiningArea* $\triangleleft$ *Downstairs*	(from 6, 8, by Definition 4)
14.	*Hall* $\triangleleft$ *House*	(from 1, 3, by Definition 4)
15.	*Bedroom* $\triangleleft$ *House*	(from 1, 4, by Definition 4)
16.	*Bathroom* $\triangleleft$ *House*	(from 1, 5, by Definition 4)
17.	*Kitchen* $\triangleleft$ *House*	(from 2, 7, by Definition 4)
18.	*Lounge* $\triangleleft$ *House*	(from 2, 6, by Definition 4)
19.	*DiningArea* $\triangleleft$ *House*	(from 8, 17, by Definition 4)
20.	*Hall* $\parallel$ *Bathroom*	(from 10, by Definition 6)
21.	*Hall* $\parallel$ *Bedroom*	(from 11, by Definition 6)
22.	*Downstairs* $\parallel$ *Upstairs*	(from 12, by Definition 6)
23.	*Kitchen* ∞ *Lounge*	(from 8, 9, by Lemma 2)
24.	*Lounge* ∞ *Kitchen*	(from 23, by Definition 5)
25.	*Bedroom* $\boxtimes$ *Hall*	(from 1-24, by Lemma 3)
26.	*Hall* $\boxtimes$ *Bedroom*	(from 25, by Definition 7)
27.	*Bedroom* $\boxtimes$ *Bathroom*	(from 1-26, by Lemma 3)
28.	*Bathroom* $\boxtimes$ *Bedroom*	(from 27, by Definition 7)
29.	*Hall* $\boxtimes$ *Bathroom*	(from 1-28, by Lemma 3)
30.	*Bathroom* $\boxtimes$ *Hall*	(from 29, by Definition 7)
31.	*Upstairs* $\boxtimes$ *Downstairs*	(from 1-30, by Lemma 3)
32.	*Downstairs* $\boxtimes$ *Upstairs*	(from 32, by Definition 7)
33.	*Hall* $\boxtimes$ *Downstairs*	(from 3, 31, by Lemma 1)
34.	*Bedroom* $\boxtimes$ *Downstairs*	(from 4, 31, by Lemma 1)
35.	*Bathroom* $\boxtimes$ *Downstairs*	(from 5, 31, by Lemma 1)
36.	*Kitchen* $\boxtimes$ *Upstairs*	(from 7, 32, by Lemma 1)
37.	*Lounge* $\boxtimes$ *Upstairs*	(from 6, 32, by Lemma 1)
38.	*DiningArea* $\boxtimes$ *Upstairs*	(from 13, 32, by Lemma 1)

Table 3.4: The set of inferred relations and their derivation paths.

ive (**owl:IrreflexiveProperty**), Transitive (**owl:TransitiveProperty**), Symmetric (**owl:SymmetricProperty**), and Antisymmetric (**owl:AntisymmetricProperty**) properties, covering the axiomatic definitions 3 – 7. Listing 3.1, encoded using *Notation 3*, an RDF shorthand notation (BLC11), shows some of these definitions. Listing 3.1 also references a set of namespaces (unique pointers to RDF and OWL vocabulary definitions) used in the document, for example *owl:*, *rdf:*, and *rdfs:*. The namespace we use for the common semantic property definitions is *domain:*, while the namespace *g:* is used to denote graphs—groupings of statements. Namespace definitions are omitted from subsequent listings for brevity.

As mentioned at the beginning of this section, we use the Jess (jes11) rule engine as part of our implementation. In addition to the rules defined within our context model, the developed ruleset includes those defined by the OWL specification. For example, the rule associated with the **owl:SymmetricProperty** property is shown in

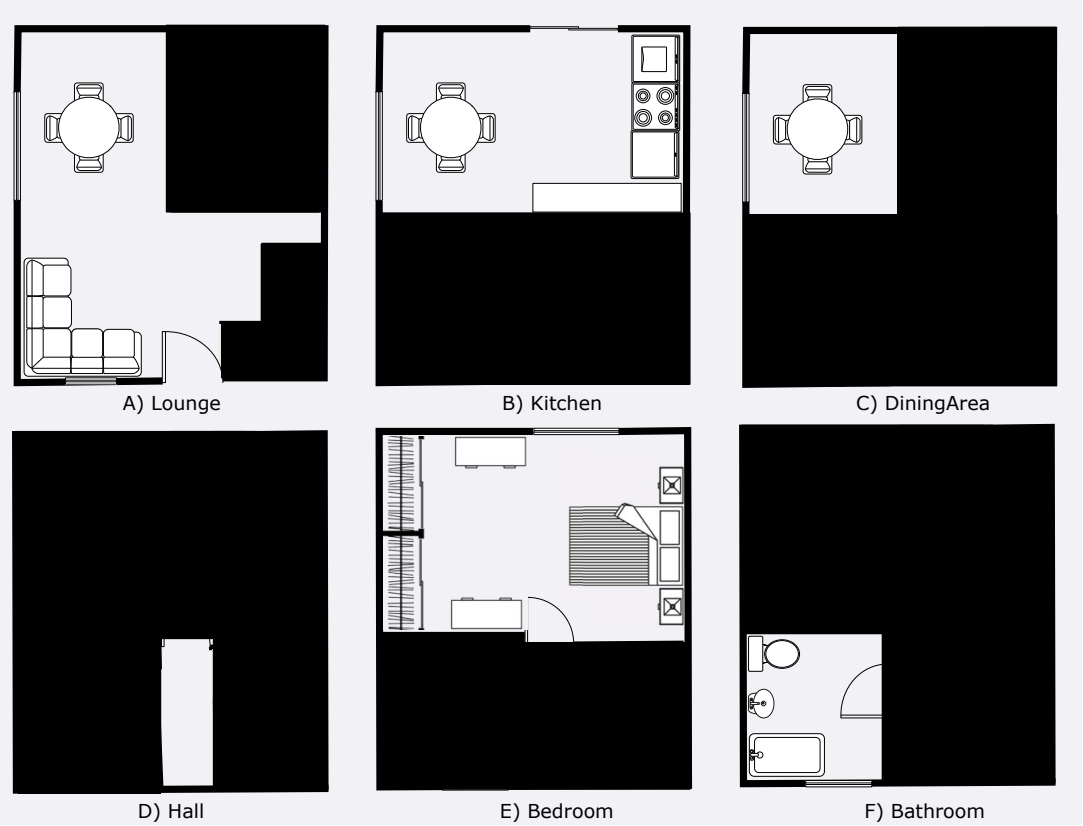


Figure 3.6: Shaded regions representing regions of space, mapped to each concept.

```
1  @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
2  @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
3  @prefix owl: <http://www.w3.org/2002/07/owl#> .
4  @prefix domain: <http://ontonym.org/2011/01/domain#> .
5  @prefix g: <_:graph> .
6
7  g:definitions{
8    domain:equalTo
9       a owl:ReflexiveProperty, owl:TransitiveProperty, owl:SymmetricProperty.
10   domain:subsumedBy
11      a owl:ReflexiveProperty, owl:TransitiveProperty, owl:AntisymmetricProperty.
12   domain:overlapsWith
13      a owl:IrreflexiveProperty, owl:SymmetricProperty.
14   domain:adjacentTo
15      a owl:IrreflexiveProperty, owl:SymmetricProperty.
16   domain:disjointWith
17      a owl:IrreflexiveProperty, owl:SymmetricProperty.
18 }
```

Listing 3.1: Defining common semantic properties using OWL.

Listing 3.2. Lines 2 and 3 on the left-hand-side of the inference rules respectively match any property, **?p** defined as being symmetric, and any statement of the form $[?s, \textbf{?p}, ?o]$ using the property, while lines 5-7 on the right-hand-side of the inference rule generate a new statement taking the form $[?o, \textbf{?p}, ?s]$ and places it in a graph with the identifier *_:inf*.

```
(defrule prp-symp (declare (salience 100))
 (logan-quad (graph ?g)(subject ?p)(predicate "rdf:type")(object "owl:SymmetricProperty"))
 (logan-quad (graph ?h)(subject ?s)(predicate ?p)(object ?o))
  =>
 (assert
  (logan-quad (graph "_:inf")(subject ?o)(predicate ?p)(object ?s))
 ))
```

Listing 3.2: Defining the effect of the **owl:symmetric** property using Jess.

Going beyond what we can specify using "pure" OWL-DL (the subset of OWL based on a decidable Description Logics fragment), we realise the more complex properties specific to our model from Lemmata 1–3 using the rules shown in Listing 3.3. For brevity, the listings present only the core part of the rule, omitting some features, such as checking that statement subjects and objects share the same concept type. In particular, note that the rule associated with Lemma 3 is triggered only after all other deductions are made, to prevent incorrect inference of disjointness.

```
;; Subsumed concepts inherit disjointness (from Lemma 6)
(defrule disjoint-inherit (declare (salience 100))
 (logan-quad (graph ?)(subject ?c1)(predicate "domain:disjointWith")(object ?c2))
 (logan-quad (graph ?)(subject ?c11) (predicate "domain:subsumedBy")(object ?c1))
 (logan-quad (graph ?)(subject ?c22)(predicate "domain:subsumedBy")(object ?c2))
=>
 (assert
   (logan-quad (graph "_:inf")(subject ?c11)(predicate "domain:disjointWith")(object ?c22))
   (logan-quad (graph "_:inf")(subject ?c11)(predicate "domain:disjointWith")(object ?c2))
   (logan-quad (graph "_:inf")(subject ?c22)(predicate "domain:disjointWith")(object ?c1))
 ))

;; Concepts sharing a subsumed concept overlap (from Lemma 7).
(defrule derive-overlap (declare (salience 90))
 (logan-quad (graph ?)(subject ?c1)(predicate "domain:subsumedBy")(object ?c2))
 (logan-quad (graph ?)(subject ?c1) (predicate "domain:subsumedBy")(object ?c3))
 (test (neq ?c2 ?c3))
 (not (logan-quad (graph ?)(subject ?c2)(predicate "domain:subsumedBy")(object ?c3)))
 (not (logan-quad (graph ?)(subject ?c3)(predicate "domain:subsumedBy")(object ?c2)))
=>
 (assert
  (logan-quad (graph "_:inf")(subject ?c2)(predicate "domain:overlapsWith")(object ?c3))
 ))

;; Subsume, overlap, and disjoint relations are exhaustive (from Lemma 8)
(defrule derive-exhaustive (declare (salience 10))
  (not (logan-quad (graph ?)(subject ?c1)(predicate "domain:subsumedBy")(object ?c2)))
  (not (logan-quad (graph ?)(subject ?c2)(predicate "domain:subsumedBy")(object ?c1)))
  (not (logan-quad (graph ?)(subject ?c1)(predicate "domain:overlapsWith")(object ?c2)))
  (not (logan-quad (graph ?)(subject ?c1)(predicate "domain:equalTo")(object ?c2)))
=>
 (assert
  (logan-quad (graph "_:inf")(subject ?c1)(predicate "domain:disjointWith")(object ?c2))
 ))
```

Listing 3.3: The Jess implementation of Lemmata 1–3

Each dimension of information, be it location, temperature, or heart rate has its own structure and relationships. In order to exploit the common semantic relation-

ships defined and implemented above, a domain ontology developer need only define relevant properties as being *sub-properties* of the corresponding Ontonym properties. For example, the developer of a location ontology with properties **loc:contains** and **loc:nextTo** need only add the statements [*loc:contains*, **rdfs:subPropertyOf**, *domain:subsumedBy*] and [*loc:nextTo*, **rdfs:subPropertyOf**, *domain:adjacent*] to their ontology to ascribe the meanings of the subsumed and adjacent relations to the **loc:contains**, and **loc:nextTo** properties respectively.

Next we illustrate how relationships between locations are inferred from a mapping to unit values, using Definition 2. For simplicity, assume that a location's area is defined as a regular two dimensional rectangle, with lower and upper bounds denoting the position of the lower left and upper right corners of the area in a Cartesian coordinate system. Four properties, **loc:lowerLeftX**, **loc:lowerLeftY**, **loc:upperRightX** and **loc:upperRightY** are defined for this purpose. Listing 3.4 gives an example definition of the concept *DiningArea*, specified with coordinates $(0.0, 10.3), (11.1, 21.4)$.

```
:DiningArea a loc:SymbolicLocation ;
            loc:lowerLeftX [ a muo:QualityValue ;
                             muo:numericalValue "0.0"^^xsd:double ;
                             muo:measuredIn ucum:metres ] ;
            loc:lowerLeftY [ a muo:QualityValue ;
                             muo:numericalValue "10.3"^^xsd:double ;
                             muo:measuredIn ucum:metres ] ;
            loc:upperRightX [ a muo:QualityValue ;
                              muo:numericalValue "11.1"^^xsd:double ;
                              muo:measuredIn ucum:metres ] ;
            loc:upperRightY [ a muo:QualityValue ;
                              muo:numericalValue "21.4"^^xsd:double ;
                              muo:measuredIn ucum:metres ] .
```

Listing 3.4: Defining the *DiningArea* concept from the location example.

With the other regions of the house similarly defined, we write a mapping function that uses the unit values to infer the semantic relations between concepts. For example, Listing 3.5 shows how to write a rule to map the containment relationship between two abstract locations to the **loc:contains** relationship, which is defined as a sub-property of the **domain:subsumes** relationship. Mappings of the other generic relationships are similarly realised either through defining a rule or writing procedural code, allowing us to infer the relations described in Table 3.3 and Table 3.4.

The above examples demonstrate two approaches to defining common semantics in individual dimensions of information, which link the top-level ontology with domain or application specific ontologies. One of the advantages of defining the common semantic relationships between domain concepts is that we can use generic rules to derive new knowledge; this reduces the knowledge engineering effort required of developers.

We remarked earlier that there exists $O(n^2)$ relationships among n concepts, given

```
(defrule location-contains (declare (salience 50))
  (logan-quad (graph ?)(subject ?a)(predicate "loc:lowerLeftX")(object ?allx))
  (logan-quad (graph ?)(subject ?a)(predicate "loc:lowerLeftY")(object ?ally))
  (logan-quad (graph ?)(subject ?a)(predicate "loc:upperRightX")(object ?aurx))
  (logan-quad (graph ?)(subject ?a)(predicate "loc:upperRightY")(object ?aury))
  (logan-quad (graph ?)(subject ?b)(predicate "loc:lowerLeftX")(object ?bllx))
  (logan-quad (graph ?)(subject ?b)(predicate "loc:lowerLeftY")(object ?blly))
  (logan-quad (graph ?)(subject ?b)(predicate "loc:upperRightX")(object ?burx))
  (logan-quad (graph ?)(subject ?b)(predicate "loc:upperRightY")(object ?bury))
  (logan-quad (graph ?)(subject ?allx)(predicate "muo:numericalValue")(object ?allxv))
  (logan-quad (graph ?)(subject ?ally)(predicate "muo:numericalValue")(object ?allyv))
  (logan-quad (graph ?)(subject ?aurx)(predicate "muo:numericalValue")(object ?aurxv))
  (logan-quad (graph ?)(subject ?aury)(predicate "muo:numericalValue")(object ?auryv))
  (logan-quad (graph ?)(subject ?bllx)(predicate "muo:numericalValue")(object ?bllxv))
  (logan-quad (graph ?)(subject ?blly)(predicate "muo:numericalValue")(object ?bllyv))
  (logan-quad (graph ?)(subject ?burx)(predicate "muo:numericalValue")(object ?burxv))
  (logan-quad (graph ?)(subject ?bury)(predicate "muo:numericalValue")(object ?buryv))
  (>= ?allxv ?ballxv)
  (>= ?allyv ?ballyv)
  (<= ?aurxv ?baurxv)
  (<= ?aurxv ?baurxv)
 =>
 (assert
  (logan-quad (graph "_:inf")(subject ?b)(predicate "loc:contains")(object ?a))
 ))
```

Listing 3.5: Inferring containment from unit value-based location definitions.

that any two concepts can either be equal, disjoint, overlap, or share a subsumption relationship, without accounting for other relationships that might exist. Using the approach we outline here, ontology developers need only *i)* define adjacency and immediate subsumption relationships between concepts in the model, or *ii)* provide the mapping to unit values with associated set of evaluation functions, in order for the relationships defined above to be derived automatically.

3.7.3 Implementing the Context Model

Definition 8 defines a context statement as a triple consisting of a subject, a predicate, and object, taking the form $[s, p, o]$, where the subject and object are concepts in two domains. Definition 11 later extends this model with temporal qualification.

We realise the entailment of context statements (Definition 9) and temporally qualified context statements (Definition 11), based on the occurrence of any context statements they subsume. Listing 3.6 gives the implementation of the rule for subject-associative predicates; the others follow similarly. The left-hand side of the rule matches any statement where the subject has a concept that subsumes it, where a relation between the subsuming concepts does not already exist, where the relation is declared to be subject-associative on the subject type. When matched, the rule right-hand-side generates a new identifier for the graph that will contain the inferred statement, and annotates the graph with the same period of temporal qualification as the statement it subsumes.

```
;; Entailment of subject-associative context statements
;; N.B. Inferred statements are added to new graphs, not the general inference graph

(defrule definition-nine (declare (salience 80))
 (logan-quad (graph ?g1)(subject ?s)(predicate ?p)(object ?o))
 (logan-quad (graph ?)(subject ?s)(predicate "rdf:type") (object ?x))
 (logan-quad (graph ?)(subject ?s)(predicate "domain:subsumedByEq")(object ?s1))
 (logan-quad (graph ?)(subject ?p)(predicate "domain:subjectAssociativeOnType")
  (object ?x))
 (logan-quad (graph ?g1)(subject ?g1)(predicate "time:temporalRelevance")(object ?time))
 (not
  (and (logan-quad (graph ?g2)(subject ?s1)(predicate ?p)(object ?o))
       (logan-quad (graph ?g2)(subject ?g2)(predicate "time:temporalRelevance")
        (object ?time))
  )
 )
=>
 (bind ?newGraph (str-cat "urn://" (UUID.randomUUID)))
 (assert
  (logan-graphname (graph ?newGraph))
  (logan-quad (graph ?newGraph)(subject ?s1)(predicate ?p)(object ?o))
  (logan-quad (graph ?newGraph)(subject ?newGraph)
     (predicate "time:temporalRelevance")(object ?time))
 ))
```

Listing 3.6: Entailing subject-associative context statements from the existence of statements they subsume.

To illustrate the rule's application, consider that it supports the inference of the statement

$$[\textit{ex:houseOccupant}, \textbf{position:locatedIn}, \textit{ex:downstairs}]^{2014-01-08T10:57:43Z}_{2014-01-08T10:59:43Z}$$

from the presence of the statement

$$[\textit{ex:bob}, \textbf{position:locatedIn}, \textit{ex:lounge}]^{2014-01-08T10:57:43Z}_{2014-01-08T10:59:43Z}$$

assuming that the concept *ex:houseOccupant* is defined as subsuming or being equal to *ex:bob*, and the concept *ex:lounge* is defined as being subsumed by or equal to *ex:downstairs*.

We use Named Graphs to express relations between context statements as described in Definition 10. Named Graphs provide a pointer to a statement or a set of statements, and we express a relation between two (or more) statements as a relation between the graphs that contain them. For example if a named graph *g:G1* contains context statements *S1* and *S2*, and a named graph *g:G2* contains statement *S3*, we can express that *S3* is disjoint with both *S1* and *S2* by writing [*g:G1*, **domain:disjointWith**, *g:G2*]. We interpret the presence of this statement in the model as the pair of relations $(S1 \boxtimes S3, S2 \boxtimes S3)$.

Statements may be present in multiple graphs if multiple relationships need to be described. For example, Listing 3.7 models the scenario where Bob is reported in dining area (graph *g:G1*). From this, we infer that he is in part of the lounge and kitchen (by their overlapping definitions), and in the house ($G2 \triangleright G1$).

```
g:G1{
     ex:bob position:locatedIn ex:diningArea .
    } .
g:G2{
     ex:bob position:locatedIn ex:kitchen .
     ex:bob position:locatedIn ex:lounge .
     ex:bob position:locatedIn ex:house .
    } .
g:G3{
     g:G2 domain:subsumes g:G1 .
    } .
```

Listing 3.7: Expressing relations between context statements.

We cater for temporal constraints associated with context statements by only inferring relationships between them if their durations overlap. To illustrate, consider a model that contains two context statements that describe Bob's presence in the bedroom (S_{be}) and the bathroom (S_{ba}). With no frame of temporal reference, these statements are disjoint ($S_{be} \boxtimes S_{ba}$), however if both statements hold during non-overlapping time periods, then there is no conflict.

To illustrate this, Listing 3.8 extends the previous example, adding a temporal constraint to *g:G1* and introducing temporally qualified graphs placing bob in the bathroom and bedroom. While the statement placing Bob in the *bathroom* temporally overlaps with the statement placing him in the dining area (*g:G7* and *g:G8*), the statement placing him in the bedroom does not (*g:G9*). Therefore, only the first two statements conflict (*g:G10*).

3.8 Discussion

To conclude this chapter, we discuss the utility of the top-level ontology model we have presented, discussing both conceptual aspects and practical considerations for adopting the model for the development of pervasive systems.

3.8.1 Formal Relations

The concept model proposed in this chapter takes set theory as its basis, using it to formulate a mapping between concepts and unit values in an information domain.

The model partitions the problem of how to transform raw sensor data into conceptually higher-level statements that can exploited by reasoners and applications as a set of uniform, clearly defined steps. No matter the information domain being modelled, the application of each step is governed by general purpose rules that inform the

```
g:G5{
     ex:bob position:locatedIn ex:bedroom .
    } .
g:G6{
     ex:bob position:locatedIn ex:bathroom .
    } .
g:G7{
     :G1 time:validDuring [a owlt:Interval ;
                           owlt:hasBeginning [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T10:00:00Z"] ;
                           owlt:hasEnd       [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T10:30:00Z"] .
   } .
g:G8{
     :G5 time:validDuring [a owlt:Interval ;
                           owlt:hasBeginning [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T10:05:00Z"] ;
                           owlt:hasEnd       [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T10:15:00Z"] .
   } .
g:G9{
     :G6 time:validDuring [a owlt:Interval ;
                           owlt:hasBeginning [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T12:40:00Z"] ;
                           owlt:hasEnd       [a owlt:Instant ;
                                              owlt:inXSDDateTime "2014-01-08T12:59:00Z"] .
   } .
g:G10 {
    :G1 domain:disjointWith :G5 .
    } .
```

Listing 3.8: Conflict between temporally qualified context statements.

transformation or augmentation of knowledge from one step to the next.

In implementing the model we utilise the constructs of OWL-DL (a particular decidable subset of Description Logics), and extend this functionality, using the Jess rule engine, to support the entailment of context statements based on the more complex definitions, where such inference procedures cannot be represented using OWL alone. We also use Jess to support reasoning over the named graphs, supporting temporal constraints, which is non standard functionality.

Unlike other general purpose information space models or data stores where there is no strict governance to regulate how stored information relates, a key result of this approach is the automated, traceable association of knowledge across all the steps in a sound reasoning schema.

3.8.2 A Rich Semantic Model

Five relations are proposed as primitives for modelling structural relations between concepts, each of which is formally defined against the unit value sets they relate. This core set of relations serve as primitives for associating context statements; this is again formally defined against concept relations, allowing such deductions to be made directly

from the conceptual mappings to unit values.

The semantics of these statements are extended in three ways. Firstly, the introduction of temporal qualification allows context statements to be bounded by time, and derivative reasoning likewise bounded. Secondly, the use of temporal semantics allows the relationships between temporally qualified statements to be inferred and represented. Thirdly, the introduction of degrees of truth describes the extent to which a statement reflects the reality of the world it models (see Section 3.6 for an example of how uncertainty can be assigned), and derivative reasoning allows confidence values to be propagated and assigned to related context statements using a variety of schemes as appropriate—all of which enhances the expressiveness of the model and the quality of knowledge it can represent.

3.8.3 Uniformity of Approach

The same set of semantic relations are applied both at the concept and statement levels. This uniformity not only simplifies the modelling of an information space, supporting the automatic derivation of knowledge from that which is explicitly specified, but also has utility in the debugging of pervasive systems. That is, the formal semantic model provides a kind of provenance. The existence of two disjoint statements in the model (where the statements are disjoint and their temporal bounds overlap) means that the model can be determined to be in an inconsistent state. Consequently, we can pinpoint the specific context statements that are the root cause of the inconsistency.

Resolution of such inconsistencies are outside the scope of the model, however, their identification provides useful feedback to ontology engineers, or can serve as input to tools or helper agents capable of analysing or resolving such conflicts (YMC+08).

3.8.4 Extension to Higher Level Activity Concepts

Although not presented in this thesis, Ye et al. (YSD11a) describes an extension of this context model to represent activities, by reapplying the same set-theoretic approach once more. Here, activities are treated as another dimension of information, whose unit values are not directly measured, but take the form of a set of descriptions or terms that represent aspects of the activity. For example the activity of watching television may be defined on descriptions of a person being in the same room as a television, sitting in a seat, and the television being switched on (Roy et al. (RGD10)). Such descriptions can be realised through associating activities with a derivation rule based on the presence of a collection of context statements that describe it, as illustrated in Listing 3.9.

```
;; One approach to deriving activities from context statements

(defrule definition-nine (declare (salience 80))
 (logan-quad (graph ?)(subject ?person)(predicate "pos:locatedIn")(object ?room))
 (logan-quad (graph ?)(subject ?tv)(predicate "pos:locatedIn")(object ?room))
 (logan-quad (graph ?)(subject ?tv)(predicate "rdf:type")(object "device:Television"))
 (logan-quad (graph ?)(subject ?tv)(predicate "state:hasState")(object "on"))
 (logan-quad (graph ?)(subject ?person)(predicate "state:isSitting")(object "true"))
=>
 (bind ?newGraph (str-cat "urn://" (UUID.randomUUID)))
 (assert
  (logan-graphname (graph ?newGraph))
  (logan-quad (graph ?newGraph)(subject ?person)(predicate "situation:engagingIn")
     (object "situation:watchingTelevision"))
 ))
```

Listing 3.9: One approach to infer the activity of watching television through a context-statement-based derivation rule.

One advantage of modelling activities as just another information dimension is that it becomes possible to infer knowledge about the relationships between activities from their context statements. Intuitively, if one activity subsumes another, then at points in time when we consider a person to be engaging in the former activity then they must also be engaging in the latter (for example, $watchingTV \triangleleft leisureActivity$), or, if two activities are disjoint, we can deduce that a person cannot engage in both simultaneously (for example, $eating \boxtimes sleeping$).

The following chapters provide an alternative to the derivation-rule-based approach to activity specification, primarily due to the difficulty in manually specifying such rules when uncertainty is present. However, this does not preclude the manual specification of direct subsumption relationships between activities, allowing other relationships to be automatically derived. Consequently, the output of the work described on situation recognition in the following chapters can be used to generate statements about which activities a subject engages in. Combined with the specification of inter-activity relationships, these statements can be used as a basis from which to derive other statements, in the same manner as all other information dimensions.

3.8.5 Extensible Derivation Rule Model

By augmenting the core model with additional rules, the common semantic relationships can be straightforwardly used to derive new knowledge across dimensions of information. That is, using the relations between concepts in one dimension as an analog for a relation in another. For example, given a collection of statically positioned objects, we can use the semantics that describe the spatial relations between their physical locations to make disjoint any context statements that describe simultaneous interactions

between a person and objects that are not co-located. That is, inconsistencies in the set of physical object interactions declared in a model can be inferred directly from the semantic relationships shared by their locations. This is illustrated in Listing 3.10: The left-hand-side of the rule matches any two statements describing object interactions, where the objects are located in separate rooms, and the locations are declared to be disjoint. If matched, the right-hand-side declares the two statements describing the interactions to be disjoint (by placing each in a Named Graph, and declaring the Named Graphs to be disjoint).

```
;;; Deriving information across information dimensions to detect inconsistencies

(defrule definition-nine (declare (salience 80))
 (logan-quad (graph ?)(subject ?o1)(predicate "pos:locatedIn")(object ?l1))
 (logan-quad (graph ?)(subject ?o1)(predicate "pos:locatedIn")(object ?l2))
 (logan-quad (graph ?)(subject ?l1)(predicate "domain:disjointWith")(object ?l2))
 (logan-quad (graph ?g1)(subject ?person)(predicate "state:interactsWith")(object ?o1))
 (logan-quad (graph ?g2)(subject ?person)(predicate "state:interactsWith")(object ?o2))
=>
 (bind ?newGraph1 (str-cat "urn://" (UUID.randomUUID)))
 (bind ?newGraph2 (str-cat "urn://" (UUID.randomUUID)))
 (bind ?newGraph3 (str-cat "urn://" (UUID.randomUUID)))
 (assert
  (logan-graphname (graph ?newGraph1))
  (logan-graphname (graph ?newGraph2))
  (logan-graphname (graph ?newGraph3))
  (logan-quad (graph ?newGraph1)(subject ?person)(predicate "interaction:with")(object ?o1))
  (logan-quad (graph ?newGraph2)(subject ?person)(predicate "interaction:with")(object ?o2))
  (logan-quad (graph ?newGraph3)(subject ?newGraph1)(predicate "domain:disjointWith")
     (object ?newGraph2))
 ))
```

Listing 3.10: An example that applies semantic relations in one information domain to detect inconsistencies in another.

This technique provides another way in which to simplify model development and increase the amount of knowledge that can be automatically inferred. In Ye et al. (YSD11a) we provide an example of this approach using a dataset from the *PlaceLab* smart home environment (LHP+07), where 32,170 relationships between 605 objects are inferred from their spatial relations.

3.8.6 Consequences for Software Engineering

Simplicity The main advantage of defining the common semantic relationships between domain concepts and using general rules to derive new knowledge is that it greatly simplifies the process of developing a context model that fully relates the data it contains. The structured, step-by-step, application of reasoning automates much of the otherwise time consuming process of generating a knowledge base, requiring developers to only specify a small percentage of knowledge in the information space.

This specified knowledge takes the form of adjacent and direct (i.e., not transient) subsumption relationships between concepts, or through providing the function that maps from unit values to concepts, allowing all other relationships to be derived automatically.

Consistency Another advantage of the mostly automated nature of this approach is that there is consequently less opportunity for the developer to specify incorrect knowledge. If they do so, and the reasoning process results in the presence of inconsistent knowledge in the model, this can be detected, highlighted, and the reasoning process can be worked backwards to identify the statements that supported the incorrect inference. In the case where model errors are due to the omission of knowledge, inspection of the fully generated model may highlight this through the absence of an expected relation.

Concept Hierarchy Construction If a non-trivial information domain is to be modelled, a developer might apply well established analysis techniques, such as formal concept analysis (GWF97), to aid in deriving a concept hierarchy from a collection of objects and their properties. At the other end of the spectrum, a developer might look towards a range of automated or semi-automated techniques for constructing a concept hierarchy (MS01; MWDB13). In both cases, and with other similar techniques, the outputs of these processes can be mapped to the model.

Opportunities for Sharing and Reuse Another key advantage of the approach, if one accepts the tenets of Linked Data – the exposing, sharing, and connecting of pieces of knowledge via Semantic Web technology – is that the developed ontologies, mapping functions, and environment models are reusable across multiple applications. Consequently, although significant effort may be required to develop an initial model of an information space as part of designing a new application, the development of subsequent applications requires less investment, the process being bootstrapped by previous efforts.

3.8.7 Limitations of the Approach

In addition to the benefits of modelling information spaces as described above, the approach described in this chapter also has some limitations that point towards potential future areas of investigation.

The model relies on the mapping of unit value sets to concepts, with the implicit assumption that the mapping of sensor output – the data entry point into the model –

directly follows. While in most cases this process is straightforward, there are some sensing modalities that present difficulties. For example, there's no obvious direct mapping between the output of a video stream and the context model; and it may be the case that information across multiple domains can be extracted from a single video feed (for example, presence, threats, and interactions). Such information source may require extensive pre-processing to extract interesting features, a process that sits outside our model. However, the output of this process, even if high level, can still be represented by the model.

Whether a centralised or decentralised implementation is used, as the volume of sensor data increases, it becomes infeasible to store a permanent record of generated states. The model presented here does not suggest a solution, however in Stevenson et al. (SYD10), we suggest that through capturing properties that describe data dynamics, policies may be defined to prioritise the discard of data that is asserted frequently but has a slow rate of change (for example, ambient temperature) over data that is frequently asserted but changes rapidly (for example, a subject's coordinate location) or pseudo-static data that is infrequently asserted (for example, building layouts).

The process of reasoning over a large data model can also be a significant performance bottleneck, however knowledge about the dynamics of data might inform an appropriate reasoning strategy. In Stevenson et al. (SYD10) we suggest reasoning on static data as it is generated, adding the inferred knowledge directly to the model, while performing reasoning on highly dynamic data only at the point where an application query is executed. Based solely on data's temporal properties, it may be possible to devise a general scheme to partition, reason on, and integrate data over several stages so as to optimise reasoning.

Relatedly, sensor-driven systems often place a significant amount of their functionality on devices with memory, computation, and communication constraints, therefore direct deployment of the model to such platforms using Semantic Web technologies is currently not practical. However, optimised versions for restricted domains or with restricted reasoning capability, given knowledge of specific information sources or application requirements, could be developed.

Finally, many systems may be interested in the complete lifecycle of data, including the transformations that have been applied to it. For example, it may matter whether the unit value data from a positioning sensor is raw or smoothed, the details of any sampling process, when and where reasoning was applied, and the provenance of the software that executed it. It is therefore vital for many systems that provenance is maintained along the data pathway. However, the lack of a standard vocabulary for capturing, interchanging and reasoning on provenance metadata challenges the realisation of these

goals. While such considerations have not been catered for in this work, the model developed in this chapter could, in the future, be extended to incorporate the findings of the Provenance Working Group (w3c13a).

3.9 Summary

We began this chapter by overviewing a range of technologies to assess their suitability for developing a model of context for an information space. The analysis showed that no single technology outshines the advantages of all others; the suitability of each technique is tightly coupled to the requirements of the system under development. However, among competing alternatives, ontological models provide the most flexible option due to the ease with which they can typically express and share complex information.

The remainder of the chapter then focused on the use of ontology-based technology, as realised by OWL, to develop a reusable top-level ontology model that provides a common substrate for developing domain and application ontologies for pervasive environments. The model employs set theory to develop a uniform approach to the modelling of concepts and describes a small core set of five semantic relations common to most dimensions of an information space: *equality*, *subsumption*, *disjoint*, *overlapping*, and *adjacent* relationships. These common semantics provide a uniform way to represent and reason on domain knowledge. Ontologies built using them are straightforwardly interpreted by evaluating and comparing these underlying semantics relations between concepts and the statements they help define.

The chapter proceeded to develop an integrated reasoning schema to support the automated derivation of *context statements* from the relationships between the *concepts* they describe. In turn, the relations between the concepts were shown to be directly inferable from *unit values*—the irreducible set of smallest values that can be *measured* in an information dimension. We introduced the concepts of temporally bound and uncertain context statements, adding the ability to restrict the times during which a statement is deemed to hold true, and provided a means of representing the confidence that a context statement reflects the true state of a sensed or profiled phenomenon—providing schemes to automatically manage the propagation of this information as new information is inferred from old. We presented a detailed worked example that brought together all the concepts discussed in the chapter and presented, in tandem, the implementation of the context model, using Semantic Web technologies, in our reasoning framework, *Logan*. Finally, we presented a discussion of the modelling approach described in the chapter, outlining several of its merits, as well as identifying some limitations that point

to future topics of research.

The resultant ontology model complements the existence of domain and application ontologies. Ontology engineers still need to define concepts and model a subset of model relationships or functions to map from unit values to concepts. They must also define rules to relate information across multiple information domains. Although these processes still involve an amount of engineering effort, our ontology model can assist in reducing the effort required, and, through Linked Data, aid in exposing, sharing, and connecting of pieces of knowledge via Semantic Web technology. Well constructed ontologies and domain models are potentially reusable across multiple applications and environments. Consequently, over time, less effort needs invested where the development process can be bootstrapped by an existing model.

In the next chapter we use the model developed in this chapter as a basis for developing an approach to the learning of *situation models*. The approach leverages the automated reasoning, semantic relations, and representation of uncertainty between concepts and context statements to generate models that describe semantically meaningful classifications of system state, allowing their occurrence to be determined from the information space model to which sensor data is mapped.

CHAPTER

FOUR

AN APPROACH TO LEARNING SITUATION RECOGNITION MODELS

In the previous chapter we used set theory to develop a uniform approach to the modelling of concepts, context statements and semantic relationships in information domains, towards the goal of providing a common substrate for application ontology development.

In this chapter we describe how this conceptual model is applied within one particular application domain, that of representing and learning intelligible *situation models*—white-box constructs from which semantically meaningful classifications of system state can be determined and differentiated from observed context. The requirement for transparent decision processes aids developers in constructing a mental model of how classification are made, and supports them in debugging poor performance and identifying how recognition may be improved.

In Section 4.1 we present mathematical models for constructing human-understandable situation specifications that are based on the information space model described in the previous chapter, followed, in Section 4.2, with a brief overview of mapping different interpretations of the information space to the recognition models. We then motivate, in Section 4.3, the use of learning to support the automatic construction of these models from training data as a means of overcoming the limits of human perception, and develop a genetic-algorithm-based approach to achieve this. Section 4.4 discusses how the efficacy of the generated models can be evaluated quantitatively and analysed through inspection, before we summarise the approach in Section 4.5.

4.1 Intelligible Situation Specification Models

In this section we outline two white box situation recognition models we examine in detail in this thesis. Considering the hypothesis of this work, we wish to investigate the extent to which such models can achieve high levels of accuracy, whilst producing decision processes that are intelligible to system designers.

4.1.1 Situation Specification Sets

A situation specification can be thought of as an inference rule that associates a particular combination of context statements (as discussed in the previous chapter) with a symbolic name that is application or human understandable. Or, put another way, a situation specification allows the conclusion as to whether or not a particular situation is occurring to be drawn from the output of a function that takes as input the set of model statements that currently hold.

Equation (4.1) gives an example of a simple specification that describes the situation of watching television as occurring when the television is switched on and someone is sat on the couch. The specification is described using boolean logic, with the conjunction ($\wedge$) of the two concepts interpreted as supporting the inference ($\leftarrow$) of the situation.

$$watch\_tv \leftarrow [:tv, \textbf{:state}, :on] \wedge [:someone, \textbf{:satOn}, :couch] \tag{4.1}$$

Given a set of situations that are interesting to an application, each is associated with an inference rule that specifies a conjunction of context statements thought to be indicative of the situation. Like the information space model that underpins them, such rules are constructed by a domain expert who is familiar with the constraints of the sensing technologies being used and the peculiarities of the environment within which they are deployed.

More formally, consider the set of context statements, S, expressible within the information space describing an environment, and a particular subset of these $\{s_1, s_2, \ldots, s_n \in S\}$ deemed relevant to detecting the occurrence of a particular situation, X. Each statement can be resolved to a boolean value of true or false, denoted as $\{\mu(s_1), \mu(s_2), \ldots, \mu(s_n)\}$, where $\mu(s_i)$ evaluates whether or not the context statement currently holds. Using this notation, situation X is specified as shown in Equation (4.2):

$$X \leftarrow \mu(s_1) \wedge \mu(s_2) \wedge \cdots \wedge \mu(s_n) \tag{4.2}$$

That is, at any time where statements $s_1 \ldots s_n$ all hold, we infer situation X to be occurring at.

For simple systems, where model statements can readily be associated with the situations under consideration, boolean-logic-style-rules can be intuitive to specify. However, such a model relies on the availability of *accurate* sensor data, an assumption that often does not hold. To illustrate this idea, consider the *watch_tv* specification described above where inference of the statement [*:someone*, **:satOn**, *:couch*] is predicated on the ability to sense that a person is sat on the couch. If a person is sat on the couch from time t_0 to t_5, a pressure sensor installed for the purpose of detecting this fact may fire only at times t_0,t_3, and t_5, leading to an oscillation in the situation inference. Consequently, alternative techniques are required to deal with this artefact of the real world.

The Presence of Uncertainty Information space models commonly include properties with which the uncertainty of data can be characterised. For example, a positioning sensor using GPS may be characterised as having a precision of one metre, indicating that the true position of the object being tracked lies with a sphere of radius 1 metre whose origin is the reported location. The sensor may also have a reported accuracy of 99%, meaning that, on average, for one in every hundred readings the true position of the tracked object lies outside the sphere's bounds.

The effect of uncertainty is not limited to physical sensors; for example, a virtual sensor that "observes" a person's personal calendar may be associated with a measure of confidence that reflects how accurately a person's calendar matches his or her true schedule, or reflects that the temporal extremities of events are rarely precise, and better characterised by techniques that gracefully increment the confidence in events commencing and ending as their calendar-stated bounds are encountered (ANH07).

More generally, uncertainty can result from the combined effect of a large number of factors that includes: sensor technical limitations, environmental noise, source trustworthiness, human misuse, and degradation of sensor quality over time (HIR02; CLC+02). Many of these factors are difficult to quantify or to learn, however, the consideration of uncertainty has been demonstrated to have an impact on the quality of situation recognition techniques (MYCD09b; YDM12).

Given the ubiquitous uncertainty of sensor data, a key requirement for developing a situation specification model is to move away from the simple boolean-logic-based approach outlined above towards one that incorporates uncertainty handling mechanisms (DHKZG08).

An Uncertainty-Aware Situation Specification Model Fuzzy logic provides one approach for reasoning over data that is approximate rather than exact (Zad73). In contrast to traditional boolean logic, where a variable evaluates to either true or false, fuzzy logic variables have a numeric *truth value* that lies in the range $[0:1]$, the extremities of 0 and 1 equating to their boolean counterparts of false and true respectively. *Degrees of truth* provide a mechanism for representing vagueness in knowledge both in cases where a concept is difficult to define crisply, as in the earlier example of a meeting start time, or where missing information, as demonstrated in the example of the couch pressure sensor, need not necessarily imply a sharp state transition from 1 to 0.

Using this notion of fuzziness, both sensor data and context statements derived therefrom may be associated with a numeric *confidence* value $c, c \in [0:1]$, that reflects the extent to which we believe the value truly reflects real world state.

In this work, we make the simplifying assumption that it is reasonable to quantify the different types of uncertainty described above in the same manner, through reduction to a degree of truth, representing the confidence that the statement reflects the true state of the sensed or profiled phenomenon, as described in Definition 12.

We can adapt the situation specification model developed above to incorporate this notion of confidence by three main steps:

- replacing the boolean evaluation of whether or not a context statement holds with their confidence-based counterparts;
- substituting the logical conjunction operator for an arithmetic alternative; and,
- changing the meaning of the specification implication ($\leftarrow$) from "a situation holds", to "a situation is assigned score *sc*", as calculated from the right hand side of the expression. This score could equate to a confidence that the situation holds, although this need not necessarily be the case, as we shall soon return to.

Given the stated goal of retaining simplicity in the specification model, there are two leading choices of arithmetic operator to replace logical conjunction: multiplication and addition. Redefining the function $\mu(..)$ to now return the confidence value that a statement holds, the earlier example of the *watch_tv* specification above is reformulated using both of these operators in Equation (4.3) and Equation (4.4) respectively.

$$sc(watch\_tv) \leftarrow [:tv, \textbf{:state}, :on] \times [:someone, \textbf{:satOn}, :couch] \tag{4.3}$$

$$sc(watch\_tv) \leftarrow [:tv, \textbf{:state}, :on] + [:someone, \textbf{:satOn}, :couch] \tag{4.4}$$

Of the two approaches, use of the multiplication operator results in the formulation closest to the original logical conjunction form. To illustrate, if the confidences in both the television being on and the couch being occupied are 0.99, the overall score assigned to the situation *watch_tv* is 0.98. The advantage of this approach is that the concept of situation score can be seen to have an intuitive link to situation confidence if we consider all confidences as probabilities under a naïve independence assumption. However the approach is limited by the likelihood of situation scores tending towards 0 in specifications constructed from a large number of statements or, as might happen more frequently, where one or more statements are associated with low confidence.

The additive model does not retain the strength of link between the concepts of situation score and situation confidence but does address both limitations of the multiplicative model. Further to this, associating each sensor input with a *weighting*, allows the contributions of each part of the specification to be considered according to its relative importance (DHKZG08). Using this approach, the overall confidence in a situation occuring may be calculated by summing the weighted confidence values of those statements deemed by a domain expert to contribute to its identification. For example, weighing determination of the television state as four times more important than whether or not somebody is sat on the couch, as illustrated in 4.5.

$$sc(watch\_tv) \leftarrow 0.8[:tv, \textbf{:state}, :on] + 0.2[:someone, \textbf{:satOn}, :couch] \tag{4.5}$$

For these reasons, we adopt the addition-based model as the basis for specification construction. Although this approach relaxes the direct link between score and confidence, learning techniques provide a basis for scoring situations, which we will return to in the next section.

