2011-03-01

Recognition Situations Using Extended Dempster-Shafer Theory

Susan McKeever
Technological University Dublin, susan.mckeever@tudublin.ie

Recognising Situations Using Extended Dempster-Shafer Theory

by

SUSAN MCKEEVER

February 2011

A Thesis submitted to the National University of Ireland, Dublin
for the degree of Ph.D in the College of Engineering,
Mathematics and Physical Sciences

School of Computer Science and Informatics

Professor Joe Carthy(Head of School)

Under the supervision of

Professor Simon Dobson

CONTENTS

List of Publications **xii**

Acknowledgements **1**

Abstract **3**

1 Introduction **3**
1.1 Pervasive Computing 3
1.2 Context-Awareness 4
1.3 Situations are key 5
1.3.1 Recognising situations 6
1.3.2 Terminology 7
1.4 Research Challenges 8
1.4.1 Removing the reliance on training data 8
1.4.2 Handling uncertainty 9
1.4.3 Addressing the challenges 10
1.5 Thesis Statement 11
1.6 Research Methodology 11
1.7 Contributions of this thesis 12
1.8 Structure of this Thesis 13

2 Background and Related Work **15**
2.1 Introduction 15
2.2 Context Awareness 15
2.2.1 Context 16
2.2.2 Nature of Context 17
2.2.3 Layers of context: sensors, abstracted context, situations 17
2.2.3.1 Context Uncertainty 19

2.3 Using situations . . . 20
2.3.1 Features of situations . . . 20
2.3.2 Discussion . . . 22
2.4 Related work - situation recognition approaches . . . 23
2.4.1 Learning approaches . . . 23
2.4.1.1 Bayes Models . . . 24
2.4.1.2 Decision Trees . . . 28
2.4.1.3 Situation lattices . . . 29
2.4.2 Specification-based approaches . . . 30
2.4.2.1 Logic based approaches . . . 30
2.4.2.2 Fuzzy membership and logic . . . 33
2.4.2.3 Custom approaches . . . 34
2.4.3 Dempster-Shafer theory-based approaches . . . 35
2.4.4 Our approach in the context of other research . . . 37
2.5 Conclusion . . . 39

3 Creating evidence decision networks using Dempster-Shafer theory 41
3.1 Introduction . . . 42
3.2 Concepts of Dempster-Shafer theory . . . 42
3.2.1 Frames of Discernment and mass functions . . . 43
3.2.2 Combining evidence . . . 43
3.2.3 Sensor discounting . . . 44
3.2.4 Example of evidence combination . . . 45
3.3 Evidential approach for situation recognition . . . 46
3.3.1 Situation DAGs . . . 46
3.3.2 Evidence decision network architecture . . . 48
3.4 Evidence operations for belief distribution and decision making 50
3.4.1 Evidence propagation across frames . . . 51
3.4.2 Combining evidence as 'OR' and 'is a type of' . . . 53
3.4.3 Distribution combined evidence to single situations . . 54
3.5 Evidence fusion issues . . . 55
3.5.1 Zadeh's paradox . . . 56
3.5.2 Single sensor dominance . . . 57
3.5.3 Evidence spread over time . . . 57
3.6 Alternative combination rules . . . 58
3.6.1 Murphy's combination rule . . . 58
3.6.2 Averaging Rule . . . 58
3.6.3 Summary of fusion issues . . . 59

3.7 Evidence decision network - summary of operations 59
3.7.1 Evidence example 59
3.8 Conclusion 62

4 Extending Dempster-Shafer theory with temporal and quality knowledge 63
4.1 Using temporal knowledge as evidence 64
4.2 Transitory evidence and situation durations 65
4.2.1 Extending the lifetime of transitory evidence 67
4.2.2 Calculating situation durations 68
4.2.3 Worked example of time extended evidence 68
4.3 Absolute time 70
4.4 Summary of temporal extensions 70
4.5 Using quality to weight evidence 71
4.5.1 Sensor and Context Quality 72
4.5.2 Using sensor quality 72
4.5.3 Identifying Static and Dynamic quality 73
4.5.3.1 Quantifying Static and Dynamic quality 75
4.6 Using quality parameters with evidential reasoning 76
4.6.1 Static quality parameters as evidence 76
4.6.2 Dynamic quality parameters as evidence 77
4.7 Time and quality notations on the situation DAG 77
4.8 Conclusion 78

5 Using evidence decision networks for real environments 80
5.1 Using evidence decision networks for pervasive environments 80
5.1.1 Generating knowledge 81
5.2 Establishing the situation DAG 83
5.2.1 Summary 86
5.3 Experimental data sets 86
5.4 Van Kastersen's Data set 87
5.4.1 Sensor Description 87
5.4.2 Situations in van Kasteren's Data Set 88
5.4.3 Van Kastersen Data preparation 89
5.4.4 Preparing the Situation DAG 89
5.5 CASL Data set 92
5.5.1 Sensor Description 92
5.5.2 Situations in CASL data set 93
5.5.3 CASL Data preparation 93

5.5.4 Preparing the Situation DAG 94
5.5.5 Situation DAG for CASL data set 98
5.6 Conclusion 98

6 Evaluation **100**
6.1 Situation recognition using the evidence decision network . . . 101
6.1.1 Worked examples using evidence decision network . . 105
6.2 Evaluation methodology for situation recognition 108
6.2.1 Evaluation parameters 109
6.2.2 Statistical Significance 110
6.2.3 Cross validation 111
6.2.4 Experimental set-up 112
6.3 Situation Recognition on van Kasteren's Data set 112
6.3.1 Experiment 1: Situation recognition accuracy without temporal knowledge 113
6.3.2 Experiment 2: Situation recognition accuracy with absolute time and time extended evidence 114
6.3.3 Experiment 3: Comparison with learning techniques . 120
6.3.4 Experiment 4: Comparison with published results . . . 122
6.3.5 Discussion of temporal evidence decision network results 123
6.4 Situation recognition on CASL data set 124
6.4.1 Experiment 1: Situation recognition accuracy without quality parameters 125
6.4.2 Experiment 2: Situation recognition accuracy with quality parameters 126
6.4.3 Experiment 3: Impact on situation recognition accuracy of individual sensor quality 129
6.4.4 Experiment 4: Sensitivity analysis of quality parameters 129
6.4.5 Experiment 4: Comparison with other inference techniques 133
6.4.6 Discussion of quality results 133
6.5 Situation Recognition using alternate fusion rules 134
6.6 Discussion 136
6.6.1 Benefits of the evidence decision network approach . . 137
6.6.2 Limitations 139
6.6.3 Summary 140

7 Conclusions and Future Work **141**
7.1 Conclusion 141

7.1.1 Contributions 143
7.1.2 Limitations of our work 144
7.2 Future work 144

A Appendix **147**

LIST OF FIGURES

1.1 Layers of Perception for a context aware system [18] 4

2.1 Context layers: sensors, context values, situations 18
2.2 Situation recognition approaches 24

3.1 Situation Directed Acyclic Graph 47
3.2 Evidence decision network architecture 49
3.3 Zadeh's paradox: scenario of location sensors in conflict on room location 57
3.4 Evidence Example 61

4.1 Transitory evidence for dinner situation 65
4.2 Time extension of evidence for 'preparing dinner' situation . 66
4.3 Impact of precision on a location sensor: (1) precision area (2) enclosed precision area (3) precision area crossing multiple areas 75

5.1 Knowledge contributions for the situation DAG 82
5.2 Layout of sensors in van Kasteren house floor plan [109] . . . 88
5.3 van Kastersen DAG for 'get drink', 'prepare breakfast', 'prepare dinner' situations 89
5.4 Sample of diary annotations CASL data set 94
5.5 Situation DAG for CASL showing sensor, context values and situations 98

6.1 Belief distribution algorithm 102
6.2 Algorithm for extending transitory evidence lifetime 102
6.3 Decision Algorithm for single situation occurrence 103
6.4 Decision algorithm for situation co-occurrence 104

6.5 Precision for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set 115
6.6 Recall for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set 116
6.7 F-measure for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set 116
6.8 Precision for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set . . 117
6.9 Recall for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set . . 118
6.10 F-measure for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set 118
6.11 Comparison of F-measure between Time extended Evidence, Naïve Bayes and J48 Decision Tree with one third training training data 121
6.12 Comparison of F-measure of time extended evidence, Naïve Bayes and J48 using Leave One Day Out cross validation. . . 121
6.13 Comparison of average precision, recall and f-measure for the evidence decision network with and without quality for the CASL data set 126
6.14 Comparison of precision by situation: with and without quality for the CASL data set 127
6.15 Comparison of recall by situation: with and without quality for the CASL data set 128
6.16 Comparison of f-measure by situation: with and without quality for the CASL data set 128
6.17 Comparison of f-measure for various sensor quality permutations. 130
6.18 Situation transitions for two situations against variable activity time decay of 0,2,4,6,8 minutes for 10 timeslices from time 11:20 to 11:29. Belief level taper more gradually for longer time decays. 132
6.19 Comparison of f-measure of evidence decision network with and without quality against NBayes and J48 classifiers on the CASL data set 133

LIST OF TABLES

2.1 Comparison of recognition approaches: ✗ indicates where knowledge must be found; ✔ indicates capabilities; **S** indicates 'sometimes' , such as HMMs can use sensor readings directly in its probabilities knowledge or it can use abstracted context values. 31

3.1 Evidence combination example 46
3.2 Evidential operations for belief distribution and decision making 51
3.3 Summary of evidential operations for the evidence decision network 60

4.1 Sample timeslices for 'getting drink' situation 68

5.1 Sample inference rule frequencies for 'prepare breakfast' . . . 91
5.2 Absolute Times for situations in van Kasteren's data set . . . 91
5.3 Quality parameters and values for CASL data set 97

6.1 McNemar's Results table 110
6.2 Precision, recall and f-measure of situation recognition on van Kastersen's data set (for evidence without time included) 113
6.3 Confusion matrix of inference on van Kasteren's data set . . . 113
6.4 Confusion Matrix on van Kasteren's data set using absolute time 115
6.5 Confusion matrix on van Kastersen's data set including absolute time and time extended evidence 119
6.6 Comparison of average f-measure for evidence decision network with no time, absolute time and time extended 119

6.7 F-measure comparison of Time extended evidence, Naïve Bayes and J48 for one third and Leave One Day Out cross validation . . . 122
6.8 Comparison of class accuracy of temporal EDN with published results from Ye [119] and van Kasteren [109] . . . 123
6.9 Precision, recall and f-measure of situation recognition on CASL data set . . . 125
6.10 Confusion matrix for CASL data set when quality not used . 126
6.11 Confusion matrix for CASL data set when quality parameters used . . . 127
6.12 Average f-measure against variable static quality parameter values for each sensor. Actual static quality values from the DAG are bolded (Ubisense 0.7, Calendar 0.6, Activity 1) . . . 130
6.13 Average f-measure against variable dynamic quality parmeters values for activity, ubisense and calendar sensor. Actual quality parameters used are highlighted in bold. . . . 132
6.14 Comparison of recognition accuracies (f-measure) for four fusion rules on van Kastersen's data set. Best results are highlighted in red. . . . 135
6.15 Comparison of recognition accuracies for three fusion rules on the CASL data set. Best results are highlighted in red. . . . 135

ABSTRACT

Weiser's [111] vision of pervasive computing describes a world where technology seamlessly integrates into the environment, automatically responding to peoples' needs. Underpinning this vision is the ability of systems to automatically track the *situation* of a person. The task of situation recognition is critical and complex: noisy and unreliable sensor data, dynamic situations, unpredictable human behaviour and changes in the environment all contribute to the complexity. No single recognition technique is suitable in all environments. Factors such as availability of training data, ability to deal with uncertain information and transparency to the user will determine which technique to use in any particular environment.

In this thesis, we propose the use of Dempster-Shafer theory as a theoretically sound basis for situation recognition - an approach that can reason with uncertainty, but which does not rely on training data. We use existing operations from Dempster-Shafer theory and create new operations to establish an evidence decision network. The network is used to generate and assess situation beliefs based on processed sensor data for an environment. We also define two specific extensions to Dempster-Shafer theory to enhance the knowledge that can be used for reasoning: 1) temporal knowledge about situation time patterns 2) quality of evidence sources (sensors) into the reasoning process.

To validate the feasibility of our approach, this thesis creates evidence decision networks for two real-world data sets: a smart home data set and an office-based data set. We analyse situation recognition accuracy for each of the data sets, using the evidence decision networks with temporal/quality extensions. We also compare the evidence decision networks against two learning techniques: Naïve Bayes and J48 Decision Tree.

LIST OF PUBLICATIONS

The following is a list of the publications related to this dissertation.

1. **Susan McKeever**, Juan Ye, Lorcan Coyle, Chris Bleakley and Simon Dobson. *Activity recognition using temporal evidence theory*, Journal of Ambient Intelligence and Smart Environments, Volume 2, Issue 3, August 2010.

2. Juan Ye, Lorcan Coyle, **Susan McKeever**, and Simon Dobson (2010). *Dealing with activities with diffuse boundaries*. Proceedings of Pervasive 2010 workshop on How to do good activity recognition research? Experimental methodologies, evaluation metrics, and reproducibility issues . Helsinki, Finland. May 17-21, 2010.

3. **Susan McKeever**, Juan Ye, Lorcan Coyle, and Simon Dobson. *Using Dempster-Shafer theory of evidence for situation inference*. In Proceedings of the 4th European Conference on Smart Sensing and Context, (EuroSSC '09), Lecture Notes in Computer Science, Springer Verlag. Guildford, UK. September 2009.

4. **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 workshop on Quality of context (QuaCon' 2009), LNCS, Springer Verlag, Stuttgart, Germany, June 2009

5. Lorcan Coyle, Juan Ye, **Susan McKeever**, Stephen Knox, Matthew Stabeler, Simon Dobson and Paddy Nixon (2009). *Gathering data sets for Activity Identification*. Workshop on Developing Shared Home Behaviour data sets to Advance HCI and Uniquitous Computing Research at CHI 2009.

6. Juan Ye, **Susan McKeever**, Lorcan Coyle, Steve Neely and Simon Dobson. *Resolving uncertainty in context integration and abstraction*. In Proceedings of the International Conference on Pervasive Services, Domenico Cotroneo and Julie McCann (ed), pages 131-140, Sorrento, IT. 2008.

7. **Susan McKeever**, Juan Ye, Lorcan Coyle and Simon Dobson. *A multi-layered uncertainty model for context aware systems*. In Adjunct Proc' of the international conference on Pervasive Computing: Late Breaking Result, pages 14, May 2008.

ACKNOWLEDGEMENTS

Firstly, I would like to thank my supervisor, Professor Simon Dobson for his expert advice and upbeat encouragement throughout this work - and for sticking with me on the Ph.D when various obstacles presented themselves.

I would also like to acknowledge the Systems Research Group in UCD Dublin. I would especially like to thank my colleagues Dr. Juan Ye (Erica) and Dr. Lorcan Coyle for their expertise and willingness to help at all times.

I would like to thank my extended family - my mum, my auntie Peig, sisters and brothers for their support, interest and encouragement. And finally, to my own gang. Thank you Patrick, Hugh, Louise and baby Sam for your unfailing knack of helping me to keep things in perspective. And to my husband Ed - thanks for his love, pride and support throughout the busy years of the Ph.D., without which I couldn't possibly have tackled this work.

Dedicated to my dad, Joe McKeever, who would have loved all of this.

CHAPTER

ONE

Introduction

> "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it. In our experimental embodied virtuality, doors open only to the right badge wearer, rooms greet people by name, telephone calls can be automatically forwarded to wherever the recipient may be, receptionists actually know where people are, computer terminals retrieve the preferences of whoever is sitting at them, and appointment diaries write themselves" (Mark Weiser, Scientific American, 1991) [111]

1.1 Pervasive Computing

Less than twenty year ago, Mark Weiser described a world where technology, seamlessly integrated into the environment, delivers the correct service to the correct user at the correct time and place. His description was visionary. In 1990, no Wireless Local Area Network standards existed, mobile devices were limited to processors running at a few tens of megahertz, PDAs had tiny amounts of memory, and PCs hosted 30-Mbyte discs [94]. Weiser's articulated vision initiated the field of ubiquitous or pervasive computing. Since then, dramatic improvements in wireless networking, intelligent mobile devices and the availability of cheap sensors have combined to make pervasive computing a realisable vision.

At the core of Weiser's vision is the concept of computers providing services without the need for people to explicitly instruct them or indeed, be aware of them. This vision includes two concepts: *invisibility* of technology and *context-awareness*. Computers are gradually achieving *invisibility* as they spread be-

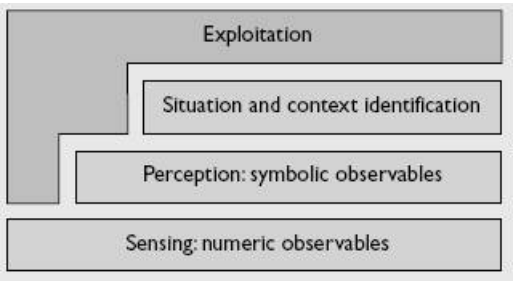


Figure 1.1: Layers of Perception for a context aware system [18]

yond the flat screened machine paradigm into tiny devices embedded in our physical environment. Users' attention and effort no longer needs to be focused in providing explicit input. Instead, intelligent devices embedded into everyday objects and worn by people sense input by detecting aspects of the environment. With the notion of *context-awareness*, a system automatically senses relevant information in its environment, and responds in some appropriate way to offer the appropriate service. It is this notion of context-awareness - the ability to automatically detect and make sense of an environment - that we will focus on in this thesis.

1.2 Context-Awareness

Context awareness is a prerequisite for pervasive computing. For computers to be invisible and calm, they are monitoring "context" in some way, in order to determine what behaviour they should execute. Early context-aware systems required applications closely bound to sensor output, such as the Active Badge Location System [110]. As context-aware systems evolved, the provision of a middleware context-service layer that encapsulates sensor details emerged. Thus, the application developer is shielded from the details of context gathering and interpretation and reusability of context is simplified. The Context Tookit [28], for example, provides sensor widgets that abstract sensor data into more meaningful, higher level contexts. More recently, the aggregation of context into the higher level abstraction of situation [26, 121, 47, 70] has provided a common semantics for higher level context of interest to applications. This layered approach to context detection is illustrated by Coutaz *et al.* [18]. In their conceptual view of context-aware systems, they separate context detection into sensing, perception and situation/context identification layers, as shown in figure 1.1.

Applications can therefore be completely decoupled from lower level sensor data, responding to or querying for situations that are monitored in a common middleware context services layer. For example, an office activity application that monitors an employee's location does not need to understand or interact with the underlying sensor tracking technologies. It is simply notified as the user goes from one situation to another such as 'busy at desk', 'gone to lunch', 'out of office' and so on.

This task of situation recognition is a critical, continuous dynamic process for pervasive applications that adapt their behaviour in or close to real time.

One of the first real context-aware systems was the Active Badge Location system [110] in the Olivetti Research Labs UK. The purpose of the system was to forward calls to the phone nearest to an employee, based on tracking the user's current location with infra red badges. Since then, various landmark real-life and research lab-based applications have appeared, grouped into two streams: (1) smart environments embedded with sensors (such as the RADAR office user tracking location system [6], the Cricket office user support system [83], the Gator Tech assisted living laboratory smart home [45] and MIT's Placelab live-in laboratory [53] that aim to sense people's activities or environmental conditions and (2) the use of smart mobile devices that can sense and respond to aspects of their environment, such as Schilit's mobile Parctab handheld device that detects nearby services and people [90], the Lancaster location-based tour guide [13] and Project Aura's continuous mobile services vision [36]. Our interest is in the smart environment category. In smart environments, sensors are embedded into everyday objects. Monitoring how these objects are used then provides clues as to what activities they are being used for. The trend is to move away from intrusive sensors that directly monitor people, such as cameras and motion detectors to cheaper, simpler, less intrusive sensors embedded into everyday objects.

1.3 Situations are key

As part of being context-aware, pervasive applications interpret the various states of the environment as *situations*. Situations are human understandable representations of the environment that are of interest to a pervasive application. Early examples of situation abstraction of context is found in Schmidt's work on context-aware infrastructure [91]. For example:

User sleeps: It is dark, room temperature, silent, type of location is indoors, time is nighttime, user is horizontal, specific motion pattern, absolute position is stable;

User is watching TV: Light level/color is changing, certain audio level (not silent), room temperature, type of location is indoors, user is mainly stationary.

As illustrated by Schmidt's examples, a situation will typically be composed of multiple pieces of context, each of which may in turn be derived from (multiple) sensor readings. By responding to situations (rather than lower level context or sensors), applications are shielded from the intricacies of sensor or low level context data. Therefore, promising context-aware applications tend to be situation-aware [121].

1.3.1 Recognising situations

Situation recognition is the ability of the pervasive system to assess the situation(s) that are occurring at any particular point in time. To illustrate the subtleties of situation recognition, we look at a particular scenario from a monitored smart home. The smart home is equipped with sensors in various locations throughout the house. In the kitchen, the cupboard doors, kettle, toaster and fridge are equipped with sensors that trigger when they are used.

The person is in the kitchen. It is morning time. They carry out a series of tasks, such as taking cereal out of the groceries cupboard, using the kettle, opening the fridge, and using the toaster. A fly-on-the wall observer might note that they are preparing breakfast. To 'know' at any point in time that the person is preparing breakfast, the observer could point to the individual tasks which in isolation may not confirm that breakfast is in progress, but considered together, form a picture of the 'preparing breakfast' situation. The fact that it is morning time indicates breakfast rather than dinner preparation. Some activities are particularly informative, such as the use of the toaster which is hardly ever used outside of breakfast time.

But for an automated situation recognition process, complications can arise:

- The 'preparing hot drink' situation, identified by the use of just the kettle and fridge, is also a possibility. Perhaps the process will distinguish this because looking at the other tasks that went before and after kettle and fridge makes breakfast a more obvious answer.
- Sensors can breakdown. If the kettle sensor does not trigger, the mon-

itoring system may be less certain that breakfast is happening, because the evidence is less compelling.

- Gaps of seconds or minutes might occur between the various tasks where 'nothing' happens, in terms of sensor activity. At the start of the breakfast preparation, the first task may not be enough to recognise the situation, but perhaps the system becomes more certain of breakfast as more tasks are done.

- The person does not prepare breakfast in the same way every day. The user sometimes uses the cooker and occasionally uses the microwave. The tasks are not performed in any particular order.

- Other situations can happen at the same time ('on telephone') while other cannot co-occur ('user asleep') so the system must declare only valid combinations of situations.

- 'Preparing breakfast' is a type of meal preparation. The system also wants meal preparation to be flagged whenever breakfast or dinner preparation is detected, so there is a hierarchy of situations to consider.

- A second occupant enters the kitchen and opens the fridge to get a drink ('preparing drink'). The system must now distinguish between the activities of each person.

Added to these types of considerations is the basic problem of what 'preparing breakfast' means in the first place. For some people, breakfast preparation will just consist of preparing a cup of tea, while for other people, such as our occupant, it involves more steps. The definition can be obtained in various ways, such as interviewing the user, applying some sort of general definition of what preparing breakfast 'usually' involves, or learning the meaning of sensor data captured in the person's house.

1.3.2 Terminology

In the pervasive systems literature, situations are sometimes referred to by the more specific term of 'activities', where the focus of interest of an application is the physical state or task of a person. Monitoring people's activities is a popular focus of situation recognition. The term 'situations' has a wider meaning, incorporating activities, but which may also refer to other scenarios of interest

to pervasive applications, such as a person's state of health, weather conditions, pollution levels and so forth. We use the term situation in this thesis to mean both activity and the wider meaning of situation.

1.4 Research Challenges

The process of situation recognition divides into two approaches (1) *Learning approaches* that require training data to develop a training model. The training model is then used to detect situation occurrence in real-world sensor readings. (2) *Specification*-based approaches that require a specification for each situation, where the specification consists of constraints on context. The constraints or rules are defined manually using expert knowledge. With these two approaches in mind, we highlight two particular research challenges:

1.4.1 Removing the reliance on training data

Pervasive systems need to understand the meaning of sensor events in order to recognise situations and thus trigger appropriate behaviour. Machine learning approaches are heavily used in the arena of situation recognition. These approaches require training data to be collected in the target environment during an explicit training phase. However, training data can be difficult and prohibitively expensive to acquire. Research in the pervasive computing domain usually relies on training data that has been captured in research labs or simulated in smart environments by users who are committed to and knowledgeable of the process [122, 68]. In real-life environments such as smart-home deployments, the privacy of users, and their willingness/ability to record their current situations in real time are major obstacles to the collection of reliable, well-annotated data. Tapia *et al*. [102], for example, collected training data in the homes of a 30 and 80 year old occupant. They used experience sampling, whereby users were periodically prompted to annotate their current activity. User fatigue, privacy and errors resulted in annotation rates of 17% and 24% respectively. Data can be annotated off-line, but this typically requires the video sequences for verification with resultant privacy concerns. Training data requires that each situation occurs during the training period. Some situations, such as security breaches or emergency scenarios cannot be easily produced. For these situations, manual specification may be the only option if training data does not exist.

A further challenge with training data is that it must be gathered for each individual environment in which a pervasive system will be used. User behaviour, sensor locations and sensor performance will differ across environments. People's homes and their furnishings have highly variable layouts, and individuals perform activities in many different ways. The same activity (e.g. brushing teeth) may result in a significantly different sensor activation profile based upon the habits, or routines of the home occupant and the layout and organization of the particular home. Likewise, if new situations are added, or sensor deployments are changed in an environment, the training data must be collected again to re-train the training model.

1.4.2 Handling uncertainty

Sensor data is inherently unreliable, noisy, prone to delay and imprecise. i.e. it suffers from uncertainty. For standard computer systems, an input such as a mouse click is definite. It may be incorrect, but it is definite. For pervasive systems which rely on sensor data, the input is ambiguous [32], so sensor data cannot be treated as fact, but simply evidence of fact. Users' actions can contribute to degradation of information quality, such as the failure of users to carry their locator tags [74]. Uncertainty of sensor information should be tracked and preserved to determine uncertainty at higher levels of context [48].

Furthermore, the process of interpretation of sensed context is also subject to ambiguity and approximation [39]. Some context concepts are fuzzy, so are subject to imprecision, such as the concepts of 'near', 'warm'. Inference rules are not constant, such as a user 'sometimes' using a microwave when engaged in the 'preparing breakfast' situation. Situations are aggregations of pieces of context. These specifications are limited to the available sensed contexts so are approximations only.

Various approaches are taken to deal with uncertainty. In Dey *et al.*'s framework [27], users are prompted to mediate in selecting the right context, if the underlying contexts are considered by the framework to be too uncertain. However, this approach would not be suitable in scenarios where user interruption is unacceptable or unavailable. Other approaches, such as those of Clear *et al.* [16] and Lei *et al.* [65] use a threshold approach at the application level, whereby an application will only select situations that exceed a particular certainty of confidence level or quality. Chalmers *et al.* [12] describe applications requesting a required certainty of context, where the level

of certainty required drives the ranges of context values returned, with wider ranges associated with higher certainty. Loke [69] suggests that context uncertainty should be linked directly to application behavioural impact. High risk behaviour should require greater certainty that the underlying context is correctly recognised. Numerous researchers include context quality parameters (such as a context confidence metric) in their work on context-aware systems, so that uncertainty can be quantified - these are discussed further in Chapter 4. Underpinning these approaches is the assumption that the reasoning process can track and quantify certainty levels for recognised situation(s).

For these reasons, a context-aware system should be able to track, quantify and resolve uncertainty associated with the situation recognition process.

1.4.3 Addressing the challenges

This thesis focuses on addressing the above challenges by applying Dempster-Shafer (evidence) theory to the problem of situation recognition. Our aim is to provide a transparent situation recognition approach that reduces or removes reliance on training data, and that caters for the uncertainty inherent in the situation recognition process. In order to apply Dempster-Shafer theory to the problem of situation recognition, we enhance basic Dempster-Shafer theory with additional evidential operations needed to process evidence. We separate evidence processing into belief distribution and decision stages in order to support the recognition process. We define two extensions to basic Dempster-Shafer theory in order to improve situation recognition rates: (1) extend evidence over time for detection of situations with duration and (2) encode rich sensor quality information.

The Dempster-Shafer approach in this thesis does not address situation recognition in multi-occupancy smart environments. It detects situations for single occupancy only, and thus is evaluated using data sets generated within single occupant environments. In a multi-occupancy scenario, more than one person (or entity) is monitored in the environment so situation recognition becomes more complex [68]. Neither does this approach attempt to deal with environments where the sensor performance or user behaviour changes over time. It assumes that parameters for the environment, such as situation definitions, are static.

1.5 Thesis Statement

The previous section explained how reliance on training data and handling uncertainty make situation recognition in pervasive environments a difficult and complex task.

To address these challenges, we apply an established mathematical theory of evidence, Dempster-Shafer theory to the problem of situation recognition. Using Dempster-Shafer theory, we identify a set of existing and create new evidential operations that will be used to generate situation beliefs for an environment. In addition, we create two extensions to Dempster-Shafer theory in order to incorporate additional sources of knowledge into the reasoning process: 1) temporal extensions to incorporate time into reasoning and 2) quality extensions to incorporate sensor performance and usage into reasoning.

The hypothesis of this thesis is that extended Dempster-Shafer theory will support situation recognition**.** Given that there are two extensions to Dempster-Shafer theory, this hypothesis will be tested in two parts:

We hypothesise that (1) the use of temporal knowledge in evidence will improve recognition accuracy (over using evidence only) and that (2) the use of quality of evidence sources will improve recognition accuracy.

1.6 Research Methodology

In our approach, we apply Dempster-Shafer theory to the problem of situation recognition. Dempster-Shafer theory is a mathematical theory of evidence [95] that supports decision making in uncertain conditions. It has been applied in a variety of domains such as information retrieval, military applications, cartography, image processing, expert systems, risk management, robotics and medical diagnosis *[92, 77, 61, 79]*. We use Dempster-Shafer theory because it addresses the main shortcomings of existing situation recognition approaches that we wish to address. Firstly, it removes the reliance on training data because it allows us to incorporate knowledge into an evidence reasoning structure. Secondly, it specifically caters for capturing and preserving uncertainty. Where uncertainty is known to exist (such as sensor reliability issues or inference rule uncertainty), this uncertainty can be quantified and separately stored so that the confidence or certainty of situation recognition is known.

We explain how to create an evidence decision network. An evidence decision

network is specific to an environment. It consists of the evidential operations needed to capture sensor evidence, propagate that evidence up to the various situations, and to determine which situation(s) are occurring. We capture the evidence operations needed to recognise situations for a particular environment diagrammatically, using our own diagramming notation, termed situation Directed Acyclic Graphs (DAGs).

We define our own extensions to Dempster-Shafer theory to cater for two sources of knowledge that inform the situation recognition process: (1) the inclusion of temporal knowledge and (2) the inclusion of rich sensor quality information. Both of these are used to enhance the accuracy of situation recognition.

To evaluate our approach, we use two real-life sensor data sets from a smart home environment and an intelligent office environment.

1.7 Contributions of this thesis

This thesis presents a novel evidence decision network approach based on Dempster-Shafer theory to enable dynamic situation recognition from sensor data. This offers improvements over existing recognition techniques because it enables situation recognition from uncertain information, using a theoretically sound framework that supports the encoding of domain knowledge. It also supports the use of temporal knowledge and sensor quality to enhance recognition.

The main contributions of this thesis are the following:

1. A situation recognition approach based on Dempster-Shafer theory to enable dynamic situation recognition from sensor data;

2. The selection of existing operations from Dempster-Shafer theory and the creation of *new* evidential operations and algorithms that propagate and process sensor evidence throughout an evidence decision network;

3. Extensions to Dempster-Shafer theory to enable both temporal evidence and sensor quality knowledge to be used in the situation recognition process. The extensions include the mathematical formalisms required to apply temporal and quality knowledge.

4. A diagramming technique (situation Directed Acyclic Graphs) to capture knowledge and structure for the evidence decision network;

5. Application of the evidence decision network approach and situation DAGs to two real-world sensor data sets;

6. Evaluation of situation recognition accuracy using our evidence decision network approach. This includes the evaluation of the temporal and quality extensions, and a comparison of evidential fusion rules.

1.8 Structure of this Thesis

The next chapter, Chapter 2, explores the research done on inferring context from sensor data. As part of this, we explore the nature and imperfections of context information. We examine the layers of context that exist from sensor level to situation level. We then examine the research done on defining situations. We select the specific features of situations, leading us to a list of features that our recognition approach will be required to cater for. We analyse the learning and specification based approaches from the body of research, examining their strengths and weaknesses.

Chapter 3 describes the structure and theoretical basis for the evidence decision network that we will use for situation recognition. Firstly, we explain the basic concepts of Dempster-Shafer theory. We explain how we document evidential knowledge for situations using a diagramming notation that we term situation DAGs. We explain how evidence from the situation DAG will be processed in two stages: belief distribution and decision making. We then explain the various evidential operations that we will use to process the knowledge in these two stages. Where no existing operation exists, we define new operations.

In Chapter 4, we define two extensions to Dempster-Shafer that we will apply in our evidential model to improve situation recognition: temporal and quality extensions. Temporal extensions use time patterns of situations to improve the reasoning process. We describe how to incorporate situation durations and absolute times of situation occurrences into the evidential reasoning process. We then describe how knowledge of sensor quality, both static and dynamic quality parameters, can be incorporated into evidence theory.

In Chapter 5, we demonstrate our evidential approach by creating situation

DAGs for two real-world data sets. We describe the data sets, and explain their temporal and quality characteristics. The creation of the DAG for each data set enables us to show how inference knowledge is captured and the evidence decision network populated.

In Chapter 6, we evaluate the accuracy of our evidential model for situation recognition, using the data sets and situation DAGs from Chapter 5. We test whether the inclusion of temporal and quality knowledge using our extensions significantly improves situation recognition results. We also compare the accuracy of our evidence decision network to two learning techniques, Naïve Bayes and J48 decision tree.

Conclusions, summary and directions of future work are contained in Chapter 7.

CHAPTER

TWO

Background and Related Work

2.1 Introduction

Since Weiser articulated his vision of pervasive computing in the early 1990s, a body of research has been devoted to developing the ability of a system to be 'context-aware' - i.e. the ability to automatically sense aspects of its environment in order to tailor the behaviour of applications.

In this chapter, we examine related work in the field of situation recognition in pervasive systems. As background, we first briefly explore the definition of context-awareness. We look at the concept of context more closely, explaining the separation of context information from sensors up to higher level of contexts, or situations. From this, we will examine the features of situations: temporal features, uncertainty, hierarchies and specifications. Finally, we will examine the approaches taken to recognising situations and identify the strengths and weaknesses of existing work to date.

2.2 Context Awareness

Pervasive computing implies an ability of technology to automatically determine its own behaviour based on detecting its input or instructions in some automated way. This notion of context-awareness requires an explicit evaluation and interpretation of context by the system. Therefore, to understand context-awareness, we need to examine how context is defined.

2.2.1 Context

The term context has been defined in pervasive computing research in a variety of ways. In earlier definitions, the focus of context was focused on describing the state of an 'entity', where an entity was usually a user. Schilit and Theimer [89] have one of the first references to the term 'context-awareness' of applications. They describe context as the user's location, identities of nearby people, objects and changes to those objects. Ryan *et al*. [87] also describe context in terms of the user, referring to context as the user's location, environment, identity and time. Other definitions are more general, and are based on the environment in which the application is set. Hull *et al*. [51] describe context in a more general way as the 'aspects of the current situation'. Ebling *et al*. [34] describe the context of an application as 'the environment in which a computation takes place'. Brown [9] defines context to be the elements of the user's environment which the computer knows about. In our opinion, definitions that are generalised descriptions of the environment are difficult to use, because there are aspects of the environment which may not be relevant to the application. The most useful definitions are application focused. Applications are the consumers of context, and therefore any definition of context should be focused on what the application needs to know in order to drive its behaviour. A heavily used definition of context comes from Dey *et al*. [28] and is described as:

"Any information that can be used to characterize the situation of an entity. An entity is a person, a place, or a physical or computational object that is considered relevant to the interaction between a user and an application, including the user and application themselves".

Dey's definition is useful because it narrows the definition to include information (via entities) that is relevant to the application. Winograd [112] reviews this definition, arguing that in using open-ended phrases such as ''any information'' and ''characterize'', it becomes so broad that context covers everything without boundary. According to Winograd, ''something is context because of the way it is used in interpretation, not due to its inherent properties. The voltage on the power lines is a context if there is some action by the user and/or computer whose interpretation is dependant on it, but otherwise is just part of the environment.'' This supports the view of context in terms of its relevance to application(s) in the environment. However, Dey's definition does impose a boundary by limiting entities to those that are of relevance to

the application. Henricksen [47] also imposes a boundary on context based on relevance to the application, stating: "the context of a task is the set of circumstances surrounding it that are potentially of relevance to its completion".

In our work, we are interested in situations. Situations are the lens through which pervasive applications view the world i.e. the states of the world to which the application will respond. Therefore, we will use a definition of context that is based around situations. This definition is close to Dey's and Henricksen's because it incorporates the concept of relevance to application, but it also incorporates the concept of situations. We also specifically refer to context *information*:

"Context information is any information that is used to identify situations, where situations represent views of the world of interest to the application".

2.2.2 Nature of Context

There are two particular aspects of context that are relevant to our work on situation recognition. Firstly, context information can be divided into a set of layers, ranging from sensor readings at the lowest layer, through to higher levels of abstracted and fused context. We will examine context as a layered hierarchy as this will provide the basic structure for our evidence-based approach. Secondly, context information is imperfect. This is a key factor in the uncertainty associated with situation recognition. We will examine the types of context imperfections that occur so that we can address the issue of uncertainty in our situation recognition approach.

