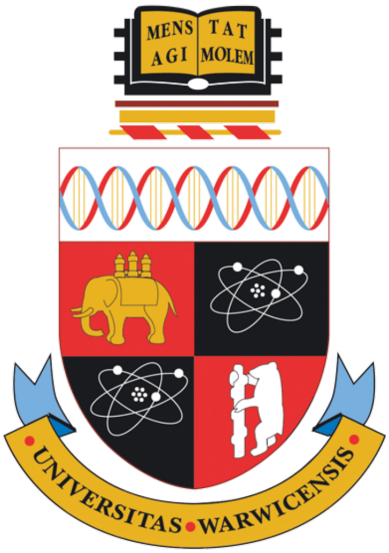


Trust assessment in the context of unrepresentative information

by

Caroline Player

A thesis submitted to The University of Warwick

in partial fulfilment of the requirements

for admission to the degree of

Doctor of Philosophy

Department of Computer Science

The University of Warwick

September 2019

Contents

List of Figures

List of Tables

List of Algorithms

I dedicate this thesis to my father, who inspires me to work hard.
To my mother, for teaching me creativity and patience.
To both of them for their unfailing love and support.

Acknowledgements

I want to thank everyone who has helped me get through this PhD, and who helped me learn so much both academically and personally. For better or worse, this has been four years I will never forget.

Firstly, I cannot emphasise enough how grateful I am for my supervisor, Nathan. This thesis would not exist without his contributions or teachings. He guided me through chocolate teapots and reinvented wheels. I am both shocked and incredibly thankful that he has supervised me through three degrees. On a personal level, I owe him a debt of gratitude for being there for me.

My appreciation goes to Mike Joy and Jane Sinclair, who as my advisors have provided me with guidance and perspective over the past four years. I'm also incredibly thankful to Phil Taylor, who always revived Flux for me to get my experiments done, and for being good company at AAMAS. And I'd like to thank everyone at Gin Lane, where I worked part time in my final year, for making it such a fun and interesting experience.

I want to thank Simon Miles for being so kind, and for giving up time and resources when he was not obliged to. His guidance gave me confidence in myself and my work. Without this generosity, I also would not have met the Lucky Biscuits at Kings College London, of whom I am especially grateful for Andreas, for his encouragement and entertainment.

To my WISC and PhD companions, notably Yvonne, Katie, Henry, Vikki, Adam, Corinne, David, Liam, James V.H and Olly, thank you for going through it with me and I wish you luck in your own endeavours. Especially to Helen, who I've shared the whole 8 year DCS journey with, and I'm so glad we've become closer over time. Finally to John Rahilly, who I thank for being such a good friend and making me a better person by example.

I am thankful for the love and support of my parents, who gave me the education and confidence to achieve this PhD, and also for supporting me in my final year when I moved back home. To Joe and William, who encouraged me in their own brotherly way, and who make us a happy family. And to Nonna, for always making me feel special and loved.

To the best friends I could ever want. Pete, your friendship has been a lifeline, and I can only hope bubbles weekends continue for the rest of our lives. To Alina, Alex, Katie, Asya, Cele, Owen, Kipp and Lysander, you guys are somehow still making me the happiest version of myself eight years on from leaving school, I love you all.

Declarations

The work presented in this thesis has not been submitted for a degree at another university.

This research was funded by the Engineering and Physical Sciences Research Council (EPSRC) (grant no. EP/L016400/1).

Parts of this thesis have been published in the following:

- This paper introduces a method to assess how representative an agent's data is of other agents, further discussed in Chapter 4.
  - Caroline Player and Nathan Griffiths. 2019. Addressing Class Imbalance in Trust and Reputation Systems. *In Proceedings of the 19th International workshop on trust in agent societies (Trust @ AAMAS '19')*
    [Best Paper award]
- This paper proposes learning and exploiting behavioural patterns in agent behaviour, discussed in Chapter 6.
  - Knowledge Engineering Review Journal, Special Issue: Adaptive Learning Agents
- These papers introduce a method to detect behaviour change in agents, discussed in Chapter 5.
  - Caroline Player and Nathan Griffiths. 2018a. Addressing concept drift in reputation assessment. *In Proceedings of the 10th International Workshop on Adaptive Learning(ALA @ AAMAS)* [Player and Griffiths, 2018a]
  - Caroline Player and Nathan Griffiths. 2018b. Addressing concept drift in reputation assessment. *In Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 20482050 [Player and Griffiths, 2018b]
- These papers first proposed tag comparisons as a stereotype assessment, described in Chapter 4.
  - Caroline Player and Nathan Griffiths. 2017b. Using tags to bootstrap stereotypes and trust. *In Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 16911693. International Foundation for Autonomous Agents and Multiagent Systems. [Player and Griffiths, 2017b]

– Caroline Player and Nathan Griffiths. 2017a. Bootstrapping trust and stereotypes with tags. *In Proceedings of the 19th International workshop on trust in agent societies (Trust @ AAMAS)* [Player and Griffiths, 2017a]

Abstract

Trust and reputation algorithms are social methods, complementary to security protocols, that guide agents in multi-agent systems (MAS) in identifying trustworthy partners to communicate with. Agents need to interact to complete tasks, which requires delegating to an agent who has the time, resources or information to achieve it. Existing trust and reputation assessment methods can be accurate when they are learning from representative information, however, representative information rarely exists for all agents at all times. Improving trust mechanisms can benefit many open and distributed multi-agent applications. For example, distributing subtasks to trustworthy agents in pervasive computing or choosing who to share safe and high quality files with in a peer-to-peer network.

Trust and reputation algorithms use the outcomes from past interaction experiences with agents to assess their behaviour. Stereotype models supplement trust and reputation methods when there is a lack of direct interaction experiences by inferring the target will behave the same as agents who are observably similar. These mechanisms can be effective in MAS where behaviours and agents do not change, or change in a simplistic way, for example, if agents changed their behaviour at the same rate. In real-world networks, agents experience fluctuations in their location, resources, knowledge, availability, time and priorities. Existing work does not account for the resulting dynamic dynamic populations and dynamic agent behaviours. Additionally, trust, reputation and stereotype models encourage repeat interactions with the same subset of agents which increase the uncertainty about the behaviour of the rest of the agent population. In the long term, having a biased view of the population hinders the discovery of new and better interaction partners. The diversity of agents and environments across MAS means that rigid approaches of maintaining and using data keep outdated information in some situations and not enough data in others. A logical improvement is for agents to manage information flexibly and adapt to their situation.

In this thesis we present the following contributions. We propose a method to improve partner selection by making agents aware of a lack of diversity in their own knowledge and how to then make alternative behavioural assessments. We present methods for detecting dynamic behaviour in groups of agents, and give agents the statistical tools to decide which data are relevant. We introduce a data-free stereotype method to be used when there are no representative data for a data-driven behaviour assessment. Finally, we consider how agents can summarise agent behaviours to learn and exploit in depth behavioural patterns.

The work presented in this thesis is evaluated in a synthetic environment designed to mimic characteristics of real-world networks and are comparable to evaluation environments from prominent trust and stereotype literature. The results show our work improves agents' average reward from interactions by selecting better partners. We show that the efficacy of our work is most noticeable in environments where agents have sparse data, because it improve agents' trust assessments under uncertainty.

Terminology

CIM	Class Imbalance Modification
DCS	Direct Comparative Stereotypes
AdWin	Adaptive Windowing
RaPTaR	Reacting and Predicting in Trust and Reputation
$\mathcal{A}$	Set of all agents
$\mathcal{A}_{te}$	Set of trustees
$\mathcal{A}_{tr}$	Set of trustors
t	Time
$n_{profiles}$	Number of profiles
$o_t \in [0, 1]$	The outcome of an interaction at time t.
$\mathcal{O}$	An agent's history of interaction experiences
n_{rf}	Number of relevant features each agent has
n_{nf}	Number of noisy features each agent has
$\overrightarrow{\tau_j}$	Set of observable features belonging to agent j
Θ	Standard deviation of relevant observable feature values
$p_{exploration}$	Probability of exploration
$\mathcal{G} \subset \mathcal{A}$	Group
p_{Gr}	Rate of gradual behaviour change
p_{Su}	Rate of sudden behaviour change
p_{leave}	Probability an agent will leave the population each round and be replaced
L	Learning interval of stereotypes
$MinPts$	Minimum points to make a cluster
ϵ	Maximum radius of a density cluster
δ	Confidence in the Kolmogorov-Smirnov test
W	An adaptive window of recent interaction outcomes $t_{W[x]}$ The time that the interaction indexed at x in window W occurred.
EU	Expected utility
$c \in \overrightarrow{C}$	A concept in the list of known concepts, $\overrightarrow{C}$
$c_c \in \overrightarrow{C}$	The currently active concept
cet	Conceptual equivalence threshold
TM	Transition matrix
tl_{cx}	The length of time c_x is believed to have been active before transitioning to c_y, a value stored in the list $L[c_x, c_y]$.

CHAPTER 1
Introduction

An MAS is a collection of agents, applied to solve real-world problems because of their autonomy, flexibility and modularity which can continuously bring together novel elements of artificial intelligence [Jennings, 2000]. However, the decentralised nature of MAS, their ability to scale, and the heterogeneous devices acting as agents, mean enforcing security protocols can be challenging [Fadul et al., 2011; Gray et al., 2006; Sicari et al., 2015]. Trust mechanisms are an alternative solution, allowing agents to learn who to interact with when they cannot rely on security protocols to enforce good behaviour. Trust fosters communication and interactions between agents, encouraging them to achieve tasks jointly and to a higher standard [Marsh, 1994a; Yamamoto, 1990].

In dynamic service-oriented computing applications, such as ad-hoc grid environments or mobile ad-hoc networks (MANETs), agents represent a multitude of connected heterogeneous devices which provide services and can join and leave the population voluntarily [Chen et al., 2016; Wang, 2016]. Agents who achieve a high level of coordination and cooperation complete tasks such that the system appears dependable and ubiquitous to the user, despite tasks being divided and delegated to multiple agents. Agents are unlikely to have the required time and resources to complete a task alone, and are typically capable of completing one component of an overall user requested task. Agents solicit help by requesting the services of others for parts of the task and then aggregating the results at the end. Trust algorithms enable agents to identify which agent is the most likely to complete a task to a high standard, including avoiding malicious agents. By considering how successfully an agent performed tasks in the past, trust algorithms estimate the probability they will perform to that standard in the future. However, agents may have insufficient time to establish strong trust relationships, and it is still vital that agents choose appropriate interaction partners [Jarvenpaa and Leidner, 1999].

Selecting an appropriate trust algorithm depends on what information is available in the application. In this thesis, we focus on improving trust models which allow for dynamic trust relationships and flexible trust assessment because MAS such as pervasive computing are becoming more popular [Sicari et al., 2015]. Several techniques for enabling cooperation in MAS have been developed, including trust, reputation and stereotypes, which can improve trust assessment by inferring an estimate of an agent's reliability from other available information. Any behaviour assessment technique relies on the availability

and accuracy of past experiences between agents, but this cannot always be guaranteed, especially in a dynamic environment.

This thesis investigates how agents can analyse the available information to determine whether or not it is relevant in a trust assessment. We demonstrate that using information which is statistically assessed to be relevant, and the appropriate method of assessment depending on which information is available, improves partner selection and agent performance.

1.1 Trust, Reputation and Stereotypes

Finding ways to accurately assess the behaviour of a target agent is an ongoing challenge. Trust, reputation and stereotype models are commonly used methods. A trustor agent uses a trust algorithm, to assess a target agent, the trustee, with direct personal experiences. Reputation mechanisms allow the trustor to aggregate experiences from other agents who have interacted with the trustee to improve the assessment. Finally, stereotype methods make the assumption that agents have observable features which correlate with their behaviour, such that these correlations can be learnt. When the trustor needs to assess a trustee with whom they have no past experiences, they may use what they have learnt about the trustee's observable features.

In this thesis, we also assume that agents have observable features, some of which are associated with their behaviour. Learning correlations between features and behaviour is the task of stereotype models, and this thesis presents techniques that improve existing approaches. We demonstrate how we can exploit correlations between observable features and behaviour as a source of information about agents even when trust algorithms take priority over stereotype models for assessing an agent's current behaviour.

Behavioural assessment methods (trust, reputation and stereotypes) all have a strong dependency on recent, accurate, data with a variety of agents, where the data correspond to past interaction experiences. Several problems with the data may render behaviour assessment methods inaccurate. First, past experiences about an agent may not exist, preventing a trust or reputation assessment to be made. Second, agents may not have past experiences with agents who have certain features, preventing stereotypes using those features to be learned. Finally, if the environment is dynamic then past data may be inapplicable, but identifying which data are useful for which agents is a challenge because of their different natures and circumstances.

A major cause of these problems is that agents are rational, a necessary and realistic assumption in MAS [Jennings, 2000]. This assumption is often made in existing work, however, there is almost no discussion of the effects of rational

agents to trust assessment. Rational agents are more inclined to choose interaction partners who they already know, leaving them unaware of the behaviour of other agents in their neighbourhood, who might be more trustworthy. Forcing agents to explore can partially solve this, however, it may not always be possible to force an agent to interact with an agent they otherwise would not do, and it can lead to worse, possibly malicious, interactions. This thesis proposes solutions to counter these problems by understanding the resulting behaviour of rational agents.

Existing trust algorithms assign a default *a priori* trust value to an agent when there is no past experiences with it, and it can be replaced by a more intelligent initial estimate such as a stereotype assessment. However, stereotype models do not consider if the past experiences use to train the stereotype model are representative of the agent being assessed. This can lead to misclassifying the agent and falsely inferring stereotypical trust.

Finally, the dynamic nature of agents can render old information unrepresentative. Agents across the MAS might change their behaviour at varying rates, and existing methods taken from centralised learning systems are too general or inflexible to be accurate for selecting current behaviour for all agents. Making an accurate trust assessment in MAS can be challenging when agent behaviours are not stationary. MAS are defined as connecting a variety of devices which can spread over a large geographical area, all experiencing different circumstances, some of which involve autonomous agents and some human interaction. Therefore, it is unrealistic to assume that agents have either consistent static behaviour or experience the same, constant, gradual rate of behaviour change. Out-dated information can hinder trust and stereotype assessment, but existing methods use techniques to forget old data which are limited in MAS. These techniques include sliding windows and forgetting factors, which do not give agents the flexibility to select data they believe are important for assessing specific agents. Additionally, stereotype models suffer from being relearnt in intervals because of the time complexity to build them, during which time behaviours may have changed.

1.2 Objectives and Contributions

This thesis aims to tackle the adverse affects of having little or unrepresentative data when making trust assessments in MAS. We set out the following objectives,

RO 1: To identify when there is insufficient past experience to learn a target agent's stereotype.

RO 2: To improve initial partner selection when there is insufficient past experience data to use machine learning or statistical approaches.

RO 3: To detect if a target agent's behaviour has changed and allow an agent to identify which past experience data are representative of the target's current behaviour.

RO 4: To exploit patterned behaviour by learning and predicting frequent behavioural changes.

We present the following contributions towards achieving the above objectives.

Contribution 1: We propose a method, Class Imbalance Modification (CIM), to judge if a stereotype model should be used to assess an agent by examining if the data it was trained on fairly represents that agent. If the method deems an agent's past experience data unrepresentative of a target agent, and therefore unlikely to have an accurately learned stereotype for the target, it suggests using an alternative assessment technique [Player and Griffiths, 2019a]. (Achieves RO 1)

Contribution 2: We introduce the Direct Comparative Stereotype (DCS) assessment method, which does not use past experience data. An agent infers another agent's behaviour based on their observable similarity. Our approach can be used when past experience data is not available or not appropriate to use [Player and Griffiths, 2017a,b, 2019a]. (Achieves RO 2)

Contribution 3: We propose a technique to bootstrap stereotypes, AdWin Tree, to statistically determine if the behaviour of the agents belonging to a stereotype has changed over time. Only past interaction data after the identified point of change is used to make a behaviour assessment, this reduces noisy data about stereotypes and improves accuracy [Player and Griffiths, 2018a,b]. (Achieves RO 3)

Contribution 4: We describe a technique to learn patterns in dynamic agent behaviour to improve partner selection by pre-empting behaviour changes. Our method, RaPTaR, prevents agents from spending time learning behaviours they have already seen [Player and Griffiths, 2019b][in submission]. (Achieves RO 4)

1.3 Thesis Structure

The remainder of this thesis is organised as follows.

- Chapter 2, a literature review covering trust, reputation and stereotypes. We identify the problems that dynamic behaviours can pose to these systems, the existing coping mechanisms and some techniques from other fields which may be applicable solutions.

- Chapter 3 defines the scope of the work using a formal notation of the agent environment and the experimental evaluation we used for the research which contributed to this thesis. This chapter serves as a reference to the main contribution chapters. Any variations are then also explained together to easily compare them alongside their justifications.

- Chapter 4 studies the impact of using unbalanced data in stereotypes. We present our model CIM to identify when stereotypes are an appropriate assessment tool, and propose using our method DCS when stereotypes are inadequate. This chapter addresses RO 1 and 2, using Contributions 1 and 2. We have published the following papers in this area:

  - AAMAS '17 *(extended abstract)* - Tags to bootstrap trust and stereotypes;
  - Trust @ AAMAS '17 - Tags to bootstrap trust and stereotypes;
  - Trust @ AAMAS '19 - Addressing class imbalance in trust and stereotype assessment.

- Chapter 5 explores two techniques to detect changes in dynamic agent behaviour over time, AdWin Tree and cluster comparisons. The work is evaluated with existing methods of coping with dynamic behaviours. This chapter addresses RO 3, using Contribution 3.

  We have published the following papers in this area:

  - AAMAS '18 *(extended abstract)* - Addressing concept drift in reputation assessment;
  - ALA @ AAMAS '18 - Addressing concept drift in reputation assessment;
  - Trust @ AAMAS '19 - Addressing class imbalance in trust and stereotype assessment.

- Chapter 6 presents a technique to learn and exploit patterned behaviour. Our method uses learned concepts as part of estimating trust rather than raw data, and considers how behaviours seen in the past may reoccur rather than estimating trust based on the most recent past interaction data. This work focuses on analysing the behaviour of agents of known

groups as opposed to identifying the stereotypes. This chapter addresses RO 4 using Contribution 3 and 4.

We have published the following paper in this area:

- KER Journal (special issue on adaptive learning agents) - Improving trust and reputation assessment with dynamic behaviour.

- Chapter 7 reflects on the objectives of this thesis, our contributions, the limitations and future work for this research field.

CHAPTER 2
Related Work

This chapter is a discussion of the related work relevant to this thesis. Section 2.1 is an overview of MAS and the importance of trust assessment. Section 2.2 formally defines trust as it is used in this thesis, how to measure trust, which information is useful or available as input to trust assessment and existing algorithms. Section 2.3 introduces stereotype algorithms as a method to provide initial, or *a priori*, trust assessments when data specific to trust algorithms are not available. Throughout this thesis, we assume that agents exhibit stereotypical behaviour, where agents have observable features, some of which correlate with their behaviour. Section 2.4 then discusses how dynamic agent behaviour prevents accurate trust assessment, and the limitations of existing techniques to cope with it.

2.1 Multi-agent Systems

MAS are an increasingly popular choice of system model because of their ability to connect multiple, homogeneous devices and scalability. New technologies and artificial intelligence can be continuously integrated with existing MAS [Jennings, 2001, 2000]. Such open and distributed systems struggle to achieve the highest average welfare for its members, known as social rationality, because selfish agents inherently refuse to take actions which incur personal cost but benefit the group [Hogg and Jennings, 2001; Jennings and Campos, 1997; Kalenka and Jennings, 1999]. Coordination and cooperation mechanisms encourage agents to work together to achieve high average beneficial outcomes as quickly as possible, despite the occasional personal costs to individual agents [Durfee, 2001]. Trust mechanisms can achieve coordinated and cooperative behaviour through learning about individual agent behaviours and how best to respond to them. The concept of *trust* between these agents is fundamental to MAS because agents must delegate, cooperate, coordinate and negotiate, which are social activities that are all based on trust [Castelfranchi and Falcone, 1998; Granatyr et al., 2015].