We express this model more formally as follows: consider the set of context statements, S, expressible within the information space describing an environment, and a particular subset of statements $\{s_1, s_2, \ldots, s_n \in S\}$ deemed relevant to the detection of a particular situation, X. The confidence that each statement holds is evaluated by a function μ, $\{\mu(s_1), \mu(s_2), \ldots, \mu(s_n)\}$, denoted more simply as $\{c_1, c_2, \ldots, c_n\}$. Furthermore, let $\{w_1, w_2, \ldots, w_n\}$ be the weighted contribution of each s_n to the specification. Using this notation, the specification for scoring a situation X can be expressed as shown in Equations 4.6 to 4.1.1:

$$sc(X) \leftarrow \sum_{i=1}^{n} w_i.c_i \tag{4.6}$$

where

$$\sum_{i=1}^{n} w_i = 1$$

and

$$0 \leq c_i \leq 1, \forall c_i \in \{c_1, c_2, \ldots, c_n\}$$

Thus a situation's score is calculated as the sum of n weighted atoms, each representing the confidence that a context statement, accurately reflecting the true state of the world, holds.

Specification Sets Using the uncertainty-aware specification model, the next step is to provide a framework for choosing the situation that best characterises the set of context statements that hold in the information space at a given time from among competing alternatives. This is straightforwardly realised by selecting the situation associated with the highest scoring specification from among all candidates.

4.1.2 Decision Trees

The second recognition technique we investigate in this thesis is the *decision tree* (WF05). Among all the standard machine learning techniques surveyed in Chapter 2, Decision trees uniquely have the potential to produce decision processes that are understandable to the human eye, depending on the complexity of the underlying phenomena they model.

To recap, decision trees take the form of a tree structure, where each node represents a decision taken with respect to the value of a feature, the outcome determining which branch should be followed. As with the specification set model, we consider the set of features to correspond to the set of all possible context statements expressible by the model, while the values on which decisions are made correspond to a particular interpretation of that context statement: Either a boolean value of true or false, or the real-valued confidence that each statement holds, as described in Definition 12.

Instead of composing specifications that correspond to individual situations, as with the specification set model, a single decision tree defines a unified decision process for selecting the most likely situation from among all possibilities, given the state space of the information space. Its evaluation begins from a single root node, and continues until a leaf node – representing a classification – is reached (WF05). Figure 4.1 illustrates a partial decision tree that might recognise the *watch_tv* situation from the same two

context statements identified earlier.

4.2 Alternative Information Space Interpretations

To this point, we have focused discussion only on the interpretation of the current state of the information space model. However, it is possible to construct specification sets and form decision trees by projecting different aspects of state onto the information space model.

To illustrate, van Kasteren (vKNEK08), whose dataset we use later, proposes three different interpretations of sensor data: *raw*, where a sensor returns the value 1 when firing and a 0 otherwise, *change point*, where a sensor takes the value of 1 only at times when its state changes, and *last changed*, in which the last sensor to change state evaluates to 1 and changes to 0 when a different sensor fires.

It is possible to construct information space models analogous to all three of these representations, that is: *raw* where a context statement evaluates to true, false, or its confidence value depending on whether it currently holds (i.e., the model discussed to this point), *change point*, where a context statement evaluates to true (or its confidence value) only at the point in time when its state changes, and *last changed*, in which only the last context statement to experience a state change evaluates to true (or its confidence value), changing to false when another state change occurs.

4.2.1 The *TimeSince* Interpretation

In addition to those representations discussed above, we propose an additional interpretation for context statements in an information space, called *TimeSince*, that aims

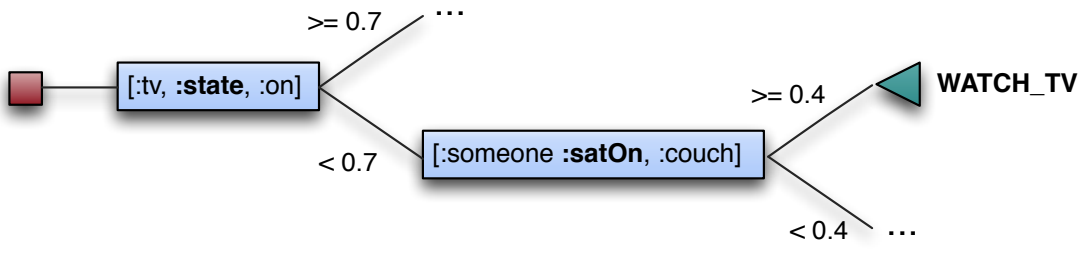


Figure 4.1: An illustration of how a decision tree may infer the occurrence of the *watch_tv* situation from the confidence values associated with two context statements.

to capture the recency with which state changes in the information space have taken place. This interpretation is inspired by the work of McKeever et al. (MYC+10) who found projecting sensor state forward in time beyond the point of change to be useful in capturing the recent occurrence of a sensor firing regardless of any subsequent state changes.

When the *TimeSince* representation is used, context statements do not evaluate to true, false, or their confidence value, but instead to a numeric value representing the time elapsed since the statements last held. This is calculated from the temporal qualification of the context statements expressed in the model. For example, if the time now is given by t_{now}, and the following is the most recent occurrence of a particular statement in the model

$$[:somebody, \textbf{:satOn}, :couch]_{t_e}^{t_s}$$

the *TimeSince* interpretation of the statement evaluates to $t_{now} - t_s$, that is, the elapsed time between the statement first holding and the time of interpretation. Figure 4.2 illustrates how this model interpretation may be combined with the standard interpretation as part of a decision tree. By specifying the statement $[:someone, \textbf{:satOn}, :couch]$ in terms of the time since it most recently held, we mitigate the state oscillation problem outlined in the previous section.

4.3 An Algorithm for Learning Situation Recognition Models

In the following, we proceed by developing an algorithm for learning specification sets and decision trees, leveraging the constructs of the information space ontology model.

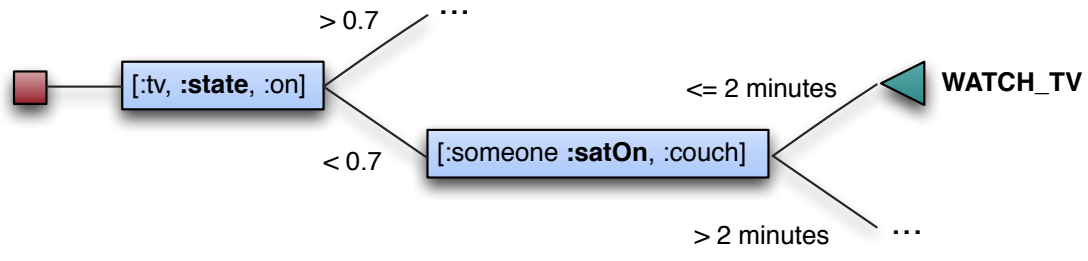


Figure 4.2: An illustration of a decision tree using the *TimeSince* model interpretation.

4.3.1 Motivation for Learning

Our choice of models for constructing situation recognition techniques is motivated by a desire to retain easy human understanding of specifications, at the possible expense of performance levels that might be attained through additional complexity. Given this motivation, it is fair to ask why it should then be necessary to adopt an approach to learning specifications and decision trees, where the model simplicity would suggest the possibility of manual construction.

Although we posit that for some environments and some situation sets it is possible to manually construct a specification set or decision tree, in general it can often be difficult for humans to write *good* models, even when the comprehension of such models is relatively straightforward. Here we offer six reasons as to why the process of manually constructing situation recognition models and decision trees can often be difficult:

Mental model A designer's mental model of which context statements and their corresponding state interpretations are most indicative of a situation may not match the real world. This may be caused by an incorrect understanding of the bounds of the situation to be recognised, by uncertainty inherent in sensor data mapped to the information space model, or by sub-optimally selecting context statements to characterise the situation.

Abstraction level Selecting the correct level of abstraction at which to represent information in a specification can be difficult. Selecting a representation that is too fine grained risks capturing only a subset of conditions that satisfy the situation, resulting in false-negatives. Conversely, selecting a representation at too coarse a granularity risks capturing conditions that lie outside the situation, resulting in false-positives.

Weight selection (specification set) Although the relative weightings of context statements in a written specification may be straightforwardly comprehended, it does not follow that selecting the correct weights is equally simple.

Boundary values selection (decision tree) Similar to the problem with weight selection, although a decision node based on the interpreted value of a context statement (e.g., confidence or *TimeSince*) may be straightforwardly comprehended, selection of the boundary value is not necessarily simple.

Consideration of relative scoring (specification set) The specification set model requires the specification associated with the situation that best characterises the

environment to score highest among all specifications. The designer must therefore consider simultaneously the relative scorings across the complete set of specifications. This can be difficult where there are a large number of situations to be specified, where situations are only subtly differentiated, or where a context statement (or a related statement at a different abstraction level) appears in multiple rules with different weights associated.

Use of contrast as a differentiator Finally, in cases where the sensing capability of an information space does not adequately support the construction of the context statements one might wish to use to specify a situation, it may be possible to describe a situation based partially or wholly on the negation of the context statements critical to the other situations in the set. That is, the presence of a situation can be inferred through the non-occurrence of the other situations. Such a specification *by contrast* can be difficult to construct manually.

We use these factors as motivation for developing an approach to specification set and decision tree learning while noting that retaining the ability for experts to easily inspect specifications can lead to insights about why a learned set of specifications may look or perform differently than an expert might expect, namely, those self same factors that complicate manual specification development. We return to this point later.

4.3.2 Algorithm Goals

Considering the formulations of the specification set model presented in Section 4.1.1 and the decision tree model presented in Section 4.1.2, the common goals of an algorithm to learn both models are expressed as follows:

1. To identify the context statements most relevant to the identification of a set of situations from that expressible in an information space;
2. To identify the most useful level of abstraction and logical primitives for interpreting information contained in context statements; and,
3. To restrict the size of the models generated to aid intelligibility.

For the specification set model, the learning algorithm is used to construct the specification sets. That is, to weight the contribution of each interpreted context statement appropriately with respect to both its containing specification and the set of specifications to be identified as a whole.

For the decision tree model, we use the standard approach to constructing trees using *information gain*. Here the learning algorithm is applied as a search heuristic, using the above criteria to guide the generation and selection of a model from among all possible decision trees.

4.3.3 Motivation for using a Genetic Algorithm

Genetic algorithms (Hol92) provide an approach to solution search and optimisation through the use of heuristics. Based on the principle of natural selection (Dar59), a population of candidate solutions is evolved towards an optimal solution.

Genetic algorithms start with an initial population of solutions, usually of some fixed size, and discard those evaluated to be poor candidates. The population is then replenished by generating new solutions through the *mating* of promising solutions, with offspring inheriting the traits of their parents. Random modifications to the solution, called mutations, are introduced to prevent the population from reaching a homogeneous state. This process mimics biological evolution, which can be phrased as the search for organisms capable of reproducing in their environment, where the solution space is the set of all possible genetic sequences (Mit98).

Solution Search Space Genetic algorithms are suited to problems that have a potentially large number of possible solutions and can benefit from parallelism by exploring many different possibilities simultaneously, which is the case here. Consider, for example, a specification set that takes the form of a specifications, each constructed from context statements whose object values are taken from b possible information domains. Each domain is described by one of c possible permutations containing an average of d concepts (the number of concepts per permutation, varying from a maximum value where all statements that constitute the domain are considered individually, through intermediate values depending on concept subsumption relationships, to a single, all subsuming, concept at the top of the hierarchy). Each statement that relates a different concept is associated with a weight e and any combination of f ontological relations (e.g., $\lhd$, $\|, \infty, \neg$) may be evaluated in combination when calculating statement scores. This search space, the number of unique solutions, is characterised by $a \times b! \times c \times d \times e \times f!$.

Consider the example of a search for a set of specifications for 7 situations in an environment described by context statements that take values from 6 domains, each with 16 possible concepts permutations, an average of 1.7 concepts per permutation. To simplify the set of possible statements that can be represented, assume that the subject

and predicate of each statement is fixed. Each statement also has a weighting between 0 and 1 considered to 2 decimal places, and 3 ontological relations: overlap, adjacency, and compliment (containment is accounted for by the permutations). This results in a total solution space of 82,252,800 possibilities ($7 \times 6! \times 16 \times 1.7 \times 100 \times 3!$)

The search space of the decision tree model is smaller, given that *i)* no weighting or negation is required, and *ii)* a single model is constructed using a common set of inputs statements. This example, perhaps indicative of a small sensor installation in a house, evaluates to 58,752 candidate solutions. While such a solution space might be searched brute force, this approach would likely not scale to larger models, such as a large-scale sensor installation in a house, exhibition hall, or business campus.

Suitability of the Problem Domain The use of a genetic algorithm requires that the problem domain satisfies three main requirements:

- Firstly, it must be possible to evaluate the relative quality of one solution to another;
- Secondly, the solution must be constructed from building blocks that are allowed to vary independently; and,
- Thirdly, a good solution must be considered acceptable, as the best solution is not guaranteed to be found.

The problem posed by the generation of specification sets and decision trees satisfies all three of these criteria:

- The relative quality of one solution to any another can be quantitatively assessed by evaluating the solutions against training data;
- As evidenced from the characterisation of the search space above, specification set solutions are described by the composition of a number of situations, within which appear context statements involving domain concepts, weights, and ontological relations, while the decision trees are described by a subset of these elements. All of these characteristics can be allowed to vary independently.
- The critical role played by uncertainty, both in human behaviour, sensor operation, and the extent to which training data captures all the ways in which a situation is realised, precludes the existence of an optimal solution for most scenarios. Were a solution with 100% recognition accuracy in training to be found, it does not follow

that the solution would achieve perfect recognition when deployed. Similarly, given two solutions A and B that achieve 87% and 88% accuracy respectively during learning, it does not necessarily hold that B will always outperform A, even if a similar level of recognition accuracy is maintained. The uncertainty inherent in the problem domain justifies why a good solution will often suffice in place of the best solution.

4.3.4 Genetic Algorithm Concepts

In a genetic algorithm, problem solutions are encoded as *chromosomes*. A chromosome consists of a number of *genes*, each of which represents a possible value, or *allele*, of a candidate solution variable. Typically genes take the form of integers, strings, or boolean values, although others gene types are possible. Composite genes, called *supergenes*, may be used to group related genes together.

To test a solution, chromosome values are "plugged" into their corresponding variables in a program of fixed structure (this is in contrast to Genetic Programming (Koz92), where the program structure is evolved as part of the solution). The efficacy of the solution this produces is evaluated by a *fitness function*—an algorithm designed by the programmer to yield a numeric rating of the solution; the lower (or higher) the fitness value, the better the solution relative to other solutions evaluated by the same function.

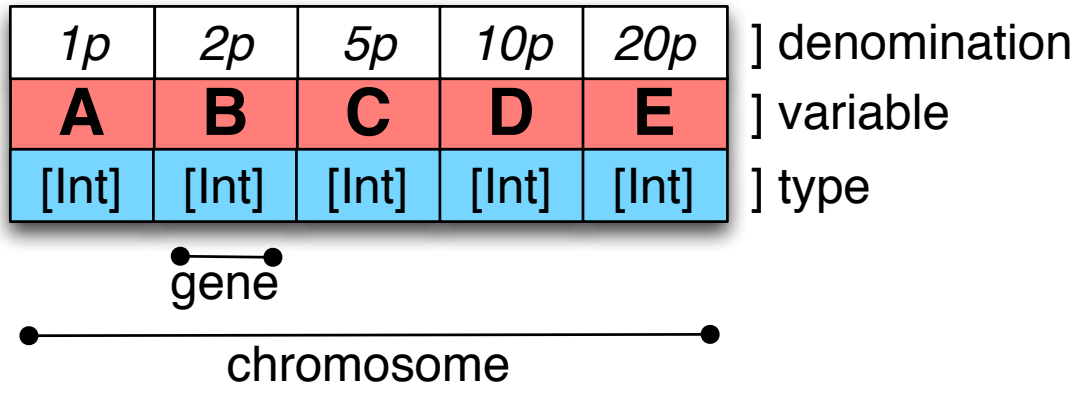


Figure 4.3: The construction of the chromosome to represent solutions to the coin problem.

We illustrate these concepts with an example[1]. Consider the problem of calculating the minimum number of coins of the pound sterling that add up to a total closest to but no greater than 100 pence, using only the coins: $1p$, $2p$, $5p$, $10p$, and $20p$. Here, candidate solutions represent a total amount of money calculated from a combination of these

[1] Based on the JGap tutorial: `http://jgap.sourceforge.net/doc/tutorial.html`

coins. The design of a chromosome to represent a solution to this problem is shown in Figure 4.3. It consists of five genes, one for each coin denomination, each restricted in the values it can take by a type (integer). Each gene is associated with a variable name, allowing it to be mapped to programs, given in Equation (4.7) and Equation (4.8), which calculate the sum of money and number of coins respectively represented by a solution i.

$$Total_i = A_i + 2B_i + 5C_i + 10D_i + 20E_i \tag{4.7}$$

$$Count_i = A_i + B_i + C_i + D_i + E_i \tag{4.8}$$

To determine whether one solution is better than another, we design a fitness function to return a numeric representation of the quality of the solution described by a particular chromosome. For this example, the fitness function must consider both the total value of the coins and the total number of coins a solution represents.

Equation (4.9) represents one possible fitness function suitable for this problem. Here, the fitness value is calculated as the magnitude of the difference between the desired total and total indicated by the solution, modified by the total number of coins. The closer the fitness value of a solution is to 0, the better the solution is considered to be.

$$FV_i = |Desired - Total_i| + 0.01 \times Count_i \tag{4.9}$$

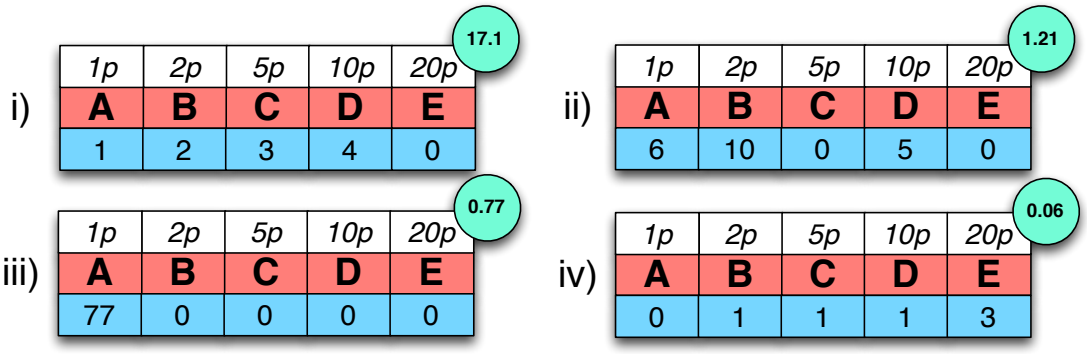


Figure 4.4: Instantiated chromosomes representing solutions to the coin problem with a target of 77 pence.

Figure 4.4 shows the encoding of four possible solutions and their fitness values for a desired total of 77 pence: solution *i)* totals 60 pence with 10 coins and scores worst. Solution *ii)* totals 76 pence using 21 coins but is outscored by solution *iii)* which totals the correct value using 77 coins; this results from the fitness function prioritising the

correct total over the number of coins used. Finally, solution *iv)* is the optimal solution, representing the desired total using 6 coins.

A genetic algorithm typically executes as follows:

1. An initial population of n chromosomes (called a *genotype*) is generated by randomly generating values for each of the genes in each chromosome.
2. The fitness of each chromosome in the genotype is evaluated to produce a numeric measure of the quality of each candidate solution.
3. The best b individuals are selected from the population based on their fitness, with the remaining chromosomes discarded.
4. The genotype is repopulated to capacity n by repeatedly:
   (a) Selecting two *parents* from the individuals retained from the previous genotype to breed.
   (b) Crossing over the pair of chromosomes, with probability p_c, at a randomly chosen point to construct two *offspring*. If no crossover takes place, two offspring that are clones of their parents are created.
   (c) Mutating each gene of the two offspring chromosomes with probability p_m.
   (d) Placing the two offspring into the genotype.
5. As the offspring share some characteristics of their parents, each successive *generation* of the genotype generally has an increased average fitness as a result of it being bred from the fittest organisms of the previous generation.
6. The process recurses from step 2 for either a fixed number of generations, a fixed length of time, or until the fitness of the best solutions in the genotype converges.

The algorithm above has three key parameters—*population size* n, *mutation rate* p_m, and *crossover rate* p_c. We describe the effect of each below.

Population size This is the size of the genome, and its selection represents a tradeoff between different performance aspects. A small population may be unable to find a good solution in an acceptable number of generations, while the fitness of a large population may take a long time to evaluate.

Mutation rate As with population size, selection of the mutation rate is also a trade off. Choosing a low probability of mutation may not allow an algorithm to escape a local-minimum, resulting in premature convergence of the population's fitness. However a high mutation rate may discard good solutions from the genome.

Crossover rate Similarly to mutation rate, the choice of crossover rate, which determine how genes from parent chromosomes combine to produce a child, may have positive or negative effects depending on the problem domain.

4.3.5 Encoding Specification Sets Solutions as Chromosomes

As discussed previously, the solution space of the learning problem is the set of all possible specification sets. A chromosome model must be capable of representing all possible solutions, with an instantiated chromosome representing one possible solution—a particular specification set among all possibilities.

Informally, we achieve this by forming the chromosome from the combination of multiple, identical subparts, each representing an individual situation specification. Each situation specification is in turn constructed from a set of supergenes—each representing a context statement template of the form $[?s, \mathbf{p}, ?o]$ described by the information space model, where p is a fixed relation, and $?s$ and $?o$ take either fixed values, or are assigned context domains that are allowed to vary. For example, the template context statement $[:bob, \mathbf{:locatedIn}, ?o[Space]]$ represents the set of statements that relate Bob to a location (a concept with type *Space*), where the location is variable.

The datasets we use in this work (see Chapter 6) consider only single person environments, with only one relation associated with each concept. That is, $?s$ and $?p$ are always fixed in the set of statements we consider. Therefore, in the following, we simplify the model by representing only the object domain as the variable part of the solution (that is we model domain permutations instead of statement permutations), although the principle is the same.

Accounting for this simplification, each supergene is further decomposed to support the representation of all possible concept permutations in the object domain of the context statement; this is achieved by associating each concept with a flag indicating whether or not the concept is present in the specification, a numeric weighting, and further flags for each of the possible semantic operators that can be applied in the process of scoring the context statement.

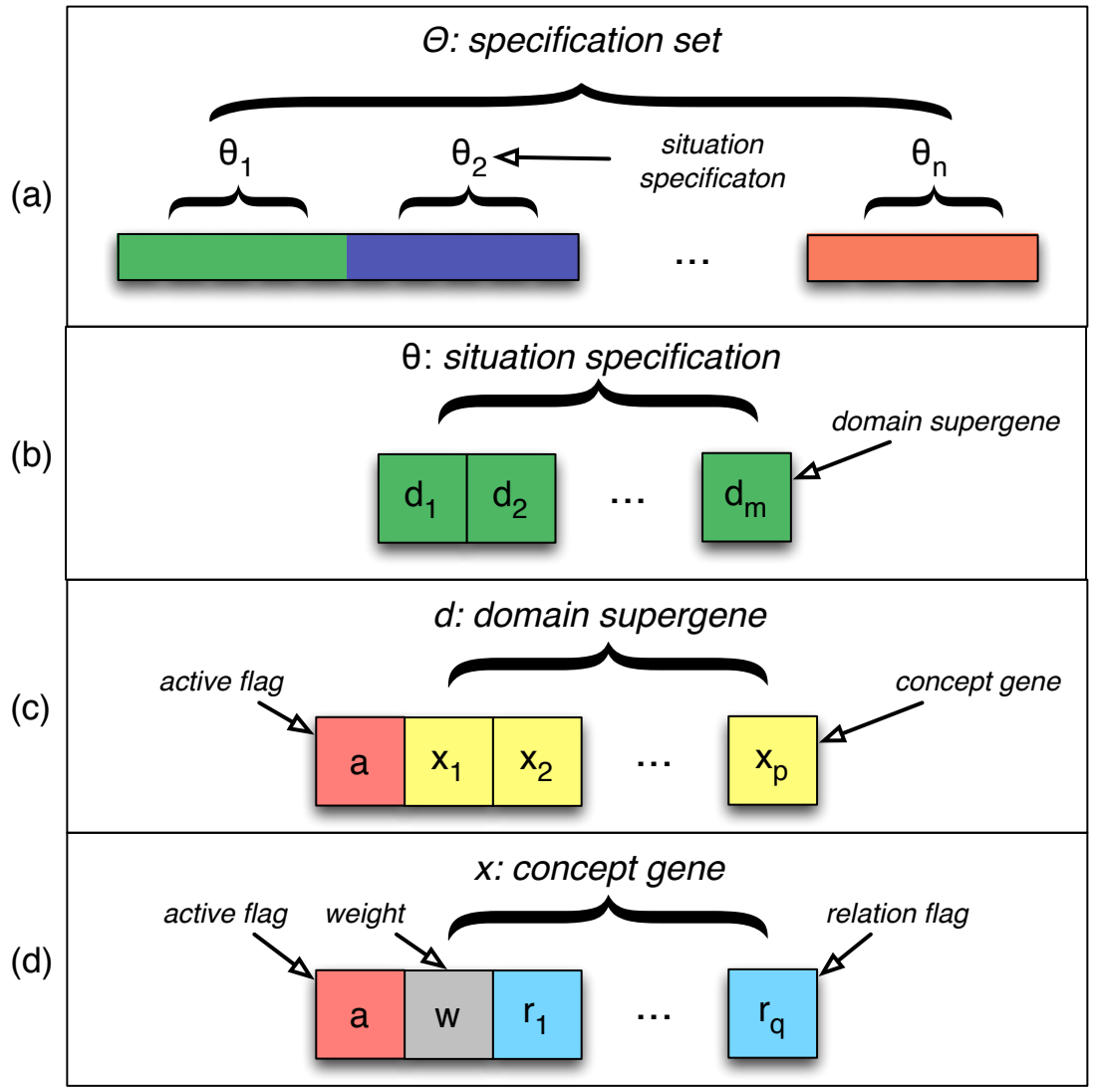


Figure 4.5: An illustration of the chromosome used to encode a set of situation specifications.

The resulting chromosome structure is represented in Figure 4.5. More formally, given a set of situations, Θ, the solution chromosome is divided into n parts, $n = |\Theta|$, each part encoding the specification of a single situation (Figure 4.5 (a)).

For the set of concept domains D, each specification then comprises m domain supergenes, $m = |D|$, that is, one for each domain described by the concept model (Figure 4.5 (b)).

The domain supergenes themselves are further decomposed into two parts (Figure 4.5 (c)). The first part is a single gene, that takes a boolean value, represents whether or not the domain, considered in its entirety, is active in the specification. The second part is a set of p genes, $p = |X|$, where X is the set of concepts modelled by the domain.

Finally, the representation of each concept is subdivided into three parts (Figure 4.5 (d)).

An activity gene taking a boolean value represents whether or not the concept forms part of the specification. Next, a weighting gene taking a real value between 0 and 1 describes the importance of the concept in the specification. Finally, a set of q boolean genes, $q = |R|$, where R is the set of semantic operators, represent which combination of the semantic operators (i.e., overlap, adjacency, and complement) are applied in scoring the context statement.

Note that a concept with a weight of 0 is equivalent to the same concept with its activity gene set to the value false. The inclusion of the activity gene increases the probability of generating specifications that use different combinations of concepts (and will more likely generate specifications with redundant concepts removed) than if the weight value is used alone. To illustrate this, consider that an activity gene has two possible states, while a weight gene, even if we only consider its value to two decimal places, has 100 possible states. That is, each time an activity gene mutates, there is a 50% chance that a concept will be "switched on" or "switched off", while using only the weight gene, there is only a 1% chance of this behaviour occurring upon mutation.

4.3.6 Encoding Decision Tree Solutions as Chromosomes

The chromosome required to represent a decision tree solution is a subpart of that required to represent the specification set: A single model is constructed for all situations, with the tree construction delegated to the standard model based on information gain. Thus only levels (b) to (e) of the chromosome depicted in Figure 4.5 are necessary, with the genes corresponding to weight and negation disabled throughout.

4.3.7 Fitness Function Selection

A fitness function provides a numerical measure of a solution's quality, and allows two solutions to be ordered according to their relative quality.

The standard approach to evaluating situation recognition accuracy is through measuring *precision* and *recall* against a dataset. Precision is defined as the ratio of times a situation is correctly inferred (N_{cor}) to the total number of times it is inferred by the recognition technique (N_{inf}). Recall is defined as the ratio of times a situation is correctly identified (N_{cor}) to the number of times it occurs in the dataset (N_{occ}). Both equations are shown in Equation (4.10) and Equation (4.11).

The F-measure combines both precision and recall into a single measure, variants of which can be used to weight the importance of one factor over the other. The standard F-

measure, shown in Equation (4.12), is calculated as the harmonic mean of the precision and recall, with each factor weighted equally.

$$Precision = \frac{N_{corr}}{N_{inf}} \tag{4.10}$$

$$Recall = \frac{N_{corr}}{N_{occ}} \tag{4.11}$$

$$FMeasure = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{4.12}$$

Accordingly, we construct a suite of fitness functions for the genetic algorithm based on this standard evaluation criteria, accounting for the type of model generated.

Specification Set Fitness Function Given a training dataset, the algorithm scores each candidate solution by calculating the average F-measure across all situations in the dataset. We augment this by imposing a penalty for the number of active statements that appear in the solution. This produces a selection pressure that helps to prune unnecessary statements from rules. That is, given two equally scoring specification sets, the set described by the fewer number of statements is deemed superior. This results in the fitness function shown in Equation (4.13).

$$Fitness = \frac{\sum_{i=1}^{n} FMeasure(\theta_i)}{n} - 0.005 \times |statements| \tag{4.13}$$

A strength of the genetic algorithm approach is that the fitness function can easily be changed to support different requirements (e.g., to use the $F_{0.5}$-measure or F_2-measure where precision is deemed more important than recall or vice versa). We explore this in the next chapter as part of an approach to ensemble classification.

Decision Tree Fitness Functions As with the specification set model, we investigate the effect on the decision tree model of imposing a penalty for the number of active statements in the solution (shown in Equation (4.13)). In addition, we also investigate the effect of penalising solutions based on the total size of the generated tree, as given by the fitness function shown in Equation (4.14), with a view to improving intelligibility.

$$Fitness = \frac{\sum_{i=1}^{n} FMeasure(\theta_i)}{n} - 0.005 \times |treeNodes| \tag{4.14}$$

4.3.8 Executing the Genetic Algorithm

Using the chromosome model for either the specification set or decision tree, a fitness function, and training data, the genetic algorithm executes following the standard process described in Section 4.3.4.

The algorithm is initialised using mutation and crossover rates of $\frac{1}{|genes|}$. That is, each gene has the same probability of mutating and of being the crossover point, with the rates proportioned according to the size of the chromosome. The initial population is created through instantiating a set of candidate solutions with randomly generated values for all their genes (active, weight, and relation).

Finally, the criteria for convergence, and termination of the algorithm, is set to be the point at which the maximum fitness of the population has not increased for 20 successive generations.

4.4 Evaluating an Intelligible Situation Specification Model

After the genetic algorithm has executed, the generated set of specifications may be evaluated in three complementary ways:

- Through automatically evaluating its recognition accuracy on unseen (test) data;
- Through visually inspecting the *confusion matrix* (Ste97) produced from the evaluation on test data; and,
- Through visually inspecting the generated specification set or decision tree.

The first approach, evaluation on test data, provides the main indicator of the expected performance level of the model when deployed in the target environment, and is the means by which we assess the efficacy of generated specifications in this work. If the performance level is satisfactory, there is likely little need to inspect the detail of the generated specifications further.

Situation	*A*	*B*	*C*
A	0.95	0.05	0.00
B	0.00	0.99	0.01
C	0.02	0.77	0.21

Table 4.1: A confusion matrix showing recognition performance on three situations.

If the performance of the generated model is not satisfactory, visual inspection of its confusion matrix is the next step. Using the output from the test data evaluation, a confusion matrix tabulates the situation classifications output from the specification set against the expected (ground truth) situation classifications. This provides a means of identifying the situations that are commonly confused, that is, where the specification set commonly classifies one situation as another. This is illustrated in Table 4.1, where it can be seen that among three situations (*A*, *B*, and *C*), instances of situations *A* and *B* are correctly classified 99% of the time, while 77% of the time *C* is misclassified as situation *B*.

From this, we finally move to manual inspection of the generated specifications, which serves three main purposes: *i)* diagnosing the underlying reasons for any confusion identified from the confusion matrix; *ii)* diagnosing general performance issues; and, *iii)* gaining insight into steps that might improve the recognition. This is the main benefit of using a white box decision process. Although detailed analysis requires the designer's expert knowledge of an environment, there are some common insights that can be gained from simple inspection:

- A poorly recognised situation described by a large number of statements or many tree decision paths may indicate a situation with few identifying characteristics that can be directly sensed.
- Similarly, the presence of the compliment operator in a situation specification may indicate that an important factor cannot be directly sensed and must be inferred from the lack of activity in sensors associated with other situations.
- The presence of the adjacency operator can indicate that sensors in the environment are not accurate enough or are affected by noise. For example, an activity described by a specification that incorporates location information adjacent to a space, may indicate a poorly calibrated positioning system.
- The omission of an expected term from a closely linked situation may indicate a faulty sensor, a situation being performed in an unexpected way, or an error in the model construction.

Application of the above combination of evaluation techniques may lead to either an application redesign to disregard situations that cannot be reliably identified; deployment of additional sensing technology in the environment to improve the ability to recognise particular situations; or, in some cases, the selection of a different learning technique.

The generation and evaluation of specifications may be conducted as an iterative process until a satisfactory level of performance is achieved.

4.5 Summary

We began this chapter by outlining two intelligible situation recognition models based on different interpretations of the information space model developed in the previous chapter, and motivated the automatic learning of these models due to several factors that are inherently difficult to account for manually.

After this, we justified the selection of a genetic algorithm for this learning task by demonstrating that the learning problem satisfies three criteria: *i)* the ability to evaluate the relative quality of one solution to another; *ii)* the solution being constructed from building blocks that are allowed to vary independently; and, *iii)* the discovery of a good, rather than optimal, solution being an acceptable outcome of the learning process.

We then introduced the core Genetic Algorithm concepts, and described the process by which the constructs of the information space model developed in the previous chapter are mapped to the chromosome structure required by the Genetic Algorithm. We also formulated several fitness functions that incorporate aspects of performance along with concept and tree size restriction.

Finally, we discussed one possible methodology for analysing the results of the specification generation process: Performance evaluation on test data, inspection of a confusion matrix, and manual inspection of specifications—outlining common problem indicators and corrective actions that may be taken in response.

In Chapter 7, we evaluate all the techniques presented in this chapter, however first, in the next chapter, we expand the approach described here towards the construction of *ensemble classifiers*, and investigate techniques that can be used to structure reasoning with the aim of improving classifier performance and intelligibility.

CHAPTER

FIVE

ENSEMBLE CLASSIFICATION

The previous chapter described how the conceptual model of an information space that we developed in Chapter 3 is applied to the task of representing and learning situation recognition models; in particular, specification set and decision-tree constructs that allow semantically meaningful classifications of system state to be determined and differentiated, based on abstracting observed sensor values to associated context statements.

This chapter extends the work of the previous chapter, investigating approaches to construct recognition model *ensembles*. The core idea of an ensemble methodology is to combine a set of models, each of which contributes to solving the original task, to obtain a better composite global model, with more accurate and reliable decisions than can be obtained from using a standalone model. Here, we seek to combine multiple specification sets or decision trees (which we collectively refer to here as *classifiers*), which, when interpreted, provide improved situation recognition accuracy over one used alone. We outline approaches investigated under two main headings: the first, a *genetic ensemble*, adapts the fitness function of the genetic algorithm to generate an ensemble of classifiers that treat different situations preferentially; the second leverages model semantics to group together similar situations in the form of a *semantic hierarchy*, with each level refining the classification performed at the level above.

We outline the approach to the genetic ensemble in Section 5.2, followed by the semantic hierarchy in Section 5.3. In describing each of the ensembles, we overview the underlying concepts, outline the methodology that describes the process of designing, generating and evaluating the ensembles, and provide an example illustrating their use. However, we first begin in Section 5.1 with a brief primer on Dempster-Shafer Theory, a mathematical theory of evidence that plays a key role in constructing the genetic ensemble.

5.1 A Dempster-Shafer Theory Primer

The literature describes many approaches to construct an ensemble classification scheme, including, for example, majority voting (AP96) and performance weighting (OS96). An approach that has proven popular in the area of pervasive- and sensor-systems, and situation recognition is Dempster-Shafer theory (DST) (Dem68; Sha76), a mathematical theory of evidence that provides the ability to combine evidence from different sources to arrive at a degree of belief, or confidence. The theory's popularity stems both from its ability to preserve ignorance (uncertainty), where sources need only assign confidence values to a subset of all possibilities, and the possibility of assigning confidence to sets of situations (for example, *working* or *reading*) as well as situations individually.

Approaches in the literature that employ DST (WSSY02; SF02; ZCZG09; MYC+10) in the construction of situation recognition models, which we discussed in Chapter 2, do so at the sensor level—relying heavily on expert knowledge to define the evidential hierarchy and to quantify the uncertainty of sensor data. Contrastingly, in this work we use DST in a different manner, applying it not to individual sensors but to individual classifiers, whose uncertainty is qualified automatically through evaluation on test data.

5.1.1 Features of Dempster-Shafer Theory

Dempster-Shafer Theory combines independent pieces of information to calculate a consensus belief in an event. The approach consists of three main elements: the *frame of discernment*, *mass function*, and *evidence combination*.

Frame of Discernment The Frame of Discernment represents the set of all possible hypotheses, $h_1, h_2 \ldots h_n \in H$, to which belief can be applied. Hypotheses can take the form of single classifications (for example, *meeting*), or sets of classifications (for example, *working OR reading*).

Given a scenario with n possible classifications, the Frame of Discernment has size 2^n, i.e., the power set that describes all possible classification combinations in the frame.

Mass Function The mass function describes how an evidence source applies belief across the hypothesis described by the Frame of Discernment. Each source has a finite amount of belief to allocate, totalling to the value of 1. Absolute uncertainty is quantified by assigning belief to the hypothesis containing all elements of the frame (i.e., a commitment to no hypothesis).

In a typical sensor-led use of DST, the mass function describes the allocation of belief based on how representative a sensor reading is of a particular situation. However, in this work, the mass function describes how representative a particular evaluation of a classifier is of a particular situation, as evidenced by training data. That is, given a scenario that describes n situations, a recognition model's mass function describes n possible belief allocations—one for each possible situation classification.

For each possibility, belief is computed as the ratio of instances where situation s_h is incorrectly classified as s_i ($N_{s_h}^{s_i}$) to the total number of times s_i is inferred ($N_{\text{inf}}^{s_i}$), as shown in equation 5.1.

$$\text{belief} = N_{s_h}^{s_i} / N_{\text{inf}}^{s_i} \tag{5.1}$$

Consider the three-situation scenario of *working*, *reading*, and *meeting*. Table 5.1 illustrates the mass functions of two classifiers. When classifier one evaluates to the situation *working*, 95% of the belief is allocated to *working*, 2% to *reading*, and 3% to *meeting* situation. Other outcomes are similarly interpreted. In particular, note that when the first classifier evaluates to *meeting* it is actually *reading* that is the most likely to be occurring. The mass function allows this uncertainty to be straightforwardly captured and adjusted for.

Evidence Combination The next step in applying DST is to combine the evidence from multiple sources to achieve a consensus result. This is achieved using Dempster's rule of combination to aggregate evidences, the core idea of which is to reinforce evidences that are commonly agreed and to normalise out evidence that is in conflict. The fusion function (Dem68) is:

$$m_{12}(A) = \frac{\sum_{\forall X,Y:X\cap Y=A} m_1(X) * m_2(Y)}{1-Z} \tag{5.2}$$

where

$$Z = \sum_{\forall X,Y:X\cap Y=\varnothing} m_1(X) * m_2(Y) \tag{5.3}$$

$m_{12}(A)$ is the combined belief for a hypothesis, A, and X and Y represent all possible subsets of the frame of discernment. The numerator of the equation represents the

Classifier One	
Classification	Belief Allocation
Working	{ [*Working*, 0.95], [*Reading*, 0.02], [*Meeting*, 0.03] }
Reading	{ [*Working*, 0.10], [*Reading*, 0.50], [*Meeting*, 0.40] }
Meeting	{ [*Working*, 0.00], [*Reading*, 0.55], [*Meeting*, 0.45] }
Classifier Two	
Classification	Belief Allocation
Working	{ [*Working*, 0.60], [*Reading*, 0.30], [*Meeting*, 0.20] }
Reading	{ [*Working*, 0.10], [*Reading*, 0.90], [*Meeting*, 0.00] }
Meeting	{ [*Working*, 0.00], [*Reading*, 0.25], [*Meeting*, 0.75] }

Table 5.1: The mass function of a classifier with three possible outcomes.

combined evidence that agrees with hypothesis A, while the denominator provides a normalisation factor, where K is a measurement of conflict that represents the combined evidence that does not match A.

To illustrate this application of DST more concretely, Table 5.2 shows the calculations used to combine the output from Table 5.1, given that both report the result *meeting*. The calculation proceeds as follows (SF02):

- To calculate the combined basic probability assignment for a particular cell, we multiply the mass assignment from the associated column and row.
- Where the intersection is nonempty, the masses for a particular hypothesis from each source are multiplied.
- Where the intersection is empty, this represents conflicting evidence and is also calculated.
- For Z as defined in Equation 5.3 there are two cells that contribute to conflict represented by empty intersections. $Z = 0.41 + 0.11 = 0.52$.
- Finally, the normalised probabilities for the two nonempty intersections are calculated, as defined in Equation 5.2:

		Classifier Two		
		Working (0)	Reading (0.25)	Meeting (0.75)
Classifier One	Working (0)	0	$\varnothing + 0$	$\varnothing + 0$
	Reading (0.55)	$\varnothing + 0$	0.14	$\varnothing + 0.41$
	Meeting (0.45)	$\varnothing + 0$	$\varnothing + 0.11$	0.34

Table 5.2: Combining evidence using Dempster's rule of combination.

$$m1(Reading)m2(Reading) = (0.55)(0.25)/[1-0.52] = 0.29$$
$$m1(Meeting)m2(Meeting) = (0.45)(0.75)/[1-0.52] = 0.71$$

Limitations of Dempster's rule of combination have been identified in the literature. Most notably, Zadeh's paradox (Zad86) identifies that when sources of evidence broadly disagree, belief is disproportionately assigned to any small overlap in a way that is counter intuitive. Alternative approaches to evidence combination that aim to address this problem are discussed in detail by McKeever et al. (MY13). However, for the purposes of work described here, sources are broadly in agreement, therefore the standard rule of combination suffices.

Our use of DST at the level of classifiers confers two main advantages: Firstly, the quantification of uncertainty associated with each classifier does not need to be manually defined, but comes directly from evaluation on a training dataset; and, secondly, quantifying the uncertainty of a classifier should, to some extent, mitigate minor errors in the evaluation of each classifier. We compare our use of DST to other methods of ensemble construction in Section 7.3.5.

5.2 Genetic Ensemble Approach

The first approach developed to improve situation recognition accuracy is based on manipulating the fitness function of the genetic algorithm during the classifier generation process. The genetic ensemble involves the generation of a set of classifiers, one for each situation, with the overall result computed by combining the output of each. That is, we build a suite of classifiers over the same data, each targeting recognition of a specific situation, rather than building a single classifier to select from among all possibilities. The methodology and a worked example are outlined below.

5.2.1 Methodology

The process of constructing a genetic ensemble is as follows. Given a set of S situations, we generate $|S|$ classifiers, each treating the recognition of a different situation preferentially. For example, in the three-situation scenario of *working*, *reading*, and *meeting*, three classifiers are generated.