2.2.3 Layers of context: sensors, abstracted context, situations

In earlier work on context aware systems, applications directly interpreted sensor readings. As context-aware systems matured, approaches included a separate middleware layer that shielded application developers from the complexities of sensor data or smaller fragments of context data. Dey's Context Toolkit [28] was the first seminal work to provide application developers with an abstracted view of sensors, using the concept of context widgets. Each widget, such as a user location widget based on GPS location, supplies context information to the application using a uniform interface. As context-aware systems

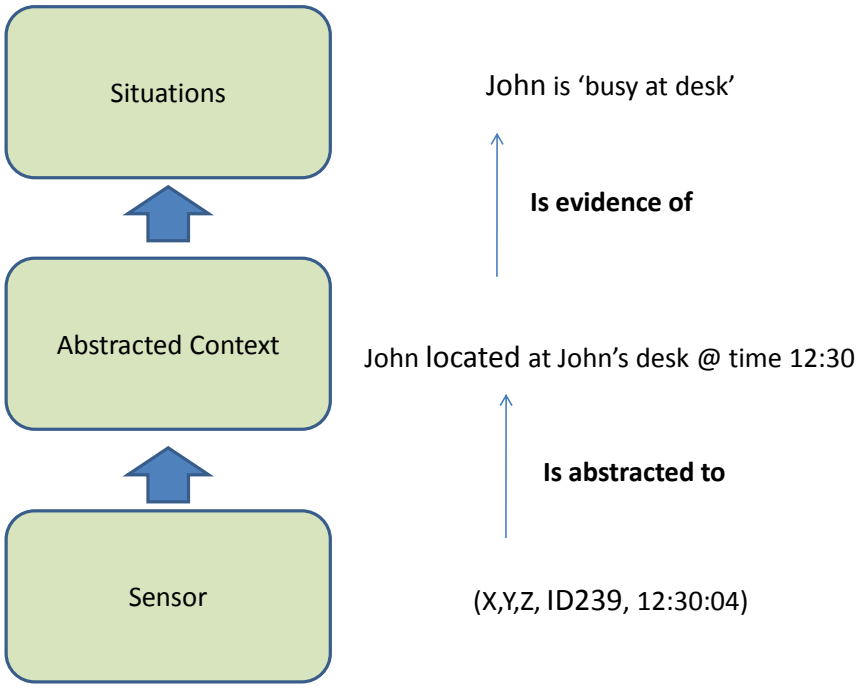


Figure 2.1: Context layers: sensors, context values, situations

have matured, this abstraction layer has become more functional, with the aim of providing applications with a higher situation level view of context, such as the work done by Loke [70], Ye *et al*. [119] and Padowitz *et al*. [80]. The description of context as a layered hierarchy is described variously in Coutaz *et al*. [18], Henricksen *et al*. [46] and our own work [73]. Looking at figure 2.1, we explain the layers of context as follows:

***Sensor layer*:** Traditionally sensors refer to physical sensors such as those that detect temperature, heat or motion. We use Loke's [70] broader definition of sensor that refers to 'any device or mechanism that is used to provide contextual information'. An electronic calendar that indicates whether the owner of the calendar is in a meeting or not is a sensor when it is used to detect the location of the user. The sensor layer produces information that is typically unreadable to humans, requiring sampling and translation in order to use. The information suffers from noise, time decay and is prone to breakdown.

Abstracted context: Raw sensor readings are translated to more human understandable forms of context by abstraction. Abstracted context is also sometimes referred to as 'secondary' context to differentiate it from primary context

(sensor readings) [28, 31]. For example, coordinate readings from a GPS location system are translated to street locations and temperature readings are translated to states of 'hot', 'warm' or 'cold' using a look-up process. Abstracted context is represented formally in the literature as part of context modelling work in a variety of ways, including: context predicates [84, 119], tuples [59] and object models [47]. For our purposes, we are interested in sensor values abstracted to context at specific points in time. We are not concerned with the modelled representation of abstracted context. We represent abstracted context diagrammatically using situation DAGs as described in Chapter 3.

Situations - Situations are high-level abstractions of context and serve as a more natural view for a developer. Situations can rarely be characterised using a single dimension of context, and are typically defined by fusing multiple information sources together. Situations have a hierarchy: situations can be subsets of other situations. Some situations can occur together at the same time ('preparing a meal' and 'using phone'). Some cannot co-occur ('person asleep' and 'person leaving house'). These and other features of situations are described in Section 2.3.1.

We are interested in the layered structure of context because our evidence-based approach will propagate evidence from sensors to abstracted context and then through to each situation in the situation hierarchy. This is explained further in Chapter 3.

2.2.3.1 Context Uncertainty

Context information suffers from a variety of imperfections that comprises the ability of a system to recognise context. Imperfections in context information have a number of causes including inaccurate and noisy sensor data, sensor failures and network delays and failures [73]. Henricksen's work [46] on modelling uncertain context provides a useful summary of the nature of context imperfection. Context is imperfect when it is:

- *unknown* when no information about the property is available; (for example, when a sensor battery expires);
- *ambiguous* when several different reports are available but the reports are conflicting (for example, when two distinct location readings for a given person are supplied by separate positioning devices);

- *imprecise* when the reported state is a correct yet inexact approximation of the true state (for example, when a person's location is known to be within a limited region, but the position within this region cannot be pinpointed to the required degree of precision); Imprecision results from the inherent limitation of sensors in precisely capturing a real world phenomenon [20];
- *erroneous* when there is a mismatch between the actual and reported states of the property.

The imperfection of context information is described via context quality metrics, such as precision, accuracy, confidence and freshness. We describe and use these metrics in our evidential approach, as described in Chapter 4. The uncertainty that results from using uncertain context is captured in our Dempster-Shafer based approach, as explained in Chapter 3.

2.3 Using situations

The purpose of our work is to develop an evidence-based approach to situation recognition. To recognise situations, we need to understand their characteristics or features. We then treat the list of features as a set of requirements for our recognition approach.

2.3.1 Features of situations

So far, we have discussed situations as high level abstractions of context that the application uses to drive their behaviour. A substantial amount of work has been carried out on defining and modelling situations. Earlier work on situations from Schmidt [91], Dey [26], Henricksen [47] and Loke [70] focus on the concept of situations as combinations of logical constraints on context information. More recent approaches from Costa *et al*. [17], Yau *et al*. [118], Ye [119] and Padovitz *et al*. [80] extend earlier work by adding temporal aspects, uncertainty and inter-relationships of situations. We analyse their work to distill out the features of situations that they have identified. This provides us with a target list of requirements for our situation recognition approach.

- *Situations can be learned or specified*. For learning techniques, training data is used to determine the underlying sensor (or abstracted sensor)

data that is associated with each situation, typically using a probabilistic model to represent occurrences of each situation. The main disadvantage of learning situations is the requirement for training data. *Specifications* of situations are required for non-learning approaches, where forward reasoning is used - inferring sensor data upwards to situation level. A situation specification consists of a set of constraints that capture the invariant characteristics of context and context combinations. A specification is typically represented in the following form: $s : \phi(\varphi_1, ...\varphi_n)$, where s is the symbolic name for a situation, $\varphi_i(1 \leq i \leq m)$ is a constraint on a certain context, and ϕ is a logical expression that composes all the context constraints using logical operators such as 'AND' or 'OR' [119]. Specification-based approaches rely on the availability of expert knowledge, and can suffer from the subjective and adhoc way of defining situations by experts [103].

- *Situations are dynamic with a variety of temporal characteristics.* A fundamental temporal aspect of situations is that they are dynamic, with continuous transition from one situation(s) to another over time as conditions in the environment change. They occur over a period of time or duration (such as a user in a 'prepare dinner' situation for 40 minutes) as captured in situation models by Costa *et al*. [17], Matheus *et al*. [60], and Yau *et al*. [118]. Thomson *et al* [103] identifies a minimum duration for some situations, to avoid ad-hoc identification of situations that must last longer then a single point in time. Situations can have an identifiable sequence or natural flow, as described by Padowitz [80]. For example a user with high blood pressure may indicate either sick or running. By looking at the previous situation, such as 'walking' the more credible transition (walking to running) can be used to help inference. Ye [119] also identifies situation sequences as important. A situation may be required to precede another with an obligatory sequence [119]. Situation may take place at particular times, and this can boost our ability to recognise them [120], such as a user's 'prepare breakfast' situation occurs in the morning. Situations can be co-occurring (such as a user can be 'reading' and 'watching TV' at the same time), or can be impossible to occur together such as 'asleep' and 'leaving home' [119]. This knowledge of co-occurrence is important because an application can become unstable if conflicting situations are declared by the system to be occurring at the same time. We view temporal aspects of situations as a rich source

of knowledge for reasoning so we incorporate temporal knowledge into our evidence based approach as described in Chapter 4.

- *Situations are inter-related in a hierarchy.* Situations can be atomic or composite [118]. An atomic situation is one that cannot be decomposed into other situations. A composite situation is one that consists of other atomic or composite situation, such as 'user asleep' consists of 'bed occupied' and 'nighttime' and 'room is dark'. The composite 'user asleep' situation is detected (at a particular time), if each of its component situations holds for the same time interval. Ye *et al*. [121] note that situations can be a generalisation of other situations. For example, a meeting can be a generalisation of a conference meeting or a supervisor meeting. Each of these is a type of 'meeting'. If a conference or supervisors meeting is occurring, then a meeting is occurring. This concept is also used by Thomas *et al*. [103] who defines re-usable situation specifications. This is based on the premise that situations are generalisations of other more specific situations. The inter-relationships of situations is important in our evidence based approach. We transfer evidence from sensors up to situations so we use the situation hierarchy as a routing method for propagating evidence. We use the nature of their inter-relationships to select the appropriate evidential operation, as described in Chapter 3.

- *Situations have an associated certainty or confidence*: Situations are approximations of the world only, and are detected from imperfect context. Therefore, there is a conceptual certainty or confidence associated with their recognition at any point in time. If available from a reasoning scheme, the confidence that a situation is occurring is typically expressed as a number between 0 and 1 [47, 63, 34, 74]. The notion of certainty of a situation that is made available to an application is supported in the situation specification modelling approach of Matheus *et al*. [60] and McKeever *et al*. [74]. However, it is usually associated with learning approaches that automatically supply some quantified certainty or probability with their output classifications. For example a Bayesian approach will output each situation (classification) with a posterior probability.

2.3.2 Discussion

Situations act as the 'interface' between applications and the lower level context information from which applications derive their behaviour. Specifica-

tions of situations are used when situations are recognised using a transparent, non-learning based approach. When learning techniques are used, the concept of situations is needed but a formal specification is less relevant.

Situations are dynamic, inter-related, and uncertain. They are detected from uncertain information and only represent approximations of the real-world. To recognise situations successfully, each of these features needs to be taken into account. In the next section, we examine related work on situation recognition. We examine the extent to which the various techniques consider the situation features in their recognition process.

2.4 Related work - situation recognition approaches

We categorise situation recognition approach into learning approaches and specification approaches [122]. Learning techniques rely on training data collected in the target environment in order to create a training model. Specification techniques such as logic approaches typically rely on domain knowledge to define how sensor data is interpreted as situations. We examine the various techniques applied by researchers in each category and discuss their relative merits and weaknesses in the discussion section 2.5.Our Dempster-Shafer based approach caters for uncertainty, does not rely on training data and includes temporal information in reasoning. Therefore, as we examine approaches from other researchers, we will highlight the extent to which their work is addressing these three features. We summarise the various techniques examined against these three criteria in Table 2.1.

A summary of the approaches, mapped against their requirement for knowledge, ability to deal with uncertainty and visibility of reasoning approach is shown in figure 2.2.

2.4.1 Learning approaches

A variety of machine learning techniques have been applied to the problem of recognising situations in pervasive environments. To apply such techniques, training data for the environment must be captured, where the data consists of sensor readings captured over a period of time. For supervised learning, the target situations to be recognised are defined in advance. The data is then

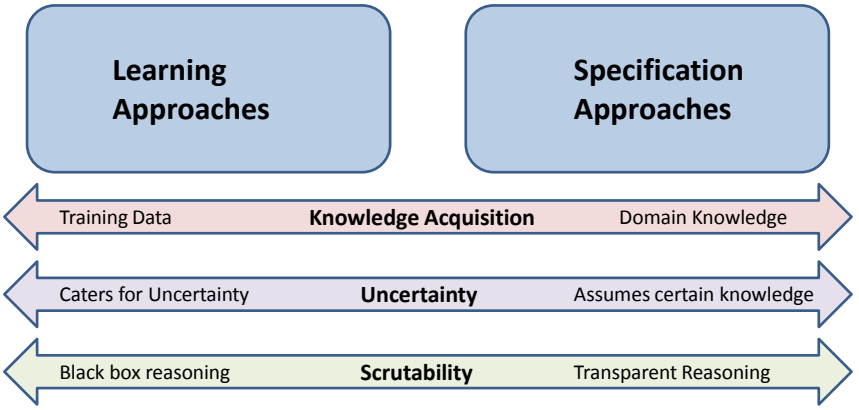


Figure 2.2: Situation recognition approaches

annotated by a participant in the environment, such as a user in a home. Alternatively, annotation is done as an off-line process by, for example, examining camera footage of the environment. The learning algorithm uses the annotated training data to establish a training model, which is used to infer future situations. For unsupervised learning methods, the learning algorithm can determine what situations are identifiable, and how to identify them [113]. The main strands of work on applying learning techniques are described next.

2.4.1.1 Bayes Models

A variety of techniques based on Bayes theorem have been applied to the problem of recognising situations, such as work from Castro *et al*. [11], Tapia *et al*. [102], Fox *et al*. [35], Korpippa *et al*. [62], Ranganthan *et al*. [84], Troung *et al*. [106, 105, 107], Albinali *et al*. [1]and van Kasteren *et al*. [109]. We will examine the three leading Bayes techniques applied to situation recognition.

Naïve Bayes Classifier

The Naive Bayes classifier is a supervised learning method that creates a probabilistic model of the environment from the training data. It assumes that all inputs (sensors) are independent. The situation with the highest probability at a point in time is typically assumed to be the recognised situation. Korpippaa *et al*. [62] used a Naïve Bayes classifier to recognise nine contexts for a user's

mobile device, such as walking, driving a car and running. Their results indicate that situations were extracted with 96% true positives in restricted scenarios of 9 situations. However, in real-world situations where they encountered situation transitions and undefined phenomena, the recognition accuracy fell to 87% true positives. Tapia *et al*. [102] use a Naïve Bayes classifier to classify the activities of two people in their own homes. Their main focus is the process of collecting training data and using simple easy-to-install sensors. In their implementation, they employed a large number of simple sensors, where the sensors were attached to objects in the home, such as beds, doors and windows. Their approach, whilst reliant on training data, incorporates temporal information in the recognition process. They consider two types of temporal information: duration and sequencing. They encode temporal information into their Naïve Bayes classifier to allow for average duration times of activities, and for sequences of sensor firings within an activity. They found that the capture of sensor firing during an activity duration was more useful in increasing recognition than using sensor firing sequences. Their focus on activity durations is interesting as we will also apply activity durations to Dempster-Shafer theory in a similar way. The real limitation for Tapia *et al*. was the capture of the initial training data. Their two participants were prompted to annotate their activity every 15 minutes using a technique called Experience Sampling Method (ESM). The response rates were 17% and 24% which made it difficult to get extensive and reliable data.

Bayesian Network

Bayesian networks are Directed Acyclic Graphs, where the nodes are random variables representing various events and the arcs between nodes represent causal relationships [44]. Bayesian networks have been applied by a number of researchers in scenarios where the causal links between sensors and situations are understood, and where training data is available to supply prior and conditional probabilities for the network. For example, Ranganathan *et al*. [84] applied a Bayesian network for detecting the activity in a meeting room, based on capturing the states of relevant aspects of the room, such as sound level, whether the projector is in use, the lighting level and number of people in the room. They trained the network on data collected in the smart room over a week,achieving recognition rates of 84% for activities in the room. Ranganathan's et al's work demonstrated how Bayesian networks can be used for activity recognition, with high recognition rates. However, they acknowl-

edge that the environment was an artificial research based environment with limited scope for unexpected events and good annotation. Their approach requires training data and does not consider temporal information. Gu *et al*. [40] use Bayesian networks in tandem with ontologies to identify a person's current activity (i.e., birthday party, reading, watching TV, dining, sleeping) in their home. They trained the network with data captured over a two week period. They do not supply actual recognition results but state that the network performs reasonably well, identifying the correct activity in most cases.This demonstrated that with a limited set of states to be recognised, a Bayesian network in combination with an ontology can gain 'good' recognition rates on two weeks of training data. Albinali *et al*. [1] address the issue of generating learning models from sparse data. They apply a set of optimisation steps to compensate for the lack of sensor data available for some situations - and then generate the Bayesian networks that best fit the resultant data. This is interesting because it addresses the difficulty of understanding the causal links between sensors and situations where sensor data is noisy or sparse [1].Whilst requiring training data, their approach reduces reliance on a full set of training data. A Bayesian network is a suitable scheme where training data is available and single higher level states are being detected without consideration of transitions between states over time. These approaches are not suited to environments where training data is difficult or impossible to obtain. Note that the Bayesian approaches address static state detection only, so do not include temporal sequence information in the reasoning process.

Hidden Markov Models and temporal learning approaches

In the following approach, temporal information is included in the recognition process in the form of sequences of situations and intervals between situations.

Dynamic Bayesian networks are Bayesian networks that also consider the *sequence* of states being detected. Therefore, they are useful when there is a pattern of situation occurrence over time. This is relevant in scenarios such as smart home environments where users perform activities in an identifiable order, such as take a shower, have breakfast and so on. The most common dynamic Bayesian network is the Hidden Markov Model (HMM). The HMM is a probabilistic model consisting of a hidden variable (situation) and an observable variable (context events/ sensor readings) at each time step. A known transition probability distribution is used to support the identification of a sit-

uation at time t, based on readings at previous time slices [20]. HMMs have been applied to detect activities with temporal patterns. Van Kastersen *et al*. apply HMMs to the detection of activities of a person in their home. They achieve up to 79.4% recognition of activity recognition. They also note that for their particular scenarios, a minimum of twelve days of training data were required to maximise their situation recognition results. Clarkson *et al*. [15] used HMMs for context recognition methods for wearable computers with the means of a wearable camera, and environmental audio signal processing. They achieve recognition rates for a simple set of situations of between 85 and 99%. HMMs consider short term sequences only, based on the previous state.

Choujaa *et al*. [14] employ temporal knowledge in their approach. They observe that long term sequences (such as activities from an earlier part of the day) are also useful, and employ both short term and long term sequences in their activity inference approach, using a probabilistic framework obtained from training data. Their approach also caters for gaps in the data. They evaluate using a data set generated from mobile phones. With eight weeks of training, user activities can be inferred with over 70% accuracy when every other hour is missing in the day.Their approach therefore alleviates the requirement to have a complete training data set, allowing for missing training data to be managed, although reliance on training data is not completely removed. Jakkula *et al*. [55] apply temporal knowledge about activities in order to detect anomalies in real time in a smart home, such as a cooker left on. The purpose of detecting anomalies is to monitor resident safety. They use training data to discover frequent sequences of sensor patterns, and temporal relations between sequences. Their approach supported the detection of anomalies occurring over a day, using 59 training days from their MavHome smart home environment, although exact results are not provided. These approaches [14, 55] are of interest to use because they illustrate the value of reasoning with temporal knowledge. Choujaa *et al*. [14] use the temporal sequence of activities to improve recognition, whilst Jakkula et al. [55] employ both sequence of activities and the length of time between sequences to detect when an exception or anomaly occurs. However, they both rely on the availability of training data for development of a probabilistic learning model.

Discussion of Bayes approaches

Bayes models allow higher level states (situations) to be captured based on a probabilistic model of lower level and/or sequence of states. Both Naïve Bayes and Bayesian networks identify situations at specific points in time. Naïve

Bayes takes a universal view across the environment, not requiring any causal links between lower and higher level states to be specified. Bayesian networks (including HMMs) require that causal links between sensor values and situations are known, so this can reduce the number of conditional probabilities that need to be encoded into the network. Because a Bayesian network and Naïve Bayes classifier do not necessarily require a transition from one state to another in order to compute the global or local state of the network, they can be useful schemes for computing a single higher level context as an abstraction of numerous primitive contexts. However, the disadvantage of this approach is that they do not explicitly support temporal knowledge about situation transition. If situation recognition is dynamic, with recognition being carried out over time (such as monitoring activities of a person throughout the day), then the inclusion of temporal knowledge using a dynamic Bayesian network, such as HMMs, may be very useful, as demonstrated by the temporal learning work of van Kasteren [109], Choujaa and Dujaa [14]

Bayesian approaches allow for context uncertainty because they build a probabilistic model based on correlating lower level context values against higher level situation occurrences. Uncertainty of sensor data and inference rules are absorbed 'invisibly' into the model, with no requirement to explicitly quantify uncertainty for the training model. This 'black box' approach, however, makes it difficult to scrutinize the reasoning process. For environments where people need to understand the reasoning process, this lack of transparency of how situations are inferred from sensor data may be problematic.

A common feature of the Bayes models is their requirement for training data to establish a model of the environment. If supervised learning is used, data must be annotated. If training data cannot be collected, then an alternative method will be required. We summarise the Bayes approaches against knowledge acquisition, transparency and ability to cater for uncertainty in table 2.1.

2.4.1.2 Decision Trees

Decision trees, built on information entropy, have also been used to classify sensor data into activities based on features extracted from sensor data. Using a training data set, decision rules are determined from the training information by finding the link between individual values and classifications (situations). The resultant decision tree forms a model for future test classifications. Bao *et al*. [7] compare classification results using decision tree and Naïve Bayes

classifier. Their experiment classifies user activities based on accelerometers attached to users that track movements. They noted that the rule-based approach of decision trees seems to capture combinations of feature values that result in good recognition accuracy. For example, the tree distinguishes "window scrubbing" from "brushing teeth" because the first activity involves more energy in hip acceleration even though both activities show high arm acceleration. The Naïve Bayes classifier, on the other hand, may not be able to model such rule combinations adequately. Their approach, however, relies on the availability of training data to determine decision rules. Also, it detects static states only, without considering temporal information such as activity sequences. Logan *et al*. [68] tested activity classification for a smart home environment using both Naïve Bayes and J48 decision tree. The decision tree classification had consistently superior performance over Naïve Bayes although the reason for this is not explained. They also noted that decision trees have the added advantage of relative transparency - allowing visibility of which sensor inputs contribute to classification.However, as with Bao *et al.'s* [7] work, their approach requires training data to determine decision rules and does not consider temporal information.

2.4.1.3 Situation lattices

Ye [121, 119] applied lattice theory to the organisation and recognition of situations. A situation lattice is a data structure which consists of a set of nodes, where each node corresponds to the set of contextual information relating to a set of situations. The situation lattice captures the relationship between situations, capturing both the generalisation relation of situations and the dependence between situations. Its structure supports the identification of situations that can occur concurrently, or that are impossible to occur together. A combination of learning and specification are used to establish and optimise the situation lattice. Given the (partial) reliance on training data, we have categorised it under learning techniques. Learning is used to uncover the relationship between abstracted context and situations. Expert knowledge is used to define how abstracted context is derived from sensor data (termed context predicates by Ye), and to tweak the lattice manually to boost recognition accuracy. Ye compares the recognition accuracy of the lattice against Naïve Bayes and J48 decision trees, with the lattice providing comparable accuracy to the other techniques. Ye notes however that the lattice computation load during recognition may be huge when faced with many situations.

Ye's work on situation lattices was completed as part of the work of the Systems Research Group, University College Dublin. It is a separate, parallel strand of work to the work in this thesis, with no overlap with the contributions of this thesis. Situation lattices are based on lattice theory whereas our approach in this thesis uses Dempster-Shafer theory. Ye's work on situation recognition using situation lattices, however, uses transparent, repeatable methodologies with a publicly available data set. Therefore, it provides a useful basis for comparing situation recognition results using situation lattices with results using the extended Dempster-Shafer approach of this thesis. This comparison is included in the evaluation work in Chapter 6 of this document.

2.4.2 Specification-based approaches

In the non-learning approaches, each situation needs a specification that is derived from expert knowledge. The specification consists of a set of constraints that define which 'pieces' of context information indicate the occurrence of the situation. Such techniques therefore do not rely on training data. Generally, these approaches do not address uncertainty as part of their approach (with the exception of Dempster-Shafer theory) but assume that sensor data is factual and reliable.

2.4.2.1 Logic based approaches

The logic-based approach uses a set of rules to evaluate sensor values and determine which situation of interest is happening. Situations are specified as constraints on context information linked by a group of logical operators. For example, the situation of 'preparing breakfast' might be specified as ('fridge used' or 'kettle used') and 'cup used'. This approach is more typically associated with earlier work on context reasoning, such as work by Dey [25] . Dey describes abstracting situations using reasoning modules called 'widgets'. Situations are abstracted from context information by logically 'AND'ing a set of context conditions together. The approach however is restrictive because it cannot cater for alternatives such as 'user in room 309 OR room 208'. Neither does it cater for the uncertainty of context information. Henricksen 's work [47] addresses these restrictions by using a richer set of logical combinations in situation specifications. Loke *et al*. [70] uses logic programming for representing and reasoning about situations. Situations are represented as logic programs,

	Knowledge acquisition					Transparency	Uncertainty		Temporal reasoning
	Training data in the environ't	External data	Domain: Sensors to context values:	Domain: Context values to situations	Domain: Inference rules		Imprecise sensors	Uncertain rules	
Bayesian network	✗		S	S			✓	✓	
Naïve Bayes	✗		S				✓	✓	
HMMs	✗		S	S			✓	✓	✓
Decision Trees	✗		S			✓	✓	✓	
Logic approaches			✗	✗	✗	✓			
Fuzzy logic			✗	✗	✗	✓	✓		
Dempster-Shafer			✗	✗	✗	✓	✓		
Situation lattices	✗		✗			✓	✓	✓	
Web mining		✗				✓		✓	
Extended Dempster-Shafer theory (this thesis)			✗	✗	✗	✓	✓	✓	✓

Table 2.1: Comparison of recognition approaches: ✗ indicates where knowledge must be found; ✓ indicates capabilities; **S** indicates 'sometimes' , such as HMMs can use sensor readings directly in its probabilities knowledge or it can use abstracted context values.

termed 'situation programs' by Loke. A situation program is a collection of rules, where each rule specifies constraints on sensor readings. Situation programs can refer to other situation programs, allowing a hierarchy of inference to be used. His approach allows for situations that can or cannot co-occur to be specified, which increases the stability of the situation recognition process.

In more recent work, logic approaches have been combined with ontologies [41], [84], [105, 106, 107]. Ontologies allow the various relationships between sensors, context and situations to be captured in a formal way. The ontologies thus provide the semantic interpretation of the context information and the logic approach provides the rules evaluation mechanism for combining knowledge in the ontology. Ranganthan *et al*. use ontologies combined with probabilistic logic in their context reasoning infrastructure Gaia [84]. They represent context information as predicates, such as *'location(jeff, in, room 3105)'* . The structure and semantics for the various valid predicates are stored in the ontology. They attach a confidence value to predicates to quantify the certainty of the statement. Gu *et al*. [41] use an ontology to capture context information as context predicates similar to Ranganathan *et al*. They derive high-level context using both ontology reasoning with description logic and user-defined reasoning that is defined in specific rules in first-order logic.

Temporal information has been incorporated into specification based approaches, using temporal logic. Jakkula and Cook [56] use Allen's temporal logic relations [2] as the basis for defining temporal rules across activities. The rules allow the sequence and overlap of activities to be encoded into a rule base. They compare the predictive accuracy of activities with and without the temporal rules, noting an improvement when temporal rules are applied. This indicates that temporal information will help with situation recognition, and is desirable for us to include into our Dempster-Shafer based reasoning approach. Augusto *et al*. [5] introduce the temporal operators 'ANDlater' and 'ANDsim' in Event-Condition-Action rules, upon which temporal knowledge on human activities can be specified. This allows events that co-occur (ANDsim) and events that happen in a sequence (linked by ANDlater) to be used in reasoning. For example, if the user in a smart home is detected as in the kitchen, 'ANDlater' no movement is detected in the kitchen or moving from the kitchen, the situation of 'user fainted' is detected. Therefore, their work indicates that co-occurrence and sequence of situations, if known, will help with recognition.

Gottfried *et al.* [38] use temporal logic to describe, constrain, and reason about

sequences between two events. They use a wider set of temporal operators than those used by Augusto *et al*. [5], allowing a richer set of sequence relationships such as events happening immediately after one another ('meets'), an event that always happens as another event is finishing ('finishes'), and events where one event encompasses another ('during'). The thirteen operators are described fully in [38].

Logic approaches rely on the ability of systems designers to specify rules in advance. This can be problematic in scenarios where the number of sensors is high and situations become more complex. This will be the case with our proposed approach. However, logic approaches generally assume that data sources are reliable, with data treated as fact rather than potentially uncertain. Dempster-Shafer theory as used in our approach is a sound basis for encoding uncertainty of data sources.

2.4.2.2 Fuzzy membership and logic

Fuzzy logic, supported by fuzzy set theory, accommodates imprecise states or variables. Fuzzy set theory allows a precise but subjective membership value to be applied to indicate the extent to which a value is part of a fuzzy group [58]. For example, if an activity sensor monitors the keyboard activity of a user at a desktop, the activity level may be described as 'active' or 'inactive'. The human understanding of these states is that 'active' gradually descends into 'inactive' rather than a hard 'sudden' boundary between them. The fuzziness membership of a state is expressed as a number between 0 and 1, where 1 indicates full membership of a state, and all below 1 indicates the degree of membership of the state. This interpretation of 'degree of membership' depends on the purpose of the fuzzy membership function. For example, fuzzy membership applied to the fuzzy concept of activity allows the 'nearness' to a state to be specified. Whereas, applying fuzzy membership to a crisp concept such as location of a person conveys the certainty that a person is in a particular location.

Fuzzy membership is useful in context-aware systems to define the level of imprecision associated with a context value. Preuveneers *et al*. [82], Ranganathan *et al*. [84], Korpippa *et al*. [62], Mantyjarvi *et al*. [72] all use fuzzy membership functions to quantify the extent of imprecision of context. Anagnostopoulos *et al*. [3, 4] define fuzzy inference rules to deal with imprecise knowledge about situational context. They use ontologies to hold situation specifications

and a fuzzy function to calculate the degree of membership of a situation i.e. the extent of belief that a user is involved in a particular situation. Their approach caters for the uncertainty of imprecise context states, and propagates this imprecision to situation belief level.Moon *et al*. [93] capture the uncertainty of temporal information by combining fuzzy and temporal logic, using fuzzy branching temporal logic. This temporal logic can model dynamic systems with uncertain temporal information and a branching time model. It has fuzzy events and fuzzy states in its temporal model. It can express fuzzy logical formulae and fuzzy temporal relationships among those formulae, such as sequences. The degree of satisfaction of the constraints on a system can then be represented as a truth value of a Fuzzy Branching Temporal Logic forumula. Moon et al's approach is interesting because it captures uncertainty of temporal information, in additional to encoding temporal information into the reasoning process.

Fuzzy membership is a useful way to quantify imprecision and uncertainty. We see that fuzzy membership is complementary to other approaches, such as temporal logic [93]and ontologies [4, 3]. In Dempster-Shafer theory, fuzzy membership will be useful in supporting the allocation of belief from sensors to abstracted context where the context suffers from imprecision or time decay, so we will incorporate fuzzy membership into our Dempster-Shafer approach.

2.4.2.3 Custom approaches

A number of other approaches fall outside the mainstream theoretical approaches, whereby researchers have applied their own custom approaches to the problem of identifying situations from sensor data. For example, Palmes *et al*. [81] use an object web mining approach to activity discovery that does not assume any particular sequence of activities. The important feature of this work is that it offers an alternative way to discovering knowledge without using training data or relying on expert knowledge. They use public web-based descriptors of activities to discover specifications. This does require that descriptors are both available and representative of the situations to be detected. Interestingly, they note that activities may have a distinct series of steps but with no particular sequence. This is an interesting observation because where it is applicable, the case for using HMMs and other sequence reliant approaches is reduced. They note that in such cases, relying on sequence of events for activity recognition may significantly limit the accuracy and applicability of models that rely particularly on object sequence.

Myllymaki *et al*. [78] uses various uncertainty parameters in their custom algorithm for detecting a user's location from a number of contributing contexts. They quantify concepts such as 'association confidence' (likelihood that an entity is with its 'sensor') to capture the problem of users not carrying their locator tag or mobile phone. To cater for conflicting reports from sensors, they apply a ranking algorithm to determine which report is most likely to be true. They also incorporate the concept of time decay by applying a time decay to the various sensors. Their approach is interesting because it quantifies and reasons with the types of quality issues that can occur. We wish to do the same in combination with Dempster-Shafer theory. The weaknesses of their approach is that it is difficult to generalise it across different environments. Also, they do not fuse sensor outputs, but treat one sensor as correct in the face of conflict. This may lead to less reliable results than fusing results from multiple sensors.

Padovitz *et al*. [80] define situations as a vector of context attributes, the equivalent of a set of context events. Their reasoning engine matches detected context values to the stored situations. If no matching occurs, they employ a distance algorithm to detect the closest situation, thereby dealing with the discrepancy caused by uncertain context. Interestingly, they go beyond discrete state recognition by using 'situation natural flow' to assist in reasoning. For example, they prevent unnatural transitions such as walking to sick based on heart rate detection. Haghighi *et al*. [43] propose a similar reasoning mechanism to [80] by using a K-nearest neighbor algorithm to find the nearest matching situation if the detected context facts do not cleanly match a situation definition. These algorithms do not rely on training data, and are conceptually simple. However, their disadvantage is unmatched situations may be deemed equally or very closely similar, thus making it difficult to distinguish between uncertain situations.

2.4.3 Dempster-Shafer theory-based approaches

In the previous section, we explained the techniques used for situation recognition. In this section, we will describe why we propose to use Dempster-Shafer theory for situation recognition, and related work by other researchers using this technique.

Dempster-Shafer theory is a theory of uncertain reasoning that is widely used in domains where information (evidence) is known to be imperfect and reasoning uncertain, such as medical diagnosis, quality control and process engi-

neering *[92, 77, 61, 79]*. The theory supports the combination of evidence from different sources, arriving at a degree of belief (represented by a belief function) that takes into account all the available evidence. More extensive details of Dempster-Shafer theory are described in Chapter 3.

Dempster-Shafer theory has been applied to a limited extent in the domain of context-aware systems, but has recently received greater attention in the work of Hong [50] and Zhang [125]. The first use of the theory for context-aware systems was Wu's approach [116, 114, 115] to fuse sensor data into higher level contexts. Wu applies Dempster-Shafer theory to the monitoring of people conversing in a meeting room, using an audio and camera sensor to detect the focus-of -attention of a user. Wu's main contribution is the definition of a dynamic discount factor for sensors that changes over time. However, the discount factor is reliant on ground truth availability shortly after fusion takes place which is not a workable assumption for many situation recognition scenarios. Wu's application of Dempster-Shafer theory deals with reasoning about lower level context only; it does not address the propagation of evidence to situation level.

Hong *et al*. [50, 49] apply Dempster-Shafer theory to define an evidence based activity (situation) model that uses sensor data for activity recognition in a smart home. Hong expands 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 advances the application of Dempster-Shafer theory to situation recognition by showing how evidence can be propagated across frames. Their approach is demonstrated via worked examples with sensors. Their worked examples are from a smart home, where simple binary sensors are embedded in everyday objects in a kitchen. As each object is used (such as opening the fridge), it contributed evidence to higher level activities such as 'making a drink'. Hong's approach does not address the temporal spread of evidence over time, but treats activity recognition as if all tasks in the activity happen at the same time. This is a simplification that we address in our approach. 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*. [125] use Dempster-Shafer theory for reasoning about activities. Their work focuses on resolving two issues (1) computation intensiveness of evidence fusion (2) Zadeh's paradox, whereby conflicting evidence sources can give paradoxical results by granting majority belief to a monority opinion.

Zadeh's paradox [124] is a well documented problem with Dempster-Shafer theory. It, and our solution to it, is described further in Chapter 3, Section 3.5.1. Zhang *et al*. address computational intensiveness by excluding evidence sources with small belief contributions, so as to reduce the size of the evidence set. To address Zadeh's paradox, they exclude the conflict from the evidence combination process, which prevents Zadeh's paradox from occurring.

In terms of temporal information, the existing Dempster-Shafer approaches to situation recognition do not support the inclusion of temporal information in the reasoning process. The inclusion of temporal information with Dempster-Shafer theory is limited to Strat's work [101] on evidence *projection*. Projection determines the impact of a body of evidence at a point in time in the past or the future. This allows evidence that occurs at a point in time to be projected to a different time, with some degree of certainty that the evidence is still valid. For example, if a person A is detected at a location X at time t, this evidence may be re-used in a location Y at a later time, provided that it is feasible for the person A to have reached location Y within the elapsed time. In our approach, we wish to include temporal information to enhance our ability to recognise situations. Strat's work on projection was aimed at scenarios where separate states under detection exist at different times but where evidence may be re-used to detect these states. Our requirement has some conceptual overlap with Strat's approach, as we wish to make evidence last over time. However, it differs in that we are detecting the same 'state' or situation that endures over time. We will use the concept of evidence being re-used over time in order to support the recognition of situations that have an identifiable duration. This is described further in Chapter 4.

2.4.4 Our approach in the context of other research

The Dempster-Shafer theory approach to reasoning with uncertainty has a number of strengths over other approaches: (1) It removes reliance on training data. (2) It supports the capture and presentation of uncertainty, which is very useful for environments with noisy data and uncertain reasoning (3) Evidence can be allocated to more than one hypothesis, which is closer to human reasoning than a probabilistic approach where all individual hypotheses are given a separate probability.

We see a number of gaps in the *existing* approaches to Dempster-Shafer theory for situation recognition:

1. None of the approaches considers the temporal aspect of evidence and situations. They treat evidence and situations as static states. Evidence is assumed to co-occur and time of occurrence is not included in reasoning.

2. Sensor reliability can be applied as a simple discount mechanism, defined as part of Shafer's original work on the theory [95]. However, richer sensor and context quality semantics that have a variable impact on sensor performance over time are not incorporated (such as time decay, precision);

3. A decision level is not provided in existing approaches that considers which situation are/can be occurring.

When considering how our approach improves situation recognition in contrast to other (non Dempster-Shafer) approaches described in Section 2.4.1 and Sections 2.4.2, we summarise as follows:

1. Data-driven machine learning approaches address the issue of uncertainty because they absorb uncertainty of sensor and rule information. However, if training data is not possible to obtain, a learning approach cannot be used.

2. Specification-based approaches using logic approaches are heavily used in situation recognition to support rules-based reasoning. Dempster-Shafer theory, as applied in this thesis, will also support reasoning with rules based information. However, Dempster-Shafer theory has the additional advantage of explicitly capturing and preserving a generic measure of uncertainty.

3. Fuzzy logic approaches capture uncertainty by capturing the concepts of degrees of truth or imprecision. It has been used as a stand alone logic approach and in combination with other logic approaches to capture context imprecision, situation uncertainty and temporal information uncertainty as explained in Section 2.4.2.2. Dempster-Shafer theory is ideally suited to capturing fuzzy concepts because it allows uncertainty to be explicitly quantified and preserved. We exploit this capability by capturing and encoding rich sensor quality and context uncertainty into our process.

4. Temporal information is used in both learning and specification approaches to enrich situation recognition. At present, it is not incorpo-

rated into Dempster Shafer theory for the purposes of situation recognition. We incorporate both situation duration and time of situation into our approach to improve situation recognition.