Agents can autonomously learn behavioural preferences through *norms* and *conventions*. Norms incite desired system level behaviour by punishing agents who do not conform to certain behavioural standards [Axelrod, 1986]. Norms can be effective and are still an active research area, however, they are out of the scope of this work because they require a central authority or some

amount of policing within the network which is an expensive overhead, not always available, and fails to scale adequately [Bicchieri et al., 1997; Marchant et al., 2014; Santos, 2017]. Establishing conventions encourages cooperation when agents learn over time which action choices receive the best response depending on the action choices of other agents [Kittock, 1993]. Over time, agents learn to have coordinated behaviour, for example, driving on the left hand side of the road to avoid collisions. Norms and conventions help an agent choose how to behave or what action to select. By selecting actions which are coordinated within the group, all agents can improve their welfare. In comparison trust mechanisms do not decide agents' action choices, but assess the probability of other agents' behaviour, and therefore trust can complement norms and conventions. Finally, an area of study which relates to rewards and punishments for agents in an attempt to guide their behaviour includes *principal agent theory* [Grossman and Hart, 1992]. The theory, which is realised in multiple real world contexts from economics to computer science, describes how one agent's decision making processes can have consequences on another's behaviour. Specifically, there is work in MAS which aims to improve agent's utilities by identifying the correct amount of reward or punishment one agent should ascribe to another to achieve the desired behaviour from that agent. This can be a difficult situation when one agent does not know how another will respond to particular kinds and amounts of rewards and punishments. Enforcing such incentives can be difficult, and typically it assumed the agents much follow the reward and punishment scheme through a contractual obligation [Cerutti et al., 2015a,b], which we do not assume exists in this thesis.

Finally, security protocols are another mechanism to enforce cooperative behaviour. Security protocols require policing, or some level of centralism to enforce constraints, which open, distributed MAS lack [Fadul et al., 2011; Gray et al., 2006]. Good practice in cyber security for distributed systems uses open algorithms, which, given enough time and resources, an attacking agent will eventually subvert. Trust algorithms can contribute to detecting and managing malicious agents, which adds a layer of protection to compliment cyber security systems [Jøsang, 2012; Kagal et al., 2001]. Malicious agents will try to exploit a lack of central authority, and take advantage of other agents by receiving data, money or services in exchange for nothing or very little. Additionally, some applications of trust algorithms in MAS, such as the Internet of Things (IoT), are networks of connected heterogeneous devices, which are so different that security protocols can be incompatible between them [Sicari et al., 2015]. Trust algorithms can be used to not just identify who is safe to interact with, but who will achieve the task to the standards the agent requires.

Trust and reputation mechanisms allow agents to identify interaction part-

ners who have demonstrated desirable behaviour in the past, by statistically analysing past interaction outcomes with them. Existing literature surrounding trust in MAS is vast because definitions, applications and methods vary, as well as addressing different complications which can arise. The discussion on existing trust and reputation continues in Section 2.2, where we narrow the scope of the literature by our definition of trust, seminal background material and state-of-the-art techniques in dynamic MAS. Many trust models exist for specific applications, but some general problems to trust and reputation systems are realised across multiple contexts. Aligned with existing work, we evaluate in a generalised, formal multi-agent environment but, also describe some applications to illustrate our motivations and objectives.

A common example of trust and stereotype models used in MAS is between buyers and sellers in online marketplaces. Agents using the trust algorithms STAGE, BLADE, BRS and HABIT to evaluate the trustworthiness of either a seller or a buyer before engaging in a transaction [Şensoy et al., 2016; Jøsang and Ismail, 2002; Regan et al., 2006; Teacy et al., 2012]. Some stereotype models also consider a marketplace example, specifically using the eCommerce dataset for evaluating the work. StereoTrust categorises consumer interests into stereotypes using written reviews of products [Liu et al., 2009].

Trust is also used to encourage safe and successful file transfers between agents in peer-to-peer (P2P) systems [Xiong and Liu, 2004]. P2P systems are an example of an open-distributed MAS where agents have the great opportunity to share and spread useful information, but as agents can be anonymous or easily disguised, the chances for attacks or low standards of task performance increase. Distributed trust and reputation systems can offer huge improvements to these networks [Javanmardi et al., 2014; Kamvar et al., 2003].

Our last example is the growing application of pervasive (or ubiquitous) computing, where agents delegate tasks depending on resources, and collate the results before returning them to a client who only sees the process as a seamless transaction with a service provider [Boswarthick et al., 2012; D'Angelo et al., 2017; Klusch and Gerber, 2002; Schmeck et al., 2003]. Such systems can often be highly dynamic, as well as agents forming small groups of other agents they trust and frequently delegate to, or groups of the most available agents for the time and place [Nguyen, 2017].

2.2 Trust and Reputation

Trust and reputation assessment methods improve agents' abilities to have successful social interactions by selecting agents to engage with who are statistically more likely to behave in the desired manner. In this section, we first discuss the

definition of trust, which can vary across computational trust models, before selecting the one used throughout this thesis. We also consider the following three questions to distinguish trust model types at a high level, i) what variables describe an agent's trustworthiness? ii) what data are available to use for trust assessment? iii) how are those data used to make the assessment?

2.2.1 Definition

The concept of trust is ubiquitous in everyday human life, but it is an abstract concept which needs formalising to be captured as a useful tool for agents in distributed artificial intelligence [Marsh, 1994a]. The research area was motivated by systems which had not yet formalised what trust meant to agents, and therefore trust was implicitly assumed [Rosenschein, 1986]. As MAS have become an increasingly large research area and more prominent tools in the real world, they have become increasingly complex. Trust needs to be assessed on a per agent basis because conflicts between agents' goals and priorities can exist as well as differing and continuously changing agents' abilities. An early definition of trust suggests trust is necessary "if a bad outcome would make you regret your action", indicating trust assessment methods can help you select interaction partners whose outcome you profit from [Luhmann, 2000]. Additional reviews on trust and reputation have since described trust and reputation as one agent's estimate of another agent's behaviour before willingly interacting with that agent [Keung and Griffiths, 2010].

We adopt a formal definition of trust widely used in MAS literature, that trust is the probability one agent believes another has to perform the requested task before the task has taken place, and given that it may affect the agent's own decisions [Gambetta, 2000]. This definition does not clearly state how to calculate the probability of a successful interaction with a particular agent. The remainder of this section discusses the practicalities of assessing and using trust in MAS.

Trust is a form of reputation, where the trustor uses only their own personal information about a trustee to make the assessment. Reputation systems aggregate information about the trustee from other agents who have had their own experiences with them [Resnick et al., 2000]. One of the most visible applications of this is a seller's reputation on a system such as eBay, however this is a centralised system where interactions between a seller and multiple agents are easily aggregated. However, in MAS this can be more complicated depending on the availability of witness providers to the trustor and their reliability.

2.2.2 Measures of Trust

Most literature following Gambetta's definition of trust use a single value to represent trust. There exists a predefined range to indicate the scale of trustworthiness, and single value trust allows us to make direct comparisons between agents' trustworthiness. However, some existing work uses a variety of other heuristics.

Confidence, also referred to as reliability, and trust heuristics, both indicate some chance that an interaction with an agent can lead to a good or bad outcome [Luhmann, 2000]. However, we formalise their differences as trust as the expectation we have in one agent to achieve a task, while confidence indicates how variable that expectation is. In some existing trust models, confidence is calculated and used to weight the trust value, assuming that high variance in trust should lower the final trust value, as in FIRE [Huynh and Jennings, 2004]. Confidence is assessed by considering the quantity of data which was used in the trust calculation. For example, in FIRE, the *reliability* is $\frac{n}{m}$ where n is the number of ratings and m is a predefined threshold of how many instances are considered "reliable", and once $n \geq m$, the reliability is always 1. This does not take into account the time that each n instance was recorded. FIRE also uses a *rating deviation reliability* measure, to indicate how much variance there is in reports of a target agent, and so how volatile their behaviours might be. The overall reliability considers both these factors. REGRET also uses a reliability measure based on the fluctuation in the reputation reports [Sabater and Sierra, 2001a], though REGRET does not use this reliability to explicitly impact on partner selection choices.

Marsh and Dibben expand on Marsh's own definitions of trust and ignorance [Marsh, 1994a], to define the concept of trust on a multidimensional scale rather than a stand alone value, advocating the need to consider trust, distrust, mistrust and untrust [Marsh and Dibben, 2005]. These definitions are used synonymously in some literature however, Marsh argues they have the following distinctions which advanced systems should consider. Mistrust is defined as having once placed trust in an agent but no longer trusting them. Distrust is an assessment that the agent will actively work against you, and is also distinct from the negation of trust. Untrust is a positive measure of trust in another agent but is insufficient for cooperation to emerge and perhaps a suggestion of high variance in the trust value. These definitions are fine grained, and most computational trust models within the scope of this thesis do not give agents the ability to distinguish between these levels of trust. Some models, such as STAGE [Şensoy et al., 2016], identify separate feature patterns for trustworthy and untrustworthy agents. However, the two groups are segregated by which

side of a trust threshold they are, meaning they are still using a single heuristic to define trust and that untrust actually means distrust from the above definitions. The definitions from Marsh and Dibben can be important when there is a strong focus on identifying attackers or bringing about emerging cooperative behaviours in societies and could be a future avenue of computational trust research.

Ramchurn *et al.* make the distinction in trust between giving agents the ability to assess trust in their partner, and how to use that trust value [Ramchurn et al., 2004a]. Throughout this thesis, we assume that an agent will choose the interaction partner which is available with the highest level of assessed trust, but that the level of trust in an interaction partner cannot affect an agent's action choice. Only the trust assessment method is within the scope of our work.

2.2.3 Information Sources

Different methods of assessing trust can use different sources of information. The available information that could contribute to trust assessment is subjective to the context. Agents should make use of as much of the information available to them as possible however, there are many considerations. First, what types of information are available? Second, can the agent monitor or sense that information is accurate, is consistently available, and will that information be worth the cost of obtaining [Falcone and Castelfranchi, 2001]? Finally, if other agents are providing information, is it reliable and honest?

Within this section, we include how stereotype models use additional information to correlate trust with agents' observable features. Stereotype models are a large focus of the research in this thesis, and a longer discussion on them can be found in Section 2.3.

Past Interaction Experiences

The most common source of information to trust algorithms is past experiences with the target agent. Reputation algorithms build on this, by allowing an agent to use past experiences with a target agent from another agent to improve the assessment. However, deciding who provides honest witness reports is a research question that is not the focus of this work, but is addressed in some of the following literature [Regan et al., 2006; Teacy et al., 2006].

Reputation providers, also known as witnesses, who share their past experiences of a trustee to another agent to improve the quantity of data that agent has in their assessment, need to be reliable themselves. Identifying trustworthy reputation providers is an open research area as they may be dishonest with their feedback [Dellarocas, 2000; Teacy et al., 2006; Whitby et al., 2004; Yu

and Singh, 2003]. Whitby *et al.* split methods of addressing unfair ratings into two categories: endogenous and exogenous. Endogenous discounting uses only witness reports to statistically identify if the report is likely to be unfair and weights it or excludes the report accordingly. Exogenous discounting uses external information such as the witness's personal reputation to identify if the report might be false. This is a separate research area, with much literature already devoted to it, and it is therefore not the focus of this work.

Three problems identified in real-world reputation systems are first, encouraging people to provide feedback at all, second, to encourage negative feedback as many users refrain out of fear of retaliation, and third, to address dishonest reporting [Resnick et al., 2000]. One problem of realistic reputation networks without a centralised system to disseminate new reputation information is the time lag that occurs when reputation information spreads by "word of mouth" or gathering evidence from trusted advisors. Not only does the time lag prevent useful information being spread as quickly as possible, but it offers a time gap for cheaters to act, a problem especially identified in markets where transactions occur over more than one time period, e.g. the long delivery times of international markets [Jøsang and Golbeck, 2009; Yao et al., 2012]. False negative feedback, potentially created by competitors was found to exist in an online review system of hotels that had a zero-cost to join compared to similar system which required payment to use [Mayzlin et al., 2014].

Context-aware Trust

A trust model associates trust with a specific agent identity, however context aware trust models take features of the interaction context into account in a trust evaluation [Jøsang et al., 2007; Sabater and Sierra, 2002]. The context of an interaction typically includes the agents involved and the service or task in consideration. CATrust for example, considers features of the environment such as channel conditions, node status, service payoff and social disposition [Wang et al., 2017]. CAST, which is a context aware trust and stereotype model, considers how combinations of both environmental and agent features can be associated with trust. Additionally, CAST identifies if trust can be inferred from one context to another [Zhou et al., 2015]. Other models which translate trust between contexts include [Nguyen and Bai, 2014; Rettinger et al., 2007; Sabater and Sierra, 2001a].

Similar to context-aware trust methods, meta-models use information about the environment to affect the trust assessment. Hoelz *et al.* show the efficacy of bootstrapping existing non-adaptive trust algorithms with a meta-model [Hoelz and Rakga, 2015; Hoelz and Ralha, 2015]. This is the only work from the trust

and reputation literature to consider how existing models are rigid and focus on specialising internal parameters rather than the process of when and how to use trust and reputation mechanisms.

Tags

In this thesis, we present research inspired by tag-based cooperation. This work draws upon the stereotype assumption, that agents' observable features can be correlated with behaviour and an in depth discussion of stereotype models is presented in Section 2.3.

Research to encourage cooperation in MAS using tags is inspired by evolutionary biology, where organisms survive as a result of kin selection and organisms are selectively altruistic within their family even at a cost to themselves [Hamilton, 1964a,b]. Identifying kin or family is done by comparing *tags* which are observable values. More descriptively, the phenomenon is described as the Green Beard effect by notable evolutionary biologist Dawkins, where two strangers will be altruistic towards each other if they both share the trait of having a green beard [Dawkins, 1976].

Tag techniques do not require any previous interaction data to make assessments of an agent's future performance but can encourage cooperative behaviour between agents to emerge. Agents interact cooperatively with agents who share the same observable tags as them. An example of tag-type trust in humans was observed in Japan, where people have a very indirect way of speaking to each other, so they don't always say what they mean, but this is their culture and so they recognise their own characteristics in other people and ultimately trust each other [Yamamoto, 1990]. Tags, and the observable features used in stereotype models are not perfectly synonymous. Tags are not a cause of an agent's behaviour, merely a tool by which agents distinguish who to cooperate with. Whereas in stereotype models, some features are considered to be a cause of behaviour, called *relevant features*, as opposed to features which are also observable about an agent but have no bearing on their behaviour, called *irrelevant features*. We consider how by observing all features as tags, as in tag cooperation, agents can still select a good partner, or avoid a bad one.

2.2.4 Methods

There are multiple trust models which use different techniques to infer a trust value. In this thesis, we present methods to improve trust algorithms by bootstrapping them. The following is a review of some prominent trust algorithms and the issues faced by cutting edge models to show their limitations and how our research can improve on them.

Beta Probability Functions

A common trust assessment technique is to use beta density probability functions first proposed in Beta Reputation System (BRS) by Jøsang and Ismail [Jøsang and Ismail, 2002]. The outcome of an experience with another agent is perceived as either good or bad, which becomes an input parameter to a beta probability function. The expectation of this distribution is known as the *belief* in the agent being evaluated. However, to prevent small data samples from dramatically misrepresenting the estimated behaviour of another agent, this belief is used in *subjective logic*. Subjective logic also takes into consideration an *a priori* which has a default, and in the original BRS work a middling value, which is weighted higher when there is less data. We discuss how the *a priori* can be replaced with more intelligent estimates throughout this thesis. TRAVOS extends BRS and aims to eliminate the assumption that reputation sources will always be accurate or have the same perceptions [Teacy et al., 2006]. Therefore, this algorithm is more appropriate in a context where reputation sources are likely to be lying or have subjectively different opinions. StereoTrust also uses the expectation from a beta probability function to derive a stereotypical assessment, and in this instance the parameters are an aggregation of the positive and negative experiences had with all agents belonging to the identified stereotype [Liu et al., 2013]. The stereotype model STAGE builds on BRS to calculate direct trust before learning stereotypes [Şensoy et al., 2016].

Reputation

A reputation system is defined as collecting, distributing and aggregating feedback about participants' past behaviour [Resnick et al., 2000]. Reputation can be introduced into trust models, or be stand alone, by aggregating evidence from other agents about a target and using those to estimate the target's behaviour [Taylor et al., 2018]. For example, BRS uses an aggregate of all the good and bad experiences from witness providers as the parameters to a beta probability function, which improves the accuracy and reliability of the trust assessment [Jøsang and Ismail, 2002].

Problems arise in reputation systems because witnesses have incentives to lie when they are trying to promote business for another agent. Equally, perceptual bias will mean two agents will think differently about a target agent, and so their experiences will not be considered the same. In this case, if there is some offset in perceptual bias which can be learnt, then recommendations can be adjusted appropriately and therefore still useful, as is demonstrated in the models TRAVOS and BLADE [Regan et al., 2006; Teacy et al., 2006; Zupancic and Trcek, 2017]. Alternatively, a window size for witness reports which varies

based on the frequency of reports deemed to be unfair can be used [Zhang and Cohen, 2006]. PeerTrust is a P2P decentralised system which uses the credibility of sources as one facet of its reputation model [Xiong and Liu, 2004]. Feedback from reputable peers is weighted more highly. A reputable peer is determined with two credibility metrics. The first is based on if they are trustworthy to interact with. There is a generalised assumption that agents who have a trustworthy behaviour are also trustworthy reputation sources. The second credibility measure of a source is based on how similarly they gave feedback to mutual interaction partners. One major distinction between different types of reputation systems is whether they have a central authority or are decentralised, and while we focus on decentralised systems, some contributions regarding centralised reputation systems are also included for comparisons.

Indirect reciprocity can emerge through a reputation technique called *image scoring*. Agents are judged by the donations they make to others in a donation game scenario, thus giving them a reputation as either a valuable or stingy member of society [Griffiths, 2008; Nowak and Sigmund, 1998].

Fuzzy Logic

Agents may perceive outcomes differently, or values of different features are not objectively good or bad, for example an agent may be "young" or "quite helpful", therefore trust models exist based on *fuzzy logic* which attempt to take these subjective descriptions into account [Griffiths, 2006; Griffiths et al., 2006; Manchala, 2000; Ramchurn et al., 2004b; Song et al., 2005].

Multi-faceted Approaches

Multi-dimensional trust assessment methods allow agents to hold different levels of trust in another agent based on different tasks or features, where as previously the assumption has been there is only one quantifiable amount of trust between any two agents. One example, known as REGRET, uses individual trust, social reputation and an ontological dimension, to combine different aspects of past interactions to assess their behaviour [Sabater and Sierra, 2001b]. This work is presented explicitly in the context of an online marketplace where a seller can be evaluated on characteristics such as value, price and quality. The disadvantage of such an approach is its inapplicability to other contexts or the level of specificity require to transfer it to a real-world application. Whereas HABIT is deliberately created to handle different representations of contexts, and it can also make use of information in one context and apply it to another based on how behaviours in those contexts have correlated in the past [Teacy et al., 2012]. One advantage of a multi-dimensional approach to trust is by assessing different aspects of an

interaction, agents can coordinate who are appropriate partners for achieving specific tasks based on their strengths in interactions [Griffiths, 2005].

A multi-faceted trust and reputation system which uses the strong mathematical foundations of the BRS is FIRE [Huynh and Jennings, 2004]. The four levels of behaviour assessment are interaction-based trust, role based trust, witness reputation and certified reputation. The first three are existing techniques, however FIRE includes certified reputation which is similar to references. The agent being evaluated is allowed to provide its own reputation by selecting the reports from other agents it believes are best. This technique is assumed to overestimate performance as a rational agent, and will select the best reports to represent itself. The technique is particularly effective in a network where an agent's neighbourhood restricts whom it can get its reports from, which is a realistic assumption. This method means an agent who has no mutual connections with a potential partner, and therefore no witness reports, still has access to a reputation.

Transitive Trust

Transitive trust is a technique which when an agent has either none, or very little experience with another agent, and relies on third party agents' trust in each other. For example, if Alice trusts Bob, and Bob trusts Derrick, should Alice trust Derrick? This type of reputation is particularly prevalent in P2P networks because of their decentralised nature. NICE is one example which has shown how transitive trust can identify cooperative groups of agents in a network using transitive trust despite a high proportion of malicious users [Lee et al., 2003]. EigenTrust is another such model which demonstrates its capabilities in a decentralised P2P network [Kamvar et al., 2003]. EigenTrust can be effective when there is a lot of information about an agent to fairly assign it a global reputation, however it is not as effective when there are sparse data. While this is not the focus of their work, they use a limited approach to combat it, where there is an assumption that a set of pre-trusted peers is already known to the agent population. Due to biases, dispositions and personal experiences, it is not always reasonable to assume transitive trust models are applicable.

Machine Learning Approaches

More recently, trust has been considered a machine learning problem. Lu *et al.* claim that mathematically rigorous models lack adaptability, as they do not take environment changes into consideration, which in multi-agent systems is inherently the case [Lu and Lu, 2017]. Adaptability is especially important to agents if they have to communicate with humans as well as other agents.