For each classifier to favour the precision and recall of a particular situation over the others, we modify the original fitness function (shown in equation 4.13), weighting the f-measure of the target situation, s_n, higher than the average f-measure of the

set of situations as a whole. Note that the selection of weights in construction of the fitness function has the overall effect that the genetic algorithm prioritises: *i)* the recognition of the target situation over everything else, *ii)* next, the average recognition of the remaining situations as a collective, and *iii)* finally, the simplification of the specification through the pruning of redundant statements.

$$Fitness = 10 \times FMeasure(\theta_t) + \frac{\sum_{i=1}^{n} FMeasure(\theta_i)}{n} - 0.005 \times |statements|$$

Using the classification accuracy achieved on the training dataset, we associate each possible situation classification with a probability distribution across each of the n situations, as illustrated in Table 5.1.

At runtime, we perform the situation classification task with each situation set, mapping the outputs to n probability distributions. For example, in the three-situation scenario, the output of performing the initial classification is three probability distributions, one for each of the *working*, *reading*, and *meeting* situations.

Next, we apply Dempster's rule of combination to combine the n probability distributions, as described in Section 5.1, before finally selecting the most probable result from the combined probability distribution as the output of the classification task.

5.2.2 Example

Consider the construction of a genetic ensemble for an expanded classification task involving six situations: *Break*, *Working*, *Reading*, *Coffee*, *Lunch*, and *Meeting*.

Specialised classifiers are generated for each possible situation, creating an ensemble consisting of six classifiers. Figure 5.1 depicts one particular execution of these classifiers, with the evaluated probability distributions of each depicted in red.

Table 5.3 next illustrates the application of Dempster's rule of combination to calculate a consensus result across the set of specifications (from left to right, in 5 stages). When complete, this process yields the classification of *Working*: the classification with the highest belief (95.7%).

5.3 Semantic Hierarchy Approach

The second approach developed to improve situation recognition accuracy is based on exploiting expert knowledge about how situations are related. The semantic hierarchy

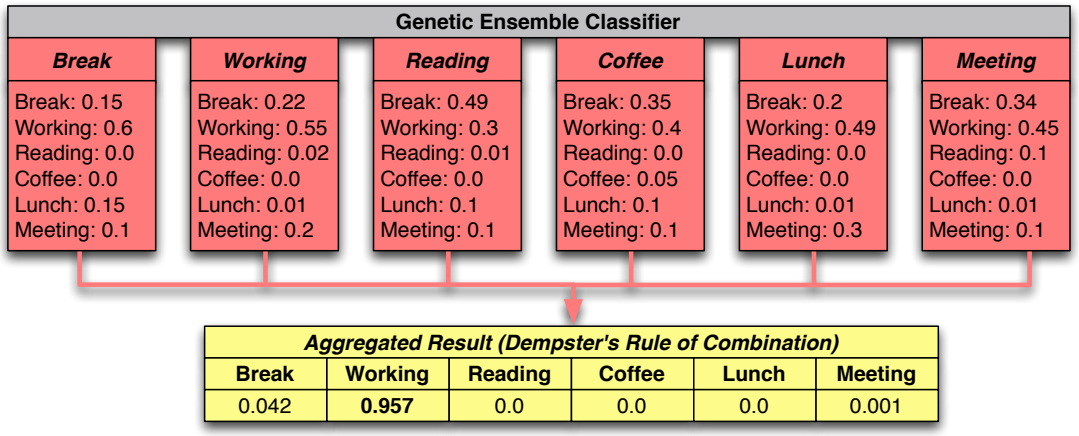


Figure 5.1: An illustration of the genetic ensemble.

Combination	Combined Classification
B&W	{ [*W*, 0.858], [*B*, 0.086], [*M*, 0.052], [*L*, 0.004], [*R&C*, 0] }
B&W&R	{ [*W*, 0.844], [*B*, 0.138], [*M*, 0.017], [*L*, 0.001], [*R&C*, 0] }
B&W&R&C	{ [*W*, 0.871], [*B*, 0.124], [*M*, 0.004], [*R&C&L*, 0] }
B&W&R&C&L	{ [*W*, 0.942], [*B*, 0.055], [*M*, 0.003], [*R&C&L*, 0] }
B&W&R&C&L&M	{ [*W*, 0.957], [*B*, 0.042], [*M*, 0.001], [*R&C&L*, 0] }

Table 5.3: Applying Dempster's rule of combination to Figure 5.1. Situation names are abbreviated to their initials.

ensemble involves the generation of multiple classifiers, exploiting the "semantic closeness" of situations to successively refine a classification. The approach begins with a high level grouping of related situations in the scenario, that is progressively refined through classifiers that focus on disambiguating those situations that are most closely related. Results are computed by following a execution path through the set of classifiers, where the output of one situation set governs the selection of the next classifier to execute.

5.3.1 Methodology

The process of constructing a semantic hierarchy is as follows. Using expert knowledge of how situations are related, either physically, or through some other shared semantic feature, we group situations into related categories and name the abstractions. For example, in the three-situation scenario of *working*, *reading*, and *meeting*, we might group together *reading* and *working* into a category of *cubicle*.

Next, using the grouped abstractions, we design the structure of a hierarchy of classifiers. At the top level, or root, of the hierarchy we place a classifier designed to distinguish

only between situations or groupings of situations at the most abstract level. At the lower levels, we design classifiers to disambiguate the situations (or groupings of situations) within individual groupings. We can design the hierarchy to an arbitrary depth as the scenario dictates.

In the three-situation scenario, two situation sets are required: One at the root level to differentiate between *meeting* and *cubicle*, and a second to differentiate between the cubicle activities *reading* and *working*.

Next, we use training data along with the standard fitness function to generate each classifier using the genetic algorithm. To generate the classifiers with grouped situations, we first transform the training data to replace all entries grouped at that level of the hierarchy with the associated group term. For example, replacing all instances of the terms *reading* and *working* with *cubicle*. This is carried out at all levels of the hierarchy.

At runtime, the root classifier is first used to classify instance data. If the returned result corresponds to a singleton situation, the classification is returned as the output. If the result corresponds to a grouping of situations, the appropriate classifier from the next level of the hierarchy is selected and executed. This process is repeated until the result of a classification is a singleton situation, at which point it is returned.

For example, in the three-situation scenario, if the classification output from the root classifier is *meeting*, it is returned as the result. However, if the output is *cubicle*, the situation set corresponding to the *cubicle* grouping is then executed to disambiguate between the two possible situations *reading* and *working*.

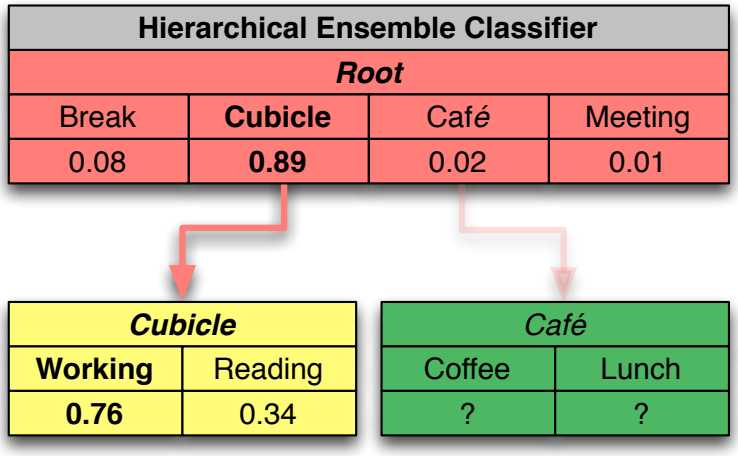


Figure 5.2: An illustration of the semantic hierarchy ensemble.

5.3.2 Example

Expanding on the above example, consider the construction of a semantic hierarchy ensemble for a classification task involving six situations: *Break*, *Working*, *Reading*, *Coffee*, *Lunch*, and *Meeting*.

To design the hierarchy, we use knowledge that coffee and lunch breaks take place in the building's café, while the activities of working and reading take place in their office cubicle. Given that each pair shares the feature of location, we create two groupings of situations to aid their disambiguation:

$$\textit{Café} = \textit{Coffee} \cup \textit{Lunch}$$
$$\textit{Cubicle} = \textit{Working} \cup \textit{Reading}$$

Using these groupings, we design the two-level structured ensemble illustrated in Figure 5.2. At the root of the ensemble, a classifier is trained to disambiguate between *Break*, *Meeting*, *Café*, and *Cubicle*, while at the lower level two further classifiers are trained to disambiguate exclusively between *Working* and *Reading*, and *Coffee* and *Lunch*.

Figure 5.2 also illustrates one particular execution pathway through these classifiers, with the evaluated pathway highlighted by the opaque red arrow. In the depicted instance, evaluation of the root classifier yields the output *Cubicle*. This in turn necessitates the evaluation of a specification-set at the second level corresponding to the *Cubicle* grouping, the execution of which yields the *Working* situation (with a known evaluated accuracy on training data of 76%).

5.3.3 Handling Idle Time Slices

Some datasets, including van Kasteren's House A dataset (vKNEK08), which we use later as part of the evaluation, include the notion of idleness: periods of time over which any non-specific situation occurs. The literature describes approaches to situation recognition that both include and ignore idleness, and we later evaluate this work taking both into consideration.

In both a standalone classifier and in the genetic ensemble, classification of *idle* situations can only be achieved by treating idle time slices as a regular situation. However, given that idleness does not describe a fixed phenomenon, but the set of all activities not otherwise specified, generation of a concise and intelligible specification set or decision tree to represent idleness can be difficult.

The semantic hierarchy approach more easily accommodates the notion of idleness, through incorporating its detection within various classifiers in the hierarchy as a filter-like construct. We have designed a two-fold approach to achieve this. Firstly, all singleton specifications are re-designated as a paired grouping consisting of the singleton and the *idle* situation:

$$Situation' = Situation \cup Idle$$

Secondly, classifiers are generated at each of the leaf nodes to support differentiation of the singleton from the *idle* situation. This process can be used to redefine the standalone classifier as a two-level classification task, or can be applied to the semantic hierarchy already constructed. The latter approach is depicted in Figure 5.3.

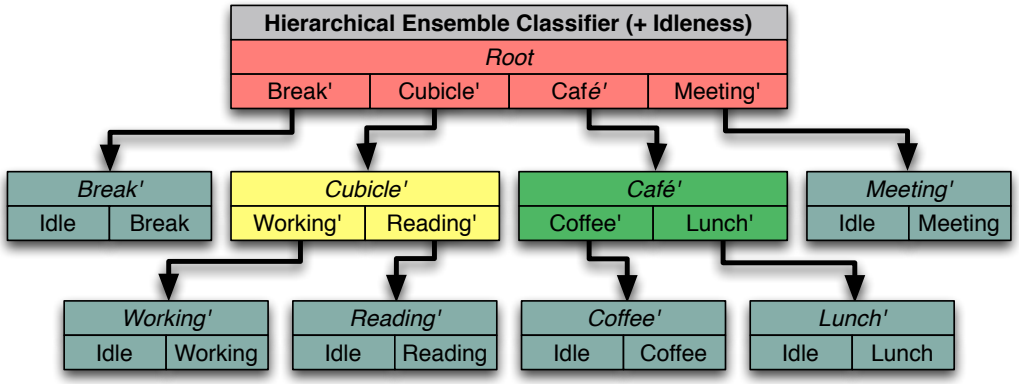


Figure 5.3: An illustration of the semantic hierarchy ensemble, extended to handle idle time slices.

This approach is based on the assumption that it is easier to generate multiple classifiers capable of profiling idleness when compared to singleton situations, than to have a single model that attempts to differentiate idleness from the set of all situations. Further to this, it allows the higher layers of the classifier to be trained only on situation specific data, and offers the potential to customise how the *idle* situation is disambiguated at the level of each situation.

5.4 Summary

This chapter discussed approaches to constructing ensembles with the goal of obtaining more accurate and reliable decisions than a standalone model and, in the case of the semantic hierarchy, to simplify the recognition process through modularising the structure of the recognition task.

We began with a primer on the use of Dempster-Shafer theory, a mathematical theory of evidence for combining information from multiple sources, that is popular in the design of pervasive- and sensor-systems. Its ability to preserve uncertainty and the possibility of assigning confidence to sets of situations, not only situations individually, plays a later role in consensus finding.

Next, we outlined two different approaches to constructing ensembles: the genetic ensemble and the semantic hierarchy. For each we overviewed the underlying concepts, design and implementation methodology, and gave an example illustrating their use.

The genetic ensemble is based on manipulating the fitness function of the genetic algorithm during the classifier generation process. It involves the generation of a set of classifiers, one for each situation, with the overall result computed by combining the output of each.

The semantic hierarchy exploits expert knowledge about how situations are related. It involves the generation of multiple classifiers to successively refine a classification from abstract groupings to specific situations. Results are computed by following an execution path thorough multiple set of classifiers, where the output of one situation set governs the selection of the next classifier to evaluate. A further contribution explores how the semantic hierarchy is extended to support detection of 'idleness'—periods of time over which non-specific situations occur.

In the next chapter we prepare datasets for the evaluation of the techniques developed in this and earlier chapters, through mapping them to our information space model.

CHAPTER

SIX

DATASET PREPARATION

To this point we have introduced a reusable top-level ontology model that provides a common substrate for developing domain and application ontologies for pervasive environments, two hybrid situation recognition models based on the constructs of this model, and approaches to ensemble classification that aim to provide improved situation recognition accuracy and intelligibility. Before evaluating these algorithms in Chapter 7, this chapter describes the process of readying a dataset for application of these techniques. We refer back to 3.8 for more general discussion surrounding the engineering effort this process requires.

The evaluation requires sensor datasets that are annotated with ground truth (situations asserted to be occurring at the time sensor readings are recorded) and that contain complex enough situation and sensor arrangements so as to support the building of concept hierarchies.

To these ends we have selected five datasets and grouped them into two categories:

- **Smart-office**
  - The **CASL** dataset (MYCD09a) captures the workday situations and sensor traces associated with a doctoral student in a laboratory setting.
  - The **Ink-12** dataset captures the workday situations and sensor traces of a doctoral student in a home-office setting; collected for this research as a study modelled on the CASL dataset.
- **Smart-home**
  - The **van Kasteren** datasets (House A, B and C) (vKNEK08; vKEK10a): Three datasets that capture the sensor traces and typical daily activities in three single occupancy houses.

While the datasets we have selected describe smart-environments, they have been chosen primarily because they are marked with ground truth, and recur in the literature, supporting comparative evaluation of the techniques we have developed. We believe the same set of techniques described in this thesis can be applied to other sensing domains, such as wearables, where a representative model of the information space can be constructed.

We overview the standard methodology for preparing a dataset in Section 6.1, before applying this methodology to each of the smart-office and smart home datasets in Section 6.2 and Section 6.3 respectively. A summary is presented in Section 6.4.

6.1 Methodology

To be useful, a situation recognition dataset must provide, as a minimum requirement, two features: timestamped traces from sensors deployed in an environment, and a ground truth that describes the occurrence and duration of situations that occur therein. Both are necessary in order to construct a snapshot of the state of the sensors at any given time, and associate it with the occurrences of situation in preparation for the application of learning (or in the case of unsupervised techniques where labelling of situations is not required, support their evaluation).

Collecting accurate ground truth annotations can be a challenging and expensive task, and is prone to error. In the CASL, Ink-12, and House C datasets, subjects recorded their activities in handwritten diaries, while in the House A and House B datasets subjects provided annotations via bluetooth headsets in concert with speech-to-text recognition. Both approaches risk subjects forgetting to annotate activities, while handwritten diary timestamps are, naturally, less accurate than their headset-captured equivalents. Alternative techniques may yield better results; one such possibility is camera tracking, however the accuracy gain comes as a trade-off with the cost to purchase and install the equipment and the typical need to manually transcribe the footage to annotations (TIL04; CYW05). In all cases, the *Hawthorne or observer effect*, whereby subjects may act differently given the knowledge that they are being 'watched', may affect how accurately the collected data reflects the subject's activities in an unmonitored situation (MWI+07).

Given an annotated dataset, four processes must be carried out to prepare it for use with the situation recognition techniques described in this thesis: *situation identification*, *situation model construction*, *context model construction*, and *sensor data to context statement mapping*. These tasks require varying levels of 'expert' knowledge which

may come from various sources including sensor manufacturers, sensor installation engineers, and the observed subjects. We outline each process below.

Situation Identification Here, the task is to enumerate the set of situations that the recognition technique must differentiate. This may be defined before data collection begins, or may be derived from a ground truth collected in free form. Knowledge of the target environment should be used to restrict the set of situations to those deemed likely determinable from the environment's sensing capability. For example, intuitively the situation *watch_tv* should only be included in the enumeration of candidate situations if at least some of its associated characteristics (for example, the television being switched on, or the subject being sat in a particular seat) can be directly sensed or inferred.

Situation Model Construction After the set of situations thought to be discernible has been identified, the next step is to explore if any of the relations defined in Section 3.3 can be used to connect them, either directly, or through the construction of an intermediate concept. For example, the identified concepts of *in_car* and *in_bus* might be united under the encompassing concept of *in_vehicle*. These connections play a role in structuring any hierarchical ensemble.

Context Domain Model Construction Next, and again following the model described in Section 3.3, concept hierarchies are constructed for each information domain for which sensors provide information. For example, Figure 6.1 illustrates one possible encoding of time, with the finest grained concepts representing hours of the day (*H0-H23*), and the more abstract terms representing labelled periods of the day first in 3, and then 6 hour durations. For clarity, the adjacency relations between the start and end of the day are not depicted. Depending on the situations to be recognised, other temporal encodings may be more useful, for example, a division based on the concepts *before_office_hours*, *office_hours*, and *after_office_hours*.

Sensor Data to Context Statement Mapping Finally, the raw sensor data is mapped to context statements using the defined concept hierarchies. This occurs in three (potentially overlapping) steps: In the first step, all sensor data corresponding to a particular domain is resolved to a single value. In the second step, a mapping function projects the result of the first step onto the concept model. In the third step the corresponding context statement is constructed.

The resolution of multiple observations to a single value may be achieved in several ways. In the simplest case, all observers agree on the value and therefore the resolution

is trivial. If the reported values differ, it may be the case that the mapping function evaluates to the same concept in each case, again resolving the conflict. If this is not the case, a number of strategies may be adopted: selecting the most recent data, averaging reported values (if numeric), or selecting the majority value among many.

The mapping function may be realised as a one-to-one relation, whereby each possible sensor value is mapped to a single concept in its associated domain model, or as a one-to-many allocation of confidences across a set of concepts; for example, mapping the time '13:24pm' to the concept *H13* with confidence 1, or the temperature '18° Celsius' to the concepts *Cold* and *Hot* with confidences 0.7 and 0.3 respectively.

Finally, the context statement is constructed from the resolved values. For example, [*:system*, **:hasTime**, *:H13*], or [*:kitchen*, **:hasTemperature**, *:Hot*].

6.2 Smart Office Datasets

In this section we apply the methodology outlined above to construct a common, shared model for both of the smart office datasets.

6.2.1 Situations in the Smart Office Datasets

Both smart office datasets describe six situations that are identifiable from their available sensors: *Working at Computer*, *Reading at Desk*, *In Meeting*, *Taking a Coffee Break*, *Taking a Lunch Break*, and *Taking an Informal Break*.

6.2.2 Situation Concept Model

To the set of identifiable situations, we add three additional situations to form a hierarchy: *Working In Cubicle* – which encompasses both *Working at Computer* and *Reading at Desk*, *Taking a Cafe Break* – which includes in scope *Taking a Coffee Break* and *Taking a Lunch Break*, and *In Office Situation*, which is the super-situation of all others represented, and trivially always holds true.

This set of nine situations forms the situation concept hierarchy shown in Figure 6.2.

This hierarchy is used for all experiments carried out using the smart office datasets described in Chapter 7.

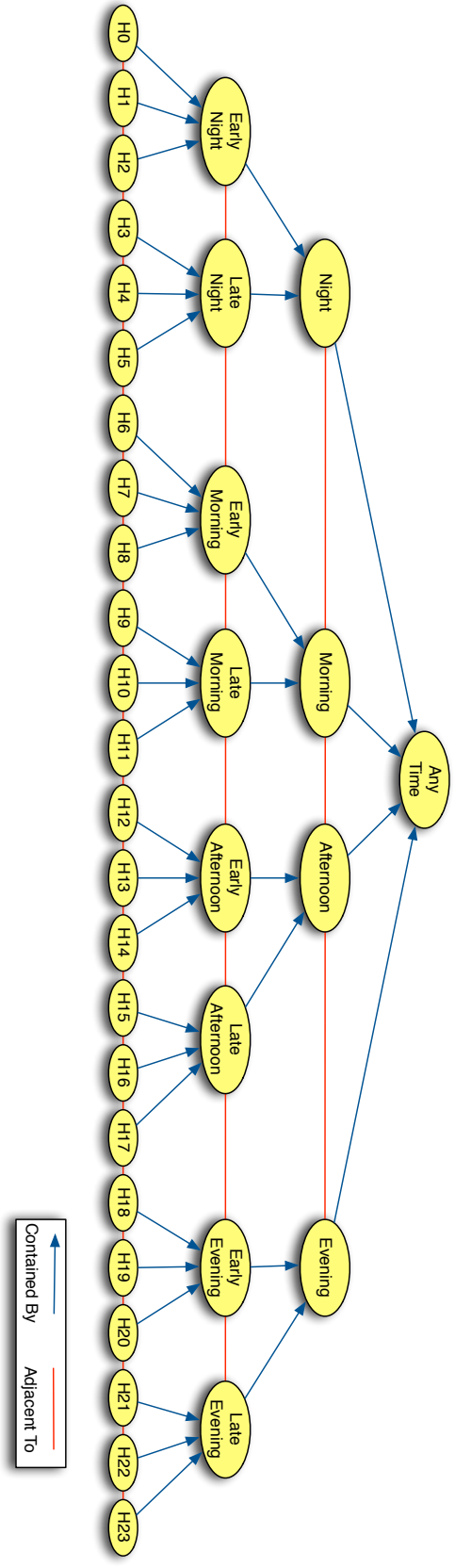


Figure 6.1: The temporal concept model applied to all the datasets.

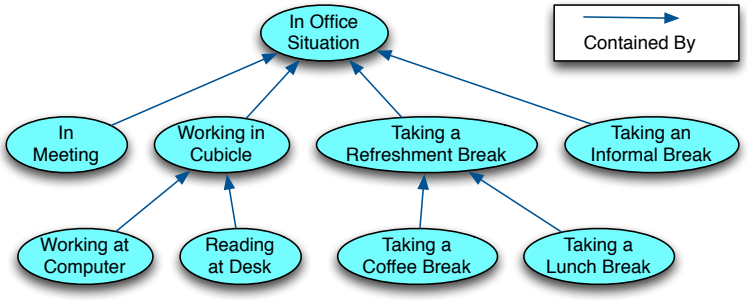


Figure 6.2: The situation concept hierarchy associated with the smart office datasets.

6.2.3 The CASL Dataset

The Complex and Adaptive Systems Laboratory (CASL) dataset (MYCD09a) was collected within a research laboratory at University College Dublin, Ireland. It captures a doctoral student's daily routine over a one week period.

Sensors and Data Collection. The CASL dataset contains three sensing mediums that capture the subject's schedule, location within the building, and computer activity. In more detail these are:

- Ubisense, an ultra-wideband tag-based positioning sensor network that tracks the subject's real-time location over two floors of the CASL building. The sensor outputs a stream of tuples of the form $< T, L, X, Y, Z >$, corresponding to a timestamp (T), a tag identifier, which is mapped to a subject (L), and the sensed 3D coordinate position of the tag (X, Y, and Z). The coordinates are measured in metres from an origin point on the south-east corner of the third floor of the building.
- An 'activity sensor' that detects all of the subject's interactions with the keyboard and mouse of their desktop computer, and records these along with the time elapsed since the last observed activity. As output, the sensor indicates one of two states, 'active', or 'inactive', with a tuneable temporal threshold to discount rapid changes between state.
- A 'calendar sensor' that extracts data from the subject's *Google calendar*, and records scheduled events along with their extent. In this dataset, only meeting events are provided.

As reported in (McK11), the subject switched on the sensors at the beginning of the work day and off at the end. During the day all occurring situations were manually recorded in a spreadsheet along with their start and end times.

Domain Concept Models and Mappings Figure 6.3 and Figure 6.4 illustrate the concept hierarchies used to model computer activity, meetings scheduled and location data in the CASL dataset. The activity and meeting models are straightforward single concept representations which may or may not hold true at any point in time. These are *Active* for the computer activity monitor, and *Scheduled* for the meeting state.

(a): The computer activity model. (b): The scheduled meeting model.

Figure 6.3: The conceptual models to which the data from the CASL activity and calendar sensors are mapped.

The location model, to which the Ubisense sensor output is mapped, consists of a set of symbolic names that describe locations in the CASL building, for example, *Cubicle A*, *Cafe*, and *Third Floor*. These concepts are mapped directly from those used in the source dataset.

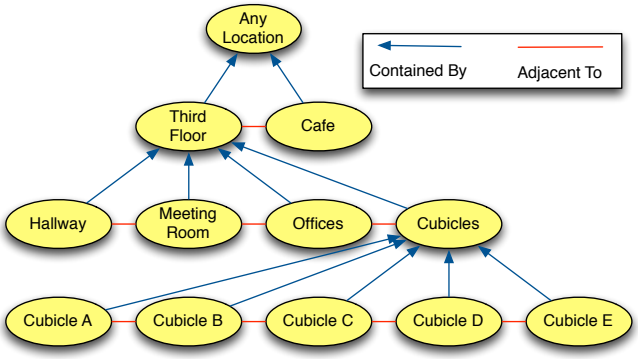


Figure 6.4: The conceptual models to which data from the CASL Ubisense sensor is mapped.

Ubisense positions are abstracted to 2D regions by using a bounding box to capture the measuring error in the positioning system. Known precision values for the x and y coordinates (here, 3.33 metres in the x-axis, and 2.2 metres in the y-axis) are used to extend a bounding box over an area of the map from the sensed coordinate. Each region covered by the bounding box is then assigned a confidence value proportional to the area of the bounding box that lies in that region. This is illustrated in Figure 6.5, where although the coordinate lies in *Region A*, the bounding box extends over *Region B*—representing that, due to error, the true position of the person is likely to be anywhere in the shaded region. However, as the shaded region lies more in *Region A* than *Region B* we assign confidences to reflect this: In this case 0.8 and 0.2 respectively.

The confidence values assigned to each concept in the hierarchy are propagated according to the processes described in Section 3.6.

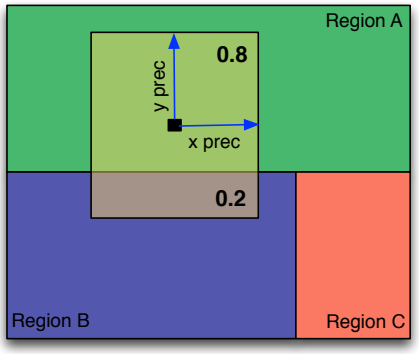


Figure 6.5: The assignment of certainty values to symbolic regions based on evaluated sensor precisions. Reproduced from (McK11).

6.2.4 The Ink-12 Dataset

The Ink-12 dataset has been collected as part of this research. It has been designed as a week long repeat study of the CASL dataset (i.e., collecting the information about the same set of situations), with two goals in mind. Firstly, to collect more accurate location data than the CASL dataset, and secondly, to demonstrate the scope for ontology reuse between the two datasets.

Sensors and Data Collection Similarly to the CASL dataset, the Ink-12 dataset contains three sensing mediums that capture the subject's schedule, location within the building, and computer activity. In more detail these are:

- A set of five Passive Infrared (PIR) sensors deployed over rooms of the building (see Figure 6.6) to track the subject's presence or passing (this suffices, as the building has a single occupant). The sensor outputs a stream of tuples of the form $< T, S, V >$, corresponding to a timestamp (T), a symbolic identifier for the location in which it is deployed (S), and a value (V) representing one of two state changes in the PIR: activated (presence detected) and deactivated (no presence detected for 60 seconds). *Marmitek SE13* wireless PIRs were used in conjunction with a *Marmitek CM15 PRO* base-station connected to a laptop to perform logging.
- An 'activity sensor' that detects all of the subject's interactions with the keyboard and trackpad of their laptop, and records these along with the time elapsed

since the last observed activity. As output, the sensor indicates one of two states, 'active', or 'inactive', with a tuneable temporal threshold to discount rapid changes between state.

- A 'calendar sensor' that extracts data from the subject's *Google calendar*, and records scheduled meetings along with their extent (any event with "meeting" in the title). For the purposes of this study, we considered scheduled conference calls as meetings, with the subject moving to a seat in the bedroom for their duration.

Figure 6.6: In-situ *Marmitek SE13* passive infrared sensors used in the Ink-12 dataset.

The data was collected in a home-office environment, with sensors turned on at the start of the working day and turned off at the end of the day. Activities were recorded in a hand written diary, and transferred later to a spreadsheet.

Domain Concept Models The Ink-12 dataset uses the same concept models as in the CASL dataset, shown in Figure 6.3, for modelling computer activity and scheduled meetings. The location model, to which the PIR sensor output is mapped, is illustrated in Figure 6.7. The building consists of five rooms, with the kitchen split into cooking and seating areas. With the exception of the bathroom, which is not instrumented, a PIR corresponds to each leaf of the location hierarchy. To maintain a correspondence with the location specific situations in the CASL dataset, Coffee and lunch breaks were taken in the kitchen, meetings (Skype calls) were taken in the bedroom, and working and reading activities took place in the office.

6.2.5 A Common Concept Model

To enable sharing and reuse of information space models between the CASL and Ink-12 datasets requires that the concepts models we develop have a correspondence. In the

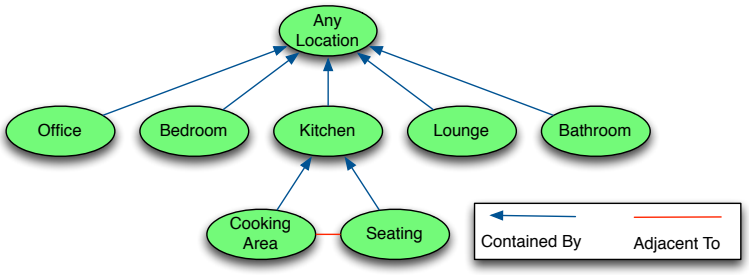


Figure 6.7: The Ink-12 physical location model.

case of the activity and meeting scheduled domains this is trivial, as they are identical. However, this is not the case for the location domain where both buildings have different structural layouts in terms of the type and number of regions and the number of floors.

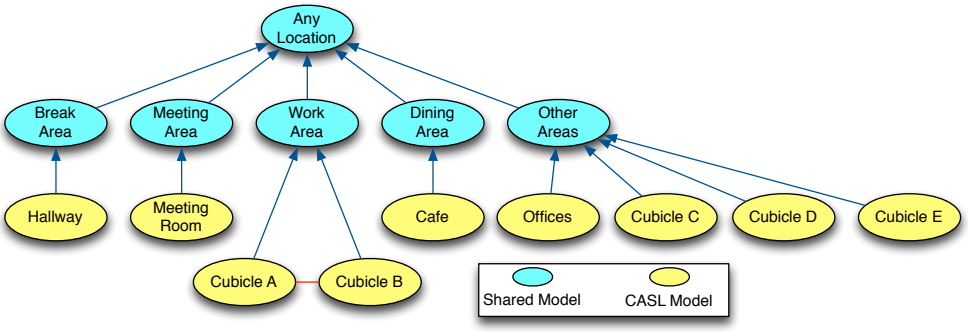
We address this by revising the location concept models to account for each location's function, rather than their spatial containment relationships. Through doing this, we define a common semantic model that may be extended in each environment to represent the spatial regions relevant to each task. Furthermore, we hypothesise that using expert knowledge to structure information in this way may boost the recognition capability of the final model.

Based on this principle, Figure 6.8 illustrates the common semantic model we use for the situations of interest in the smart office environments, derived using "expert knowledge" along with the spatial extensions for both the CASL and Ink-12 datasets. Here we can see, for example, that the CASL *Meeting Room* and Ink-12 *Bedroom* concepts both correspond to the general concept of a *Meeting Area*, that the Ink-12 concepts of *Cooking Area* and *Seating* correspond to the common *Dining Area* concept, and that *Cubicle A* and *Cubicle B* in the CASL correspond to the concept of *Work Area*—this choice is to account for the high degree of imprecision in differentiating between both areas using Ubisense technology.

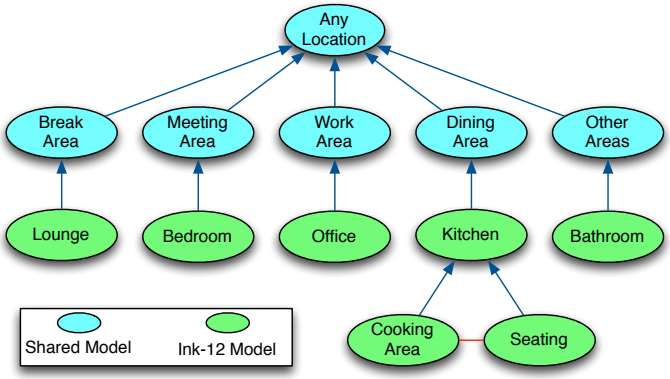
This common model can serve as the basis for developing further smart office environment models, where similar sets of situations are concepts need to be represented, reducing the construction effort required.

6.3 Smart Home Datasets

In this section we apply the methodology described at the start of the chapter to construct a model for the three smart home datasets.



(a): The CASL semantic location model.



(b): The Ink-12 semantic location model.

Figure 6.8: The CASL (yellow) and Ink-12 (green) location models defined as extensions of a common semantic model (blue).

6.3.1 Situations in the Smart Home Datasets

The smart home datasets include a set of eight situations: *Left House*, *Using Toilet*, *Using Shower*, *Sleeping*, *Preparing Breakfast*, *Preparing Dinner*, *Preparing Drink*, and *Idle*.

6.3.2 Situation Concept Model

As with the smart home datasets, we add three situations to the set of those identifiable: *Eating/Drinking* – which is characterised by the three situations *Preparing Breakfast*, *Preparing Dinner*, and *Preparing Drink*; *Hygiene* – which includes *Using Toilet* and *Using Shower*, and *Situation*, which is the super-situation of all others represented, and trivially always holds true.

This set of ten situations forms the situation concept hierarchy shown in Figure 6.9.

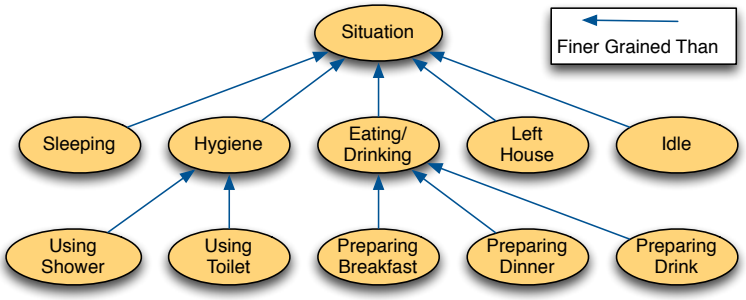


Figure 6.9: The situation concept hierarchy associated with the smart home datasets.

This hierarchy is used for all experiments carried out using the smart home datasets described in Chapter 7.

6.3.3 The House A Dataset

The House A dataset tracks the daily activities of a 26 year old man in a single occupancy residence. The dataset was collected over a period of 28 days by van Kasteren et al. (vKNEK08), logging 2120 sensor events and 245 activities.

Sensors and Data Collection The House A dataset uses only digital reed switches, each mounted to a RF *Monolithics DM1810* module to form a wireless network. The sensor modules are attached to 14 objects of interest across rooms in the house, each producing a binary reading to indicate whether or not a sensor is firing. The instrumented objects are:

- In the kitchen: *microwave*, *groceries cupboard*, *plates cupboard*, *freezer*, *fridge*, *pans cupboard*, *cups cupboard*, *washing machine*, and *dishwasher*.
- In the toilet: *door*, and *toilet flush*.
- In the bathroom: *door*.
- At the front door: *door*.
- In the bedroom: *door*.

The wireless sensor network is connected to a base-station to enable logging of the data. Activity data was collected via a bluetooth headset worn by the subject, activated by button press, with speech recognition applied to the spoken annotation in order to log it.

Domain Concept Models and Mappings Figure 6.10 illustrates the concept hierarchies we constructed to model the firing of the above set of sensors, grouped by the room of the house in which each is found. The bedroom, front door, and bathroom models shown in *a) b)*, and *c)* respectively are concerned only with single sensors, while the toilet model, shown in *d)*, encompasses the two sensors located in that room.

In the kitchen model, shown in *e)*, we introduce intermediate concepts into the hierarchy to group sensors attached to objects with similar usage semantics under the two categories of *food storage* (*groceries cupboard*, *fridge*, and *freezer*), *heating* (*pans cupboard*, access to which we assume precedes a heating-related action, and *microwave*).

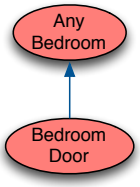
Several possible interpretations of the information space model, discussed in Section 4.2, are used to evaluate statements that involve these concepts.

6.3.4 The House B Dataset

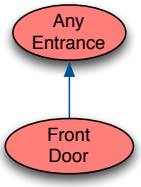
With a similar setup to House A, the House B dataset tracks the daily activities of a 28 year old male in a single occupancy residence over a period of 14 days (KEK11).

Sensors and Data Collection House B is instrumented with a larger and more diverse set of sensing modalities than House A, including pressure mats for detecting the occupant sitting and lying down, mercury contacts to detect the movement of objects, and passive infrared to detect motion. The instrumented objects are:

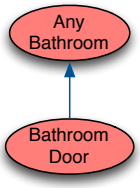
- In the kitchen: *microwave*, *groceries cupboard*, *plates cupboard*, *fridge*, *cutlery drawer*, *stove lid*, *toaster*, and a *PIR sensor*.



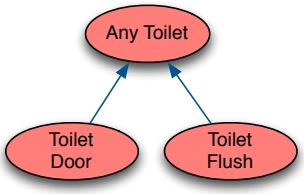
(a): The bedroom concept model.



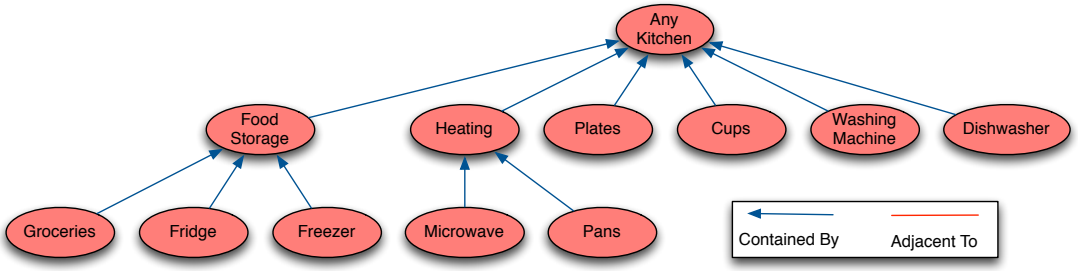
(b): The front door concept model.



(c): The bathroom concept model.



(d): The toilet concept model.



(e): The kitchen concept model.

Figure 6.10: The conceptual models to which the data from the House A sensors are mapped.

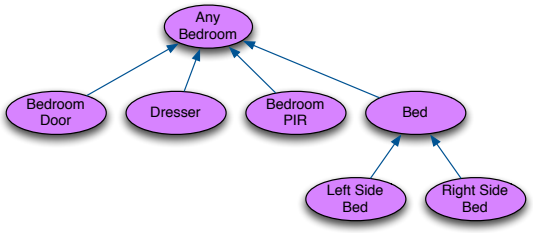
- In the toilet: *door*, *toilet flush*, *sink (float)*, and a *PIR sensor*.
- At the front door: *door*.
- In the bedroom: *door*, *left and right side of bed (pressure mats)*, *dresser*, and a *PIR sensor*.

Also included in the dataset are sensors for a seat pressure mat, balcony door, and window. However the documentation for the dataset is unclear as to the location of these sensors within the house. However, these, and a pressure sensor associated with a piano in the lounge, are not critical to the set of situations under investigation.

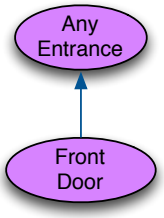
Activity data was collected via a handwritten diary, written on sheets of paper left in the places where the set of activities are performed. Start and end times of the activities were recorded from the participant's watch.

Domain Concept Models and Mappings Figure 6.11 illustrates the concept hierarchies we constructed to model the firing of the above set of sensors, grouped by

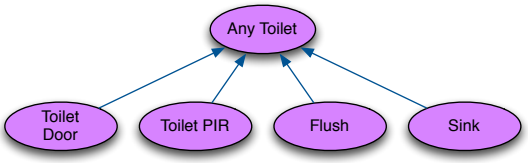
the room of the house in which each is found. The bedroom, front door, toilet, and kitchen models are shown in *a) b)*, *c)* and *d)* respectively.



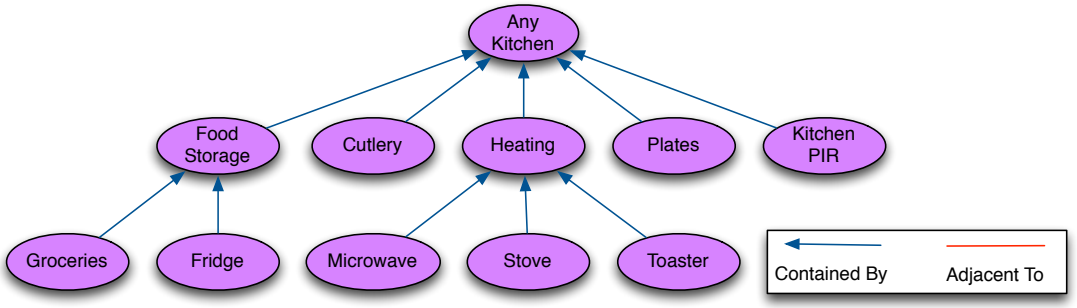
(a): The bedroom concept model.



(b): The front door concept model.



(c): The toilet concept model.



(d): The kitchen concept model.

Figure 6.11: The conceptual models to which the data from the House B sensors are mapped.

As with the kitchen model from House A, the bedroom and kitchen models shown in *a)*, and *e)* introduce intermediate activities into the hierarchy to group sensors attached to objects with similar usage semantics.

6.3.5 The House C Dataset

Finally, the third house in the dataset, House C, tracks the daily activities of a 58 year old male, also in a single occupancy residence over a period of 19 days (KEK11). This house differs from the others by virtue of being split over two floors, with each floor having a toilet (i.e., multiple rooms in which the same situation can take place).

Sensors and Data Collection House C is instrumented similarly to House B, using reed switches, pressure mats, mercury contacts and passive infrared motion sensors: The instrumented objects are:

- In the kitchen: *groceries cupboard*, *cup and bowl cupboard*, *herb and plate cabinet*, *fridge*, *freezer*, *food scraps bin*, *cutlery drawer*, *pots and pans cupboard*, *microwave*, *drawer with keys to the backdoor* and a *PIR sensor*.
- In the upstairs toilet: *toilet flush*.
- In the downstairs toilet: *door*, and *toilet flush*.
- In the bathroom: *left and right swing doors*, *sink (float)*, and a *bathtub PIR*.
- At the front door: *door*.
- In the bedroom: *door*, *left and right side of bed (pressure mats)*, and *dresser*.

Also included in the dataset is a pressure sensor for the couch in the lounge, although these are not useful to the set of situations under investigation.

Activity data was collected using the same bluetooth headset approach as described for House A.

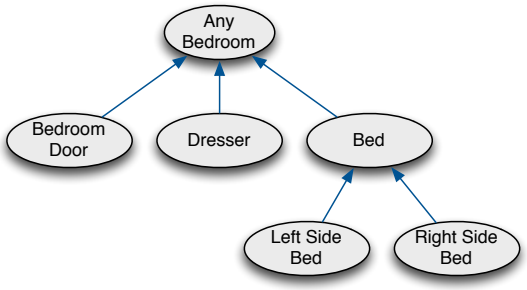
Domain Concept Models and Mappings Figure 6.12 illustrates the concept hierarchies we constructed to model the firing of the above set of sensors, grouped by the room of the house in which each is found. The bedroom, front door, toilet, and kitchen models are shown in *a) b)*, *c)* and *d)* respectively.

As with the kitchen model for House A, the bedroom and kitchen models shown in *a)*, and *e)* introduce intermediate activities into the hierarchy to group sensor-attached-objects with similar functions.

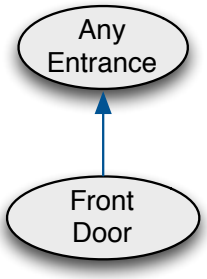
6.3.6 A Common Concept Model

As with the smart office models, we extract a common conceptual model between House A, B, and C to support sharing and reuse, the development of which is a straightforward process.