2.5 Conclusion

In this chapter, we explored the background to context information with a view to understanding the status of approaches in the field to situation recognition. We explained the link between sensors, abstracted context and situation. Situations are the states of the environment of interest to application. From looking at situation features identified in the literature, we noted that situations are hierarchical, dynamic with rich temporal features, and driven by uncertain sensor data and inference rules. These complexities drive the requirements for whatever reasoning technique is used for situation recognition. From reviewing the related work, it is clear that no single reasoning scheme is a panacea for situation recognition. Each approach has its own advantages and limitations, so different environments will suit different techniques. Our conclusions on the existing body of work on situation recognition are:

- No single approach to situation recognition is a solution for all scenarios. Each reasoning scheme is applicable in certain scenarios, depending upon such factors as the availability of training data, the requirement for humans to be able to understand the context reasoning process, the expert understanding of causal links between lower level and higher level contexts and the requirement to identify multiple situations in a hierarchy versus identifying from a 'flat' set of situations.
- Some researchers provide actual recognition results for their recognition technique. However, it is not appropriate to compare situation recognition results across different experimental set-ups from different researchers because the sensors, accuracies, situations and complexities are specific to the environment set-up used in each case. Bao *et al*. [7], for example, note a difference in inference results of 95.6% in a lab environment versus 66.7% in a naturalistic setting. Direct comparison would require the same data-set to be used by multiple reasoning schemes, using the same methodology.
- Temporal factors are not catered for in existing Dempster-Shafer approaches to situation recognition. Other (non Dempster-Shafer) ap-

proaches have exploited temporal information in situation detection, where this temporal information includes relative times of situations, sequence of two or more situations, and duration of a situation. Dynamic Bayesian approaches use sequences of states to boost recognition. Temporal logic approaches encodes temporal information such as situation sequences, duration, overlaps and intervals between events to enhance recognition. Within the scope of this thesis, our approach will aim to incorporate duration and time of occurrence of a situation into our Dempster-Shafer approach. Inclusion of other types of temporal information such as sequences and overlaps into our approach is desirable as indicated by its usefulness in other approaches [5, 56], but is outside the scope of the work in this thesis.

- Fuzzy sets are applicable if context descriptions are imprecise, and may be incorporated into with other reasoning schemes; We will use fuzzy sets to combine sensor quality with Dempster-Shafer theory, as described further in Chapter 4.

- Learning schemes are a useful technique for situation recognition where training data is available. They implicitly deal with uncertainty by blindly absorbing uncertainty of sensor and rule information into the probabilistic model. However, if training data is an issue, Dempster-Shafer theory is the only specification-based approach that has strong theoretical support for processing uncertainty in an environment.

Table 2.1 summarises the requirement for domain knowledge versus training data, and the functionality of the various techniques.

Our particular interest is in a Dempster-Shafer based approach because it is not reliant on training data, but can cater for uncertainty. Existing work using Dempster-Shafer theory does not address the dynamic nature of situation or the inclusion of sensor quality. In the next chapter, we describe the theoretical foundations for our evidence based reasoning approach.

CHAPTER

THREE

Creating evidence decision networks using Dempster-Shafer theory

In the previous chapter, we examined approaches to situation recognition. We will use Dempster-Shafer theory as the basis of our approach. To do this, we need to describe the process by which sensor data will be processed as evidence, and distributed across situations.

The purpose of this chapter is to explain the concepts of Dempster-Shafer theory and how we apply it to reasoning about situations. We define the evidence processing needed to determine situation occurrence based on sensor evidence, using a variety of evidential operations. In Section 3.2, we describe the basic elements of Dempster-Shafer theory: frames of discernment, mass functions, evidence fusion and sensor discounting. In Section 3.3, we explain how we document the relationships between sensors and situations using our own diagramming technique, situation Directed Acyclic Graphs (DAGs). We also explain the architecture of the evidence decision network. Section 3.4 explains the evidential operations that will be required to process evidence for belief distribution and decision making. These operations consist of operations from basic Dempster-Shafer theory, extensions to the theory by other researchers, and our own new operations where none exist to meet our requirements. Issues with evidence combination are discussed in Section 3.5, and our solution to these issues are explained in Section 3.6. A summary of the finalised set of evidence operations to support evidence based reasoning is presented in Section 3.8.

3.1 Introduction

The term evidence theory is used interchangeably in the literature with Dempster-Shafer theory. Dempster-Shafer theory concepts was originally introduced by Arthur Dempster [23, 22] and refined by Glen Shafer[95]in the 1970s. Since then, various extensions to Dempster-Shafer theory have appeared in the literature to cater for alternative scenarios. These are usually referred to under the term of Dempster-Shafer theory. An exception to this is Smet's [100] transferable belief model (TBM), which is described further in section 3.4.3. We will use the term Dempster-Shafer theory to refer to evidence theory, but we will distinguish where we use aspects of Smet's TBM.

3.2 Concepts of Dempster-Shafer theory

Dempster-Shafer theory combines separate pieces of information (evidence) to calculate the belief in an event. Its key features are (1) its ability to specifically quantify and preserve ignorance, (2) its facility for assigning evidence to combinations of choices - such as user in "'kitchen OR bedroom" as well as singletons (unlike probability theory which must allocate probability to singletons), and (3) its use of domain knowledge as a method for belief distribution [37]. These features are relevant to situation recognition in context aware systems for several reasons:

- Sensors are unreliable; an ability to quantify this lack of reliability and preserve the resulting uncertainty will support the quantification of situation uncertainty;
- Rules are uncertain, and this uncertainty can be used to contribute to situation uncertainty calculations;
- The theoretically sound basis for incorporating domain knowledge offers us a way to encode knowledge without relying on training data.

Each evidence source has a total available amount of belief to be allocated, totalling to a value of 1. The mass function for each evidence source allocates a source's belief across a set of choices. These choices are collectively called the Frame of Discernment. In this section, we describe the basic concepts of Dempster-Shafer theory: frames of discernment, mass functions, rule of evidence combination and discounting.

3.2.1 Frames of Discernment and mass functions

In a Dempster-Shafer theory reasoning scheme, the set of possible hypotheses are collectively called the *frame of discernment*. This frame Ω represents the set of choices $\{h_{1,}h_{2,}...h_n\}$ available to the reasoning scheme, where sources (such as sensors) assign belief or evidence across the hypotheses in the frame. Hypotheses can be any subsets of the frame. i.e. to singletons in the frame or to combinations of elements in the frame. For example, a calendar sensor that monitors whether a user is scheduled to be in a meeting or not assigns belief across a frame of discernment that includes hypotheses *{meeting, coffee break, busy at desk}.* When the calendar indicates that the user does not have a meeting, belief is assigned by the calendar sensor to 'not meeting' situations i.e. the combination of *{coffee break, busy at desk}*. It assigns zero belief to *{meeting}*.

Formally, 2^{Ω} denote the set of all subsets of Ω to which a source of evidence can apply its belief. The function $m : 2^{\Omega} \rightarrow [0, 1]$ is called a mass function that defines how belief is distributed across the frame, if the function satisfies the following conditions, for hypotheses A:

$$m(\phi) = 0 \tag{3.1}$$

$$\sum_{A \subseteq \Theta} m(A) = 1 \tag{3.2}$$

Based on these conditions, belief from an evidence source cannot be assigned to an empty or null hypothesis, and belief from the evidence source across the possible hypotheses (including combinations of hypotheses) must sum to 1, similar to probability theory. The least informative evidence (uncertainty) is the assignment of mass to a hypothesis containing all the elements $\{h_1, h_2, ...h_n\}$, because this evidence does not commit to any particular hypothesis. This 'uncertainty' is denoted by the symbol Θ.

3.2.2 Combining evidence

A crucial part of the process of assessing evidence is the ability to fuse evidence from multiple sources. In Dempster-Shafer theory, the combination of evidence from two different independent sources is accomplished by Demp-

ster's combination rule:

$$m_{12}(A) = \frac{\sum_{X \cap Y = A} m_1(X).m_2(Y)}{1 - \sum_{X \cap Y = \phi} m_1(X).m_2(Y)} \tag{3.3}$$

where $m_{12}(A)$ is the combined belief for a given hypothesis A, and X and Y represent all possible subsets of the frame. The numerator in equation 3.3 represents evidence for hypotheses whose intersection is the exact hypothesis of interest, A. i.e. the agreement across the two sources about hypothesis A. The denominator, $1 - K$ is a normalisation factor, where *K* is a *conflict* factor representing all combined evidence that does not match the hypothesis of interest, A. The value of conflict, *K*, when combining evidence is indicative of the level of disagreement amongst the sources of their belief in hypothesis A.

Dempster's rule can be considered as a strict AND operation of the evidence sources [92]. An alternative will be required to cater for where sources are combined as OR scenarios.

3.2.3 Sensor discounting

Shafer defined an evidential operation for discounting sensor evidence [95]. When an evidence source is known to be less than 100% reliable, a discounting factor between 0 and 1 is applied to the source's beliefs. Unused belief as a result of discounting is assigned to uncertainty. If a source is completely reliable ($r = 0$) discounting has no effect. For example, a location sensor identifies a user's location with belief distribution of 0.3 in Room A, and 0.7 in Room B or C. If a discount factor of 0.2 is applied (i.e. the sensor is 80% reliable), the belief assignments are discounted and the leftover discounted belief is assigned to uncertainty. The re-distributed belief is then $RoomA = 0.24, RoomB \vee C = 0.56, \Theta = 0.2$. The impact of the discounting factor on beliefs is represented formally by Lowrance as [71] as follows:

For a discount factor, d, where $(0 \leq d \leq 1)$, where Θ represents uncertainty:

$$m_d(A) = \begin{cases} (1-d)m(A) & \text{if } A \neq \Theta \\ d + (1-d)m_d(\Theta) & \text{if } A = \Theta \end{cases} \tag{3.4}$$

3.2.4 Example of evidence combination

We use a worked example based on our own intelligent office data set to explain the concepts of mass functions, frames of discernment and evidence combination.

Two sensors are used to detect user location in an office. The locations of interest to an application are the cafe, the user's desk, the meeting room and 'anywhere else' in the building. Each of our sensors are capable of discerning these locations. The frame of discernment for each of the sensors includes the singleton hypotheses $\{desk, cafe, meetingRoom, other\}$. It also, in theory, includes all possible combinations of the singletons, such as$\{desk \wedge cafe, desk \wedge cafe \wedge other\}$ and the complete ignorance hypothesis $\{desk \wedge cafe \wedge meetingRoom \wedge other\}$ which is represented as Θ. The first sensor detects the user's location in the cafe. The sensor is 70% reliable, so its belief is assigned across the frame as $\{cafe\, 0.7,\ \Theta\, 0.3\}$. The allocation of 0.3 to ignorance is generated by discounting the sensor evidence by 70%, as calculated using equation 3.4. The second sensor has conflicting evidence, assigning its belief across the frame as: $\{meetingRoom\, 0.2,\ desk \wedge cafe \wedge other\, 0.6,\ \Theta\, 0.2\}$.

Combining those beliefs using Dempster's rule of combination in equation 3.3, the calculations are shown in table 3.1. Prior to normalisation, the unnormalised masses as captured in the combined evidence from table 3.1 are:

$mass_{Cafe} = 0.42 + 0.14 = 0.56$;

$mass_{meeting} = 0.06$;

$mass_{desk,cafe,other} = 0.18$;

$mass_{\Theta} = 0.06$;

$conflict = 0.14$

The conflict of 0.14 represents the conflicting evidence from the sensors which has no overlap (i.e. desk versus meeting).

To normalise out conflict, the normalising factor is $k = 1 - 0.14 = 0.86$

The revised masses after normalisation, using equation 3.3 are:

$mass_{cafe} = 0.65$;

$mass_{meeting} = 0.07$;

Mass Assignment from sources	Cafe 0.7	Uncertainty 0.3
Desk, cafe, other 0.6	Cafe 0.42	Desk, Cafe, Other 0.18
Meeting 0.2	Conflict 0.14	Meeting 0.06
Uncertainty 0.2	Cafe 0.14	Uncertainty 0.06

Table 3.1: Evidence combination example

$mass_{desk,cafe,other} = 0.21$;

$mass_{uncertainty} = 0.07$

3.3 Evidential approach for situation recognition

In order to apply Dempster-Shafer theory to situation recognition, we will apply evidential operations that will propagate sensor evidence through a hierarchy of higher level context states. For each environment, these operations will be applied using an *evidence decision network* that reflects the sensors, context values and situations specific to that environment. We will be incorporating knowledge about the situation features identified in Chapter 2: the situation specification, temporal knowledge, situation hierarchies and known uncertainties, such as sensor quality and inference rule frequencies. We want to capture this knowledge in a clear, unambiguous way that can be used by system developers and designers when creating the evidence decision network. To do this, we use a Directed Acyclic Graph (DAG), which we term a situation DAG, to document this knowledge and to annotate the evidential operations that will be needed. Once complete, the evidential operations required to propagate evidence can be read from the DAG.

3.3.1 Situation DAGs

The situation DAG captures knowledge about the environment that is relevant to the evidential reasoning process: sensors, sensor quality, abstracted context, inference rules, temporal information and situation hierarchies. The notation for the situation DAG is shown in figure 3.1. Sensors are the root nodes at the base of the diagram. Sensor readings are abstracted or mapped to more

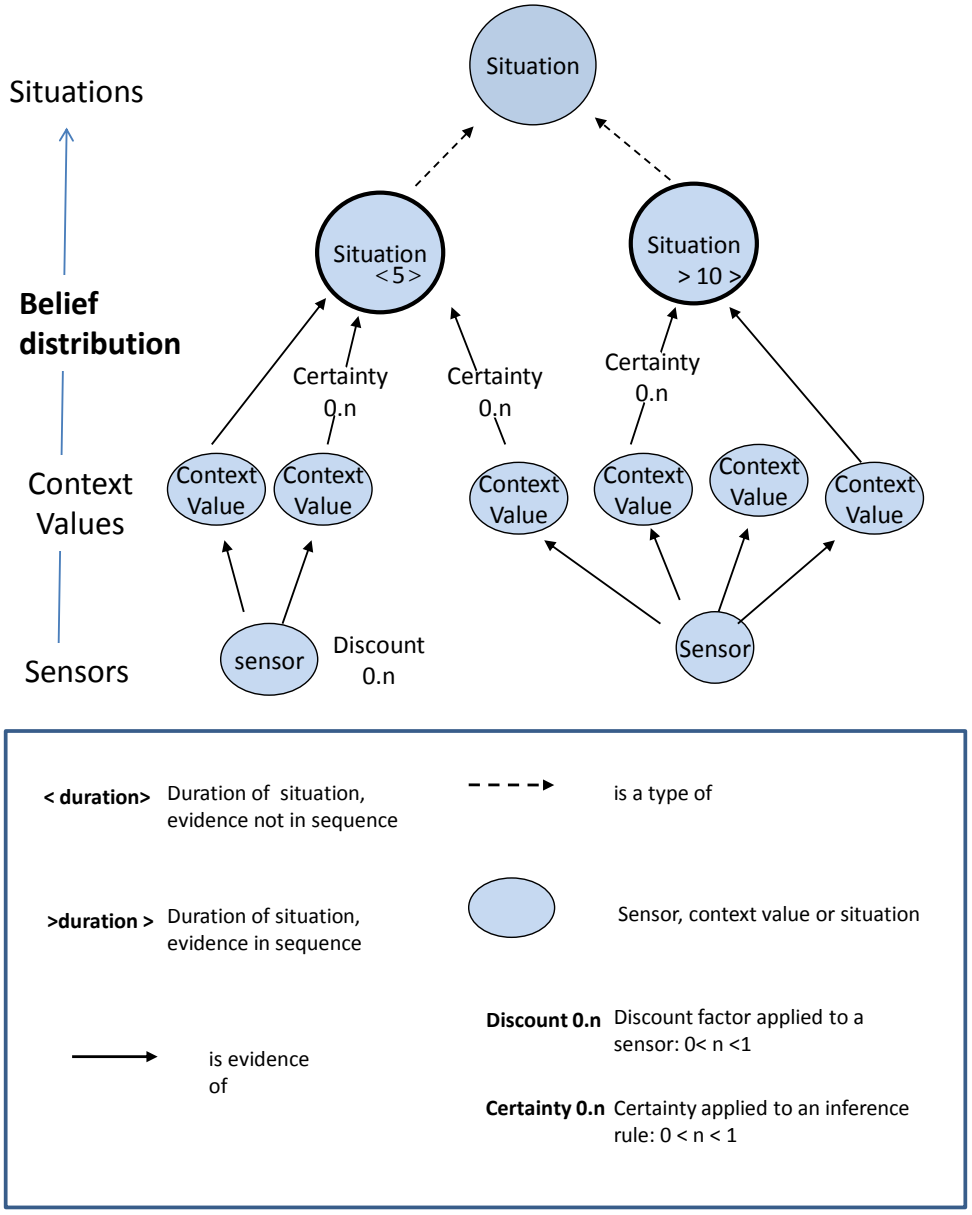


Figure 3.1: Situation Directed Acyclic Graph

human-understandable context values. A context value is meaningful information about an entity, abstracted from sensor data, that is relevant to detecting situations. Moving up the hierarchy, each context value will be mapped to one or more situations, indicating that the occurrence of a particular context value 'is evidence' of the situation occurring; i.e. an inference rule. Depending upon the complexity and range of situations in the system, higher-level situations may also be inferred from lower level situations. Each solid directed arrow in the graph is interpreted as 'is evidence of''.

At sensor level, if a sensor is known to have a particular quality or reliability, this quantified reliability is included as a sensor discount, as described previously in section 3.2 . Inference rules frequency or uncertainty are also denoted on the DAG. For example, in the home data set that we will examine in Chapter 5, a user 'sometimes' uses the microwave when preparing breakfast. This is quantified as 40% of the time, by examining sample occurrences of the 'prepare breakfast' situation in the data. Therefore, a belief of 0.4 is applied along the edge or inference rule from the context value 'microwave used' to the situation of 'prepare breakfast', with the remaining 0.6 assigned to uncertainty.

A summary situation is one that generalises two or more lower level situations. For example, a 'leave house' situation may be detected by either 'front door used' or 'all sensors quiet'. We denote this on the graph using the 'is a type of' notation. If either lower level situation is detected as occurring, the summary situation is also occurring. The set of situations that are being inferred by a situation recognition process are shown as bolded on the situation DAG.

If the time duration of a situation is used as evidence, this is denoted on the DAG in the situation node. The use of time durations as evidence is one of our extensions to Dempster-Shafer theory and is described in detail in Chapter 4.

Looking at the left hand side of the DAG, we term the transfer of evidence from sensors up to situations 'belief distribution'. Once belief has been distributed to all nodes in the DAG, a decision step is required to determine which situations(s) are occurring.

3.3.2 Evidence decision network architecture

We separate the situation recognition process into two steps 1) belief distribution and 2) decision making.

Belief distribution: This step populates belief levels throughout the situation

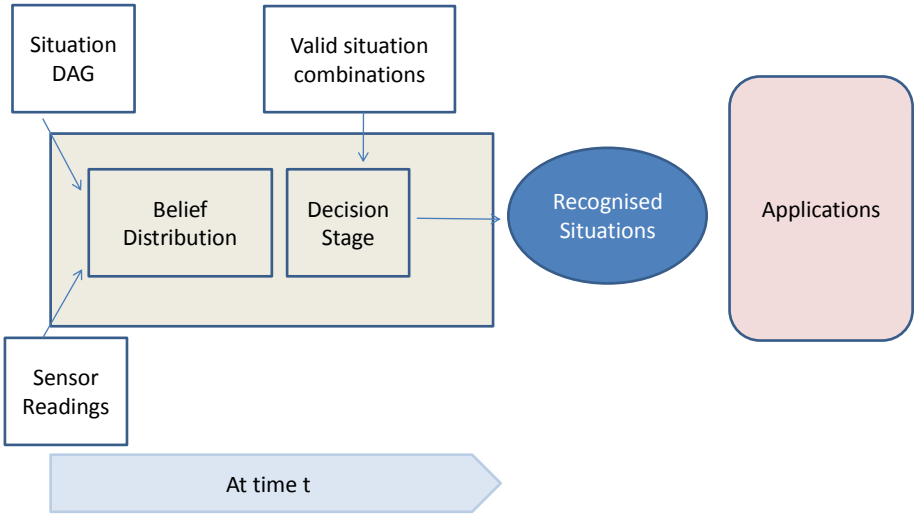


Figure 3.2: Evidence decision network architecture

DAG to all nodes. At regular time intervals, mass functions for all sensors distribute mass to context values for a set of sensor readings. Evidence is fused and propagated to higher levels on the DAG until every node on the DAG has an assigned belief level. The architecture for the evidence decision network is shown in figure 3.2.

Decision making: Once evidence has been propagated to all nodes in the DAG in the belief distribution stage, a decision process is then required to select the recognised situations based on the distributed belief. The first step is to distribute any evidence on combined elements (i.e. to two or more situations), so that all evidence is allocated to singleton elements or situations. This is achieved using our variant of Smets decision rule as described in equation 3.12. Once all evidence has been distributed, the decision algorithm is driven by the recognition requirements for the environment:

1. What situations are being recognised in the environment? The DAG denotes all situations, where only a subset may be used to drive application behaviour. The set of situations that are being inferred are shown as bolded on the situation DAG

2. Can situations co-occur or can only one situation occur at a time?

3. If situations can co-occur, what are the invalid situation occurrence combinations?

In an environment where a single situation only can be occurring, the situation

with the highest belief is deemed to be occuring. If situations are allowed to co-occur, a belief threshold level will be required as a cut-off. If any invalid co-occurrences of situation exist (e.g. making breakfast and having a shower at the same time), these will be incorporated into the threshold check. The algorithms for decision making are described in detail in Chapter 6, section 6.1.

In the next section, we explain the detailed evidential operations required to support belief distribution.

3.4 Evidence operations for belief distribution and decision making

The evidential operations that will be required to infer situations in an environment can be determined from the Situation DAG established for that environment. The basic premise is that at regular intervals of time t, sensor readings will be propagated from sensors up to situations, providing belief levels for the possible situations. The situation(s) with the highest belief are believed to be occurring. The detailed operations are as follows:

- Each node must be placed in a *frame of discernment* so that belief is assigned to it
- *Mass functions* interpret sensor readings and assign evidence across context values for the sensor;
- Sensor evidence is *discounted* if sensor is unreliable
- The evidence for each context value needs to be transferred or *propagated* towards the next level of the DAG.
- When nodes below each situation have been populated with evidence, the evidence from each node must be fused using an *appropriate fusion rule*, to determine belief in the situation. The fusions should cater for 'AND', 'OR' and 'is a type of' fusion.
- Dempster-Shafer theory allows evidence to be assigned to groups of elements in the frame. Evidence may therefore be assigned to combinations of situations. When deciding which situation is 'occurring', we need to allocate out this combined evidence to single situations.

Evidence operation	Description	Source
Belief distribution		
Mass functions	Distribute sensor belief to context values	basic Dempster-Shafer theory
Sensor discounting	Reduce sensor evidence by discount factors	basic Dempster-Shafer theory
Evidence propagation	Move evidence from context values up to higher levels in the DAG	*required*
AND	Merge evidence from two or more sources	basic Dempster-Shafer theory: Dempster's rule of combination
OR	Combine evidence from two or more source as 'OR'	*required*
is a type of	Combine evidence for a summary situation	*required*
Decision making		
distribute combined evidence to single situations	Evidence that is shared across multiple situations has to be split into single situations as part of decision making	*required*

Table 3.2: Evidential operations for belief distribution and decision making

Table 3.2 shows how each of these requirement is met:

Basic Dempster-Shafer theory provides some of the evidential processing needed: mass fusions, discounting and basic evidence fusion. As shown in the table, we also need additional evidential operations to complete the process of situation recognition. We need to: (1) propagate evidence from sensors up to higher levels in the situation DAG; (2) fuse evidence in OR combinations (3) process 'is a type of ' situation hierarchies and (4) distribute combined evidence using some form of decision making operation. In the next subsections, we explain how each of these four requirements is met from existing Dempster-Shafer theory and by our own defined evidential operations where none exist to meet these requirements.

3.4.1 Evidence propagation across frames

We want to propagate belief of context values up to the remaining nodes in the DAG. Lowrance *et al*. [71] described a set of evidence operations to allow mass from evidence sources to be propagated from one frame to another frame. This matches our requirement to propagate evidence at context event

level upwards to situations. We use Lowrance's operations for evidence propagation, as also used by Hong *et al*. [50] in their evidential model: *compatibility relations*, *compatibility mapping* and *translation*. We also define an additional operation to translate evidence between frames when the relationship between elements is uncertain.

- *Compatibility relations* define which elements from two frames of discernment can be true simultaneously. We use them to define the exact destination in higher level frames of discernment to which belief from sensors will be transferred. For example, a fridge sensor may have a frame of context values: *{fridgeUsed, fridgeNotUsed,* Θ*}*. The use of the fridge is indicative of the 'get drink' situation, which has a part of a frame of discernment: *{getDrink; notGetDrink,* Θ*}*. '*FridgeUsed*' is compatible with '*get drink*' (i.e. they are both true simultaneously) and so on for the remaining elements in both frames. Defined formally by Lowrance *et al*. [71], the compatibility relation between frames Ω_A and Ω_B is a subset of the cross product of the two frames. A pair (a_i, b_j) is included if and only if they can be true simultaneously. There is at least one pair (a_i, b_j) included for each a_i in Ω_A :

$$\Omega_{A,B} \subseteq \Omega_A \times \Omega_B \tag{3.5}$$

- *Compatibility Mapping*: Using the compatibility relation $\Omega_{A,B}$, Lowrance *et al*. [71] define a compatibility mapping $C_{A \to B}$ for translating belief expressed relative to Ω_A to statements relative to Ω_B. Belief for a statement A_K indicates belief for statement $C_{A \to B}(A_K)$. Continuing with the '*fridgeUsed*' example, there is a compatibility mapping between '*fridgeUsed*' and '*get drink*'.

- *Translation*: Lowrance then defines how to transfer mass between frames for which compatibility mapping has been defined. We use translations along the inference paths defined by compatibility mapping, enabling the mass of compatible elements to be transferred; e.g., mass of '*fridgeUsed*' will be propagated to '*get drink*'. To translate M_A from frame Ω_A to frame Ω_B via compatibility mapping $C_{A \to B}$, the mass translated is defined formally by [71] as follows:

$$M_B(B_j) = \sum_{C_{A \to B}(A_i) = B_j} M_A(A_i) \tag{3.6}$$

- *Translating uncertain relationships*: The operations of defining compatibility relations, mapping and translating evidence allows evidence to move from one frame to another, but it assumes that there is a certain relationship in the compatibility relations (e.g. that '*fridgeUsed*' is *always* indicative of '*get drink*'). If uncertainty exists in the relationships, we will need to refine equations 3.6 to process uncertainty. For example, if '*fridgeUsed*' is indicative of '*get drink*' 60% of the time that the fridge is used, the relationship has a frequency or certainty of 0.6. Therefore 0.6 of the '*fridgeUsed*' belief will transfer to '*get drink*'. The remaining 0.4 of the '*fridgeUsed*' belief will be allocated to uncertainty in the '*get drink*' frame. We represent this formally as follows: to translate M_A from frame Ω_A to frame Ω_B via compatibility mapping $C_{A \to B}$, where the compatibility mapping $C_{A \to B}$ has a certainty $0 \leq \omega \leq 1$,

$$M_B(B_j) = \sum_{C_{A \to B}(A_i) = B_j} M_A(A_i).\omega_i \tag{3.7}$$

The resultant uncertainty in frame Θ_B as a result of translation is as follows:

$$M_B(\Theta_j) = \sum_{C_{A \to B}(A_i) = B_j} (1 - \omega_i) \tag{3.8}$$

3.4.2 Combining evidence as 'OR' and 'is a type of'

Evidence may need to be combined under 'OR' conditions. For example, a user may be assessed as 'in meeting' if the user's location is assessed as 'meeting room 1' OR 'meeting room 2'. On the DAG, this is denoted using a 'meeting room' node, with both meeting rooms notated as 'is a type of' meeting room. We process this "OR" scenario in two ways:

1. If the lower nodes (i.e. choices) are in the same frame of discernment, the belief of the upper node is calculated as the sum of the two lower nodes. This is because their belief is derived from the same evidence sources because they are in the same frame. Therefore, their belief can be combined as follows:

$$m_{sum} = \sum m_{lower} \tag{3.9}$$

2. If the lower nodes are in different frames of discernment, we select the belief of the highest node. The evidence operation used is the

max operator. We select the maximum belief of the lower level nodes, $m_{lower_1} - - - m_{lower_n}$, from which the summary node is comprised, in the same way as done by Hong *et al*. [49]:

$$m_{sum} = max(m_{lower_1}, ..., m_{lower_n}) \tag{3.10}$$

3.4.3 Distribution combined evidence to single situations

Once all belief has been fused and propagated, we then need to assess the beliefs against the choice of situations under assessment in the environment. Intuitively, we will select the situation(s) with the maximum belief. If our goal in the environment in question is to recognise single situations (e.g. eating, showering) as opposed to combinations (eating or showering), then any belief that is assigned to *combinations* of situations needs to be re-distributed to individual situations in order to support the decision process.

Smets [98] defined a decision level for processing of evidence as part of his Transferable Belief Model (TBM). The TBM is similar to Shafer's evidence theory but which allows for 'open world' scenarios where hypotheses that are not in a frame may be considered [100]. Smets defines a pignistic probability as a probability that a rational person will assign to an option when required to make a decision. He uses pignistic probabilities to distribute belief at the decision level of a belief model [96]. To redistribute belief that is assigned to multiple elements (i.e. combinations of situations), Smet distributes the belief across each of the hypotheses based on the number of elements in the hypothesis. He also redistributes belief assigned to uncertainty across the hypotheses. Defined formally [98], the value of the pignistic probability for A, where A may occur in one or more sets (hence sum operator) is calculated as:

$$Bet(A) = \sum_{X \subseteq \Omega} \frac{|A \cap X|}{|X|} . \frac{m(X)}{1 - m(\Theta)} \tag{3.11}$$

For our model, we will use uncertainty as part of our situation selection algorithm as described in Section 6.1. Therefore, we do not wish to normalise out uncertainty as done by Smets. We provide a modification to Smet's pignistic probabilities that does not re-distribute uncertainty. Our re-distribution of belief to smaller hypotheses (A), based on A's occurrence in one or more sets, X, is calculated as:

$$M_{decision}(A) = \sum_{X \subseteq \Omega} \frac{|A \cap X| \, .m(X)}{|X|} \tag{3.12}$$

3.5 Evidence fusion issues

Dempster's rule of combination, as described in equation 3.3, was the core fusion mechanism provided as part of the original Dempster-Shafer theory. This rule combines sources as an 'AND' configuration. The rule has been enhanced or changed by researchers in order to cater for specific applications of Dempster-Shafer theory because it is not suitable in all fusion scenarios. Generally, these modifications are motivated by two problems. The first problem is how to treat conflict when sources disagree. Dempster's rule can result in unexpected results when fusing conflicting evidence. This problem, termed Zadeh's paradox [124] is described further in section 3.5.1. The second problem is how to cater for evidence sources of varying trust, so that the credibility of sources is correctly reflected in the fusion result.

Various fusion rule modifications exist to cater for these problems. Yager's [117] modified Dempster rule provides an alternative to the treatment of conflicting evidence. Unlike Dempster's rule, evidence in agreement is not normalised by a conflict factor, and conflict is stored separately. This gives a more transparent picture of conflict, so is useful in domains where conflict is critical, such as medical diagnosis. Dubois and Prade [33] defined a disjunctive fusion rule, where at least one source of information is telling the truth, but not all sources are trustworthy. The weakness of this rule is that if belief from a single source is 0, the fusion of all sources will be 0. This is not a credible outcome in a pervasive environment if a sensor breaks down. Murphy's combination rule [76] eliminates the possibility of a single source dominating all other sources and is described further in section 3.6.1. Smet's transferable belief model assumes that if sources conflict, such conflict can be assigned to other undefined options outside the frame of discernment - i.e. an open world assumption [99]. This is not suitable for pervasive systems if a closed situation group is under detection. Other modifications of Dempster's rule that cater for varying scenarios of conflict and trustworthiness of sources are Dubois and Prade's exclusive and mixed disjunctive rules [24] and Inagaki's Unified combination rule [92].

A final problem that existing modifications do not target is the issue of ev-

idence spread over time. Dempster's rule assumes that all evidence is co-occurring. We wish to cater for evidence that may not be occurring at the same time, as described further in Chapter 4.

In the next section, we explain three particular problems for situation recognition when we apply Dempster's rule in our evidence decision network environment: Zadeh's paradox, single sensor dominance and evidence spread over time. We then explain how we aim to address these in our evidence decision network by selecting two alternative fusion rules *Murphys* and *averaging*.

3.5.1 Zadeh's paradox

Zadeh's paradox is a well-documented problem with Dempster's rule of combination [92]. Zadeh highlighted the fact that when sources in high conflict are combined using Dempster-Shafer rule, the results can be completely counter intuitive [124]. Using the example illustrated in figure 3.3, two location sensors assign belief across a frame *{room a, room b, room c}*. The first sensor assigns a belief of 0.99 that the correct location is *{room a}* and 0.01 belief to *{room b}*. The second sensor disagrees, assigning most of its belief 0.99 to *{room c}* and 0.01 to option *{room b}*. The sensors are almost completely conflicting, with a small degree of overlap. When the sensor masses are fused using Dempster rule of combination in equation 3.3, room b obtains all belief, with zero belief assigned to rooms a or c. This is because room b is the only option on which both sensors overlap, and all disagreeing evidence is normalised out. One reason to account for this is the possibility that both sensors are correct [42]. This possibility will work for domains such as medical diagnosis where two diseases can be detected together. But it does not work with context detection such as location, where a person can only be in one place at one time. Another possibility is that one source is not reliable [42, 99, 64], a possibility that we address by enabling rich sensor quality knowledge to be included as described in Chapter 4. A final possibility [99, 64] is the open world assumption in that both sources are wrong because choices outside the frame may exist. This is not a workable explanation in a pervasive environment where a closed world of possible values (i.e. a known set of situations is being detected) is a reasonable assumption - unlike the medical domain where diseases outside the suggested possibilities may be possible. The most correct treatment in our opinion is to highlight an even distribution of belief for both room a and room c, with a small degree of belief for room b (and a large measure of conflict).

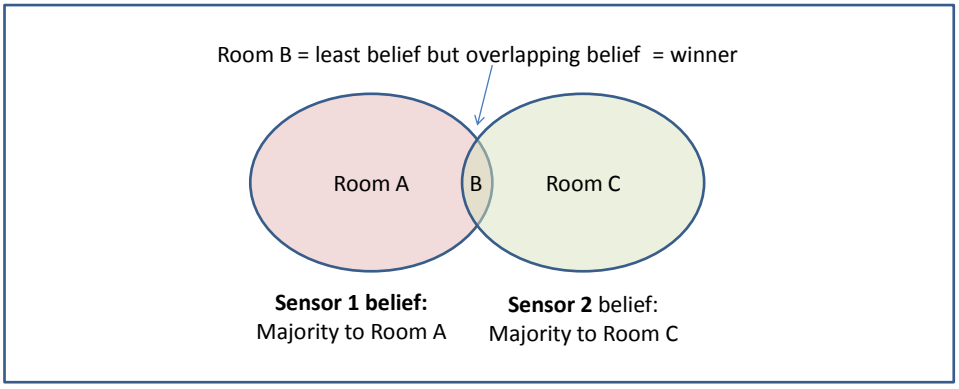


Figure 3.3: Zadeh's paradox: scenario of location sensors in conflict on room location

3.5.2 Single sensor dominance

A second problem that has gained far less attention in the literature is the potential dominance of a single sensor. Murphy [76] described how a single disagreeing sensor can overrule multiple other agreeing sensors in the fusion process. A categorical belief function is where all belief is assigned to one hypothesis in a frame [67]. For example, if five sensors are used to determine the location of a user in the house, a single categorical sensor that assigns all of its belief to a contradictory option will negate the evidence from the other four sensors. We suggest that this is particularly problematic for binary sensors which are increasingly being used in Smart Home deployments. Binary sensors have small frames of discernment, with just three states: *{on, off , ignorance}*. Unless discounted, they will categorically assign all of their belief to the 'on' or 'off' states. A single malfunctioning binary sensor can in theory therefore overrule evidence from other correct binary sensors during the fusion process. A more intuitive result would be to allow the agreeing sensors to 'win' but to represent the disagreeing sensors' evidence as conflict.

3.5.3 Evidence spread over time

A third problem that we have observed occurs when sensor evidence of a higher level state is spread over time. For example, the detection of a breakfast activity by the triggering of a fridge sensor, then a kettle sensor, then the toaster sensor and so forth. At any point in time, only one of the sensors may be "on", so fusion of all the sensor values at any point in time may result in the 'on' sensor evidence being lost. The fusion rule should capture that some

evidence of the situation was observed even though it has been greatly contradicted by sensors that are off. It should not be wiped out by the overruling of the contradictory sensors, as will occur with Dempster's rule of combination.

3.6 Alternative combination rules

In this section, we will describe two alternative combination rules to Dempster's rule of combination, that have appeared in the literature [76] that partially address the problems highlighted.

3.6.1 Murphy's combination rule

Murphy [76] proposed an alternative rule of combination that will eliminate the dominance of a single sensor and allow contradictory evidence to be preserved to some degree. Evidence is averaged prior to combining it using Dempster's rule of combination. Formally, if there are n sources of evidence, we use equation 3.3 to combine the weighted averages of the masses *n-1* times. Evidence for each hypothesis, h, from n sources is summed, and averaged across all evidence sources. This eliminates the dominance of a single sensor by reducing its contribution according to the number of sources. Use of Murphy's combination rule will also eliminate Zadeh's paradox [124] because the evidence is averaged prior to combination. In our work, we use Murphy's rule for fusion of evidence instead of Dempsters's original rule of combination. We compare the results from the use of both in the evaluation in Chapter 6.

3.6.2 Averaging Rule

In his original work [95], Shafer combined belief functions by averaging all the evidence for each hypothesis (instead of the combination rule), as follows:

$$M(A) = \frac{1}{n}(M_1(A) + ... + M_n(A)) \tag{3.13}$$

Averaging can be used to eliminate the influence of any strongly conflicting single belief [95] so would cater for both single sensor dominance and Zadeh's paradox. The use of averaging provides an accurate record of contributing beliefs because no belief is 'lost', but it lacks convergence. Both Dempster's and

Murphy's rule allows evidence from sources that are in agreement to reinforce each other, and disagreeing evidence to be dropped. In contrast, averaging does not increase the measure of belief in the dominant subset but provides a less conclusive picture because conflict is not normalised out. However, it is simpler to compute with less calculations. We anticipate that averaging will be useful to counteract the expected problem of conflicting sensors in binary sensors.

3.6.3 Summary of fusion issues

Of the two alternative rules (Murphy's and Averaging) and Dempster's original rule, Murphy's combination rule appears to have the benefits of considering all sensor evidence, allowing evidence convergence and presenting a fair spread of the evidence. In our evaluation in Chapter 6, we will test Dempster's, Murphys and the Averaging rule in order to determine which results in greatest situation recognition accuracy and why.

All of the three rules assume that evidence is co-occurring, so they do not in their current form provide a solution to evidence that is spread over time. We will address the issue of time-spread evidence in Chapter 4.