Machine learning tools help the agents learn and reason about their partners so they can autonomously act on behalf of another user [Glass et al., 2008]. Neural networks are gaining popularity in many areas of artificial intelligence, and are used in the SOM and CAST trust and stereotype models [Lu and Lu, 2017; Zhou et al., 2015].

Reciprocity

The existing work discussed above assumes that agents will behave the same regardless of their partners. However, some literature considers agents who can choose their action at any time, and the task becomes to identify agents who will cooperate with them specifically. *Reciprocity* trust models approach this in two ways: direct and indirect [Nowak and Sigmund, 1998]. Direct reciprocity captures the intuition "you scratch my back, and I'll scratch yours". This is a trust algorithm because it assesses how your partner behaved towards you in the past. Agents are inclined to continue cooperating as a result of the *shadow of the future* effect, which suggests that agents should continue to cooperate with each other in case they meet each other again and want to be treated nicely. Indirect reciprocity is a property found in a network of agents who either do not know each other or do not have the memory to remember each other. Cooperation emerges because agents act how they have been treated previously, known as a tit-for-tat strategy [Axelrod, 1984; Trivers, 1971].

2.3 Initial Trust Estimates and Stereotypes

Trust algorithms use direct or indirect experiences with a target agent to make assessments about that agent's future behaviour, which means that information is assumed to not be useful for assessing another agent. However, this information can be useful in other contexts [Zhou et al., 2015]. Some trust and reputation models have an *a priori* value indicating trustworthiness which is not based on direct past experiences. This can either be some initial default value for trust which an agent has to improve through interactions, or this value can be substituted with a more intelligent initial estimate of an agent. In the case of the former, a default value implying the agent is neither good nor bad would be 0.5 when trust is in the range $[0, 1]$. In the latter case, the *a priori* can depend on other factors. For example in HABIT, agents keep track of first interactions with unknown agents and use the average experience to assume another unknown agent will behave the same [Teacy et al., 2012]. Teacy *et al.* state that future work for HABIT includes applying the *a priori* approach to more specific groups of agents to improve their accuracy. Alternatively, a more

recent research area called *stereotypes* attempts to use past interaction experiences with all agents to learn correlations between agents' observable features, instead of their ID, with trust. A new agent can then be assessed based on their observable features.

Swift Trust is a theory describing how, in the context of group work, trust can be immediately established despite there being no direct experiences between the agents. The contexts are typically ad-hoc groups formed to do specific tasks such as emergency rescues or a specialised task force where the agents trust each other because of their skill set that brought them together [Debra et al., 1995]. This concept inspired stereotypes in MAS, where trust can be learned and inferred from the observable features an agent displays.

Stereotype literature exploits the assumption that agents exhibit observable features, e.g., product prices, product descriptions and photo aesthetics. These features are recorded with the outcome of the interaction into an agent's memory. Using these data, machine learning tools can learn correlations between the observed features and outcomes of interactions. Another example includes a web crawler learning correlations between website characteristics and how trustworthy that website is, for example, websites with more pop-ups, or servers located at certain geographical locations, might be less trustworthy [Seckler et al., 2015]. When a new agent joins the system for whom no one has any experience of, stereotyping can draw conclusions about them by classifying their observable features. Stereotyping still requires data collection, though it can perform well with significantly less data than trust models and a common way to apply these models is to bootstrap a trust algorithm to use stereotyping while there are insufficient data for the trust model [Burnett et al., 2010; Liu et al., 2009, 2014].

Seminal work by Burnett *et al.* , which has been used as the basis for the work achieved thus far in this project, is to build a decision tree to identify which features correlate with behaviour. With enough data, this work is resilient when some features do not correlate with behaviour but the agents are unaware of this, known as *irrelevant features* [Burnett et al., 2010]. Liu *et al.* discuss how decision tree stereotype models are increasingly popular because they are human readable [Liu et al., 2014]. This work is shown to be effective in group situations. Supporting the SwiftTrust hypothesis, Burnett *et al.* showed that bootstrapping a trust algorithm with a stereotype model improved cooperation and reward when agents were forced to interact with a group of unknown agents.

Other prominent stereotype models include STAGE, which focuses on identifying stereotypical characteristic of cheaters [Şensoy et al., 2016]. STAGE addresses the limitation in the work from Burnett *et al.* where features are simplistic e.g. age and location, and instead exploits more complicated features such as "agents with specific patterns of relationships are less trustworthy".

These complex features are represented with an ontology, which has the power to describe relationships between features as well as their values. Graph analysis applied to these ontologies identifies patterns of trustworthy and untrustworthy agents. The patterns then train a linear regression model, to correlate graph patterns with behaviour. A negative or positive pattern is a subgraph that repeats frequently in the ontological-graph representing either untrustworthy or trustworthy agents, respectively. This method allows each agent to maintain one graph for desirable partners, and one for undesirable partners which is a similar notion to trust and distrust, as discussed above. This work aims to identify patterns of behaviour in particularly bad agents such as whitewashers and Sybil attackers. One limitation in this work is the assumption that "motifs may be observed as long as the agent does not change its behaviour". We address the issue of dynamic behaviour further in Section 2.4. CAST is another model which focuses on sophisticated attackers who do not simply have one obvious feature identifying them as cheaters [Zhou et al., 2015]. The model uses deep learning to build a belief network which uses latent context-specific information to classify an agent's expected behaviour and identify malicious patterns. The *generalised interaction partner model* uses clustering to stereotype human interaction partners. This work has been applied in the real world, to robots working in disaster recovery zones [Wagner, 2009, 2010, 2012].

Simple correlation between features and behaviour is theoretically a useful tool, however realistically there can be complications. Further work by Burnett *et al.* explores how different social contexts can provoke different behaviours in some types of agents [Burnett et al., 2011]. Different social contexts include the set of interaction partners agents maintain, and the type of relationship they share with them, for example, an agent sharing a *friendship* with several criminals is an indication of distrust, however a *counsellor* with the same connections is maintaining professional relationships and therefore this should not impact their trustworthiness.

Stereotypes are useful to existing agents who need to evaluate a newcomer to the population however, if a newcomer to the system does not have any information they cannot build a stereotype model yet. *Stereotype reputation* is a potential solution to this problem, as agents can share their stereotype models with a new agent [Burnett et al., 2013]. Stereotype reputation is subject not only to those problems with sharing direct experiences in a trust model, but also by sharing stereotypes there is a risk of reinforcing biased views.

The StereoTrust model is evaluated using the *epinions* dataset to demonstrate that their stereotyping is more insightful than other trust models [Liu et al., 2009, 2013]. This model relaxes the assumption that agents can only belong to one stereotype. This model combines the expected behaviour of a

stereotype the agent is associated with, weighted by the probability they are a member of that stereotype. We use a similar definition of agent groups, formalised in Chapter 3. The literature also demonstrates how a dichotomy-based extension of the model improves its accuracy, similarly to how STAGE allows agents to maintain separate models for good and bad partners. Uncertainty in direct trust and reputation assessment is accounted for by the quantity of data used. Only StereoTrust considers that a lack of data might impact the accuracy of stereotypical trust assessment [Liu et al., 2009]. However, this model assumes that stereotypes are identifiable and so, similarly to trust algorithms, StereoTrust uses the quantity of data associated with agents from a groups as a measure of uncertainty. A lack of data about a type of agent can prevent identifying the stereotype in the first instance.

One problem common to all stereotype models includes missing, inaccurate or maliciously falsified observable features. Missing values can be imputed as in machine learning, however, this can be inaccurate and does not account for inaccurately perceived feature values [Burnett et al., 2013]. Another open research area is stereotypical bias, divided into perceptual and behavioural bias. Agents with perceptual bias have a partial ordering of agents they prefer to interact with and how they perceive the outcomes of those interactions. A behavioural bias affects how the agent behaves towards others based on their observable features [Burnett, 2011]. Finally, stereotype models can incorrectly classify an agent resulting in an inaccurate stereotypical trust assessment, leaving agents unfairly ignored or risking interactions with bad agents.

2.4 Dynamic Behaviour

Agents are defined as having autonomy over their own behaviour [Jennings, 2000], and flexibility to adjust that behaviour to meet the goal [Jennings, 2000; Wooldridge and Jennings, 1995]. A key requirement of trust and reputation mechanisms in large scale open systems is to adapt to highly dynamic environments [Luhmann, 2000; Ramchurn et al., 2004a]. Throughout the existing trust, reputation and stereotype models discussed previously, they either do not consider that agent behaviour may change over time or use limited techniques to address this issue, and do not evaluate their models with dynamic agent behaviour. Instead, trust is correlated with an agent, a context or a feature set, which are inflexible to changes. As a result of the discussion below, addressing dynamic agent behaviour is one of the main objectives of this thesis.

Agents act autonomously in a multitude of applications where their decisions are driven by their task load, beliefs, desires, intentions and resource availability amongst other things. The dependencies between these factors and the ultimate

effect on agents' behaviour are highly complex [Brazier et al., 1997; Rettinger et al., 2007]. In this thesis, we explore how trust models need to adapt to changing agent behaviours.

2.4.1 Dispositions and Biases

Agents who have a disposition or a bias skew their trust perception of others, such that two agents with the same information about another would have different levels of trust in that agent. Marsh *et al.* discuss how dispositions can be optimistic or pessimistic [Marsh, 1994b]. Optimists ascribe a high initial trust in an agent they have no experience of, while a pessimist will assume a low trust rating. Optimists weight good interactions higher than bad ones and vice-versa for pessimists [Griffiths, 2005].

To account for bias from reputation providers, Burnett categories the approaches as exogenous or endogenous [Burnett, 2011]. Exogenous approaches account for perceptual biases by weighting reputation information depending on how well aligned the reputation provider's recommendation have been with the agent's actual experience in the past. Prominent models which follow this approach include HABIT, BLADE, TRAVOS and QADE [Regan et al., 2006; Teacy et al., 2006, 2012; Zupancic and Trcek, 2017]. Agents with different dispositions may recommend agents they genuinely believe are trustworthy who another agent does not. This is not malicious, however BLADE focuses on detecting deceptive recommendations where agents are knowingly spreading false information. The endogenous approach to bias identifies how statistically different a recommender's opinions are to the aggregated reputation from all witness providers [Whitby et al., 2004]. This approach does not fairly treat witnesses who have a perceptual bias and are not maliciously providing different reports.

Burnett *et al.* also identify two types of bias that can exist when using stereotype models, perceptive and behavioural. They describe how agents can be biased towards or against agents from the same group or another, respectively. Perceptual bias describes how agents can perceive interaction outcomes differently, and behavioural bias describes how agents behave differently depending on who their interaction partner is [Burnett et al., 2013]. Similarly, in tag-based cooperation, agents might vary their level of cooperation depending on who they are interacting with [Howley and O'Riordan, 2005].

2.4.2 Sliding Windows and Forgetting Factors

A sliding window, W, of size ω retains the most recent ω experiences. The intuition is that the most recent ω instances capture agents' current behaviour, and older interaction records which are no longer relevant are filtered out. For

a new interaction outcome, o^t, at time t, the window is updated as:

$$W \leftarrow (W \backslash W[\omega]) \cup o^t \tag{2.1}$$

A forgetting factor, also referred to as a longevity factor, similarly forgets at a constant rate with all instances being retained and more recent interactions weighted higher. An interaction from time t at the current time, t', which can be used in trust assessment is weighted as:

$$o_{t,t'} = \lambda^{t'-t} \times o_t \tag{2.2}$$

Models using forgetting factors include BRS [Jøsang and Ismail, 2002], FIRE [Huynh and Jennings, 2004] and REGRET [Sabater and Sierra, 2001a]. Forgetting factors, and methods which use forgetting factors such as WoLF [Bowling and Veloso, 2001], are not appropriate for the work presented in this thesis because they maintain Markov states, and not a granular history. This does not give us the flexibility to delete past instances identified as unrepresentative.

Sliding windows and forgetting factors require the user to select the size of the window or rate of forgetting respectively. The best values will depend on the context but in many scenarios the best values might be different at different times, or if multiple concepts exist then these can require different values at the same point in time. Therefore, these approaches, which give agents one global approach to handling the interaction data for all their previous partners are too general. A more successful agent should be capable of detecting when data are irrelevant. Stereotype models can be incredibly accurate when the concepts being learned remain static however, in this thesis that assumption is removed.

The POYRAZ model allows an agent to specify a window size over time instead of the number of interactions, and this amount of time can be specified for each partner agent [Şensoy et al., 2009]. If the agent provides many transactions then the time scale span is smaller, so that only the most recent interactions are included, with the aim of keeping up with their current behaviour pattern. For those agents who provide few interactions, the time span is larger. The limitations of this approach include that even if the agent is providing many interactions, a smaller time span could encapsulate a high proportion of noise. Potentially useful information is excluded which could improve the accuracy of trust assessment. For those agents with fewer interactions over a long time period, the behaviour may have changed in that time and it may be more advisable to use fewer instances for the calculations than to include data which has become unrepresentative of their current behaviour.

Similarly to a forgetting factor, the Win or Learn Fast (WoLF) reinforcement

learning algorithm, an extension to Q-learning [Wakins, 1992], can adapt to dynamic environments based on the accuracy of the current prediction model [Bowling and Veloso, 2001]. While WoLF is predicting accurately, the agent uses a slow forgetting factor, thus retaining a lot of historical data to be as accurate as possible. When the predictions incur high error, the agent switches to a higher forgetting factor to only use more recent data. WoLF can be an appropriate algorithm if there is a strict memory bound, as it only maintains the current expectation by summarising historical data, but not retaining the historical data itself. However, the disadvantage of this is firstly it cannot pinpoint when the time change occurred because all of the data is not held in memory, thus preventing an ability to feed this information to a trust system. Secondly, having two forgetting factors is not as flexible as an adaptive window, which can retain any amount of data depending on what is deemed relevant. Therefore we investigate whether *concept drift* models are a more appropriate tool for managing the memory requirements of agents detecting behaviour changes. With explicit drift detection we can learn more about the changing concepts.

Some trust and reputation models investigate whether a good reputation should be earned over a long time but can be quickly lost to protect against negative behaviour change [Liang and Shi, 2005; Srivasta et al., 2005]. Specifically, TrustGuard has an averaging component to prevent one off fluctuations from damaging the overall trust value. The reputation has three major components: current reputation which is based on very recent experiences, a reputation history which uses older experiences, and a reputation fluctuation over the recent experiences. A sliding window is still used for the reputation history aspect and it is suggested it should be as large as possible without too much hindrance to computational efficiency. Different weightings are assigned to each of these components, however these parameters are inputs to the system, and therefore need to be known, or guessed, in advance. Equally, the values set may not be appropriate at all time points in the system, i.e., the first half of time might see more volatile agent behaviour than the second, thus different weights would be more appropriate at different time points.

The work from Ruohomaa *et al.* specifically focuses on how to react to anomalous dynamic behaviour to improve reputation assessment by first identifying epochs of consistent behaviour [Ruohomaa and Kutvonen, 2010; Ruohomaa et al., 2011]. Similarly to methods which allow reputation to be built slowly and lost quickly, this approach assigns a higher weighting to negative interaction experiences. This work chooses to compress an agent's history of past interactions into aggregates of different possible outcomes to save memory. This means an interaction outcome is not stored as its own item with a time step, and so time-based analysis cannot be applied, which makes it distinct from

the work presented in this thesis.

2.4.3 Concept Drift

Many current state-of-the-art trust and stereotype models draw upon machine learning techniques to analyse past behaviour of agents to predict their future performance. Machine learning models identify correlations between a feature set X and a target variable Y. A concept is the joint probability $P(XY)$, and therefore, machine learning models relying on the assumption that this joint probability of the features and the target remains static over time i.e. $P_t(XY) = P_{t'}(XY)$ for all times t and t' [Tsymbal, 2004; Webb et al., 2016].

Concept drift can be either real or virtual. Real concept drift occurs when there is a change in the target concept, meaning $P_t(Y|X) \neq P_{t'}(Y|X)$. For example, in an online market place when sellers who stock a particular product are all of a high quality at time t, but due to a change in supplier, the same product has a low quality at time t'. The agents' features have remained the same (stocking that particular product), but their behaviour has changed. Virtual concept drift occurs when the prior probability distribution of the features changes i.e. $P_t(X) \neq P_{t'}(X)$. This represents the proportion of agents with certain features changing i.e. more sellers begin to sell a certain product than before but stop selling another. Virtual concept drift is out of the scope of this thesis, and from here on we refer to real concept drift as just concept drift.

A concept can change gradually or suddenly, where sudden drift may also be known as *concept shift*. If the relationship between a set of features and a class instantly disappears in favour of some other concept there is sudden drift, while a slow progression of one concept becoming less prevalent while another begins to dominate is known as gradual drift. Gradual drift is considered harder to detect, especially when the differences in the two concepts are not empirically large but are important [Hoens et al., 2012].

A recurring concept is one which disappears in favour of another for a time but reappears later. For example, when predicting the weather you might consider winter a recurring concept. One approach to improve prediction is to maintain a concept history whereby a newly learned concept is saved and can be reused if it identified again rather than relearning the concept from scratch [Wu and Zhu, 2005]. RePro is one such algorithm which does this, and it records the transition pattern between concepts. RePro accurately predicts the next concept based on the current concept, known as the *Pro*active mode, but if a change in concept occurs and none of the previously identified concepts is appropriate then a new concept is learned from incoming instances, known as the *Re*active mode. The RePro algorithm can use a mixture of these approaches to

use the mode which best suits the current circumstance.

A concept drift system can be viewed on a high level as having four components: memory, learning, loss estimation and change detection [Gama et al., 2014]. The loss estimation and learning components are not considered in this thesis because we focus using trust and stereotype techniques for predicting agent behaviour which are more suited to this context. We focus on integrating change detection and memory management to select relevant data as input to trust and reputation algorithms.

The memory component decides which data to retain for predictions. One approach is a variable window which adapts its size to encapsulate the amount of data considered relevant. The window shrinks when drift is detected, and expands with new instances while they are believed to all be from the same distribution. In Chapter 5 we demonstrate how the Adaptive Windowing (AdWin) model can be applied in MAS [Bifet and Gavaldà, 2007]. The AdWin algorithm is proven to perform well for both sudden and gradual drift.

The change detection module attempts to identify a point or range of time where concept drift has occurred. This is how the memory component knows which data to remove. Gama *et al.* [Gama et al., 2014] identify four general techniques for this component: sequential analysis, control charts, monitoring two distributions, and contextual. We focus on monitoring two distributions, as this has the advantage of identifying the point in time where the change occurred, unlike sequential analysis which is very similar but does not store all the data points and therefore cannot pinpoint the exact time of change. Models which do not pinpoint the time of change but can feed their results into the learning module are not applicable because we are substituting the learning component for trust and reputation algorithms. The time of change must be made available to the agent to integrate updating its data as it is formatted by the trust and reputation system.

One technique for change detection is to divide the data into subsets, estimate the distribution which generated each subset and determine if they are distinct enough by disproving the null hypothesis that the data were generated from the same distribution. When using this approach, we must consider whether the data are continuous or binned, and are we comparing one distribution to a known distribution, or two unknown distributions [Press et al., 2007]. If the class variable is cateogrical, comparing them might use the chi-square test, but for continuous data over a single variable, the Kolmogorov-Smirnov (K-S) test can be more appropriate which uses cumulative distribution functions. The exact test used can be substituted, and in this case, we could use a binning method on agnet trust values to make it categorical and then substitute the K-S test for the chi-square test. However, this minor difference is not the focus

of this work and we select the K-S test to use on the continuous data.

An issue facing concept drift models is *class imbalance*, where of all the true class values which exist in the dataset, the samples used for the learning algorithm are dominated by only one class label [Chawla, 2009]. This problem is relevant to our work because the agents using the trust and reputation model will choose the next agent to interact with based on the model, so they will bias their choice of interaction partner towards agents from the stereotype they currently believe to be the best. Most existing work to combat the class imbalance problem uses ensemble methods [Chen et al., 2010; Gao et al., 2008; Lichtenwalter and Chawla, 2009]. Certain sampling methods can also improve the performance, however this is not a viable approach here as this would require the agents to have a fixed exploration strategy, thus reducing their performance on average. Exploration can be a viable strategy in a static environment as the agents only need to explore initially and the long term benefits of the information gained can be reaped without any more exploration [Player and Griffiths, 2017b]. However, if the concepts are continually changing then exploration would need to occur continuously.

CHAPTER 3
Agent Environment

This chapter formally describes the multi-agent system we apply our work to. We include details about the specific evaluation setup that procured the results we present in this thesis. One of the major themes in this thesis is dynamic agent behaviour, therefore, in this chapter we introduce new notation to describe how agent behaviour can change over time. We describe a general environment to demonstrate that the work is applicable across different contexts. Our environment is comparable to existing literature in the field of trust, reputation and stereotypes.