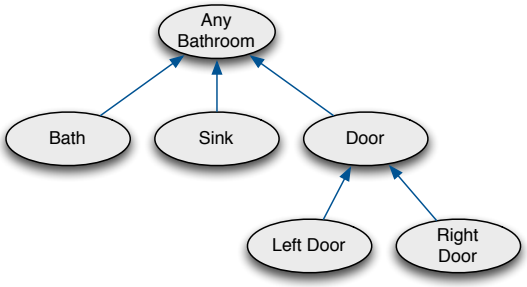
The single-concept model for the front door is shared in all three datasets, with little work required to define a common semantic model for the remaining sensors in the datasets. To illustrate the process, the common semantic models for the bedroom and



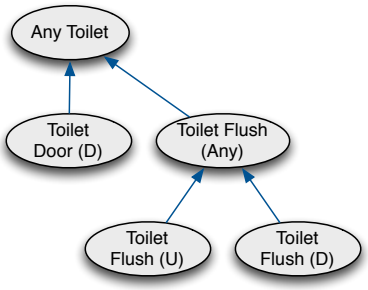
(a): The bedroom concept model.



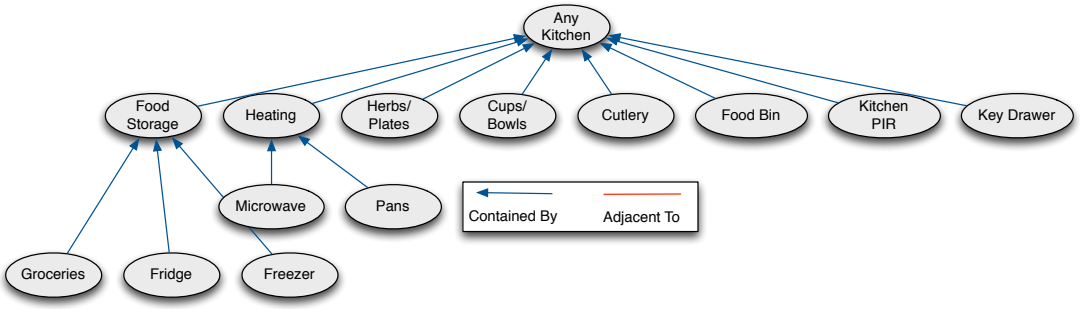
(b): The front door concept model.



(c): The bathroom concept model.



(d): The toilet concept model.



(e): The kitchen concept model.

Figure 6.12: The conceptual models to which the data from the House C sensors are mapped.

kitchen are depicted in Figure 6.13 and Figure 6.14 respectively. In each case the common concepts belonging to the upper ontology are coloured blue, with specific concepts from each individual model coloured red, purple, and grey. Equivalent concepts are denoted by an orange connection. The models for the other rooms in the houses are constructed similarly.

In the depiction of the bedroom model we can see that House B and C share a similar instrumentation and correspondence between sensors, with House A only supplying one sensor. The kitchen model is more diverse, with the majority of sensors corresponding to six main categorisations, and a few remaining sensors with no sub-categorisation, such as House C's food scraps bin and House A's dishwasher directly associated to the root concept.

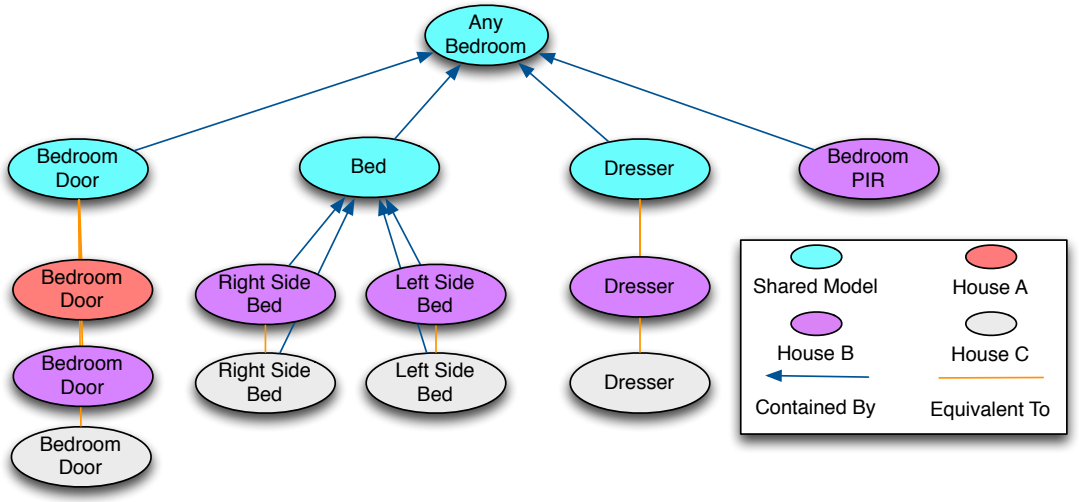


Figure 6.13: The House A (red), House B (purple) and House-C (grey) bedroom models defined as extensions of a common semantic model (blue).

The contents of two of the instrumented cupboards in House C, *Cups/Bowls* and *Herbs/Plates*, relate to multiple concepts and could have been placed differently in the shared model than shown. In the first case, as the use of a bowl would suggest the triggering of another sensor in the process of filling it, and as this does not necessarily hold for use of the cup, the decision was taken to treat this concept as equivalent to *cup*. In the second case, as plates are likely to be used more frequently than herbs (and as the use of herbs in cooking will likely trigger other sensors, the dual-concept is treated as equivalent to the *plate* concept.

This common model can serve as the basis for developing further smart home environment models, where similar sets of situations and concepts need to be represented, reducing the construction effort required.

6.4 Summary

This chapter illustrated the key steps in the process of preparing a dataset for the application of the situation recognition techniques described in this thesis.

The four stages of this process – *situation identification*, *situation model construction*, *context domain model construction*, and *sensor data to context domain model mapping* – were applied to five selected datasets. The individual datasets were grouped into two categories – *smart-home* and *smart-office* – with common situation sets and conceptual models built to support sharing and reuse across between these environments, and across other smart-home and smart-office environments with similar recognition goals.

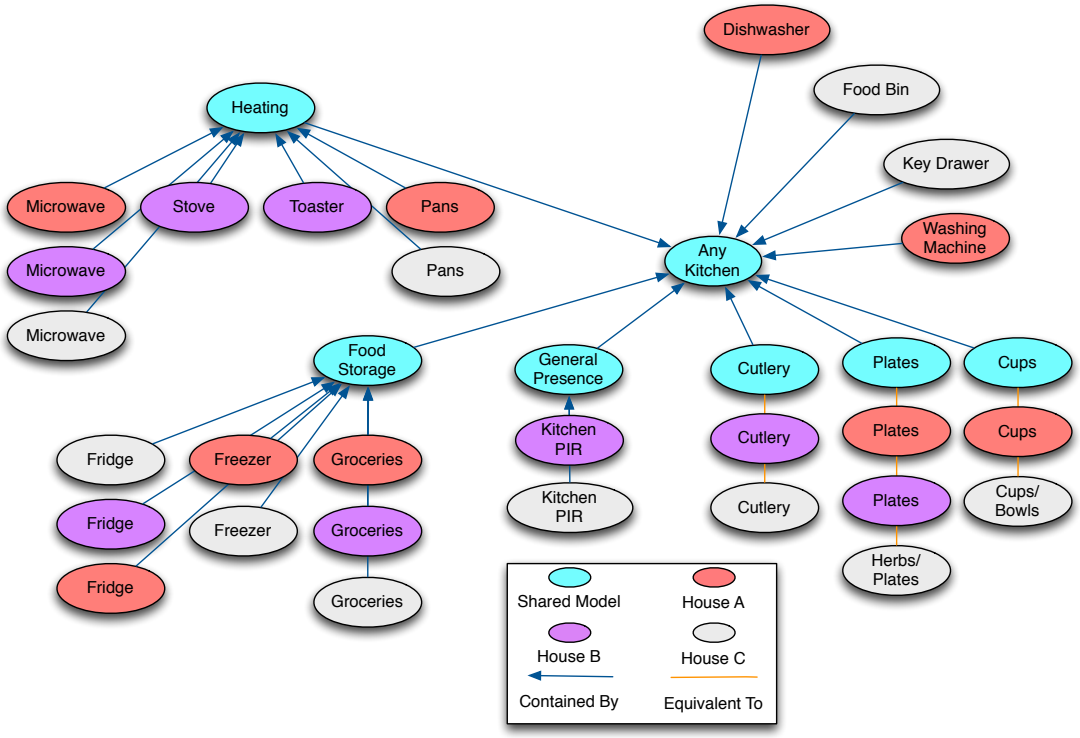


Figure 6.14: The House A (red), House B (purple) and House-C (grey) kitchen models defined as extensions of a common semantic model (blue).

In the next chapter we use the models developed here to evaluate the techniques developed in this thesis.

CHAPTER

SEVEN

EVALUATION

7.1 Introduction

The previous chapter described the process of readying several datasets for representation using the mathematical information space model developed in Chapter 3, and application of situation recognition techniques described in Chapters 4 and 5.

In this chapter we seek to examine the claims made in the hypothesis of this thesis, namely:

1. A top-level ontology with associated reasoning framework can simplify domain and application model development for pervasive environments by providing a uniform modelling approach to infer and relate new knowledge from that which is specified, in a principled way;

2. The semantic constructs of the ontology model provide an expressive basis from which to learn situation recognition models that display comparable performance with more complex machine learning models, while retaining intelligibility; and,

3. The hybrid semantic model and learning-based approach can be further exploited to automatically construct ensemble classifiers with either improved recognition accuracy, intelligibility or both;

We argue that we validated the first of these claims in the discussion section of Section 3.8. In this chapter we focus on examining the extent to which the situation recognition techniques described in this thesis satisfy the remaining claims. To validate these claims we primarily use the CASL (MYCD09a) and House A (vKNEK08) datasets, which afford the widest opportunity for comparison with other published work in the area.

Section 7.2 outlines the experimental procedure for the evaluation. Next, Section 7.3 and Section 7.4 present the experimental results obtained from the CASL and House A datasets, before we present results obtained using the additional datasets in Section 7.5. Section 7.6 assesses the benefits and limitations of the approach, before we conclude this chapter in Section 7.7 by summarising the results obtained from each set of experiments and draw some conclusions.

7.2 Experimental Setup

Unless explicitly stated otherwise, we use a standard approach to conduct the experiments across all the datasets. Here we present the measurements we use to quantify the performance of each situation recognition technique, the experiment methodology, and the makeup of the software framework that generates the specifications and runs the experiments.

7.2.1 Evaluation Parameters

In presenting the evaluation results, we use the following measurements:

- Average precision: The ratio of the number of times a recognition techniques correctly infers a situation to the total number of times the situation is inferred, averaged over each situation in the dataset.
- Average recall: The ratio of the number of times a situation is correctly identified to the number of times the situation occurs in the dataset, averaged over each situation in the dataset.
- Average F-measure: The mean of the individual F-measures of the situations in the dataset.
- Weighted F-measure: Similar to the average F-measure, but instead of weighting the f-measure of each situation equally, each f-measure is weighted proportionately to the situation's occurrence ratio in the dataset.
- Total Proportion: The proportion of test instances identified correctly across all folds in the dataset.

For the majority of the experimental results in this chapter the average F-measure and total proportion are selected as the primary measures. The average F-measure

provides a suitable measurement for assessing the recognition capability of a model considering the identification of each situation in a dataset to be equally important. The total proportion measurement is useful to illustrate the overall percentage of instances in the dataset that the model identifies correctly, and serves as an indication of the overall percentage of time that a model can be expected to identify situations correctly, assuming the dataset reflects the relative occurrence frequencies of each situation.

Going beyond recognition accuracy, we adopt *model size* as a metric for assessing a model's intelligibility. We define model size as the number of context statements that occur in a set of situation specifications, or in a decision tree (the tree's size).

We do not consider intelligibility from the context of a subjective measure of that which might be obtained, for example, by performing a user study. Instead, we take the view here that a comparison of model size provides a measurement of the *relative* intelligibility of two models. That is, given two similarly performing models, we assume that it is more likely that the specification set or decision tree with smaller size will be more easily understood by a developer. Therefore, the more intelligible a poorly performing model is, the more likely it is that a developer will be able to determine the cause(s) of any discrepancies between observed and expected behaviour through a visual inspection of the generated model (see Section 4.4). We illustrate this idea with an example in Section 7.3.2.

7.2.2 Evaluation Methodology

To assess situation recognition accuracy, we use 10-fold stratified cross validation (WF05), a standard technique for estimating how accurately a predictive model will perform in practice.

In cross-validation, we partition the available dataset into two complimentary subsets: the first is used to train the model, while the second is used to test the trained model's accuracy. In *n*-fold validation, the data is randomly partitioned into *n* subsets, with *n*-1 subsets used for training, and the remaining subset used for testing. The process is repeated *n* times, with each fold used once as the test data. The evaluation statistics – precision, recall, and f-measure – are calculated from this process's results.

To avoid the possibility of the training or test datasets not being representative of the overall data set, due to the low occurrence frequency of some situations, we use stratification to ensure that each fold has approximately the correct proportion of each of the class values, which helps reduce the variance in the estimate (WF05).

The process of preparing the datasets described in Chapter 6 for application of the

method is identical. Each dataset is segmented into intervals 60 seconds in length and is associated with the sensor values reported during that period.

During each training phase, the genetic algorithm runs until no change is observed in the best fitness displayed by the population over a period of 20 generations. This value limits the number of generations evaluated, while avoiding premature termination of the algorithm.

Weka implementations of Naïve Bayes, Support Vector Machine (SMO), and decision tree (J48) algorithms are applied to the raw datasets using the same methodology to provide a baseline comparison. These techniques appear as standard benchmarks for comparison, and are often the best performing techniques, throughout the literature (BI04; MCLC04; KNM+06; MRSS06b; LHP+07; Ye09; McK11; MKSS13)

We also compare our approaches to published results. McKeever (McK11) uses 10-fold cross validation to evaluate an extended Dempster-Shafer based technique on the CASL dataset, Ye (Ye09) used 10-fold cross validation on the House A dataset, while van Kasteren (vKNEK08) and McKeever (MYC+10) evaluate techniques on the House A dataset (vKNEK08) using a 'miss one day out' cross-validation approach, in which 28 folds are constructed, each corresponding to a day in the dataset. After our initial experiments, we repeat our experiments using this methodology to support a comparison.

7.2.3 Evaluation Framework

Due to the availability of actively maintained, 3rd party software libraries supporting the application of machine learning techniques and genetic algorithms we choose to implement our evaluation framework using Java-based technologies. The implementation primarily uses the following libraries:

- JGap (Zha07), which provides a basic framework for implementing genetic algorithms that supports the construction and evolution of solutions modelled as chromosomes.
- The Weka workbench (FHH+05), which provides a suite of standard machine learning techniques, a framework for their evaluation, and is extensible to support the development of new classifiers.

Taking the modelling framework developed in Chapter 3, we mapped the relevant concepts, context statements, and model interpretations to the JGap chromosome model,

and the Weka interface. Our implementations of ensemble classification techniques described in Chapter 5 are also integrated into this framework.

7.3 Experiments on the CASL Smart Office Dataset

Using the approach outlined above, we perform a series of experiments on the CASL dataset to assess: *i)* the situation recognition accuracy of the standalone and ensemble situation recognition models we propose, *ii)* the intelligibility of the models and the capability of the learning process to generate smaller models, and *iii)* the model performance in comparison to other approaches. In addition, we examine alternative approaches to constructing a genetic ensemble classifier and compare them with the Dempster-Shafer-theory-based approach we propose.

- *Experiment 1:* How accurately does a generated specification set recognise situations? We use this result as a basis for comparing with ensemble classifiers and alternative machine learning techniques in later experiments;
- *Experiment 2:* What effect does the presence or exclusion of the three modelling relations – containment, adjacency, and complement – have on situation recognition accuracy and the size of the specifications generated? We test all operator combinations to assess their impact;
- *Experiment 3:* Assuming that shorter specifications are more easily understood, what effect does pruning clauses from generated specifications have on situation recognition accuracy? We iteratively increase the number of clauses pruned from generated specification and analyse the effect;
- *Experiment 4:* How does an ensemble classification approach affect the situation recognition accuracy of the specification set? We evaluate the semantic hierarchy and genetic ensemble approaches and compare them to the standalone approach;
- *Experiment 5:* How is the result of the specification set ensemble classification affected by using different aggregation algorithms? We compare several alternative aggregation approaches with the Dempster-Shafer approach used in the previous experiment;
- *Experiment 6:* How does incorporating domain knowledge into the concept model affect situation recognition accuracy? We replace the location model with one

accounting for human knowledge of each location's function in the situation recognition task to assess its impact;

- *Experiment 7:* How does applying the knowledge and genetic learning model to the construction of a decision tree affect situation recognition accuracy? We investigate the impact of using different fitness functions to guide tree construction.
- *Experiment 8:* How does an ensemble classification approach affect situation recognition accuracy of the decision tree? We evaluate the semantic hierarchy and genetic ensemble approaches and compare them to the standalone approach;
- *Experiment 9:* How does the performance accuracy of the generated specifications compare with other approaches? We compare the results of the standalone and ensemble approaches with Weka's Naïve Bayes, and J48 implementations and discuss results published in the literature.

7.3.1 Exp. 1: Recognition Accuracy of the Standalone Specification Set Model

This first experiment evaluates the performance of a standalone specification set generated using the conceptual model of the CASL data concepts presented in Section 6.2.3.

Table 7.1 presents the results of the standalone specification set as a *confusion matrix*. A confusion matrix provides a means of visualising the recognition performance and causes of error (confusion) of a classification algorithm. The leftmost column of the matrix lists the set of 6 situations within the dataset. Each row indicates how each situation's instance data is classified, corresponding to the repeated situation set in the row header. For each situation, the corresponding row gives the proportion of all instances belonging to that class that are classified as each possible situation, with the values in each row totalling 1. For example, the final row of the matrix shows that 94% of *meeting* instances were classified correctly, with the remaining 6% incorrectly classified as *reading* and *coffee*.

The proportion of correct inferences are shown along the table diagonal (each situation's recall), and are colour coded as follows:

- $0 \leq x < 0.2$ (very poor recognition) ;
- $0.2 \leq x < 0.4$;
- $0.4 \leq x < 0.6$;

$0.6 \leq x < 0.8$;

$0.8 \leq x < 1$ (very good recognition) .

Situation	*Break*	*Working*	*Reading*	*Coffee*	*Lunch*	*Meeting*
Break	0.70	0.10	0.12	0.02	0.04	0.02
Working	0.03	0.95	0.01	0.00	0.01	0.00
Reading	0.02	0.77	0.21	0.00	0.00	0.00
Coffee	0.10	0.08	0.10	0.70	0.00	0.03
Lunch	0.10	0.05	0.01	0.00	0.84	0.00
Meeting	0.01	0.00	0.05	0.01	0.00	0.94

Table 7.1: A confusion matrix showing the performance of the standalone specification model on the CASL dataset.

By far the poorest recognition performance is witnessed on the *reading* activity, with only 21% of instances correctly classified. From the confusion matrix, we see that 77% of all *reading* instances are incorrectly classified as *working* at the computer.

Specifications generated for these two situations are shown below. As we described in Chapter 3, the subjects and predicates of the context statements in these examples are fixed, therefore we show only the object value for brevity.

Situation	Specification
Working	$[= computerActive]^{1}$
Reading	$[= cubicleB]^{0.33} + [= cubicleA]^{0.32} + [= meetingRoom]^{0.25} + [= hallway]^{0.1}$

Examining these specifications gives insight into the source of this confusion:

1. The specification for *working* is based solely on the computer being active, as might be expected.

2. As the *reading* specification makes no mention of computer activity, any *reading* time-slice that is misclassified as *working* must involve computer activity. Manual inspection of the dataset confirms this analysis.

The ability to scrutinise these specifications allows two conclusions to be drawn: Firstly, according to the definition of these situations, the diary data is inaccurate. Secondly, that as the computer is often active during the *reading* activity we can state that the environment would benefit from the addition of sensors to more easily distinguish between the two situations. For example, one reason for the discrepancy might be frequent checking of email; were computer activity reporting available at a finer lever

of granularity, such as the level of the application in focus, use of such an application could be discounted from the recognition process. Secondly, the makeup of the reading specification immediately highlights the inaccuracy of the positioning system, by virtue of it referencing four locations, when the true location is known to be *cubicleB*.

The presence of inaccurate location data is also evident from inspecting the *coffee* and *lunch* specifications shown below.

Situation	Specification
Coffee	$[= hallway]^{0.43} + [= cafe]^{0.3} + [= offices]^{0.27}$
Lunch	$[= earlyAfternoon]^{0.47} + [= cafe]^{0.35} + [= 3rdFloor]^{0.18}$

Although both mention *cafe* and are differentiated by time (the inclusion of *earlyAfternoon* in the lunch specification) as expected, both also reference additional locations, highlighting this inaccuracy.

Table 7.2 summarises the recognition performance of the single specification model in terms of average precision, recall, and f-measure across the situation set, weighted f-measure and total proportion of instances correctly identified.

Ave. Precision	*Ave. Recall*	*Ave. F-measure*	*Wtd. F-measure*	*Proportion*
0.77	0.72	0.74	0.80	0.82

Table 7.2: A performance summary of the standalone specification model on the CASL dataset.

7.3.2 Exp. 2: Effect of Semantic Relations on Specification Set Recognition Accuracy

This experiment seeks to investigate the impact on *i)* accuracy, and *ii)* the size of specification generated of employing each of the three core modelling relations – subsumption ($\triangleleft$), adjacency ($\|$), and complement ($\neg$) – in the specification generation process. All six possible combinations were tested.

Table 7.3 compares the situation recognition performance of the eight models. The data show that while the overall range in performance only differs by 4%, improvements are gained through introducing the adjacency, complement and subsumption relations. Table 7.3 also charts each of these models against the average specification size of each (the average number of clauses in each test fold's specification set). We observe that the inclusion of the adjacency and complement relations corresponds to an increase in the average specification size.

Relations	*Ave. F-measure*	*Ave. No. of Specification Clauses*
None	0.732	31
$\|\|$	0.739	33
$\neg$	0.716	35
$\neg \, \|\|$	0.745	35
$\lhd$	0.730	22
$\lhd \, \|\|$	0.725	22
$\lhd \, \neg$	0.731	24
$\lhd \, \neg \, \|\|$	0.736	27

Table 7.3: A comparison of the average f-measure performance and average specification set size of models with different specification operators enabled.

The containment relation serves to reduce the average specification size overall. We can examine this effect by comparing specifications generated for the same data fold with and without use of the containment relation. First, we consider a specification generated without the relation:

Situation	Specification
Break	$[= hallway]^{0.3} + [= cubicleB]^{0.24} + [= meetingScheduled]^{0.16}$ $+[= cubicleA]^{0.11} + [= cubicleE]^{0.11} + [= meetingRoom]^{0.08}$
Working	$[= computerActive]^{1}$
Reading	$[= cubicleB]^{0.33} + [= cubicleA]^{0.32} + [= meetingRoom]^{0.25} + [= hallway]^{0.1}$
Coffee	$[= H9]^{0.28} + [= cubicleC]^{0.18} + [= H10]^{0.13} + [= hallway]^{0.13}$ $+[= cubicleB]^{0.12} + [= cafe]^{0.09} + [= offices]^{0.07} + [= cubicleE]^{0.02}$ $+[= H16]^{0.03}$
Lunch	$[= H13]^{0.13} + [= meetingRoom]^{0.12} + [= offices]^{0.12} + [= cubicleE]^{0.11}$ $+[= cafe]^{0.09} + [= cubicleB]^{0.09} + [= meetingScheduled]^{0.06}$ $+[= cubicleA]^{0.05} + [= cubicleD]^{0.04} + [= H16]^{0.04} + [= H12]^{0.02}$
Meeting	$[= meetingScheduled]^{0.42} + [= meetingRoom]^{0.4} + [= cubicleE]^{0.18}$

We observe from this specification set that while some are intuitive, for example *working* and *meeting*, the logic behind others is difficult to discern, and the effect of poor positioning data is again evident throughout. We contrast this with the specification set generated with the containment relationships enabled.

Situation	Specification
Break	$[\triangleleft lateAfternoon]^{0.54} + [\triangleleft thirdFloor]^{0.31} + [\triangleleft lateMorning]^{0.12}$ $+[= meetingScheduled]^{0.03}$
Working	$[= computerActive]^{1}$
Reading	$[\triangleleft cubicles]^{1}$
Coffee	$[= cafe]^{0.73} + [= offices]^{0.07}$
Lunch	$[\triangleleft earlyAfternoon]^{0.44} + [= cafe]^{0.35} + [\triangleleft thirdFloor]^{0.18} + [\triangleleft lateMorning]^{0.03}$
Meeting	$[= meetingRoom]^{0.29} + [= meetingScheduled]^{0.18} + [= hallway]^{0.11}$ $+[= cubicleA]^{0.09} + [= cubicleE]^{0.09} + [= cafe]^{0.07} + [= offices]^{0.08}$ $+[= cubicleD]^{0.04}$

Here, use of the concept hierarchy to abstract terms leads to the generation of a shorter, and, we argue, more easily comprehended specification set. We offer the following analysis:

- The core constituents of the *break* situation relate to the subject being positioned on the third floor, with the time periods of late morning and late afternoon providing a differential effect with other specifications.
- *working* is defined solely against the subject's computer activity as previously.
- *reading* is defined on the subject being located in the cubicle area. This accounts for some imprecision in the positioning system.
- The *coffee* specification concerns the subject's position not being on the third floor (the cafe is located on the fourth). The inclusion of the offices concept is again due to poor positioning data (the offices are located directly underneath the cafe).
- The core of the *lunch* specification describes subject being located in the cafe in the early afternoon.
- The dominant clauses in the *meeting* specification relate to the subject being located in the meeting room and having a meeting scheduled, although positioning artefacts distort the specification generated.

We note that the specifications for each situation should not be interpreted individually, but as a set. For example, while the *break* specification references the subject being positioned on the third floor, the *working* specification will score higher if the subject's computer is active, and the *reading* specification will score higher if the subject's location on the third floor is in the cubicle area. We posit that this form of differential specification can be straightforwardly interpreted from reading a specification set, but would be more difficult if not impossible to encode manually.

In summary, on the CASL dataset, inclusion of each of the semantic relations provides marginal performance improvements by means of compensating for inaccuracies in the location sensor data. While the trend of the adjacency and complement relations is to increase the average number of specification clauses, the containment relation serves to reduce this number, resulting in shorter specifications that are more easily understood.

7.3.3 Exp. 3: Effect of Specification Pruning on Recognition Accuracy

The next experiment examines the extent to which the genetic learning algorithm is effective at generating specifications containing only necessary concepts (i.e., no redundancy), and the effect on the accuracy of the specification set of pruning of atoms.

To achieve this, we use the "all relation" specifications generated by the 10-fold stratified cross evaluation process in the previous experiment and recursively prune the least significant concept from each specification set, rerunning the evaluation each time. For example, in the specification set shown above, the least significant concept is $[= meetingScheduled]^{0.03}$, taken from the *break* specification, the next least significant is $[= cubicleD]^{0.04}$ taken from the *meeting* situation, and so on.

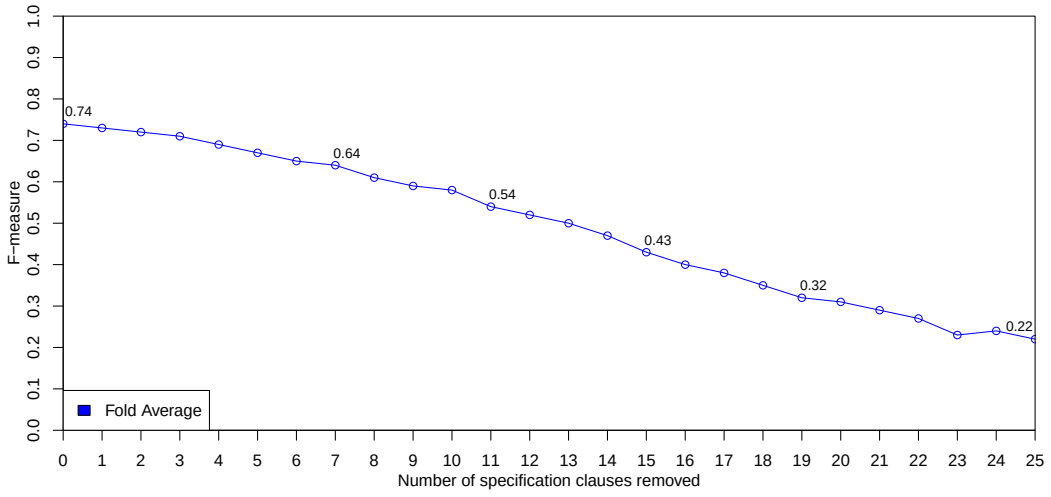


Figure 7.1: The effect of pruning specifications on recognition accuracy.

Figure 7.1 shows the effect of this pruning process on the recognition, with an approximately linear reduction in accuracy as terms are removed from the specification set. The f-measure score reduces an average of 0.02 for each term eliminated. In addition, the recognition accuracy declines immediately when the first – and least significant–

term is eliminated. Therefore we can conclude that the genetic algorithm is good at generating specifications without redundant terms.

7.3.4 Exp. 4: Recognition Accuracy of Specification Set Ensemble Techniques

In this experiment we evaluate recognition accuracy of the two ensemble approaches presented in Chapter 5 – the hierarchical and genetic ensemble – and compare their performance with the single specification model.

Hierarchical Ensemble The hierarchical ensemble approach partitions the classification space into multiple layers. A top-level classifier coarsely partitions the situation space by grouping similar situations together, while nested sub-level classifiers differentiate between the situations belonging to each grouping. For the CASL dataset, the classifier hierarchy is arranged as follows:

- A *cubicle* classifier, which differentiates between the two cubicle based situations, *working* and *reading*.
- A *cafe* classifier, which differentiates between the situations of *coffee* and *lunch*.
- A top level classifier, which differentiates between *break*, *meeting*, *cafe*, and *cubicle*.

This arrangement of classifiers is illustrated in Figure 5.2. The figure shows one possible evaluation, where in the top level classifier the *cubicle* result scores highest among competing specifications. This leads to an invocation of the associated *cubicle* classifier, where the *working* result is obtained and returned as the classification result.

Genetic Ensemble The genetic ensemble for the CASL dataset is constructed by the following process:

1. The genetic algorithm's fitness measure is adjusted to weight the f-measure of a particular situation ten times higher than the average f-measure.
2. Using the modified fitness measure, six specification sets are generated, each favouring a different situation in the CASL dataset.

3. For each specification set, the training results are used to assign each possible outcome with a set of probabilities that reflect how likely that outcome reflects the correct classification (the prior probability).

4. During the test phase, the prior probabilities corresponding to the result of each specification set's classification are aggregated using Dempster's rule of combination, with the most probable situation chosen as the classification result.

Figure 5.1 illustrates the application of Dempster's rule of combination to the output of the set of classifiers that constitute the CASL ensemble. The results shown in the red classifiers correspond to the priors associated with its classification. Dempster's rule of combination is applied to give the final result, shown in yellow.

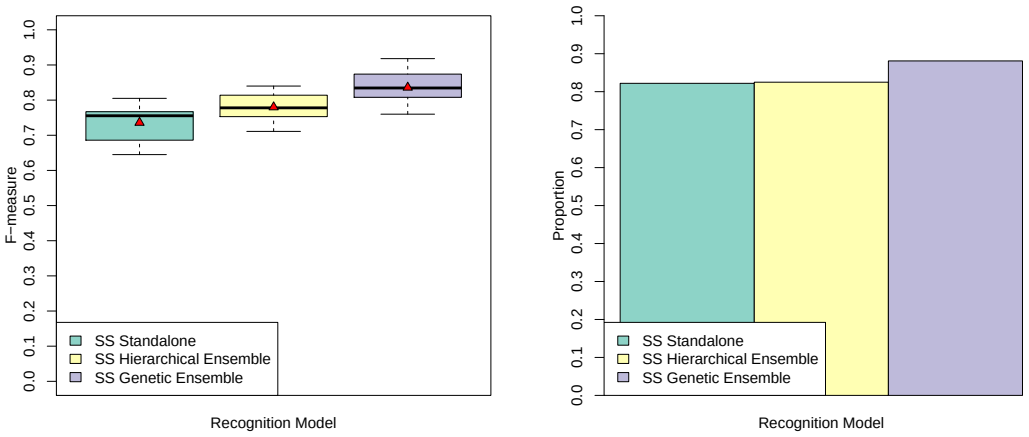


Figure 7.2: A comparison of the overall performance of the standalone and ensemble specification set approaches on the CASL dataset.

Results Figure 7.2 compares the overall recognition accuracy of both ensemble approaches with the single specification model in terms of average f-measure and the total proportion of instances recognised across all 10 folds. The boxplot is interpreted as follows: The black line represents the median value, the box 'hinges' represent the the first and third quartile, the box whiskers extend to the most extreme data point which is no more than 1.5 times the interquartile range from the box, and further outliers are denoted by a unfilled circle. Mean values, which serve as our basis for comparison in this and the following discussions around the experiments, are denoted by a red triangle.

The data show that the hierarchical ensemble exhibits an increase in average f-measure of 4% over the single specification model, with no substantial gain in total proportion

recognised. The genetic ensemble exhibits an increase in average f-measure of 10% over the single specification model, with a 6% gain in total proportion recognised.

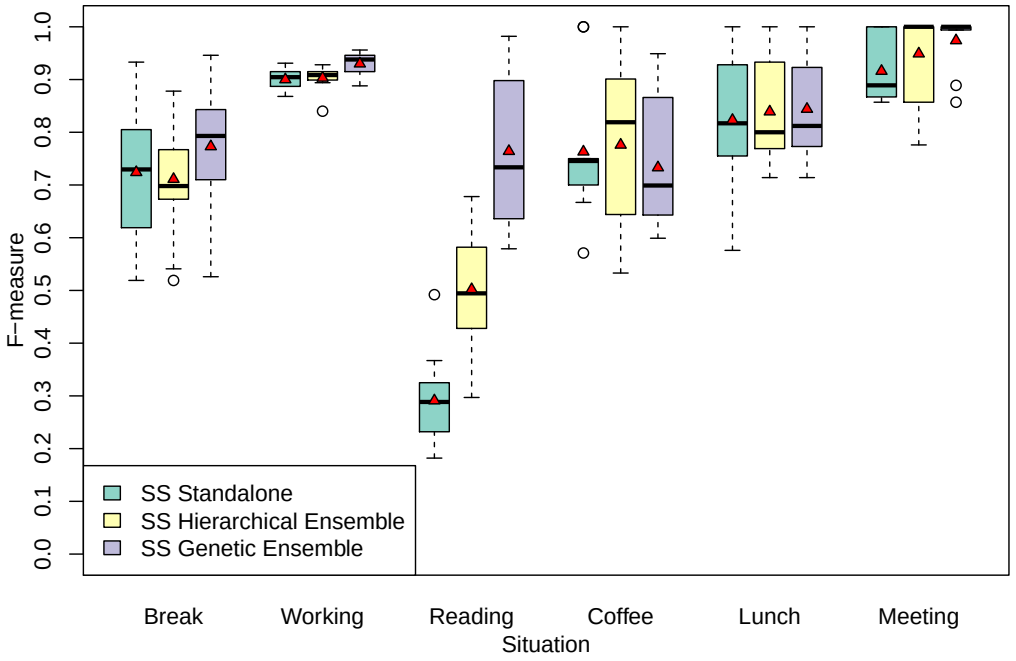


Figure 7.3: A comparison of the average f-measure performance of the standalone and ensemble specification set approaches on the CASL dataset.

Figure 7.3 shows that both ensembles consistently exhibit higher recognition accuracy than the single specification model, with *break* the only situation for which the single specification outperforms the hierarchal ensemble, and *coffee* the only instance where the single specification outperforms the genetic ensemble. With the exception of *coffee* the genetic ensemble outperforms the hierarchical ensemble across the set of situations. A key benefit of the ensembles is their improved ability to recognise the *reading* situation (increases in f-measure from 29% to 50% and 76% for the hierarchical and genetic ensembles respectively); the confusion matrix for the genetic ensemble, shown in Table 7.4, highlights this improvement, with 88% of *reading* situations successfully recognised, compared with the standalone model's recognition of 21% (shown in Table 7.1).

In summary, both ensemble approaches improve upon the recognition capability of the single specification. The genetic ensemble achieves the best overall performance, but is not consistently the most accurate when considering situations individually.

Situation	*Break*	*Working*	*Reading*	*Coffee*	*Lunch*	*Meeting*
Break	0.78	0.09	0.04	0.05	0.04	0.00
Working	0.03	0.90	0.06	0.00	0.01	0.00
Reading	0.02	0.09	0.88	0.01	0.00	0.00
Coffee	0.11	0.04	0.04	0.82	0.00	0.00
Lunch	0.10	0.03	0.02	0.00	0.86	0.00
Meeting	0.00	0.00	0.03	0.03	0.00	0.95

Table 7.4: A confusion matrix showing the performance of the genetic specification set ensemble on the CASL dataset.

7.3.5 Exp. 5: Recognition Accuracy of Alternative Aggregation Approaches

This experiment examines the implication of choosing Dempster's rule of combination to perform the aggregation of classifier results in the genetic ensemble. We compare the results obtained with three alternatives – *average*, *voting*, and *maximum* – defined as follows:

- *average*: Each situation is assigned a confidence value equal to the sum of its associated confidences across the ensemble, divided by the ensemble size. The situation with the highest averaged confidence is chosen as the classification.
- *maximum*: The situation associated with the highest confidence across the classifier ensemble is chosen as the classification.
- *voting*: Each situation is associated with a counter equal to the number of classifiers in the ensemble in which that situation has the highest assigned confidence. The situation with the highest counter (number of votes) is chosen as the classification.

Figure 7.4 charts the average f-measure performance and total proportion accuracy of all four aggregation techniques. The ensemble constructed using Dempster's combination rule displays the best performance of all the techniques, outperforming average by 3% in f-measure and total proportion correctly recognised. The voting and maximum algorithms fare worst in both comparisons.

In this instance, Dempster-Shafer-based aggregation performs well due to high levels of agreement in the evidence being combined. As discussed earlier, were this not the case, use of the average algorithm would likely yield a more accurate ensemble due to the effect of Zadeh's paradox (Zad86).

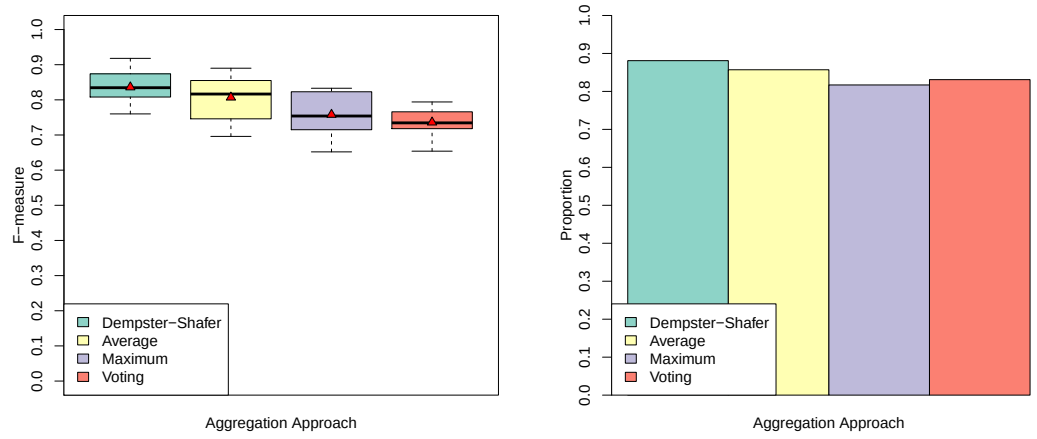


Figure 7.4: A comparison of the average f-measure performance and proportion accuracy of different aggregation techniques.

7.3.6 Exp. 6: Effect on Recognition Accuracy of Additional Expert Knowledge

This experiment investigates whether expert knowledge of sensor performance in the target environment can be used to improve situation recognition accuracy. With respect to the CASL dataset we define this expert knowledge as:

- knowing that the positioning system performs poorly in many cases;
- knowing that *cubicleA* is consistently confused with *cubicle B*;
- knowing that the remaining cubicles and offices play no role in any of the situations.

Figure 6.7(a) shows an alternative location model that replaces the floor-based building structure with a structure that accounts for the above knowledge of the target environment into account. Here, the two commonly confused cubicles are grouped together under the concept of *work area*, and the locations known to play no role are contained within an *other areas* concept. The other locations appear unchanged.

Using this location model we construct standalone and two ensemble specification models for the CASL dataset as before. Figure 7.5 charts the comparative performance of these three classifiers with their earlier versions using the physical-based location model.

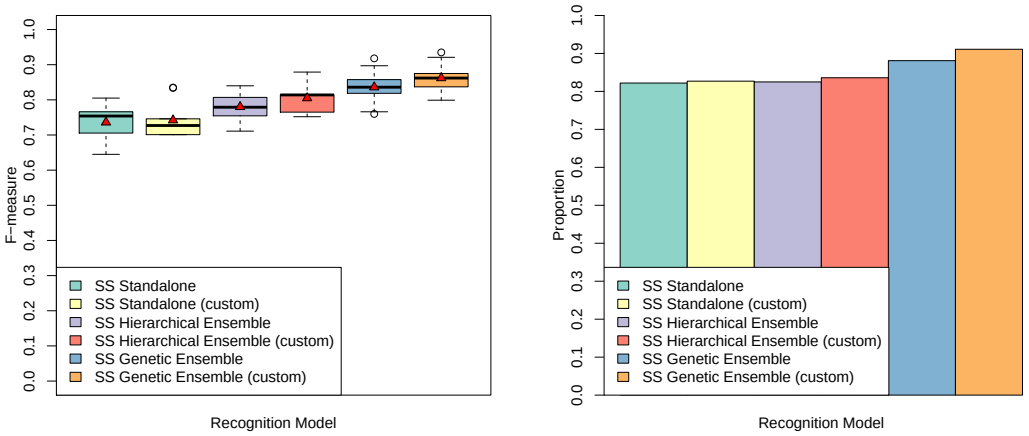


Figure 7.5: A comparison of the average f-measure performance and proportion accuracy of specification sets using the physical and functional location models.

The data show that in each case, the specification constructed using the expert location knowledge outperforms its physical-based counterpart.

This result indicates that, where such expert knowledge can be applied, it is possible to mitigate the effect of noisy sensor data and increase the recognition accuracy of the generated specifications. In general, this approach can be applied to other information domains; for example, to explicitly account for times or combinations of sensor features that are situation relevant.

7.3.7 Exp. 7: Applying the Domain and Genetic Learning Models to Decision Tree Construction

In this experiment, the decision tree model replaces the specification set as the decision making process. The knowledge model interpretation of sensor data provides the inputs to the decision tree, while the genetic algorithm *i)* guides the selection of the subset of knowledge primitives used as input, that is, it automates the concept selection process, and *ii)* employs its fitness function as a search heuristic to select from all possible decision trees those constrained a) in total size and b) in number of inputs, as a means of trading off recognition accuracy with understandability of the decision tree.

Four different variants are investigated:

- *Unrestricted*: A regular decision tree is constructed, evaluating all concepts in the knowledge model and treating them as a linear array of features. A baseline.

- *MaxScore*: The genetic algorithm is used to search for the decision tree yielding the highest recognition accuracy on the training data.
- *RestrictTreeSize*: as *MaxScore*, with the total number of nodes in the decision tree penalised linearly.
- *RestrictConcepts*: as *MaxScore*, with the total number of knowledge model concepts used as input features penalised linearly.

Table 7.5 compares the average features across folds of the decision trees generated by each of the variant models in term of: size (number of decision nodes), leaves (total number of decision paths), concepts used (number of features), and performance (f-measure and total proportion). The data show that the model produced without use of the genetic fitness function has a slightly poorer recognition accuracy, around 2% less than the other approaches, which perform similarly.

While the performance of the variants generated using the fitness functions is similar, the data shows that the performance can be achieved by a smaller tree or a tree referring to fewer concepts than the other models, with corresponding implications for ease of decision process scrutiny. Notably, the *RestrictConcepts* model achieves near identical accuracy than the *Unrestricted* model using only 40% (7.5/16.6) of the number of concepts, although the generated tree (size and leaves) is 110% (75.2/68.3) larger. The *RestrictTreeSize* model achieves its performance improvement over the baseline model with a smaller model (77% (52.8/68.3) of the size and leaves), also achieving a reduction in the amount of concepts by 11% (16.6/18.6). No single tree characteristic of either the *Unrestricted* or *MaxScore* improves upon these variants.