3.7 Evidence decision network - summary of operations

All evidential operations to distribution belief and support decision making are now defined. The operations are derived from original Dempster-Shafer theory, modifications by other researchers, and our additions when no existing operation catered for our evidence processing requirements. The evidence operations, their source and purpose are summarised in table 3.3.

3.7.1 Evidence example

The example situation in figure 3.4 illustrates how evidential operations are applied. This is a simple situation of "get drink", detected from two pieces of evidence: 'fridge used' and 'cup used'. Sensor mass functions translate the sensor readings into belief across each sensor frame of discernment. The

Evidence operation	Description	Source	Equation
Belief distribution			
Mass functions	Distribute sensor belief to context values	Shafer	3.1, 3.2
Sensor discounting	Reduce sensor evidence by discount factor	Shafer	3.4
Evidence propagation: 1) Compatibility relations 2) Compatibility mapping 3) Translation 4) Uncertain translation	Move evidence from context values up to higher levels in the DAG by mapping compatibility across frames	1) Lowrance *et al.* 2) Lowrance *et al.* 3) Lowrance *et al.* 4) defined for this thesis	3.5,3.6,3.7,3.8
AND:	Merge evidence from two or more sources	1) Dempster's rule (Dempster) 2) Murphy's rule (Murphy) 3) Averaging rule (Shafer)	(1) 3.3, (3) 3.13
OR and 'is a type of'	Combine evidence from two or more nodes as 'OR' or as 'is a type of' 1) nodes are in the same frame of discernment 2) nodes are in different frames of discernment	1) defined for this thesis 2) Hong *et al.*	3.9, 3.10
Decision making			
distribute to single nodes	Evidence that is shared across multiple situations has to be split into single situations as part of decision making.	Modified Smets rule defined in this thesis	3.12

Table 3.3: Summary of evidential operations for the evidence decision network

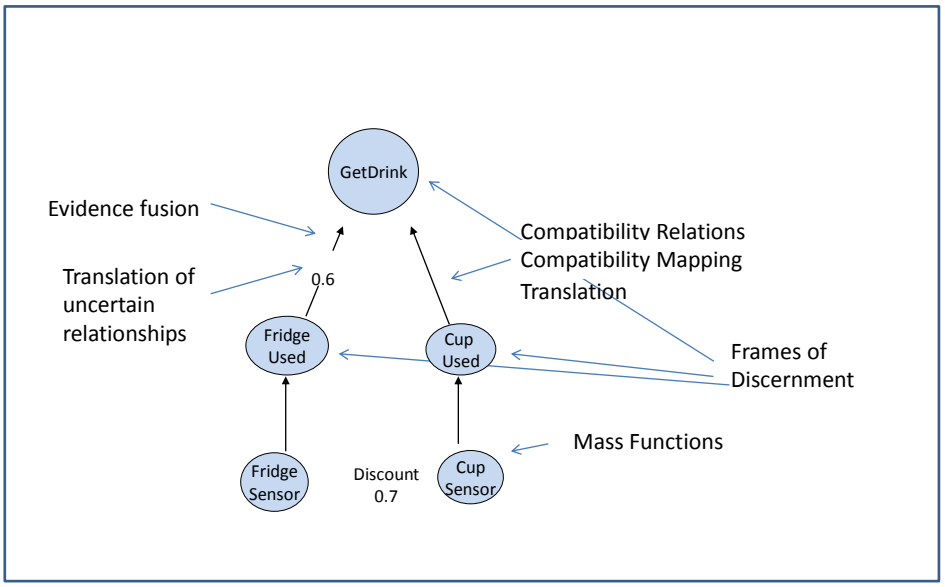


Figure 3.4: Evidence Example

frames of discernment for the fridge and cup sensors are *{fridgeUsed, fridgeNotUsed, uncertainty}* and *{cupUsed, cupNotUsed, uncertainty}* respectively. The cup sensor is known to be reliable 70% of the time, so 0.3 of its belief will be assigned to uncertainty. Once belief has been assigned to the sensor frames, this belief is propagated up to the situation frame using compatibility relations, compatibility mapping and translation. 'Fridge used' is compatible with 'get drink' (and conversely 'fridge not used is compatible with 'not get drink' although for simplicity the negative scenarios are not shown). 'Cup used' is compatible with 'get drink'. The belief from the sensor frames is then translated along the mapping to the 'get drink' situation, which has a frame of discernment *{getDrink, notGetDrink, uncertainty}*. The 'fridge used' context value has an uncertain relationship with the 'get drink' frame, as it is used 60% of the time. Therefore, the evidence will be translated using our modified translation for uncertain relationships in equations 3.7 and 3.8. Once the evidence has been translated to the 'get drink' frame, the belief from the two sources is fused using Murphy's combination rule. At this point, a decision process takes place to determine compare belief of 'get drink' with belief levels of other situations in the environment.

3.8 Conclusion

In this chapter, we presented the evidential operations needed for the creation of the evidence decision network. These operations are taken from the original Dempster-Shafer theory of evidence, existing extensions to the theory, and our three new operations (1) to translate uncertain belief from one frame to another (2) to process summary or OR scenarios for nodes in the same frame of discernment and (3) to distribute belief from combined elements to singleton elements using a modified version of Smets rule. The complete set of operations as summarised in table 3.3 provides us with a toolkit of operations from which we can construct an evidence decision network for an environment.

We described how we use situation DAGs to capture knowledge for an environment. The DAG drives the structure of the evidence decision network - enabling the identification of evidential operations to use for belief distribution and decision making for the environment in question. We see the DAG as a critical output of the systems design process, used as a specification to develop detailed evidence processing.

Three potential problems for the evidence-based approach have been preemptively described - Zadeh's paradox, single sensor dominance and time distributed evidence. We described two alternative combinations from Shafer and Murphy that will address both the Zadeh paradox and single sensor dominance. In the next chapter, we explain two additional features that we will incorporate into Dempster-Shafer theory in order to widen the breadth of knowledge that can be used to recognise situations: temporal knowledge (which will cater for evidence spread over time) and sensor quality knowledge.

CHAPTER

FOUR

Extending Dempster-Shafer theory with temporal and quality knowledge

In the previous chapter, we explained how Dempster-Shafer theory can be applied to a network of sensors, context values and situations, described by a situation DAG. We used basic concepts and variations of Dempster-Shafer theory to develop a structure that allows the distribution and assessment of evidence for situation recognition. In this chapter, we will explain two extensions to Dempster-Shafer theory that we have created in order to include additional knowledge to reason with: temporal extensions and quality extensions. Temporal extensions allow knowledge about situation time patterns to be included in the evidence gathering process. Quality extensions enable sensor performance and context uncertainty to be used to modify the strength of evidence originating from a sensor.

Section 4.1 describes how temporal knowledge can be treated as evidence. The nature of transitory evidence, and its relevance to higher level situations that have a time duration is explained in section 4.2. As part of this, we define the equations to extend sensor mass over a period of time, and how to fuse extended mass using Dempster's rule of combination. A worked example is provided in Section 4.2.3. In Section 4.5, we address how to to include sensor quality with evidential reasoning. We explain the nature of sensor and context quality information in Section 4.5.1. We then explain how quality is applied as part of the sensor mass function in Section 4.5.2. Notation of temporal and quality knowledge on the situation DAG is explained in Section 4.7.

4.1 Using temporal knowledge as evidence

Temporal knowledge is a natural human way to reason about current activities or situations. For example, when assessing the current activity of a person in the home, the time of day may determine whether they are preparing breakfast or dinner; the length of time they spent in the kitchen may help us decide whether they were preparing a meal or just getting a drink, and so on. Time durations of situations, sequential patterns in which situations occur and discernible patterns over time are examples of temporal knowledge that can improve our ability to recognise which situation(s) is occurring.

We hypothesise that using temporal features of situations in the evidence decision network will improve the accuracy of their recognition.In Chapter 2, we saw a variety of temporal information used in reasoning, including situation duration, sequences of events/situations, intervals between situations, overlaps, and relative times. In our work, we will aim to include situation duration and the time of situation occurrence in our Dempster-Shafer approach. This is not to say that the remaining temporal information will not be useful or possible to include at a future point in our approach - but they are outside the scope of our work at this point. We explain our reasons and approach for catering for situation duration and time of situation as follows:

(1) Transitory evidence: We use the term transitory evidence to refer to evidence that only occurs for part of the time duration of a situation [75]. Looking at a sample situation in figure 4.1, a 'preparing dinner' situation may typically endure for about 40 minutes, with indicative evidence from various sensors such as 'grocery cupboard used' and 'fridge used'. None of this evidence is necessarily occurring at the same time. At present, existing evidential approaches for activity recognition assume that evidence is co-occurring [50, 125]. In reality, evidence may be spread out over time, co-occurring or not, in no particular sequence and with no particular order expected. Events may co-occur and/or occur separately, with gaps between events, such as the examples shown in figure 4.1. The user opens the plate cupboard and fridge in the same sampling period, then uses the pans cupboard and freezer, then retrieves groceries. Such evidence for a higher level state that does not endure for the full time duration of the state is transitory evidence. We describe a mechanism for extending the lifetime of transitory evidence so that it lasts for the duration of the higher level state. This provides a stronger basis for detecting the higher level state than simply using individual sporadic 'events' in isolation.

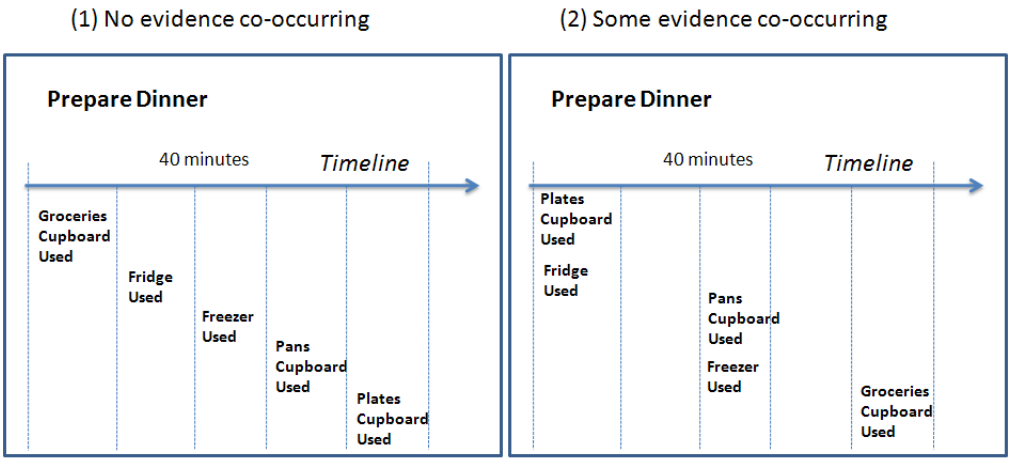


Figure 4.1: Transitory evidence for dinner situation

(2) Absolute time is the time at which a situation occurs [120]. This can be an exact timespan such as 'between 8:00 and 10:00" or a semantic description such as "morning".

In the next section, we explain how transitory evidence and absolute times are included in an evidence decision network.

4.2 Transitory evidence and situation durations

During the inference process, when evidence for that situation is detected, the duration for that situation is triggered to start. Looking at the situation of 'preparing dinner' in figure 4.2, when any of the groceries cupboard, fridge, freezer, pans cupboard or plates cupboard sensors are fired, the reasoning system will 'start' the dinner activity. The lifetime of the triggered sensor evidence for that activity will be extended to last for the remaining duration stored for that situation. As inference continues over time, the lifetime of any further evidence for the situation will be extended for the duration that is left of the situation (i.e. situation duration less elapsed time). Once the full duration of the situation is reached, the evidence will expire. By extending the lifetime of the evidence, at any point in time, the evidence sources can be fused as if they are co-occurring.

Evidence of a situation may also be evidential of other situations. Sensors that provide transitory evidence for more than one situation will trigger more than

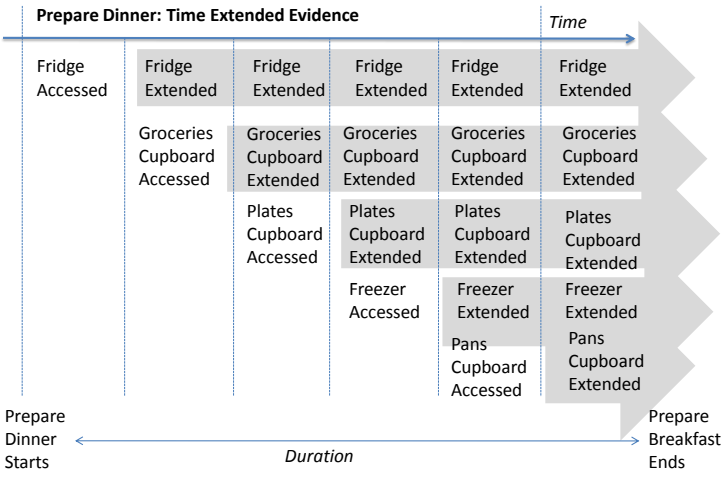


Figure 4.2: Time extension of evidence for 'preparing dinner' situation

one situation to start. Continuing with the meal preparing scenario, two other situations exist in this environment: 'preparing breakfast' typically lasts for 15 minutes and 'getting a drink' takes about 3 minutes. Both of these are also inferred from the fridge sensor. If the fridge sensor fires and none of these situations are in progress, the duration will kick off for 'preparing dinner', 'preparing breakfast' and 'getting a drink'. The 'fridge used' context value will have three separate lifetimes, one for each of the durations of the three situations. That is, the 'fridge used' context will expire after 3 minutes as evidence of 'preparing drink', after 15 minutes for 'preparing breakfast' and after 40 minutes for 'preparing dinner'.

If multiple simultaneous sensor events happen at the same time, where the events are evidential of different activities, the evidence is allocated to the relevant activity as per the situation DAG. For example, if a toaster sensor activates in the kitchen in the same sampling period as a sensor in the bathroom, evidence will be allocated to the 'preparing breakfast' and 'showering' activities respectively. The interpretation of these activities as co-occurring or not will be environment specific. If, for example, there are multiple inhabitants of the house, both 'breakfast' and 'showering' may be recognized as co-occurring as it is possible that two activities can happen at the same time. In this case, a belief threshold may be used to filter situations, with situations that have belief levels exceeding the threshold as 'occurring'. In an environment where activities can only occur one at a time, as in the case of the smart home dataset used in our evaluation, the activity with the greatest evidence (highest belief)

is deemed to be occurring.

4.2.1 Extending the lifetime of transitory evidence

To apply time extensions to transitory evidence, the mass function definition and the fusion rule for combining mass require alteration. Formally, a frame of discernment contains one or more hypotheses h_n, each of time duration $t_{dur(h_n)}$. Belief for a hypothesis from transitory evidence sources is assigned a time duration equal to the duration of the hypothesis. If the hypothesis has already been detected by earlier evidence and is 'in progress', the lifetime of belief assigned after the hypothesis duration starts is the remainder of the duration, $t_{rem(h_n)}$. The remainder is calculated as hypothesis time duration minus the elapsed time for that hypothesis: $t_{dur(h_n)} - t_{elapsed(h_n)}$. Therefore, for mass assigned to hypothesis, h_n, of time duration,$t_{dur(h_n)}$ at time t, the mass assigned to h_n at time t, will continue to exist for the remaining time of the hypothesis duration, $t_{rem(h_n)}$. Time extended mass, for hypothesis h_n,is represented as:

$$m_{t+t_{rem}}(h_n) = m_t(h_n) \tag{4.1}$$

where

$$t_{rem(h_n)} = t_{dur(h_n)} - t_{elapsed(h_n)}$$

To fuse extended mass, we use the evidence combination rule 4.2. To fuse evidence for two extended masses for enduring hypothesis h_n, fuse the evidence during the hypothesis occurrence at time $t + t_{rem(h_n)}$ using the extended mass values at time $t+t_{rem(h_n)}$. We substitute the masses in Dempster's combination rule with the *time extended* masses. Therefore, Dempster's combination rule when used for for two transitory extended masses $m_1$and m_2 for a hypothesis h, with duration t_{dur}, at time $t + t_{rem}$ will become:

$$m_{12(t+t_{rem})}(h) = \frac{\sum_{X \frown Y = h_{t_{dur}}} m_{1t+t_{rem}}(X).m_{2t+t_{rem}}(Y)}{1 - \sum_{X \frown Y = \phi} m_{1t+t_{rem}}(X).m_{2t+t_{rem}}(Y)} \tag{4.2}$$

where

$$t_{rem} = t_{dur} - t_{elapsed}$$

Timeslice	Sensor events	Preparing drink evidence
9:49	Fridge, cup	Fridge, cup (0.8)
9:50	Fridge	Fridge, cup (0.8)

Table 4.1: Sample timeslices for 'getting drink' situation

4.2.2 Calculating situation durations

To use time-extended evidence, we need to determine how much time the evidence should last for. Intuitively, the evidence should last until the situation is finished, i.e. for the remaining duration of the situation. Situation may last for a different time for each occurrence (such as the variation in the time for 'preparing breakfast' each day). Therefore, we want to capture the 'typical' duration. Duration can be captured from user questions (how long do you typically take to prepare breakfast?), user observation, or training data. If training data is available, we calculate duration as the mean of the situation durations, where $duration = endtime - starttime$ of the annotated situation. The smaller the standard deviation, the greater the consistency of the situation durations over time. The extended lifetime of underlying evidence will then more accurately reflect the real lifetime or duration of the situation.

4.2.3 Worked example of time extended evidence

A simple worked example is provided to illustrate time extended evidence. For each situation, a frame of discernment $\{activity, \neg activity, \Theta\}$is defined. Table 4.1 shows two timeslices from the dataset, during which the occupant is preparing a drink. The fridge and cup sensors are used to detect the 'preparing drink' situation. The fridge sensor has a frame of discernment $\{fridgeUsed, \neg fridgeUsed, \Theta\}$ and the cup sensor $\{cupUsed, \neg cupUsed, \Theta\}$. The occupant always uses the fridge and 'usually' uses a cup, with 80% frequency of using the cup for a drink. Typical duration of the 'preparing drink' situation is three minutes (obtained from user interviews, observation or training data), with both fridge and cup as transitory evidence sources. The inference steps for each timeslice are as follows:

At a time of **9:49**, the fridge and cup sensors fire. Both of these events are indicative of the 'preparing drink' activity, which is not currently in progress. The elapsed time of drink is set to 1 minute (length of timeslice).

Step 1: Use *sensor mass functions* to obtain context value beliefs. Both the fridge

and cup sensors fired:

$\{fridgeUsed = 1,\ \neg fridgeUsed = 0,\ \Theta = 0,\}$

$\{cupUsed = 1,\ \neg cupUsed = 0,\ \Theta = 0\}$

Step 2: Transfer belief from context values to activities. The fridge and cup sensor evidence is propagated to the 'preparing drink' frame using compatibility relations and evidence propagation:

$\{fridgeUsed = 1,\ \neg fridgeUsed = 0,\ \Theta = 0,\}$ propagated to $\{prepDrink = 1, \neg prepDrink = 0, \Theta = 0,\}$

A cup is used with certainty of 0.8 when preparing a drink, with the remainder classified as uncertainty.

$\{cupUsed = 1,\ \neg cupUsed = 0,\ \Theta = 0\}$ propagated to $\{prepDrink = 0.8,\ \neg prepDrink = 0,\ \Theta = 0.2,\}$

Step 3: Combine evidence using Murphy's combination rule to obtain belief for the 'preparing drink' frame. As Murphy's version of the combination rule is being used, the evidence is averaged prior to combining:

$\{prepDrink = 0.9,\ \neg prepDrink = 0,\ \Theta = 0.1,\}$

The averaged evidence is then fused using Dempster's rule of combination, to obtain belief for the 'preparing drink' frame of discernment at time 9:49 as:

$\{prepDrink = 0.99,\ \neg prepDrink = 0,\ \Theta = 0.01,\}$

At the next timeslice **9:50,** the fridge sensor fires again:

$\{fridgeUsed = 1,\ \neg fridgeUsed = 0,\ \Theta = 0,\}$ propagated to $\{prepDrink = 1,\ \neg prepDrink = 0,\ \Theta = 0,\}$

The cup sensor does not fire, but the cup context values from the previous timeslice are extended as they are within the 3 minute duration of the 'preparing drink' activity. The lifetime, t_{rem} of the cup context values is calculated as the 'preparing drink' time duration (3 minutes) less the elapsed time of 'preparing drink' (1 minute), as per equation 4.1:

$\{cupUsed = 1,\ \neg cupUsed = 0,\ \Theta = 0\}$ propagated to
$\{prepDrink = 0.8,\ \neg prepDrink = 0,\ \Theta = 0.2,\}$

Using the extended evidence of the cup and the fridge sensor, the evidence is fused using the temporal version of Dempster's combination rule in equation 4.2. Evidence is averaged prior to fusion as per Murphy's variation on the combination rule, resulting in belief at time 9:50 for 'preparing drink' as:

$\{prepDrink = 0.99,\ \neg prepDrink = 0,\ \Theta = 0.01,\}$

This evidence reasoning process is also conducted for all other nodes in the smart space. At time t, the situation with the highest belief is selected (assuming that only one situation can be happening at one time). If more than one situation can be occurring at the same time, a belief threshold approach can be used to establish which situations are occurring.

4.3 Absolute time

Absolute time is used when a situation occurs at a predictable time. As explained for Situation DAGs, it can be an actual time or time range or a semantic description of the time. Evidential reasoning can easily incorporate domain knowledge, so is suited to the inclusion of absolute time as part of the inference process. This can be done by treating 'time' as a virtual evidence source with its own mass function. A virtual time sensor will be included on the situation DAG and inference rules used to connect the time context values to situations. This will be useful if there is some uncertainty involved such as 'breakfast usually takes place in the morning'. If no uncertainty is included, absolute time can be used directly in the decision algorithm to filter the set of possible situations that can be occurring. For example, if 'preparing breakfast' always takes place in the morning, the activity will only be considered as possible to occur outside of the times defined as within 'morning'. The range of times defined as morning is obtained from the domain knowledge of the user, or from examining training data.

4.4 Summary of temporal extensions

We explain how to include two sources of temporal knowledge into the evidence decision network: transitory evidence and absolute time. Some situations are detected from evidence that does not last for the duration of the situation. This transitory evidence cannot be fused with other transitory evi-

dence for the situation unless they co-occur. To enable the fusion of transitory evidence, we apply a lifetime to transitory evidence so that it lasts for the duration of the situations it is indicative of. We provide equations to explain how evidence lifetime is extended and fused with other evidence. Absolute time can be incorporated as a sensor, if uncertainty exists, or as part of the decision algorithm. We hypothesise that the use of these two types of temporal knowledge will improve the situation recognition results from the evidence decision network. Next, we examine how sensor quality knowledge can be incorporated into the evidence decision network.

4.5 Using quality to weight evidence

Information from sensors is imperfect, as explained in Chapter 2. The strength of evidence from a sensor will intuitively depend on how 'trustworthy' a sensor is - i.e. on its quality. In simple terms, if a sensor cannot be fully trusted, we need to inform the evidence decision network to what extent the sensor is not trustworthy so that evidence from the sensor can be modified. Our hypothesis is that by quantifying sensor imperfection and using it to weight sensor evidence in the evidence decision network, we will improve our ability to recognise situations.

Shafer's original Dempster-Shafer publication included a sensor discount factor as described in section 3.2.3. This allows evidence from a sensor to be weighted or discounted. But it is a static discount only; the same discount applies to all evidence/ sensor readings. We note that sensor quality can result in *dynamic* quality discounts that differ across sensor readings. In our work, we hypothesise that the use of quality parameters in the evidence decision network will improve situation recognition. We define what types of quality issues can be accommodated by Shafer's static sensor discount factor. We also expand on sensor discounting by defining how dynamic quality measures can be included.

An extensive body of work in the context-aware systems field has been published on describing sensor and abstracted context quality in various ways. We will briefly look at these in Section 4.5.1, to explain the nature of quality information available for different types of sensors. We will then explain how to apply sensor quality parameters into the evidence processing, supplying the equations needed to process quality to an abstracted context level.

4.5.1 Sensor and Context Quality

Various researchers in the field have described context quality. They use a variety of quality parameters to describe the type of issues that can arise in context information. Most of the research differentiates between descriptions of quality for sensors, versus quality issues for higher level abstracted context.

Sensor Quality: Sensors are heterogeneous, so it is challenging to provide a set of quality parameters that applies across all sensors [74]. However, the parameters commonly used to capture sensor quality are: precision [30, 10], accuracy [39, 30, 57, 41], resolution [39, 52], range [52, 32], and timestamp [39]. *Precision* indicates the range within which the sensor is correct. i.e. the smallest level of detail it can resolve [32]; A GPS system can have a precision of 1 metre, whereas a GSM triangulation system may be precise to only 50 metres [32]. The impact of precision depends on the context values generated. To detect the location of a user on a street, the GSM may not be precise enough, but it will be sufficient to determine what area of a city the user is in. *Accuracy* indicates the error rate or frequency of correctness of the sensor, for a given precision; *Frequency* or *time stamp* of sensor readings can be used to support the calculation of a time decay measure of abstracted context.

Context quality: The quality of higher level context information is typically described using a general confidence measure [65, 34, 30, 85, 57, 41], and a timestamp or freshness measure [65, 34, 57, 41]. This work on defining quality parameters provides useful semantics for exploring the nature of context quality issues.

The quality of higher level context is dependant on the underlying sensor quality. Lower quality sensor data results in less confidence in derived context. *It is this process of using sensor quality to determine higher level context belief that we want to incorporate into our evidential reasoning process*. By including sensor quality in the evidence decision network, we will be providing more knowledge to the reasoning process: informing the reasoning process where evidence sources suffer from quality issues and adjusting belief levels to reflect this.

4.5.2 Using sensor quality

To incorporate quality into our evidence decision network, we firstly differentiate between two types of sensor quality measures: *static* and *dynamic* quality.

We define static quality for a sensor as quality that is constant for all sensor readings. The static quality equates to Shafer's discount factor from section 3.2.3. For example, a manufacturer's accuracy of 95% for a temperature sensor will result in all belief assignments reduced by 95% of their original value (with the remaining 0.05 assigned to uncertainty).

Unlike static quality parameters, dynamic quality parameters will have a *variable* impact on the belief generated from the sensor. The impact will vary depending upon the abstracted value of the sensor reading(s) and the time of the sensor reading(s). For example, an activity sensor that monitors the keyboard and mouse activity of a PC can have a quality parameter that defines a time decay period of 4 minutes. i.e. after 4 minutes of inactivity, the belief in the 'active' state will have reduced to 0. During the four minute period, belief will reduce from 1 to 0 (with values driven by the time decay function used). So, the impact of the time decay quality parameter varies over time.

4.5.3 Identifying Static and Dynamic quality

To identify whether a sensor has any quality (static or dynamic) issues, the relationship between sensors and the phenomena that they are measuring in the system need to be well understood. At any point in time, we need to determine how closely the sensor outputs reflect the actual state of what is being measured. To what extend does it differ? Why does it differ? Over time, how frequently does it differ? What measure(s) can be used to describe the difference(s)? We will illustrate how static and dynamic quality parameters are identified using two sensor examples:

Location sensor: An office based location system detects the room location of a user in a building, based on tracking the location tag worn by the user. The sensor system generates timestamped coordinate readings as the user moves throughout the building. The coordinate readings are abstracted to room level. The sensor system readings are not correct all of the time. They suffer from several problems: 1) *Precision*: the coordinate readings do not pinpoint the exact coordinate location of the user, but are within some range of the user; 2) *Accuracy*: the readings only fall within a particular precision for a proportion of the time; 3) *Reliability*: A third potential problem is that users sometimes forget to wear their tags so there is a reliability issue for the readings. Taking each of these parameters in turn:

- Precision - The effect of precision on belief from the sensor differs across sensor readings, depending on where in a room the user is located to. As shown in figure 4.3, the effect of precision is to define an area around the actual tag's coordinate reading. The real tag reading may be located anywhere within the precision area, as shown in the left hand diagram. If the precision area is totally encompassed into an area, as shown in the centre diagram, then all belief from the sensor goes to that area. However, if the sensor reading is abstracted closer to an area boundary as shown in the right hand diagram, the belief will be distributed across all the intersecting rooms, with beliefs proportional to the size of the intersection. Therefore, precision is a dynamic quality parameter, with a variable impact on belief distribution. The belief distribution will depend on the sensor reading values and its relationship to the size of the areas to which they are mapped.

- Accuracy- For the particular precision, the location system is accurate x% of the time, where x is linked with a particular precision value. This accuracy applies equally to all sensor readings so it is a static quality parameter.

- Reliability of tag wearing - this captures the proportion of time a user wears their tag. For example, 20% of the time a user forgets to wear their tag, so reliability is 80%. This is a static quality attribute which would apply as a discount equally to all sensor readings. If we were to identify that the user typically forgets to wear their tag first thing in the morning, this could be treated as a dynamic quality attribute with a time function.

Calendar sensor: A calendar sensor monitors a user's electronic calendar. It is used to indicate whether the user is scheduled to be in a meeting or not. The sensor is 'usually' correct in determining whether the user is in a meeting or not. But it is sometimes wrong. There are two reasons for errors: (1) the user does not always attend meetings that are scheduled in their diary so there is an overall 'inaccuracy' of the calendar sensor (2) The user is often late for meetings so at any point in time at the beginning of a meeting, the user may not be there yet. Taking each of these problems in turn:

Accuracy: The user sometimes misses meetings in their diary. If there is no particular time pattern associated with this, the problem can be captured as a static frequency. For example, if the user adheres to 60% of the meetings

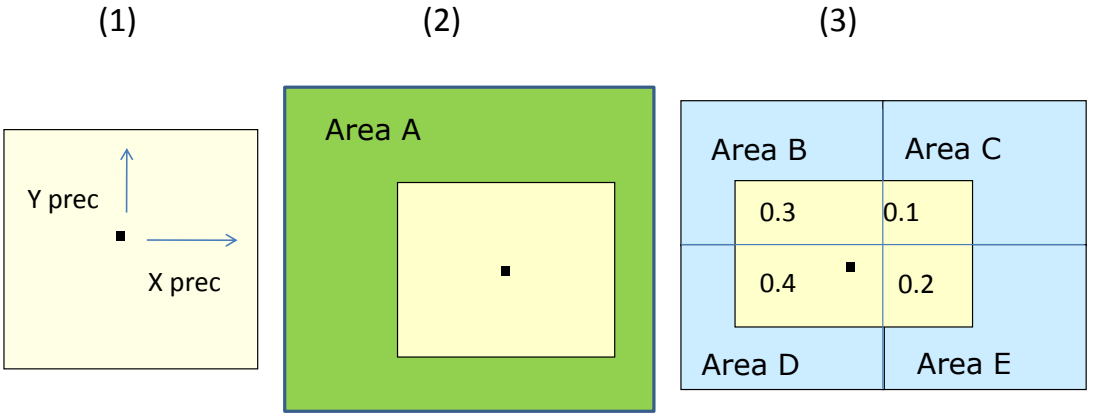


Figure 4.3: Impact of precision on a location sensor: (1) precision area (2) enclosed precision area (3) precision area crossing multiple areas

in their diary, this will be applied as a static quality parameter to all calendar sensor readings.

*Fuzzy start and end time*s: For the meetings that the user does attend, the user is often late by about ten minutes and the meeting usually overruns by about ten minutes. At the scheduled meeting start time, the calendar sensor detects that the user is scheduled to be at a meeting, but can discount its belief that the user is already at the meeting. Ten minutes into the scheduled meeting, the sensor's belief that the user is at the meeting has risen. At the scheduled end of the meeting, the belief that the user is present starts to decrease, reducing to zero ten minutes after the meeting end. Therefore, a dynamic quality parameter (e.g. meeting start and end delay) can be identified that discounts heavily at the beginning and end of a meeting, but discount reduces over time. This parameter is effectively a fuzzy membership function. It will have a shape (e.g. linear) and a calculated membership value between 0 and 1 (dynamic quality value) to describe the discount at any point in time.

4.5.3.1 Quantifying Static and Dynamic quality

The nature and examples of static and dynamic quality parameters for sensors were described in the previous section. To obtain actual values for the parameters, knowledge must be obtained about how the sensors perform and are used in the target environment. i.e. how often does the sensor break down? What proportion of readings are correct to a particular precison? How often does the user forget to wear their location tags. How good is the user at adhering to

the schedule in their calendar?

For each sensor, we require knowledge about how the sensor is used and how well it performs. Sources of this knowledge are training/observing of sensor readings; using expert knowledge such as manufacturer specifications for noise or sensor drift; and user knowledge about their own behaviour. For example, the precision and accuracy of the location system can be calculated via training from a set of controlled location readings, where the annotated readings are obtained in the target environment. Based on knowing the actual (annotated) location and the reported location from the location system, the precision can be determined and the rate of correctness (accuracy) for that precision can be captured. We performed this exercise for our in-house location system [123], observing that for 70% of readings, the reported readings fell within 3.33m/ 2.2m (x /y-axes) of the actual tag location. Therefore, the accuracy is 0.7, for precisions of 3.33/2.2m. To quantify the reliability of a user wearing their tag, the user can be asked to record this in a formal exercise over a period of time, or it can simply be derived from the user's own knowledge.

In Chapter 5, we use a data set where the sensors are characterised by the quality issues described in Section 4.5.2. We explain how we obtain values for the quality parameters for each sensor through training, observation and user knowledge.

4.6 Using quality parameters with evidential reasoning

To combine quality with evidential reasoning, we need to incorporate the quality values into the mass functions for each sensor so that they modify belief in an appropriate way.

4.6.1 Static quality parameters as evidence

Static quality impacts all sensor readings equally so can be applied in a similar manner to the discount factor defined by Shafer. Static quality parameters are expressed as a value, Q_s between 0 and 1. Belief is reduced by the static quality value, Q_s, with the remaining belief assigned to uncertainty. The mass assigned to a hypothesis, A when modified by static quality parameter Q_s, where $(0 \leq Q_s \leq 1)$, and Θ represents uncertainty, m is a mass function, is

calculated as:

$$m_s(A) = \begin{cases} Q_s.m(A), & A \neq \Theta \\ m(\Theta) + (1 - Q_s).m(\Theta), & A = \Theta \end{cases} \tag{4.3}$$

4.6.2 Dynamic quality parameters as evidence

The impact of dynamic quality parameters depends upon the nature of the quality issue. The effect of the dynamic quality parameters is to redistribute belief after the original mass function has finished its distribution. The total mass of the mass function must sum to 1 and no mass can be distributed to the null set (as per standard Dempster-Shafer mass function). i.e. for a mass function m, a dynamic quality function Q_d redistributes belief of the mass function m, within the following constraints

$$Q_d(m_\phi) = 0 \tag{4.4}$$

and

$$\sum_{A \subseteq \Theta} Q_d(m(A)) = 1 \tag{4.5}$$

Sometimes, a sensor will have both static and dynamic quality parameters. In this case, the dynamic quality function is applied first, and each belief is then multiplied by the static quality parameter(s). For a mass function m with static quality parameter $Q_s$and dynamic quality function Q_d, the modified mass of hypothesis A, $m_Q(A)$ after applying quality to the original mass $m(A)$ is calculated as :

$$m_Q(A) = \begin{cases} Qs_s.Q_d(m(A)), & A \neq \Theta \\ Q_d(m(\Theta)) + (1 - Q_s).m(\Theta), & A = \Theta \end{cases} \tag{4.6}$$

4.7 Time and quality notations on the situation DAG

Temporal knowledge and quality are annotated in the Situation DAG if they are relevant to the environment.

- On the situation DAG, we annotate situations that are inferred from transitory evidence using the '<>' identifier under the situation name. This indicates that the situation has an identifiable duration over time, and is detected from transitory evidence.
- On the situation DAG, the absolute time at which a situation occurs, is documented above the activity title between the ': :' symbols.
- The existence of quality parameters is denoted beside the sensor node. Static quality Q_s is included with its actual value. Dynamic quality for a sensor, if used, is shown with its relevant parameter, such as $Q_{d:precision}$ or $Q_{d:timedecay}$. The exact working of the dynamic function is sensor specific and is not stored on the situation DAG.

4.8 Conclusion

In this chapter we created two extensions for Dempster-Shafer theory to enable additional knowledge to be used in the evidence decision network: (1) temporal and (2) quality extensions. The temporal extension will allow transitory evidence to be used as evidence. Time extension of transitory evidence extends evidence lifetime to last for the duration of situations. This allows evidence to co-occur instead of occurring sparsely or sporadically throughout the duration of the situation. This will address the simplifying assumption of existing approaches that all evidence is co-occurring. Absolute times of situation occurrence can also be used to boost recognition. The use of sensor quality as evidence is explained. Static quality is a uniform degradation of sensor output. Dynamic quality parameters have a variable impact on sensor output, such as precision for a location system, or fuzzy time decay functions. For each sensor, systems developers will determine whether static or dynamic (or none) parameters are relevant for each sensor.

At this point, we have established the theory of our evidence-based approach, supplemented with temporal and quality knowledge. Many separate pieces of knowledge can be incorporated in reasoning : inference rule certainties, absolute times, transitory evidence, rich sensor quality measures, situation hierarchies. It is worth emphasising that only knowledge *relevant* to an environment will be included. It is not required to use all types of knowledge in the evidence approach - system designers can select those relevant in the environment of an application.

In the next chapter, we determine the evidence decision networks for two data sets - a home and office based data set. To do this, we establish the situation DAG for each data set.

CHAPTER

FIVE

Using evidence decision networks for real environments

Chapters 3 and 4 presented the theoretical foundations for our Dempster-Shafer based reasoning approach. In this chapter, we demonstrate the feasibility of using evidence decision networks using two data sets from two different environments: a smart home data set from the intelligent autonomous systems in University of Amsterdam [109] and an in-house office data set from the CASL research lab in University College Dublin [74]. The chapter explains a key step in setting up an evidence decision network: the creation of the situation DAG to support belief distribution and decision steps. In Section 5.1, we explain how knowledge for the evidence decision network is obtained. This is followed in Section 5.2 by a detailed walkthrough of the steps required to set up the situation DAG. In Section 5.3, we explain our requirements for the data sets. Sections 5.4 and 5.5 describe how the situation DAGs are established for each of the two data sets. A discussion and summary are provided in Section 5.5.

5.1 Using evidence decision networks for pervasive environments

To use the Dempster-Shafer reasoning in a particular pervasive environment, systems designers must establish the structure of the evidence decision network: sensor mass functions, context values, inference rules, rule certainty, situation hierarchy, temporal and sensor quality knowledge. This knowledge is captured in the situation DAG.In Chapters 3 and 4, we explained the theory of how evidence is applied in the evidence decision network. In this chapter,

we explain how to use the evidence decision networks in two smart environments. This consists of the following

- For each of the two environments under detection, we determine the structure and content of the situation DAG. The DAG notates what situations are being recognised, what sensors are in the environment, the inference paths from sensors to context values to situations, and additional information about sensor quality and temporal information, if relevant. We will create situation DAGs for two environments, described by two publicly available annotated data sets.
- Once the DAG is established for an environment, the structure of the evidence decision network is known. That is, the process of propagating sensor evidence is defined because the evidential operations can be read from the DAG, as explained in Chapter 3.
- An automated process is then required to implement the evidence propagation and conduct situation recognition. This process reads in time sliced sensor readings as input, and produces situation recognition beliefs as output, as explained in the evidence decision network architecture in Figure 3.2. For our work, we developed our own software to process evidence. This is described further in Section 6.2.4.