3.1 Agents

A set of agents is divided into disjoint subsets of trustors, $\mathcal{A}_{tr}$, and trustees, $\mathcal{A}_{te}$. Trustor agents assess available trustees and choose the trustee assessed to have the maximum trust value as an interaction partner.

Agents can be described by their ID and *observable features*, denoted as a vector, $\overrightarrow{\tau_j}$, for agent j. Each feature corresponds to a characteristic of an agent, for example the technical specifications of a device in a network, or the brand, logo, location, items and prices of a seller in an online marketplace. Some features correlate with behaviour, known as *relevant features* but some features are observable and do not impact on behaviour, called *noisy features*, also known as irrelevant features. We assign the relevant features of an agent depending which profile they belong to (as all agents of one profile have the same relevant features and behaviour), while noisy features are assigned at random to each agent. This represents a realistic assumption that not all observable characteristics of an agent affect, or are indicative of, their behaviour. Agents do not know in advance which features correlate with behaviour. The number of relevant and noisy features each agent has, n_{rf} and n_{nf} respectively, are experimental parameters.

3.2 Behaviour

Each agent has a behaviour value, $bhv \in [0, 1]$, symbolising how objectively good they are at executing a task. When an agent has an interaction, the interaction outcome the trustor receives is a binary value of good with a probability bhv, or bad otherwise. A higher value of behaviour implies a higher chance of a good

interaction outcome with that agent. The behaviour of an agent represents how consistently they can perform tasks to a standard and time frame that is acceptable to the trustor. Research exists on the subjective value of interaction outcomes, however here we assume it is objective.

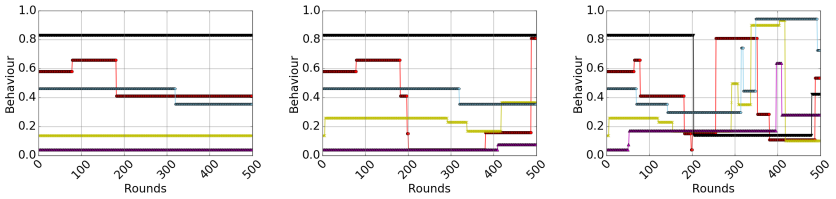
Existing trust and stereotype literature is evaluated with static agent behaviours implying $\forall_{t,t' \in T} bhv^t = bhv^{t'}$ where T is the set of timesteps. In this section, we define how behaviour changes over time using the variables p_{Gr} and p_{Su} to represent behaviour can change at a gradual and sudden rate, respectively. Behaviour is static when p_{Gr} and p_{Su} are both set to 0. A profile might have a current behaviour value which will last for a random amount of time in the range [0,100] timesteps. When that behaviour ends, a new value is chosen and the behaviour is modified by the amount p_{Gr} each timestep until it has reached the new value. The behaviour can spontaneously change to a new value with chance p_{Su}. In Chapter 6, dynamic behaviour can also be patterned. In this instance, a maximum of 5 behaviour values are randomly generated in advance and cycled through in the same process described above. When the last behaviour is over, the first behaviour is transitioned to and the behaviours are repeated.

A visualisation of varying degrees of dynamic behaviour, achieved by increasing the values of p_{Gr} and p_{Su}, can be seen in Figure 3.1. We also can see from these images that by the time p_{Gr} and p_{Su} reach 0.01, they are quite extreme, hence why our experiments vary dynamic behaviour with these two variables within this range.

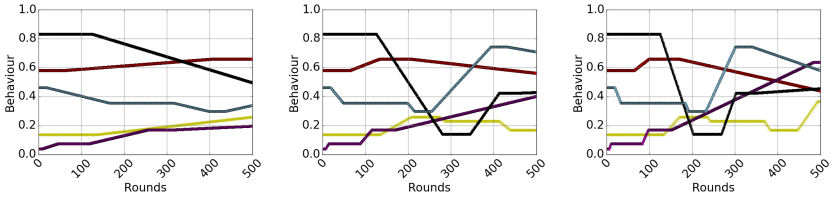
3.3 Profiles

A profile represents a type of agent, where the agents share observable characteristics and behaviours for example, a device model in a mobile ad hoc network. Agents might not always be able to identify exactly which profile an agent belongs to. However agents can use observable features to estimate those profiles, known as stereotyping.

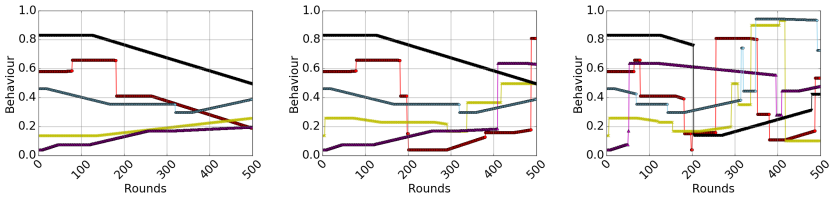
Each profile defines a set of relevant observable features and behaviour that each agent of that profile has. We use numerical values to abstract these feature values. Agents also have observable irrelevant features which do not correlate with behaviour and it is part of the learning task to identify these. Unlike existing work, we relax the assumption that the relevant observable features are exactly the same for all agents of a profile, as there could be a range of values or a distribution over possible values. We define a feature f as $f : \langle \mu_f, \Theta_f \rangle$, where $\mu_f > 0$ is the mean value of the feature and $\Theta_f <= \Theta$ is the standard deviation of the feature value, which is drawn randomly with a maximum value



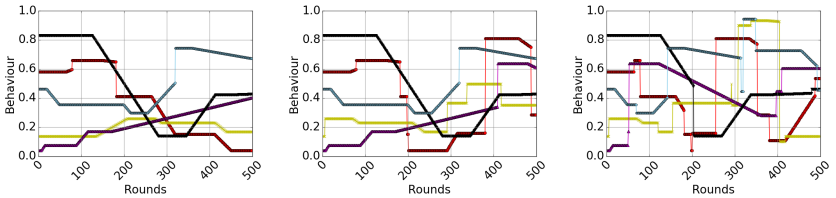
(a) $p_{Su} = 0.001$ (b) $p_{Su} = 0.005$ (c) $p_{Su} = 0.01$



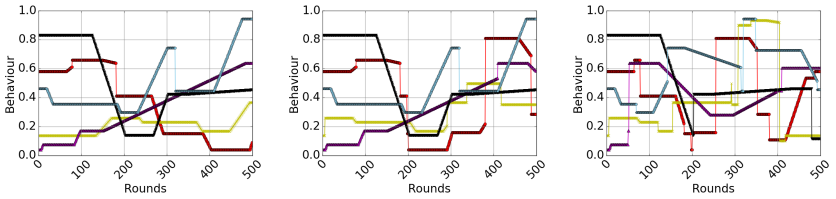
(d) $p_{Gr} = 0.001$ (e) $p_{Gr} = 0.005$ (f) $p_{Gr} = 0.01$



(g) $p_{Su} = 0.001$, $p_{Gr} = 0.001$ (h) $p_{Su} = 0.001$, $p_{Gr} = 0.005$ (i) $p_{Su} = 0.001$, $p_{Gr} = 0.01$



(j) $p_{Su} = 0.005$, $p_{Gr} = 0.001$ (k) $p_{Su} = 0.005$, $p_{Gr} = 0.005$ (l) $p_{Su} = 0.005$, $p_{Gr} = 0.01$



(m) $p_{Su} = 0.01$, $p_{Gr} = 0.001$ (n) $p_{Su} = 0.01$, $p_{Gr} = 0.005$ (o) $p_{Su} = 0.01$, $p_{Gr} = 0.01$

Figure 3.1: Agent Profiles with Dynamic Behaviour

of Θ. Features become noisier as Θ increases, representing that feature values can include a range of values that becomes harder to correlate with behaviour as that range increases. This parameter can also represent an agent's inability

to perceive feature values perfectly. Most of our evaluations fix Θ at 0.2, but we also include an evaluation of increasing this noise. Profiles' relevant features and behaviours are generated randomly for each experiment.

For example, a profile, p, of 5 relevant features and a behaviour value, may be defined as follows:

$$p : \langle f_0 : \langle 2, 0.1 \rangle, f_1 : \langle 0, 0.13 \rangle, f_2 : \langle 3, 0.03 \rangle, f_3 : \langle 1, 0.16 \rangle, f_4 : \langle 5, 0.15 \rangle \rangle, bhv = 0.8$$

An agent, j, belonging to this profile, and who also has 2 irrelevant features generated randomly, might have the following observable feature vector:

$$\overrightarrow{\tau_j} : \langle 2.05, 0.1, 2.97, 1.17, 4.87, 4.1, 0.6 \rangle$$

Each agent from this profile will have values for their observable features drawn from the respective Gaussian distribution defining that feature for the profile, and they will all have a behaviour value of 0.8. As described in Section 3.2, this behaviour is a mean value for a Gaussian distribution for which interaction outcomes are pulled from when this agent is selected as a partner.

Our evaluations in this thesis use numerical features in the range $[0, 5]$ however, using categorical feature values is also possible if a distance metric is defined for them [Player and Griffiths, 2017a].

Throughout this thesis, we demonstrate how agents need to select relevant data about agents to make accurate assessments. This often requires identifying a group of agents, G, whom we believe belong to the same profile. Formally, an agent with an ID and observable features, $\langle ID, \overrightarrow{\tau} \rangle$ needs to be associated with a group. In Chapters 4 and 5, the set of groups $\overrightarrow{G}$ is identified using a stereotype model $S(\overrightarrow{\tau}) \rightarrow \overrightarrow{G}$, and so an agent's group is $S(\overrightarrow{\tau}) \rightarrow G$. In Chapter 6, agents can immediately be identified to their group, $f(ID) \rightarrow G$.

Our definition of groups is similar to that from StereoTrust. An agent uses a grouping function to determine if another agent belongs to a particular group or not. Depending on the context, the agent could belong to multiple groups, weighted by probabilities etc. In the simple case, agents have a probability of one of being in a group and a probability of 0 of being in any other group. The grouping function is interchangeable, for example we use a stereotype model in Chapters 4 and 5. However, for Chapter 6, it is considered out of scope and we have a simple mapping of agents to groups.

3.4 Interactions

An interaction between two agents can represent an exchange of goods or services between provider and consumer. A trustor agent, tr, makes a trust assessment about a trustee agent, te, and if they are the most trustworthy available agent, the trustor selects te for the interaction. Agents interact over time steps and record their experiences as a set, $\mathcal{O}_{tr}$, of tuples of the form:

$$\langle te, \overrightarrow{\tau}, r, s, t \rangle$$

where te is the trustee's ID, $\overrightarrow{\tau}$ is their observable features, r and s are the cumulative sum of good and bad interactions with te including the outcome from this interaction, and t is the time of the interaction. Stereotype models use only $\overrightarrow{\tau}$ and a value of $trust$ calculated from r and s in each tuple while trust algorithms use only te and r and s. There is a new tuple for each interaction, a tuple is not updated for each te because we analyse their behaviour over time.

If an old interaction tuple is deleted from time t' (because it is old or deemed no longer representative of current behaviour), any tuple with agent te from before that time is deleted, and any tuple from a time afterwards is updated:

$$\langle te, \overrightarrow{\tau}, r, s, t \rangle \leftarrow \langle te, \overrightarrow{\tau}, r - r^{t'}, s - s^{t'}, t \rangle \tag{3.1}$$

Measuring the success of the agents is the average utility each agent receives over a specific time period T:

$$avg\_utility = \frac{\sum_{t \in T} \sum_{a \in \mathcal{A}_{tr}} o^t_{tr,te}}{n} \tag{3.2}$$

where n is the total number of interactions in that time period, $o^t_{tr,te}$ is the outcome of the interaction between tr and te at time t. The results we present use either $T = 1$, to show the average outcome per agent at a particular timestep, or where T is the total number of rounds, to show the average utility per agent per timestep. We present the 'optimal' results throughout this thesis, which describe the best average utility the agents could attain collectively if they selected the best available partners.

The accuracy of agents' assessments is measured using the Root Mean Squared Error (RMSE), calculated as:

$$RMSE = \sqrt{\frac{\sum_{a \in \mathcal{A}} (trust_{tr,te} - bhv^t_{te})^2}{n}} \tag{3.3}$$

where, $trust_{tr,te}$ is the assessed trust tr has in te and bhv^t_{te} is the trustee's true behaviour at time t.

3.5 Population

As introduced in existing work [Burnett et al., 2010], to represent a dynamic population and the idea that trustors continually face new trustees to evaluate, each trustee has a probability, $p^{te}_{leave} \in p_{leave}$, of leaving the population each round and being replaced by a new agent from a random profile. Trustors do not leave the population because we monitor their ability to collect data, and adapt their histories to the changing available partners. In line with existing work, we evaluate our work with a default value of 0.1 for p_{leave}, but demonstrate the effect of varying this parameter as part of our evaluations.

Agents are connected in a topology such that they can only assess a neighbouring agent. We present results for fully connected, small world and scale free graph topologies. These graph types exhibit different characteristics from real world networks, which can affect an agent's abilities to gather information, distribute reputation and make accurate behaviour assessments. A small world network describes a graph where nodes can reach any other node with very few number of hops because of the high level of connectivity [Watts and Strogatz, 1998]. A scale free network describes graphs where nodes display the "preferential attachment" property, meaning nodes are likely to connect to other nodes which already have a high degree [Barabási and Albert, 1999]. This results in a network where there are a large number of nodes with a very low degree, and exponentially decreasing nodes have a very high degree. Most the results we present in this thesis are with a fully connected graph, to represent the general case where agents have a wide variety of partner choice, and might represent a subgraph of a larger real world network.

The evaluations in this thesis use a fully connected graph with 100 agents. However, we present some results in other topologies to investigate the efficacy of our work in networks which exhibit characteristics of a variety of real world networks.

The network topology affects agents' neighbourhood sizes, limiting the number of available partners. Table 3.1 shows the mean degree of an agent (i.e., its neighbourhood size) in the respective network topologies[1] and the standard deviation of that mean across all the agents in the network. The standard deviation of the degree in a small-world network is considerably lower than in a scale free network, so almost all the agents in a small-world network have a selection of 5 to 7 agents to partner with and collect witness reports from. In a scale-free network, most agents have only one or two possible partners, while a small minority of agents will have an extremely high degree.

[1]Small-world topologies are defined as $n \times n$ lattices in the implementation and therefore require a square number of agents.

Table 3.1: Network topology mean degree and standard deviation.

	Total number of agents in network, $\|\mathcal{A}\|$	Mean Neighbourhood Size (Agent degree)	Standard Deviation
Fully Connected	20	19.0	0.0
	40	39.0	0.0
	100	99.0	0.0
Small World	100	6.0	1.00995
	196	5.98980	0.92024
	484	6.0	0.99378
Scale Free	100	5.86	4.33594
	200	5.93	5.4877
	500	5.9720	5.86372

3.6 Trust and Stereotype Evaluation Models

Our work can be applied to any trust and reputation model which takes in past interactions, $TR(\mathcal{O})$, and outputs an expectation of behaviour, which may need to be normalised in the range $[0, 1]$. Similarly, a stereotype model which outputs a description of identified stereotypes based on the data, $\overrightarrow{S} \leftarrow S(\mathcal{O})$ can be bootstrapped by our work. Therefore any method for TR and S can be used. In this thesis, we choose to evaluate with the trust models BRS and DRS [Jøsang and Ismail, 2002; Jøsang and Haller, 2007], and the stereotype model from Burnett *et al.* [Burnett et al., 2010].

When using BRS, interaction outcomes are perceived as binary, for example, the trustor rates an interaction with a trustee as either good or bad. If a higher granularity of interaction outcome is available, the more generalised version of BRS, Dirichelet Reputation System (DRS), can be applied. BRS is the specific version of DRS where the distribution parameter k is 2, however in DRS k can be any integer greater than 1. We present results for other values of k when using DRS throughout this thesis.

Reputation systems aggregate reports about a trustee from any available trustors, known as opinion providers, *op*, or witnesses. The number of good and bad interactions with te, r_{te} and s_{te}, becomes:

$$r_{te} = \sum_{op}^{\mathcal{A}_{tr}} r_{te}^{op}, \quad s_{te} = \sum_{op}^{\mathcal{A}_{tr}} s_{te}^{op} \tag{3.4}$$

The number of good and bad interactions are the input parameters to a beta probability density function, whose expected value is the estimate of the trustee's behaviour, $trust(te)$. An agent can aggregate witness reports from available agents to improve the accuracy of the assessment. For now, we assume

that these values are relevant data about the current behaviour of te. The *belief*, b_{te}, in te is their expected behaviour based solely on their previous interactions, as calculated below:

$$b_{te} = \frac{r_{te} + 1}{r_{te} + s_{te} + 2} \tag{3.5}$$

BRS weights older interactions less according to a forgetting factor. To account for uncertainty in this belief, an *a priori*, a, is weighted by an uncertainty factor, u_{te}, which decreases as more interaction experiences are collected, signifying a higher confidence in the belief factor.

$$u_{te} = \frac{2}{r_{te} + s_{te} + 2} \tag{3.6}$$

The default value for the *a priori*, a, is either 0.5, which assumes that an agent is neither good nor bad, or if a stereotype model is available to provide an alternative assessment:

$$a_{te} = \begin{cases} stereotype(\overrightarrow{\tau_{te}}), & \text{if using stereotypes} \\ 0.5, & \text{otherwise} \end{cases} \tag{3.7}$$

The decision as to whether to use stereotypes is usually dependent on whether enough data are available to build the stereotype model, and then the default is never returned to. However, in this thesis we present models which show whether to use the stereotype model should depend on other circumstances.

The overall expected behaviour according to BRS, $trust(te)$, is calculated using subjective logic, namely:

$$trust(te) = b_{te} + u_{te} \times a_{te} \tag{3.8}$$

DRS is a generalisation of BRS, as it divides interaction outcomes into k possible values. While BRS is a prominent, mathematically rigorous model, the limitation of only two possible outcomes from an interaction can be unrealistic. For example, if $k = 5$, interactions can fall into one of the following categories: [1-bad, 2-mediocre, 3-average, 4-good or 5-excellent] [Jøsang and Haller, 2007]. When tr calculates the reputation of te at time t using DRS, the set of tr's past interactions with te, $\mathcal{O}_{tr}^{te} \subset \mathcal{O}_{tr}$, are each labelled with one of k values to become the input parameters to a multivariate Dirichlet PDF. From a Dirichlet distribution, a reputation value can be calculated using point estimate representation, which is easily computable and human understandable. Generalising storing interaction outcomes from BRS, agents have an evidence vector, $\overrightarrow{R}_{te} = (R_{te}(i)|i = 1...k)$, where $R_{te}(i)$ indicates the number of interactions from

$\mathcal{O}_{tr}^{te}$ that were rated in the i^{th} category, and the first category is $i = 1$. Each category i is given a point value, $v(i) = \frac{i-1}{k-1}$, such that the values are evenly distributed in the range $[0, 1]$. An overall reputation value is a weighted average calculated as in Equation 3.9.

$$reputation(te) = \sum_{i=1}^{k} v(i)S(i) \tag{3.9}$$

where the vector $\overrightarrow{S}$ is the expectation of each category considering both the evidence and an *a priori*, calculated in Equation 3.10.

$$\overrightarrow{S}_{te} : (S_{te}(i) = \frac{R_{te}(i) + Ca(i)}{C + \sum_{j=1}^{k} R_{te}(j)}; |i = 1, ..., k) \tag{3.10}$$

Each element, $S_{te}(i)$, in the vector $\overrightarrow{S}_{te}$ describes the multinomial probability reputation of the i^{th} element given the aggregate reports in $\overrightarrow{R}_{te}$, a constant to weight the *a priori*, C, and a vector of *a priori* probabilities for each category. Higher values of C will raise the dependence of the *a priori* in the final reputation of te, and we use the recommended value $C = 2$ [Jøsang and Haller, 2007]. The vector of *a priori* probabilities, $\overrightarrow{a}$, is a list of the prior probabilities of each i^{th} category, subject to the constraints $\sum_{i=1}^{k} a(i) = 1$ and $a(1), ..., a(k) > 0$. In this context, agents have an *a priori* estimate of another agent which is a single point value, which needs to be translated into this vector of length k for DRS. We define a function, $apriori(x) : Gaussian(a_{te}, \frac{1}{k})$. We can then define $\overrightarrow{a}_{te}$ as:

$$\overrightarrow{a}_{te} : (a_{te}(i) = apriori(\frac{i-1}{k-1}); |i = 1, ...k) \tag{3.11}$$

Each value in the vector $\overrightarrow{a}_{te}$ is normalised such that $\sum \overrightarrow{a}_{te} = 1$, as this is a requirement of DRS. The initial point value of the *a priori* depends on whether the agent can use a stereotype model, or is using the default value as seen in Equation 3.12.

$$a(i) = \begin{cases} a(i) \in \overrightarrow{a} \text{ if } a \textit{ priori} \text{ model available} \\ \frac{1}{k} \text{ otherwise} \end{cases} \tag{3.12}$$

The stereotype model we use is an M5 decision tree algorithm proposed by Burnett *et al.* , where nodes discriminate between features with high entropy and the leaves have linear regression models to learn numerical values for trust. This distinguishes M5 from other decision tree algorithms which are limited to discrete outputs[2]. The tree is built, or rebuilt, after a learning interval L. The

[2]Though we modify this algorithm in some cases, described where appropriate

data collected in that time trains the new tree. The intuition is that the tree is then up to date with current behaviour. The tree is not rebuilt every time step because this is too computationally expensive.