Technique	*Size*	*Leaves*	*Concepts*	*F-Measure*	*Proportion*
Unrestricted	68.3	34.8	18.6	0.874	0.928
MaxScore	74.4	27.5	14.7	0.893	0.944
RestrictConcepts	75.2	38.1	**7.5**	0.877	0.926
RestrictTreeSize	**52.8**	**26.9**	16.6	**0.894**	**0.945**

Table 7.5: A summary of the attributes of the learned decision tree variants.

In conclusion, applying both the knowledge model and genetic algorithm to the task of constructing a decision tree produces significant gains in recognition accuracy over the specification set variants. This alone is not surprising as the underlying decision model is more complex. Use of the fitness functions to guide the search process is effective in restricting the size of the decision process, both in terms of total size and number of input features. The *RestrictTreeSize* variant performs best, but all variants improve upon accuracy of the baseline approach.

The decision tree model was selected because of its transparent decision making process. However, despite the improved accuracy, the generated decision processes are less intelligible than the equivalent specification sets. Compared to an average of 26 specification set terms (from Exp. 2), the decision tree variants investigated here have an average size of 53 to 75 nodes (from Table 7.5). While it is arguable that such a decision process may still be understood, we assume the ease of inspection is less than the specifications sets, which are 35% – 49% (i.e., 26/75 to 26/53) of this size.

7.3.8 Exp. 8: Recognition Accuracy of Decision Tree Ensemble Techniques

In this experiment we evaluate recognition accuracy of the two ensemble approaches presented in Chapter 5 – the hierarchical and genetic ensemble – and compare their performance with the single decision tree.

The construction of both the hierarchical and genetic ensembles follows the structure for the specification set ensembles in Experiment 4.

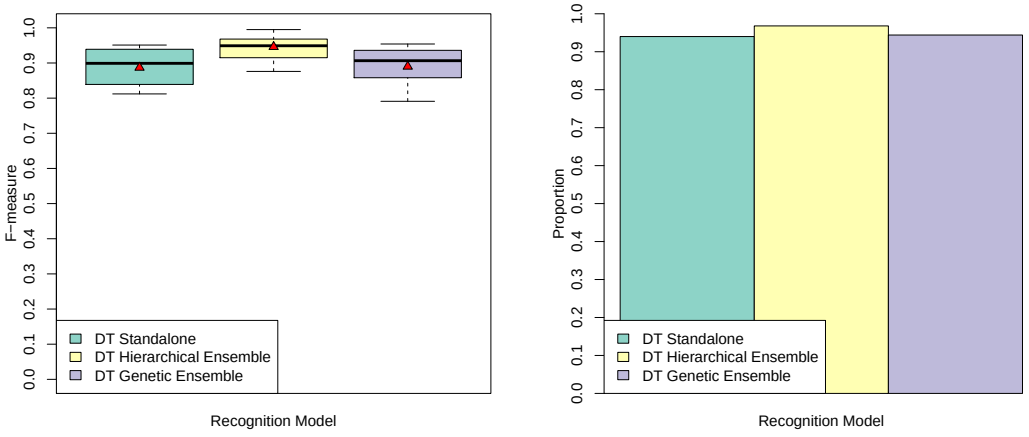


Figure 7.6: A comparison of the overall performance of the standalone and ensemble decision tree approaches on the CASL dataset.

Figure 7.6 compares the overall recognition accuracy of both ensemble approaches with the single specification model. The data show that the genetic ensemble exhibits an increase in average f-measure of 0.3% over the single decision tree model, with no substantial gain in total proportion recognised. The hierarchical ensemble exhibits an increase in average f-measure of 5.9% over the single decision tree model, with a 2.8% gain in total proportion recognised.

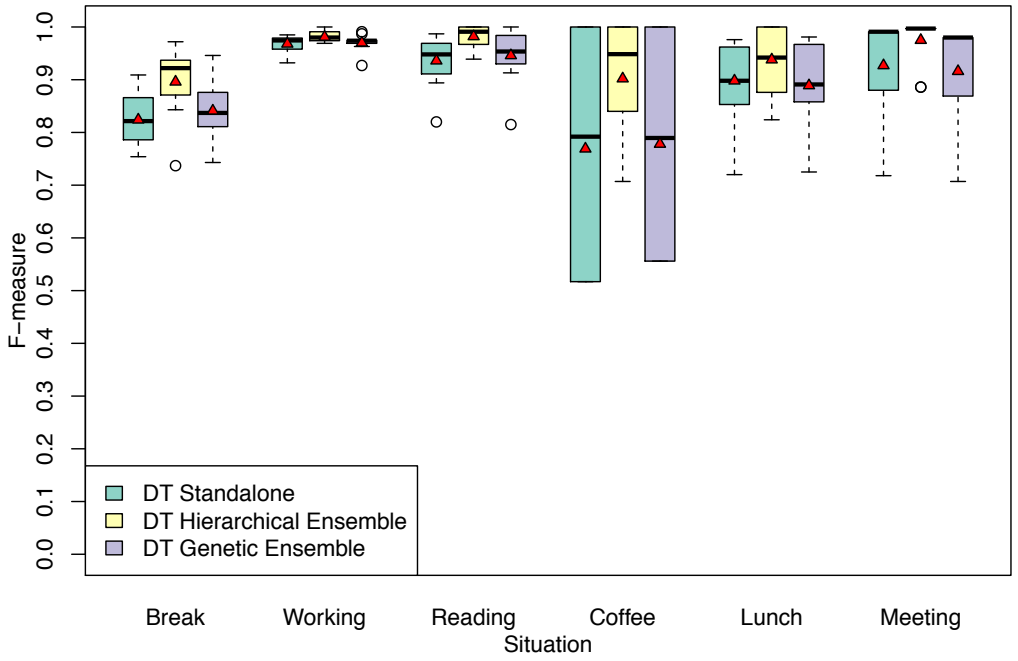


Figure 7.7: A comparison of the average f-measure performance of the standalone and ensemble decision tree approaches on the CASL dataset.

Figure 7.7 shows that, as with the specification set models, both tree ensembles consistently exhibit higher recognition accuracy than the single decision tree model. In this instance, the hierarchical ensemble outperforms the other approaches across all situations. The largest improvement over the other approaches is the f-measure associated with the *coffee* situation. We can see from the confusion matrix for the hierarchical ensemble, shown in Table 7.6, that this is due to increased precision—no other situations are misclassified as *coffee*. The recall figure of 0.82 is identical to the recognition capability of the genetic specification set model shown in Table 7.4

Situation	*Break*	*Working*	*Reading*	*Coffee*	*Lunch*	*Meeting*
Break	0.87	0.09	0.00	0.00	0.03	0.01
Working	0.01	0.99	0.00	0.00	0.00	0.00
Reading	0.01	0.02	0.98	0.00	0.00	0.00
Coffee	0.07	0.07	0.00	0.82	0.00	0.04
Lunch	0.00	0.05	0.00	0.00	0.95	0.00
Meeting	0.00	0.00	0.00	0.00	0.00	1.00

Table 7.6: A confusion matrix showing the performance of the hierarchical decision tree ensemble on the CASL dataset.

In summary, the hierarchical ensemble significantly improves upon the recognition

capability of the single decision tree and the genetic ensemble, consistently achieving better recognition across the complete situation set.

7.3.9 Exp. 9: Comparison with Alternative Situation Recognition Approaches

Figure 7.8 compares the classifiers generated using the CASL dataset against the Weka Naïve Bayes and J48 decision tree implementations (the Support Vector Machine performed similarly to Naïve Bayes and is not shown).

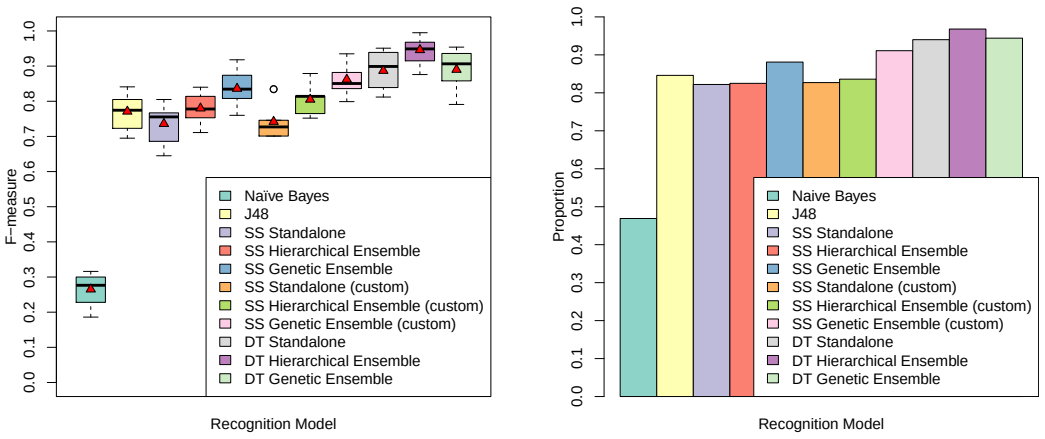


Figure 7.8: A summary of the performance of selected standard machine learning techniques along with our approaches using both the physical and custom location models.

The Naïve Bayes classifier performs worse than the other approaches, which may be accounted for by the large amount of noise in the data. The standard J48 decision tree performs significantly better, with an f-measure of 0.77 — outperforming both variants of the standalone specification set classifier by 3%, however, the average size of tree is 48 nodes, nearly double the size of the specification sets (an example is shown in Appendix B). The two specification set hierarchical ensemble variants improve upon the standard J48 tree by 1% and 3%, with the genetic ensemble variants performing better still, scoring 0.84 and 0.86: improvements of 7% and 9% respectively. In all cases, the shared knowledge model with locations grouped by functionality serve to improve recognition accuracy.

The semantic decision trees constructed using the knowledge model and genetic algorithm provide further performance improvements. The best, the *RestrictTreeSize*

variant, achieves an f-measure of 0.89, 12% higher than the regular decision tree. The semantic decision tree ensembles serve to improve recognition accuracy further, with the hierarchical ensemble achieving an f-measure of 0.95. This represents a 6% increase in accuracy over the standalone semantic decision tree, a 10% improvement over the genetic specification set ensemble, and an 18% improvement over a single standard J48 decision tree.

In the literature, McKeever's time-extended Dempster-Shafer technique (McK11) reports an f-measure of 0.74 on the CASL dataset; this is equivalent to the performance of the standalone specification set. All the additional techniques we have developed improvements relative to this baseline, with the best being the hierarchical ensemble (0.95), representing an increase in recognition accuracy of 21%.

7.4 Experiments on the House A Dataset

In this section we perform experiments on the House A dataset to assess further: *i)* the situation accuracy of the standalone and ensemble situation recognition models we propose, *ii)* the intelligibility of the models and the capability of the learning process to generate smaller models, and *iii)* the model performance in comparison to other approaches. In addition, we examine the effect of the *TimeSince* model interpretation on recognition accuracy.

- *Experiment 1:* How accurately does a single genetic specification model recognise situations? We use this result as a basis for comparing with ensemble classifiers and alternative machine learning techniques in later experiments;
- *Experiment 2:* How does an ensemble classification approach affect situation recognition accuracy of the specification set? We evaluate the semantic hierarchy and genetic ensemble approaches and compare them to the standalone approach;
- *Experiment 3:* How does applying the *TimeSince* interpretation of data affect the situation recognition accuracy of a J48 decision tree using the genetic algorithm? We examine the accuracy attained using all possible combinations.
- *Experiment 4:* How does applying the knowledge and genetic learning model to the construction of a J48 decision tree affect situation recognition accuracy? We investigate the impact of using different fitness functions to guide tree construction;

- *Experiment 5:* How does an ensemble classification approach affect situation recognition accuracy of the decision tree? We evaluate the semantic hierarchy and genetic ensemble approaches and compare them to the standalone approach;
- *Experiment 6:* What are the run times for training and recognition? We evaluate the standalone, semantic hierarchy, and genetic ensemble approaches for both the specification set and decision tree models and compare the results;
- *Experiment 7:* How does the inclusion of unmarked time-slices affect situation recognition accuracy? We compare the impact of including time-slices that don't correspond to the core situation set to the performance of the standalone and ensemble classifiers;
- *Experiment 8:* How does the performance accuracy of the generated specifications compare with other approaches? We compare the results of the standalone and ensemble approaches with Weka's J48, and State Vector Machine implementations and discuss results published in the literature.

7.4.1 Exp. 1: Recognition Accuracy of the Standalone Specification Set Model

As with the CASL dataset, the first experiment evaluates the performance of a standalone specification set generated using the conceptual model of the House A data concepts presented in Section 6.3.3.

Table 7.1 (colour coded as before) presents the confusion matrix corresponding to the generated standalone specification set.

Situation	*Leave*	*Toilet*	*Shower*	*Bed*	*Breakfast*	*Dinner*	*Drink*
Leave	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Toilet	0.02	0.79	0.01	0.12	0.02	0.03	0.02
Shower	0.01	0.05	0.93	0.00	0.01	0.00	0.00
Bed	0.00	0.00	0.00	1.00	0.00	0.00	0.00
Breakfast	0.00	0.10	0.00	0.01	0.88	0.00	0.01
Dinner	0.01	0.01	0.00	0.01	0.00	0.97	0.00
Drink	0.00	0.00	0.00	0.06	0.09	0.37	0.49

Table 7.7: A confusion matrix showing the performance of the standalone specification model on House A dataset.

Situation recognition accuracy is high, above 88% for all situations excluding two: *toilet*, which is recognised correctly 79% of the time, and *drink* situation, instances of which are recognised correctly only 49% of the time. The misclassified *toilet* instances are

predominantly confused with *bed*, while misclassified instances of *drink* are split across the *breakfast* and *dinner* classifications, with misclassifications as *dinner* approximately three times as likely as *breakfast*.

An example of a generated specification set is shown below. Using to the *Change-Point+LastChanged* representation, the subscript C corresponds to the concept hierarchy that interprets the current values of sensors, while the subscript L corresponds to the concept hierarchy reflecting the last value to be changed.

Situation	Specification
Leave	$[= frontDoor]^{1}_{L}$
Toilet	$[= plates]^{0.35}_{C} + [\not\parallel \lhd foodStorage]^{0.25\cdot}_{C} + [\not\parallel = cups]^{0.23}_{C} [\not\parallel \lhd heating]^{0.12}_{C}$ $+[= frontDoor]^{0.06}_{L}$
Shower	$[= earlyNight]^{0.29} + [\parallel \lhd foodStorage]^{0.23}_{C} + [\neq frontDoor]^{0.16}_{C}$ $+[= bathroomDoor]^{0.11}_{L} + [\neq toiletDoor]^{0.08}_{C}$
Bed	$[\not\lhd anyKitchen]^{0.41}_{L} + [= bedroomDoor]^{0.39}_{L} + [\not\lhd evening]^{0.11} + [\not\lhd morning]^{0.09}$
Breakfast	$[\lhd morning]^{0.39} + [\not\lhd evening]^{0.33} + [\lhd anyKitchen]^{0.11}_{L} + [\lhd bathroomDoor]^{0.07}_{L}$
Dinner	$[\lhd anyKitchen]^{0.39}_{L} + [\not\lhd anyBathroom]^{0.33}_{C} + [\lhd anyKitchen]^{0.30}_{C}$
Drink	$[\lhd foodStorage]^{0.17}_{L} + [= cups]^{0.16}_{L} + [\neq frontDoor]^{0.16}_{L} + [\neq microwave]^{0.13}_{L}$ $+[\lhd earlyMorning]^{0.12} + [\neq toiletDoor]^{0.10}_{L} + [\lhd afternoon]^{0.09} + [\parallel \lhd evening]^{0.07}$

As with the specifications for the CASL dataset, we contend that the subsumption relationships of the concept hierarchy lead to the generation of a specification set that lends itself to be understood both in terms of inspecting the information that contributes strongly to accurate classifications. Additionally, as we described in Section 4.4, analysing the reasons behind misclassification and the general efficacy of the sensors installed. We offer the following analysis that we posit would not be easy to discern through a black-box decision process:

- Intuitively, the *leave* situation is defined solely on the front door sensor being the last to fire.
- As identified above, *toilet* is one situation that is recognised poorly relative to other situations. The associated specification includes no mention of either of the sensors associated with the toilet, indicating that the situation is identified primarily through creating a contrast to sensor firings that indicate other situations (primarily in the kitchen). The two, unrelated, concepts that contribute positively to the specification suggest that the situation often occurs in close proximity to entering the house or following an activity in the kitchen; these being produced as an artefact of the segmentation process.
- A similar case holds for the *shower* specification with the bathroom door sensor and time of day providing a positive indicator for the situation. Without a sensor

directly associated with the situation (such as a water flow sensor), the remainder of the specification forms a contrast to other situations. Note, in particular, that the appearance of the toilet door concept here serves to directly differentiate between these two situations. The appearance of the *foodStorage* concept indicates that this situation occurs in close proximity to kitchen activity and is an artefact of the segmentation process.

- The *bed* specification is self explanatory, with strong focus on there being no activity in the kitchen, coupled with the last sensor to fire being in the bedroom. The time of day being either in the evening or morning also contributes.
- Also clearly delineated, the *breakfast* specification describes kitchen activity taking place in the morning. The concept contributing least is the *bathroomDoor*, likely an artefact of the segmentation process.
- The *dinner* situation is also clearly formed; it too describes kitchen activity, with the *breakfast* specification providing the temporal distinction.
- Finally, the *drink* situation is the longest of the specifications, as might be expected from its low recognition rate. The concepts that contribute most are *foodStorage* and *cups*. However, these concepts do not uniquely define this situation, as they also form part of the other kitchen based situations. The remainder of the specification can therefore be interpreted as an attempt to contrast this situation with those. The lack of any unique combination of concepts or temporal pattern associated with the *drink* situation accounts for its frequent misclassification as *breakfast* and *dinner*.

Table 7.8 summarises the overall recognition performance on the House A dataset of the single specification model in terms of average precision and recall, f-measure and weighted f-measure, and total proportion of instances correctly identified.

Ave. Precision	*Ave. Recall*	*Ave. F-measure*	*Wtd. F-measure*	*Proportion*
0.855	0.865	0.858	0.995	0.995

Table 7.8: A performance summary of the standalone specification model on the House A dataset.

7.4.2 Exp. 2: Recognition Accuracy of Specification Set Ensemble Techniques

In this experiment we evaluate recognition accuracy of the two ensemble approaches presented in Chapter 5 – the hierarchical and genetic ensemble – on the House A dataset

and compare their performance with the single specification model.

Hierarchical Ensemble As before, the hierarchical ensemble approach partitions the classification space into multiple layers. A top-level classifier coarsely partitions the situation space by grouping similar situations together, while nested sub-level classifiers differentiate between the situations belonging to each grouping. For the House A dataset, the classifier hierarchy is arranged as follows:

- A *hygiene* classifier, which differentiates between the situations, *toilet* and *shower*.
- An *eating/drinking* classifier, which differentiates between the situations of *breakfast*, *dinner*, and *drink*.
- A top level classifier, which differentiates between *leave*, *hygiene*, *bed*, and *eating/drinking*.

Genetic Ensemble The genetic ensemble for the House A dataset is constructed by applying the process described in Section 7.3.4

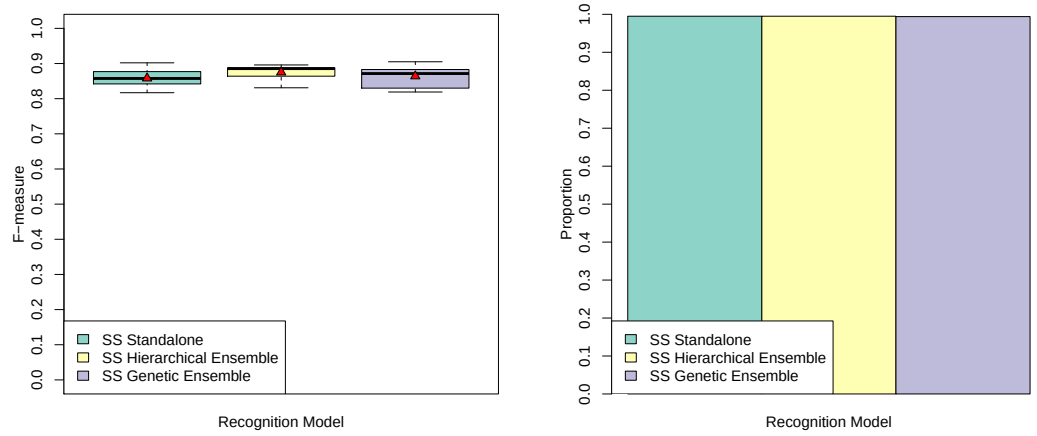


Figure 7.9: A comparison of the overall performance of the standalone and ensemble specification set approaches on the House A dataset.

Results Figure 7.9 compares the overall recognition accuracy of both ensemble approaches with the single specification model, while Figure 7.10 gives a detailed breakdown of the performance of each technique across each situation in the specification set. The data show that the hierarchical ensemble exhibits an increase in average f-measure

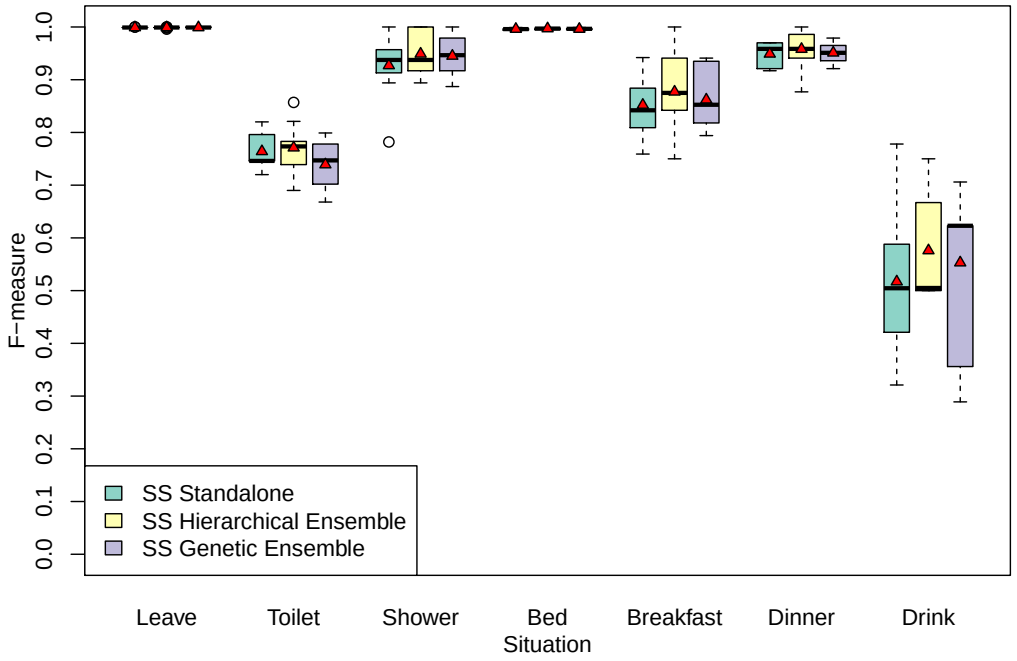


Figure 7.10: A comparison of the average f-measure performance of the standalone and ensemble specification set approaches on the House A dataset.

of nearly 1.7% over the single specification model, with no significant gain in total proportion recognised. The genetic ensemble exhibits an increase in average f-measure of 0.6% over the single specification model, again with no significant gain in the total proportion recognised.

In summary, both ensemble approaches improve upon the recognition capability of the single specification model. However, unlike with the CASL dataset, the gains are less significant. The hierarchical ensemble achieves the best overall performance, but is not consistently the most accurate when considering situations individually.

7.4.3 Exp. 3: Effect on Recognition Accuracy of Using the *TimeSince* Data Representation

This experiment investigates whether improved recognition accuracy be attained through increasing the complexity of the data representation format instead of selecting a more complex learning technique.

The original work by van Kasteren et al. (vKNEK08) uses Hidden Markov Models

and Conditional Random Fields precisely because of their strength in discovering and encoding sequential temporal relationships. Given that the decision tree does not have these properties, the motivation behind the *TimeSince* format, described in Section 4.2, is to augment the data representation with implicit temporal information, aiding accurate model construction without requiring a more complex modelling technique.

Table 7.9 shows the f-measure obtained using the *TimeSince* representation of sensor data to construct an *Unrestricted* decision tree model, compared with the three other representations proposed by van Kasteren et al— Raw (*RW*), Change Point (*CP*), Last Changed (*LA*), and Time Since (TS). The table also shows all possible technique combinations, 15 in total.

Combination	*F-measure*	*Combination*	*F-measure*
RW	57.9	CP+TS	89.6
CP	66.4	**LA+TS**	**91.2**
LA	86.8	RW+CP+LA	85.9
TS	88.9	RW+CP+TS	85.9
RW+CP	69.8	RW+LA+TS	88.7
RW+LA	86.3	**CP+LA+TS**	**91.2**
RW+TS	87.8	RW+CP+LA+TS	90.7
CP+LA	87.4		

Table 7.9: The accuracy achieved using different combinations of model interpretations for the House A dataset.

Considering each technique in isolation, the results show that the TS format achieves higher accuracy than the other three, at 88.9%. This is 2.1% higher than the *LA* representation, which is next best. Looking at pairs of techniques, we see that all three combinations involving the TS representation outperform the others, with the $TS+LA$ achieving 91.2%, an improvement of 2.3% over TS alone. We note also that the tree generated using the $CP+LA$ representation, outperforms the single specification set (that uses this representation) by 1.6%, and has near equivalence with the hierarchical ensemble (a difference of 0.1%).

Combining three or all four techniques provides no further improvements in accuracy.

7.4.4 Exp. 4: Applying the Domain and Genetic Learning Models to Decision Tree Construction

Following on from the previous experiment, we now carry out the same procedure as for the CASL dataset to evaluate the effect of using the genetic algorithm to generate decision tree variants—guiding the selection of input features and using the fitness

function to constrain the search in terms of tree size and the number of input features.

Technique	*Size*	*Leaves*	*Concepts*	*F-Measure*	*Proportion*
Unrestricted	103	52	32	**0.912**	**0.996**
MaxScore	162.4	81.7	12	0.887	0.995
RestrictConcepts	352.8	179.6	**7.4**	0.819	0.991
RestrictTreeSize	**78.2**	**39.6**	21.8	**0.912**	**0.996**

Table 7.10: A summary of the attributes of the learned decision tree variants applied to the House A dataset.

Table 7.10 compares the average features of the decision trees generated by each of the variant models in term of: size (no. of decision nodes), leaves (total no. of decision paths), concepts used (no. of features), and performance (f-measure and total proportion). The data shows that both the *Unrestricted* and the *RestrictTreeSize* models attain the same performance of 91.2%. Both outperform the *MaxScore* variant by 2.5% (due to the search terminating before an equivalent model was found), and the *RestrictConcepts* variant by 9.3%.

There are two interesting results: Firstly, although the performance of the *Unrestricted* and the *RestrictTreeSize* variants is the same, the data show that the *RestrictTreeSize* variant achieves this performance with a model 24% smaller (78.2/103), and using approximately 10 fewer concepts. Secondly, although the *RestrictConcepts* variant has a reduced average f-measure of 81.9%, it achieves this using an average of only 7.4 concepts: These correspond to 7 physical sensors (toilet, bathroom and front doors, the microwave, and pans, cups and groceries cupboard), and the temporal abstraction of *lateMorning*.

The confusion matrix for the *RestrictTreeSize* variant is shown in Table 7.11. Compared with the matrix for the standalone specification set in Table 7.7, it can be seen that the technique provides small recognition improvements across the board, with the major contributor to the improvement being an increase of 21% in the ability to recognise the *drink* situation.

In conclusion, the use of the *TimeSince* representation in concert with the knowledge model and genetic algorithm generates a decision tree which provides gains in recognition accuracy over the specification set variants: 5.4% over the standalone specification set and 3.7% over the hierarchical ensemble.

As with the CASL dataset, the *RestrictConcepts* fitness function is effective in restricting the total size of the decision tree compared to the *Unrestricted* model, without loss of accuracy. However, the average number of nodes is still larger than the number of terms in the equivalent specification set, making it less intelligible. Finally, although it

Situation	*Leave*	*Toilet*	*Shower*	*Bed*	*Breakfast*	*Dinner*	*Drink*
Leave	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Toilet	0.02	0.81	0.03	0.10	0.02	0.01	0.02
Shower	0.01	0.03	0.94	0.00	0.01	0.00	0.00
Bed	0.00	0.00	0.00	1.00	0.00	0.00	0.00
Breakfast	0.00	0.00	0.00	0.02	0.96	0.00	0.01
Dinner	0.01	0.00	0.00	0.00	0.00	0.98	0.01
Drink	0.00	0.03	0.03	0.00	0.09	0.14	0.71

Table 7.11: A confusion matrix showing the performance of the *RestrictedTreeSize* decision tree on the House A dataset.

displayed the worst performance overall, the *RestrictConcepts* fitness function shows that an accuracy of approximately 82% can be achieved using 7 sensors—just under 50% of those installed in the house.

7.4.5 Exp. 5: Recognition Accuracy of Decision Tree Ensemble Techniques

In this experiment we evaluate recognition accuracy of the two decision tree ensemble approaches presented in Chapter 5 – the hierarchical and genetic ensemble – on the House A dataset and compare their performance with the single decision tree model.

The construction of both the hierarchical and genetic ensembles follows the structure for the specification set ensembles in Experiment 2.

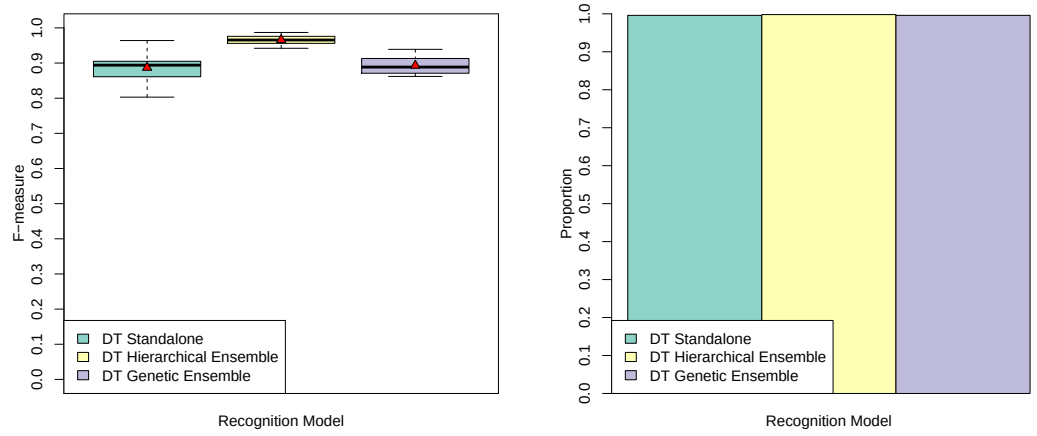


Figure 7.11: A comparison of the overall performance of the standalone and ensemble decision tree approaches on the House A dataset.

Figure 7.11 compares the overall recognition accuracy of both ensemble approaches

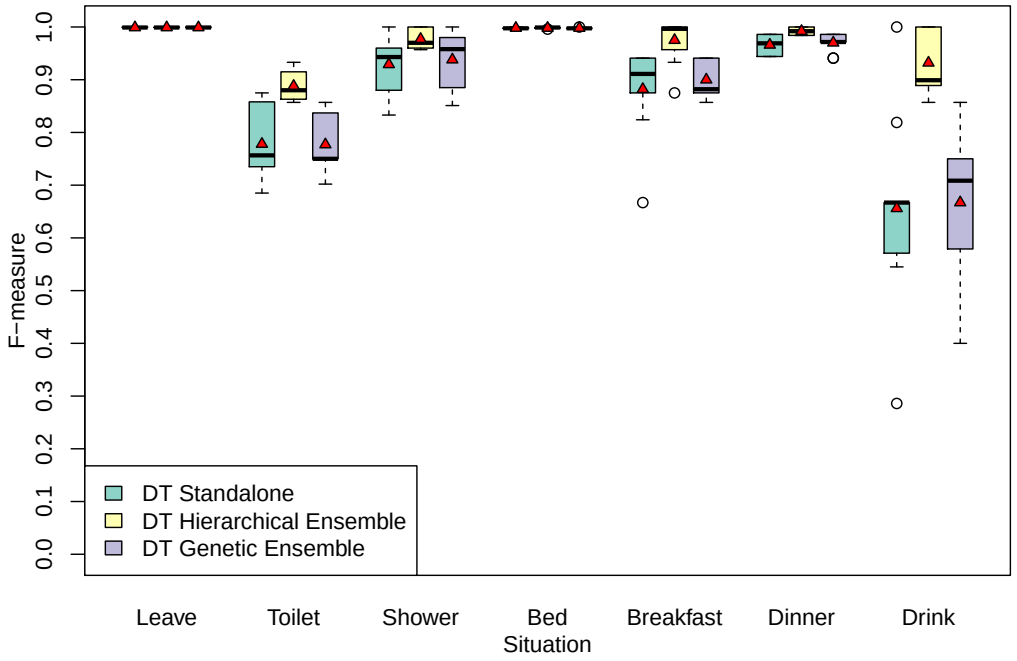


Figure 7.12: A comparison of the average f-measure performance of the standalone and ensemble decision tree approaches on the House A dataset.

with the single decision tree model, while Figure 7.12 breaks down the performance of each technique across each situation in the dataset. The data show that the hierarchical ensemble exhibits an increase in average f-measure of nearly 7.9% over the single decision tree model, and the genetic ensemble exhibits an increase in average f-measure of 0.6% over the decision tree model, neither with a significant gain in the total proportion recognised.

Appendix C lists the decision tree models for the standalone and hierarchical ensembles. It can be seen that the overall size of the hierarchical ensemble is smaller than the standalone model. In addition to this, the modularity of the approach also aids the intelligibility of the decision process.

In summary, as with the specification set models, the hierarchical decision tree ensemble achieves the best overall performance. It is also consistently the most accurate when considering situations individually, and provides improvements over the standalone model in terms of intelligibility.

7.4.6 Exp. 6: Performance Comparison of the Studied Techniques

This experiment evaluates the run times for training and recognition of the standalone, semantic hierarchy, and genetic ensemble approaches using both the specification set and decision tree models and compares the results.

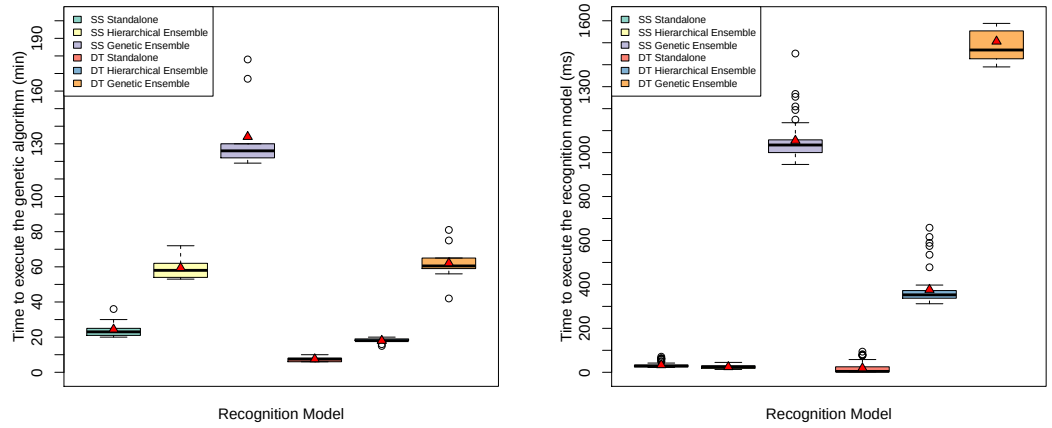


Figure 7.13: A comparison of the overall run times for training and recognition of the standalone, semantic hierarchy, and genetic ensemble approaches using both the specification set and decision tree models.

Figure 7.14 charts the training and recognition performance of the 6 approaches. Considering learning performance first, the figure illustrates the average learning time across the 10 folds of the dataset. The results confirm our earlier assertion that learning using a genetic algorithm is computationally complex. However, we maintain that because the learning need take place only once (or very infrequently), this cost is acceptable.

Across both the specification set and decision tree models the data show that learning performance is in line with the number of models that combine to create the standalone model (1), the hierarchical ensemble (3), and the genetic ensemble (7). The genetic ensemble classifiers operate over the complete set of situations, leading to a construction time of at least 7 times the cost of the standalone model, while the three classifiers of hierarchical ensemble each solve a simplified part of the problem, reducing the construction time of each (compared to constructing 3 standalone models), and hence the overall construction time.

For each pair of corresponding techniques learning of the decision tree variant is significantly quicker than the specification set variant. This is because the genetic algorithm is responsible for both feature selection and specification structure in the

latter variant, but only feature selection in the former (which serves as input to the decision tree construction process).

The runtime recognition performance figures describe the processing of a single fold of data containing approximately 4000 instances. As with learning, processing times are roughly proportional to the complexity of the underlying models, although we note that the standalone and hierarchical specification set models have very similar average execution times (a difference of approximately 9ms). The runtime performance of the specification set models outperform their decision tree counterparts, likely due to the simpler underlying model that is generated by the learning process. Both genetic ensemble models display the slowest performance (1–1.6 seconds).

We reiterate that the runtime performance figures are a measure of the processing of approximately 4000 instances, indicating that even in the worst performing model, runtime performance *per instance* has a sub-millisecond execution time on average, which implies that all models are appropriate for real-time evaluation of sensor data.

We discuss the implications of these findings on the scalability of the proposed approaches further in Section 7.6.

7.4.7 Exp. 7: Effect on Recognition Accuracy of Idle Time Slices

This experiment examines the effect on recognition accuracy of introducing 'idle' time slices in the dataset—time periods over which no activity is explicitly recorded as having taken place, but where sensors may still fire.

Here, we reassess the situation recognition accuracies of each of the standalone and ensemble approaches for both the specification set and decision tree models, and explore four strategies for each, eight in total:

1. Treating *idle* as a regular situation, generating a standard standalone model.
2. Adding a post-classification filter per classification to the standalone classifier, described in previous experiments, that differentiates between "true" classifications of each situation and the *idle* situation, as described in Section 5.3.3.
3. Adding a post-classification filter per classification to the hierarchical classifier, described in previous experiments, that differentiates between "true" classifications of each situation and the *idle* situation, as described in Section 5.3.3.
4. Treating *idle* as a regular situation, generating a standard genetic ensemble.

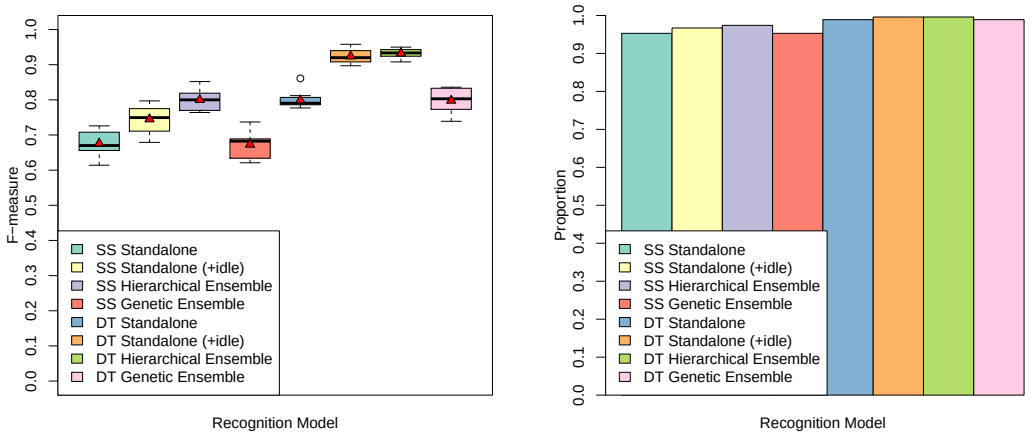


Figure 7.14: A comparison of the overall performance of the standalone and ensemble classification approaches on the House A dataset including idle time slices.

Figure 7.14 compares the overall recognition accuracy of all eight approaches. The data show that, with the exception of the hierarchical specification set, which marginally outperforms the standalone semantic decision tree (both with an f-measure of 0.8), the decision tree approaches attain better performance. The lowest f-measure attained is 0.68 by the standalone specification set, and the highest is 0.93, attained by the hierarchical decision tree ensemble. The results also show that adding a post-processing filter to differentiate between *idle* time slices and true classifications boosts performance significantly for both the specification set and decision tree approaches: an increase of 7% in the former and 12% in the latter.

Figure 7.15 charts the performance of each technique across each situation in the specification set. By comparison with earlier results, we can see that for most techniques, the introduction of idle time slices reduces significantly their ability to recognise the *toilet*, *shower*, and *drink* situations. This indicates the presence in the dataset of idle time slices where the sensors normally associated with these situations have fired, leading to confusion in the decision processes.

We conclude that the standalone specification set and decision tree perform poorly relative to their hierarchical counterpart strategies, as do both genetic ensembles. The two biggest factors resulting in improved performance are the introduction of a post-classification *idle* filter, and the structuring of the classification as a semantic hierarchy.

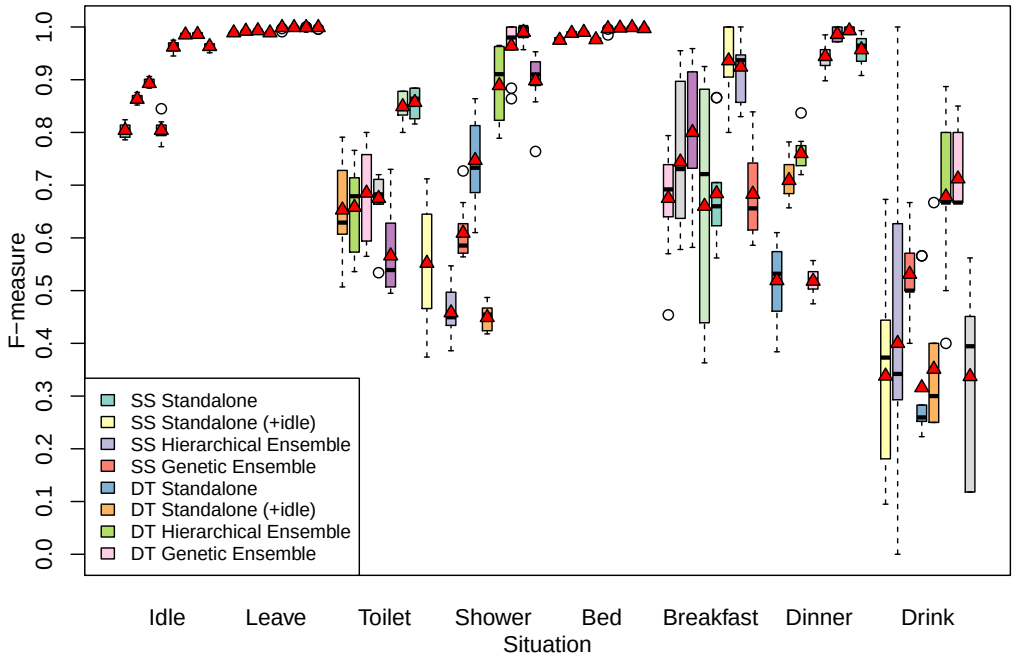


Figure 7.15: A comparison of the average f-measure performance of the standalone and ensemble classification approaches on the House A dataset including idle time slices.

7.4.8 Exp. 8: Comparison with Alternative Situation Recognition Approaches

Figures 7.16 and 7.17 compare the classifiers generated using the House A dataset against the Weka Naïve Bayes, standard J48 decision tree, and Support Vector Machine implementations both when idle time slices are excluded and included.

The relative performance of the techniques remains similar to the CASL dataset evaluation. Of the standard machine-learning techniques, the J48 decision tree outperforms the Naïve Bayes and Support Vector Machine implementations when *idle* time slices are present (by 15%–16%) and when they are not (by 4%).

The J48 decision tree also outperforms the standalone specification set (by 6%) when *idle* time slices are included, however the standalone specification set exhibits 2% better performance when they are excluded. All other techniques improve upon these standard models, with their relative performance discussed in the previous experiment.

The work reported in the literature varies in its experimental methodology. Ye's Context Lattice (Ye09) is evaluated using 10-fold validation, with idle time slices excluded.