To support these steps, we must gather knowledge in order to create situation DAGs. Firstly, we describe how this knowledge can be discovered to support the set up of the evidence decision network. We then describe each of the steps for setting up the situation DAG.

5.1.1 Generating knowledge

Knowledge will be discovered from a number of sources:

(1) System developers may provide expert knowledge about the structure of the DAG. In particular, as shown in figure 5.1 they will have knowledge about the lower levels of the DAGs: the available sensors, the ranges of values of the sensor and sensor performance. This knowledge would be used to establish the sensor mass functions, context values and quality parameters.

(2) Applications designers and application users will determine the highest level of the DAG - the situations that will be monitored. This layer drives

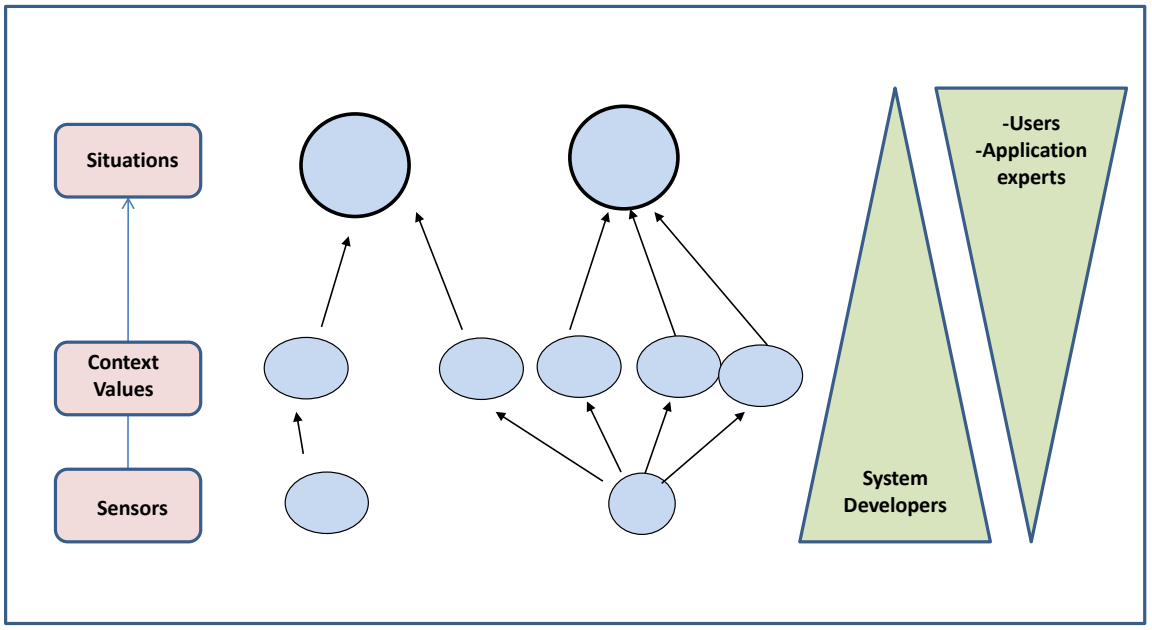


Figure 5.1: Knowledge contributions for the situation DAG

the behaviour of the application and is a fundamental part of the application requirements definition.

(3) User interviews and user observation can be used for environments where people are central to the situations being tracked, such as interviewing users about their steps in activities in a smart home. This will help to discover which objects and locations are associated with which situations. Their knowledge, if used, will be important to discover the middle layers of the DAG, where individual tasks or steps contribute to situations. User knowledge is also a useful source of temporal knowledge, defining when they typical undertake activities and for how long. In user interviews, frequencies for inference rules can be captured semantically such as 'sometimes', 'usually', 'always', and translated to numeric frequencies between 0 and 1. An advantage of using user knowledge is that the user will form an accurate conceptual knowledge of the environment, thus enabling them to use the system more easily [86].

It is essential for humans to form an accurate conceptual model of the environment so that they can interact with the environment easily.

(4) Training data for subsets of the environment can be used to help discover inference rules, to quantify frequencies for rules and to discover temporal knowledge. The use of training data can be localised for portions of the situation DAG where other knowledge is not available or where pockets of train-

ing data are reasonable to capture. For example, in a monitored smart home, kitchen activities could be tracked using a local camera. More sensitive situations, such as bathroom activities, could then be specified without capturing training data.

(5) More recently, the use of unsupervised data mining techniques has been employed by Palmes *et al*. [81] to capture general activity (situation) definitions. For human activity recognition, they obtain 'activity models' by mining activity definitions from a variety of publicly available web definitions. The activity models define which objects are used in an activity, with weights attached to the importance of each object as an indicator of the activity. This approach is very similar in concept to our use of inference rule frequency as an indicator of how often the rule applies. This approach is complementary to our evidential approach because it provides domain knowledge that might not be available from other sources.

In summary, availability of user knowledge, system designer knowledge, localised training data, user privacy issues and cost of data capture will determine which techniques will be used to establish the situation DAG. An environment does not need to be limited to a single approach of knowledge discovery. In practice, a hybrid of approaches can be employed.

5.2 Establishing the situation DAG

The set-up of the situation DAG is a critical step in using our Dempster-Shafer based reasoning approach. We used the generated knowledge to map evidence paths from sensors to situations, as per the sample DAG explained in figure 3.1. The steps involved in defining the information for the DAG are as follows:

	Step
1	Identify target situations
2	Identify context values, mass functions and sensor frames
3	Establish inference paths and frequencies
4	Identify absolute times
5	Identify situations durations and transitory evidence
6	Define sensor quality parameters
7	Identify frames of discernment

Identify target situations: The situations requiring recognition are the trig-

ger points for application behaviour. Application designers and application owners will identify the requirements for this top layer of the DAG. As part of listing the situations and understanding what they mean, the co-occurrence of the situations should be identified i.e. which situations can/cannot co-occur. This will feed into the decision algorithm in the evidence decision network.

Identify context values, mass functions and sensor frames: Each sensor in the system generates readings that describe the state of something of interest in the environment, such as whether a door is open or closed, or the physical location of a location tag. To process sensor evidence, these readings are translated into more human understandable context values using the knowledge of system developers. The set of context values for each sensor depends on the *reading granularity* available from the sensor and the *inference requirements* of the situations. For example, our second data set uses a tag-based location system called Ubisense [19] to track user locations in an office building. For the Ubisense sensor system, the possible context values are determined by the level of granularity of the system and by the areas in the office that are used to deduce higher level situation. Ubisense generates 3-D coordinate readings for a tag. The areas of interest in the building are 'user's desk', 'cafe', 'meeting room' and 'all other areas (as a group)', because situations happen in these places. Ubisense's coordinate readings are sufficiently granular to abstract to these areas, so the context values are: user's desk, meeting room, cafe or other. Taking an example from our first data-set which is based in a smart-home, a simple binary door sensor on the bedroom door can output readings of 0 (door not used) or 1 (door used). The context values for the sensor are simply 'door used' or 'door not used'.

The context values for a sensor form the basis for the frames of discernment for each sensor. In the case of the binary door sensor, the frame of discernment for the sensor for this door object are $\{doorUsed, doorNotUsed, \Theta\}$. In the case of the location sensor, the singleton elements (plus uncertainty) in the frame are $\{userDesk, meetingRoom, cafe, other, \Theta\}$.

Establish inference paths and frequencies: Once the context values are defined, the inference rules and their associated frequencies from context values to situations are defined. The knowledge of inference rules and frequencies can be captured in a variety of ways, as explained in Section 5.1.1: using developer knowledge, user knowledge, localised training data and web mining techniques.

Identify absolute times: If situations have an identifiable occurrence time, the absolute time will be noted either as a time range, or a semantic description, such as 'morning'. Absolute times can be identified through participant interviews, participant observation (such as a user annotating their own activities times over a period of time) or high level training data, where the time of annotated situations is captured. Absolute time can be treated as a sensor with a mass function, or it can be encoded into the decision level algorithm.

Identify situation durations and transitory evidence: Systems developers determine whether evidence of a situation is transitory or continuous, as this requires an understanding of the sensor operations. Where transitory evidence occurs and the situations have a typical duration, a representative duration is identified. To identify durations, user interviews ('how long do you typically take to prepare breakfast?'), user observation or localised training data can be used.

Define sensor quality parameters: For each sensor that is deemed to have quality issues, static and/or dynamic quality parameters are identified. Quality issues can be technical or user driven. Knowledge of systems developers is required to identify quality parameters associated with a sensor's technical issues, such as the precision of a location system. The values of such parameters are supplied by the sensor manufacturer or measured through localised training. For any sensor that is worn or operated by a user, user behaviour may result in quality issues, such as the failure of users to wear a locator tag, or the lack of adherence to their calendar for a user's calendar sensor. The value of user controlled parameters are identified directly by the user or through observation.

Identify frames of discernment: Every node in the situation DAG needs to be placed in a frame of discernment, so that belief can be assigned to the node as appropriate. The number of frames and their elements can be done in different ways, depending on the spread of sensor evidence across situations. We describe two configurations for frames within the DAG:

1. Each situation can be allocated its own frame consisting of $\{situation, notSituation, \Theta\}$. This is particularly suitable if sensors are evidential of single situations, with no obvious grouping of sensors and situations. This is the case with our first data set where 14 sensors are used to detect 7 situations, but different sensors detect different situations. We place each situation into its own frame.

2. Alternatively, if a group of situations share a common set of sensors, these situations can be grouped into a frame. This is the case in our second data set, where the situations to be detected are all determined from the same three sensors. In this case, the frame of discernment will contain the individual situations $\{situation_1, situation_2, ...situation_n\}$ and their combinations:$\{situation_1 \wedge situation_2, ..., \Theta\}$.

5.2.1 Summary

In summary, the situation DAG uses one or more sources of knowledge from system developers, expert knowledge, user knowledge, user observations, localised training data or data mining techniques. Each layer of the DAG for the environment is then established as a set of steps, from sensor mass functions up to situation frames of discernment.

In the next section we establish the situation DAGs for two experimental data sets.

5.3 Experimental data sets

To evaluate our evidence decision network, we need to use data sets that capture annotated situations and sensor readings for some type of environment occupied by people. Our preference is for data sets collected in a real-life environment, so as to avoid problems of simulated environments [68]. We need experimental data to include both *temporal* characteristics and *sensor quality* issues, so that we can evaluate our extensions to Dempster-Shafer theory.

Temporal characteristics tend to be available in smart home data sets because situations (or activities) of occupants in a home tend to have a pattern across a day. Increasingly, simple sensors are being used in smart environment deployments, such as simple binary sensors that can be installed quickly and remain largely invisible to the user (as opposed to user tags, audio or camera systems). Such binary sensors offer transitory, event-style evidence when objects to which they are attached as used in situations. In order to evaluate our temporal extensions, we required a smart home dataset that contains situations with discernible time durations over a time period. Our requirement was to use a real-life smart home dataset rather than one captured in a laboratory environment. Availability of published smart home datasets is still a chal-

lenge in the pervasive computing field, particularly where published results are desirable, using repeatable methodologies [122]. We use van Kastersen's smart home data set which is collected in a real-life home. It contains temporal patterns of situation occurrence. Also, there are published situation inference results available for the data set [119, 109], produced using well documented *repeatable* methodologies that we can compare against.

Finding a data set with known quality parameters is more difficult. Data sets do not typically come with sensor performance characteristics included. Also, domain knowledge of the data set required to understand quality resides with the creators of the data set. To use a data set with quality issues and to illustrate the use of domain knowledge in establishing situation DAGs, we collected our own annotated office-based training data at our research laboratory. As part of this, we identified and quantified quality parameters for each of our sensors using a combination of training exercises and domain knowledge. The data sets and the creation of their situation DAGs are described next.

5.4 Van Kastersen's Data set

This data set originates from the intelligent autonomous systems group in University of Amsterdam [109]. The data is recorded in the apartment of a 26-year old man over a 28 day period. 14 digital sensors were installed in the house and left unattended for the duration of the data collection. Figure 5.2 shows a layout of the house. The sensor locations are marked with red 'X' symbols.

5.4.1 Sensor Description

Each sensor is attached to a wireless sensor network node. Each node sends a sensor event when the state of the digital input changes. The sensors are installed on the hall-toilet door, hall-bathroom door, hall-bedroom door, front door, microwave, fridge, freezer, washing machine and each of the cups/plates/pans cupboards. When a sensor is fired, it outputs a value 1 as its reading in the sensor output file. A snapshot of the content in the sensor output file is shown in Appendix, Section 3. In total, the data set results in 1,319 sensor events.

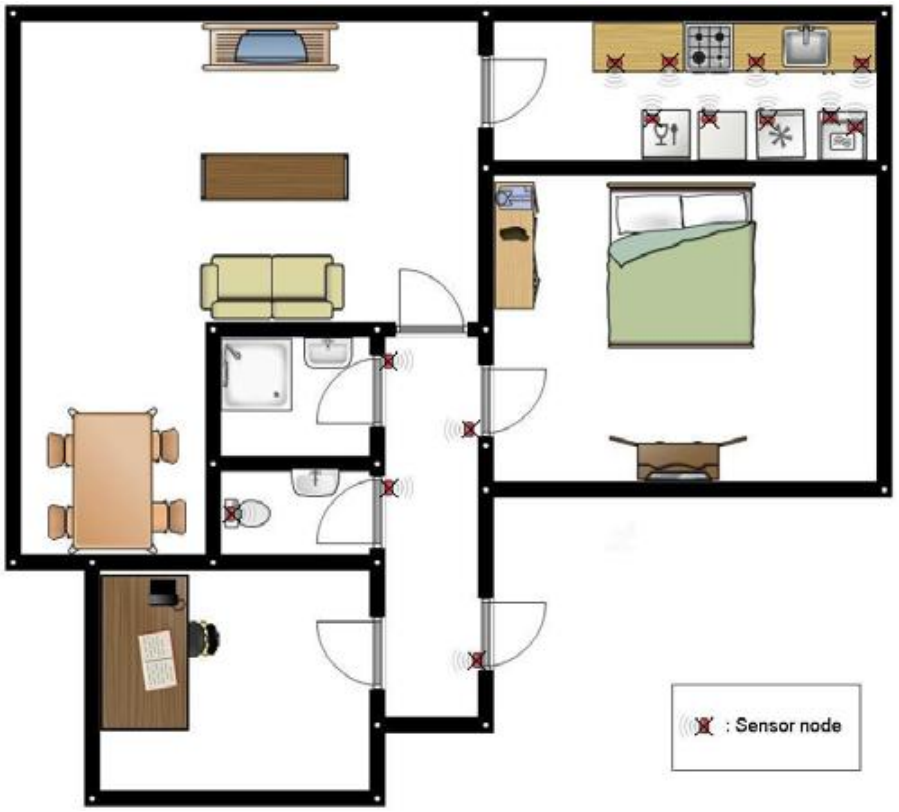


Figure 5.2: Layout of sensors in van Kasteren house floor plan [109]

5.4.2 Situations in van Kasteren's Data Set

Seven situations (termed 'activities') are annotated by the occupant of the house:1) leave house 2) use toilet 3) take shower 4) go to bed 5) prepare breakfast 6) prepare dinner 7) get drink. These situations were chosen based on the Katz ADL index, a commonly used tool in healthcare to assess cognitive and physical capabilities of an elderly person [109]. All the activities are annotated by the occupant of the house using a Bluetooth headset that was combined with speech recognition software. When a button on the headset is pressed by the subject, then the speech recognition engine is triggered to start listening for commands it can recognise. When the command was recognised, a new annotation entry would be added. This annotation method yielded close to perfect recognition results [109] due to its efficiency and accuracy of annotating on the spot. The annotation contains 245 separate activity instances, a sample of which are shown in the appendix.

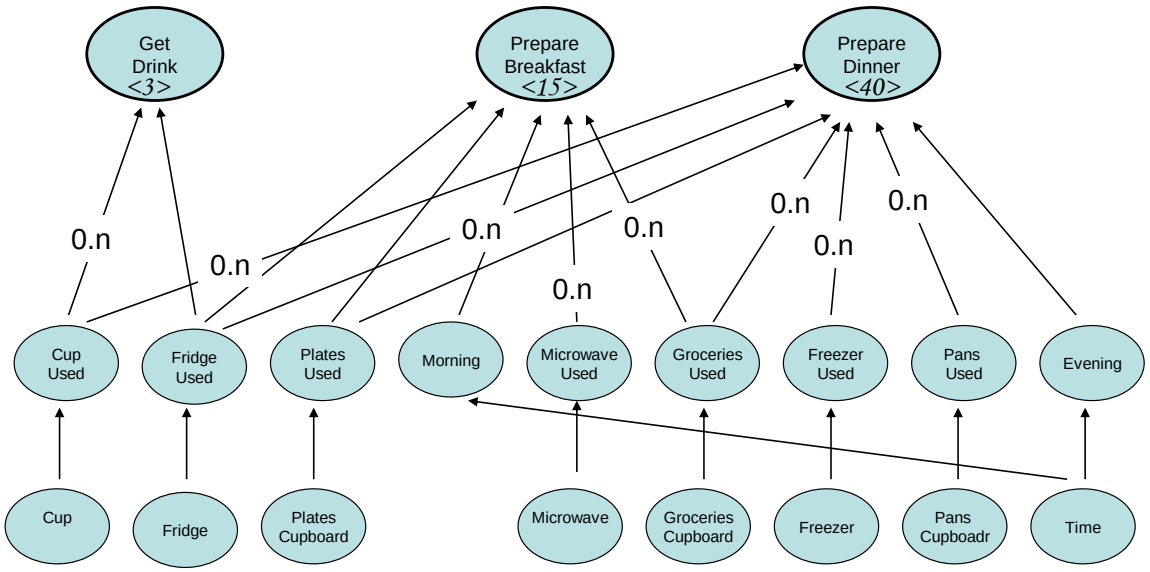


Figure 5.3: van Kastersen DAG for 'get drink', 'prepare breakfast', 'prepare dinner' situations

5.4.3 Van Kastersen Data preparation

To use the data set for testing situation recognition accuracy, we need to timeslice the data. All sensor readings for all sensors are timesliced into regular time intervals, and annotated with the occurring situation for that timeslice. We match all sensor instances against the times for which annotated situations are provided. The data is timesliced into timeslices of one minute, which is long enough to be discriminative and short enough to provide high accuracy labeling results [109]. This is the same time slice length used by other researchers who have used the data set, Ye [119] and van Kastersen [109], so paves the way for results comparison. For every minute at which a situation annotated is available, the sensor values are annotated. After time slicing the data, there are 25,680 annotated sensor instances. This is the format of the data that we use for building the situation DAG and for testing the accuracy of situation recognition.

5.4.4 Preparing the Situation DAG

A situation DAG is required to cover all situations and sensors in the data set. We executed our situation DAG set-up steps as follows:

Step 1: Identify target situations: There are seven target situations to be recognised as listed in Section 5.4.2. They cannot co-occur, with only one situation happening at a time.

Step 2: Identify sensor mass functions, context values and sensor frames: From the documentation supplied with the data set, we can deduce the mass functions, context values and frames. The sensors are all simple binary sensors. Each sensor therefore has two elements, plus uncertainty: $\{sensorUsed, sensorNotUsed, \Theta\}$. The grocery cupboard door sensor, for example, has a frame $\{groceryUsed, groceryNotUsed, \Theta\}$. A sensor value of '0' maps to *sensorNotUsed*. A sensor value of '1' maps to *sensorUsed*. Since sensor quality is not an issue, the sensors will never assign any belief to uncertainty, but it has been included in the frame for completeness.

Step 3: Establish inference path and frequencies from context values to situations: As discussed in Section 5.1.1., domain knowledge is required to establish the situation DAG. In a real-life environment, the inference paths between sensors and activities can involve user interviews. Questions such as 'what do you do when preparing breakfast?' will establish which sensors are being triggered for each activity. As we are using a third party data set, we do not have easy access to the user to glean domain knowledge, so we need to establish knowledge about the environment without relying on user-specific knowledge. To achieve this, we use a limited amount of training data, combined with domain knowledge to establish our situation DAG. A common practice in machine learning is to use two thirds for training with a third for testing [113]. We reverse these proportions to illustrate the scenario of limited training data (one third only) relative to test data [75]. Using a third of the data set, the sensors that are triggered for each activity are identified. We repeat this process for each of the data thirds (folds) in a cross validation exercise. In addition, common sense domain knowledge of home activities enables the following assumptions:

- Activities in the kitchen (breakfast, dinner, drink) only involve sensors that are located in the kitchen;
- No sensors will be firing when 'leave home' and 'sleeping' are happening because the occupant is not active;
- Door sensors are of interest when their state is changing, but a door left open (with an ongoing value of '1') is not useful for inference.

Context Value	Inference Rule frequency
Microwave Used	0.1
Cup Used	0.1
Fridge Used	1
Plates Used	1
Pans	0.3
Freezer	0.4
Groceries	0.6

Table 5.1: Sample inference rule frequencies for 'prepare breakfast'

Activity	Absolute Time
Prepare breakfast	Morning
Prepare dinner	Evening
Showering	Morning
Leave home	Daytime
Sleeping	Nighttime

Table 5.2: Absolute Times for situations in van Kasteren's data set

Looking at an activity example, table 5.1 shows the inference rule uncertainties generated for the 'preparing breakfast' activity for one of the thirds of the data set. The pan cupboard sensor triggering is 0.3, indicating that for occurrences of 'preparing breakfast', the pan cupboard is used for 30% of these occurrences. The remaining 0.7 is allocated to uncertainty.

Step 4: Identify absolute times for situations with identifiable occurrence times: The starting assumption based on domain knowledge are that breakfast is taken in the morning, and dinner in the evening. Training data samples confirm the actual times denoted by the semantic labels of 'morning' and 'evening'. The data also shows that showers are taken in the morning. 'Preparing drink' occurs at various times during the day and night so no particular time pattern is evident. The absolute times are shown in table 5.2

Step 5: Identify situation durations for situations with transitory evidence: Of the seven situations, five are detected by sensor events that may not co-occur i.e. transitory evidence: 'take shower', 'use toilet', 'prepare breakfast', 'prepare dinner' and 'prepare drink'. 'Leave home' and 'Go to bed' are identified by lack of sensor activation. In the absence of user knowledge, we establish an average duration using the folds of data in our cross validation approach.

Step 6: Identify and quantify static and dynamic quality parameters: the sensors have no documented or apparent quality issues, as confirmed by us

with the publisher of the data set. Therefore, sensor quality is not incorporated into the situation DAG.

Step 7: Frames of Discernment: Each sensor is evidential of just one of two situations so we use a separate frame for each situation: $\{situation, notSituation, \Theta\}$, as described in Section 5.2.

The portion of the situation DAG for the kitchen based situations is shown in figure 5.3. Sensors are shown at the bottom of the diagram, context values are the upper nodes for each sensor, and each context value maps to one or more situations. Inference rule uncertainty is annotated on the DAG, but actual values will depend upon which portion of the data set is used for training so will be assigned during experiment runs, as described in our evaluation methodology in Chapter 6.

5.5 CASL Data set

Our second data set, termed the 'CASL' data set was generated in the Complex and Adaptive Systems research laboratory of University College Dublin. As this is our own data set[1], collected by the author, we have a deep understanding of the sensors and situation involved, so we can employ domain knowlege to establish the situation DAG. The purpose of this data is to support the evaluation of using sensor quality with the evidence decision network. To examine sensor quality, we needed to use sensors that display a variety of static and dynamic quality issues, such as inaccuracy, time decay and imprecision.

5.5.1 Sensor Description

We built our own infrastructure to capture the following data about a person in our research lab environment: their computer activity, their calendar entries, and their physical location in the building. Describing each sensor in turn:

- The computer activity sensor runs on the participant's desktop PC and monitors the rate of key presses and mouse clicks, along with the length of time since the last activity. The data from this sensor is used to indicate whether the user is 'active' or 'inactive' at their desktop.

[1]This data set is described and used in [74]. It is available to download at www.comp.dit.ie/smckeever/research

- The calendar sensor collects information about the user's scheduled Google calendar events, including meeting schedule state, start time and end time. The data from this sensor indicates whether the user has a meeting or not for the current time.

- For Location sensing, we have a Ubisense system deployment on the third and fourth floor of our research lab. Ubisense is a tag-based 3-D location tracking system. It provides an X, Y and Z coordinate, based on the number of metres from an origin point – in our case the bottom corner of the third floor in CASL. Ubisense covers areas on the third and fourth floors in the CASL building, using 30 wall mounted sensors. Ubisense tracks a location tag belonging to the user, providing co-ordinate readings for the location tag when the tag is moved.

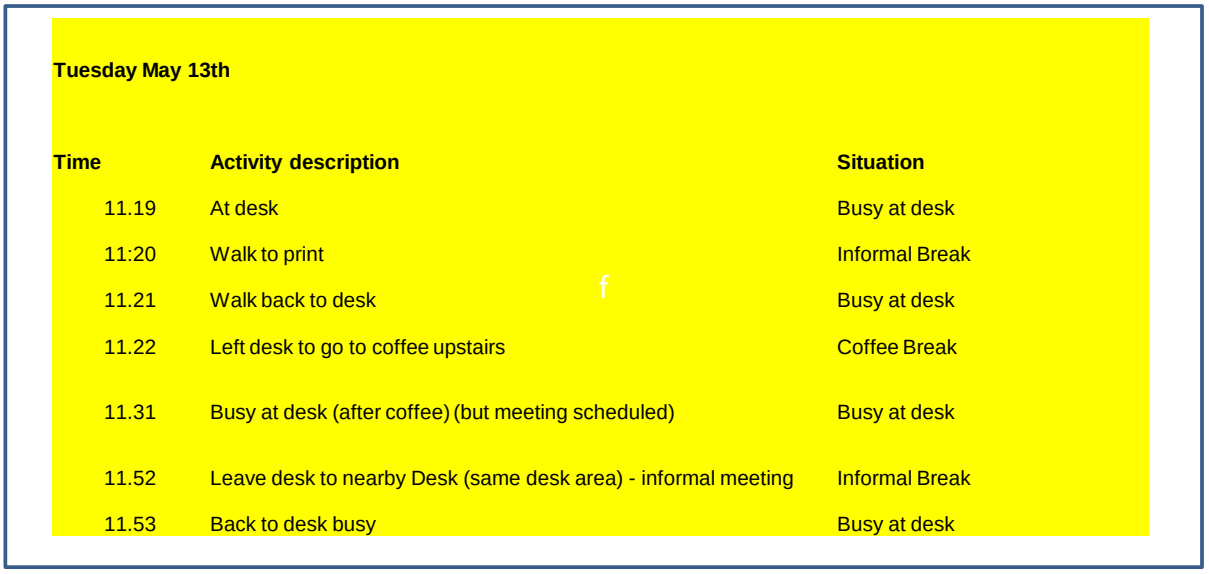
The participant gathered a data set over a 5-day period. We used a period of 5 days as this is a sufficient amount of time to capture a person's regular working-day routine (i.e., some instances of them going for lunch, having meetings, working at their desk, etc.). The participant turned the sensors on at the beginning of the work day and off at the end. The participant annotated the situations that she encountered throughout her days using an electronic diary. She logged her activities in the spreadsheet for each day, noting the start and end times of different situations, and a description of the situation. A sample of the diary is shown in figure 5.4.

5.5.2 Situations in CASL data set

Six situations were annotated: (1) busy at computer, (2) busy reading at desk, (3) coffee break (4) lunch break, (5) informal break and (6) at meeting. Each of these is detectable from the combination of the three sensor systems in our infrastructure. At any point in time, the participant can only be engaged in one of these situations, but is always engaged in one of the situations.

5.5.3 CASL Data preparation

Similar to van Kastersen's data set preparation, we process the data into timesliced annotated sensor instances. For each sensor, we time slice the readings into timeslices of one minute. We process the translation of sensor readings



Tuesday May 13th

Time	Activity description	Situation
11.19	At desk	Busy at desk
11:20	Walk to print	Informal Break
11.21	Walk back to desk	Busy at desk
11.22	Left desk to go to coffee upstairs	Coffee Break
11.31	Busy at desk (after coffee) (but meeting scheduled)	Busy at desk
11.52	Leave desk to nearby Desk (same desk area) - informal meeting	Informal Break
11.53	Back to desk busy	Busy at desk

Figure 5.4: Sample of diary annotations CASL data set

into context events in the same step. Ubisense may have several readings occurring in a single minute, in which case we take the last reading that occurred before the end of the time slice. After time slicing the data, there are 1,453 annotated sensor instances, each instance containing the time, the abstracted values for the three sensors and the occurring situation.

5.5.4 Preparing the Situation DAG

As owners of this data set, we can use domain knowledge to establish the majority of the inference knowledge for the situation DAG, and supplement it with observation and localised training exercises where domain knowledge is not available. Using each of the steps to establish the DAG:

Step 1: Identify target situations:

There are six situations to be detected as listed in Section 5.5.2. They cannot co-occur, with only one situation happening at a time. The user is always in one of the situations.

Step 2: Identify sensor mass functions, context values and sensor frames:

For the three sensors in the system:

(1) The activity sensor on the user's desktop captures whether the user is active or not. The context values used are 'active' and 'inactive'. If a key/mouse

usage occurs during the timeslice, the sensor mass function assigns its belief to the 'active' state. If no usage occurs, the belief is assigned to 'inactive'. The frame for the sensor mass function is $\{active, inactive, \Theta\}$.

(2) The calendar sensor indicates whether the user is scheduled to be in a meeting or not. The context values used are 'meeting' and 'no meeting'. The sensor mass function assigns its belief to 'meeting' if there is a meeting scheduled in the diary for the time in question, and assigns belief to 'no meeting' if the diary is empty. Quality issues are explained in Step 6.

(3) The ubisense sensor indicates what location the user is in. The sensor is capable of detecting the user's tag, with a point coordinate reading, on the third or fourth floor. For situation detection, as previously stated, we are interested in four particular locations: desk, cafe, meeting room and 'all other locations'. Therefore, the context values are 'user desk', 'cafe,' 'meeting room', 'other'. The frame for the sensor mass functions contains the singleton elements:$\{desk, cafe, meetingRoom, other\}$ and all possible combinations of the singletons:$\{desk \wedge cafe, desk \wedge cafe \wedge meeting, ..., \Theta\}$. In practice, most of the combination elements are never used.

Step 3: Establish inference path and certainties from context values to situations. As owners of the data set, we have expert knowledge about the inference rules linking context values with higher levels states. These are shown in the situation DAG in figure *5.5*. Each rule is certain, with no frequency attached.

Step 4: Identify absolute times for situations with identifiable occurrence times: In this data set, the only absolute time we can discern is for the 'lunch' situation, which takes place each day between 12:00 and 2:00. We treat this as a certain time (i.e. lunch always takes place at this time) so we apply it in the decision algorithm, rather than as a virtual sensor, as discussed in Section 4.3.

Step 5: Identify situation durations for situations with transitory evidence: The sensors do not transmit transitory, event-style evidence. They provide continuous values about the calendar, activity and location states of the user so evidence can be fused from all three sensors at each point in time.

Step 6: Identify and quantify static and dynamic quality parameters for sensors: Unlike the simple binary sensors in van Kastersen's data set, the sensors in this data set do not detect their higher level state in an immediate, reliable manner all of the time. In order to factor in sensor performance into the evidence reasoning, we need to identify the **types** of quality problems (quality

parameters) and then **quantify** each type of problem (values for quality parameters). This means examining why sensor readings are not correct (if ever), and to what extent they are not correct, as previously described in Section 4.5.3 for the CASL data set sensors. This can be done in two ways (1) using domain knowledge (2) capturing training data. Domain knowledge is useful for identifying the type of quality problem. Training data is more useful for applying numbers to the quality parameters, although in the absence of training data, domain knowledge can be used. A summary of the quality parameters for the CASL data set is shown in table 5.3.

Calendar sensor: In advance of the data collection exercise, we monitored the attendance rate of the participant. Of 36 meetings in the diary of a month's period, 22 (60%) were attended i.e. an accuracy of 0.6. This is an indication of the overall accuracy of the calendar as an indicator of actual meeting activity. In addition, during attended meetings, the attendance time did not exactly match the scheduled time. The participant tended to arrive late meetings and the meetings tended to run over. i.e. a time delay appears around the start and end time. We estimated this imprecision around the start and end time of meetings to be 10 minutes, based on the participants input. Based on this, we applied a fuzzy membership function spread over ten minutes. i.e. at 5 minutes past start time, the meeting membership is 0.5, increasing to 1 by 10 minutes after start time, and vica versa at meeting end times.

Activity sensor: This sensor is used to detect whether the participant is using the PC or not. The readings are abstracted to values of 'active' or 'inactive'. 'Inactive' indicates that the user has not recently used their keyboard or mouse. The performance of the activity sensor itself is not prone to error. i.e. the sensor always captures activity on the keyboard or mouse. However, error creeps in when the sensor is abstracted to active or inactive. If no fuzziness is applied, the user moves instantaneously from active to inactive after whatever time period is chosen (e.g. 1 minute). In reality, transition from active to inactive may be a more gradual process, with the extent of inactivity increasing the longer the time since they used their computer. We apply a fuzzy time decay function to capture this gradual slide into inactivity rather than a step change. The selection of the time period (4 minutes) is subjective: it seems a reasonable time to assume inactivity. However, from our DAG, we can see that other states that rely on "inactive" (coffee break, informal break, meeting, reading, meeting) will take longer to recognise if the system takes longer to become inactive.

Sensor	Static Quality	Dynamic Quality
Ubisense	Accuracy (0.7)	Precision (3.33m x-axis, 2.2m y-axis)
Calendar	Accuracy (0.6)	Fuzziness at meeting start/end (10 minutes)
Activity	None	Time decay (4 minutes)

Table 5.3: Quality parameters and values for CASL data set

Ubisense sensor: The Ubisense location system has the most severe quality issues, with large discrepancies between exact location and coordinate readings from the system. Although the Ubisense manufacturer supplied precision figures at installation time, issues of user orientation, metal reflection in the room, walls and so forth all contribute to degradation in performance in our office environment, as discussed in [19]. We conducted our own training exercise, as described in [123] whereby we captured sample readings and used them to measure the precision and accuracy of the system. Based on this, we calculated that 70% of readings fall within 3.33m along the x-axis and 2.2m along the y-axis respectively[2]. These are much worse than the manufacturer's quality measures (precision of 15cm). Based on this, it is possible that the wrong room will be detected as the location even if a reading falls within the precision distance, as discussed in Section 4.5.3. For each reading that we use, we abstract the coordinate reading to the relevant context value location (desk, cafe, meeting room, other) using the floor maps for the building. All belief is assigned to the abstracted location. Using quality parameters, we then modify the original belief distribution using the dynamic precision and static accuracy (0.7) quality parameters. We apply the precision-x and precision-y values as the dynamic quality parameters. As described in Section 4.5.3, we use the precisions to define a precision area around the coordinate reading. The extent of overlap of the precision area with context value locations determines the belief allocation that the tag is in that room. If the precision area spreads outside the building, the proportion of the area located outside is reallocated to the overlapping areas that fall inside the building. The application of the dynamic precision quality re-distributes the belief from the original location across the overlapping locations. The static quality parameters (accuracy) are then applied to further modify the mass function based on quality.

Step 7: Identify the Frames of Discernment: All six situations are detected from the same sensors, as shown on the situation DAG. Therefore, they will be placed in the same frame of discernment, as explained in section 5.2 .

[2]No error was assessed along the z-axis because the system always detected the floor level correctly

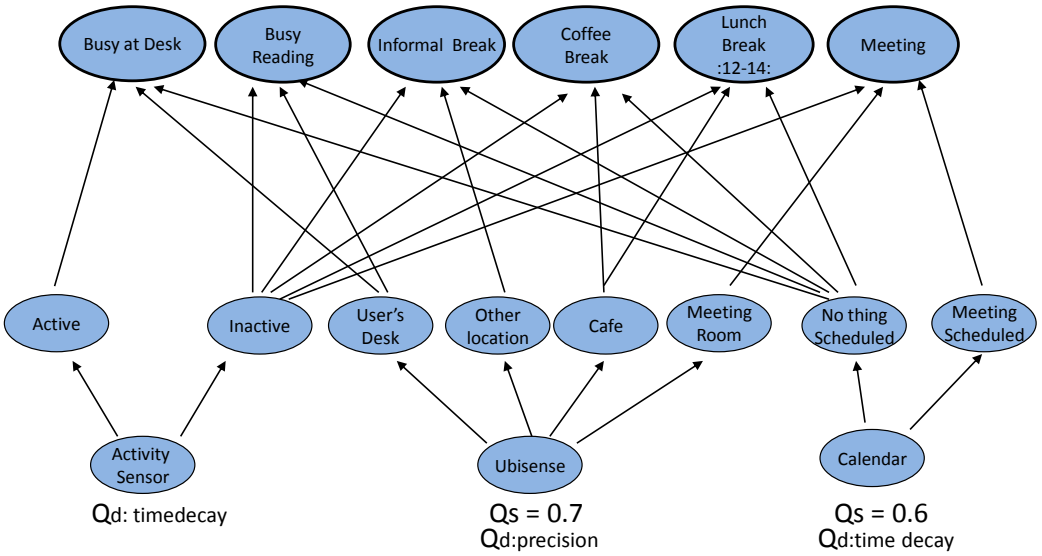


Figure 5.5: Situation DAG for CASL showing sensor, context values and situations

5.5.5 Situation DAG for CASL data set

The situation DAG, established using the seven steps described, is show in figure *5.5*. The situations are derived from the various abstracted contexts of the sensors. There is a particular reliance on the location of the user, as user location is richly informative of their situation, telling us if the user is in the cafe, or at their desk and so on. Both coffee and lunch breaks take place in the cafe. The only detectable difference between coffee and lunch breaks is the time of day that lunch break takes place between 12:00 and 14:00. So we will rely on absolute time of day to differentiate them. In this data set, there is no inference rule frequency used, as each rule is categorical. For example, the user is "always" at their desk when in the situation 'busy at desk'.

5.6 Conclusion

In this chapter, we demonstrated how situation DAGs are established for two real world datasets. This is a critical step in applying the extended Dempster-Shafer reasoning approach because the DAG contains the structure and knowledge needed for the evidence decision network. i.e. to process the belief distribution and decision stages for an envrionment. We noted that knowledge for

the DAG may come from a variety of sources: system developer knowledge, user interviews, user observation, application stakeholders, localised training data and data mining knowledge. In practice, we anticipate that a hybrid approach will be used, with knowledge from different sources populating different parts of the situation DAG.