3.7 Summary

A summary of the environmental parameters and their definitions are presented in Table 3.2. Throughout this thesis, our evaluations show different values for these variables.

Table 3.2: Environmental parameters.

Parameter	Definition
$\|\mathcal{A}\|$	Number of agents in the graph
$n_{profiles}$	Number of profiles agents can belong to
n_{rf}, n_{nf}	Number of relevant and irrelevant features, respectively
graph	The graph topology agents are connected in
p_{Gr}	Speed of gradual change
p_{Su}	Probability of sudden change
k	The number of bins that an interaction outcome fall in towards calculating trust in DRS
L	Number of new interaction records required to (re)build the stereotype model

CHAPTER 4
Information Assessment

In this chapter, we highlight the importance of using representative data in trust and stereotype assessment, and how to handle situations where they do not exist. This can occur when agent behaviours are static or dynamic. The focus of this chapter is on the causes of unrepresentative data when agent behaviours are static, and we further investigate the effects of dynamic agent behaviour in Chapters 5 and 6. This chapter addresses ROs 1 and 2, regarding agents having a lack of data for estimating the behaviour of some or all agents. We present two techniques, Class Imbalance Modification (CIM) and Direct Comparative Stereotypes (DCS). The former identifies whether past interaction experiences are representative of a stereotype and therefore whether a stereotype model can accurately assess an agent belonging to it. The latter is a stereotype technique that does not rely on past interaction experiences and so can be used where such data is inappropriate or unavailable.

4.1 Introduction

Trust and reputation algorithms are typically able to identify when the necessary data (experience data with a target agent) are not dependable by using the quantity of data to estimate a level of confidence, or uncertainty, in the trust calculation. If appropriate, a default or a stereotype assessment can either be solely used, or weighted more highly, depending on the level of uncertainty. Without considering how small amounts of data could affect trust assessment, trustors would struggle to overcome initial bad opinions based on an unfair sample of data. A similar problem exists in stereotype models but it is not addressed in existing work. Past interaction data, which a trustor uses to build a stereotype model, might not represent every type of agent that is assessed using that stereotype model. This *class imbalance* in the history of past interactions is caused by the realistic assumption that agents are rational and therefore make decisions based on their local knowledge to select who they believe are the most trustworthy partners, leading to maximise their own reward. However, once an agent has interacted with another that they know to be more trustworthy than the default trust value ascribed to unknown agents, they continue to interact with them and never expand their knowledge. This prevents agents from accurately learning a representative range of stereotypes, and ultimately from discovering more trustworthy partners.

Another cause for a lack of representative data is dynamic behaviour, because when an agent changes their behaviour their past performance is no longer indicative of current behaviour. Few existing models consider the impact of dynamic behaviour, and those that do use techniques such as sliding windows and forgetting factors, whose effectiveness is limited in MAS as discussed in Chapter 2.4.2. While dynamic behaviour is not the focus of this chapter, we include a brief evaluation of CIM and DCS when agent behaviours are dynamic towards the end of this chapter.

In this chapter, we present the Class Imbalance Modification (CIM) method to complement stereotype models, which considers whether a trustor's history represents the agent being assessed, described in Section 4.2. Second, we present a data-free stereotype technique, Direct Comparative Stereotypes (DCS), which can be used if CIM deems a data-driven stereotype technique inappropriate, described in Section 4.3. DCS is inspired by tag-based cooperation where agents believe that agents who look like themselves will behave similarly [Riolo et al., 2001], and aims to provide an initial estimate of behaviour. Our results show that CIM and DCS offer significant improvements to trust assessment of partners for whom there are insufficient representative past experience data.

4.2 Class Imbalance Modification (CIM)

CIM uses C-DenStream, a semi-supervised, density based, online clustering algorithm [Ruiz et al., 2009]. A density based clustering algorithm uncovers groupings in the data without knowing how many groups may exist in advance, or given an agent may only encounter a subset of those groups. Stereotype models use fully labelled datasets regardless of the certainty in those trust labels. However, CIM uses a semi-supervised clusterer which only labels instances where the agent is highly confident in trust value, meaning that it was based on multiple repeat interactions, making it more likely to be accurate. Finally, CIM updates online, so it is efficient and can exploit new information as soon as it becomes available. Stereotype models, which typically use machine learning algorithms, have to be learnt in intervals because of their time complexity to update. This is one reason they are vulnerable when agent behaviours are dynamic, as the model producing *a priori* assessments was built on data representing old behaviour and is not updating with new information until it is too late. We show how CIM can integrate with a stereotype model, S, that is built every L time steps.

An agent records an interaction as a tuple: $\langle te, \overrightarrow{\tau}, o, t \rangle$, where $\overrightarrow{\tau}$ is the observed features of te, o is the interaction outcome and $t \in T$ is the timestep out of T total time steps. Recall that all tuples are stored in an agent's history

of interaction outcomes, $\mathcal{O}_t r$. Trust in te is assessed before an interaction using the trust algorithm. Throughout this thesis we use the trust algorithm DRS, described in Chapter 3. DRS accepts interaction outcomes which are categorised into one of k bins. In this chapter, we use $k = 2$, representing the prominent BRS trust model. Only the observable features and trust are used to train the stereotype model, so the data are reduced to: $\langle \overrightarrow{\tau}, trust \rangle$. The stereotype model is fully supervised so retains the trust label for all instances of the data. However, CIM is semi-supervised, so only keeps the trust label for an interaction record if the uncertainty, u_{te}, calculated in Equation 3.6, is below the threshold υ. We demonstrate in Section 4.4 that agent performance is not sensitive to the value for the threshold υ, showing that CIM is robust to any amount of labelled data.

An overview of how trustors use behaviour assessment algorithms as part of the whole experiment is given in Algorithm 1. Trustors choose the partner they believe will behave the best in an interaction. Assessing an agent's behaviour with trust, stereotypes and CIM is further described in Algorithm 3. Line 5 describes the interaction experience which was discussed in Section 3.4. When the interaction is over, the methods used for behaviour assessment are updated with the interaction outcome, and this process is described in Algorithm 2.

Algorithm 1 Agent Simulation Process.

```
1: function SIMULATION
2:     for t ∈ T do
3:         for tr ∈ A_tr do
4:             partner ← te|max_{te∈A_te} Behaviour_Assessment(te, O_tr)
5:             o^t ← Interaction(partner)      ▷ depends on partner's behaviour
6:             Update(partner, o^t)
7:         end for
8:     end for
9: end function
```

We first describe how the behaviour assessment methods are built and updated (Algorithm 2) before describing how they are used (Algorithm 3). First, the outcome interaction is added to the trustor's history of interactions, $\mathcal{O}_{tr}$. These data are used to train the stereotype model, S, every L interactions, as seen in Line 5 of Algorithm 2, and later for trust assessment in Algorithm 3. S are rebuilt after a predefined time interval, L, because it is too computationally complex to relearn them with every new interaction outcome. Over time, the stereotype model lags behind representing the agent's full set of data up till that point until it rebuilds S again. For this reason, we maintain two sets of clusters, one which represents the data at time S was built, C_s, and one which represents the dataset at time t, C_t. If a trustee's features are not associated

with a cluster in C_s, then they are not represented by the stereotype model. As the clusterer is online, it is easy to continuously update the clusters to produce C_t. We discuss how CIM uses C_t to account for dynamic behaviour below.

Algorithm 2 Updating CIM, stereotype and trust models after an interaction at time t.

1: **function** UPDATE($partner$, o^t)
2: $\mathcal{O}_{tr} \cup \langle partner, \overrightarrow{\tau}_{partner}, o^t, t \rangle$ ▷ update agent's history of interactions
3: $C_t \leftarrow$ Update_Clusters($partner, o^t, t$) ▷ Most up to date clustering of $\mathcal{O}_{tr}$
4: **if** t mod L **then**
5: $S \leftarrow Build\_Stereotype(\mathcal{O}_{tr})$
6: $C_s \leftarrow C_t \leftarrow Initialise\_Clusters(data)$
7: **end if**
8: **end function**
9: **function** $Initialise\_Clusters$(data)
10: MC_p, $MC_o \leftarrow$ DBSCAN(data)
11: **return** MC_p
12: **end function**
13: **function** $Update\_Clusters(partner, o^t, t)$
14: $MC_p, MC_o \leftarrow CDenStream(MC_p, MC_o, \overrightarrow{\tau}_{partner}, o^t, t)$
15: **return** MC_p
16: **end function**

C-DenStream summarises raw data into *micro-clusters*. At first, the micro-clusters have to be initialised by clustering a bulk amount of data with the DBSCAN algorithm [Ester et al., 1996], and then the micro-clusters can be maintained online by assigning a new data point to a micro-cluster and updating the weight associated with that micro-cluster. The initial micro-clusters are built at the same time S is built, shown on lines 5 and 6[1]. Specifically, micro-clusters are circular points summarising an area of raw data points (given some parameters). Once the initial micro-clusters are built, they can be iteratively updated with each new data point, described in more detail below. A micro-cluster has a weight which is increased as new instances are added to it, and fades over time. If the weight of a micro-cluster is above a threshold it is a potential-micro-cluster *p-mc*$\in MC_p$, else it is an outlier-micro-cluster, *o-mc*$\in MC_o$. An *o-mc* might grow into a *p-mc* over time, and a *p-mc* can fade into an *o-mc*. Forming micro-clusters requires the parameters ϵ and $MinPts$ representing the neighbourhood radius and the minimum number of points in a neighbourhood, respectively. We fix their values at 2 and 3 respectively based on results from Section 4.4.2. These values reflect that a *p-mc* must have a cardinality of at least 3, otherwise it is an *o-mc*. For CIM, we are only interested in the set of potential-micro-clusters, MC_p, which represent a set of similar agents we have

[1]Line 6 shows C_s and C_t are initialised to the same set of micro-clusters

interacted with in the past, and therefore that we have enough representative data about them for S to be accurate about their behaviour. We still keep track of MC_o because they may grow into potential-micro-clusters if future points are added to them. In CIM, we maintain a snapshot of MC_p at two different times, where C_t represents the MC_p for the current time and C_s represents what MC_p was when S was built. Line 3 calls for an online update to C_t using the C-DenStream algorithm, so that C_t can be updated every timestep efficiently. C-DenStream uses a forgetting factor, λ, to give precedence to new data by reducing the weights of micro-clusters every timestep.

As part of building and maintaining micro-clusters with a semi-supervised algorithm, Cannot-Link constraints are enforced between labelled data which are transformed to constraints between the micro-clusters each instance belongs to. The label is the trust value binned into one of 10 bins, and if no trust value was provided because the certainty in it was not high enough, the label is -1. Points and micro-clusters need to be *label consistent* amongst members of the same cluster, meaning they are either labelled with the same trust value, or one is labelled and the other is not. Once a labelled point is added to a previously unlabelled micro-cluster, the micro-cluster is labelled and a label weight, l_w, is initialised for the micro-cluster. If an instance of the same label is added to the micro-cluster, the label weight is increased by one, otherwise it fades according to the forgetting function: $l_w = l_w * \lambda^{t-t'}$, where t is the current time, t' is when l_w was last updated, and $\lambda \in [0, 1]$ is a forgetting factor. Forgetting factors are a common technique for models to give more weighting to recent data, we use $\lambda = 0.98$ to prevent forgetting too quickly but also not retaining everything. If few, or no, labelled instances arrive, then l_w will fall below a threshold, calculated in the same way as a micro-cluster weight threshold for being deleted described in the literature, and the micro-cluster will lose its label. We do not enforce Must-Link constraints because if two agents are assessed to have the same trust value this does not imply they are guaranteed to belong to the same group, as either the trust could be miscalculated or two different groups have the same behaviour at the time.

The way that agents use CIM to aid their decision making is described in Algorithm 3, which uses C_s and C_t to decide if a trustee should be assessed with the stereotype model. We account for dynamic behaviour by comparing C_t with C_s. If there are label changes between the two clusters then we assume that behaviour has changed. Additionally, line 13 checks that there has been enough data collected to use CIM or the stereotype model, both of which are built once L amount of data are collected.

Algorithm 3 Behaviour assessment using CIM with trust and stereotype assessment.

```
1: function BEHAVIOUR_ASSESSMENT(te,O_tr)
2:     if CIM then
3:         a ← S(te)
4:         return a
5:     else
6:         a ← 0.5
7:         return a
8:     end if
9:     trust ← trust_alg(O_tr, te, a)
10:    return trust
11: end function
12: function CIM
13:    if C_s and C_t are empty then
14:        return False
15:    end if
16:    label_s ← Cluster(τ⃗_te, C_s)
17:    if no cluster found then return False        ▷ This accounts for class imbalance
18:    end if
19:    label_t ← Cluster(τ⃗_te, C_t)
20:    if label_s ≠ label_t then return False       ▷ This accounts for dynamic behaviour
21:    end if
22:    return True
23: end function
24: function CLUSTER(τ, Clustering C)
25:    label ← −1
26:    for Cluster c ∈ C do
27:        if dist(τ⃗, c_center) < ϵ then
28:            label ← c_label
29:        end if
30:    end for
31:    return label
32: end function
```

4.3 DCS

DCS provides an initial assessment of a trustee without using the trustor's past interaction data, and can be used when no stereotype model exists or when CIM has assessed the stereotype model to be inappropriate for assessing the trustee. DCS compares an agent's own features to the trustee's, and if they are similar enough, the trustor assumes the trustee has the same behaviour as well. We assume an agent has awareness of its own observable features, though not which are relevant or irrelevant, and its own behaviour. DCS has two advantages, first, it does not require any past experience data so can be used immediately by new agents, or for agents whom no one knows anything about. Second, it does not rely on information from reputation, which may come from lying or biased sources.

The similarity between two agents can be calculated using a distance metric between each value in the set of observable features. Using a Euclidean distance metric, the similarity, $sim_{tr,te} \in [0, 1]$, between a trustor, tr, and a trustee, te, with observable features of length, n, can be calculated as:

$$sim_{tr,te} = \sqrt{\sum_{i=0}^{n} (|\overrightarrow{\tau^i_{tr}} - \overrightarrow{\tau^i_{te}}|)^2} \tag{4.1}$$

If $sim_{tr,te}$ is within a threshold of similarity, ϕ, the trustor infers that the trustee will have the same behaviour, and this replaces the *a priori* value of Equation 3.7. We demonstrate in Section 4.4.3 that the value of ϕ should be high to prevent misinterpreting a partner, and we use $\phi = 0.85$ as default value in all other evaluations based on those results.

$$a_{te} = \begin{cases} stereotype(a_{tr}), & \text{if stereotype model available} \\ bhv_{tr}, & \text{if } sim_{tr,te} < \phi \\ 0.5, & \text{otherwise} \end{cases} \tag{4.2}$$

DCS only gives a trustor insight into a small subset of trustees, but if this encourages interactions with good agents, or avoids bad agents, the resulting interaction outcomes are propagated by the reputation algorithm to the benefit of all trustors. In some existing literature and our own initial proposal of DCS, agents can seed a stereotype model with their own observable features [Player and Griffiths, 2017a,b; Wagner, 2012]. However, in this thesis we show that a simple threshold of similarity for DCS is sufficient to achieve good results in the short time period before a stereotype model can be built. DCS is a simple approach, which could be subverted by a simple attack such as mimicking. However, certain observable feature might be difficult to mimic in the application

e.g. requiring certificates to display certain feature values.

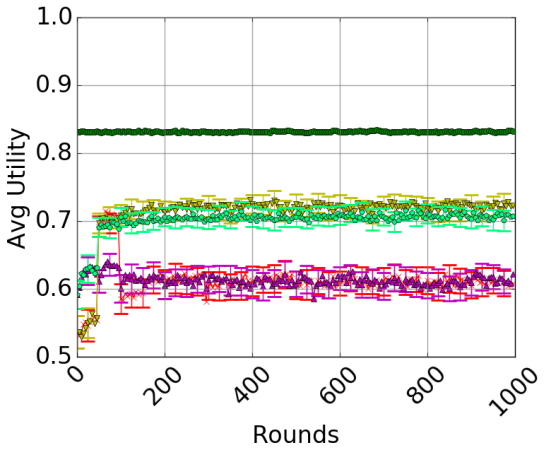
4.4 Evaluation

In this chapter we present the results of the performance of CIM and DCS when working in conjunction with the M5 decision tree stereotype model from Burnett *et al.* [Burnett et al., 2010], which is described in detail in section 3.6. CIM can account for class imbalance (CI) and optionally for dynamic behaviour (DB) as well. For results where CIM is accounting for both, we denote it as CIM-CIDB, and when the results are only accounting for class imbalance, we use CIM-CI. The evaluation environment in which we generate these results was described in Chapter 3, where the parameters and their values were listed in Table 3.2. All the results presented in this chapter are statistically significant using a paired t-test where $p < 0.001$.

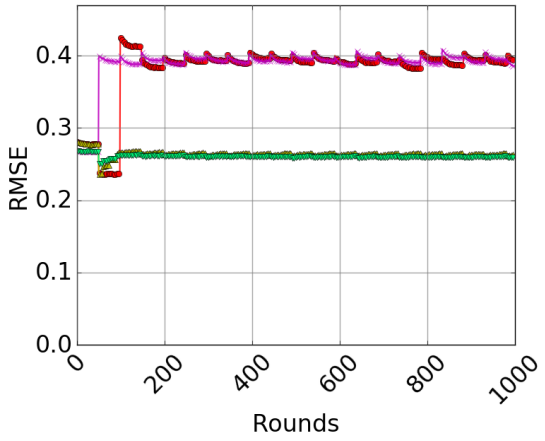
The performance metric is average (avg) utility, as this describes if the group of agents as a whole increased their average welfare. Most of the results in this thesis also display the "optimal" average utility for comparison, which is the average utility the agents could collectively achieve if they made the best partner choices. The models' accuracy is measured using the RMSE metric, to calculate the error between the trustor's trust assessment of trustees and the trustees' true behaviour. These metrics are described in more detail in chapter 3.4.

4.4.1 Class Imbalance

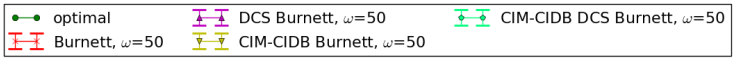
Figure 4.1 depicts how the stereotype model can be improved with CIM. Agents have static behaviour, a population turnover rate, p_{leave}, of 0.1 and are connected in a fully connected graph topology. We use the DRS trust model where $k = 2$, this is specifically known as BRS. From Figure 4.1(a) we can see that with or without DCS, CIM significantly improves the average utility agents receive. There is no added benefit from DCS in these results because with static behaviours, agents learn the best profile to interact with quickly which encourages further data collection for them and so CIM suggests using the stereotype model to evaluate them which leads to continually selecting the best agents. DCS does not improve partner selection because the stereotype model is already doing the best it can in this static environment. In Section 4.4.3, and in the following chapter on dynamic behaviour, we review results in an environment where agents depend on DCS to identify good interaction partners because the stereotype model cannot. We can see that the error in the trust assessment of the agent is significantly lower when using CIM, because agents are not being misclassified. CIM improves accuracy because in the early stages of a trustor's



(a) Avg. Utility per agent.



(b) Avg. RMSE between trust and behaviour.



(c) Legend.

Figure 4.1: Agent performance when using CIM and/or DCS.

life, CIM identifies that most the available trustees cannot be assessed with the stereotype model. This promotes interactions with either agents that are accurately classified as a good stereotype, or unknown agents. In the latter case, the trustor then collects more data about an unknown agent which reduces the level of class imbalance in the data and ultimately builds a more accurate stereotype model.