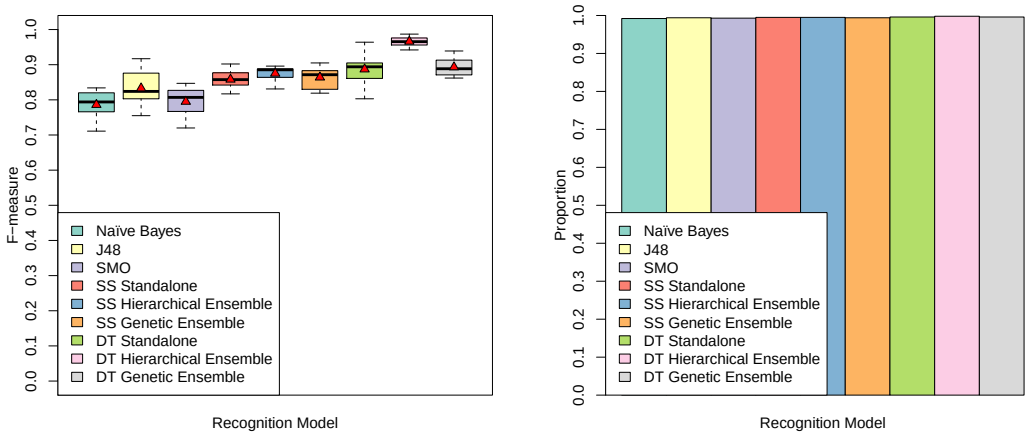


Figure 7.16: A comparison of the average f-measure performance of selected techniques on the House A dataset excluding idle time slices.

Ye's evaluation also ignores time slices where no sensors fire. This potentially leads to improved results over when such time slices are included. McKeever's approach using extended Dempster-Shafter theory (McK11) also excluded idle time slices, but adopts a 'miss one day out methodology', as described in Section 7.2.2. Also adopting a 'miss one day out methodology', van Kasteren (vKNEK08), evaluate Hidden Markov Model and Conditional Random Field techniques (we compare with the HMM here, which performs best), this time including idle time slices. The results of each of these approaches are summarised in Table 7.12. In the table, the term *class accuracy*, which is used in the literature, shares its definition with average recall per class, and is used by these authors as the primary metric for comparison. The f-measure for van Kasteren's HMM and CRF techniques does not appear in the literature, but was calculated from the author's data (vKNEK08).

We compare these results directly with the hierarchical semantic decision tree classifier that provides the best recognition results from this work. The approach shows a class accuracy improvement of 9.5% and f-measure improvement of 0.07 in comparison to Ye's work. In comparison to McKeever, there is a class accuracy improvement of 17.9% and an f-measure improvement of 0.23. In comparison with the HMM of van Kasteren et al., this work exhibits a slightly lower class accuracy of 1.7%, but a higher f-measure overall of 0.18 due to higher precision. The general reduction in recognition accuracy when using the miss one day out evaluation is likely due to the effect of no stratification being used, leading to the training or test datasets that are less representative of situation occurrences in the overall data set.

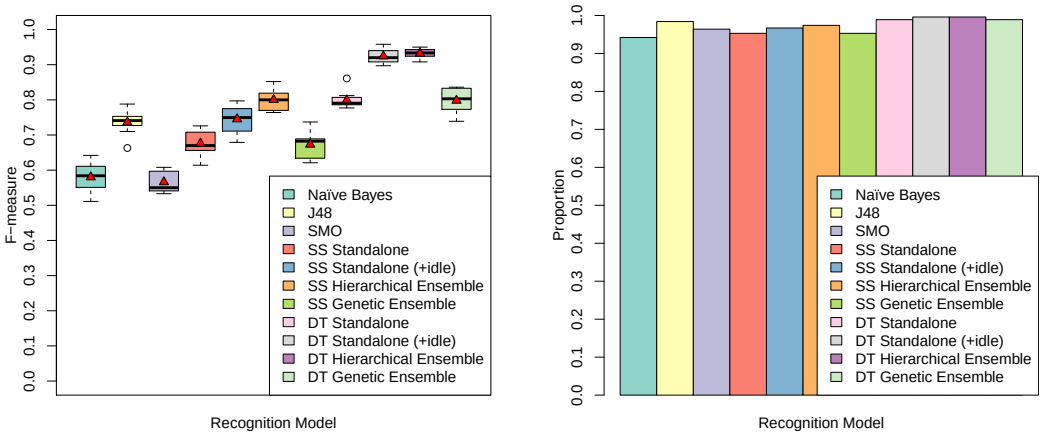


Figure 7.17: A comparison of the average f-measure performance of selected techniques on the House A dataset including idle time slices.

Technique	**Ye's Lattice**	**McKeever's EDST**	**van Kast. HMM**
Intelligible	Yes	Yes	No
Methodology	10-fold	Miss One Day	Miss One Day
Idle time slices	No	No	Yes
Class Accuracy	88.3%	69%	79.4%
F-measure	0.9	0.7	0.63
Technique	**Hierarchical DT**	**Hierarchical DT**	**Hierarchical DT**
Intelligible	Yes	Yes	Yes
Methodology	10-fold	Miss One Day	Miss One Day
Idle time slices	No	No	Yes
Class Accuracy	97.8%	86.9%	77.7%
F-measure	0.97	0.86	0.81

Table 7.12: A summary of evaluations using the House A dataset taken from this thesis and the literature (Ye09; McK11; vKNEK08).

7.5 Accuracy on Further Datasets

Appendix D charts the situation recognition accuracy on the Ink-12, House B, and House C datasets.

7.6 Discussion

To validate the final two parts of our hypothesis, that:

1. The semantic constructs of the ontology model provide an expressive basis from which to learn situation recognition models that display comparable performance

with more complex machine learning models, while retaining intelligibility;

2. The hybrid semantic model and learning-based approach can be further exploited to automatically construct ensemble classifiers with either improved recognition accuracy, intelligibility or both.

we carried out experiments, using the CASL and House A datasets, to assess the situation accuracy of the standalone and ensemble situation recognition models we propose, the intelligibility of the models, and the learning process's capability to generate smaller models. We also investigated alternative approaches to constructing a genetic ensemble classifier, and examined the effect of the *TimeSince* model interpretation on recognition accuracy. Across both datasets, we compared all our techniques to standard machine learning techniques, and to other results published in the literature.

Based on the results of this evaluation, we discuss the benefits and limitations of our proposed approaches to situation recognition.

Performance of the Standalone Models In both the CASL and House A datasets, the standalone specification set model – the simplest approach we propose – provides good situation recognition performance. It outperforms both the standard Weka Support Vector Machine and Naïve Bayes implementations on both, and slightly outperforms the standard J48 implementation on the House A dataset when idle time slices are excluded (by 2%), however, the standard J48 model outperforms the standalone specification set when idle time slices are included (by 6%), and on the CASL dataset (by 4%).

The standalone decision tree model we propose, which takes advantage of the information space model and uses the genetic-algorithm as a search heuristic, uniformly outperforms all the standard machine learning techniques, and the standalone specification set, across both datasets. In our experiments on the House A dataset, we also showed that the *TimeSince* interpretation of data we proposed improves recognition accuracy over other model interpretations presented in the literature.

The performance of both standalone techniques degrades when idle time slices are included in the House A dataset.

On the CASL dataset, the standalone specification set performs comparably with McKeever's manually tuned, extended Dempster-Shafer theory approach (McK11), while the other techniques we present improve on this significantly. On the House A dataset, the hierarchical decision tree model improves upon the results of Ye (Ye09) and McKeever (McK11) when idle time slices are not considered. When these time

slices are included, our model displays lower recall than van Kasteren's Hidden Markov Model approach (vKNEK08), but higher precision due to the explicit handling of idle time slices in a post-processing step.

Intelligibility of the Standalone Models We have shown that the performance of the standalone specification set model, although lower in accuracy, is typically achieved with reference to a smaller number of context statements than the decision tree model. Experiments on the CASL dataset show both that the primary contributor to this is the existence of subsumption relationships in the expert-encoded information space model, and that the genetic algorithm is effective in pruning redundant context statements from the specification.

Although the decision tree model consistently contains more nodes than the specification set does terms, experiments on both datasets also show that using the genetic algorithm as a search heuristic supports the construction of decision trees that are both smaller in size and achieve higher recognition accuracy than a baseline approach fed all model information as input.

Through analysing examples of standalone decision processes, we highlighted how their intelligibility can be used to diagnose problems with poor recognition accuracy, and give insight into where additional sensing capability is likely to improve recognition accuracy.

We noted that the performance of both standalone techniques degrades when idle time slices are included in the House A dataset. One knock on effect is that the intelligibility of the models also decreases due to this recognition confusion.

It is not straightforward to objectively compare the different techniques from the literature in terms of intelligibility. McKeever's approach relies on manually mapping sensors to the situations that their firing might be indicative of, so the resultant models are clearly intelligible at the sensor–situation level. It is unclear if the reported lower performance of this approach is a consequence of the technique used, or due to the limits of human capability to construct a better performing model. However, we might reasonably assume that if machine learning was applied to generate a model structure (for example, by adapting our approach), the result would be intelligible, although perhaps less so than the manual approach. The output of Ye's lattice model gives a set of probabilities corresponding to likelihood of each situation's occurrence given the conjunction of sensors currently firing. Taken in isolation, the result says nothing about the contribution each sensor makes to the result, so we argue that this is less intelligible that either our approach or McKeever's. However, the complete lattice

structure could be studied by an expert to gain insight into the workings of the model. Finally the Hidden Markov Model used by van Kasteren is a black-box technique, the decision processes of which cannot easily be scrutinised. On balance, we consider that the ability to abstract knowledge that our information space model supports, in addition to using a learning algorithm that targets the generation of small specification sets and decision tree models, makes it likely that our approaches are more intelligible than these alternatives.

Performance of the Ensemble Models We have shown that the specification set genetic ensemble significantly outperforms the hierarchical ensemble and standalone model on the CASL dataset, primarily due to its increased ability to distinguish the *reading* situation. For the decision tree model, the hierarchical ensemble provides the biggest performance gain overall.

On the House A dataset, it is the hierarchical ensemble that performs best for both models, consistently identifying the *drink* situation more accurately than the other approaches. Further to this, a clear gain in accuracy is observed from introducing specialised handling for idle time slices, whether it is used to post-process the results of a standalone classifier, or a hierarchical classifier of either technique.

We note that, on the House A dataset, the hierarchical specification set model has near identical performance to the more powerful standalone decision tree model.

Intelligibility of the Ensemble Models When factoring in intelligibility, there is a clear preference towards the hierarchical ensemble over the genetic ensemble. This is primarily because the genetic ensemble requires one full specification set or decision tree per situation, whereas the hierarchical ensemble partitions the recognition problem into multiple, smaller, classification problems, of which there are less overall. As each grouping in the hierarchy either *i)* tries to capture a grouping at only a high level of abstraction, or *ii)* tries to disambiguate only a small number of closely related things, this leads to more readable models overall. We illustrated this in Appendix C with an example using the House A dataset.

A strength of introducing idle filtering as a post-classification step when either the standalone or hierarchical ensemble are used is that these time slices do not affect the construction of the core part of the model—allowing them to be constructed without consideration of this extra noise. This too aids readability.

Impact of Model Engineering The system engineer has an important role to play in developing an accurate ontology model to serve as an input to the automatic learning process. For example, experiment 6 on the CASL dataset (Effect on Recognition Accuracy of Additional Expert Knowledge) illustrates that a location model based on a space's functional use provides a performance improvement over the physical location model by mitigating some of the noise produced by the positioning system.

However, it is important to note a corollary to this—a poorly designed model will limit the potential performance of the overall system. Failure to design a model that captures the appropriate relationships between sensors that are important to the set of situations to be recognised may hamper recognition accuracy. Further to this, a poor model may also impact the engineer's ability to scrutinise the generated decision processes in order to understand the generated model or debug it.

In this respect, one strong advantage of the hierarchical ensemble is that the engineer can consider the performance of multiple, smaller models individually. For example, the poorest recognition in the House A dataset comes from differentiating the *drink* situation with the two other situations that take place in the kitchen: *breakfast* and *dinner*. If the engineer devises a way to improve *drink* recognition – either through the addition of more sensors, or discovering how the learned model can be tweaked – then only this smaller part of the overall model may need re-designed or re-learned.

Finally, we note that the engineer has an ongoing role to play in monitoring model performance after it has been generated and deployed. If the model performance degrades over time, for example, due to sensor drift (degradation), the engineer may need to reassess how sensor outputs are mapped to the model, or rerun the learning process—in effect recalibrating or rebuilding the model to account for changes that have taken place in the environment over time.

Impact of Recognition Performance for Applications The imperfect nature of situation recognition present questions surrounding performance requirements of situation recognition, namely, what levels of accuracy qualify as "good enough" for a particular application, and when is a small improvement in recognition performance "worth it" if it comes as trade off with computational performance or intelligibility of the decision process?

Ultimately, the nature of the application that uses the recognition process is critical to answering these questions. Not all applications will require the same level of accuracy to be usefully deployed, while some applications will require near perfect accuracy before their deployment is tenable. Such analysis is often subjective, however it can be

guided by considering the potential impact of false positives and false negatives on the application in question. To illustrate, we explore three applications in the following and discuss the impact that recognition performance has on them.

The Whereabouts Clock (BTI+07) is a situated display for the family home that visualises the general whereabouts of family members, inferred from cell tower data, through use of a clock metaphor. In related work, we adapted the Whereabouts Clock to visualise the present and predicted situations of people inferred from sensor data (SYD+13). Here, the display serves as an ambient information source and does not drive any further system behaviours. Thus, the impact of a false positive (displaying someone as engaging in the wrong situation) or negative (failing to detect the occurrence of a situation), is very low. If the situation in which a person is engaged is of interest to the clock observer, other methods of verifying the information exist (for example, by phone call or text message).

Next, consider a simple example where the recognition process directly drives behaviour: The automatic switching of mobile phone profiles based on recognised situation (SAT+99)—for example, to automatically silence a phone during a meeting, or ring loudly in a noisy environment. Considering the meeting situation, a false positive may potentially result in missing an important phone call, while a false negative may cause annoying disruption during a meeting (especially if the "noisy environment" situation is incorrectly recognised in its place). The overall impact of incorrect recognition, although not critical, is higher than in the Whereabouts Clock application. Note that potential exists to design the application in such a way as to mitigate the impact of a misclassification, for example, using the phone vibrate feature as a backup to the ringer when switched off.

Finally, consider the example of a in-home fall detection system for the elderly, designed to alert emergency assistance when a fall is detected. Here, a false negative may result in an elderly person remaining on the floor without assistance for a long period of time, which can result in hypothermia, dehydration, broncho-pneumonia and pressure sores (IMP13). False positives can also be expensive (both in terms of money and time) if assistance is dispatched to the home of the subject unnecessarily, and frequent false positives may lead to overall rejection of the detection system (IMP13). Clearly, for this application, high recognition accuracy is more critical to the viability of the application than in the previous examples.

Although recognition performance requirements are application specific, and hence subjective, these examples provide insight into assessing when a small improvement in recognition performance is "worth it". In the example of the situation-aware Whereabouts Clock, as its performance has little real-world impact, minor improvements in

recognition are unlikely worth pursuing if the overall performance is deemed "good enough". In the example of the mobile phone profiles, where recognition accuracy has limited consequences, there may be some benefit in chasing small improvements. However, if such improvements can only be obtained though a variant with additional computational requirements, this tradeoff may not be desirable if the mobile phone is used as the processing platform. Finally, in the fall detection application, where recognition accuracy may have huge impact on the health of users, the viability of the system, the deployment of resources, and strong links to expenditure, even the smallest performance gains are likely worth the additional effort, and computational cost to achieve them.

Impact of Computational Performance There are two aspects of computational performance to consider in evaluating the above standalone and ensemble techniques: the performance of the learning algorithm and the runtime performance of the generated decision processes. Generalising from the performance results of Experiment 5 on the House A dataset, we discuss the implications for the scalability of the proposed approaches. We also comment on whether the ensemble approaches are worth the extra processing required to provide them.

We begin with the learning algorithm. Genetic-algorithm-based learning is computationally complex, and is orders of magnitude slower than other common techniques. In the cases where the hierarchical ensemble is employed, the learning algorithm must be run once for each grouping of situations, while the genetic ensemble requires the learning algorithm to be run as many times as there are situations.

This thesis investigated two types of decision model: the specification set and the decision tree. The runtime performance of the specification set is characterised as $O(nm)$ where m is the number of situations, and n is the number of concepts. All situation specifications must be evaluated to determine which is the highest scoring, however, only a small subset of all possible concepts are likely to occur in each. The runtime complexity of the decision tree improves on this, given that a single path from root to leaf will only be followed each time the tree is evaluated (that is, the complexity of evaluating the tree is proportional to its depth d). However, there is an opportunity to optimise the implementation of the former approach, as the specification sets associated with each situation are independent and can potentially be evaluated concurrently.

The hierarchical ensemble displays similar computational complexity properties to the standalone models in both cases. In the case of the decision tree, the ensemble has the practical effect of replacing some leaf nodes of one tree with the root node of another. The complexity remains proportional to the combined depth, which, due to the more

general nature of the "grouped" situations, will not necessarily exceed the depth of the equivalent path in a standalone model. In the case of the specification set, the "grouping" effect of the hierarchical ensemble adds one extra situation to be evaluated per level of depth of the hierarchy. However, the number of individual situation specifications to be scored at each level of the hierarchy is reduced as a consequence. As only a single path through the hierarchy is followed, the overall effect of employing the hierarchy will potentially offer a reduction in the total number of situation specifications scored (i.e., ignoring groupings in alternative branches).

Finally, evaluation of the genetic ensemble adds computational complexity to both decision models, requiring, as it does, the evaluation of a classifier for each situation of interest (thus $O(nm^2)$ or $O(dm^2)$ for the two decision processes respectively), and the combination of their results using Dempster's Rule of Combination (which is also proportional to the total number of situations under consideration).

To summarise: The learning algorithm is by far the most computationally complex component of the process described in this thesis, and both ensemble approaches require the genetic algorithm to be run multiple times. With the hierarchical ensemble, the number of groupings is application specific, so may grow slowly as the number of situations increases, however, the genetic ensemble scales poorly with respect to the number of situations, and may become impractical if computational resources are scarce. These concerns are mitigated to some extent by the fact that learning process need only occur once (or very infrequently) for each environment. Considering the runtime performance of the decision processes, the evaluation of the genetic ensemble is more complex than the other techniques, while the runtime performance of the hierarchical ensemble may often rival that of a standalone model. Our experimental results show that runtime performance of all the techniques is appropriate for applications with real-time processing requirements.

No hard and fast rules govern whether the additional complexity of an approach is worth the extra processing required to support it. However, assuming that a significant gain in recognition accuracy is obtainable using a given technique (with "significant" defined with respect to a particular application domain, as discussed above), we draw the following conclusions from the above analysis: *i)* the learning process requires significant computational resources and is not suitable to be run on low powered devices, for example, on a wireless sensor network, *ii)* for reasons of power consumption, it is unlikely you would want to perform learning on a mobile device, such as a smartphone or tablet, unless the dataset and associated model is small *iii)* for platforms with the above restrictions, the learning phase could, as an alternative, be conducted using cloud technology or other support infrastructure, *iv)* the learned standalone or hierarchical

ensemble models of both the decision process variants could be straightforwardly run on such platforms with little difficulty due to their low complexity (indeed, Bharatula et al. (BSLT05) find the Decision Tree to provide the best tradeoff between energy consumption and recognition accuracy on a low-power wearable device from among a range of machine learning techniques), and, *v)*, the computational complexity of evaluating the genetic ensemble at runtime, relative to the other approaches explored in this thesis, would likely limit its desirability in any form of resource constrained device.

7.7 Summary

Having argued earlier that a top-level ontology with associated reasoning framework can simplify domain and application model development for pervasive environments, this chapter sought to validate the remaining two claims of our hypothesis.

To this end we provided evidence using third party datasets to demonstrate experimentally that *i)* the semantic constructs of the ontology model provide an expressive basis from which to learn situation recognition models that display comparable performance with more complex machine learning models, while retaining intelligibility; and, *ii)* the hybrid semantic model and learning-based approach can be further exploited to construct ensemble classifiers with either improved recognition accuracy, intelligibility or both. We examined the core features of each of the models we propose, compared our results with others published in the literature, and discussed the implications of the approaches and performance in terms of model engineering, computation complexity, and impact on different application domains.

The above analysis suggests that we have successfully verified both parts of the hypothesis targeted in this evaluation.

In the next chapter, we present our final conclusions and discuss opportunities for future work.

CHAPTER

EIGHT

CONCLUSION AND FUTURE WORK

This thesis was motivated by the need for situation recognition techniques in pervasive environments to handle noisy real-world data, while remaining intelligible to support the inspection of reasoning decisions such that factors affecting performance can be manually assessed.

The result of this work is a novel hybrid approach to sensor-based situation recognition combining both knowledge- and learning-based aspects. This combination offers improvements over existing techniques, without sacrificing either the intelligibility of the decision processes that the use of machine learning alone often implies, or robustness when knowledge-based techniques are used in isolation.

We began by developing a mathematical model of context, realised as a reusable top-level ontology model, to provides a conceptual backbone for developing domain and application ontologies for pervasive environments. Building on this, we developed a genetic-algorithm-based learning scheme, using the model concepts and relations as predicates, to generate situation recognition decision processes using two human-understandable models. Finally, we proposed two ensemble schemes to enhance the recognition accuracy of the basic models.

The concept and context models developed as part of this thesis extend initial collaborative research (Stevenson et al. (SKDN09), Stevenson et al. (SYDN10), Stevenson et al. (SYD10), and Ye et al. (YSD11a) with additional constructs for modelling adjacent concepts and for supporting the modelling and propagation of uncertainty values associated with information.

The early work reported in Ye et al. (YSD11a) supports only the manual specification of situations through the logical conjunction of statements in the context model—an approach limited by the difficulties associated with specifying such rules when uncertainty is present (as discussed in Section 4.1.1). This thesis provides an alternative approach, by employing learning to automatically identify the relevance and contribution (weight-

ing) of context statements to the process of recognising a set of situations. The process of learning via genetic algorithm and use of ensemble classifiers has no overlap with this earlier published work.

In addition to comparing our works with standard machine learning techniques (including the HMM used by van Kasteren et al. (vKNEK08)), we also compared our work to the approaches of Ye et al. (Ye09) and McKeever et al. (MYC+10) who have published work evaluated on the same datasets. Although McKeever uses Dempster Shafer Theory to map each sensor to the possible set of situations it may entail, this is a manually performed process and involves no intermediate context model. In contrast, we apply DST to merge the output a genetic classifier ensemble, the mapping (each classifier's frame of discernment) being automatically derived from each classifier's performance on training data (Section 5.2). While both our approach and Ye's Context Lattice (Ye09) model the structure of information, Ye's lattice exhaustively models all possible conjunctions of sensor firings in the environment, labelling each with the associated probability of each situation's occurrence given the set of firing sensors. This model is substantially different from our approach, which is based on a engineer design context model that abstracts away from sensor data, and supports the inference of additional information based on what is asserted. Our approach uses the set of possible statements that model the environment, and their defined semantic relationships, as primitives for a genetic algorithm capable of learning specification- and decision-tree-based situation recognition models.

In this chapter we summarise the main research contributions of this thesis and discuss opportunities for future work.

8.1 Contributions

We summarise the contributions of this thesis as follows:

A Survey of the State-of-the-art In Chapter 2 we presented an overview of the significant situation recognition techniques in the pervasive computing literature. We extended from taxonomies used in recent surveys to present an updated survey including literature not covered elsewhere, and presented discussion on the main distinguishing features of the approaches. We identified an emerging trend towards hybrid knowledge- and learning-based approaches, and concluded that despite the vast research effort, there is no "one size fits all" approach to situation recognition. Many open research areas remain.

An Ontology-based Model for Developing Pervasive Systems In Chapter 3 we assessed the different technologies available for modelling pervasive systems, and presented a detailed analysis of the strengths and weaknesses of using Semantic-Web based technology, our preferred choice, for this purpose. This was followed by the development of a reusable top-level ontology model that employs set theory to develop a uniform approach to the modelling of concepts, using a small, core set of semantic properties to relate them. We discussed how the automated reasoning afforded by the model supports the propagation and transformation of information from its low level definitions to higher level concepts, and analysed its benefits, limitations, and practical considerations for adopting the model in the development of pervasive systems.

A Heuristic-based Approach to Generating Performant, Intelligible Situation Recognition Models In Chapter 4 we presented a genetic-algorithm-based learning approach that uses the rich constructs of the ontology model to construct situation recognition models from training data. We applied this learning approach to two human-understandable models: The first based on summing the weighted contributions of context statements, and the second based on the decision tree learning model. In both cases the heuristic provided by the genetic algorithm's fitness function is tailored to search for accurate models whose decision processes remain understandable through restricting the model size.

Two Ensemble Techniques Demonstrated to Increase Recognition Accuracy In Chapter 5 we presented two approaches to constructing recognition model ensembles that leverage aspects of the ontology model semantics and use of genetic algorithm learning approach: The *genetic ensemble* approach, adapts the fitness function of the genetic algorithm to generate an ensemble of classifiers that treat different situations preferentially, while the *semantic hierarchy* leverages expert knowledge to group together similar situations into a multi-stage classification process.

A Methodology for Preparing Datasets In Chapter 6 we provided a guide and walkthrough describing the process of preparing five datasets for application of the learning techniques described in this thesis. Each walkthrough showed how expert knowledge is applied to create the concept hierarchies and define the relationship common to each, and the mapping of sensor data to the model concepts. We also demonstrated how the sharing and reuse of model concepts between datasets, in the manner promoted by our top-level ontology model, reduces engineering effort.

An Evaluation of the Proposed Techniques In Chapter 7 we conducted a full evaluation of the situation recognition techniques proposed in this thesis. The evaluation includes assessment of the recognition accuracy and intelligibility of the standalone and ensemble models, and comparison with standard machine learning techniques and results published in the literature. We presented experiments to validate various aspects of our approach, and compared our results with standard machine learning techniques and other results from the literature.

8.2 Limitations

We have identified several limitations that impact the approach described in this thesis. Firstly, the datasets used to validate our approach, only capture single occupancy residences and single worker behaviour. In each of these datasets, only one situation occurs at any given point in time. Thus, the techniques developed here are not immediately applicable to environments where multiple people carry out independent activities, or where situations overlap.

Secondly, the requirement to construct ontologies by hand depends on the availability of a person with suitable expertise and knowledge. This is a relatively straightforward task for the scale of environments and number of sensors described by the datasets used in this study, however, it becomes more difficult to develop such a model as the size of environment, number of sensors installed, and number of situations to recognise increases. Larger scales will also impact the ease with which decision processes can be scrutinised. Narrow, deep concept hierarchies offer the best opportunities for leveraging generality in the learning process, resulting in smaller decision processes. In contrast, broad, shallow concept hierarchies are likely to result in models that involve many concepts, unless there is a high degree of redundancy present.

Thirdly, genetic-algorithm-based learning is computationally complex, and thus learning is orders of magnitude slower than other common techniques. The critical factors affecting learning are the population size, the complexity of evaluating each member of the population, and the number of generations required to find a solution. However, this concern is largely mitigated by the fact that learning need only occur once (or very infrequently) for each environment, and that the evaluation of a population of candidate solutions is parallelisable, giving scope for optimisation.

Finally, the assessment of a situation model's intelligibility has been based primarily on the absolute and relative sizes of generated models, and our own expert knowledge of the environments and situations that the datasets describe. This thesis has not attempted

to tackle this question of intelligibility from the perspective of a layperson, which we suggest would require a suite of tools to aid the presentation of this information.

8.3 Future Work

During the course of this research, we have identified several opportunities for future work:

Supporting Multi-person and Concurrent Situations We described one of the limitations of the described techniques as not being directly applicable to environments where multiple people carry out activities independently, or where situations overlap. Work described in Ye and Stevenson (YS13) presents initial work towards partitioning the sensor events of concurrent or interleaved situations into distinct streams based on the semantic similarity of the concepts they represent. Thus, the challenge of recognising concurrent situations is simplified to recognising multiple independent situations. This general motivation: Pre-processing data or incorporating expert knowledge to simplify what needs to be recognised, instead of increasing model complexity is an important avenue of investigation when pursuing concurrent, overlapping, and multi-person situations.

Towards Unsupervised Modelling and Segmentation The two most time consuming activities of the situation recognition process we present lie in *i)* the collection of the datasets and their annotation with accurate ground truth, and *ii)* the construction of the necessary domain ontologies to accompany a dataset. There is scope for automating, or partially automating, both of these aspects.

With respect to reducing the need for dataset annotation to a semi-supervised level, we might use the ability to partition sensor streams based on their semantics (YS13) to yield streams of sensor events that could be given a single annotation, with windowing applied over the stream duration. This may have the added benefit of excluding semantically different sensor events that occur at the end of the previous situation, or start of the next, to provide cleaner recognition at points of transition. It may be possible to extend this to an unsupervised approach by using the duration of the situation and the sensors it involves to automatically lookup and assign a label to the segmented sensor stream. That is, the inverse of the approach described by Gu et al. (GCTL10).

The automatic construction of taxonomies and ontologies is a research field in itself (MWDB13), and applying existing techniques to gain a comparative performance

baseline is a starting point towards automating or semi-automating the knowledge specification process.

Supporting Transfer Learning An important target for research in situation recognition is the ability to learn a model in one environment for which training data is available, and successfully apply that model to an unseen environment for which no, or little, training data exists. Cook et al. (CFK13). propose some grand challenges in this area.

Approaches to transfer learning typically require the mapping of the problem domain from one environment to another. The approach we present in this thesis to building environment models using a shared conceptual substrate may provide a foundational step towards a formal mapping process, improving upon existing ad hoc mappings (vKEK10a).

A Metric Capturing Semantic Confusion Precision, recall, and f-measure are the standard measures by which recognition accuracy are assessed. However, based on the observation that there are degrees of misclassification severity – for example, misclassifying an instance of *Dinner* as *Drink* is less significant than misclassifying it as *Shower* – we consider that a new measure, capable of capturing the *semantic distance* between the ground truth and its misclassification, may provide a useful metric.

Standard Evaluation Procedures A large challenge, primarily one of software engineering, is the need for a standard benchmark for evaluating situation recognition techniques for pervasive computing. The literature describes many techniques and many datasets, but very few of either are publicly available.

Although standard representation formats for encoding sensor datasets (YSD+11b; YSD+12; BMJ+11) have been proposed, this only addresses one aspect of this problem, and must go hand in hand with a suite of tools to automate the evaluation process, and standard sets of evaluations measures: In addition to our proposal for a metric capturing semantic confusion, researchers have proposed other metrics, such as evaluating whether an activity was recognised uninterruptedly, or whether it was recognised at all, or quantifying the smoothness of recognition transition from one situation to the next.

Explanatory Information and Monitoring for Decision Processes Clear et al. (CSH+09) present an interactive tool to aid the manual design of logical situation specifications, while recent works by Lim et al. (LD10) and Fong et al. (FIR13) outline

frameworks for representing learned decision processes symbolically and verbally to provide users with intelligible explanations. This includes, for example, measuring how much each sensed feature contributes to inference of a state, the confidence in a classification, and indicating why other states were not inferred. We see such research as complementary to the work in this thesis: While we do not consider HCI techniques for presenting model explanations to end users, we can intuitively assume that, on balance, a simpler model lends itself to easier explanation than a more complex one.

There is scope to expand work in this area to the runtime monitoring and representation of a decision process's performance over time. For example, if one particular decision pathway become less frequently activated, or stops being activated all together, this may indicate an underlying sensor error. In the context of monitoring Activities of Daily Living, changes in how situations occur, either in their frequency, duration, or sequence may provide health indicators that could be useful to a medical practitioner or care giver.

Mobile and Highly Distributed Situation Recognition While the work in this thesis focusses on smart environments, the possibility to have millions of computational devices interconnected across urban environments opens up novel application areas. In such highly distributed scenarios, applications must gain awareness as a result of opportunistic encounters with co-located devices, a departure from traditional reasoning approaches. Here, pervasive applications are challenged to become aware of their surroundings: to discover, filter and reason on information relevant to their goals. Without centralised services to control information flow, decentralised mechanisms must partition these responsibilities across the environment.

In Stevenson et al. (SFMM+) we present a vision towards realising situation awareness in such environments, while in Stevenson et al. (SPM+13) and Stevenson et al. (SCY+13), we outline a bio-chemically inspired approach towards constructing complex self-organising awareness algorithms for data collection, reasoning, and querying.

With respect to situation recognition, one of the most challenging research opportunities lies in developing techniques that exhibit real-time responsiveness and robustness to changes across the environment. In such scenarios, the receipt of uncertain, untrustworthy, and out-of-date information will be common, and situation recognition models must account for this. The evaluation of such models is also challenging due to the lack of real-world urban datasets with annotated ground truth. Simulation must typically be relied on, however state-of-the-art tools are currently limited in their realism.

BIBLIOGRAPHY

[ACHZ11] Keith Alexander, Richard Cyganiak, Michael Hausenblas, and Jun Zhao. Describing linked datasets with the VoID vocabulary. W3C interest group note, W3C, March 2011.

[ADF07] Fahd Albinali, Nigel Davies, and Adrian Friday. Structural learning of activities from sparse datasets. In *Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications*, PERCOM '07, pages 221–228, Washington, DC, USA, 2007. IEEE Computer Society.

[AE04] W. Abdelsalam and Y. Ebrahim. A bayesian-networks-based approach for managing uncertainty in location-tracking applications. In *Canadian Conference on Electrical and Computer Engineering*, volume 4, pages 2205–2208 Vol.4, May 2004.

[AKA91] David W. Aha, Dennis Kibler, and Marc K. Albert. Instance-based learning algorithms. *Machine Learning*, 6(1):37–66, 1991.

[ANH07] C. B. Anagnostopoulos, Y. Ntarladimas, and S. Hadjiefthymiades. Situational computing: An innovative architecture with imprecise reasoning. *Journal of System and Software*, 80(12):1993–2014, 2007.

[AP96] Kamal M. Ali and Michael J. Pazzani. Error reduction through learning multiple descriptions. *Machine Learning*, 24:173–202, September 1996.

[AR11] J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, 43(3):16:1–16:43, April 2011.

[ATH03] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in Neural Information Processing Systems 15*, pages 561–568. MIT Press, 2003.

[AY09] Louis Atallah and Guang-Zhong Yang. Review: The use of pervasive sensing for behaviour profiling - a survey. *Pervasive and Mobile Computing*, 5(5):447–464, October 2009.

[Bar05] Jakob E. Bardram. The java context awareness framework (jcaf) - a service infrastructure and programming framework for context-aware applications. In *Proceedings of the Third International Conference on Pervasive Computing*, PERVASIVE'05, pages 98–115, Berlin, Heidelberg, 2005. Springer-Verlag.

[BCM+03] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[BCR09] O. Brdiczka, J.L. Crowley, and P. Reignier. Learning situation models in a smart home. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):56–63, Feb 2009.

[BCW05] C. Bizer, R. Cyganiak, and E. R. Watkins. NG4J – Named Graphs API for Jena. In *Proceedings of the 2nd European Semantic Web Conference*, Heraklion, Greece, 2005.

[BG07] Irad Ben-Gal. Bayesian Networks. In F. Ruggeri, R. Kenett, and F. W. Faltin, editors, *Encyclopedia of statistics in quality and reliability*. John Wiley and Sons, 2007.

[BGB06] Bruno Bouchard, Sylvain Giroux, and Abdenour Bouzouane. A smart home agent for plan recognition of cognitively-impaired patients. *Journal of Computers*, 1(5):53–62, 2006.

[BH06] M. Babik and L. Hluchy. Deep integration of Python with web ontology language. In *Proceedings of the 2nd Workshop on Scripting for the Semantic Web*, Budva, Montenegro, 2006.

[BHBL09] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 5(3):1–22, March 2009.

[BI04] Ling Bao and Stephen S. Intille. Activity recognition from user-annotated acceleration data. In Alois Ferscha and Friedemann Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 1–17. Springer Berlin Heidelberg, 2004.

[BK06] Christian Bauer and Gavin King. *Java Persistence with Hibernate*. Manning Publications Co., Greenwich, CT, USA, 2006.

[BLC11] Tim Berners-Lee and Dan Connolly. Notation3 (N3): A readable rdf syntax. W3C team submission, W3C, 2011. `http://www.w3.org/TeamSubmission/n3/`.

[BLHL01] Tim Berners-Lee, James Hendler, and Ora Lassila. The semantic web. *Scientific American*, 284(5):34–43, May 2001.

[Blu04] Phil Blunsom. Hidden markov models. Technical report, 2004.

[BMJ+11] Arne Bröring, Patrick Maué, Krzysztof Janowicz, Daniel Nüst, and Christian Malewski. Semantically-enabled sensor plug & play for the sensor web. *Sensors*, 11(8):7568–7605, 2011.

[BMPM09] Payam Barnaghi, Stefan Meissner, Mirko Presser, and Klaus Moessner. Sense and sens'ability: Semantic data modelling for sensor networks. In *Proceedings of the ICT Mobile Summit*, 2009.

[BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, March 2003.

[BOP97] M. Brand, N. Oliver, and A. Pentland. Coupled hidden markov models for complex action recognition. In *Proceedings of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 994–999, Jun 1997.

[Bou05] A. Bouzouane. Towards an authority sharing based on the description logic action model. In *Intelligent Agent Technology, IEEE/WIC/ACM International Conference on*, pages 277–280, Sept 2005.

[BPd+08] A. Bandara, T. Payne, D. de Roure, N. Gibbins, and T. Lewis. Semantic resource matching for pervasive environments: The approach and its evaluation. Technical report, University of Southampton, 2008.

[BPSM+] Tim Bray, Jean Paoli, C. Michael Sperberg-McQueen, Eve Maler, and François Yergeau. World Wide Web Consortium, Recommendation REC-xml-20081126, November.

[BPSW70] Leonard E Baum, Ted Petrie, George Soules, and Norman Weiss. A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains. *The annals of mathematical statistics*, pages 164–171, 1970.

[Bro96] P. J. Brown. The Stick-e Document: a Framework for Creating Context-aware Applications. In *Proceedings of EP'96, Palo Alto*, pages 259–272, January 1996.

[BSLT05] Nagendra B Bharatula, Mathias Stäger, Paul Lukowicz, and Gerhard Tröster. Empirical study of design choices in multi-sensor context recognition systems. In *IFAWC-International Forum on Applied Wearable Computing*, 2005.

[BTI+07] Barry Brown, Alex S. Taylor, Shahram Izadi, Abigail Sellen, Joseph 'Jofish' Kaye, and Rachel Eardley. Locating family values: A field trial of the whereabouts clock. In *Proceedings of the 9th International Conference on Ubiquitous Computing*, UbiComp '07, pages 354–371, Berlin, Heidelberg, 2007. Springer-Verlag.

[BVMR06] Oliver Brdiczka, Dominique Vaufreydaz, Jerome Maisonnasse, and Patrick Reignier. Unsupervised segmentation of meeting configurations and activities using speech activity detection. In Ilias Maglogiannis, Kostas Karpouzis, and Max Bramer, editors, *Artificial Intelligence Applications and Innovations*, volume 204 of *IFIP International Federation for Information Processing*, pages 195–203. Springer US, 2006.

[Car01] Sandra Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, March 2001.

[CBB+12] Michael Compton, Payam Barnaghi, Luis Bermudez, Raul Garcia-Castro, Oscar Corcho, Simon Cox, John Graybeal, Manfred Hauswirth, Cory Henson, Arthur Herzog, Vincent Huang, Krzysztof Janowicz, W. David Kelsey, Danh Le Phuoc, Laurent Lefort, Myriam Leggieri, Holger Neuhaus, Andriy Nikolov, Kevin Page, Alexandre Passant, Amit Sheth, and Kerry Taylor. The SSN Ontology of the W3C Semantic Sensor Network Incubator Group. *Web Semantics: Science, Services and Agents on the World Wide Web*, 0(0), 2012.

[CCDG05] Joëlle Coutaz, James L. Crowley, Simon Dobson, and David Garlan. Context is key. *Communications of the ACM*, 48(3):49–53, 2005.

[CCKM01] Paul Castro, Patrick Chiu, Ted Kremenek, and Richard R. Muntz. A probabilistic room location service for wireless networked environments. In *Proceedings of the 3rd International Conference on Ubiquitous Computing*, UbiComp '01, pages 18–34, London, UK, UK, 2001. Springer-Verlag.

[CFJ03] Harry Chen, Tim Finin, and Anupam Joshi. An ontology for context-aware pervasive computing environments. *Knowledge Engineering Review*, 18(3):197–207, 2003.

[CFJ04] Harry Chen, Tim Finin, and Anupam Joshi. Semantic web in the context broker architecture. In *PERCOM '04: Proceedings of the second IEEE international conference on pervasive computing and communications*, pages 277 – 286, 2004.

[CFK13] Diane Cook, Kyle D. Feuz, and Narayanan C. Krishnan. Transfer learning for activity recognition: a survey. *Knowledge and Information Systems*, 36(3):537–556, 2013.

[CFPV12] Pedro Chahuara, Anthony Fleury, Francois Portet, and Michel Vacher. Using markov logic network for on-line activity recognition from non-visual home automation sensors. In Fabio Paterna, Boris Ruyter, Panos Markopoulos, Carmen Santoro, Evert Loenen, and Kris Luyten, editors, *Ambient Intelligence*, volume 7683 of *Lecture Notes in Computer Science*, pages 177–192. Springer Berlin Heidelberg, 2012.

[CGF+10] Gabe Cohn, Sidhant Gupta, Jon Froehlich, Eric Larson, and Shwetak N. Patel. Gassense: Appliance-level, single-point sensing of gas activity in the home. In *Proceedings of the 8th International Conference on Pervasive Computing*, Pervasive'10, pages 265–282, Berlin, Heidelberg, 2010. Springer-Verlag.

[CH92] S. Le Cessie and J. C. Van Houwelingen. Ridge Estimators in Logistic Regression. *Applied Statistics*, 41(1):191–201, 1992.

[CHL+09] Michael Compton, Cory Henson, Laurent Lefort, Holger Neuhaus, and Amit Sheth. A survey of the semantic specification of sensors. In *Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN09)*, pages 17–32, 2009.

[CHN+12] Liming Chen, J. Hoey, C.D. Nugent, D.J. Cook, and Zhiwen Yu. Sensor-based activity recognition. *Systems, Man, and Cybernetics, Part C:*

Applications and Reviews, IEEE Transactions on, 42(6):790–808, Nov 2012.

[CLC+02] Norman H. Cohen, Hui Lei, Paul Castro, John S. Davis II, and Apratim Purakayastha. Composing pervasive data using iQL. In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, WMCSA '02, pages 94–104, Washington, DC, USA, 2002. IEEE Computer Society.

[CLK08] Guanling Chen, Ming Li, and David Kotz. Data-centric middleware for context-aware pervasive computing. *Pervasive and Mobile Computing*, 4(2):216–253, 2008.

[CMD99] Keith Cheverst, Keith Mitchell, and Nigel Davies. Design of an object model for a context sensitive tourist guide. *Computers and Graphics*, 23(6):883–891, December 1999.

[CNM+08] Liming Chen, Chris Nugent, Maurice Mulvenna, Dewar Finlay, Xin Hong, and Michael Poland. A logical framework for behaviour reasoning and assistance in a smart home. *International Journal of Assistive Robotics and Mechatronics*, 9(4):20–634, 2008.

[CNM+09] Liming Chen, Chris Nugent, Maurice Mulvenna, Dewar Finlay, and Xin Hong. Semantic smart homes: Towards knowledge rich assisted living environments. *Intelligent Patient Management*, 189:279–296, 2009.

[CP99] B. Clarkson and A. Pentland. Unsupervised clustering of ambulatory audio and video. In *Proceedings of the Acoustics, Speech, and Signal Processing, 1999. On 1999 IEEE International Conference - Volume 06*, ICASSP '99, pages 3037–3040, Washington, DC, USA, 1999. IEEE Computer Society.

[CSE09] D.J. Cook and M Schmitter-Edgecombe. Assessing the quality of activities in a smart environment. *Methods of information in medicine*, 48(5):480, 2009.

[CSH+09] Adrian K. Clear, Ross Shannon, Thomas Holland, Aaron J. Quigley, Simon A. Dobson, and Paddy Nixon. Situvis: A visual tool for modeling a user's behaviour patterns in a pervasive environment. In *Pervasive*, pages 327–341, 2009.

[CWP12] Belkacem Chikhaoui, Shengrui Wang, and Helene Pigot. ADR-SPLDA: Activity discovery and recognition by combining sequential patterns and Latent Dirichlet Allocation. *Pervasive and Mobile Computing*, 8(6):845–862, December 2012.