The situation DAGs for our two data sets establish the evidence processing needed for the evaluation in Chapter 6. The smart home data set contains the temporal patterns needed to evaluate our extensions for transitory evidence. As this is a third party data set, we used a portion (one third) for training data in order to establish the DAG in the absence of detailed domain knowledge for the smart home environment and user. The in-house office data set has sensor quality parameters that we can use to evaluate our quality extensions. We have expert domain knowledge about workings and performance of the sensors, and their roles in tracking situations. We established the situation DAGs for both data sets.

In the next chapter, we will use our data sets and situation DAGs for evaluating our evidence decision network.

CHAPTER

SIX

Evaluation

In the previous chapter, we demonstrated the use of our approach by establishing the situation DAGs for two separate data sets. In this chapter, we will use the situation DAGs as input to the evidence decision network for each of the two data sets. We will assess the accuracy of situation recognition in order to evaluate the evidence decision network approach to situation recognition. In Chapter 1, we hypothesised that

(1) the use of temporal knowledge in evidence will improve recognition accuracy (over using evidence only);

(2) the use of sensor quality in evidence will improve recognition accuracy (over using evidence only).

Temporal extensions to evidential theory are evaluated using van Kastersen's data set. Quality extensions are examined using the CASL data set. The experiments conducted test the precision and recall of using evidential reasoning for situation recognition, both with and without our time and quality extensions.

In Section 6.1, we explain the belief distribution and decision algorithms used for situation recognition in the evidence decision network. The evaluation methodology is described in Section 6.2. The results of temporal extensions using van Kasteren's data set are described in Section 6.3. Section 6.4 describes the quality extensions evaluation using the CASL data set. Section 6.5 examined how alternate fusion rules impact situation recognition results for the two data sets. A summary and discussion is provided in Section 6.6.

6.1 Situation recognition using the evidence decision network

Situation recognition using the evidence decision network is split into two parts, as explained in Chapter 3: *belief distribution* and *decision level.* In chapters 3 and 4, we explained the evidential operations that support the distribution of belief and decision stages. In this section, we explain the algorithms that we have created to process belief distribution and decision stages.These are required to support the implementation of evidence processing in the evidence decision architecture as shown in figure 3.2.

Belief distribution algorithm: The algorithm for belief distribution is shown in figure 6.1. Belief from sensors is distributed at regular time intervals. Each mass function is executed, populating belief levels for all context values in the DAG. The belief from each context event is then propagated to compatible upper nodes for each context event. Once belief for all context events has been propagated upwards, belief for each upper node is fused and then propagated to the next upper node(s). The propagation and fusion process continues until all nodes in the DAG have been processed.

Time extension of evidence algorithm: If frames exist that are deduced from transitory evidence, the life time of the evidence will be extended to tie in with the duration of the frame. For the duration of the frame, all occurring *and* extended evidence is considered. The algorithm for extending evidence lifetime assigns the duration or remainder of the duration for the situation to the occurring evidence, as shown in figure 6.2.

Decision algorithm: Once all belief has been processed, the decision algorithm is applied. The first step is to distribute belief where belief is assigned to combinations of situations, as explained in Section 3.4.3. The distribution is achieved using Smets modified decision rule in equation 3.12. If only one situation is allowed to occur in the environment at any point in time, the algorithm for single situation occurrence is executed, as shown in figure 6.3. This algorithm selects the situation with the highest belief. If two or more situations have the same maximum belief, the algorithm will check which situation exhibits greater certainty.

For environments where more than one situation can occur at the same time, a threshold belief can be used to filter situations, as described by Loke [69] and Clear [16]. In these environments, the decision algorithm for situation co-

```
input: Sensor readings R from sensors S at time t, situation DAG, frames of
discernment F, valid situation combinations S
output: occurring Situation(s)

foreach sensor S do
         execute mass function
         if quality used
                  modify belief distribution
       end
end
foreach context event
    foreach upper node
         if upper node has duration
                  execute time extension for context event
       end
       if inference rule uncertain
           execute propagate uncertain belief to compatible node
       else
          propagate certain belief to compatible node

    end
end
foreach upper node
     process OR (using Max)
     fuse AND  // including time extended belief

end
Execute decision algorithm
Return occurring situation(s)
```

Figure 6.1: Belief distribution algorithm

```
// for a situation S that is part of a frame with duration d, with context event
evidential of S:

if situation S not in progress
// duration d = remaining duration

   context event lifetime = d;
   remaining duration = d;

else
   if situation S in progress
     if context event has belief > 0
        context event lifetime = remaining duration;
     end
   end
end
remaining duration: = remaining duration – timegap;
// timegap = time between timeslices
```

Figure 6.2: Algorithm for extending transitory evidence lifetime

```
input: situations S to be detected
output: occurring situation, O.

// distribute combined belief
foreach combined belief allocation
   apply modified Smets rule;
end

// apply any absolute times used for situations
foreach situation
   if absolute time of situation not= current time
    exclude situation from candidate list
end

// determine which situation(s) occurring by
finding max belief of remaining situations //
if number of situations with max belief = 1
    O = max situation
else
  if number of situations with max belief > 1 // tie
     find minimum uncertainty of these situations
     if number of situation with minimum uncertainty >1
         return default of no decision
     else
        O = minimum uncertainty situation
     end
  else
  // there was no max, belief is zero
      return default or no decision
  end
end
```

Figure 6.3: Decision Algorithm for single situation occurrence

```
input: situations S to be detected, invalid situation combinations, belief
threshold b.
output: occurring situations, O

// distribute combined belief
foreach combined belief allocation
   apply modified Smets rule;
end

// apply any absolute times used for situations
foreach situation
   if absolute time of situation not= current time
    exclude situation from candidate list
end

// apply a belief threshold and remove invalid co-occurrences
select situations with belief > t
  if invalid situation combination exist in selected situations:
     sort selected situations from lowest to highest belief
     while (more situations or invalid combinations complete)
         if situation is part of occurring invalidation combination
            remove situation from selection
            check if invalid combinations complete
         end
      end
   end
return situations in selection
```

Figure 6.4: Decision algorithm for situation co-occurrence

occurrence is executed, as shown in figure 6.4. This algorithm must consider the invalid combinations of situations so that only situations that can occur together are returned. This is done by dropping situations with the lowest belief, if they are part of an invalid combination, where all the situations in the combination have exceeded the threshold. Invalid combinations can be hand crafted, or can be detected using an automated process, based on Ye's approach [119]. This work defines conflicting context values, which cannot co-occur, such as 'having a shower' in the bathroom location cannot happen at the same time as 'eating dinner' in the kitchen location. By checking which context values conflict, situations that are impossible to occur together can be determined. The setting of the threshold will be environment specific. If applications execute high risk behaviour, the threshold should be set to prevent situations with insufficient belief levels [69]. Also, the quality of the sensors and the certainty of the inference rules will determine the level of belief that is possible for situations to be achieved in the first place. Highly uncertain environments will need to set lower thresholds in order to detect situations.

6.1.1 Worked examples using evidence decision network

In order to explain the use of the evidence decision network, we describe two examples of how the network is used: (1) from the van Kasteren data set and (2) from the CASL data set.

Van Kasteren worked example

In the van Kasteren data set, the data has been time sliced into time stamped sensor readings for each of the 14 sensors. Taking an example from the timesliced data, at a particular time 27/02/2008 19:03:00, the freezer sensor is set to 1, all other sensors are set to 0. This means that the only sensor that triggered during that timeslice was the freezer door sensor. This timeslice of sensor readings is processed through the two stages of the EDN (1) belief distribution and (2) decision stages as follows:

(1) Belief Distribution stage

The belief distribution algorithm described in Figure 6.1 is applied to the set of sensor readings for the timeslice. First, the mass functions for each sensor are executed:

- The freezer sensor is set to '1' so its mass functions assigns evidence as mass 1,0,0 to its frame hypotheses (freezerUsed, freezerNotUsed, Θ).
- The mass functions for the remaining 13 sensors assign their belief to their context values as (0,1,0) to the relevant sensor context values with frame hypotheses: (sensorUsed, sensorNotUsed, Θ).

Propagate evidence for each context event, each of which has frame of hypotheses (sensorUsed, sensorNotUsed, Θ):

1. The situation DAG for van Kasteren as shown in figure 5.3 shows that the freezer used is indicative of the 'prepare dinner' situation, with a certainty of 0.n,where n is a uncertainty weighting between 0 and 1. Therefore, evidence can be transferred from the freezer context value frame of discernment to the 'prepare dinner' frame using compatibility relations from equations 3.5 and 3.6 to define the mapping.
2. Situation duration is used to extend transitory evidence lifetime in the van Kasteren data set. As per the belief distribution for time extended evidence algorithm in Figure 6.2, the 'prepare dinner' situation has a du-

ration, so the lifetime of the context event evidence 1,0,0 for hypotheses freezerUsed, freezerNotUsed, Θ is set to the 'prepare dinner' duration.

3. The propagation of uncertain evidence between the frames is carried out using equations 3.7 and 3.8. As a result, the belief assignments to the 'prepare dinner' frame of discernment (prepareDinner, notPrepareDinner, Θ) is (0.n, 0, 1-n) respectively from the freezer context value. The other context values that are indicative of the 'prepare dinner' situation (cup, fridge, plate, groceries cupboard, pans cupboard) all have their belief assigned to uncertainty as a result of the sensors not triggering.

4. The evidence captured for the remaining 13 context values are propagated via compatibility mapping and evidence propagation to the various situations on the DAG using the inference paths.

5. For each situation, the evidence is fused. The 'prepare dinner' has 6 context values each assigning all their belief to uncertainty, and the freezer context values assigning its belief as (0.n, 0, 1-n) to the frame. These seven evidence sources are fused using Murphy's time extended evidence fusion rule in equation 4.2.

6. The remaining duration of the 'prepare dinner' situation is decremented by 1 minute.

7. All situations now have a belief level: The 'prepare dinner' situation is the only situation with belief assigned, with 0 for all the remaining situations.

(2) Decision stage

1. The decision algorithm for single situation occurrence as described in Figure 6.3 is used as only one situation can be occurring in this environment at any point in time.

2. As per the decision algorithm, the absolute time of situations is applied. The timeslice occurs in the evening (19:03) and the 'prepare dinner' situation has an absolute time set to evening, so it remains in the candidate list.

3. The situation with the max belief is 'prepare dinner' so it is deemed to be occurring.

CASL Worked Example

The second worked example is from the CASL data set. In this data set, there are three sensors: the Ubisense location sensor, the desktop activity sensor and the calendar sensor. The data is timesliced into one minute intervals.

Taking a time slice at 10.06 on Friday from the CASL data set, the ubisense sensor co-ordinate reading of (8.24 13.78 2.21) maps to the informal meeting room. The activity sensor does not trigger, and has not triggered for the previous three timeslices. The calendar sensor indicates that a meeting is scheduled at that time.

(1) Belief distribution Stage

1. The mass function for each of the sensors distributes their belief using the belief distribution algorithm in figure 6.1.

2. Quality parameters are used as defined for the CASL data set in Table 5.3, so are used to modify belief as follows:

   (a) The ubisense sensor maps to the 'informal meeting' room so assigns belief to the 'meeting room' hypothesis. The *dynamic* quality parameter that defines a precision area around the coordinate point reading is then applied, and this precision area stretches to include some of the hallway (0.17). Belief is thus reassigned with 0.83 to the meeting room and 0.17 to the hallway using the dynamic quality equations 4.4 and 4.5. The *static* quality parameter of 0.7 is then applied to represent the inaccuracy of ubisense using equation 4.3, further re-assigning belief as meeting room 0.58, hallway 0.12, and uncertainty of 0.3. The sequence of applying dynamic first, and then static quality is dictated by equation 4.6.

   (b) The activity sensor follows the dynamic quality time decay function indicated in Table 5.3 so assigns belief as 0.25 active, and 0.75 inactive. There is no static quality issue with the activity sensor.

   (c) The calendar sensor is set to meeting, so assigns its belief as 1 to 'meeting scheduled'. The dynamic quality parameter indicates a fuzzy function that indicates the user is usually up to 10 minutes late. The time of 10:06 is 6 minutes into the scheduled meeting time, so after applying the dynamic quality function, the mass of belief to

indicate a meeting has degraded to 0.7, with 0.3 that a meeting is not occurring. A static quality parameter of 0.6 applied which readjusts these belief levels, as per equation 4.3. The reassigned beliefs are meeting 0.42, no meeting 0.13 , and uncertainty of 0.45.

3. The belief values are then mapped to the situations using the inference paths indicated in the CASL situation DAG in Figure 5.5.

4. Evidence is not time extended in the CASL data set so the time extension algorithm is not needed.

5. Evidence for each situation is *fused* using Dempster Shafer's rule of combination in 3.3

Decision Stage

1. The decision algorithm for single situation occurrence as described in Figure 6.3 is used as only one situation can be occurring in this environment at any point in time.

2. Smet's rule is applicable to the CASL data set because evidence is assigned to the combinations of situations. The calendar sensor has propagated evidence of 0.13 to 'not meeting' which is a combination of all situations apart from meeting i.e. busy at desk, busy reading, coffee break, informal break, lunch. Our modified Smet's rule in equation 3.12 is used to distribute belief to the six *individual* situations.

3. After distributing the combined belief, the 'meeting' situation has the highest belief, so is deemed to be occurring.

6.2 Evaluation methodology for situation recognition

We will use our two data sets to infer situations using our evidence decision network. Our aim is to test the following hypotheses:

1. Temporal extensions to Dempster-Shafer theory will improve situation recognition accuracy, where temporal features exist in the environment situations;

2. Quality extensions to Dempster-Shafer theory will improve situation recognition accuracy, where sensor quality is an issue in the environment.

To test these hypotheses, our evaluation will answer the following questions.

	Question	Data set
1	How accurate is the evidence decision network for situation recognition?	Both
2	Do temporal extensions to the evidence decision network improve situation recognition?	van Kasteren
3	Do the quality extensions to the evidence decision network improve situation situation recognition?	CASL

In addition, we wish to compare our approach with other situation recognition techniques used in the field, in order to check that our approach can produce viable results. The purpose of the comparison is not to say whether our approach is quantitatively better or worse than other techniques. We have already explored the differences in functionality between our approach and other techniques (both specification and learning) in Chapter 2. We are simply testing whether our approach is producing results that can compare reasonably well in terms of other commonly used techniques. For this reason, we choose two classic machine learning techniques (Naïve Bayes and J48 decision tree) that have been used by other researchers for comparison with their own recognition techniques, both learning [68, 119] and specification techniques [125] . We do this comparison using both the van Kasteren and CASL data sets.

6.2.1 Evaluation parameters

We employ three widely used statistical classification parameters to assess accuracy of situation recognition. (1) Precision is the ratio of the times that a situation is correctly inferred ($N_{infCorr}$) to the times that it is inferred (N_{inf}), i.e. it is a measurement of exactness of situation recognition. (2) Recall is the ratio of the times that a situation is correctly inferred to the times that it actually occurs in the data set (N_{act}). It is a measure of how completely the situation has been recognised throughout its occurrences. (3) F-measure is the weighted mean of

n_{00}= the number of examples misclassified by both recognition techniques	n_{01}= the number of examples misclassified by technique A but classified correctly by technique B
n_{10}= the number of examples classified correctly by technique A but misclassified by technique B	n_{11}= the number of examples classified correctly by both techniques

Table 6.1: McNemar's Results table

precision and recall and is used to summarize inference accuracy. It is a useful way to balance precision and recall because improvements in precision may be at the expense of recall and vica versa. The equations for these three standard measures are:

$$Precision = N_{infCorr}/N_{inf} \tag{6.1}$$

$$Recall = N_{infCorr}/N_{act} \tag{6.2}$$

$$F = \frac{2 \times precison \times recall}{precision + recall} \tag{6.3}$$

For van Kasteren's data set, we generate results for multiple folds of data during cross validation. We use averages for precision, recall and f-measure. These averages are calculated as the sum of the relevant measure for each fold, divided by the number of folds.

6.2.2 Statistical Significance

We hypothesise that (1) the use of temporal knowledge in evidence will improve recognition accuracy over using evidence only and that (2) the use of quality in evidence will improve recognition accuracy over evidence only. To validate our results, we need to test whether significant differences exist using evidence only, versus using quality and time. McNemar's test is used to calculate confidence levels when comparing the performance of two systems [29, 88]. Given two systems A and B, for each example x which is an element of T, where T is the set of test queries, a contingency table is constructed as follows, using the definitions in table 6.1:

The total number of timeslices to be tested is $n = n_{00} + n_{01} + n_{10} + n_{11}$. If there is no performance difference between the two recognition techniques (null hypothesis), then $n_{10} = n_{01}$. McNemar's test is based on a χ^2 (chi squared) test for goodness of fit which compares the distribution of counts expected under the null hypothesis to the observed counts. χ^2 is calculated as per equation 6.4 from [29]:

$$\frac{(|n_{01} - n_{10}| - 1)^2}{n_{01} + n_{10}} \tag{6.4}$$

This statistic is distributed (approximately) as χ^2 with one degree of freedom. McNemar's test has a lower probability of incorrectly detecting a difference when no difference exists but it also possesses good discriminative power (the ability to detect a difference where one does exist) [29].

We will populate McNemar contingency tables when assessing whether temporal and quality extensions to the evidence decision network are statistically significant.

6.2.3 Cross validation

For van Kasteren, as a third party data set, we must generate knowledge by holding back some data for training. Cross validation is the process whereby the data is partitioned into *n* folds, with each of the *n* folds used for testing, while the remainder is used for training. The process is repeated *n* times [113]. As described in Chapter 5, we split the data set into thirds and use one third for training with the remainder for testing, cross validated. Existing published results from Ye [119] and van Kasteren [109] use the 'leave one day out' technique, whereby each day of the 28 day data set is in turn used for testing, with the remainder 27 days used for training. In order to compare the evidence decision network with these results, we also apply the 'leave one day out' technique so that the same methodology is used to produce results for comparison.

In the CASL data set, we have domain knowledge about the inference rules and sensor mass functions for the environment. Therefore, the full data set can be used as a test set for testing situation recognition accuracy using the evidence decision network. To compare with other techniques, we use 10-fold cross validation for the comparison learning techniques. This is a standard technique for measuring the error rate of a learning technique [113]. The data

is divided into 10 folds, and the process repeated 10 times.

6.2.4 Experimental set-up

To test our evidence based approach to situation recognition, we needed to develop our own software as no tools exist to meet our requirements for processing and propagating evidence. We developed our own software for evidence decision network processing using java JDK 1.6. The choice of software was based on its support for OO development techniques. This software executes the belief distribution and decision steps of the evidence decision network. That is, the software reads in time sliced sensor readings, executes the mass functions for all sensor reading values in the timeslice, propagates evidence, distributes belief and executes the decision stage. To test learning techniques on our two data sets, we used the Waikato Environment for Knowledge Analysis (WEKA) workbench, which is a standard tool amongst the machine learning community.

6.3 Situation Recognition on van Kasteren's Data set

Our experiments using van Kasteren's data set are based around the three questions:

- Experiment 1: How accurate is the evidence decision network for situation recognition? This tests situation recognition without using absolute time or transitory evidence in order to establish a baseline;
- Experiment 2: Do temporal extensions to the evidence decision network improve situation recognition? Our temporal extensions cater for absolute time and situations with transitory evidence. We test each of these separately so that their impact can be assessed.
- Experiment 3: How well does the evidence decision network perform when compared to other recognition techniques? We compare against Naïve Bayes and J48 Decision Tree classifiers.

	precision	recall	f-measure
leave house	0.78	1	0.87
use toilet	0.64	0.42	0.50
take shower	0.35	0.26	0.29
go to bed	0	0	0
prepare breakfast	0.40	0.27	0.30
prepare dinner	0.80	0.20	0.31
get drink	0.36	0.75	0.45

Table 6.2: Precision, recall and f-measure of situation recognition on van Kastersen's data set (for evidence without time included)

Situations	leave house	use toilet	take shower	go to bed	prep. break't	prep dinner	get drink
leave house	**0.998**	0.001	0.001	0.0	0.0	0.0	0.0
use toilet	0.358	**0.419**	0.223	0.0	0.0	0.0	0.0
take shower	0.723	0.021	**0.256**	0.0	0.0	0.0	0.0
go to bed	0.982	0.009	0.008	**0.0**	0.0	0.0	0.0
prep break't	0.365	0.028	0.030	0.0	**0.274**	0.116	0.187
prep dinner	0.589	0.013	0.004	0.0	0.127	**0.198**	0.069
get drink	0.162	0.023	0.046	0.0	0.012	0.011	**0.746**

Table 6.3: Confusion matrix of inference on van Kasteren's data set

6.3.1 Experiment 1: Situation recognition accuracy without temporal knowledge

In the first experiment, we examine the situation recognition results on van Kastersen's data set using the evidence decision network. We use limited (one third) training data, with cross validation. Temporal knowledge (absolute time and transitory evidence) are not included. Table 6.2 shows the average precision, recall and f-measure. The confusion matrix in table 6.3 shows a breakdown of situation recognition errors in more detail. In this matrix, the occurring situations are shown in the left most columns. Each row provides a detailed breakdown of how each situation was recognised correctly, or incorrectly as another situation. The correct inferences are shown in bold along the diagonal. For example, the first row on "leave house" can be read as: 0.998% of the occurring 'leave house' timeslices were correctly inferred. 0.001% of them were classified incorrectly as 'use toilet' and 0.001% as 'take shower'.

Looking at table 6.2, the 'go to bed' situation is not recognised at all. Looking at the confusion matrix, it is classified almost entirely as 'leave house' . This is because these two situations use the same sensors and context values as ev-

idence, so they cannot be distinguished. Absolute time (day or night) is the only distinguishing evidence but is excluded from this experiment. Its value in distinguishing these two situations will be shown in experiment 2. Recognition accuracy of the other five situations ranges from 0.29 for 'take shower' to 0.5 for 'use toilet'. Looking at the confusion matrix, each of these five situations are incorrectly inferred as 'leave house' to some degree. The lower accuracy situations of 'take shower' and 'prepare dinner' have a longer duration than other situations such as 'get drink' and 'use toilet'. Lack of sensor activity is indicative of the 'leave home' situation. This affects the longer duration situations such as 'prepare dinner' (59%) and 'take shower' (72%) which have longer gaps in their evidence than the short situations of 'get drink' and 'use toilet'. In these situations, the sensor events happen over shorter periods of time, with less gaps in the evidence. This suggests that extending evidence should boost longer duration situations such as 'take shower' and 'prepare dinner' than shorter duration situations such as 'get drink'. This will be examined in the next experiment.

6.3.2 Experiment 2: Situation recognition accuracy with absolute time and time extended evidence

We then included temporal knowledge into the inference process. This was done in two steps so that we could examine the contribution of both type of temporal knowledge 1) inclusion of absolute time and 2) inclusion of both absolute time and time extended evidence. Absolute time was incorporated into the inference process using the time patterns listed in table 5.2. Looking at this table, all situations except for 'use toilet' and 'get drink' are associated with an absolute time.Figures 6.5 (precision), 6.6 (recall) and 6.7 (f-measure) show the situation recognition results using the evidence decision network with and without absolute time on van Kastersen's data set.

Precision improves for all but two of the situations, 'toileting' and 'get drink'. Neither of these situations have an identifying time of occurrence so do not benefit from the inclusion of absolute time. For recall, all situations improve with the exception of 'leave home'. Without the use of absolute time, all scenarios with no sensor activity were classified as 'leave home'. But with the inclusion of absolute time, 'sleeping' can now be identified separately from 'leave home', with a resultant small erosion of 'leave home' recall. Looking at f-measure, the use of absolute time improves the inference accuracy for the five

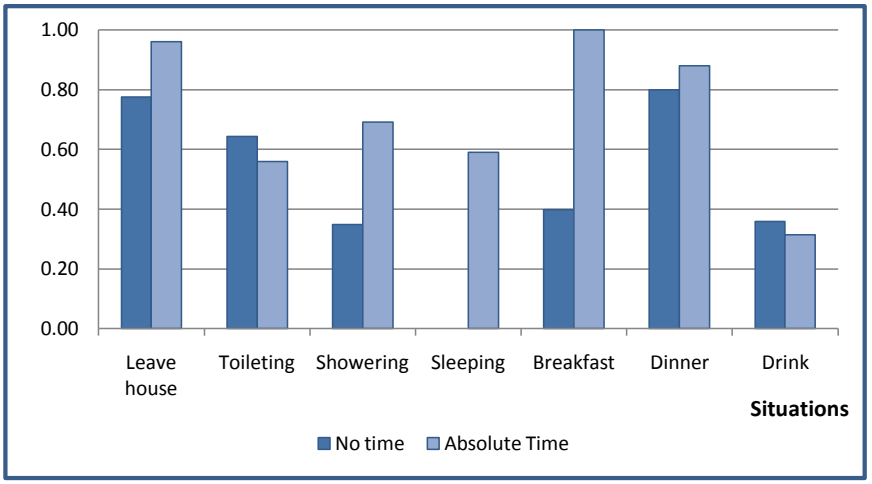


Figure 6.5: Precision for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set

Situations	leave house	use toilet	take shower	go to bed	prep break't	prep dinner	get drink
leave house	**0.832**	0.002	0.00	0.165	0.0	0.001	0.0
use toilet	0.185	**0.587**	0.061	0.168	0.0	0.0	0.0
take shower	0.629	0.076	**0.201**	0.094	0.0	0.0	0.0
go to bed	0.041	0.017	0.001	**0.941**	0.0	0.0	0.0
prep breakf't	0.363	0.039	0.018	0.035	**0.324**	0.0	0.221
prep dinner	0.589	0.018	0.0	0.0	0.0	**0.314**	0.079
get drink	0.077	0.059	0.0	0.023	0.0	0.048	**0.794**

Table 6.4: Confusion Matrix on van Kasteren's data set using absolute time

situations that have an absolute time. 'Leave home' and 'go to bed' can now be distinguished because of their time pattern of day and night time occurrence, respectively. Therefore, the f-measure for 'go to bed' has jumped from 0 to 0.73. 'Prepare dinner' and 'prepare breakfast' are detected from similar sensor events. Both improve when absolute time is used because they occur at mutually exclusive times of the day. Looking at the confusion matrix in table 6.4, these two situations are no longer confused with each other, as was happening when absolute time was not used. All situations continue to have a high proportion of their occurrences confused with the 'leave home' situation, with some confusion now occurring with the 'go to bed' situation now that it is recognised as occurring at nighttime.

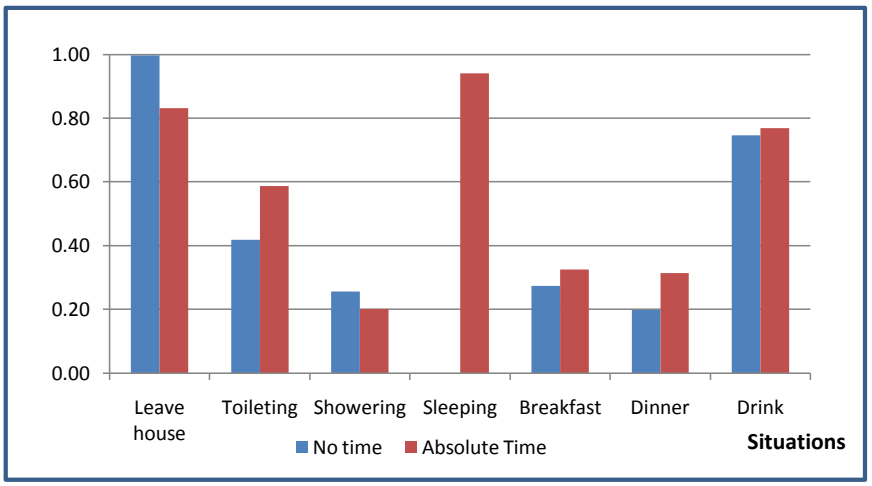


Figure 6.6: Recall for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set

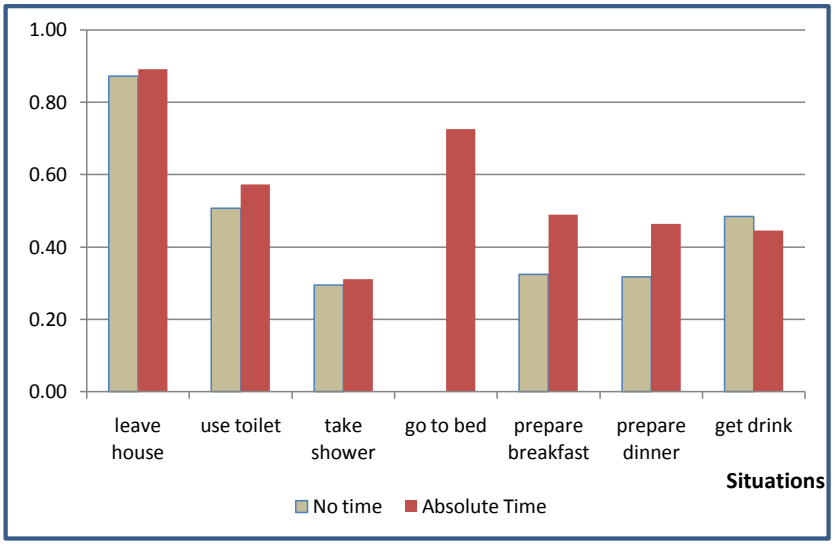


Figure 6.7: F-measure for evidence decision network with 1) no time and 2) absolute time using van Kasteren's data set

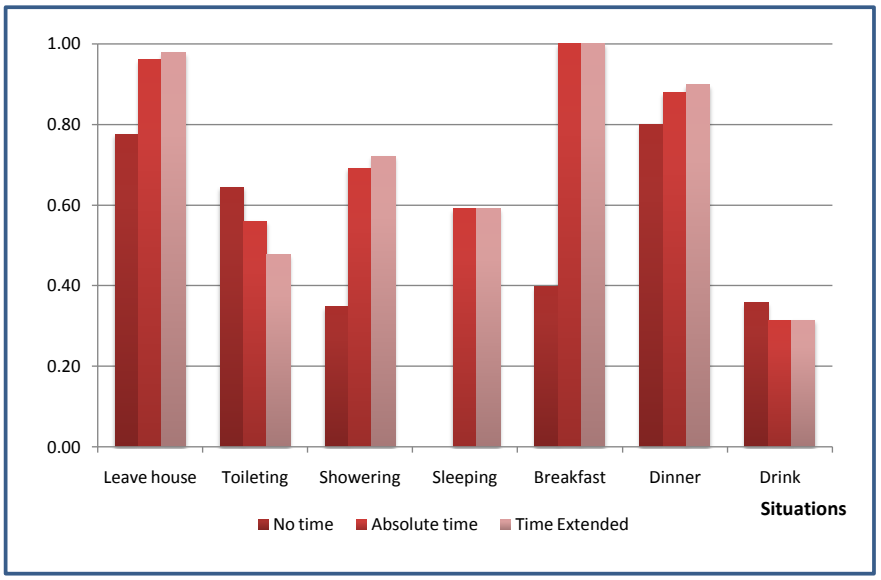


Figure 6.8: Precision for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set

For the second part of the experiment, we then added time extension of evidence into the evidence decision network. Five of the situations are derived from transitory evidence: Durations are applicable for 'prepare breakfast', 'prepare dinner', 'get drink', 'use shower' and 'use toilet' as each of their context events can be spread over time. No sensor is usually fired during 'leave home' and 'go to bed' situations so no time extension of evidence is used for these situations. Durations are calculated as the average of the situation duration from the training data folds, as explained in section

4.2.2. Figures 6.8 (precision), 6.9 (recall) and 6.10 (f-measure) compare the inference results of using no time, absolute time and absolute time plus extended evidence. Looking at precision, three situations improve, two remain the same, and 'toileting' situation precision is reduced. Extended time evidence has limited improvement for the 'toileting' activity because it has the same sensor activations as 'showering' situation but does not benefit from an absolute time. Recall for all situation occurrences that use transitory evidence improves because gaps between evidence have been removed. Therefore, annotated timeslices with no sensors triggering can be correctly recognised. Using f-measure, when time extension of transitory evidence is also used, recognition accuracy improves for four out of the five enduring situations. Time extension is not used for 'leave house' and 'go to bed' situations, and as expected their inference accuracy is almost identical. For the remaining five time-extended situations, the biggest improvements is shown in 'take shower', 'prepare break-

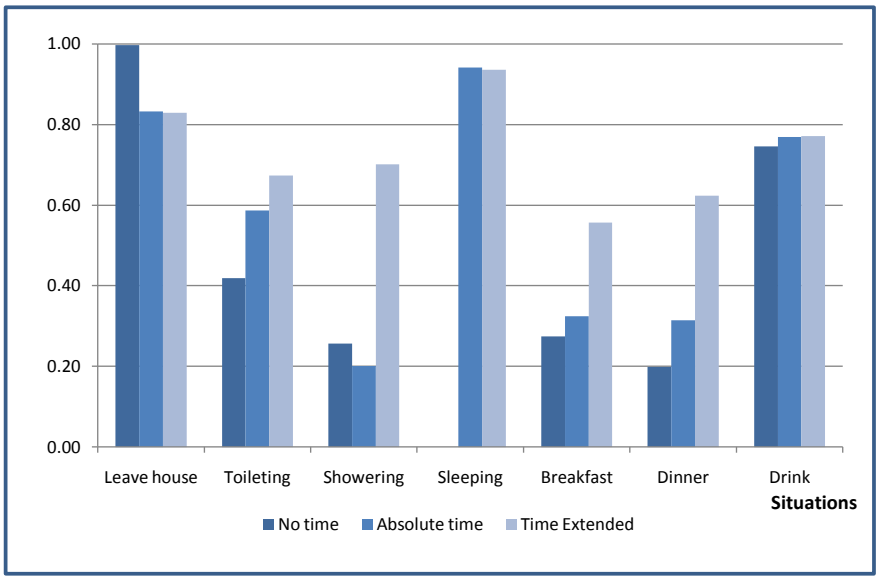


Figure 6.9: Recall for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set

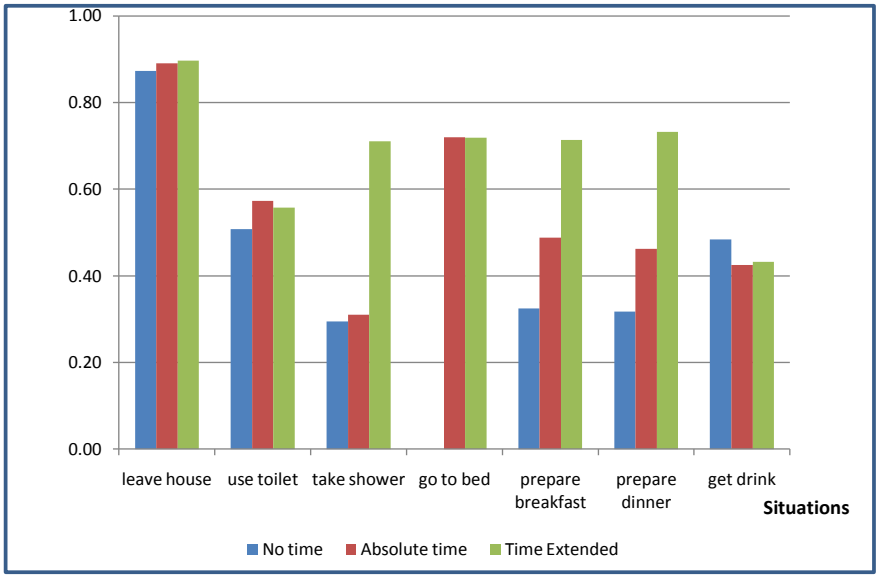


Figure 6.10: F-measure for 1) no time, 2) absolute time and 3) absolute time and time extended evidence using van Kasteren's data set

Situations	leave house	use toilet	take shower	go to bed	prep break't	prep dinner	get drink
leave house	**0.829**	0.003	0.001	0.165	0.0	0.001	0.001
use toilet	0.137	**0.673**	0.077	0.108	0.0	0.003	0.0
take shower	0.094	0.123	**0.701**	0.082	0.0	0.0	0.0
go to bed	0.037	0.025	0.002	**0.936**	0.0	0.0	0.0
prep break't	0.172	0.091	0.035	0.0	**0.557**	0.024	0.121
prep dinner	0.245	0.038	0.0	0.0	0.0	**0.624**	0.093
get drink	0.034	0.069	0.0	0.023	0.0	0.103	**0.771**

Table 6.5: Confusion matrix on van Kastersen's data set including absolute time and time extended evidence

	No Time	Absolute time	Time extended and absolute
F-measure	0.40	0.55	0.68

Table 6.6: Comparison of average f-measure for evidence decision network with no time, absolute time and time extended

fast' and 'prepare dinner'. These activities are longer in duration than the 'get drink' and 'use toilet' situations, so their evidence is sparser throughout the duration with longer gaps where nothing happens. Therefore, they benefit more from the extension of their transitory evidence. The 'use toilet' situation recognition actually decreases very slightly with the use of time-extended evidence. This is because the sensors used in 'use toilet' overlap with those for 'take shower' and the two situations were often performed sequentially. Looking at the confusion matrix in Table 6.5, the extent to which the enduring situations are confused with 'leave house' and 'go to bed' has reduced. i.e. the static periods where nothing happens is reduced.

The impact of the evidence decision network with temporal extensions is summarized in table 6.6. This shows average F-measure for all situations when no time is used in reasoning, when absolute time is used, and when both time extension and absolute time are used. F-measure improves by 70% with the use of both time reasoning techniques. The data set has strong time patterns so we expected the inclusion of this temporal knowledge to improve our results.

Our hypothesis is that the evidence decision network with temporal extensions improves inference results over the evidence decision network only. To check whether the difference in our results is significant, we construct McNemar's contingency table as described in Section 6.2.2. The table, shown in Appendix A, Section 1, contains the classification result totals for the evidence decision network only versus the evidence decision network using absolute

time and time extended time. Using McNemar's equation 6.4, we find that the difference between the evidence only and temporal evidence classification approaches were statistically significant to the 99.99% level.

6.3.3 Experiment 3: Comparison with learning techniques

In this experiment, we compare our evidence decision network results with two class machine learning techniques, Naïve Bayes Classifier and J48 Decision Tree. The purpose of this comparison is to examine whether we can get comparable results using the evidence decision network to learning techniques, as a 'sense check' that our evidence theory-based approach can perform reasonably in comparison to other recognition approaches. The experiments were run in two cross validation modes (1) using limited training data (one third, cross validated) and (2) using a 'leave one day out' cross validation approach. When limited training data was used, the evidence decision network with temporal extensions clearly out performed both Naïve Bayes and J48 as shown in figure 6.12. Both evidence decision network approaches incorporate domain knowledge so their performance on limited training data is not significantly improved on the 'leave one day out' technique. The learning techniques, on the other hand, rely totally on training data for their results, performing much better on the 'leave one day out' approach than on limited training data, as shown in figure 6.12.