Table 4.1 gives a brief overview of how CIM performs when 100 agents are connected in different graph topologies. The results show that the only topology there is statistically significant difference in is a fully connected graph. Recall

from Table 3.1 that the graph topology affects the size of agents' neighbourhood. When this neighbourhood is small, trustors quickly gather sufficient information about their neighbours to use trust assessment methods. In this thesis, we aim to improve partner selection in the context of little and sparse data. This problem is most prominent in a fully connected graph and hence why CIM offers the most improvement here. The remainder of our results specifically address this case. We present results for different topologies in each chapter, to show the effect of agents' connectivity however, they all show a similar pattern.

Table 4.1: Average utility per agent per time step in different graph topologies.

	Graph		
	Scale Free	Small World	Fully Connected
Burnett	0.645 (0.018)	**0.609** (0.021)	0.611 (0.019)
Burnett CIM	**0.659** (0.018)	0.607 (0.021)	**0.711** (0.015)

4.4.2 Noise

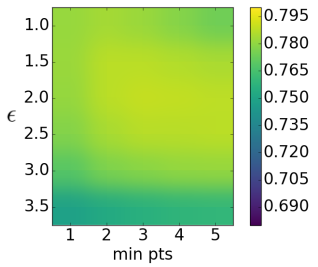
The number of irrelevant features, n_{nf}, and the variation in the values of relevant features, Θ, could prevent the clustering algorithm from accurately identifying agents who are members of the same profile. In this section, we first vary the amount of noise to identify the best values for the clustering parameters, $MinPts$ and ϵ, by comparing utility and purity. We then present the improved performance on the stereotype model when using those parameters.

One measure of cluster accuracy is *purity*. Clusters in CIM should contain instances from the same ground truth class (a profile, in this case), even if those instances are split over multiple clusters. The clusterer should not contain instances from differing profiles in one cluster. Purity is calculated as:

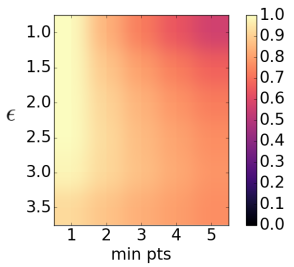
$$\frac{1}{N}\sum_{m\in M} max_{d\in D}|m \cap d| \tag{4.3}$$

where, N is the total number of data points, M is the set of clusters and D is the set of classes. So for each cluster, we count how many point form the most common class and average this value across all the clusters. If an instance is in an outlier micro-cluster (possibly of just itself) then it is counted as a misclustered instance, because it will not contribute to future classifications of instances, and has failed to be captured in a potential micro-cluster.

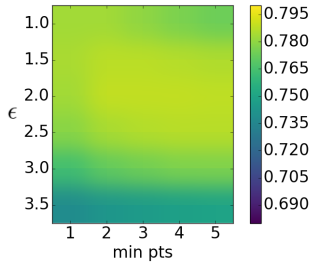
Figures 4.2 and 4.3 show how CIM performs with regards to the average utility agents receive per timestep, and the purity of the clusters formed by CIM, in environments with increasing noise. Figure 4.2 increases the feature variance Θ while the number of noisy features is fixed, $n_{nf} = 2$. Then Figure 4.3 shows



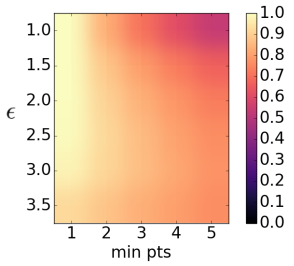
(a) Avg. utilty per timestep, $\Theta = 0.0$.



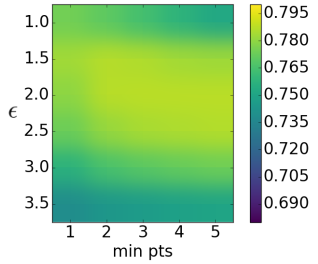
(b) Purity, $\Theta = 0.0$.



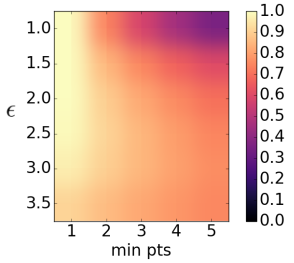
(c) Avg. utilty per timestep, $\Theta = 0.2$.



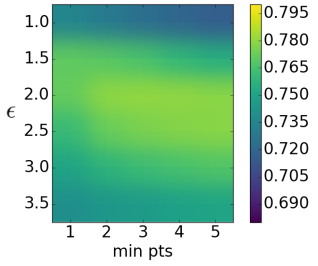
(d) Purity, $\Theta = 0.2$.



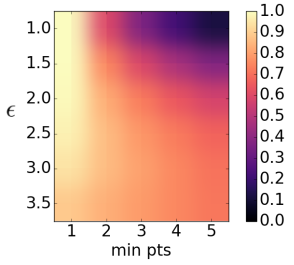
(e) Avg. utilty per timestep, $\Theta = 0.5$.



(f) Purity, $\Theta = 0.5$.



(g) Avg. utilty per timestep, $\Theta = 1.0$.

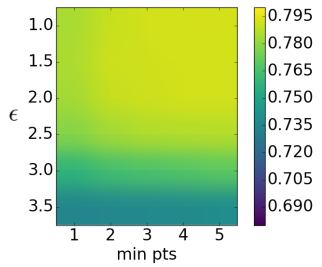


(h) Purity, $\Theta = 1.0$.

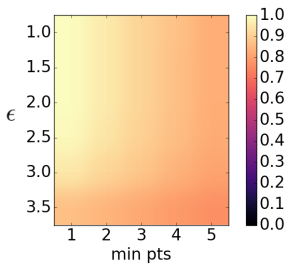
Figure 4.2: Performance of CIM for different $MinPts$ and ϵ when varying noise with Θ.

the same results as n_{nf} increases and Θ is fixed at 0.2.

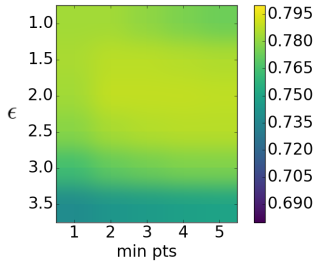
Both Figures 4.2 and 4.3 show that as noise increases, the average utility



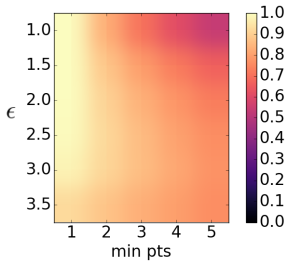
(a) Avg. utilty per timestep, $n_{nf} = 0$.



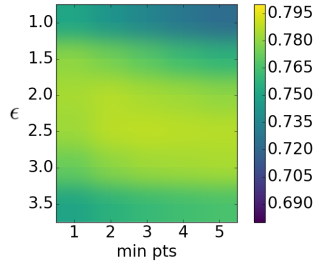
(b) Purity, $n_{nf} = 0$.



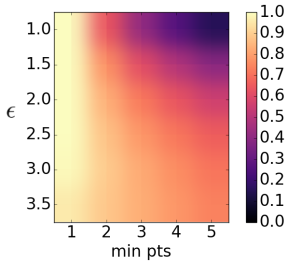
(c) Avg. utilty per timestep, $n_{nf} = 2$.



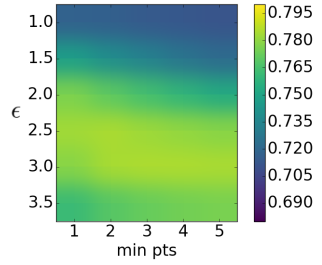
(d) Purity, $n_{nf} = 2$.



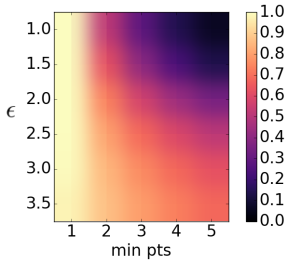
(e) Avg. utilty per timestep, $n_{nf} = 4$.



(f) Purity, $n_{nf} = 4$.



(g) Avg. utilty per timestep, $n_{nf} = 6$.



(h) Purity, $n_{nf} = 6$.

Figure 4.3: Performance of CIM for different $MinPts$ and ϵ when varying noise with n_{nf}.

agents receive per timestep and the purity of the clusters decline. This means that as noise increases, stereotypes are harder to identify for both CIM and

the stereotype model, and ultimately this leads to inaccurately assessing the *a priori* trust of an agent leading to poor partner selection. Overall, values of ϵ between 2 and 3 seem to perform best in all scenarios, which could be attributed to the purity of the clusters. A smaller cluster radius groups only agents who are very similar for all features, relevant and irrelevant, but will identify more clusters. While some agents are considered anomalies by CIM as a result of these smaller radii because their irrelevant features are too different, it also prevents misclassifying agents of one profile as another, which we confirm with the high purity values. Figure 4.3(g) shows when ϵ falls in the range [2,3], CIM still performs well when there are more noisy features than relevant ones. One disadvantage of smaller ϵ values is that multiple, smaller clusters will fade faster than one larger cluster which encapsulates all agent interactions of one profile because lots of smaller individual clusters are not updated as regularly. However, we can see that as we increase $MinPts$ and ϵ, larger clusters become impure, so CIM can no longer identify stereotypes.

Some values of $MinPts$ and ϵ appear more resilient to noise. However, we recognise that these parameters can be sensitive and selecting their values may be a domain specific task. Based on the results from Figures 4.2 and 4.3, we fix the values of MinPts and ϵ at 3 and 2, respectively, for the remainder of experiments using CIM in this thesis.

Table 4.2: Effects of noise parameters, without DCS.

	Θ			
n_{nf}	0.0	0.2	0.5	1.0
0	6.535%	9.775%	9.251%	7.727%
2	7.427%	9.806%	9.127%	7.716%
4	7.23%	8.965%	8.148%	5.057%
6	4.598%	6.035%	4.343%	0.733%

Having chosen parameters of CIM, Table 4.2 shows the percentage improvement of CIM when it is applied to a stereotype model in environments with increasing noise. Table 4.3 shows the percentage increase in performance from a stereotype model when using CIM and DCS. As n_{nf} increases, agents perform better when using DCS compared to without. As noise increases, the stereotype model cannot separate agents into their true profiles accurately. CIM detects this by also not identifying sufficient clusters and recommends the agent not to use the stereotype model. In both tables, there is increased performance from doing this. However, agents using DCS have more insight to initiate good interactions compared to agents who do not use DCS and only a default *a priori*.

Table 4.3: Effects of noise parameters, with DCS.

	Θ			
n_{nf}	0.0	0.2	0.5	1.0
0	6.529%	10.204%	9.706%	7.808%
2	7.594%	10.052%	9.642%	7.935%
4	7.917%	9.998%	8.961%	6.016%
6	7.202%	8.422%	6.806%	2.857%

4.4.3 DCS

In this chapter, we introduced a method for agents to infer a simple stereotypical trust value, called DCS. Agents compare their own observable features their partner's observable features and if they are within a threshold of similarity, ϕ, assume their behaviour is the same as their own. This method can be used when a more sophisticated stereotype model is unavailable, because either there are too little data to build one or because CIM has indicated that a stereotype model will be biased against the agent being assessed. The value of ϕ should be high because agents who are identical are likely to have the same behaviour, but agents who are not observably similar have no reason to act similarly, even though they may. Figure 4.4 shows how the performance varies as the value of ϕ increases.

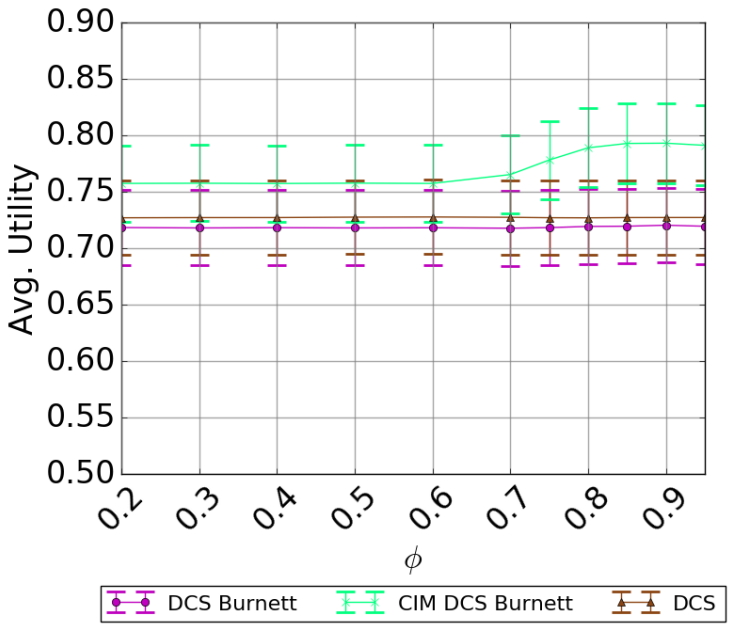


Figure 4.4: Varying ϕ threshold in DCS.

The results for DCS and DCS Burnett are not statistically significantly dif-

ferent for different values of ϕ. When using DCS Burnett, the DCS component is not being relied on because once the stereotype model is initialised after L interactions and is always used for the remainder of the trustor's life. A flaw in stereotypes is their inability to realise that the model is inappropriate, and revert to a different method. CIM outperforms the stereotype model, because as the stereotype model is deemed inappropriate it can rely on DCS. The results for CIM-DCS-Burnett show increasing improvement until ϕ gets sufficiently high that it can accurately identify only the agents in its profile.

DCS can only foster interactions with the best agents by those agents who are know they too are the best, and vice-versa it can also prevent agents who are really bad from interacting with really bad agents. This technique alone would not improve results much, however after these initial interactions have happened, they are propagated as a result of reputation, allowing agents who could not identify the best agents through DCS to now select good partners.

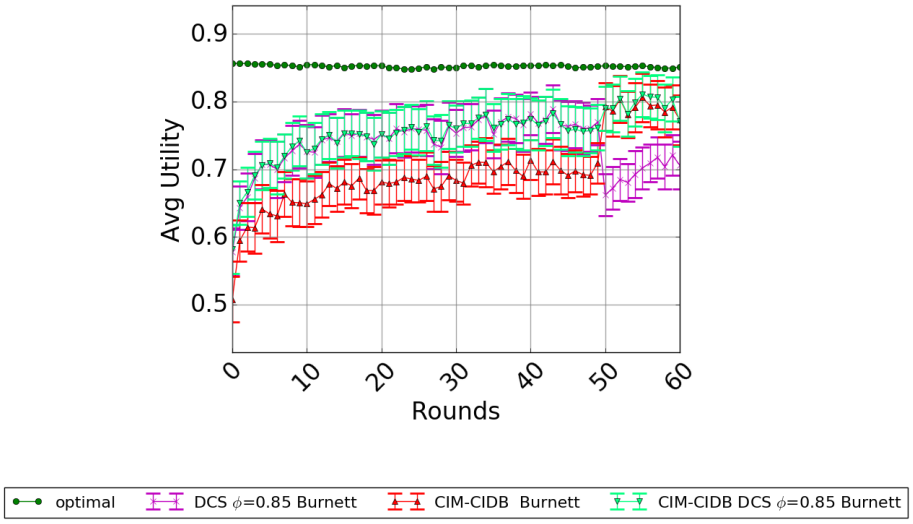
Figure 4.5(a), where $\phi = 0.85$ based on the results from Figure 4.4, shows how in the early stages of trustors' lives, before the learning interval is over and data-driven stereotypes are available, how each model performs. The model without DCS is significantly worse, and as soon as the learning interval, $L = 50$[2], is over, both CIM models perform best regardless of their use of DCS, as they manage when to use and not use the stereotype model. Then looking at Figure 4.5(b), we can confirm that a very low value of ϕ is worse than not using DCS because it is misinterpreting partners, but that high values of ϕ perform very well over not using DCS at all.

DCS shows such high improvements here because it is the beginning of the trustor's life and they have no alternative information to use. We demonstrate in Chapter 3 how DCS can be used for an *a priori* when there are not enough data. Therefore, if an agent has too few data after removing old data then we can revert back to using DCS.

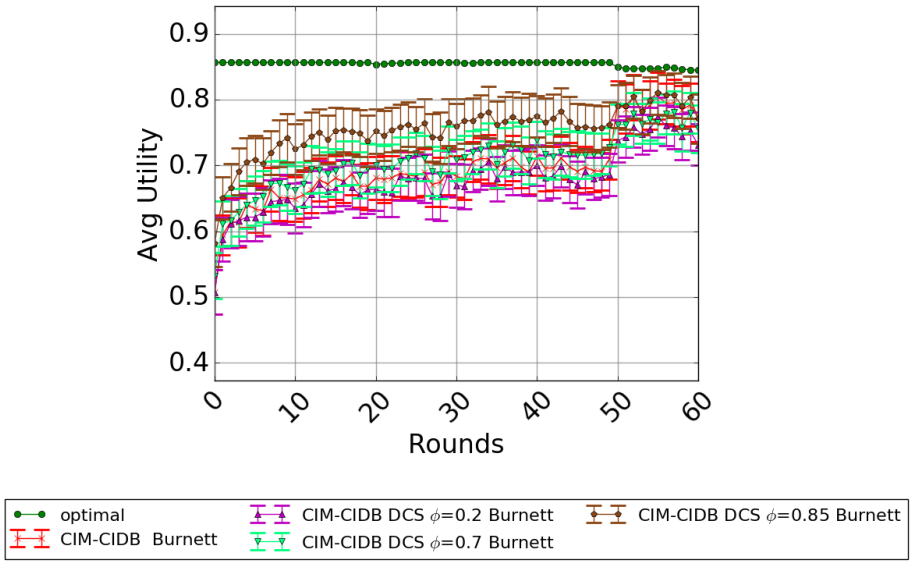
4.4.4 Dynamic behaviour

One cause for a stereotype model to become unrepresentative is if the correlations of features to behaviour it has learned have since changed. In Chapter 5 we investigate further the effects of dynamic behaviour on trust and stereotype assessment and how this can render past experience data unusable. We include a brief evaluation of CIM with dynamic behaviour here to demonstrate how robust CIM is. Dynamic behaviour was described in detail in Section 3.2. It is defined by p_{Gr} and p_{Su} to describe the extent of gradual and sudden behaviour

[2]This value is selected from the stereotype literature used for this evaluation [Burnett et al., 2010], however, we demonstrate in Chapter 5 that other values of the learning interval do not affect the long term efficacy of the models.



(a) Comparing DCS models with non-DCS models.



(b) Comparing values of ϕ in DCS.

Figure 4.5: Performance of DCS.

change, respectively. When both p_{Gr} and p_{Su} are 0 behaviour is static.

We can see from Tables 4.4 and 4.5 that both CIM and CIM-DCS are robust to dynamic behaviour. Intuitively, CIM with DCS is more robust than without DCS because agents use information about their own behaviour with DCS, and this is up to date. CIM will identify that behaviour has changed, and guide the agent to use DCS more frequently.

The threshold υ specifies the maximum amount of uncertainty an agent can

Table 4.4: Percentage improvement of average utility of stereotype model using CIM.

	Gradual			
Sudden	0.0	0.001	0.005	0.01
0.0	9.86%	9.475%	7.108%	6.256%
0.001	8.506%	8.305%	6.857%	5.899%
0.005	6.539%	6.248%	5.538%	4.13%
0.01	5.304%	4.932%	4.03%	3.569%

Table 4.5: Percentage improvement of average utility of stereotype model using CIM and DCS.

	Gradual			
Sudden	0.0	0.001	0.005	0.01
0.0	10.228%	9.963%	8.665%	8.283%
0.001	9.55%	9.547%	8.508%	8.215%
0.005	8.467%	8.535%	8.124%	7.628%
0.01	7.944%	7.836%	7.343%	7.17%

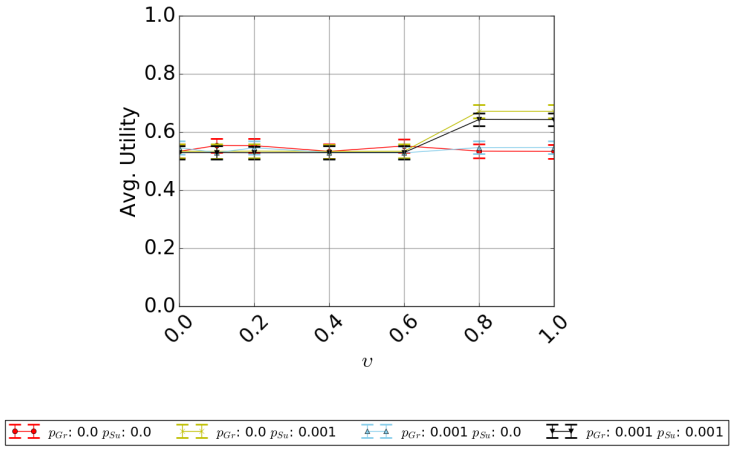


Figure 4.6: Average utility per agent per timestep for differing υ and dynamic behaviour.

have in a trust assessment to add the interaction record as a labelled instance to the semi-supervised clusterer. Figure 4.6 shows υ has no effect on the efficacy of CIM when agent behaviours are static because cluster labels are not changing anyway, so it does not matter how many instances are labelled. As p_{Gr} and p_{Su} increase, we can see that particularly sudden dynamic behaviour, benefits from higher values of υ.