[CY05] Xiaoyong Chai and Qiang Yang. Multiple-goal recognition from low-level signals. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 1*, AAAI'05, pages 3–8. AAAI Press, 2005.

[CYL+07] Lorcan Coyle, Juan Ye, Emerson Loureiro, Stephen Knox, Simon Dobson, and Paddy Nixon. A proposed approach to evaluate the accuracy of tag-based location systems. In *Ubicomp 2007 Workshops Proceedings of the First Workshop on Ubiquitous Systems Evaluation*, pages 292–296, 2007.

[CYW05] Datong Chen, Jie Yang, and Howard Wactlar. A study of detecting social interaction with sensors in a nursing home environment. In Nicu Sebe, Michael Lew, and Thomas S. Huang, editors, *Computer Vision in Human-Computer Interaction*, volume 3766 of *Lecture Notes in Computer Science*, pages 199–210. Springer Berlin Heidelberg, 2005.

[DA00] Anind K. Dey and G. D. Abowd. Towards a better understanding of context and context-awareness. In *Workshop on The What, Who, Where, When, and How of Context-Awareness, as part of the 2000 Conference on Human Factors in Computing Systems (CHI 2000)*, The Hague, The Netherlands, April 2000.

[Dar59] Charles Darwin. *On the origin of species by means of natural selection, or, the preservation of favoured races in the struggle for life*. John Murray London, 1859.

[DBPV05] T.V. Duong, H.H. Bui, D.Q. Phung, and S. Venkatesh. Activity recognition and abnormality detection with the switching hidden semi-markov model. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 838–845 vol. 1, June 2005.

[Dem68] Arthur P Dempster. A generalization of bayesian inference. *the Royal Statistical Society, Series B*, 30:205–247, 1968.

[DHKZG08] Pari Delir Haghighi, Shonali Krishnaswamy, Arkady Zaslavsky, and Mohamed Medhat Gaber. Reasoning about context in uncertain pervas-

ive computing environments. In *Smart Sensing and Context*, volume 5279 of *Lecture Notes in Computer Science*, pages 112–125. Springer Berlin Heidelberg, 2008.

[DNC+07] Simon Dobson, Paddy Nixon, Lorcan Coyle, Steve Neely, Graeme Stevenson, and Graham Williamson. Construct: An open source pervasive systems platform. In *CCNC 2007: 4th IEEE Consumer Communications and Networking Conference*, pages 1203–1204, Las Vegas, NV, USA., January 2007.

[Dou04] P. Dourish. What We Talk About When We Talk About Context. *Personal and Ubiquitous Computing*, 8(1):19–30, February 2004.

[DP04] Zhongli Ding and Yun Peng. A probabilistic extension to ontology language owl. In *Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 4 - Volume 4*, HICSS '04, pages 40111.1–, Washington, DC, USA, 2004. IEEE Computer Society.

[DSA01] Anind K. Dey, D. Salber, and G.D. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human Computer Interaction*, 16(2-4):97–166, 2001.

[Ebe02] A. Eberhart. Automatic generation of Java/SQL based inference engines from RDF Schema and RuleML. In *Proceedings of the First International Semantic Web Conference, Chia, Sardinia, Italy*, pages 102–116. Springer, 2002.

[EBMM03] A.A. Efros, A.C. Berg, G. Mori, and J. Malik. Recognizing action at a distance. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 726–733 vol.2, Oct 2003.

[emp14] Empire: A JPA implementation for RDF. Website, 2014. `https://github.com/clarkparsia/Empire`.

[ETB04] Rainer Eckstein, Robert Tolksdorf, and Christian Bizer, editors. *SWEB 2004: Proceedings of the international workshop on semantic web technologies in electronic business*, 2004.

[EWK90] Ramez Elmasri, Gene T. J. Wuu, and Yeong-Joon Kim. The time index—an access structure for temporal data. In *Proceedings of the sixteenth international conference on Very large databases*, pages 1–12, Brisbane, Australia, 1990. Morgan Kaufmann Publishers Inc.

[FAI+05] David A. Forsyth, Okan Arikan, Leslie Ikemoto, James O'Brien, and Deva Ramanan. Computational studies of human motion: part 1, tracking and motion synthesis. *Foundations and Trends in Computer Graphics and Vision*, 1(2-3):77–254, 2005.

[FBC08] Jeff Forcier, Paul Bissex, and Wesley Chun. *Python Web Development with Django*. Addison-Wesley Professional, 1 edition, 2008.

[FBF77] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):209–226, September 1977.

[FD13] Lei Fang and Simon Dobson. In-network sensor data modelling methods for fault detection. In *Evolving Ambient Intelligence*, volume 413 of *Communications in Computer and Information Science*, pages 176–189. Springer International Publishing, 2013.

[FGP11] Katayoun Farrahi and Daniel Gatica-Perez. Discovering routines from large-scale human locations using probabilistic topic models. *ACM Transactions on Intelligent Systems Technology*, 2(1):3:1–3:27, January 2011.

[FHH+05] Eibe Frank, Mark A. Hall, Geoffrey Holmes, Richard Kirkby, Bernhard Pfahringer, Ian H. Witten, and Leonhard Trigg. WEKA - a machine learning workbench for data mining. In Oded Maimon and Lior Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer, 2005.

[FHT98] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[FIR13] J. Fong, J. Indulska, and R. Robinson. A framework to support intelligibility in pervasive applications. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*, pages 37–42, March 2013.

[FM11] L. Ferrari and M. Mamei. Discovering daily routines from google latitude with topic models. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 432–437, March 2011.

[FPSD09] M. Foth, E. Paulos, C. Satchell, and P. Dourish. Pervasive computing and environmental sustainability: Two conference workshops. *Pervasive Computing, IEEE*, 8(1):78–81, Jan 2009.

[FRMZ11] Laura Ferrari, Alberto Rosi, Marco Mamei, and Franco Zambonelli. Extracting urban patterns from location-based social networks. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, LBSN '11, pages 9–16, New York, NY, USA, 2011. ACM.

[FST98] Shai Fine, Yoram Singer, and Naftali Tishby. The hierarchical hidden markov model: Analysis and applications. *Journal of Machine Learning*, 32(1):41–62, July 1998.

[GB04] Ramanathan V. Guha and Dan Brickley. RDF vocabulary description language 1.0: RDF schema. W3C recommendation, W3C, February 2004. `http://www.w3.org/TR/2004/REC-rdf-schema-20040210/`.

[GCTL10] Tao Gu, Shaxun Chen, Xianping Tao, and Jian Lu. An unsupervised approach to activity recognition and segmentation based on object-use fingerprints. *Data and Knowledge Engineering*, 69(6):533–544, June 2010.

[GD05] K. Grauman and T. Darrell. The pyramid match kernel: discriminative classification with sets of image features. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1458–1465 Vol. 2, Oct 2005.

[GDL+04] Robert Grimm, Janet Davis, Eric Lemar, Adam Macbeth, Steven Swanson, Thomas Anderson, Brian Bershad, Gaetano Borriello, Steven Gribble, and David Wetherall. System support for pervasive applications. *ACM Transactions on Computer Systems*, 22(4):421–486, 2004.

[GHM+08] Bernardo Cuenca Grau, Ian Horrocks, Boris Motik, Bijan Parsia, Peter Patel-Schneider, and Ulrike Sattler. OWL 2: The Next Step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322, October 2008.

[GHV07] Claudio Gutierrez, Carlos A. Hurtado, and Alejandro Vaisman. Introducing time into RDF. *IEEE Transactions on Knowledge and Data Engineering*, 19(2):207–218, 2007.

[GKK+03] Phillip B. Gibbons, Brad Karp, Yan Ke, Suman Nath, and Srinivasan Seshan. Irisnet: An architecture for a worldwide sensor web. *IEEE Pervasive Computing*, 02(4):22–33, 2003.

[GMBM05] Yolanda Gil, Enrico Motta, V. Richard Benjamins, and Mark A. Musen, editors. *ISWC '05: Proceedings of the 4th International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, Galway, Ireland, 2005. Springer.

[Gol03] N. M. Goldman. Ontology-oriented programming: Static typing for the inconsistent programmer. In *Proceedings of the 2nd International Semantic Web Conference*, Sanibel Island, FL, USA, 2003.

[GPBB12] Tobias Grosse-Puppendahl, Eugen Berlin, and Marko Borazio. Enhancing accelerometer-based activity recognition with capacitive proximity sensing. In Fabio Paterna, Boris Ruyter, Panos Markopoulos, Carmen Santoro, Evert Loenen, and Kris Luyten, editors, *Ambient Intelligence*, volume 7683 of *Lecture Notes in Computer Science*, pages 17–32. Springer Berlin Heidelberg, 2012.

[GPVD99] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2. RFC 2608, 1999.

[GPZ+04] Tao Gu, H. K. Pung, D. Q. Zhang, Hung Keng Pung, and Da Qing Zhang. A bayesian approach for dealing with uncertain contexts. In *Advances in Pervasive Computing at Pervasive' 04*, pages 205–2010, 2004.

[GPZ05] Tao Gu, Hung Keng Pung, and Da Qing Zhang. A service-oriented middleware for building context-aware services. *Journal of Network and Computer Appliciations*, 28(1):1–18, January 2005.

[Gro07] Object Management Group. OMG Unified Modeling Language (OMG UML), Infrastructure, V2.1.2. Technical report, November 2007.

[Gru95] Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-computer Studies*, 43(5-6):907–928, 1995.

[GS09] Paul Gearon and Simon Schenk. SPARQL 1.1 update. W3C working draft, W3C, October 2009. http://www.w3.org/TR/2009/WD-sparql11-update-20091022/.

[GSMT+08] Shudi Gao, C. Michael Sperberg-McQueen, Henry S. Thompson, Noah Mendelsohn, David Beech, and Murray Maloney. W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures. World Wide Web Consortium, Working Draft WD-xmlschema11-1-20080620, June 2008.

[Gua98] Nicola Guarino. Formal ontology and information systems. In *FOIS'98: Proceedings of the first international conference on formal ontology in information systems*, pages 3–15, Trento, Italy, June 1998. IOS Press.

[Gui02] Veronica Guidetti. Intelligent information integration systems: extending a lexicon ontology. Master's thesis, Computer Science, University of Modena and Reggio Emilia, 2002.

[GWF97] Bernhard Ganter, Rudolf Wille, and Cornelia Franzke. *Formal concept analysis: mathematical foundations*. Springer-Verlag New York, Inc., 1997.

[GWPZ04] Tao Gu, Xiao Hang Wang, Hung Keng Pung, and Da Qing Zhang. An ontology-based context model in intelligent environments. In *In Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conferences*, pages 270–275, 2004.

[GWW+09] Tao Gu, Zhanqing Wu, Liang Wang, Xianping Tao, and Jian Lu. Mining emerging patterns for recognizing activities of multiple users in pervasive computing. In *Mobile and Ubiquitous Systems: Networking Services, MobiQuitous, 2009. MobiQuitous '09. 6th Annual International*, pages 1–10, July 2009.

[HB11] Matthew Horridge and Sean Bechhofer. The OWL API: A Java API for OWL Ontologies. *Semantic Web Journal*, 2(1):11–21, January 2011.

[HBS07] Tam Huynh, Ulf Blanke, and Bernt Schiele. Scalable recognition of daily activities with wearable sensors. In Jeffrey Hightower, Bernt Schiele, and Thomas Strang, editors, *Location- and Context-Awareness*, volume 4718 of *Lecture Notes in Computer Science*, pages 50–67. Springer Berlin Heidelberg, 2007.

[Hec99] David Heckerman. A tutorial on learning with bayesian networks. In Michael I. Jordan, editor, *Learning in Graphical Models*, pages 301–354. MIT Press, Cambridge, MA, USA, 1999.

[HGKZ07] P. Haghighi, M. Gaber, S. Krishnaswamy, and A. Zaslavsky. An architecture for context-aware adaptive data stream mining. In *Proceedings of the International Workshop on Knowledge Discovery from Ubiquitous Data Stream*, September 2007.

[HI06] Karen Henricksen and Jadwiga Indulska. Developing context-aware pervasive computing applications: Models and approach. *Pervasive and Mobile Computing*, 2(1):37 – 64, 2006.

[HIM05] Karen Henricksen, Jadwiga Indulska, and Ted McFadden. Modelling context information with ORM. *Lecture Notes in Computer Science*, 3762:626–635, 2005.

[HIR02] Karen Henricksen, Jadwiga Indulska, and Andry Rakotonirainy. Modeling context information in pervasive computing systems. In *Proceedings of the First International Conference on Pervasive Computing*, Pervasive '02, pages 167–180, London, UK, UK, 2002. Springer-Verlag.

[HLI04] Karen Henricksen, Steven Livingstone, and Jadwiga Indulska. Towards a hybrid approach to context modeling, reasoning and interoperation. In *The First International Workshop on Advanced Context Modelling, Reasoning and Management*, pages 54–61, Nottingham, U.K., September 2004.

[HMJ+09] Raffay Hamid, Siddhartha Maddi, Amos Johnson, Aaron Bobick, Irfan Essa, and Charles Isbell. A novel sequence representation for unsupervised analysis of human activities. *Artificial Intelligence*, 173(14):1221 – 1244, 2009.

[HN09] Xin Hong and C.D. Nugent. Partitioning time series sensor data for activity recognition. In *Information Technology and Applications in Biomedicine, 2009. ITAB 2009. 9th International Conference on*, pages 1–4, Nov 2009.

[HNM+09] Xin Hong, Chris Nugent, Maurice Mulvenna, Sally McClean, Bryan Scotney, and Steven Devlin. Evidential fusion of sensor data for activity recognition in smart homes. *Pervasive and Mobile Computing*, 5:236–252, 2009.

[HNS11a] R. Helaoui, M. Niepert, and H. Stuckenschmidt. Recognizing interleaved and concurrent activities: A statistical-relational approach. In *Pervasive*

Computing and Communications (PerCom), 2011 IEEE International Conference on, pages 1–9, March 2011.

[HNS11b] Rim Helaoui, Mathias Niepert, and Heiner Stuckenschmidt. Recognizing interleaved and concurrent activities using qualitative and quantitative temporal relationships. *Pervasive and Mobile Computing*, 7(6):660–670, December 2011.

[Hol92] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA, 1992.

[HPSB+04] Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Technical report, National Research Council of Canada, Network Inference, and Stanford University, May 2004.

[HRLL08] M. Hasan, H.A. Rubaiyeat, Yong-Koo Lee, and Sungyoung Lee. A reconfigurable HMM for activity recognition. In *Proceedings of the 10th International Conference on Advanced Communication Technology*, volume 1, pages 843–846, Feb 2008.

[HY08] Derek Hao Hu and Qiang Yang. Cigar: Concurrent and interleaving goal and activity recognition. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pages 1363–1368. AAAI Press, 2008.

[ILB+05] Stephen S. Intille, Kent Larson, J. S. Beaudin, J. Nawyn, E. Munguia Tapia, and P. Kaushik. A living laboratory for the design and evaluation of ubiquitous computing technologies. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1941–1944, New York, NY, USA, 2005. ACM.

[IMP13] Raul Igual, Carlos Medrano, and Inmaculada Plaza. Challenges, issues and trends in fall detection systems. *BioMedical Engineering Online*, 12(1):66, 2013.

[jas14] Jastor: Typesafe, ontology driven RDF access from Java. Website, 2014. `http://jastor.sourceforge.net/`.

[jen14] Jenabean: A library for persisting Java Beans to RDF. Website, 2014. `http://code.google.com/p/jenabean/`.

[jes11] Jess, the rule engine for the Java platform. `http://www.jessrules.com/`, 2011.

[JF02] Brad Johanson and Armando Fox. The event heap: A coordination infrastructure for interactive workspaces. In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, WMCSA '02, pages 83–, Washington, DC, USA, 2002. IEEE Computer Society.

[JS03] Glenn Judd and Peter Steenkiste. Providing contextual information to pervasive computing applications. In *PERCOM '03: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, page 133, Washington, DC, USA, 2003. IEEE Computer Society.

[jso13a] JSON: Javascript object notation. Website, 2013. `http://json.org/`.

[jso13b] JSON schema. Website, 2013. `http://json-schema.org/`.

[KC14] Narayanan C. Krishnan and Diane J. Cook. Activity recognition on streaming sensor data. *Pervasive and Mobile Computing*, 10:138–154, February 2014.

[KEK11] T. L. M. Kasteren, G. Englebienne, and B. J. A. Kröse. Human activity recognition from wireless sensor network data: Benchmark and software. In *Activity Recognition in Pervasive Intelligent Environments*, volume 4 of *Atlantis Ambient and Pervasive Intelligence*, pages 165–186. Atlantis Press, 2011.

[KFM+63] Sidney Katz, Amasa B. Ford, Roland W. Moskowitz, Beverly A. Jackson, and Marjorie W. Jaffe. Studies of Illness in the Aged. *The Journal of the American Medical Association*, 185(12):914–919, September 1963.

[KGS+08] Takayuki Kanda, Dylan F. Glas, Masahiro Shiomi, Hiroshi Ishiguro, and Norihiro Hagita. 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, 2008. ACM.

[KJ04] A. Kalyanpur and D. Jimenez. Automatic mapping of OWL ontologies into Java. In *Proceedings of Software Engineering and Knowledge Engineering*, 2004.

[KMK+03] P. Korpipaa, J. Mantyjarvi, J. Kela, H. Keranen, and E.-J. Malm. Managing context information in mobile devices. *Pervasive Computing, IEEE*, 2(3):42–51, July 2003.

[KNM+06] D. M. Karantonis, M. R. Narayanan, M. Mathie, N. H. Lovell, and B. G. Celler. Implementation of a real-time human movement classifier using a triaxial accelerometer for ambulatory monitoring. *Trans. Info. Tech. Biomed.*, 10(1):156–167, January 2006.

[Koz92] John R. Koza. *Genetic programming: on the programming of computers by means of natural selection*. MIT Press, Cambridge, MA, USA, 1992.

[KPSR+09] Markus Krötzsch, Peter F. Patel-Schneider, Sebastian Rudolph, Pascal Hitzler, and Bijan Parsia. OWL 2 web ontology language primer. Technical report, W3C, October 2009. http://www.w3.org/TR/2009/REC-owl2-primer-20091027/.

[KS86] R Kowalski and M Sergot. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95, January 1986.

[KTH+10] G. Krassnig, D. Tantinger, C. Hofmann, T. Wittenberg, and M. Struck. User-friendly system for recognition of activities with an accelerometer. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on-NO PERMISSIONS*, pages 1–8, March 2010.

[KZP06] Sotiris B Kotsiantis, Ioannis D Zaharakis, and Panayiotis E Pintelas. Machine learning: a review of classification and combining techniques. *Artificial Intelligence Review*, 26(3):159–190, 2006.

[KZSM12] Love Kalra, Xinghui Zhao, AxelJ. Soto, and Evangelos Milios. A two-stage corrective markov model for activities of daily living detection. In Paulo Novais, Kasper Hallenborg, Dante I. Tapia, and Juan M. Corchado Rodriguez, editors, *Ambient Intelligence - Software and Applications*, volume 153 of *Advances in Intelligent and Soft Computing*, pages 171–179. Springer Berlin Heidelberg, 2012.

[Lan13] Markus Lanthaler. Creating 3rd generation web APIs with hydra. In *Proceedings of the 22nd international conference on World Wide Web companion*, WWW '13 Companion, pages 35–38, Republic and Canton of Geneva, Switzerland, 2013. International World Wide Web Conferences Steering Committee.

[LCB06] Jonathan Lester, Tanzeem Choudhury, and Gaetano Borriello. A practical approach to recognizing physical activities. In Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron Quigley, editors, *Pervasive Computing*, volume 3968 of *Lecture Notes in Computer Science*, pages 1–16. Springer Berlin Heidelberg, 2006.

[LCK+05] Jonathan Lester, Tanzeem Choudhury, Nicky Kern, Gaetano Borriello, and Blake Hannaford. A hybrid discriminative/generative approach for modeling human activities. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, IJCAI'05, pages 766–772, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc.

[LD05] Daniel Lowd and Pedro Domingos. Naive bayes models for probability estimation. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, pages 529–536, New York, NY, USA, 2005. ACM.

[LD07] Daniel Lowd and Pedro Domingos. Efficient weight learning for markov logic networks. In *In Proceedings of the Eleventh European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 200–211, 2007.

[LD10] Brian Y. Lim and Anind K. Dey. Toolkit to support intelligibility in context-aware applications. In *Proceedings of the 12th ACM International Conference on Ubiquitous Computing*, Ubicomp '10, pages 13–22, New York, NY, USA, 2010. ACM.

[LFK07] Lin Liao, Dieter Fox, and Henry Kautz. Extracting places and activities from GPS traces using hierarchical conditional random fields. *Int. J. Rob. Res.*, 26(1):119–134, January 2007.

[LH03] L. Li and I. Horrocks. A software framework for matchmaking based on Semantic Web technology. In *Proceedings of the 12th international conference on World Wide Web*, pages 331–339, New York, NY, USA, 2003.

[LHH+13] Cassim Ladha, Nils Hammerla, Emma Hughes, Patrick Olivier, and Thomas Ploetz. Dog's life: Wearable activity recognition for dogs. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 415–418, New York, NY, USA, 2013. ACM.

[LHP+07] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Munguia Tapia, and Stephen Intille. A long-term evaluation of sensing modalities for activity recognition. In *Proceedings of the 9th International Conference on Ubiquitous Computing*, UbiComp '07, pages 483–500, Berlin, Heidelberg, 2007. Springer-Verlag.

[LLD07] Fatiha Latfi, Bernard Lefebvre, and Céline Descheneaux. Ontology-based management of the telehealth smart home, dedicated to elderly in loss of cognitive autonomy. In *The OWLED 2007 Workshop on OWL: Experiences and Directions*, Innsbruck, Austria, june 2007.

[LLPG05] Pantelis Lilis, Irene Lourdi, Christos Papatheodorou, and Manolis Gergatsoulis. A metadata model for representing time-dependent information in cultural collections. In *Proceedings of the first online metadata and semantics research conference (MTSR' 05)*, pages 1–12. Rinton Press, November 2005.

[LMP01] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[Lok04a] Seng W. Loke. Logic programming for context-aware pervasive computing: Language support, characterizing situations, and integration with the web. In *Proceedings of the 2004 IEEE/WIC/ACM International Conference on Web Intelligence*, WI '04, pages 44–50, Washington, DC, USA, 2004. IEEE Computer Society.

[Lok04b] Seng W. Loke. Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective. *Knowledge Engineering Review*, 19(3):213–233, 2004.

[LPLP12] Oscar D. Lara, Alfredo J. Perez, Miguel A. Labrador, and Jose D. Posada. Centinela: A human activity recognition system based on acceleration and vital sign data. *Pervasive and Mobile Computing*, 8(5):717–729, October 2012.

[LVB+03] Jorge E. López de Vergara, Víctor A. Villagrá, Julio Berrocal, Juan I. Asensio, and Roney Pignaton. Semantic management: application of ontologies for the integration of management information models.

In *Proceedings of the IFIP/IEEE eighth international symposium on integrated network management*, pages 131–134, March 2003.

[Mar03] Dimitris Margaritis. *Learning Bayesian network model structure from data*. PhD thesis, University of Pittsburgh, 2003.

[MBK08] Joseph Modayil, Tongxin Bai, and Henry Kautz. Improving the recognition of interleaved activities. In *Proceedings of the 10th International Conference on Ubiquitous Computing*, UbiComp '08, pages 40–43, New York, NY, USA, 2008. ACM.

[MBSM04] Martin Mühlenbrock, Oliver Brdiczka, Dave Snowdon, and Jean-Luc Meunier. 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 13–23, Washington, DC, USA, 2004. IEEE Computer Society.

[McB02] Brian McBride. Jena: A semantic web toolkit. *IEEE Internet Computing*, 6(6):55–59, 2002.

[McK11] Susan McKeever. *Recognising Situations Using Extended Dempster-Shafer Theory*. Ph.D. thesis, University College Dublin, 2011.

[MCLC04] M.J. Mathie, B.G. Celler, N.H. Lovell, and A.C.F. Coster. Classification of basic daily movements using a triaxial accelerometer. *Medical and Biological Engineering and Computing*, 42(5):679–687, 2004.

[MDEK13] G. Meditskos, S. Dasiopoulou, V. Efstathiou, and I. Kompatsiaris. Sp-act: A hybrid framework for complex activity recognition combining OWL and SPARQL rules. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*, pages 25–30, March 2013.

[MEBK09] Jörg Müller, Juliane Exeler, Markus Buzeck, and Antonio Krüger. Reflectivesigns: Digital signs that adapt to audience attention. In *7th International Conference on Pervasive Computing*, pages 17–24, Nara, Japan, May 2009.

[MHK06] Thomas B. Moeslund, Adrian Hilton, and Volker Kruger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(23):90 – 126, 2006.

[Mit97] Thomas M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA, 1st edition, 1997.

[Mit98] Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.

[MKSS13] Takuya Maekawa, Yasue Kishino, Yasushi Sakurai, and Takayuki Suyama. Activity recognition with hand-worn magnetic sensors. *Personal Ubiquitous Comput.*, 17(6):1085–1094, August 2013.

[MM04] Eric Miller and Frank Manola. RDF primer. W3C recommendation, W3C, February 2004. `http://www.w3.org/TR/2004/REC-rdf-primer-20040210/`.

[MMMD11] John C. McCarthy, Christopher Mines, Pascal Matzke, and Yahor Darashkevich. Mobile app internet recasts the software and services landscape. *Forrester Research Consumer PC And Tablet Forecast, 2011 to 2016 (US)*, 2011.

[MRSS06a] U. Maurer, A. Rowe, A. Smailagic, and D.P. Siewiorek. eWatch: a wearable sensor and notification platform. In *Wearable and Implantable Body Sensor Networks, 2006. BSN 2006. International Workshop on*, pages 4 pp.–145, April 2006.

[MRSS06b] Uwe Maurer, Anthony Rowe, Asim Smailagic, and Daniel Siewiorek. Location and activity recognition using eWatch: A wearable sensor platform. In Yang Cai and Julio Abascal, editors, *Ambient Intelligence in Everyday Life*, volume 3864 of *Lecture Notes in Computer Science*, pages 86–102. Springer Berlin Heidelberg, 2006.

[MS01] Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. *IEEE Intelligent Systems*, 16(2):72–79, 2001.

[MSW+05] D. Minnen, T. Starner, J.A. Ward, P. Lukowicz, and G. Tröster. Recognizing and discovering human actions from on-body sensor data. In *IEEE International Conference on Multimedia and Expo*, pages 1545–1548, July 2005.

[MWDB13] Olena Medelyan, Ian H Witten, Anna Divoli, and Jeen Broekstra. Automatic construction of lexicons, taxonomies, ontologies, and other knowledge structures. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3(4):257–279, 2013.

[MWI+07] Rob McCarney, James Warner, Steve Iliffe, Robbert van Haselen, Mark Griffin, and Peter Fisher. The hawthorne effect: a randomised, controlled trial. *BMC Medical Research Methodology*, 7(1):1–8, 2007.

[MY13] Susan McKeever and Juan Ye. A comparison of evidence fusion rules for situation recognition in sensor-based environments. In Michael J. O' Grady, Hamed Vahdat-Nejad, Klaus-Hendrik Wolf, Mauro Dragone, Juan Ye, Carsten Rocker, and Gregory O' Hare, editors, *Evolving Ambient Intelligence*, volume 413 of *Communications in Computer and Information Science*, pages 163–175. Springer International Publishing, 2013.

[MYC+10] Susan Mckeever, Juan Ye, Lorcan Coyle, Chris Bleakley, and Simon Dobson. Activity recognition using temporal evidence theory. *Journal of Ambient Intelligence and Smart Environment*, 2(3):253–269, 2010.

[MYCD09a] Susan McKeever, Juan Ye, Lorcan Coyle, and Simon Dobson. A context quality model to support transparent reasoning with uncertain context. In *Proceedings of the 1st international conference on quality of context*, QuaCon'09, pages 65–75, Berlin, Heidelberg, 2009. Springer-Verlag.

[MYCD09b] Susan McKeever, Juan Ye, Lorcan Coyle, and Simon Dobson. Using Dempster-Shafer theory of evidence for situation inference. In Payam M. Barnaghi, Klaus Moessner, Mirko Presser, and Stefan Meissner, editors, *EuroSSC 2009: Proceedings of the 4th European Conference on Smart Sensing and Context*, volume 5741 of *Lecture Notes in Computer Science*, pages 149–162. Springer, 2009.

[MYK+10] Takuya Maekawa, Yutaka Yanagisawa, Yasue Kishino, Katsuhiko Ishiguro, Koji Kamei, Yasushi Sakurai, and Takeshi Okadome. Object-based activity recognition with heterogeneous sensors on wrist. In Patrik Floren, Antonio Kruger, and Mirjana Spasojevic, editors, *Pervasive Computing*, volume 6030 of *Lecture Notes in Computer Science*, pages 246–264. Springer Berlin Heidelberg, 2010.

[NDHC10] Ehsan Nazerfard, Barnan Das, Lawrence B. Holder, and Diane J. Cook. Conditional random fields for activity recognition in smart environments. In *Proceedings of the 1st ACM International Health Informatics Symposium*, IHI '10, pages 282–286, New York, NY, USA, 2010. ACM.

[neo] The NeOn Toolkit. `http://neon-toolkit.org`.

[New96] Gregory B. Newby. Metric multidimensional information space. In *Proceedings of the 6th Text Retrieval Conference (TREC)*, Maryland, USA, November 1996.

[NFM00] Natasha F. Noy, Ray W. Fergerson, and Mark A. Musen. The knowledge model of protege-2000: combining interoperability and flexibility. In *EKAW '00: Proceedings of the 2nd international conference on knowledge engineering and knowledge management*, Juan-les-Pins, France, 2000.

[NN07] P. Natarajan and R. Nevatia. Coupled Hidden Semi Markov Models for activity recognition. In *IEEE Workshop on Motion and Video Computing*, pages 10–10, Feb 2007.

[NP13] Yunyoung Nam and Jung Wook Park. Physical activity recognition using a single triaxial accelerometer and a barometric sensor for baby and child care in a home environment. *Journal of Ambient Intelligence and Smart Environments*, 5(4):381–402, July 2013.

[ODG+07] E. Oren, R. Delbru, S. Gerke, A. Haller, and S. Decker. ActiveRDF: object-oriented semantic web programming. In *Proceedings of the 16th international conference on World Wide Web*, pages 817–824, New York, NY, USA, 2007. ACM.

[OS96] David W. Opitz and Jude W. Shavlik. Generating accurate and diverse members of a neural-network ensemble. In *Advances in Neural Information Processing Systems*, pages 535–541. MIT Press, 1996.

[OWL01] Leo Obrst, Robert E. Wray, and Howard Liu. Ontological Engineering for B2B E-commerce. In *FOIS '01: Proceedings of the international conference on formal ontology in information systems*, pages 117–126, New York, NY, USA, 2001. ACM Press.

[owl14] OWL2Java: A Java code generator for OWL. Website, 2014. `http://www.incunabulum.de/projects/it/owl2java`.

[PFKP05] D.J. Patterson, D. Fox, H. Kautz, and M. Philipose. Fine-grained activity recognition by aggregating abstract object usage. In *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on*, pages 44–51, Oct 2005.

[PG11] F. Paganelli and D. Giuli. An ontology-based system for context-aware and configurable services to support home-based continuous care. *Information Technology in Biomedicine, IEEE Transactions on*, 15(2):324–333, March 2011.

[PGM+08] David Peterson, Shudi Gao, Ashok Malhotra, C. Michael Sperberg-McQueen, and Henry S. Thompson. W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes. World Wide Web Consortium, Working Draft WD-xmlschema11-2-20080620, June 2008.

[PKPS02] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic matching of web services capabilities. In *Proceedings of the First International Semantic Web Conference on The Semantic Web*, ISWC '02, pages 333–347, London, UK, 2002. Springer.

[PLFK03] DonaldJ. Patterson, Lin Liao, Dieter Fox, and Henry Kautz. Inferring high-level behavior from low-level sensors. In Anind K. Dey, Albrecht Schmidt, and Joseph F. McCarthy, editors, *UbiComp 2003: Ubiquitous Computing*, volume 2864 of *Lecture Notes in Computer Science*, pages 73–89. Springer Berlin Heidelberg, 2003.

[Pog09] A. Poggi. Developing ontology based applications with O3L. *WSEAS Transactions on Computers*, 8:1286–1295, August 2009.

[Pop10] Ronald Poppe. A survey on vision-based human action recognition. *Image and Vision Computing*, 28(6):976 – 990, 2010.

[PPFP04] Mike Perkowitz, Matthai Philipose, Kenneth Fishkin, and Donald J. Patterson. Mining models of human activities from the web. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pages 573–582, New York, NY, USA, 2004. ACM.

[PRK+07] Shwetak N. Patel, Thomas Robertson, Julie A. Kientz, Matthew S. Reynolds, and Gregory D. Abowd. At the flick of a switch: Detecting and classifying unique electrical events on the residential power line. In *Proceedings of the 9th International Conference on Ubiquitous Computing*, UbiComp '07, pages 271–288, Berlin, Heidelberg, 2007. Springer-Verlag.

[PUS08] Andrea Pugliese, Octavian Udrea, and V. S. Subrahmanian. Scaling RDF with Time. In *Proceeding of the 17th international conference*

on World Wide Web (WWW '08), pages 605–614, Beijing, China, 2008. ACM.

[PVdBW+04] Davy Preuveneers, Jan Van den Bergh, Dennis Wagelaar, Andy Georges, Peter Rigole, Tim Clerckx, Yolande Berbers, Karin Coninx, Viviane Jonckers, and Koen De Bosschere. Towards an extensible context ontology for ambient intelligence. *Journal of Ambient Intelligence*, 3295:148–159, 2004.

[PvdPS10] M. Pijl, S. van de Par, and Caifeng Shan. An event-based approach to multi-modal activity modeling and recognition. In *Pervasive Computing and Communications (PerCom), 2010 IEEE International Conference on*, pages 98–106, March 2010.

[RAMC04] A. Ranganathan, J. Al-Muhtadi, and R. H. Campbell. Reasoning about uncertain contexts in pervasive computing environments. *IEEE Pervasive Computing*, 3(2):62–70, April 2004.

[RB11a] Daniele Riboni and Claudio Bettini. Cosar: hybrid reasoning for context-aware activity recognition. *Personal and Ubiquitous Computing*, 15(3):271–289, 2011.

[RB11b] Daniele Riboni and Claudio Bettini. OWL 2 modeling and reasoning with complex human activities. *Pervasive and Mobile Computing*, 7(3):379–395, 2011.

[RC04] Gaëtan Rey and Joëlle Coutaz. The contextor infrastructure for context-aware computing. In *18th European Conference on Object-Oriented Programming (ECOOP 04), Workshop on Component-oriented approach to context-aware systems*, 2004.

[RC11] Parisa Rashidi and Diane J. Cook. Activity knowledge transfer in smart environments. *Pervasive and Mobile Computing*, 7(3):331–343, June 2011.

[RCHSE11] P. Rashidi, D.J. Cook, L.B. Holder, and Maureen Schmitter-Edgecombe. Discovering activities to recognize and track in a smart environment. *IEEE Transactions on Knowledge and Data Engineering*, 23(4):527–539, April 2011.

[RDM+11] Alberto Rosi, Simon Dobson, Marco Mamei, Graeme Stevenson, Juan Ye, and Franco Zambonelli. Social sensors and pervasive services: Approaches and perspectives. In *PerCol2011: Proceedings of the Second*

International Workshop on Pervasive Collaboration and Social Networking, March 2011.

[RDML05] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L. Littman. Activity recognition from accelerometer data. In *Proceedings of the 17th Conference on Innovative Applications of Artificial Intelligence - Volume 3*, IAAI'05, pages 1541–1546. AAAI Press, 2005.

[RGD10] Nirmalya Roy, Tao Gu, and Sajal K. Das. Supporting pervasive computing applications with active context fusion and semantic context delivery. *Pervasive and mobile computing*, 6(1):21 – 42, 2010.

[RHC+02] Manuel Román, Christopher Hess, Renato Cerqueira, Anand Ranganathan, Roy H. Campbell, and Klara Nahrstedt. Gaia: a middleware platform for active spaces. *Mobile Computing and Communications Review*, 6(4):65–67, 2002.

[RKT05] D.J. Russomanno, C. Kothari, and O. Thomas. Sensor ontologies: From shallow to deep models. In *Proceedings of the 37th Southeastern Symposium on System Theory (SSST '05)*, pages 107–112, March 2005.

[RMCM03] Anand Ranganathan, Robert E McGrath, Roy H. Campbell, and M. Dennis Mickunas. Use of ontologies in a pervasive computing environment. *Knowledge Engineering Review*, 18(3):209–220, 2003.

[RSK+06] Ioanna Roussaki, Maria Strimpakou, Nikos Kalatzis, Miltos Anagnostou, and Carsten Pils. Hybrid context modeling: A location-based scheme using ontologies. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on pervasive computing and communications workshops*, page 2, Washington, DC, USA, 2006. IEEE Computer Society.

[SAT+99] Albrecht Schmidt, Kofi Asante Aidoo, Antti Takaluoma, Urpo Tuomela, Kristof Van Laerhoven, and Walter Van de Velde. Advanced interaction in context. In *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, HUC '99, pages 89–101, London, UK, UK, 1999. Springer-Verlag.

[SAW94] Bill Schilit, Norman Adams, and Roy Want. Context-Aware Computing Applications. In *IEEE Workshop on Mobile Computing Systems and Applications*, pages 85–90, Santa Cruz, CA, 1994.

[SCSE09] Geetika Singla, Diane J Cook, and Maureen Schmitter-Edgecombe. Tracking activities in complex settings using smart environment technologies. *International journal of biosciences, psychiatry, and technology (IJBSPT)*, 1(1):25, 2009.

[SCY+13] Graeme Stevenson, Gabriella Castelli, Juan Ye, Alberto Rosi, Simon Dobson, and Franco Zambonelli. A bio-chemically inspired approach to awareness in pervasive systems. In *First International Workshop on Sensing and Big Data Mining*, SenseMine '13, November 2013.

[SD06] York Sure and John Domingue, editors. *ESWC '06: Proceedings of the 3rd European Semantic Web Conference on the semantic web: research and applications*, volume 4011 of *Lecture Notes in Computer Science*, Budva, Montenegro, 2006. Springer.

[SD11] Graeme Stevenson and Simon Dobson. Sapphire: Generating Java runtime artefacts from OWL ontologies. In *Proceedings of the Third International Workshop on Ontology-Driven Information Systems Engineering*, June 2011.

[SF02] Kari Sentz and Scott Ferson. Combination of evidence in Dempster-Shafer theory. Technical Report SAND 2002-0835, Thomas J. Watson School of Engineering and Applied Science, 2002.

[SFMM+] Graeme Stevenson, Jose Luis Fernandez-Marquez, Sara Montagna, Alberto Rosi, Juan Ye, Giovanna Di Marzo Serugendo, Mirko Viroli, Simon Dobson, and Akla Esso Tchao. Towards situated awareness in urban networks: A bio-inspired approach. In José Luis Fernandez-Marquez, Sara Montagna, Andrea Omicini, and Franco Zambonelli, editors, *1st International Workshop on Adaptive Service Ecosystems: Natural and Socially Inspired Solutions (ASENSIS 2012)*, SASO 2012, Lyon, France, September.

[SH09] Andy Seaborne and Steve Harris. SPARQL 1.1 query. W3C working draft, W3C, October 2009. http://www.w3.org/TR/2009/WD-sparql11-query-20091022/.

[Sha76] Glenn Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[SHS08] Amit Sheth, Cory Henson, and Satya S. Sahoo. Semantic sensor web. *Internet Computing*, 12(4), July/August 2008.

[SKDN09] Graeme Stevenson, Stephen Knox, Simon Dobson, and Paddy Nixon. Ontonym: a collection of upper ontologies for developing pervasive systems. In *CIAO '09: Proceedings of the 1st Workshop on Context, Information and Ontologies*, pages 1–8. ACM, 2009.

[SLP04] T. Strang and C. Linnhoff-Popien. A context modeling survey. In *Proceedings of the Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing*, Nottingham, UK, September 2004.

[som14] Sommer: Semantic object (medata) mapper. Website, 2014. `http://sommer.dev.java.net/`.

[spa14] Sparta: a simple API for RDF. Website, 2014. `https://github.com/mnot/sparta/`.

[SPG+07] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, 5:51–53, June 2007.

[spi14] Spira: A linked data ORM for Ruby. Website, 2014. `https://github.com/datagraph/spira`.

[SPJB07] Matthias C. Sala, Kurt Partridge, Linda Jacobson, and James Begole. An exploration into activity-informed physical advertising using pest. In Anthony LaMarca, Marc Langheinrich, and Khai N. Truong, editors, *Pervasive Computing*, volume 4480 of *Lecture Notes in Computer Science*, pages 73–90. Springer Berlin Heidelberg, 2007.

[SPM+13] Graeme Stevenson, Danilo Pianini, Sara Montagna, Mirko Viroli, Juan Ye, and Simon Dobson. Combining self-organisation, context-awareness and semantic reasoning: the case of resource discovery in opportunistic networks. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, Coimbra, Portugal, March 2013.

[SR06] Amarnag Subramanya and Alvin Raj. Recognizing activities and spatial context using wearable sensors. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI*, 2006.

[SRO+08] Thomas Stiefmeier, Daniel Roggen, Georg Ogris, Paul Lukowicz, and Gerhard Tröster. Wearable activity tracking in car manufacturing. *IEEE Pervasive Computing*, 7(2):42–50, April 2008.

[SS09] Maja Stikic and Bernt Schiele. Activity recognition from sparsely labeled data using multi-instance learning. In Tanzeem Choudhury, Aaron Quigley, Thomas Strang, and Koji Suginuma, editors, *Location and Context Awareness*, volume 5561 of *Lecture Notes in Computer Science*, pages 156–173. Springer Berlin Heidelberg, 2009.

[SSDN09] Matthew Stabeler, Graeme Stevenson, Simon Dobson, and Paddy Nixon. Basadaeir: harvesting user profiles to bootstrap pervasive applications. In *Proceedings of the 7th International Conference on Pervasive Computing, Pervasive 2009.*, Nara, Japan, May 2009.

[SSS07] D. Skoutas, A. Simitsis, and T. K. Sellis. A ranking mechanism for Semantic Web service discovery. In *IEEE SCW*, pages 41–48, 2007.

[ST94] Bill Schilit and Marvin Theimer. Disseminating Active Map Information to Mobile Hosts. *IEEE Network*, 8(5):22–32, September 1994.

[Ste97] Stephen V. Stehman. Selecting and interpreting measures of thematic classification accuracy. *Remote Sensing of Environment*, 62(1):77 – 89, 1997.

[Str87] Thomas M. Strat. The generation of explanations within evidential reasoning systems. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'87, pages 1097–1104, San Francisco, CA, USA, 1987. Morgan Kaufmann Publishers Inc.

[SVL01] A. Schmidt and K. Van Laerhoven. How to build smart appliances? *Personal Communications, IEEE*, 8(4):66–71, Aug 2001.

[SVY+] Graeme Stevenson, Mirko Viroli, Juan Ye, Sara Montagna, and Simon Dobson. Self-organising semantic resource discovery for pervasive systems. In José Luis Fernandez-Marquez, Sara Montagna, Andrea Omicini, and Franco Zambonelli, editors, *1st International Workshop on Adaptive Service Ecosystems: Natural and Socially Inspired Solutions (ASENSIS 2012)*, SASO 2012, Lyon, France, September.

[SYD10] Graeme Stevenson, Juan Ye, and Simon Dobson. On the impact of the temporal features of sensed data on the development of pervasive systems. In *PMMPS: Proceedings of the International Workshop on Programming Methods for Mobile and Pervasive Systems*, May 2010.

[SYD+13] Graeme Stevenson, Juan Ye, Simon Dobson, Mirko Viroli, Jose Luis Fernandez-Marquez, and Franco Zambonelli. Towards dynamic and decentralised situation awareness in SAPERE. *Newsletter of the Awareness Proactive Initiative*, 8:8–9, Spring 2013.

[SYDN10] Graeme Stevenson, Juan Ye, Simon Dobson, and Paddy Nixon. LOC8: A location model and extensible framework for programming with location. *IEEE Pervasive Computing*, 9(1):28 – 37, January 2010.

[SZC11] Saguna, A. Zaslavsky, and D. Chakraborty. Recognizing concurrent and interleaved activities in social interactions. In *IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing (DASC)*, pages 230–237, Dec 2011.