It is interesting to compare the evidence decision network results of limited training data (with greater reliance on domain knowledge) with the learning results from the more comprehensive leave-one-day-out training technique, as per the bolded figures in table 6.7. An evidence based approach will typically be used when training data is limited or not available, so the results from limited training data are encouraging - the technique with temporal knowledge exceeds the learning techniques, and without temporal knowledge still performs well compared to learning techniques. Since evidence theory is suited to the incorporation of domain knowledge, this result is encouraging. In particular, temporal knowledge of absolute times and transitory evidence for enduring situations can be incorporated and used to improve recognition accuracy. Evidence theory will be useful when training data is not easily available and where domain knowledge can be gleaned from expert knowledge and user knowledge. These sources can be used to obtain inference knowledge in a piecemeal approach, with users providing information on absolute times, sit-

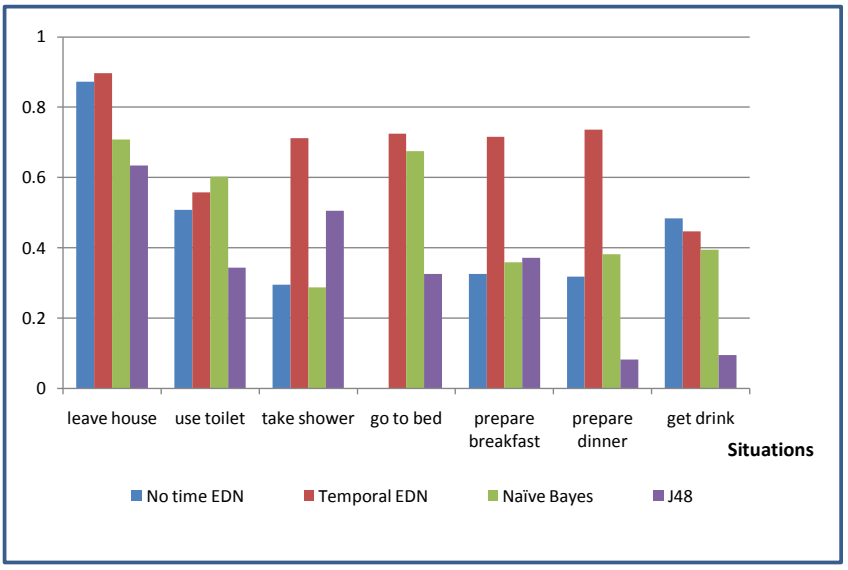


Figure 6.11: Comparison of F-measure between Time extended Evidence, Naïve Bayes and J48 Decision Tree with one third training training data

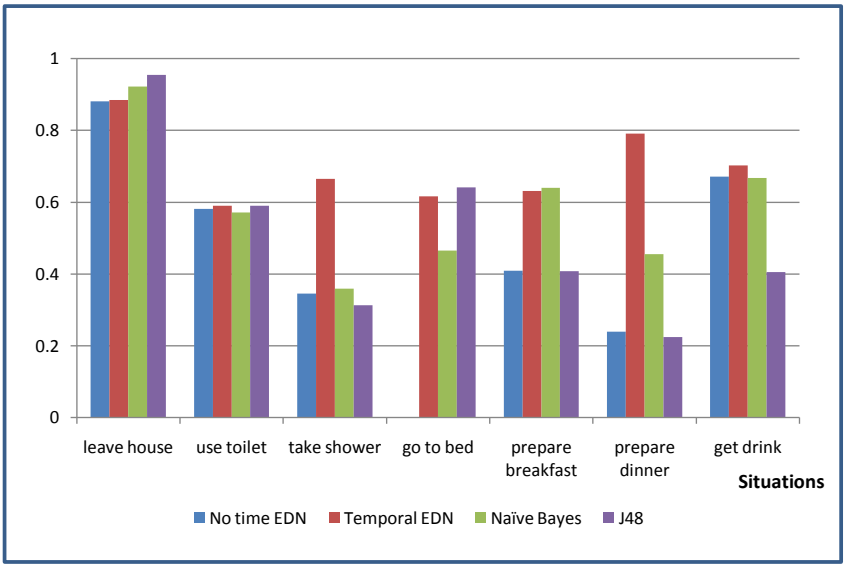


Figure 6.12: Comparison of F-measure of time extended evidence, Naïve Bayes and J48 using Leave One Day Out cross validation.

	EDN without time	EDN with temporal	Naïve Bayes	J48 Decision Tree
One Third Training	**0.40**	**0.68**	0.49	0.34
Leave One Day Out	0.45	0.70	**0.58**	**0.51**

Table 6.7: F-measure comparison of Time extended evidence, Naïve Bayes and J48 for one third and Leave One Day Out cross validation

uation descriptions and durations, and experts providing knowledge of sensor mass functions and sensor quality.

6.3.4 Experiment 4: Comparison with published results

Van Kasteren [109] and Ye [119] have both published inference results based on using a 'leave one day out' cross validation technique. Van Kasteren evaluated Hidden Markov Models (HMM) and Conditional Random Field (CRF) recognition techniques on the data set. Each technique was tested in three sensor representations:

- raw sensor representation which returns a 1 when a sensor fires;
- change point sensor representation which returns a 1 when a sensor reading is changed;
- last observation sensor representation which returns a 1 if a sensor continually fires, and gives a 0 when a different sensor fires.

Van Kasteren *et al*. only published the raw sensor representations in which all the sensor values are only 1, so we compare with raw sensor representation results only. To measure accuracy, they use a class accuracy measure calculated as average percentage of correctly recognized timeslices per situation:

$$class = \frac{1}{C} \cdot \sum_{c=1}^{C} \left\{ \frac{\sum_{n=1}^{N} (inferred_c(n) = true_c(n)}{N_c} \right\} \tag{6.5}$$

N is the total number of time slices, C is the number of classes and N_c is the total number of time slices for class c. We calculated van Kasteren *et al.*'s class accuracy measure calculation for our time-extended evidence using the same 'leave one day out' cross validation approach as Van Kasteren. A summary of the comparative results is shown in table 6.8.

Ye Lattices	evidence decision network with temporal	van Kasteren HMM	van Kasteren CRF
88.3%	69%	49.2%	44.6%

Table 6.8: Comparison of class accuracy of temporal EDN with published results from Ye [119] and van Kasteren [109]

Ye's situation lattices approach yields a class accuracy of 88.3% using raw sensor representations and the 'leave one day out' cross validation technique. This is higher than the results from the temporal evidence framework (69%) and van Kasteren *et al.*'s HMM results (49.2%). Ye's lattice method includes absolute time in the inference method, and combines both training and domain knowledge. However, timeslices in which no sensor changes take place are excluded in Ye's results, but are included in the evidence decision network and in van Kastersen's work. These 'inactive' timeslices are hard to infer because of the lack of sensor information so the data set is likely to yield improved results to some degree.

HMMs consider the sequence in which situations occur. They do not consider absolute times (unless explicitly captured in the training data) or durations of situations. The relative performance of HMMs and CRFs for the van Kasteren data set are explored further in [109]. Van Kastersen achieves an accuracy of 79.4% using a richer sensor representation, 'changepoint plus last sensor' representation as described in [109].

6.3.5 Discussion of temporal evidence decision network results

The evidence decision network enhanced with temporal knowledge significantly improves situation recognition accuracy, over the basic evidence decision network on van Kasteren's data set. Absolute times are most useful where it is the only discriminating factor between two or more situations, as was the case with the 'leave house' and 'go to bed' situations. It is also useful where situations have some sensor overlap and have different absolute times, as with 'prepare breakfast' and 'prepare dinner'. Time extension of transitory evidence proved useful for situations with gaps between evidence. The longer the gaps (such as 'prepare dinner'), the greater the impact time extended evidence can provide.

The comparison with published results is interesting. Van Kasteren's results using HMMs are worse - so perhaps the situation duration is more important as an identifying factor in recognising situations in this data set than the sequence of situations as used by HMMs. Also, the evidence decision network approach benefits from the incorporation of domain knowledge.

For the comparison with classic machine learning techniques, the purpose is to sense check our evidence decision network results. Clearly, if training data is available, a pure learning technique will, in theory, be preferable. But if training data is not available or just limited 'pockets' of localised training data is available, then the evidence decision network approach is a theoretically sound alternative. This is indicated in our comparison of f-measures in table 6.7.

Further discussion of the overall evidence decision network approach is provided in Section 6.6.

6.4 Situation recognition on CASL data set

For the CASL data set, we used domain knowledge to establish the situation DAG, where the situation DAG is illustrated in Section 5.4.4.1, figure *5.5*. Because of the availability of domain knowledge, we can use the full data set for testing as no data is needed for training. Our hypothesis is that the use of quality parameters for imperfect sensors will improve the situation recognition accuracy using the evidence decision network (over not including quality parameters). We test this hypothesis as follows:

- Experiment 1: How accurate is the evidence decision network for situation recognition? This tests situation recognition without quality in order to establish a baseline;
- Experiment 2: Does situation recognition accuracy significantly improve when quality parameters are used?

The remaining experiments give greater insight into the use of the quality parameters, as follows:

- Experiment 3: How do the quality attributes for individual sensors contribute to the situation accuracy results?

	precision	recall	f-measure
busy at desk	0.96	0.61	0.75
busy reading	0.42	0.47	0.44
informal break	0.59	0.68	0.63
coffee break	0.44	0.54	0.48
lunch break	0.53	0.68	0.60
at meeting	0.76	0.64	0.69

Table 6.9: Precision, recall and f-measure of situation recognition on CASL data set

- Experiment 4: How sensitive are the recognition results to changes in the sensor quality parameter values?
- Experiment 5: How well does the evidence decision network with quality perform when compared to other recognition techniques? We compare against Naïve Bayes and J48 Decision Tree classifiers.

As described in Chapter 5, we merge and time slice the data from our three sensors so that the data is represented as annotated timesliced sensor readings. Within each time slice, quality values are provided for each sensor. The evidence decision network process uses these values when testing situation recognition with quality, and ignores them when excluding quality.

6.4.1 Experiment 1: Situation recognition accuracy without quality parameters

First, we examine situation recognition accuracy when quality parameters are not used in the evidence decision network. i.e. all sensors are equally trusted, with a static quality of '1' and no dynamic quality applied. The precision, recall and f-measures by situation are shown in table 6.9. The confusion matrix is in table 6.10. The 'undetermined' column consists of timeslices where situations were tied on maximum belief in the decision algorithm. Various default options can be applied in this case. For example, if the tying situations can be summarised - such as 'busy at desk' and 'busy reading' can be summarised by 'busy', then the summary situation can be applied.

Situations	busy at desk	busy read-ing	informal break	coffee break	lunch break	at meet-ing	undeter'd
busy at desk	**0.61**	0.0	0.04	0.01	0.03	0.01	0.29
busy reading	0.29	**0.47**	0.24	0.0	0.0	0.0	0.0
informal break	0.07	0.13	**0.68**	0.03	0.02	0.0	0.07
coffee break	0.07	0.07	0.21	**0.54**	0.0	0.0	0.11
lunch break	0.04	0.03	0.12	0.0	**0.68**	0.0	0.13
at meeting	0.0	0.0	0.26	0.0	0.0	**0.64**	0.10

Table 6.10: Confusion matrix for CASL data set when quality not used

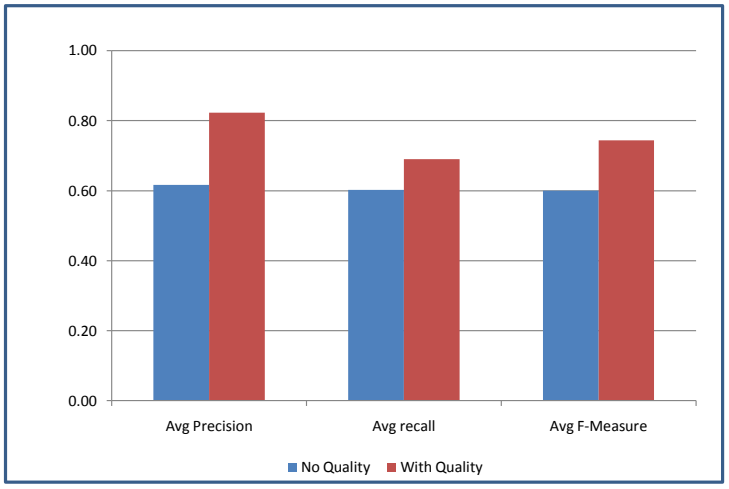


Figure 6.13: Comparison of average precision, recall and f-measure for the evidence decision network with and without quality for the CASL data set

6.4.2 Experiment 2: Situation recognition accuracy with quality parameters

In this experiment, we applied quality parameters to each of three sensors, using the quality parameter values listed in 5.3. Figure 6.13 shows the average precision and recall for the evidence decision network with and without quality.

Both precision and recall improve when quality values are used to modify the strength of the sensors' evidence, with an overall improvement of 24% in f-measure. Drilling down to the situation level, figures 6.14 (precision), 6.15 (recall) and 6.16 (f-measure) compare situation recognition with and without quality. Looking firstly at precision, rates improve for all situations except the 'busy at desk' situation'. This is because the user is sometimes reading at their

Situations	busy at desk	busy read-ing	informal break	coffee break	lunch break	meeting	undeter'd
busy at desk	**0.98**	0.0	0.01	0.0	0.0	0.0	0.0
busy reading	0.49	**0.33**	0.18	0.0	0.0	0.0	0.0
informal break	0.25	0.07	**0.67**	0.0	0.01	0.0	0.01
coffee break	0.18	0.04	0.18	**0.54**	0.0	0.04	0.04
lunch break	0.19	0.0	0.10	0.0	**0.70**	0.0	0.01
meeting	0.05	0.0	0.03	0.0	0.0	**0.92**	0.0

Table 6.11: Confusion matrix for CASL data set when quality parameters used

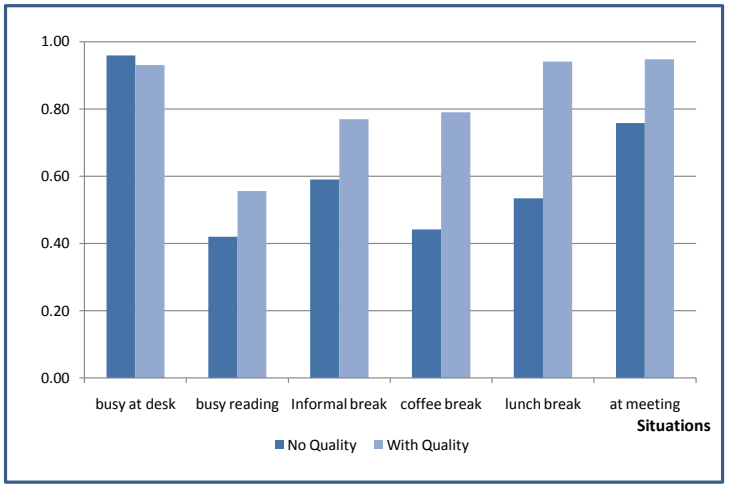


Figure 6.14: Comparison of precision by situation: with and without quality for the CASL data set

desk, but briefly uses the desk top, triggering the activity sensor. These timeslices are then classified as 'busy at desk' because of the higher 'quality' of the activity sensor. Recall improves for all situations, with the exception of the 'busy reading' situation. Again, because the user sometimes uses their desktop, some of the 'busy reading' situations are not retrieved, classified as busy at desk. Looking at f-measure,recognition accuracy improves for all situations when using quality, except for the 'busy at desk' situation which slightly disimproves. Again, this is because the 'busy reading' sitaution is sometimes interrupted by brief uses of the desktop. Because the activity monitor is 'trusted' more than ubisense, these are classified as the 'busy at desk' situations.

'Busy at desk' recognition accuracy improves substantially because Ubisense readings that occur 'near' to the desk area, but which are not actually in the desk area are recognised by the imprecise Ubisense mass function when quality is used. In addition, gaps in computer use which are annotated as 'busy at

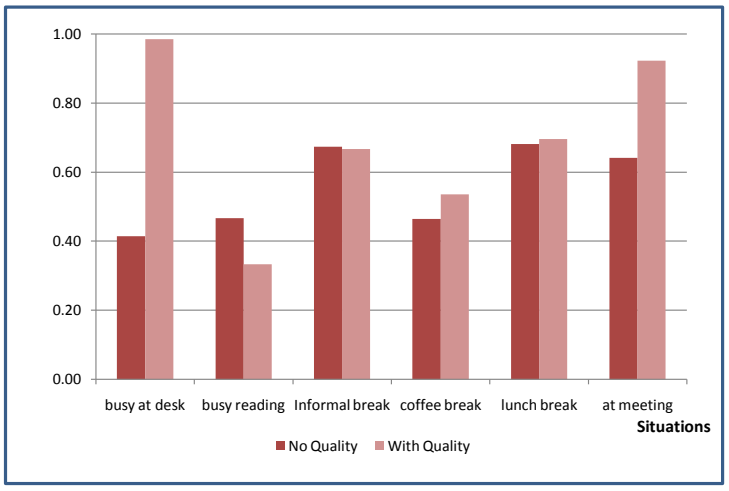


Figure 6.15: Comparison of recall by situation: with and without quality for the CASL data set

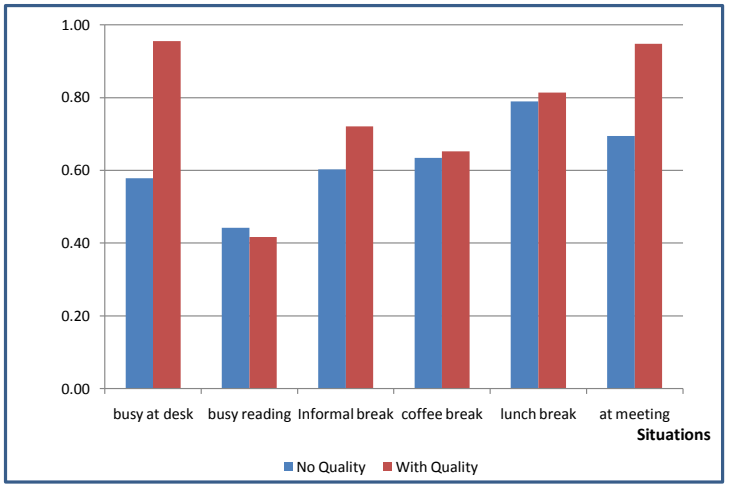


Figure 6.16: Comparison of f-measure by situation: with and without quality for the CASL data set

desk' are still recognised by the system as 'busy at desk' because of the gradual decay of the activity sensor.

To test statistical significance of using quality, we construct McNemar's contingency table for the two recognition modes (1) evidence decision network without quality (2) evidence decision network with quality parameters. McNemar's table is shown in Appendix A, Section 2. Using equation 6.4, the improvement in situation recognition accuracy using quality versus no quality parameters with this data set is statistically significant to the 99.99% level.

6.4.3 Experiment 3: Impact on situation recognition accuracy of individual sensor quality

Next, we wanted to see the impact of the quality parameters for each sensor individually. When quality is used, average f-measure increases from 0.6 to 0.74 by including quality, as shown in figure 6.13, so this analysis will indicate how this improvement is facilitated by each sensor. We used quality for one sensor at a time (with no quality parameters used for the remaining two). We also used excluded quality for one sensor at a time. The average f-measure for these analyses are shown in figure 6.17. Looking from the left hand side of the chart, the worst results are for "no quality" used. When quality is used for each sensor individually, the results improve. The calendar sensor shows the smallest improvement in overall situation recognition accuracy. This is because its contribution is to improve the detection of the 'at meeting' situation but with limited impact on the remaining five situations. Likewise, the calendar sensor is the least damaging to the overal result when it is excluded, as shown under 'activity and ubisense'. When quality is used with two sensors at a time, the results improve. The best result is when quality is used for all three sensors as shown on the right hand side, achieving an average f-measure of 0.74.

6.4.4 Experiment 4: Sensitivity analysis of quality parameters

In this experiment, we analyse how situation recognition accuracies vary as the sensor quality parameter values change. We do a separate analysis for static and dynamic quality parameters.

Static parameter sensitivity analysis: For each sensor in turn, we varied the

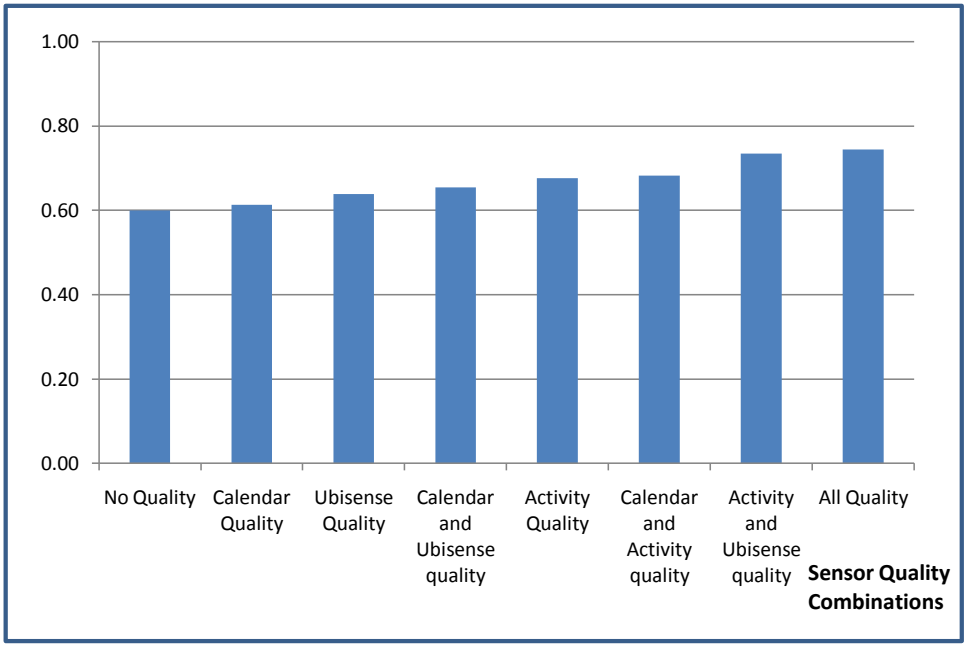


Figure 6.17: Comparison of f-measure for various sensor quality permutations.

Static quality:	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Ubisense	0.45	0.64	0.64	0.74	0.74	0.74	**0.75**	0.75	0.71	0.70
Calendar	0.70	0.73	0.73	0.73	0.73	**0.75**	0.74	0.74	0.74	0.73
Activity	0.56	0.58	0.65	0.66	0.66	0.68	0.70	0.75	0.75	**0.75**

Table 6.12: Average f-measure against variable static quality parameter values for each sensor. Actual static quality values from the DAG are bolded (Ubisense 0.7, Calendar 0.6, Activity 1)

static quality parameter value from 0.1 to 1 (full trusted) at intervals of 0.1, whilst keeping the remaining two sensors at their assigned quality values. The average f-measure mapped against the variable static quality for each of the sensors is shown in table 6.12. Both the Ubisense and activity sensors have a clear impact on f-measure, with both achieving their maximum f-measure to coincide with the assigned values (of 0.7 and 1 respectively). For both sensors, accuracy is not sensitive at nearby values. The activity sensor reaches its maximum at 0.8, but then dips from 0.7 downwards. At 0.7, the activity sensor is considered equally trustworthy as Ubisense, and from there downwards, less trustworthy - indicating that the *relative* quality value of the sensors may be important. Ubisense degrades the average f-measure when its static quality value dips below 0.4 and above 0.8. The calendar sensor does not impact average f-measure as much as the other two sensors, tying in with the calendar sensor's relatively low impact shown in figure 6.17

Dynamic parameter sensitivity analysis: Next, we examined how recognition accuracy sensitivity to dynamic parameter value changes. Table 6.13 shows how the dynamic values impact on average f-measure. Each of the three dynamic parameters (activity sensor time decay, calendar start/end time delays, ubisense precision) are varied as shown in the tables, and the impact on recognition accuracy noted. The activity sensor achieves the best result at four minutes time decay, as used in our analysis. Ubisense trained precision at 3.33m/2.2m is just below the value obtained at using 3m/3m for x and y axes, with a difference of 0.003 in average f-measure. The calendar sensor achieves the best f-measure at 5 minutes (as opposed to 10 minutes set up for the situation DAG) indicating that our estimate of meeting start and end times of 10 minutes delay is not adhered to strictly by the user.

Static quality parameters apply to every timeslice in the data set. Dynamic time-based quality parameters, as used for the calendar and activity sensors, apply only to specific pockets of the data when a situation transitions to another situation. For example, the activity sensor tails off to inactivity using a dynamic time decay, so only timeslices affected by the time function when the activity sensor goes from active to inactive are affected. The nature of the time decay will depend on how quickly the sensed phenomenon is to be ignored, and thus how gradually (or not) situations using that sensor should transition to and from other situations. Figure 6.18 shown a sample situation transitions from 'busy at desk' to a 'coffee break', for differing values of activity sensor time decay. In the case of the activity sensor, four minutes was used as a reasonable estimate of how long non-use of the computer takes to reach 'inactive' status. If the time decay is shortened, the 'busy at desk' situation belief will decrease more quickly and thus the situation will more rapidly transition to the next situation (informal break). Longer time decays will lengthen the situation transition. The annotations by the user are captured as a step change, from 'busy at desk' to 'busy reading' in a single moment, with no concept of partially in one situation and partially in another.

The selection of the time decay value, the nature (linear versus trapezoidal) and whether to use such a value will ultimately be driven by the requirements of the application that is relying on situation recognition.

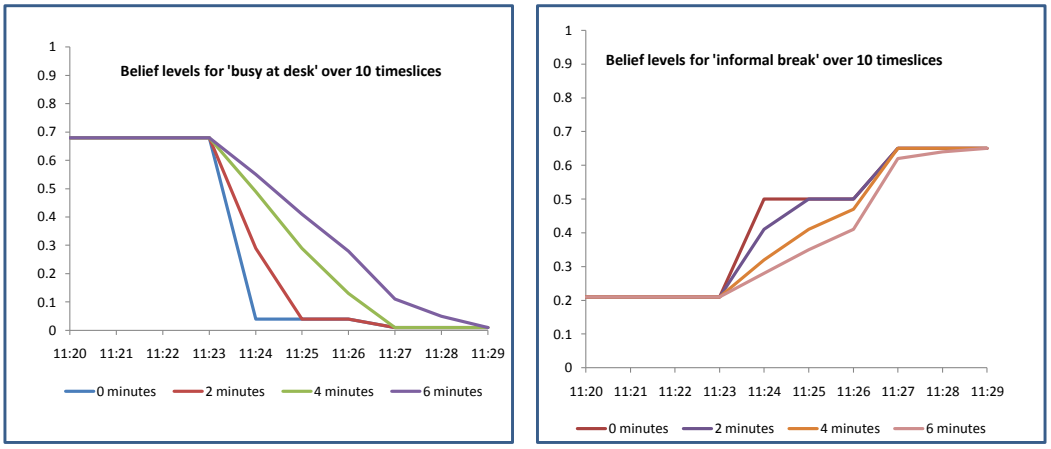


Figure 6.18: Situation transitions for two situations against variable activity time decay of 0,2,4,6,8 minutes for 10 timeslices from time 11:20 to 11:29. Belief level taper more gradually for longer time decays.

Activity sensor Time decay (minutes)	0	2	**4**	6	8	10
Average f-measure	0.654	0.731	**0.744**	0.740	0.732	0.724

Ubisense sensor precision x/y (m)	0	1/1	2/2	**3.33/2.2**	3/3	4/4	5/5
Average f-measure	0.717	0.725	0.743	**0.744**	0.747	0.659	0.732

Calendar sensor time delay at start/end of meetings (mins)	0	5	**10**	15	20
Average f-measure	0.740	0.751	**0.744**	0.735	0.732

Table 6.13: Average f-measure against variable dynamic quality parmeters values for activity, ubisense and calendar sensor. Actual quality parameters used are highlighted in bold.

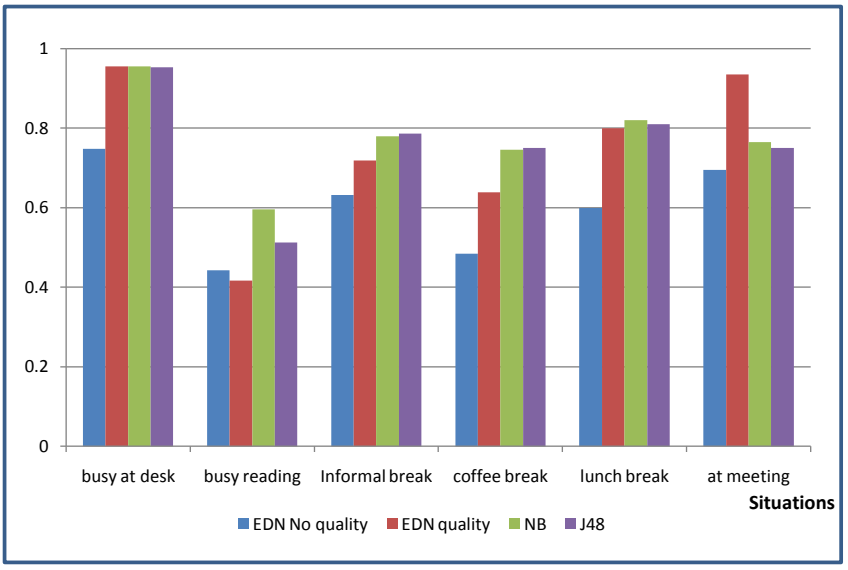


Figure 6.19: Comparison of f-measure of evidence decision network with and without quality against NBayes and J48 classifiers on the CASL data set

6.4.5 Experiment 4: Comparison with other inference techniques

We used both Naïve Bayes and J48 Decision Tree learning techniques on the data set in order to sense check the classification results of our evidence theory results. Naïve Bayes and J48 were applied using 10 fold cross validation. The results, comparing average precision, recall and f-measure for the evidence decision network with quality, Naïve Bayes and J48 are shown in figure 6.19. Average f-measure accuracy of situation recognition using the evidence decision network with quality is below but close to Naïve Bayes (4%) and J48 (2%). This is intuitively correct. When using quality, the extent of uncertainty in the system is being explicitly quantified. When using learning techniques, probabilistic distributions of the crisp values are doing the same thing, but with the benefit of actual values in the test set. Our quality parameters are obtained from a different exercise from the test data, so their 'success' is the extent to which they match the actual behaviour of sensors within the test data.

6.4.6 Discussion of quality results

The results when using quality with the evidence decision network were better than when quality was excluded, significant to the 99.99% level. The improvement in situation recognition accuracy is expected because the quality

parameters inform the system to what extent the sources can be believed. In this data set, we know through observation and training that the sensors are not fully trustworthy. In effect, we have applied a weighting scheme on the sensors and propagated this up to situation belief levels. The benefit of the approach is that known quality issues can be encoded into the system. The disadvantage is that if the quality parameters are "wrong" (i.e. they do not reflect real quality performance), then this will impact the inference results. This is reflected in the sensitivity analysis where situation accuracy starts to decline as the sensor quality values move further away from their assigned values.

The 'correctness' of quality parameters is an issue where sensors are prone to drift or degradation of quality over time, in an unpredictable or unknown way. This issue is akin to temporal information drift (such as human behaviour patterns changing) and both are part of the wider problem of environmental changes. i.e. how can knowledge in a specification or, indeed, a learning model be changed as the environment changes? Clearly, some type of feedback loop is required, where updates to knowledge in the model can be provided. This is discussed further in section 6.6.

The time-based quality parameters for the calendar and activity sensors impact at situation transition time, because they change the nature of the belief assignment at the beginning (calendar) and end (calendar and activity) of context values from these sensors. Since this controls the speed of situation transition switchover as shown in figure 6.18, it will be of interest to applications when the speed of changes in situations are important, such as critical health state monitoring, as discussed by Haghighi *et al.* [21].

6.5 Situation Recognition using alternate fusion rules

In Chapter 3, we discussed the theoretical benefits of using Murphy's alternative fusion rule over the standard Dempster's rule of combination. In this section, we will examine the impact on situation recognition accuracy of using four different fusion rules. In addition to Murphy's combination rule, we will use the standard Dempster rule of combination and the averaging rule described in Chapter 3. To recap, Dempster's rule allows single disagreeing sensors to overrule a majority if it disagrees. Therefore, for situations with evidence spread over time, fusion will be distorted if a single sensor is 'off' for

Fusion Rule:	**OR**	**Averaging**	**Dempster**	**Murphy's**
evidence decision network without temporal	**0.57**	0.54	0.48	0.55
evidence decision network with temporal	0.66	0.65	0.60	**0.68**

Table 6.14: Comparison of recognition accuracies (f-measure) for four fusion rules on van Kastersen's data set. Best results are highlighted in red.

	Averaging	Dempster's	Murphy's
evidence decision network without quality	0.5	**0.62**	0.60
evidence decision network with quality	0.65	**0.75**	0.74

Table 6.15: Comparison of recognition accuracies for three fusion rules on the CASL data set. Best results are highlighted in red.

that timeslice. We will also compare a fourth fusion approach, using an 'OR' combination rule. This may be useful for task-driven situations where the evidence is spread over time because in theory, only a single piece of evidence needs to be triggered in order to detect the situation. In the 'OR' fusion scenario, at each timeslice, the situation belief will be calculated as the maximum or summed belief of the context events associated with the situation, based on our 'OR' fusion equations 3.9 and 3.10.

Table 6.14 shows a comparison of each of the four rules for van Kasteren's data set. Murphy's combination achieves the best results for both basic evidence decision network without time extension and with time extended evidence. We anticipated this because Dempster's rule will allow a single sensor to overrule other sensors. Therefore, unless all sensors are fired at the same time in a time slice, the evidence from any firing sensors will be lost. The averaging rule produces results that are almost as good as Murphy's rule. The averaging rule uses less computation but it does not provide any measure of conflict. At present, we do not use the conflict metric in our inference process so averaging may provide a faster alternative to evidence fusion. Interestingly, the 'OR' fusion rule performs best when used without time extension of evidence. Only the 'OR' rule disregards co-occurrence of evidence, because it only relies on single pieces of evidence. The 'OR' approach also has issues distinguishing situations if they share sensors. For example, if a cup is used, and this is equally evidential of breakfast and drink (with no absolute time to distinguish), the system cannot distinguish which situation is occurring. When evidence is time-extended, all approaches, including the 'OR' fusion approach, benefit from longer durations of evidence.

We ran a similar analysis for the CASL data set, contrasting results for fusion using averaging, Dempster's rule and Murphy's rule. We excluded the 'OR' combination rule because all evidence is co-occurring in the CASL situations so there is no theoretical benefit in using 'OR' fusion. Situation recognition was conducted both with and without quality on the sensors in case this affected the results in some way. Looking at table 6.15, in the CASL data set, Demptser's rule of combination performs slightly better than Murphy's rule, in both without quality (3%) and with quality (1%) modes. Unlike van Kastersen's data set, all evidence is continuous so the problem of single sensors being 'off' and overruling other firing sensors is not an issue. The averaging rule performs relatively poorly. In the CASL data set, evidence is frequently applied to combinations of situations (as opposed to the van Kasteren data set where evidence was applied to single situations). For example, if the computer activity sensor is 'inactive', this is indicative of any of the situations 'busy at desk', 'coffee break', 'lunch break', 'informal break' or 'at meeting'. When averaging is used, the evidence is simply divided up amongst the elements prior to averaging. In Dempster's rule and Murphy's rule, agreeing evidence is merged so combined evidence converges more distinctly towards situations that have further evidence. As a result, the averaging rule results are more successful in van Kastersen's data set than in the CASL data set.

6.6 Discussion

We evaluated situation recognition accuracy using two data sets. We tested our two hypotheses: (1) that the use of temporal knowledge in the evidence decision network improves recognition accuracy over its non-use in the evidence decision network (using van Kasteren data set) and (2) that the integration of sensor quality with the evidence decision network improves recognition accuracy where sensor quality is an issue (using the CASL data set). Our results were significant indicating that both temporal and quality enhancements to the evidence decision network increased recognition accuracy. We also compared the evidence decision network approaches with two classic machine learning techniques, Naïve Bayes and J48 decision tree to check that their relative performance - with the extended evidence decision network closely matching (quality) or exceeding (temporal) the learning techniques accuracies. For the van Kasteren data set, we compared with existing published results, noting that the evidence decision network out performed the HMM approach of van

Kasteren but did not perform as well as Ye's situation lattice approach. Based on our evaluation and experience of applying the evidence decision network, we discuss the benefits and limitations of our approach for situation recognition.

6.6.1 Benefits of the evidence decision network approach

The evidence decision network approach enables the encoding of temporal knowledge, quality knowledge and situation hierarchy knowledge into the inference process, as demonstrated using the van Kasteren and CASL data sets. The most obvious benefit of the evidence decision network approach is that it is not reliant on *training data* for the environment, but still can yield good situation recognition results. The extended evidence decision network performed well in comparison to the two machine learning techniques. We do not claim that the evidence decision network approach is better than learning techniques when training data is available - the learning technique were not pruned or optimised in any way. However, our results show that our evidence-based approach can produce good results for scenarios where training data is difficult or costly to obtain.

In the evidence decision network approach, the knowledge used to recognise situations is traceable and visible. For example, sensor quality parameters, inference rules and situation durations are known and captured in the situation DAG. Therefore, any *change* in the environment, such as a new situation or removal of a sensor can be applied in a controlled, deliberate way because there is a transparent model of the environment.

Related to the previous point, *intelligibility* of system decisions in context-aware systems is important [8, 66, 31]. Lack of system intelligibility can lead to loss of user trust, satisfaction and acceptance of these systems [54]. One mechanism to alleviate lack of intelligibility in intelligent context-aware systems is through automatically generated explanations that are human understandable versions of the reasoning process [66]. An example of an explanation, as tested on users in Lim's work [66], is "`Activity predicted as not exercising because body temperature` $\leq$ `5 and pace` $\leq$ `3.0`. We suggest that the evidence decision network approach will support intelligibility by enabling the creation of explanations for application users. In the evidence decision network approach, situation recognition is transparent because the treatment of

knowledge is decided in advance and captured in the situation DAG. At any point in time, situation beliefs can be traced back to the context values that were used to determine them. Therefore, a system using evidence decision networks should be able to provide an explanation of its decision process. In learning processes, the decision process is less transparent, more akin to 'black box' reasoning so the system's decision process is hidden from the user. This point on explanations is discussed further under future work in Chapter 7.

The evidence decision network approach supports situation hierarchies and situation combinations, because the hierarchy of situations is known in the situation DAG and directly used in the evidence decision network. Thus our approach supports hand crafting solutions to invalid or uncertain scenarios, safeguarding against system instability. For example, where certain combinations of situations "cannot" occur together (such as 'running' and 'asleep'), these can be encoded as invalid combinations into the decision algorithm - as described in section 6.1 - and thus prevented from being returned together by the recognition process. This will help to prevent unstable scenarios occurring at the application level. Similarly, situation recognition is not limited to 'single layer' recognition of situations as is typically the case with learning techniques. For example, in the case of the CASL dataset, the system may determine that the user is at their desk, but it cannot discern whether they are reading or using their desktop. A summary situation can be specified of 'at desk', and the belief of the lower level situations assigned to this. The evidence approach supports the detection of summary situations, thus allowing richer interpretations of the environment's situations.