4.5 Discussion

In this chapter, we address RO 1 by proposing CIM, a method to identify whether an agent is fairly represented by the data, and therefore whether the data can be used to build an appropriate stereotype model that will accurately assess the agent. CIM uses an online, density-based, semi-supervised clustering algorithm which avoids the need to specify the number of true profiles in advance, and enables clusters to be updated over time. The selfish nature of agents encourages them to interact with agents with whom they have prior experiences, biasing their history of interaction outcomes to a subset of agents and a subset of profiles. Therefore, their history of interactions contains a class imbalance. Most machine learning literature discusses, or accounts for, the class imbalance, and in this chapter we have seen how it can adversely affect stereotype models in MAS. We demonstrated that agents who address class imbalance using CIM could improve their trust assessments and interaction outcomes without the need to explore. Exploration can tackle class imbalance but incurs the cost of bad interactions. In contrast, CIM identifies when class imbalance exists for a particular agent and adjusts its trust assessments of that agent accordingly. A future direction would be to integrate class imbalance detection as a prompt for agents to begin exploring, and try to balance the need for exploration with exploiting the knowledge gained from it. Another key finding was that the level of noise in profile feature values impacted the clustering technique, however, through experimentation we found values for the input parameters of CIM, $MinPts$ and ϵ, which increased resilience to this noise. Additionally, as the noise levels increased, using CIM was always found to improve on using a stereotype model alone.

In this chapter, we proposed DCS to address RO 2, improving trust assessments when past experience data are unavailable. DCS is highly effective in the early stages of a trustor's lifetime, before it can make stereotype assessments. DCS can offer long term improvements when working with CIM, as a fall back assessment method when CIM deems the stereotype model inappropriate for assessing an agent. DCS can only help a trustor evaluate agents who have very high similarity, which is a limited subset of the available partners. This helps some trustors identify good agents, or some trustors avoid bad partners. The efficacy of DCS is increased when working in conjunction with a reputation model, because when the first few interaction partners are chosen based on DCS, they are better than a random selection. Those interaction outcomes are then propagated through a reputation algorithm to inform other trustors.

We evaluated CIM in the context of both static and dynamic agent behaviour. CIM offers improvements given agents have dynamic behaviour, be-

cause CIM identifies that the steretoype model should not be used more often than in a static environment. The agent can further improve their performance if they also use DCS, because DCS provides insight for partner selection when the stereotype model is unavailable. Additionally, agents rely on DCS more frequently when agent behaviours are dynamic compared to static, so the effect of DCS is more noticeable. A further evaluation of CIM when dynamic agent behaviours exist is provided in the next chapter.

CHAPTER 5
Dynamic Behaviour

In Chapter 4, we explored the negative impact of using unrepresentative information when assessing trust, and how we can overcome this in a static environment. In this chapter, we explore how dynamic agent behaviours can increase the amount of irrelevant information agents have, which affects the accuracy of their trust assessments. We propose a technique, which addresses RO 3, to improve trust and reputation assessment while accounting for agent behaviour changes. Our approach monitors interaction outcomes with agents from identified stereotypes using concept drift techniques. We use the term concept in this context, as a single point value representation of agent behaviour. Agents may change behaviour over time, implying they are changing concepts. An overview of concept drift techniques was provided in Chapter 2, and we describe the details of the specific algorithms drawn up in our work as we describe our models. Specifically, we use the Adaptive Window (AdWin) concept drift method to monitor a decision tree stereotype model, which we refer to as AdWin Tree. By analysing the interaction outcomes from identified groups separately, agents have the flexibility to select the data that they believe are representative of that group, which may span a different length of time from those for other groups. Our evaluation includes a comparison of AdWin Tree to CIM, and the combination of the two techniques, to explore if they can collectively account for class imbalance and dynamic behaviour.

5.1 Introduction

Dynamic agent behaviour is a characteristic of MAS which is typically accounted for in a simplistic way by existing trust, reputation and stereotype models. Agents in open distributed MAS may not experience behaviour changes at exactly the same time or rate. For example, agents in the same location may experience communication issues or power outages together, affecting their ability to provide services and their trustworthiness, but agents in other areas may be unaffected. Quickly adapting trust to reflect an agent's true behaviour will allow agents to choose interaction partners who are most likely to succeed at the task at the current time. In some contexts, quick adaptability is vital, such as in emergency response scenarios.

Dynamic agent behaviour renders past interaction experiences about a specific agent inappropriate for making a trust assessment because the data no

longer represent that agent's current behaviour. Unrepresentative past interaction records affect stereotype models because the increased noise in the data prevents accurate correlations from being learnt between relevant observable features and trust values. Existing trust and stereotype models use techniques such as sliding windows and forgetting factors to account for noisy data but these approaches still capture a lot of unrepresentative information for three reasons. Firstly, they can only account for gradual change and not sudden changes of behaviour. Secondly, they require knowing in advance what the gradual rate of change is to tune the parameters to capture relevant information. Thirdly, they assume that all agents have the same rate of change, and there is no flexibility to account for different rates depending on the target agent.

In this chapter, we propose a technique for agents to cope with dynamic behaviour exhibited by a group of agents, given this may occur at different times or speeds depending on the agent. Our method, AdWin Tree, is an adaptation of an existing stereotype method [Burnett et al., 2010], which uses a decision tree to correlate agents' observable features with behaviour. AdWin Tree monitors interaction outcomes of identified stereotypes by adding them to an adjustable window of data at the leaves of the tree and performing the statistical two-sample Kolmogorov-Smirnov (K-S) test. Our approach is applied to a decision tree stereotype model, however, the intuition is to monitor interaction outcomes separated by their identified stereotypes, and therefore any stereotype model that outputs a description of the stereotype could be used. Our approach could also be applied to individual agents, however, it is unlikely that there will be sufficient interaction experiences with every individual to accurately analyse their behaviour in this way. In Chapter 6, we will build on the work presented in this chapter which detects behaviour change, to learn and exploit agents' behaviour patterns.

Both AdWin Tree and CIM (described in the previous chapter) aim to account for dynamic behaviour. However, CIM uses a forgetting factor which suffers the limitations described above. As part of this chapter, we include an in-depth evaluation of CIM in a dynamic environment and compare its efficacy to AdWin Tree. There are four differences between CIM and AdWin Tree regarding the handling of dynamic agent behaviour. Firstly, AdWin Tree only deletes interaction records when change is detected. This gives AdWin Tree more flexibility to retain different amounts of data for each stereotype. One disadvantage of this, is that AdWin Tree can retain data forever if no change is ever identified. This can add noise to the dataset and prevent accurate stereotypes from being learnt, as well as potentially exceeding an agent's memory capacity. Alternatively, CIM uses a forgetting factor that assumes that change has occurred unless there is recent data maintaining the status quo. Secondly, AdWin

Tree analyses data belonging to a group of agents, and therefore it depends on how accurately the groups of agents have been identified. For example, if groups are not immediately identifiable so we use a stereotype model, some agents may be incorrectly classified. Monitoring interaction outcomes for a group where the agents are not truly behaving similarly adds noise to the data and makes it more difficult to analyse. CIM does not depend on any existing model, and groups agents together itself. Unlike AdWin Tree, CIM can work with or without a stereotype model. Thirdly, AdWin Tree monitors interaction outcomes for changes over time, whereas CIM analyses trust over time. Outcomes can be more appropriate to analyse because they are not biased by the choice of trust algorithm, and represent a data point from agents' current behaviour. Trust assessments are influenced by data which may no longer be representative of the behaviour. Finally, AdWin Tree detects changes and then alters an agent's history of interaction outcomes, which affects the data used in trust assessment as well, whereas CIM only decides whether or not the stereotype model is appropriate.

We show how the class imbalance element of CIM can work in conjunction with AdWin Tree to account for dynamic behaviour, instead of CIM using a forgetting factor. However, ultimately we show that when behaviour is static, CIM is the most effective technique to use on its own, and when behaviour is dynamic, AdWin Tree is the most successful model on its own.

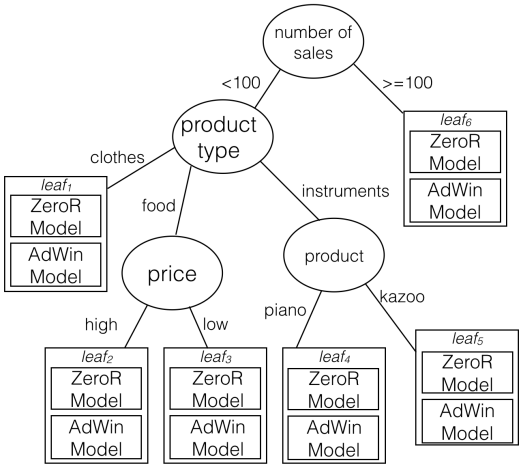


Figure 5.1: Example AdWin Tree

5.2 Accounting for Dynamic Behaviour

We present a method for agents to analyse the interaction outcomes with a group, to determine if the group's behaviour has changed over time. Our technique allows agents to react to behaviour change by updating trust and stereotype assessment to reflect the group's current behaviour. This method draws upon an assumption used by stereotype models, that agents display some observable features which correlate with their behaviour. Our technique can bootstrap any existing stereotype model that outputs a set of identified stereotypes, $\overrightarrow{S}$. Stereotypes are found by correlating agents' observable features with their trust. Our method takes in the set of discovered stereotypes, $\overrightarrow{S}$, the history of past interactions, $\mathcal{O}$, and the new outcome after an interaction, o^t, at time t, and updates the set of interaction outcomes if change is detected in a stereotype by deleting records for that stereotype from before the identified point of behaviour change i.e. $\mathcal{O} \leftarrow f(\overrightarrow{S}, \mathcal{O}, o^t)$. Any trust and reputation model can be used provided that its output can be normalised to be in the range $[0, 1]$. Choosing an appropriate trust and reputation model is application dependent. Our work is evaluated using the decision tree stereotype model from Burnett *et al.* [Burnett et al., 2010] and the DRS reputation algorithm [Jøsang and Haller, 2007]. Algorithm 4 describes how the AdWin Tree is used to substitute an *apriori* value, a, into a trust algorithm, if there is sufficient data for the stereotype model to have been built. Line 3 describes how the trustee's features are classified to identify which stereotype the trustee belongs. Line 4 then explains that the ZeroR model for that stereotype, which is at the leaf of the stereotype decisions tree as depicted in Figure 5.1, outputs the behaviour estimate.

Algorithm 4 Behaviour Assessment using the AdWin Tree.

```
1: function BEHAVIOUR_ASSESSMENT(te,O_tr)
2:     if |O_tr| > L then
3:         leaf ← classify(τ⃗_te)
4:         a ← leaf_ZeroR()      ▷ use the ZeroR model for behaviour assessment
5:     else
6:         a ← 0.5
7:     end if
8:     trust ← trust_alg(O_tr, te, a)
9:     return trust
10: end function
```

Algorithm 5 describes how the AdWin tree is built and updated with an agent's interaction history and new interaction outcomes. To monitor the behaviour of groups we create an adaptive window (AdWin) model for each identified group, meaning we create a window for each stereotype in the identified set of stereotypes. Specifically, for the decision tree stereotype model used in our

evaluation, the leaves of the tree represent stereotypes and therefore the AdWin models are attached to the leaves of the tree, depicted in Figure 5.1. When the tree is built, the data which filter to a leaf is a subset of the agent's history of interactions, $\mathcal{O}_{tr}^{leaf} \in \mathcal{O}_{tr}$, and are interaction records with agents from that detected stereotype. The outcome values of those interactions are added to a variable sized window in chronological order and monitored with an adaptation of the AdWin concept drift algorithm [Bifet and Gavaldà, 2007]. Concept drift algorithms identify changes in a data stream over time. The efficacy of our approach is limited by how accurately the stereotype model identifies agent profiles. If one leaf incorrectly represents multiple profiles then the interaction records added to the adaptive window will be from different profiles, increasing the level of noise in the adaptive window. Noisy data prevent the concept drift model from accurately detecting a change in behaviour for a single profile. We modify the M5 algorithm from the original work by Burnett *et al.* [Burnett et al., 2010], by replacing the linear regression models at the leaves with ZeroR models. When the tree is being built and pruned, testing whether a node of the tree should have a subtree to further filter instances, or a leaf, depends on which would be more accurate, which is assessed with the accuracy of the trust prediction. Linear regression models allow for data from multiple profiles to exist at a leaf and still be distinguished, however this is inappropriate for AdWin Tree where a leaf should represent a single profile. Therefore, ZeroR models are more effective for predicting one class at a leaf. Using linear regression models in an M5 algorithm is not a limitation of the existing work as it is a very effective method for estimating trust however, an added requirement of our work is for the stereotype model to output a set of stereotypes. A parameter of building the decision tree is the minimum number of instances at a leaf. We set this to the minimum allowed value from the library used in our implementation, which is 2, since trustors may interact with some agent types as little as once or twice.[1].

AdWin Tree is first built when the trustor has collected L interaction records. In the decision tree stereotype model, a sliding window of past interaction records of size, ω, is used. The value of ω is the same as L, so the stereotype model is built every L interactions with the new data collected in that time. AdWin Tree only rebuilds when behaviour change is detected, but it needs to be initialised in the first instance. Agents initialise AdWin Tree when they have as few as 20 interaction records, and we show in our evaluation that larger values only delay reaping the benefits of a stereotype model. If subsequent to behaviour change being detected and handled there are fewer than 20 interaction records remaining, the trustor falls back on the default *a priori* of

[1]The original AdWin code provided by Bifet [Bifet and Gavaldà, 2007] which was adapted for this work can be found at https://github.com/abifet/adwin/.

Algorithm 5 Building and Updating the AdWin Tree.

1: **function** UPDATE_ADWIN_TREE(te, $\mathcal{O}_{tr}$, o^t)
2: $\mathcal{O}_{tr} \leftarrow \cup\langle te, \overrightarrow{\tau}_{te}, o^t, t\rangle$ ▷ update agent's history of interactions
3: **if** $|\mathcal{O}_{tr}| = L$ **then**
4: $Rebuild\_AdWin\_Tree(\mathcal{O}_{tr})$
5: **else**
6: $leaf \leftarrow classify(\overrightarrow{\tau}_{te})$
7: $leaf_W \leftarrow leaf_W \cup o^t$ ▷ Update the AdWin model at the leaf
8: $leaf_{ZeroR} \leftarrow leaf_{ZeroR} \cup o^t$ ▷ Update the ZeroR model at the leaf
9: $Behaviour\_Change\_Test(te, o^t, leaf_W)$
10: **end if**
11: **end function**
12: **function** $Behaviour\_Change\_Test(te, o^t, W)$
13: **for** $i \in [0, |W|]$ **do**
14: $W_0 \leftarrow W_{0:i}$
15: $W_1 \leftarrow W_{i:|W|}$
16: **if** KS Test(W_0, W_1) **then**
17: $\mathcal{O}_{tr} \backslash W_0$ ▷ Remove old window of data from history
18: $W \leftarrow W_1$ ▷ Remove old window of data from AdWin
19: $Rebuild\_AdWin\_Tree(\mathcal{O}_{tr})$
20: **end if**
21: **end for**
22: **end function**
23: **function** $Rebuild\_AdWin\_Tree(\mathcal{O}_{tr})$
24: $tree \leftarrow M5(\mathcal{O}_{tr})$ ▷ Using our modified M5 algorithm[2]
25: **for** leaf $\in tree\_leaves$ **do**
26: $leaf_W \leftarrow \{\}$
27: $leaf_{ZeroR} \leftarrow \{\}$
28: **for** $outcome \in \mathcal{O}_{tr}^{leaf}$ **do**
29: $leaf_W \cup outcome$
30: **end for**
31: $leaf_{ZeroR} \leftarrow \frac{\sum leaf_W}{|leaf_W|}$
32: **end for**
33: **end function**

0.5 until sufficient data are collected again.

5.2.1 Detecting Drift

When the tree is built, the trustor proceeds with interactions. After each interaction, the interaction outcome, o^t, is appended to the window of the AdWin model at the appropriate leaf of the decision tree, which is found by classifying the trustee's observable features, $\overrightarrow{\tau}_{te}$. This process describes line 6 of Algorithm 5. This leaf describes the stereotype we believe the trustee belongs to. The interaction outcome is dependant on the trustee's behaviour and how the trust algorithm perceives their behaviour. For example, when using the trust

assessment model DRS, where $k = 5$, the agent perceives o^t as being in one of 5 categories, so depending which category the interaction outcome falls into, the agent records it as the corresponding utility $o^t \in [0, 0.25, 0.5, 0.75, 1]$. When $k = 2$, the interaction is either good or bad, and so o^t has the perceived utility of 1 or 0, respectively. More details on DRS are given in Chapter 3. The original stereotype decision tree model uses linear regression models at the leaves, which are only rebuilt every L instances because of their time complexity. Therefore, new interaction outcomes are stored until the model is rebuilt and they have no immediate effect on stereotypical trust. AdWin Tree, however, can immediately update stereotypical trust assessment because updating a ZeroR model with o^t is simple.

To detect whether drift has occurred, we use a modification of the AdWin algorithm [Bifet and Gavaldà, 2007]. In the AdWin Tree, there are multiple windows at each leaf, referred to as $leaf_W$ in Algorithm 5 to denote the window is specific to that leaf. However, for the purposes of describing the AdWin algorithm which all the windows at all the leaves follow, we describe only one adaptive window, W. This section can also be seen in function $Behaviour\_Change\_Test$, in Algorithm 5. The adaptive window of data is divided into every possible split of two subwindows. The inputs to the AdWin algorithm are a confidence value $\delta \in [0, 1]$ and a sequence of real values $o^1, o^2, ..., o^t \in [0, 1]$ where the value of o^t is made available at time t, and represents the outcome of the interaction from time t. Each o^t is generated according to a distribution D_t, however the distribution may change over time. This represents the changing behaviour of agents in a profile. The confidence value is an input parameter to the test which determines how confident we are that two sets of data are drawn from different underlying distributions. We use $\delta = 0.2$ as a suggested value from the original work [Bifet and Gavaldà, 2009], and we demonstrate with experimental results in our evaluation that AdWin Tree is resilient to the value chosen for δ. We denote the length of window W at any point as n. As n increases, and if the underlying distribution generating the data does not change, i.e., agents' behaviour does not change, then the window size should grow. Having a larger quantity of data should improve the accuracy of the trust assessment.

Concept drift is identified using the K-S test. If a split of the window W into two subwindows, W_0 and W_1, exhibits "distinct enough" cumulative probability distributions, then the data in the two subwindows are assumed to have been generated from different underlying distributions with confidence $1 - \delta$. The K-S test is appropriate because it is compatible with continuous data, which means that our work can bootstrap trust algorithms which output any value type [Press et al., 2007]. We only start to use the K-S test when the size of W

is at least 4, allowing for a split of two subwindows of size at least 2. This is a generic small value because we can sometimes have very little data on each stereotype and prefer to start detecting behaviour change very early. Using less data can lead to false detection of behaviour change, but in this application we prefer to be overly sensitive to behaviour change in order to reduce the amount of noise in the data. However, this variable can be changed depending on the application, the amount of information available and the level of sensitivity to change detection required[3].

In the original AdWin algorithm, W is iteratively split into all combinations of W_0 and W_1 by first isolating the oldest instance and incrementally moving the first instance from W_1 into W_0, until either change is detected or all the instances in W are assumed to be from the same distribution. We identified that there is a performance difference in the AdWin algorithm if W is split into W_0 and W_1 in reverse order, such that the most recent instance is isolated first into W_1 with all other instances in W_0, then iteratively moving the last instance of W_0 into W_1. To keep the retained data as relevant as possible, conducting the split starting at the most recent instance is more effective. A gradual change in the distribution over data, which does not have one clearly defined time point of change, can mean there are several consecutive split points at which the null hypothesis could be rejected. By identifying the most recent split point, the most relevant data are retained in W_1. Additionally, if some subtle changes are missed in the earlier side of the window then it becomes noisy and harder to identify changes, but the more recent side of the window will retain interaction outcomes from one behaviour. Alternatively, starting the tests from older instances will result in the change being detected closer to the start of the window and thus retaining more, possibly irrelevant, data in W_1.