[TB09] Jonas Tappolet and Abraham Bernstein. Applied Temporal RDF: Efficient Temporal Querying of RDF Data with SPARQL. In *Proceedings of the 6th European Semantic Web Conference on The Semantic Web*, pages 308–322, Heraklion, Crete, Greece, 2009. Springer-Verlag.

[TCSU08] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *IEEE Trans. Cir. and Sys. for Video Technol.*, 18(11):1473–1488, November 2008.

[TH06] Bruce Tate and Curt Hibbs. *Ruby on Rails: Up and Running*. O'Reilly Media, Inc., 2006.

[TIL04] Emmanuel Munguia Tapia, Stephen S. Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In Alois Ferscha and Friedemann Mattern, editors, *Pervasive Computing*, volume 3001 of *Lecture Notes in Computer Science*, pages 158–175. Springer Berlin Heidelberg, 2004.

[TLL05a] Binh An Truong, Young-Koo Lee, and Sung-Young Lee. Modeling and reasoning about uncertainty in context-aware systems. In *IEEE International Conference on e-Business Engineering*, pages 102–109, Oct 2005.

[TLL05b] Binh An Truong, Young-Koo Lee, and Sung-Young Lee. Modeling uncertainty in context-aware computing. In *Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science*, ICIS '05, pages 676–681, Washington, DC, USA, 2005. IEEE Computer Society.

[TLL05c] Binh An Truong, Young-Koo Lee, and Sung-Young Lee. A unified context model: Bringing probabilistic models to context ontology. In Tomoya Enokido, Lu Yan, Bin Xiao, Daeyoung Kim, Yuanshun Dai, and LaurenceT. Yang, editors, *Embedded and Ubiquitous Computing at EUC 2005 Workshops*, volume 3823 of *Lecture Notes in Computer Science*, pages 566–575. Springer Berlin Heidelberg, 2005.

[TMKL06] Alessandra Toninelli, Rebecca Montanari, Lalana Kagal, and Ora Lassila. A semantic context-aware access control framework for secure collaborations in pervasive computing environments. In *Proceedings of the 5th international conference on The Semantic Web*, ISWC'06, pages 473–486, Athens, GA, 2006. Springer-Verlag.

[top] TopBraid Composer. `http://www.topquadrant.com/products/TB_Composer.html`.

[tra14] TRAMP: Makes RDF look like Python data structures. Website, 2014. `http://www.aaronsw.com/2002/tramp/`.

[TSTN06] Graham Thomson, Graeme Stevenson, Sotirios Terzis, and Paddy Nixon. A self-managing infrastructure for ad-hoc situation determination. In *Proceedings of the 1st International Conference On Smart homes & heath Telematics (ICOST'2006) "Smart Homes and Beyond"*, pages 157–164, Belfast, UK, June 2006. IOS Press.

[VA13] Sarvesh Vishwakarma and Anupam Agrawal. A survey on activity recognition and behavior understanding in video surveillance. *The Visual Computer*, 29(10):983–1009, 2013.

[VBRM+07] M. Vanden Bossche, P. Ross, I. MacLarty, B. Van Nuffelen, and N. Pelov. Ontology driven software engineering for real life applications. In *Proceedings of the 3rd International Workshop on Semantic Web Enabled Software Engineering*, 2007.

[vKEK10a] T van Kasteren, G Englebienne, and B Kröse. Transferring knowledge of activity recognition across sensor networks. In *Pervasive Computing*, page 283Ð300. Springer, 2010.

[vKEK10b] T. L. M. van Kasteren, G. Englebienne, and B. J. A. Kröse. Activity recognition using semi-markov models on real world smart home datasets. *J. Ambient Intell. Smart Environ.*, 2(3):311–325, August 2010.

[vKK07] T. van Kasteren and B. Kröse. Bayesian activity recognition in residence for elders. In *3rd IET International Conference on Intelligent Environments*, pages 209–212, Sept 2007.

[vKNEK08] Tim van Kasteren, Athanasios Noulas, Gwenn Englebienne, and Ben Kröse. Accurate activity recognition in a home setting. In *Proceedings of the 10th international conference on Ubiquitous computing*, UbiComp '08, pages 1–9, New York, NY, USA, 2008. ACM.

[VLKS08] Kristof Van Laerhoven, David Kilian, and Bernt Schiele. Using rhythm awareness in long-term activity recognition. In *Proceedings of the 2008 12th IEEE International Symposium on Wearable Computers*, ISWC '08, pages 63–66, Washington, DC, USA, 2008. IEEE Computer Society.

[VM03] Christian Vogler and Dimitris N. Metaxas. Handshapes and movements: Multiple-channel american sign language recognition. In *Gesture Workshop*, pages 247–258, 2003.

[Vok06] M. Vokel. RDFReactor – from ontologies to programmatic data access. In *The Jena Developer Conference*, Bristol, UK, 2006.

[VV04] Achille Varzi and Laure Vieu. Formal ontology in information systems. In *FOIS '04: Proceedings of the third international conference on formal ontology in information systems*, Turin, Italy, November 2004. IOS Press.

[VVL07] Douglas L. Vail, Manuela M. Veloso, and John D. Lafferty. Conditional random fields for activity recognition. In *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS '07, pages 235:1–235:8, New York, NY, USA, 2007. ACM.

[VZSD12] Mirko Viroli, Franco Zambonelli, Graeme Stevenson, and Simon Dobson. From SOA to pervasive service ecosystems: An approach based on Semantic Web technologies. In Javier Cubo and Guadalupe Ortiz, editors, *Adaptive Web Services for Modular and Reusable Software Development: Tactics and Solution*, chapter 8, pages 207–237. IGI Global, 2012.

[w3c] Review of sensor and observation ontologies. `http://www.w3.org/2005/Incubator/ssn/wiki/Review_of_Sensor_and_Observations_Ontologies`.

[w3c13a] W3c provenance working group. Website, 2013. http://www.w3.org/2011/prov/.

[w3c13b] W3c semantic sensor network incubator group. Website, 2013. http://www.w3.org/2005/Incubator/ssn.

[Wal00] Jim Waldo. *The Jini Specifications*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 2000.

[WAS+12] Tracy L. Westeyn, Gregory D. Abowd, Thad E. Starner, Jeremy M. Johnson, Peter W. Presti, and Kimberly A. Weaver. Monitoring children's developmental progress using augmented toys and activity recognition. *Personal and Ubiquitous Computing*, 16(2):169–191, February 2012.

[Wei95] Mark Weiser. The computer for the 21st century. *Human-computer interaction: toward the year 2000*, pages 933–940, 1995.

[WF05] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[WGT+11] Liang Wang, Tao Gu, Xianping Tao, Hanhua Chen, and Jian Lu. Recognizing multi-user activities using wearable sensors in a smart home. *Pervasive and Mobile Computing*, 7(3):287–298, June 2011.

[WLYjH07] Tsu-yu Wu, Chia-chun Lian, and Jane Yung-jen Hsu. Joint recognition of multiple concurrent activities using factorial conditional random fields. In *2007 AAAI Workshop on Plan, Activity, and Intent Recognition, Technical Report WS-07-09. The AAAI Press, Menlo Park*, 2007.

[WNKL13] Gary M. Weiss, Ashwin Nathan, J.B. Kropp, and Jeffrey W. Lockhart. Wagtag: A dog collar accessory for monitoring canine activity levels. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, UbiComp '13 Adjunct, pages 405–414, New York, NY, USA, 2013. ACM.

[WNS06] Christian Wojek, Kai Nickel, and Rainer Stiefelhagen. Activity recognition and room-level tracking in an office environment. In *Multisensor Fusion and Integration for Intelligent Systems, 2006 IEEE International Conference on*, pages 25–30. IEEE, 2006.

[WRB11] Daniel Weinland, Remi Ronfard, and Edmond Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding*, 115(2):224 – 241, 2011.

[WSSY02] Huadong Wu, Mel Siegel, Rainer Stiefelhagen, and Jie Yang. Sensor fusion using Dempster-Shafer theory. In *Proceedings of the IEEE Instrumentation and Measurement Technology Conference*, volume 1, pages 7–12, Anchorage, AK, USA, May 2002.

[YCDN07] Juan Ye, Lorcan Coyle, Simon Dobson, and Paddy Nixon. Ontology-based models in pervasive computing systems. *Knowledge engineering review*, 22:315–347, December 2007.

[YD10] Juan Ye and Simon Dobson. Exploring semantics in activity recognition using context lattices. *Journal of Ambient Intelligence in Smart Environments.*, 2(4):389–407, December 2010.

[YDM12] Juan Ye, Simon Dobson, and Susan McKeever. Situation identification techniques in pervasive computing: A review. *Pervasive Mob. Comput.*, 8(1):36–66, February 2012.

[Ye09] Juan Ye. *Exploiting Semantics with Situation Lattices in Pervasive Computing*. Ph.D. thesis, University College Dublin, 2009.

[YHGY06] Stephen S. Yau, Dazhi Huang, Haishan Gong, and Yisheng Yao. Support for situation awareness in trustworthy ubiquitous computing application software. *Software: practice and experience*, 36(9):893–921, 2006.

[YMC+08] Juan Ye, Susan McKeever, Lorcan Coyle, Steve Neely, and Simon Dobson. Resolving uncertainty in context integration and abstraction. In *ICPS 2008: Proceedings of the international conference on pervasive services*, pages 131–140, New York, NY, USA, July 2008. ACM.

[YS13] Juan Ye and Graeme Stevenson. Semantics-driven multi-user concurrent activity recognition. In Juan Carlos Augusto, Reiner Wichert, Rem Collier, David Keyson, Albert Ali Salah, and Ah-Hwee Tan, editors, *Ambient Intelligence*, volume 8309 of *Lecture Notes in Computer Science*, pages 204–219. Springer International Publishing, 2013.

[YSD11a] Juan Ye, Graeme Stevenson, and Simon Dobson. A top-level ontology for smart environments. *Pervasive and Mobile Computing*, 7(3):359 – 378, 2011. Knowledge-Driven Activity Recognition in Intelligent Environments.

[YSD+11b] Juan Ye, Graeme Stevenson, Simon Dobson, Michael O'Grady, and Gregory O'Hare. PI: perceiver and interpreter of smart home datasets. In *Proceedings of the 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth 2011)*, May 2011.

[YSD+12] Juan Ye, Graeme Stevenson, Simon Dobson, Michael O' Grady, and Gregory O' Hare. Perceiving and interpreting smart home datasets with PI. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–13, 2012.

[YSD14a] Juan Ye, Graeme Stevenson, and Simon Dobson. USMART: an unsupervised semantic mining activity recognition technique. *ACM Transactions on Interactive and Intelligent Systems*, (Accepted) 2014.

[YSD14b] Juan Ye, Graeme Stevenson, and Simon Dobson. KCAR: A knowledge-driven approach for concurrent activity recognition. *Pervasive and Mobile Computing*, (In Press) 2014.

[YT12] Koji Yatani and Khai N. Truong. Bodyscope: A wearable acoustic sensor for activity recognition. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 341–350, New York, NY, USA, 2012. ACM.

[Yu10] Shun-Zheng Yu. Hidden semi-markov models. *Artificial Intelligence*, 174(2):215 – 243, 2010. Special Review Issue.

[YWC08] Jhun-Ying Yang, Jeen-Shing Wang, and Yen-Ping Chen. Using acceleration measurements for activity recognition: An effective learning algorithm for constructing neural classifiers. *Pattern Recognition Letters*, 29(16):2213 – 2220, 2008.

[Zad73] Lotfi A Zadeh. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions On Systems Man And Cybernetics*, 3(1):28–44, 1973.

[Zad86] Lotfi A Zadeh. A simple view of the Dempster-Shafer theory of evidence and its implication for the rule of combination. *AI Magazine*, 7(2):85–90, July 1986.

[Zai13] Jennifer Zaino. W3C's semantic web activity folds into new data activity, 2013.

[ZCZG09] Daqiang Zhang, Jiannong Cao, Jingyu Zhou, and Minyi Guo. Extended Dempster-Shafer theory in context reasoning for ubiquitous computing environments. In *CSE '09: Proceedings of the 2009 International Conference on Computational Science and Engineering*, pages 205–212, Washington, DC, USA, 2009. IEEE Computer Society.

[Zha07] Jing Hua Zhao. GAP: Genetic analysis package. *Journal of Statistical Software*, 23(8):1–18, 12 2007.

[Zha12] Zhengyou Zhang. Microsoft kinect sensor and its effect. *IEEE MultiMedia*, 19(2):4–10, Feb 2012.

[ZHK10] Weishan Zhang, Klaus Marius Hansen, and Thomas Kunz. Enhancing intelligence and dependability of a product line enabled pervasive middleware. *Pervasive and Mobile Computing*, 6(2):198–217, 2010.

[ZHY09] Vincent Wenchen Zheng, Derek Hao Hu, and Qiang Yang. Cross-domain activity recognition. In *Proceedings of the 11th International Conference on Ubiquitous Computing*, Ubicomp '09, pages 61–70, New York, NY, USA, 2009. ACM.

APPENDIX

A

RELATIONSHIPS BETWEEN CONTEXT STATEMENTS

Proofs supporting the inference of relationships between context statements from the relationships in their corresponding dimensions of information, introduced in Section 3.4.2

Lemma 4 (Inferring statement equality from concepts)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$, where s_i, $s_j \in S^a$, o_i, $o_j \in O^a$, $s_i \not\trianglelefteq s_j$, $o_i \not\trianglelefteq o_j$, $s_j \not\trianglelefteq s_i$, and $o_j \not\trianglelefteq o_i$, $[s_i, p, o_i] = [s_j, p, o_j]$, if $s_i = s_j$ and $o_i = o_j$.

Proof. From Definition 8 $\forall (s, o) \in S^a \times O^a$, if (s, o) entails $[s_i, p, o_i]$ and $[s_j, p, o_j]$ then
$s \trianglelefteq s_i$ (1).
$o \trianglelefteq o_i$ (2).
$s \trianglelefteq s_j$ (3).
$o \trianglelefteq o_j$ (4).
If $s_i = s_j$, $o_i = o_j$, $s_i \not\trianglelefteq s_j$, $o_i \not\trianglelefteq o_j$, $s_j \not\trianglelefteq s_i$, and $o_j \not\trianglelefteq o_i$ then from (1), (2), (3), (4), and Definition 3, $s = s_i = s_j$ and $o = o_i = o_j$ (5).
From (5) and Definition 8, (s, o) entails $[s_i, p, o_i]$ and $[s_j, p, o_j]$ (6).
Thus, by (6) and Definition 10, $[s_i, p, o_i] = [s_j, p, o_j]$. □

Lemma 5 (Inferring statement subsumption from concepts)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$, where s_i, $s_j \in S^a$ and o_i, $o_j \in O^a$, $[s_i, p, o_i] \trianglelefteq [s_j, p, o_j]$, if $s_i \trianglelefteq s_j$ and $o_i \trianglelefteq o_j$.

Proof. From Definition 8 $\forall (s, o) \in S^a \times O^a$, if (s, o) entails $[s_i, p, o_i]$ then $s \trianglelefteq s_i$ (1).
$o \trianglelefteq o_i$ (2).
If $s_i \trianglelefteq s_j$ and $o_i \trianglelefteq o_j$, then from (1), (2), and Definition 4, $s \trianglelefteq s_j$ and $o \trianglelefteq o_j$ (3).
From (3) and Definition 8, (s, o) entails $[s_j, p, o_j]$ (4).
Thus, by (4) and Definition 10, $[s_i, p, o_i] \trianglelefteq [s_j, p, o_j]$. □

Lemma 6 (Inferring statement overlap from concepts)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$, where s_i, $s_j \in S^a$ and o_i, $o_j \in O^a$, $[s_i, p, o_i] \bowtie [s_j, p, o_j]$, if $s_i = s_j$ and $o_i \bowtie o_j$, or $o_i = o_j$ and $s_i \bowtie s_j$, or both.

Proof. Given that $s_i = s_j$ and $o_i \bowtie o_j$, there exists a concept $o_k \in O^a$ such that $\mu(o_k) = \mu(o_i) \cap \mu(o_j) \neq \varnothing$ (1).
From (1), $o_k \trianglelefteq o_i$ and $o_k \trianglelefteq o_j$ (2).
By (2) and Definition 8, (s_i, o_k) entails both $[s_i, p, o_i]$ and $[s_j, p, o_j]$ (3).
Also by Definition 2, $o_i \bowtie o_j$ implies that there exists o_m and $o_n \in O^a$ such that $\mu(o_m) = \mu(o_i) \setminus \mu(o_j) \neq \varnothing$ and $\mu(o_n) = \mu(o_j) \setminus \mu(o_i) \neq \varnothing$ (4).
By (4), $\forall o_{m'} \trianglelefteq o_m$, $(s_i, o_{m'})$ are the pairs that entail $[s_i, p, o_i]$ but not $[s_j, p, o_j]$.
Similarly, $\forall o_{n'} \trianglelefteq o_n$, $(s_j, o_{n'})$ are the pairs that entail $[s_j, p, o_j]$ but not $[s_i, p, o_i]$ (5).
Thus, by (5) and from Definition 10, $[s_i, p, o_i] \bowtie [s_j, p, o_j]$.
Showing that two context statements overlap when $o_i = o_j$ and $s_i \bowtie s_j$ is proved similarly. □

Lemma 7 (Inferring statement adjacency from concepts)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$, where s_i, $s_j \in S^a$ and o_i, $o_j \in O^a$, $[s_i, p, o_i] \parallel [s_j, p, o_j]$, if $s_i \parallel s_j$ or $o_i \parallel o_j$.

Proof. Given that $s_i = s_j$ and $o_i \parallel o_j$, from Definition 2 $\mu(o_i) \cap \mu(o_k)$ and $\exists k \in \mu(o_i)$ such that $k + \varepsilon \in \mu(o_j)$, where ε is the smallest unit value increment (1).
Let $\mu(o_x) = k$ and let $\mu(o_y) = k + \varepsilon$ (2).
From (2), $o_x \parallel o_y$ (3).
From (1), (2) and Definition 8, $o_x \trianglelefteq o_i$ and $o_y \trianglelefteq o_j$ (4).
From (4) and Lemma 5, (s, o_x) entails $[s_i, p, o_i]$, but not $[s_j, p, o_j]$ (5).
From (4) and Lemma 5, (s, o_y) entails $[s_j, p, o_j]$, but not $[s_i, p, o_i]$ (6).
Thus, from (3), (5), (6) and by Definition 10, $[s_i, p, o_i] \parallel [s_j, p, o_j]$.
Showing that two context statements are adjacent when $o_i = o_j$ and $s_i \parallel s_j$ is proved similarly. □

Lemma 8 (Inferring statement disjointness from concepts)**.**

Given two context statements $[s_i, p, o_i]$ and $[s_j, p, o_j]$, where s_i, $s_j \in S^a$ and o_i, $o_j \in O^a$, $[s_i, p, o_i] \boxtimes [s_j, p, o_j]$, if $s_i = s_j$ and $o_i \boxtimes o_j$, or $o_i = o_j$ and $s_i \boxtimes s_j$.

Proof. From Definition 8 $\forall (s, o) \in S^a \times O^a$, if (s, o) entails $[s_i, p, o_i]$ then $s \trianglelefteq s_i$ (1).
$o \trianglelefteq o_i$ (2).
If $s_i = s_j$ and $o_i \boxtimes o_j$, (1), (2) and Lemma 1, give $o \boxtimes o_j$ (3).
Given $s_i = s_j$, by (3) (s_i, o) cannot validate $[s_j, p, o_j]$; any pair of values that validates $[s_i, p, o_i]$ cannot validate $[s_j, p, o_j]$. (4).
Thus, by (4) and from Definition 10, $[s_i, p, o_i] \boxtimes [s_j, p, o_j]$.
Showing that two context statements conflict when $o_i = o_j$ and $s_i \boxtimes s_j$ is proved similarly. □

APPENDIX

B

A CASL J48 TREE

```
COMPUTER_INTERACTION = 0
|   MEETING_SCHEDULED = 0
|   |   TIME <= 1210935720000
|   |   |   UBISENSE_Z <= 2.59
|   |   |   |   TIME <= 1210930560000
|   |   |   |   |   UBISENSE_X <= 18.3
|   |   |   |   |   |   UBISENSE_X <= 10.74
|   |   |   |   |   |   |   TIME <= 1210928580000: BREAK
|   |   |   |   |   |   |   TIME > 1210928580000
|   |   |   |   |   |   |   |   TIME <= 1210930440000: MEETING
|   |   |   |   |   |   |   |   TIME > 1210930440000: COFFEE
|   |   |   |   |   |   UBISENSE_X > 10.74: COFFEE
|   |   |   |   |   UBISENSE_X > 18.3: BREAK
|   |   |   |   TIME > 1210930560000
|   |   |   |   |   UBISENSE_X <= 16.2
|   |   |   |   |   |   TIME <= 1210932840000: READING
|   |   |   |   |   |   TIME > 1210932840000
|   |   |   |   |   |   |   TIME <= 1210933920000
|   |   |   |   |   |   |   |   TIME <= 1210933320000: WORKING
|   |   |   |   |   |   |   |   TIME > 1210933320000: BREAK
|   |   |   |   |   |   |   TIME > 1210933920000: READING
|   |   |   |   |   UBISENSE_X > 16.2
|   |   |   |   |   |   UBISENSE_X <= 16.96: COFFEE
|   |   |   |   |   |   UBISENSE_X > 16.96: BREAK
|   |   |   UBISENSE_Z > 2.59: COFFEE
|   |   TIME > 1210935720000
|   |   |   TIME <= 1210942380000
|   |   |   |   TIME <= 1210936860000: BREAK
|   |   |   |   TIME > 1210936860000
|   |   |   |   |   UBISENSE_X <= 16.96
|   |   |   |   |   |   TIME <= 1210939740000: LUNCH
|   |   |   |   |   |   TIME > 1210939740000
|   |   |   |   |   |   |   TIME <= 1210940340000: BREAK
|   |   |   |   |   |   |   TIME > 1210940340000: LUNCH
|   |   |   |   |   UBISENSE_X > 16.96: BREAK
|   |   |   TIME > 1210942380000
|   |   |   |   TIME <= 1210948740000
|   |   |   |   |   TIME <= 1210943460000: WORKING
|   |   |   |   |   TIME > 1210943460000: BREAK
|   |   |   |   TIME > 1210948740000
|   |   |   |   |   TIME <= 1210949520000: WORKING
|   |   |   |   |   TIME > 1210949520000: BREAK
|   MEETING_SCHEDULED = 1
|   |   TIME <= 1210932660000: MEETING
|   |   TIME > 1210932660000: BREAK
COMPUTER_INTERACTION = 1: WORKING
```

APPENDIX

C

HOUSE A DECISION TREE MODELS

This appendix shows example decision trees generated for the standalone and hierarchical models using the House A dataset. Not only does the hierarchical model improve performance, but it can be seen that the overall size of the hierarchical classification scheme is smaller than the standalone model, and its modularity aids the intelligibility of the decision process.

Standalone Model (Size 87, Leaves 44)

```
LC_FRONT_DOOR = 0
|   LC_ANY_KITCHEN = 0
|   |   LC_BEDROOM_DOOR = 0
|   |   |   LC_ANY_TOILET = 0
|   |   |   |   TE_BEDROOM_DOOR <= 8: BED
|   |   |   |   TE_BEDROOM_DOOR > 8
|   |   |   |   |   NIGHT = 0
|   |   |   |   |   |   EVENING = 0: SHOWER
|   |   |   |   |   |   EVENING = 1: BED
|   |   |   |   |   NIGHT = 1
|   |   |   |   |   |   TE_TOILET_FLUSH <= 7: BED
|   |   |   |   |   |   TE_TOILET_FLUSH > 7: TOILET
|   |   |   LC_ANY_TOILET = 1
|   |   |   |   TE_ANY_TOILET <= 7
|   |   |   |   |   TE_FOOD_STORAGE <= 4
|   |   |   |   |   |   EVENING = 0: LEAVE
|   |   |   |   |   |   EVENING = 1: DINNER
|   |   |   |   |   TE_FOOD_STORAGE > 4
|   |   |   |   |   |   TE_BEDROOM_DOOR <= 9
|   |   |   |   |   |   |   MORNING = 0: BED
|   |   |   |   |   |   |   MORNING = 1
|   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 0
|   |   |   |   |   |   |   |   |   TE_BEDROOM_DOOR <= 2: BED
|   |   |   |   |   |   |   |   |   TE_BEDROOM_DOOR > 2: TOILET
|   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 1: BED
|   |   |   |   |   |   TE_BEDROOM_DOOR > 9
|   |   |   |   |   |   |   NIGHT = 0
|   |   |   |   |   |   |   |   MORNING = 0: TOILET
|   |   |   |   |   |   |   MORNING = 1
|   |   |   |   |   |   |   |   TE_ANY_TOILET <= 1
```

```
|   |   |   |   |   |   |   |   |   |   TE_PLATES <= 9: SHOWER
|   |   |   |   |   |   |   |   |   |   TE_PLATES > 9: TOILET
|   |   |   |   |   |   |   |   |   TE_ANY_TOILET > 1
|   |   |   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 0: TOILET
|   |   |   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 1
|   |   |   |   |   |   |   |   |   |   |   LC_TOILET_DOOR = 0: TOILET
|   |   |   |   |   |   |   |   |   |   |   LC_TOILET_DOOR = 1: SHOWER
|   |   |   |   |   |   |   NIGHT = 1
|   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 0: TOILET
|   |   |   |   |   |   |   |   LC_TOILET_FLUSH = 1
|   |   |   |   |   |   |   |   |   LC_BATHROOM_DOOR = 0: BED
|   |   |   |   |   |   |   |   |   LC_BATHROOM_DOOR = 1: TOILET
|   |   |   |   |   TE_ANY_TOILET > 7
|   |   |   |   |   |   LC_TOILET_FLUSH = 0: LEAVE
|   |   |   |   |   |   LC_TOILET_FLUSH = 1: BREAKFAST
|   |   |   LC_BEDROOM_DOOR = 1: BED
|   |   LC_ANY_KITCHEN = 1
|   |   |   MORNING = 0
|   |   |   |   NIGHT = 0
|   |   |   |   |   LC_CUPS_CUPBOARD = 0
|   |   |   |   |   |   TE_BATHROOM_DOOR <= 9
|   |   |   |   |   |   |   LC_FRIDGE = 0: DINNER
|   |   |   |   |   |   |   LC_FRIDGE = 1: DRINK
|   |   |   |   |   |   TE_BATHROOM_DOOR > 9: DINNER
|   |   |   |   |   LC_CUPS_CUPBOARD = 1
|   |   |   |   |   |   TE_ANY_KITCHEN <= 8
|   |   |   |   |   |   |   LC_FRIDGE = 0
|   |   |   |   |   |   |   |   TE_FRIDGE <= 1: DRINK
|   |   |   |   |   |   |   |   TE_FRIDGE > 1: DINNER
|   |   |   |   |   |   |   LC_FRIDGE = 1: DRINK
|   |   |   |   |   |   TE_ANY_KITCHEN > 8: TOILET
|   |   |   |   NIGHT = 1
|   |   |   |   |   TE_CUPS_CUPBOARD <= 5: DRINK
|   |   |   |   |   TE_CUPS_CUPBOARD > 5: TOILET
|   |   |   MORNING = 1
|   |   |   |   TE_ANY_KITCHEN <= 3
|   |   |   |   |   TE_BEDROOM_DOOR <= 2
|   |   |   |   |   |   TE_ANY_TOILET <= 5: TOILET
|   |   |   |   |   |   TE_ANY_TOILET > 5: BREAKFAST
|   |   |   |   |   TE_BEDROOM_DOOR > 2
|   |   |   |   |   |   LC_CUPS_CUPBOARD = 0: BREAKFAST
|   |   |   |   |   |   LC_CUPS_CUPBOARD = 1
|   |   |   |   |   |   |   TE_HEATING <= 6: BREAKFAST
|   |   |   |   |   |   |   TE_HEATING > 6: DRINK
|   |   |   |   TE_ANY_KITCHEN > 3
|   |   |   |   |   TE_FOOD_STORAGE <= 9: TOILET
|   |   |   |   |   TE_FOOD_STORAGE > 9
|   |   |   |   |   |   LC_CUPS_CUPBOARD = 0: SHOWER
|   |   |   |   |   |   LC_CUPS_CUPBOARD = 1: TOILET
LC_FRONT_DOOR = 1
|   LC_GROCERIES_CUPBOARD = 0
|   |   LC_BATHROOM_DOOR = 0: LEAVE
|   |   LC_BATHROOM_DOOR = 1
|   |   |   TE_ANY_TOILET <= 5: LEAVE
|   |   |   TE_ANY_TOILET > 5: DINNER
|   LC_GROCERIES_CUPBOARD = 1: DINNER
```

Hierarchy: Root (Size 45, Leaves, 23)

```
LC_FRONT_DOOR = 0
|   LC_ANY_KITCHEN = 0
|   |   LC_BEDROOM_DOOR = 0
|   |   |   TE_BEDROOM_DOOR <= 9
|   |   |   |   MORNING = 0: BED
|   |   |   |   MORNING = 1
|   |   |   |   |   LC_TOILET_FLUSH = 0
|   |   |   |   |   |   TE_BEDROOM_DOOR <= 2: BED
|   |   |   |   |   |   TE_BEDROOM_DOOR > 2: HYGIENE
|   |   |   |   |   LC_TOILET_FLUSH = 1: BED
|   |   |   TE_BEDROOM_DOOR > 9
|   |   |   |   NIGHT = 0
|   |   |   |   |   EVENING = 0
|   |   |   |   |   |   LC_TOILET_FLUSH = 0: HYGIENE
|   |   |   |   |   |   LC_TOILET_FLUSH = 1
|   |   |   |   |   |   |   TE_ANY_TOILET <= 7: HYGIENE
|   |   |   |   |   |   |   TE_ANY_TOILET > 7: KITCHEN
|   |   |   |   |   EVENING = 1
|   |   |   |   |   |   TE_GROCERIES_CUPBOARD <= 6: KITCHEN
|   |   |   |   |   |   TE_GROCERIES_CUPBOARD > 6
|   |   |   |   |   |   |   LC_ANY_TOILET = 0: BED
|   |   |   |   |   |   |   LC_ANY_TOILET = 1: HYGIENE
|   |   |   |   NIGHT = 1
|   |   |   |   |   TE_TOILET_FLUSH <= 7
|   |   |   |   |   |   LC_BATHROOM_DOOR = 0: BED
|   |   |   |   |   |   LC_BATHROOM_DOOR = 1
|   |   |   |   |   |   |   LC_ANY_TOILET = 0: BED
|   |   |   |   |   |   |   LC_ANY_TOILET = 1: HYGIENE
|   |   |   |   |   TE_TOILET_FLUSH > 7: HYGIENE
|   |   LC_BEDROOM_DOOR = 1: BED
|   LC_ANY_KITCHEN = 1
|   |   TE_ANY_KITCHEN <= 9: KITCHEN
|   |   TE_ANY_KITCHEN > 9
|   |   |   LC_FRIDGE = 0
|   |   |   |   LC_CUPS_CUPBOARD = 0: KITCHEN
|   |   |   |   LC_CUPS_CUPBOARD = 1: HYGIENE
|   |   |   LC_FRIDGE = 1: HYGIENE
LC_FRONT_DOOR = 1
|   LC_GROCERIES_CUPBOARD = 0
|   |   LC_BATHROOM_DOOR = 0: LEAVE
|   |   LC_BATHROOM_DOOR = 1
|   |   |   TE_ANY_TOILET <= 5: LEAVE
|   |   |   TE_ANY_TOILET > 5: KITCHEN
|   LC_GROCERIES_CUPBOARD = 1: KITCHEN
```

Hierarchy: Hygiene (Size 19, Leaves 10)

```
LC_ANY_TOILET = 0
|   EVENING = 0
|   |   NIGHT = 0
|   |   |   LC_CUPS_CUPBOARD = 0
|   |   |   |   LC_BEDROOM_DOOR = 0: SHOWER
|   |   |   |   LC_BEDROOM_DOOR = 1: TOILET
|   |   |   LC_CUPS_CUPBOARD = 1: TOILET
|   |   NIGHT = 1: TOILET
|   EVENING = 1: TOILET
LC_ANY_TOILET = 1
|   TE_ANY_TOILET <= 1: TOILET
|   TE_ANY_TOILET > 1
|   |   LC_TOILET_FLUSH = 0: TOILET
|   |   LC_TOILET_FLUSH = 1
|   |   |   MORNING = 0: TOILET
|   |   |   MORNING = 1
|   |   |   |   LC_TOILET_DOOR = 0: TOILET
|   |   |   |   LC_TOILET_DOOR = 1: SHOWER
```

Hierarchy: Eating/Drinking (Size: 19, Leaves 10)

```
MORNING = 0
|   EVENING = 0: DRINK
|   EVENING = 1
|   |   LC_CUPS_CUPBOARD = 0
|   |   |   TE_BATHROOM_DOOR <= 9
|   |   |   |   LC_FRIDGE = 0: DINNER
|   |   |   |   LC_FRIDGE = 1: DRINK
|   |   |   TE_BATHROOM_DOOR > 9: DINNER
|   |   LC_CUPS_CUPBOARD = 1
|   |   |   LC_FRIDGE = 0
|   |   |   |   TE_FRIDGE <= 1: DRINK
|   |   |   |   TE_FRIDGE > 1: DINNER
|   |   |   LC_FRIDGE = 1: DRINK
MORNING = 1
|   LC_CUPS_CUPBOARD = 0: BREAKFAST
|   LC_CUPS_CUPBOARD = 1
|   |   TE_HEATING <= 6: BREAKFAST
|   |   TE_HEATING > 6: DRINK
```

APPENDIX

D

ADDITIONAL DATASET RESULTS

This appendix gives the charts associated with the performance of the proposed techniques on the INK-12, House B, and House C datasets. The boxplots are interpreted as follows: The black line represents the median value, the box 'hinges' represent the the first and third quartile, the box whiskers extend to the most extreme data point which is no more than 1.5 times the interquartile range from the box, and further outliers are denoted by a unfilled circle. Mean values are denoted by a red triangle.

Ink-12 Results The results of applying the techniques developed in this thesis to the INK-12 dataset are shown in Figure D.1 and Figure D.2.

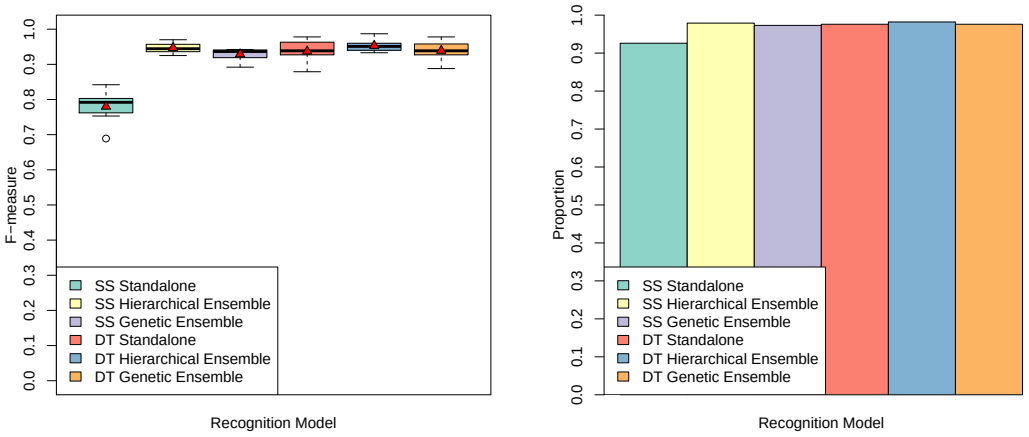


Figure D.1: A summary of the overall performance of selected classification approaches on the INK-12 dataset.

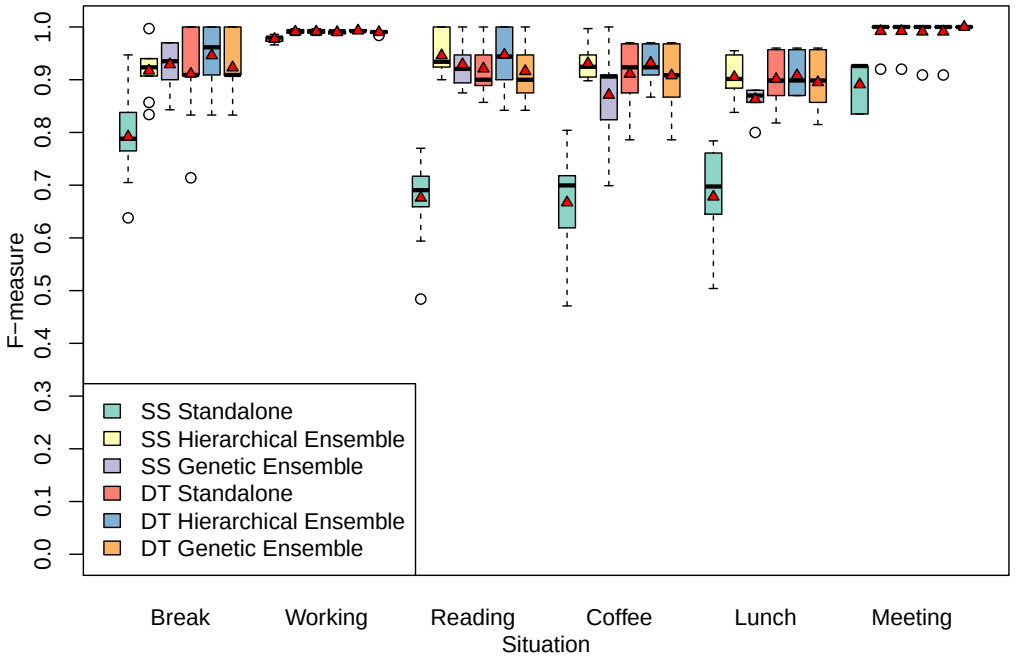


Figure D.2: A comparison of the f-measure performance of selected classification approaches on situations in the INK-12 dataset.

House B Results The results of applying the techniques developed in this thesis to the House B dataset are shown in Figure D.3 and Figure D.4.

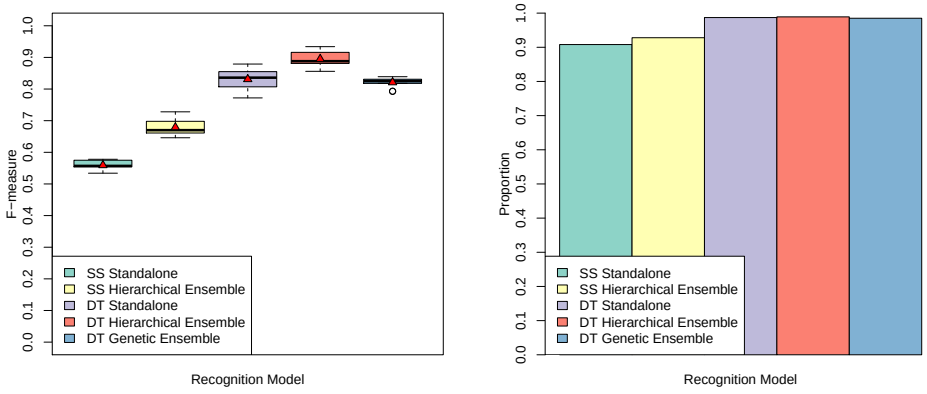


Figure D.3: A summary of the overall performance of selected classification approaches on the House B dataset.

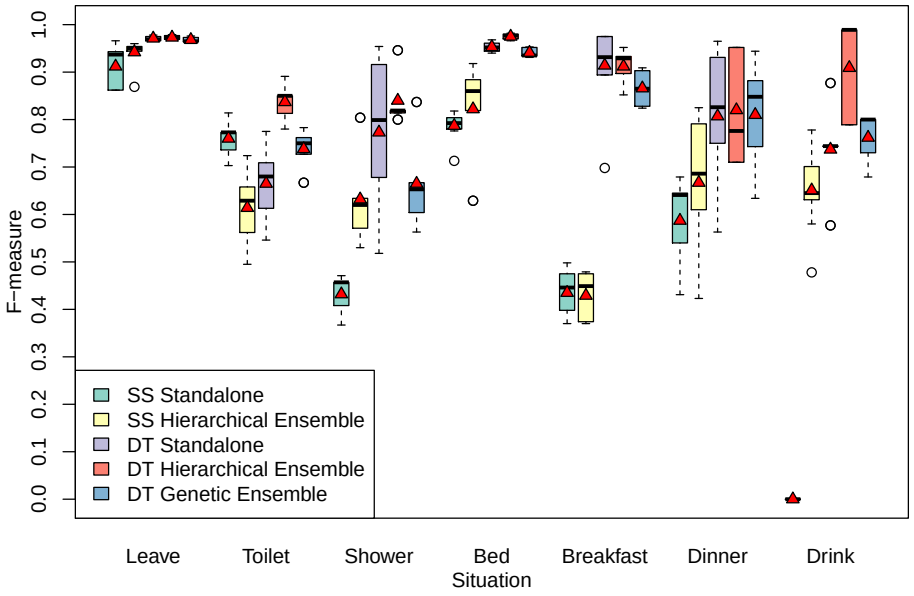


Figure D.4: A comparison of the f-measure performance of selected classification approaches on the House B dataset.

House C Results The results of applying the techniques developed in this thesis to the House C dataset are shown in Figure D.5 and Figure D.6.

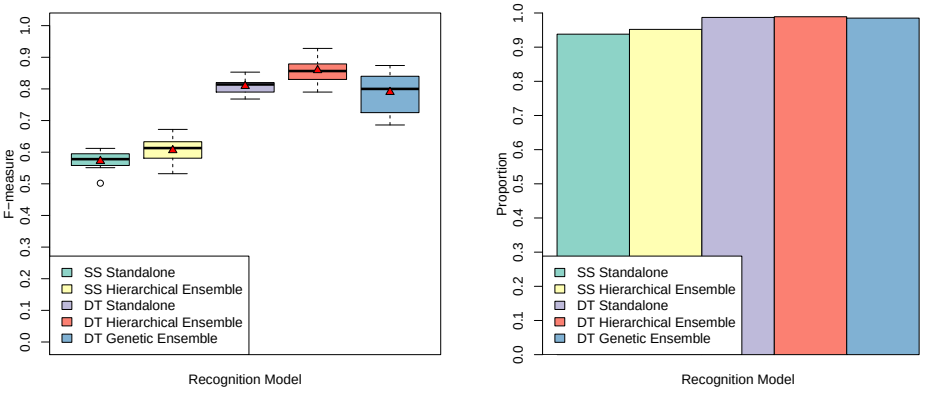


Figure D.5: A summary of the overall performance of selected classification approaches on the House C dataset.

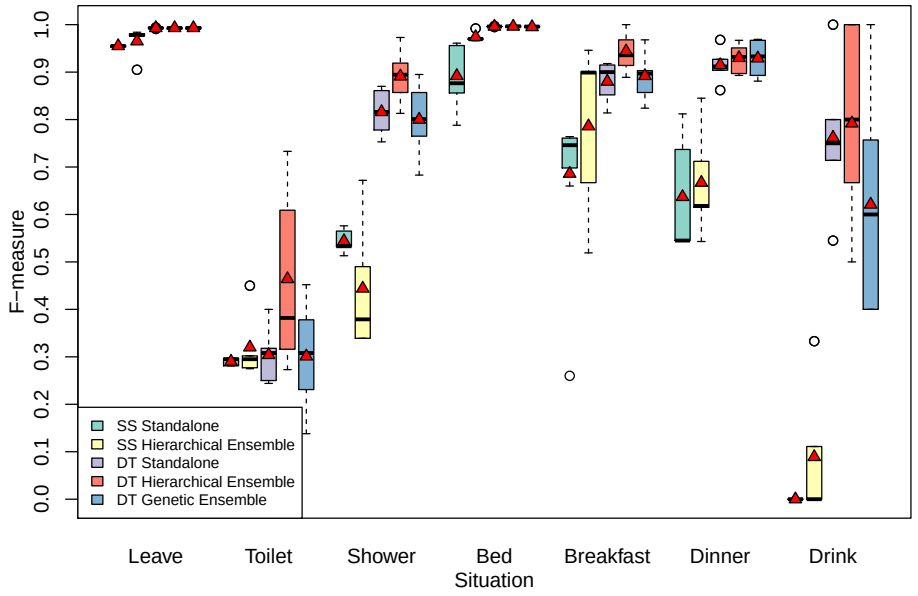


Figure D.6: A comparison of the f-measure performance of selected classification approaches on the House C dataset.