A final benefit of the evidence-based approach is the range of knowledge that can be used to enhance recognition accuracy. We have described two particular extensions for temporal and quality knowledge to add to the available 'tools' of the systems developers. Only knowledge relevant to the environment will be used in the evidence decision network. For example, if absolute times and/or transitory evidence is relevant in the environment, these should be included. If not available, they should be ignored. In future work we expect to incorporate new types of knowledge which will widen the choice for systems developers to enhance the situation recognition approach with new 'clues'.

6.6.2 Limitations

Whilst the evidence decision network is a good approach for exploiting domain knowledge, this knowledge must be available. In environments where the relationship between sensors and situations is detectable and known in advance and training data availability is a problem, an evidence decision network approach is a potential scheme. However, if there is a complex relationship from sensors to situations that is *only* discernible via training data, the evidence decision network is not suitable. It is worth emphasising, however, the difference between localised training and universal training data for the environment. It may be possible to obtain localised training data to establish the relationship between sensor and situations, even if universal training data for the environment is not available. For example, a complex situation can be acted out, and sensor activations observed, without requiring full training data capture.

The evidence decision network approach assumes human crafting of knowledge for the situation DAG. For environments with hundreds of sensors and situations, this task becomes more difficult. It is not easy to quantify the limit on when manual inspection becomes impractical - perhaps it is the limit at which a user can expect transparency of system reasoning - but it is a definite consideration when selecting whether to use specification versus learning approaches.

Another possible limitation is the *computational complexity* of the fusion rules. For both Murphy's and Dempster Rule of Combination, evidence can be applied for any *combination* of hypotheses in a frame, so the number of permutations for n hypotheses in a frame can be as high as the power set, 2^n. But in reality, all combinations of hypotheses may never be used and there is no need to always build all the possible values of belief. There are many cases where the knowledge is very simple and where there are very few non-null masses - making the belief function computation lighter than its competitors [97]. In addition, we noted that the simple averaging rule achieved comparable results with Murphy's decision rule on van Kastersen's data set. The averaging rule works well if applied to evidence that support single hypotheses only in a frame. Another way to reduce computational effort is to limit calculations to active parts of the environment. For example, using an example from Hong *et al.*'s [50] smart home work, a motion sensor may detect motion in the kitchen. At that point in time, situation beliefs will only be calculated for kitchen-based

situations. Zhang *et al*. [125] provide another solution for computational complexity by excluding low contribution beliefs from the evidential process in order to reduce computational load.

A final limitation that we see with the evidence decision network approach is the *expert knowledge about evidence theory* required by system designers to specify evidential operations. This expert knowledge may be difficult to acquire or share in a system design environment. One solution is to package the evidence decision network so that the designer/developer is shielded from the intricacies of selecting evidential operations. The situation DAG could be defined (in a software tool) for a specific environment, and the recommended evidential operations for the evidence decision network automatically gleaned from the DAG. For example, if the DAG contains a situation with a duration (indicating transitory evidence) and the underlying frames indicate binary sensors, then Murphys fusion rule is automatically selected. This is largely a software engineering challenge but is an important consideration for the practical usage of an evidence decision network in real world systems.

6.6.3 Summary

To summarise, if universal training data for an environment is not available, our evidence-based approach offers a good domain knowledge alternative to the traditional machine learning techniques. It caters for sensor and rule uncertainty, enables the encoding of temporal and quality knowledge, and provides a transparent method for reasoning about situations. Machine learning techniques, on the other hand, learn from training data with less flexibility for tweaking with additional knowledge about the environment such as sensor quality and temporal knowledge.

CHAPTER

SEVEN

Conclusions and Future Work

7.1 Conclusion

In this thesis, we have motivated the need for a reasoning technique that can reason with uncertainty, but that allows the inclusion of domain knowledge as a way of reducing reliance on training data. We applied Dempster-Shafer theory to develop an evidence decision network approach that captures and processes knowledge as evidence in order to recognise situations.

Chapter 2 analysed the key features of situations, and the key learning and specification-based techniques applied to situation recognition. We noted that no single reasoning technique is suitable in all environments, with considerations such as availability of training data, requirement for transparency of decision making and level of uncertainty of sensor and environments to be considered in the selection of a technique. We identified the need for an approach that caters for the inference uncertainty in the reasoning process, but which is not reliant on training data to establish a learning model. Dempster-Shafer theory addresses both of these issue. Existing approaches using Dempster-Shafer have not provided an end to end solution for situation recognition, and do not cater for temporal and quality knowledge.

Chapter 3 presented the theoretical foundation for our Dempster-Shafer based approach. We researched the appropriate evidential operations from basic Dempster-Shafer theory and extensions to the theory by other practicioners. We also defined the additional evidence operations for processing evidence where no operations existed in the literature. Situation DAGs, our diagramming technique, are explained - they are used to capture knowledge that drives the structure of the evidence decision network for an environment.

Chapter 4 described extensions to the evidence decision network approach

to cater for two specific types of knowledge: temporal knowlege (of absolute times and transitory evidence) and sensor quality knowledge. We developed mathematical formalisms to integrate this knowledge as evidence. The resultant evidential decision network supports the recognition of situations using variable quality sensor data, uncertain inference rules and evidence with identifiable time patterns.

The feasibility of the evidence decision network approach was demonstrated in Chapter 5 by applying it to two real-world datsets. The situation DAGs were defined for each of the data sets, and used as key inputs for the evaluation of our approach in Chapter 6. Using the smart home data set from van Kasteren, we discovered that the inclusion of temporal knowledge in the evidence decision network significantly improves situation recognition accuracy. When comparing the evidential decision network to two well-known classifiers, Naïve Bayes and J48 decision tree, we revealed that temporal knowledge boosted the evidential decision network above the two classifiers. The network has limited benefit from increased training data, in contrast to the two learning classifiers. With the office-based CASL data set, situation recognition accuracy signficantly improved with the incorporation of sensor quality. Dynamic quality parameters associated with time decay impact at situation transitions, with a need for system developers to consider the requirements of the responding application when using such parameters. When used against learning classifiers, the evidence network with quality produced situation recognition accuries just below the classifiers. Sensor quality parameters attempt to mimic the impact of sensor noise, so their performance against learning techniques relies on the accuracy of the quality parameters.

The key benefit of the evidence decision network approach is its ability to incorporate domain knowledge, thus reducing or eliminating the requirement for training data. However, the converse of this benefit is that domain knowledge is required. Richer knowledge about the environment (such as temporal, quality and situation hierarchicies) will boost the evidence decision network's ability to decipher the correct situation(s). Less knowledge provides a weaker basis for recognition. Learning techniques are suited where training data is available. Unlike the evidence decision network, systems developers and users are shielded from the complexities of sensor/situations relationships: but this leads to a lack of scrutability. With the evidence approach, decisions are transparent: with full decision making paths shown on the situation DAG.

7.1.1 Contributions

We summarise the contributions of this thesis as follows:

1. We have presented and evaluated a situation recognition approach based on Dempster-Shafer theory. This approach enables situation recognition using sensor data in uncertain environments based on domain knowledge.

2. We have selected a set of evidence operations to enable the Dempster-Shafer approach to be applied in various environments via an evidence decision network. This involved the selection of existing evidence operations from the literature and the creation of *new* evidential operations that propagate and process sensor evidence. The new operations support belief distribution and decision making in the network and are summarised in table 3.3. We also define algorithms to process the belief distribution, time extensions and decision steps, as described in Section 6.1.

3. We have created two extensions to Dempster-Shafer theory to support the incorporation of two types of environmental knowledge: (1) temporal knowledge of transitory evidence and absolute times (2) sensor quality knowledge, with both static and dynamic quality parameters. The extensions include the development of mathematical formalisms required to apply temporal and quality knowledge into evidence processing.

4. We have developed a diagramming technique which we term situation Directed Acyclic Graphs to capture knowledge. This is a key element for creating an evidence decision network approach because the evidence operations are determined from the knowledge denoted in the situation DAG.

5. We use two real-world sensor data sets to test the feasibility of our approach. We create a situation DAG for each of the data sets using a predefined set of steps.

6. We evaluate the situation recognition accuracy of the evidence decision networks for the two data sets. We discovered a significant improvement when temporal and quality knowledge was applied. We compare the evidence approach to two learning classifiers.

7. We evaluate the advantages of a choice of fusion rules: Murphys', Dempster's rule of combination, averaging rule and 'OR' rule. We identify the

scenarios in which each should be used, using accuracy results from the two data sets.

7.1.2 Limitations of our work

As discussed in Section 6.6.2, there are a number of limitations with our approach in the thesis. Domain knowledge is required in order to establish the structure of the evidence decision network, and the temporal/quality parameters to be used. This becomes more complex as the number of situations and sensors increases. We do not propose an empirical limit - perhaps it is the limit at which a user can expect to understand the reasoning process when presented with the set of situations and sensors in the system - but this is a consideration when using our approach.

Computational complexity is a factor in evidence processing due to the potentially large number of hypotheses in each frame. However, as explained in 6.6.2, various approaches to reduce computational load can be taken including the exclusion of small belief contribution, processing of evidence only in the part of the environment where activity is detected, the use of the averaging rule if appropriate, and the occurrence of null masses for many of the hypotheses at any point in time.

In our evaluation, we applied our Dempster-Shafer approach to two smart environment data sets - the van Kasteren smart home data set, and our own in-house CASL data set. Both of these data sets are single occupancy, relatively small (28 and 5 days duration respectively) with limited numbers of sensors. Whilst the situation recognition results using our approach are promising, we suggest that further application of our approach to larger data sets will be needed to confirm the robustness and generalisability of our approach.

7.2 Future work

One of the challenges in pervasive computing is the requirement to re-create a model of each new environment in which an application will reside. An activity monitoring application may need to cater for different sensors, different user behaviours and so forth when applied across different homes. With machine learning approaches, training data must be collected for any change in environment. With the evidence approach, the transparency of knowledge in

the DAG suggests that it may be possible to transfer knowledge from one environment to another. Adjustments to the situation DAGs for known changes in sensors, or activity definitions can be applied and the knowledge re-used. Van Kasteren *et al*. [108, 104] are currently looking at this problem of 'transfer learning' to examine how to use annotated training data to label training data from another similar environment. They use two smart home data sets to demonstrate mapping techniques for transfer learning. They plan to release their data sets, which we may use to determine how transfer learning may be applied in our evidence decision network approach.

As described in Chapter 6, the transparency of knowledge in the evidence approach should in theory support the use of explanations in reasoning. Researchers such as Lim et al. [66] are actively testing the usefulness of explanations on applications users. We need to examine *how* and *what* to generate for explanations using our evidence approach.

One of the difficulties in the pervasive domain is the lack of real-world published data sets, a fact that we have recently raised in our research [122]. When real world environments are used, complexities appear, such as task interruption, multi-tasking, multiple users, unexpected user behaviour, as described by Logan *et al.* [68] where they look at activity monitoring in a smart home. As part of this problem, we will need to examine whether new measures for evaluating reasoning techniques should be used. At present, the research community focusses on classification accuracies using traditional machine learning measures - i.e. obtaining the 'right' answer for as many instances (timeslices), akin to classifying static knowledge such as documents. But in pervasive environments, situations are dynamic, of varied duration, sequential, interleaved; and application behaviours and transitions need to be smooth and controlled. For example, rather than checking the proportion of a situation correctly recognised, it may be more useful to check whether an activity was detected at all over a period of time (e.g in a monitored smart home, did the user prepare breakfast today at all?). Boundaries between situations may be important for health applications [102], such as whether a person's heart rate has moved from normal to high within a certain period of time. For our next phase of research, we will examine what real-world complexities we can address with our evidence approach, and what new measures we should consider for evaluations in the future.

In Chapter 6, we described how our evidence approach could be packaged to shield systems developers from the complexities of selecting individual fusion

rules, frames of discernment and other evidential operations. Whilst this is a software engineering challenge rather than a research challenge, we would see it as an important part of enabling the evidence decision network approach to be used in the future in real world systems development.

Appendix

Section 1 McNemar's table for temporal knowledge test using van Kasteren data set

The classifications are based on using one third training, two thirds testing with cross validation. Each time slice is used for training once, but is tested twice. The total entries in McNemar's table is the total number of tested timeslices. McNemar's contingency table for Dempster-Shafer Inference vs Dempster-Shafer Inference plus temporal is:

42509	183
489	4177

Using McNemar's equation 6.4, χ^2 is calculated as 138.43

Section 2 McNemar's table for quality test on the CASL data set

To test statistical significance of using quality, we construct McNemar's contingency table for the two recognition modes (1) Dempster-Shafer inference without quality (2) Dempster-Shafer inference with quality, as shown in table X. Using equation X, we calculate that difference in results using no quality versus with quality and Dempster-Shafer inference with this data set is statistically significant to the 99.99% level. McNemar's contingency table for evidence decision network with and without quality is:

666	129
8	649

Using McNemar's equation 6.4, χ^2 is calculated as 623.44

Section 3 Sample data from the van Kasteren data set

The following is sample activities instances from the van Kasteren dataset.

activity id list:
1: 'leave house'
4: 'use toilet'
5: 'take shower'
10:'go to bed'
13:'prepare Breakfast'
15:'prepare Dinner'
17:'get drink'

Start time	End time	ID
25-Feb-2008 00:22:46	25-Feb-2008 09:34:12	10
25-Feb-2008 09:37:17	25-Feb-2008 09:38:02	4
25-Feb-2008 09:49:23	25-Feb-2008 09:53:28	13
25-Feb-2008 10:02:28	25-Feb-2008 10:12:42	5
25-Feb-2008 10:19:06	25-Feb-2008 16:55:38	1
25-Feb-2008 17:00:31	25-Feb-2008 17:01:34	4
25-Feb-2008 17:54:55	25-Feb-2008 17:55:58	17

The following is sample sensor instances from the van Kasteren dataset.

Start time: Time sensor started firing **End time:** Time sensor stopped firing **ID**: sensor id (see list below) **val**: value of sensor (always 1)

Sensor Ids: [1]'Microwave' [5]'Hall-Toilet door' [6] 'Hall-Bathroom door' [7]'Cups cupboard' [8] 'Fridge' [9] 'Plates cupboard' [12]'Frontdoor' [13]'Dishwasher' [14 'ToiletFlush' [17] 'Freezer' [18] 'Pans Cupboard' [20] 'Washingmachine' [23]'Groceries Cupboard' [24]'Hall-Bedroom door'

Start time	End time	ID	Val
25-Feb-2008 00:20:14	25-Feb-2008 00:22:57	24	1
25-Feb-2008 09:33:41	25-Feb-2008 09:33:42	24	1
25-Feb-2008 09:33:47	25-Feb-2008 17:21:12	24	1
25-Feb-2008 09:36:43	25-Feb-2008 09:37:04	5	1
25-Feb-2008 09:37:20	25-Feb-2008 09:37:23	6	1
25-Feb-2008 09:37:51	25-Feb-2008 09:37:52	14	1
25-Feb-2008 09:37:55	25-Feb-2008 09:37:56	14	1
25-Feb-2008 09:37:58	25-Feb-2008 09:38:01	6	1
25-Feb-2008 09:49:27	25-Feb-2008 09:49:28	9	1
25-Feb-2008 09:49:31	25-Feb-2008 09:49:38	9	1
25-Feb-2008 09:49:39	25-Feb-2008 09:49:44	8	1

BIBLIOGRAPHY

[1] Fahd Albinali, Nigel Davies, and Adrian Friday. Structural learning of activities from sparse datasets. *Fifth Annual IEEE International Conference on Pervasive Computing and Communications (PerCom'07)*, 0:221–228, 2007.

[2] James F. Allen and George Ferguson. Actions and events in interval temporal logic. *Journal of Logic and Computation*, 4:531–579, 1994.

[3] C. Anagnostopoulos, Y. Ntarladimas, and S. Hadjiefthymiades. Situation awareness: Dealing with vague context. *International Conference on Pervasive Services*, 0:131–140, 2006.

[4] C.B. Anagnostopoulos, Y. Ntarladimas, and S. Hadjiefthymiades. Situational computing: An innovative architecture with imprecise reasoning. *Journal of Systems and Software*, 80(12):1993 – 2014, 2007. Selected papers from the International Conference on Pervasive Services (ICPS 2006).

[5] Juan C. Augusto, Jun Liu, Paul McCullagh, Hui Wang, and Jian-Bo Yang. Management of uncertainty and spatio-temporal aspects for monitoring and diagnosis in a smart home. *International Journal of Computational Intelligence Systems*, 1(4):361–378, 2008.

[6] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784, 2000.

[7] Ling Bao and Stephen S. Intille. Activity recognition from user-annotated acceleration data. In *Pervasive*, pages 1–17, 2004.

[8] V. Bellotti and K. Edwards. Intelligibility and accountability: Human considerations in context-aware systems. *Human-Computer Interaction*, 16(2-4):193–212, 2001.

[9] 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.

[10] Thomas Buchholz, Axel Kupper, and Michael Schiffers. Quality of context: What it is and why we need it. In *Tenth Workshop of the HP OpenView University Association (OVUA '03),Geneva, Switzerland*, July 2003.

[11] Paul Castro, Patrick Chiu, Ted Kremenek, and Richard R. Muntz. A probabilistic room location service for wireless networked environments. In *UbiComp '01: Proceedings of the 3rd international conference on Ubiquitous Computing*, pages 18–34, London, UK, 2001. Springer-Verlag.

[12] Dan Chalmers, Naranker Dulay, and Morris Sloman. Towards reasoning about context in the presence of uncertainty. In *Workshop on Advanced Context Modelling, Reasoning And Management at UbiComp 2004*, 2004.

[13] Keith Cheverst, Nigel Davies, Keith Mitchell, and Adrian Friday. Experiences of developing and deploying a context-aware tourist guide: the guide project. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 20–31, New York, NY, USA, 2000. ACM.

[14] Driss Choujaa and Naranker Dulay. Activity inference through sequence alignment. In *LoCA '09: Proceedings of the 4th International Symposium on Location and Context Awareness*, pages 19–36, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Brian Clarkson, Alex Pentland, and Kenji Mase. Recognizing user context via wearable sensors. *Wearable Computers, IEEE International Symposium*, page 69, 2000.

[16] Adrian Clear and Simon Dobson. An approach to dealing with uncertainty in context aware pervasive computing systems. In *In Proceedings of the UK/IE IEEE SMC Cybernetic Systems Conference 2007*. IEEE Press 2007, 2007.

[17] Patricia Dockhorn Costa, Giancarlo Guizzardi, Joao Paulo A. Almeida, Luis Ferreira Pires, and Marten van Sinderen. Situations in conceptual modeling of context. In *EDOCW '06: Proceedings of the 10th IEEE International Enterprise Distributed Object Computing Conference Workshops*, page 6, Washington, DC, USA, 2006. IEEE Computer Society.

[18] Joëlle Coutaz, James L. Crowley, Simon Dobson, and David Garlan. Context is key. *Commun. ACM*, 48(3):49–53, 2005.

[19] 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. pages 292–296, Innsbruck, Austria, 16/09/2007 2007.

[20] Waltenegus Dargie. The role of probabilistic schemes in multisensor context-awareness. *IEEE International Conference on Pervasive Computing and Communications Workshops*, pages 27–32, 2007.

[21] Pari Delir Haghighi, Shonali Krishnaswamy, Arkady Zaslavsky, and Mohamed Medhat Gaber. Reasoning about context in uncertain pervasive computing environments. In *EuroSSC '08: Proceedings of the 3rd European Conference on Smart Sensing and Context*, pages 112–125, Berlin, Heidelberg, 2008. Springer-Verlag.

[22] Arthur Dempster. Upper and lower probabilities induced by a multivalued mapping. *The Annals of Statistics*, 38:325–339, 1967.

[23] Arthur P. Dempster. A generalization of bayesian inference. In *Classic Works of the Dempster-Shafer Theory of Belief Functions*, pages 73–104. 2008.

[24] Thierry Denoeux. Conjunctive and disjunctive combination of belief functions induced by non distinct bodies of evidence. *Artificial Intelligence*, 172(2-3):234–264, 2008.

[25] Anind Dey. *Providing Architectural Support for Building Context-Aware Applications*. PhD dissertation, Georgia Institute of Technology, 2000.

[26] Anind Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5(1):1–81, 2001.

[27] Anind Dey, Jennifer Mankoff, Gregory Abowd, and Scott Carter. Distributed mediation of ambiguous context in aware environments. In

UIST '02: Proceedings of the 15th annual ACM symposium on User interface software and technology, pages 121–130, New York, NY, USA, 2002. ACM.

[28] Anind K. Dey and Gregory D. Abowd. The context toolkit: aiding the development of context-enabled applications. In *CHI '99: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 434–441, New York, NY, USA, 1999. ACM.

[29] Thomas G. Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *neural computation*, 10, 1998.

[30] Simon Dobson, Lorcan Coyle, and Paddy Nixon. Hybridising events and knowledge as a basis for building autonomic systems. *Journal of Trusted and Autonomic Computing*, 2006.

[31] Simon Dobson and Paddy Nixon. More principled design of pervasive computing systems. In *Human computer interaction and interactive systems*, volume 3425 of *Lecture Notes in Computer Science*. Springer Verlag, 2004.

[32] Simon Dobson and Paddy Nixon. Whole-systems programming of adaptive ambient intelligence. *Universal Access in HCI, AmbientInteraction*, 4555/2007:73–81, 2007.

[33] Didler Dubois and Henri Prade. Representation and combination of uncertainty with belief functions and possibility measures. *Computational Intelligence.*, 4(3):244–264, 1988.

[34] Maria R. Ebling, Guerney D. H. Hunt, and Hui Lei. Issues for context services for pervasive computing. In *Advanced Workshop on Middleware for Mobile Computing Middleware '01*, November 2001.

[35] Deiter Fox, Jeffrey Hightower, Henry Kauz, Lin Liao, and Donald J. Patterson. Bayesian techniques for location estimation. In *In proceedings of the 2003 Workshop on Location Aware computing*, pages 16–18, 2003.

[36] David Garlan, Daniel P. Siewiorek, and Peter Steenkiste. Project aura: Toward distraction free pervasive computing. *IEEE Pervasive Computing*, 1:22–31, 2002.

[37] J. Gordon and E. H. Shortliffe. The dempster-shafer theory of evidence. pages 529–539, 1990.

[38] Björn Gottfried, Hans W. Guesgen, and Sebastian Hübner. Spatio-temporal reasoning for smart homes. *Designing Smart Homes*, 4008:16–34, 2006.

[39] Phillip Gray and Daniel Salber. Modelling and using sensed context information in the design of interactive applications. In *Engineering for Human Computer Interaction*, volume 2254/2001 of *Lecture Notes in Computer Science*, pages 317–335. Springer Berlin/Heidelberg, 2001.

[40] Tao Gu, Hung Keng Pung, and Da Qing Zhang. A bayesian approach for dealing with uncertain contexts. In *Austrian computer society*, pages 205–210, 2004.

[41] 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 Conference*, pages 270–275, 2004.

[42] R. Haenni. Shedding new light on zadeh's criticism of dempster's rule of combination. *Information Fusion, 2005 8th International Conference on*, 2:6–10, July 2005.

[43] P. Haghighi, M. Gaber, S. Krishnaswamy, A Zaslavsky, and Loke S. An architecture for context-aware adaptive data stream mining. In *nternational Workshop on Knowledge Discovery from Ubiquitous Data Streams (IWKDUDS07)*, pages 423–434, Warsaw, Poland, September 2007.

[44] David Heckerman. A tutorial on learning with bayesian networks. Technical report, 1996.

[45] Sumi Helal, William Mann, Hicham El-Zabadani, Jeffrey King, Youssef Kaddoura, and Erwin Jansen. The gator tech smart house: A programmable pervasive space. *Computer*, 38:50–60, 2005.

[46] K. Henricksen and J. Indulska. Modelling and using imperfect context information. *Pervasive Computing and Communications Workshop, 2004. Proceedings of the Second IEEE Annual Conference on*, pages 33–37, March 2004.

[47] Karen Henricksen. *A framework for context-aware pervasive computing applications*. PhD dissertation, University of Queensland, 2004.

[48] Jeffrey Hightower, Barry Brumitt, and Gaetano Borriello. The location stack: A layered model for location in ubiquitous computing. In *WMCSA '02: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*, page 22, Washington, DC, USA, 2002. IEEE Computer Society.

[49] Xin Hong, Chris Nugent, Weiru Liu, Jianbing Ma, Sally McClean, Bryan Scotney, and Maurice Mulvenna. Uncertain information management for adl monitoring in smart homes. *Intelligent Patient Management*, 189/2009:315–332, 2009.

[50] 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(3):236–252, 2009.

[51] Richard Hull, Philip Neaves, and James Bedford-Roberts. Towards situated computing. In *ISWC '97: Proceedings of the 1st IEEE International Symposium on Wearable Computers*, page 146, Washington, DC, USA, 1997. IEEE Computer Society.

[52] Jadwiga Indulska and Peter Sutton. Location management in pervasive systems. In *ACSW Frontiers '03: Proceedings of the Australasian information security workshop conference on ACSW frontiers 2003*, pages 143–151, Darlinghurst, Australia, Australia, 2003. Australian Computer Society, Inc.

[53] S. S. Intille, K. Larson, E. Munguia Tapia, J. Beaudin, P. Kaushik, J. Nawyn, and R. Rockinson. Using a live-in laboratory for ubiquitous computing research. In *Proc. of PERVASIVE06*, pages 349–365, 2006.

[54] S.S. Intille. Designing a home of the future. *Pervasive Computing, IEEE*, 1(2):76–82, Apr-Jun 2002.

[55] Vikramaditya R. Jakkula, Aaron S. Crandall, and Diane J. Cook. Enhancing anomaly detection using temporal pattern discovery. *Advanced Intelligent Enviroments*, pages 175–194, 2009.

[56] Vikramaditya R. Jakkula and Diane J.Cook. Using temporal relations in smart environment data for activity prediction. In *ICML '07 Proceedings of the 24th international conference on Machine Learning*, June 2007.

[57] 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.

[58] George Klir and Tina Folger. *Fuzzy sets, uncertainty and information*. Prentice hall, New Jersey, USA, 1988.

[59] Stephen Knox, Adrian K. Clear, Ross Shannon, Lorcan Coyle, Simon Dobson, Aaron Quigley, and Paddy Nixon. Towards scatterbox: a context-aware message forwarding platform. In *Fourth International Workshop on Modeling and Reasoning in Context (MRC 2007)*, pages 13–24, August 2007.

[60] Mieczyslaw M. Kokar, Christopher J. Matheus, and Kenneth Baclawski. Ontology-based situation awareness. *Inf. Fusion*, 10(1):83–98, 2009.

[61] D. Koks and S. Challa. An introduction to bayesian and dempster-shafer data fusion. Technical Report DSTO-TR-1436, Defence Science and Tech Org, Edinburgh, Australia, August 2003.

[62] 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-Sept. 2003.

[63] Panu Korpipää, Miika Koskinen, Johannes Peltola, Satu-Marja Mäkelä, and Tapio Seppänen. Bayesian approach to sensor-based context awareness. *Personal Ubiquitous Comput.*, 7(2):113–124, 2003.

[64] E. Lefevre, O. Colot, and P. Vannoorenberghe. Belief function combination and conflict management. *Information Fusion*, 3(2):149 – 162, 2002.

[65] Hui Lei, Daby M. Sow, II John S. Davis, Guruduth Banavar, and Maria R. Ebling. The design and applications of a context service. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):45–55, 2002.

[66] Brian Y. Lim, Anind K. Dey, and Daniel Avrahami. Why and why not explanations improve the intelligibility of context-aware intelligent systems. In *CHI '09: Proceedings of the 27th international conference on Human factors in computing systems*, pages 2119–2128, New York, NY, USA, 2009. ACM.

[67] Weiru Liu. Analyzing the degree of conflict among belief functions. *Artif. Intell.*, 170(11):909–924, 2006.

[68] Beth Logan, Jennifer Healey, Matthai Philipose, Emmanuel Tapia, and Stephen Intille. A long-term evaluation of sensing modalities for activity recognition. In John Krumm, Gregory D. Abowd, Aruna Seneviratne, and Thomas Strang, editors, *UbiComp 2007: Ubiquitous Computing*, volume 4717 of *Lecture Notes in Computer Science*, chapter 28, pages 483–500. Springer Berlin Heidelberg, 2007.

[69] Seng. W. Loke. Facing uncertainty and consequence in context-aware systems: towards an argumentation approach. In *Ubicomp 2004: In Proceedings of the Workshop on Advanced Context Modelling, Reasoning and, Management*, Lecture Notes in Computer Science, Nottingham, England, 2004. Springer Verlag.

[70] 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.

[71] J. D. Lowrance, T. D. Garvey, and T. M. Strat. *A framework for evidential-reasoning systems*, pages 611–618. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.

[72] Jani Mantyjarvi and Tapio Seppanen. Adapting applications in handheld devices using fuzzy context information. *Interacting with Computers*, 15(4):521–538, 2003.

[73] Susan McKeever, Juan Ye, Lorcan Coyle, and Simon Dobson. A multilayered uncertainty model for context aware systems. In *Adjunct Proceedings of the international conference on Pervasive Computing: Late Breaking Result*, pages 1–4, Sydney, Australia, May 2008.

[74] Susan McKeever, Juan Ye, Lorcan Coyle, and Simon Dobson. A context quality model to support transparent reasoning with uncertain context. In *Quality of Context*, volume 5786/2009 of *Lecture Notes in Computer Science*, pages 65–75. Springer Verlag, 2009.

[75] Susan McKeever, Juan Ye, Lorcan Coyle, and Simon Dobson. Activity recognition using temporal evidence theory. *Journal of Ambient Intelligence and Smart Environments*, 2(3), 2010.

[76] Catherine K. Murphy. Combining belief functions when evidence conflicts. *Decis. Support Syst.*, 29(1):1–9, 2000.

[77] R. R. Murphy. Dempster-Shafer theory for sensor fusion in autonomous mobile robots. *Robotics and Automation, IEEE Transactions on*, 14(2):197–206, 1998.

[78] Jussi Myllymaki and Stefan Edlund. Location aggregation from multiple sources. In *MDM '02: Proceedings of the Third International Conference on Mobile Data Management*, pages 131–138, Washington, DC, USA, 2002. IEEE Computer Society.

[79] Jean Marc Nigro and Michèle Rombaut. Idres: A rule-based system for driving situation recognition with uncertainty management. *Information Fusion*, 4(4):309 – 317, 2003.

[80] A. Padovitz, S.W. Loke, A. Zaslavsky, and B. Burg. Towards a general approach for reasoning about context, situations and uncertainty in ubiquitous sensing: Putting geometrical intuitions to work. In *Second International Symposium on Ubiquitous Computing Systems (UCS'04)*, pages 44–50, Tokyo, Japan, November 2004.

[81] Paulito Palmes, Hung Keng Pung, Tao Gu, Wenwei Xue, and Shaxun Chen. Object relevance weight pattern mining for activity recognition and segmentation. *Pervasive Mob. Comput.*, 6(1):43–57, 2010.

[82] D Preuveneers and Y Berbers. Quality extensions and uncertainty handling for context ontologies. In *Proceedings of the 2nd International Workshop on Contexts and Ontologies: Theory, Practice and Applications*, page 62, 2006.

[83] Nissanka Bodhi Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *6th ACM MOBICOM*, Boston, MA, August 2000.

[84] A. Ranganathan, J. Al-Muhtadi, and R.H. Campbell. Reasoning about uncertain contexts in pervasive computing environments. *Pervasive Computing, IEEE*, 3(2):62–70, April-June 2004.

[85] Anand Ranganathan, Jalal Al-Muhtadi, Shiva Chetan, Roy Campbell, and M. Dennis Mickunas. Middlewhere: a middleware for location

awareness in ubiquitous computing applications. In *Middleware '04: Proceedings of the 5th ACM/IFIP/USENIX international conference on Middleware*, pages 397–416, New York, NY, USA, 2004. Springer-Verlag New York, Inc.

[86] Anand Ranganathan, Robert E McGrath, Roy H. Campbell, and M. Dennis Mickunas. Use of ontologies in a pervasive computing environment. *Knowl. Eng. Rev.*, 18(3):209–220, 2003.

[87] N. S. Ryan, J. Pascoe, and D. R. Morse. Enhanced reality fieldwork: the context-aware archaeological assistant. In *Computer Applications in Archaeology 1997*, British Archaeological Reports. Tempus Reparatum, October 1998.

[88] Steven Salzberg. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1:317–327, 1997.

[89] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5):22–32, 1994.

[90] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *In Proceedings of the Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society, 1994.

[91] A. Schmidt and K. van Laerhoven. How to build smart appliances? *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 8(4):66–71, Aug 2001.

[92] Karl Sentz and Scott Ferson. Combination of evidence in dempster shafer theory. Technical Report SAND2002-0835, Sandia National Laboratories, Albuquerque, New Mexico, April 2002.

[93] Doheon Lee Seong-ick Moon, Kwang Hyung Lee. Fuzzy branching temporal logic. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 34(2):1045–1055, 2004.

[94] Nigel Shadbolt. Ambient intelligence. *IEEE Intelligent Systems*, 18(4):2–3, 2003.

[95] Glenn Shafer. *A mathematical theory of evidence*. Princeton Unverisity Press, 1976.

[96] Philippe Smets. Practical uses of belief functions. In *UAI*, pages 612–621, 1999.

[97] Philippe Smets. Data fusion in the transferable belief model. In *Proceedings of the 3rd International Conference Information Fusion Fusion 2000*, pages 21–33, 2000.

[98] Philippe Smets. Decision making in the tbm: the necessity of the pignistic transformation. *Int. J. Approx. Reasoning*, 38(2):133–147, 2005.

[99] Phillipe Smets. The combination of evidence in the transferable belief model. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(5):447–458, May 1990.

[100] Phillippe Smets and Robert Kennes. The transfer belief model. In *Symbolic and Quantitative Approaches to Uncertainty*, pages 91–96. Springer Berlin/ Heidelberg, 1991.

[101] Thomas M. Strat. The generation of explanations within evidential reasoning systems. In *The 10th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1097–1104, August 1987.

[102] Emmanual Tapia, Stephen Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In *Pervasive Computing*, volume 3001/2004 of *Lecture Notes in Computer Science*, pages 158–175. Springer Verlag, 2004.

[103] Graham Thomson, Sotirios Terzis, and Paddy Nixon. Situation determination with reusable situation specifications. In *PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, pages 620–623, Washington, DC, USA, 2006. IEEE Computer Society.

[104] G. Englebienne T.L.M. van Kasteren and B.J.A. Kröse. Transferring knowledge of activity recognition across sensor networks. In *Eighth International Conference on Pervasive Computing (Pervasive 2010*, pages 283–300, Helsinki, Finland, 2010.

[105] Binh An Truong, Young-Koo Lee, and Sung-Young Lee. Modeling and reasoning about uncertainty in context-aware systems. In *ICEBE '05: Proceedings of the IEEE International Conference on e-Business Engineering*, pages 102–109, Washington, DC, USA, 2005. IEEE Computer Society.

[106] Binh An Truong, Young-Koo Lee, and Sung-Young Lee. Modeling uncertainty in context-aware computing. *Computer and Information Science, 2005. Fourth Annual ACIS International Conference on*, pages 676–681, 2005.

[107] Binh An Truong, Young-Koo Lee, and Sung Young Lee. A unified context model: Bringing probabilistic models to context ontology. In *EUC Workshops*, pages 566–575, 2005.

[108] Tim van Kasteren, Gwenn Englebienne, and Ben Krose. Recognizing activities in multiple contexts using transfer learning. In *Proceedings of the AAAI Fall Symposium on AI in Eldercare: New Solutions*, pages 142–150. AAAI Press, 2008. ISBN=978-1-57735-394-2.

[109] 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, September 2008.

[110] Roy Want, Andy Hopper, Veronica Falc, and Jonathan Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, 1992.

[111] Mark Weiser. The computer for the 21st century. *Scientific American*, 265, 1991.

[112] Terry Winograd. Architectures for context. *Hum.-Comput. Interact.*, 16(2):401–419, 2001.

[113] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations (The Morgan Kaufmann Series in Data Management Systems)*. Morgan Kaufmann, 2nd edition, 2005.

[114] Huadong Wu. *Sensor fusion using dempster shafer theory*. PhD dissertation, Carnegie Mellon University, 2003.

[115] Huadong Wu, Mel Siegel, and S Ablay. Sensor fusion using dempster-shafer theory ii: static weighting and kalman filter-like dynamic weighting. In *Proceedings of Instrumentation and Measurement Technology Conference IMTC '03*, USA, May 2003.

[116] Huadong Wu, Mel Siegel, and Rainer Stiefelhagen. Sensor fusion using dempster-shafer theory. In *in Proceedings of IEEE Instrumentation and Measurement Technology Conference*, volume 1, pages 7–12, 2002.

[117] Ronald R. Yager. On the dempster-shafer framework and new combination rules. *Inf. Sci.*, 41(2):93–137, 1987.

[118] Stephen S. Yau, Dazhi Huang, Haishan Gong, and Yisheng Yao. Support for situation awareness in trustworthy ubiquitous computing application software: Papers from compsac 2004. *Softw. Pract. Exper.*, 36(9):893–921, 2006.

[119] Juan Ye. *Exploiting Semantics with Situation Lattices in Pervasive Computing*. PhD dissertation, University College Dublin, 2008.

[120] Juan Ye, Adrian Clear, Lorcan Coyle, and Simon Dobson. On using temporal features to create more accurate human-activity classifiers. In *Proceedings of the 20th Conference on Artificial Intelligence and Cognitive Science*, pages 274–283, UCD, Dublin, 2009.

[121] Juan Ye, Lorcan Coyle, Simon Dobson, and Paddy Nixon. Representing and manipulating situation hierarchies using situation lattices. *Revue d'Intelligence Artificielle*, 22(5), 2008.

[122] Juan Ye, Lorcan Coyle, Susan McKeever, and Simon Dobson. Dealing with activities with diffuse boundaries. In *In Proceedings of the Workshop on How to do good activity recognition research: Experimental methodologies, evaluation metrics and reproducibility issues at PERVASIVE 2010*, Helsinki, Finland, May 2010.

[123] Juan Ye, Susan McKeever, Lorcan Coyle, Steve Neely, and Simon Dobson. Resolving uncertainty in context integration and abstraction: context integration and abstraction. In *ICPS '08: Proceedings of the 5th international conference on Pervasive services*, pages 131–140, New York, NY, USA, 2008. ACM.

[124] Lotfi A. Zadeh. A simple view of the dempster-shafer theory of evidence and its implication for the rule of combination. *AI Magazine*, 7:85–90, 1986.

[125] Daqiang Zhang, Minyi Guo, Jingyu Zhou, Dazhou Kang, and Jiannong Cao. Context reasoning using extended evidence theory in pervasive computing environments. *Future Gener. Comput. Syst.*, 26(2):207–216, 2010.