5.2.2 Handling Drift Detection

If drift is detected at time t' for one of the stereotypes, then the agent needs to remove interaction records from before time t' with any agent from that stereotype. Entries more recent than t' contain the aggregate of events with a trustee, including from before t', and therefore these records must be updated according to Equation 5.1. The M5 decision tree is rebuilt after the data have been updated.

$$\langle j, \overrightarrow{\tau_j}, t, r_j, s_j \rangle \leftarrow \langle j, \overrightarrow{\tau_j}, t, r_j - r_j^{t'}, s_j - s_j^{t'} \rangle \tag{5.1}$$

The overall effect is that from the point of change detection onwards, both

[3]The original AdWin algorithm, and our previously published work [Player and Griffiths, 2018a,b] used a different test, however any test can be substituted.

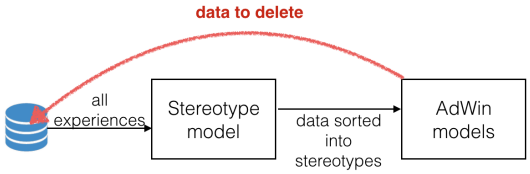
trust and stereotype calculations are not utilising any data that have been deemed redundant. An advantage of building separate AdWin models at the leaves of the stereotype decision tree is that data are retained from different time intervals depending on each stereotype's behaviour changes. This accounts for how different agent profiles can change behaviour at different rates and times.

The tree is only rebuilt after behaviour change is detected, and not every time L instances are collected, because behaviour change may be occurring but has not yet been detected. If the stereotype model was rebuilt, then the data would be noisy because some recent interactions with that stereotype have slightly different outcomes. The new stereotype model would learn two distinct stereotypes with different class values and behaviour change could not be detected because the data are split across separate adaptive windows. The amount of data would increase because behaviour change is never detected. Therefore, if there is no change detected in the stereotypes, it is not necessary to rebuild the stereotype model.

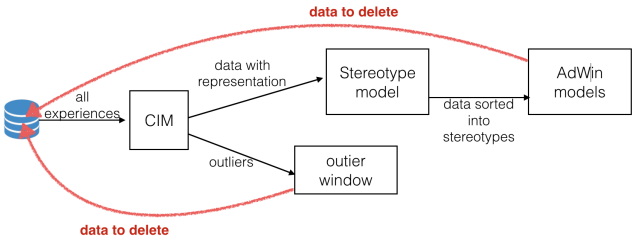
5.3 Integration with CIM

In the previous chapter, we introduced a tool to handle class imbalance in an agent's history of interactions. Integrating CIM with a stereotype model in the previous chapter was simple because the stereotype model did not influence which data needed to be deleted from the agent's history. Interaction experiences were stored as instances in a sliding window of size n, and when a new instances arrives, the $n^{th} + 1$ instance is deleted from the agent's history and will not be included when CIM and the stereotype model is rebuilt. AdWin Tree without CIM would update an agent's set of interaction outcomes when behaviour was detected, and as all interaction outcomes are used to update AdWin Tree, there is the possibility that all interaction outcomes can eventually be removed from $\mathcal{O}$, depicted in Figure 5.2(a). However, the following edge case arises when when deleting data from $\mathcal{O}$ when CIM is integrated with AdWin Tree. If CIM determines that an interaction partner is not represented by the stereotype model, then AdWin Tree should not be updated with the interaction outcome because it will add noise to the stereotype model. As the AdWin Tree is the only method to determine when data should be removed from $\mathcal{O}$, this gives rise to the possibility that data not added to the tree cannot be deleted from $\mathcal{O}$. Therefore, we create a window of outlier instances which are maintained differently. The outlier window is of variable size which adjusts to be the the average size of the AdWin models at the leaves of AdWin Tree. When the outlier window exceeds this value, older instances are removed from both the outlier window and $\mathcal{O}$, depicted in Figure 5.2(b). We retain these

records in the outlier window and do not delete them immediately because they have not yet been determined unrepresentative of current behaviours and can be used to assess agents with the trust and reputation algorithm. The variable outlier window does not need a specific input parameter for its size, and also reflects the dynamic nature of the agent behaviours. If the sizes of the AdWin models for each stereotype are small, it is an indication that agent behaviours are changing rapidly and so we subject the outlier instances to a small window too. Conversely, when agent behaviours are static or changing rarely and slowly then AdWin Tree will have large AdWin models at the leaves, and so the outlier model will be large too.



(a) AdWin Tree without CIM determines the relevant records to be removed from $\mathcal{O}$.



(b) AdWin Tree with CIM handles outlier data instances with a variable outlier window.

Figure 5.2: How AdWin Tree handles interaction records over time.

5.4 Evaluation

This section presents an evaluation of AdWin Tree against sliding windows and forgetting factors when they are applied to the DRS trust assessment model and a decision tree stereotype model. We evaluate this work in environments where agents can have gradual and sudden dynamic behaviour. Chapter 3 described the general evaluation environment, where the environmental parameters are summarised in Table 3.2. In this chapter, we introduced δ, the only input parameter to the AdWin Tree. We compare AdWin Tree against a stereotype decision tree using a fixed size sliding window of size ω. Additionally, for CIM we fix the values of $MinPts$ and ϵ at 3 and 2, respectively, based on the results from

the previous chapter. All the results presented in this chapter are statistically significant using a paired t-test where $p < 0.001$.

Table 5.1 compares how AdWin Tree performs against the benchmark model from Burnett *et al.* [Burnett et al., 2010], in contexts with increasing dynamic behaviour. The cells represent the average utility per agent per timestep, with the standard deviation to that in brackets. The columns are divided into different types of behaviour: static, gradual change, sudden change and a combination of both gradual and sudden. The different behaviour types are defined by the parameters p_{Gr} and p_{Su}, which are explained in detail in Section 3.2. Table 5.1 compares the effect of different input parameters for the models. Figure 5.3 visualises the results from each behaviour type, and compares the models over time. In these figures, we use the input parameters for each model which achieve the highest utility in Table 5.1, namely, $\omega = 100$ for a sliding window, $\delta = 0.2$ for AdWin Tree.

Table 5.1: Comparing AdWin Tree with decision tree stereotype model in environments with increasing dynamic agent behaviour.

		Dynamic Behaviour								
		Static	Gradual		Sudden		Both			
		p_{Gr}:0.0 p_{Su} : 0.0	p_{Gr}:0.001 p_{Su} : 0.0	p_{Gr}:0.005 p_{Su} : 0.0	p_{Gr}:0.0 p_{Su} : 0.001	p_{Gr}:0.0 p_{Su} : 0.005	p_{Gr}:0.001 p_{Su} : 0.001	p_{Gr}:0.001 p_{Su} : 0.005	p_{Gr}:0.005 p_{Su} : 0.001	p_{Gr}:0.005 p_{Su} : 0.005
AdWin Tree, δ	0.05	0.694 (0.017)	0.641 (0.021)	0.62 (0.022)	0.665 (0.023)	0.644 (0.025)	0.636 (0.021)	0.624 (0.023)	0.615 (0.023)	0.607 (0.024)
	0.1	0.698 (0.017)	0.646 (0.021)	0.626 (0.022)	0.668 (0.023)	0.649 (0.026)	0.64 (0.022)	0.629 (0.023)	0.621 (0.023)	0.612 (0.024)
	0.2	**0.699** (0.017)	**0.65** (0.022)	0.634 (0.022)	**0.671** (0.024)	**0.66** (0.025)	0.643 (0.022)	0.633 (0.024)	0.626 (0.023)	0.616 (0.024)
	0.3	0.697 (0.017)	0.649 (0.021)	0.634 (0.022)	0.669 (0.023)	0.656 (0.026)	0.644 (0.022)	0.636 (0.024)	0.629 (0.023)	0.619 (0.024)
	0.4	0.697 (0.017)	0.649 (0.021)	0.638 (0.022)	0.67 (0.023)	0.658 (0.026)	**0.646** (0.022)	0.638 (0.024)	0.631 (0.022)	0.622 (0.024)
	0.5	0.697 (0.017)	0.65 (0.021)	**0.639** (0.022)	0.669 (0.023)	**0.66** (0.025)	0.645 (0.021)	0.639 (0.024)	0.632 (0.022)	0.624 (0.024)
	0.6	0.698 (0.017)	0.65 (0.021)	0.64 (0.022)	0.671 (0.023)	**0.66** (0.026)	0.647 (0.022)	**0.64** (0.024)	**0.633** (0.022)	**0.625** (0.024)
Sliding Window, ω	20	0.627 (0.018)	0.601 (0.02)	0.589 (0.02)	0.606 (0.021)	0.609 (0.023)	0.594 (0.02)	0.594 (0.022)	0.588 (0.02)	0.588 (0.022)
	50	0.644 (0.017)	0.615 (0.02)	0.603 (0.02)	0.622 (0.022)	0.62 (0.024)	0.607 (0.02)	0.604 (0.023)	0.6 (0.021)	0.595 (0.023)
	100	0.648 (0.018)	0.62 (0.02)	0.599 (0.021)	0.624 (0.023)	0.618 (0.025)	0.607 (0.021)	0.596 (0.024)	0.594 (0.022)	0.583 (0.024)
	200	0.644 (0.018)	0.612 (0.021)	0.573 (0.023)	0.613 (0.023)	0.595 (0.026)	0.597 (0.022)	0.58 (0.024)	0.572 (0.023)	0.565 (0.025)

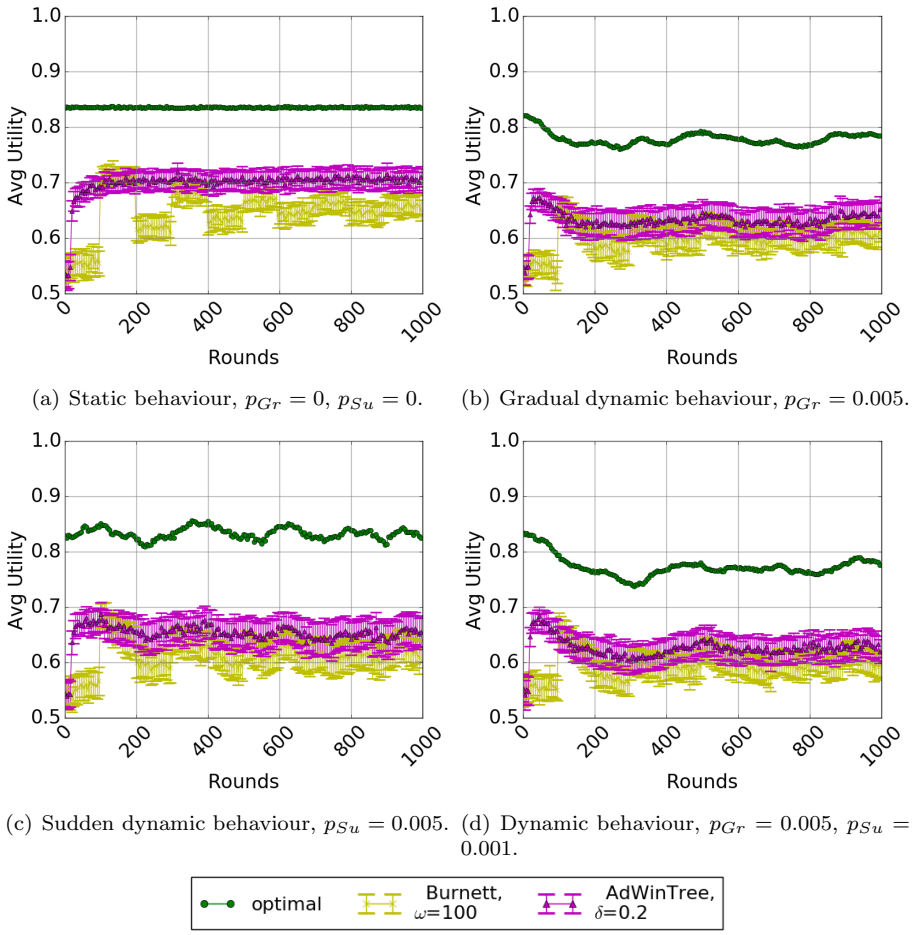
Our results show that AdWin Tree achieves a higher average utility per timestep than using a fixed window. As well as outperforming the benchmark model in a context of dynamic behaviour, we can see from Table 5.1 and Figure 5.3(a) that AdWin Tree also improves performance when behaviour is static. When behaviour is static, the benchmark model shows fluctuations over time. Similarly to in the previous chapter, this is a symptom of class imbalance. The data collected in a window of 100 instances only represent the agents that were interacted with in these 100 instances. At timestep 0 when all agents are new, they interact with a variety of agents to identify who they trust, and so their initial stereotype model built at time 100 has data collected from interactions with different agent types to accurately assess the trustees. However, between time steps 100 and 200, trustors only choose to interact with trustees they have assessed to be good, and the stereotype model built at time 200 only reflects those trustees. Therefore, the trustor's performance is reduced in the next interval because they cannot accurately assess all trustees or choose the best partners. AdWin Tree only deletes old interaction experiences if change is detected, and so older data exist to distinguish the good agents from bad agents.

The fluctuations reduce when there is sudden and gradual behavioural change because if a trustor continually interacted with the same subset of agents and those agents' behaviour is changing, by definition the interaction results with them will change. Additionally, as the interaction outcomes with agents change over time, the trust assessments update to reflect this. The trustor may choose different interaction partners as a result of the new trust values and this reduces class imbalance. AdWin Tree performs better than a fixed size sliding window, however, the increased difficulty to assess the best agents to interact with reduces the average utility in general.

5.4.1 AdWin confidence

AdWin Tree uses the K-S test to determine if two input sets of data were drawn from the same distribution. This null hypothesis is rejected if that probability is below a threshold, δ. The confidence in that decision is $1-\delta$. Table 5.1 presented results for different values of δ, which showed very little difference. Figure 5.4 visualises how sensitive AdWin Tree is to δ in the context of different levels of dynamic behaviour. Figure 5.4(a) presents the average utility per agent per timestep, and Figure 5.4(b) represents the average window size of the variable window.

Figure 5.4(b) shows that the adaptive window has an average size of between 80 to 210, approximately, in the differing environments of dynamic behaviour.



(a) Static behaviour, $p_{Gr} = 0$, $p_{Su} = 0$.

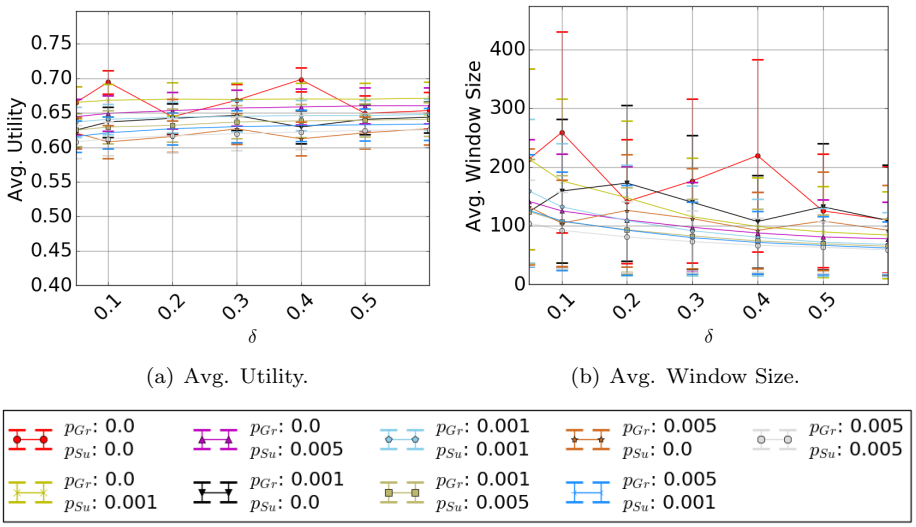
(b) Gradual dynamic behaviour, $p_{Gr} = 0.005$.

(c) Sudden dynamic behaviour, $p_{Su} = 0.005$.

(d) Dynamic behaviour, $p_{Gr} = 0.005$, $p_{Su} = 0.001$.

Figure 5.3: Average utility agents receive each timestep in environments with different levels of dynamic behaviour.

However, from Table 5.1 we can observe that if the window was fixed at either of these sizes, e.g., ω equals either 100 or 200, then the performance is significantly worse. We verify our hypothesis that agents need to adapt their window size to the circumstances by considering the standard deviation for the average window size. The standard deviation in the window size is approximately double the average, indicating the window size frequently changes. The average window size intuitively reduces as δ increases, because more changes are being detected and it is not given the chance to expand. Figure 5.4(a) shows that the shrinking window does not have a significant impact on the average utility agents receive. One reason is that when δ is small, even though the average window size is large, so is the standard deviation, signifying that changes are still being detected at critical times. If δ were to equal 1 then the AdWin Tree would always assume



(a) Avg. Utility.

(b) Avg. Window Size.

Figure 5.4: Investigating the dependence of AdWin Tree on the input value to the K-S test, δ.

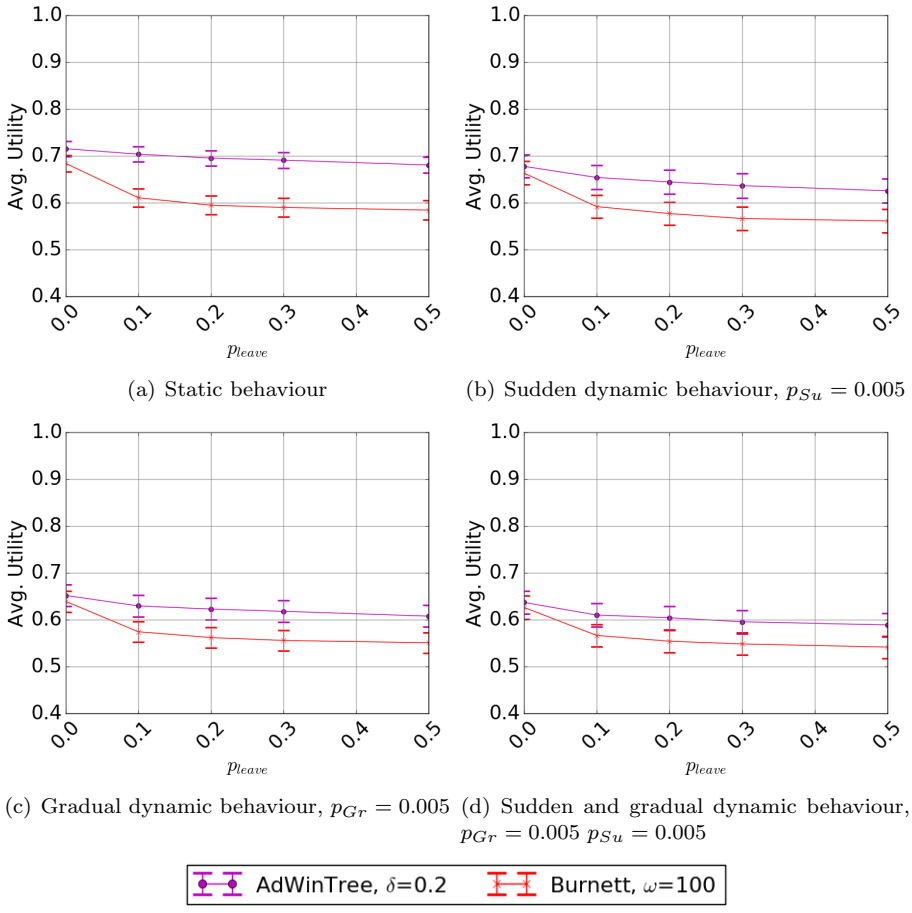
that behaviour change had occurred and continuously remove data until there is fewer than L data. The tree would then rebuild once there is L amount of data, and again immediately detect behaviour change and remove it. This means that the trustor would be continually relying on a small amount of recent data and the trust model, and an *a priori* value of 0.5 for its assessments. One advantage of this approach is that δ of any value does not wholly affect the agents ability to make any behaviour assessment at all, only the amount of data available.

Finally, we can see that as both dynamic behaviour and δ increase, there are occasions where the window size is below L, clearing the stereotype model until more instances can be collected. This is advantageous because the trustor relies on the default *a priori* of 0.5. All unknown trustees are perceived equally and the trustor is forced to interact with them, again reducing class imbalance. A similar effect was observed in the previous chapter, when class imbalance was detected by CIM and so the trustor was forced to use the default *a priori* instead of a stereotype model.

5.4.2 Agent Population

In this section, we consider two aspects of how the agent population can affect trust and reputation assessment in a dynamic environment. Firstly, agents have a probability of leaving the population, known as agent turnover, denoted by the parameter p_{leave}. Agents are immediately replaced to maintain the population size but there is no interaction history with the new agent. Secondly,

we investigate the effects of the graph topology that agents are connected in.



(a) Static behaviour

(b) Sudden dynamic behaviour, $p_{Su} = 0.005$

(c) Gradual dynamic behaviour, $p_{Gr} = 0.005$

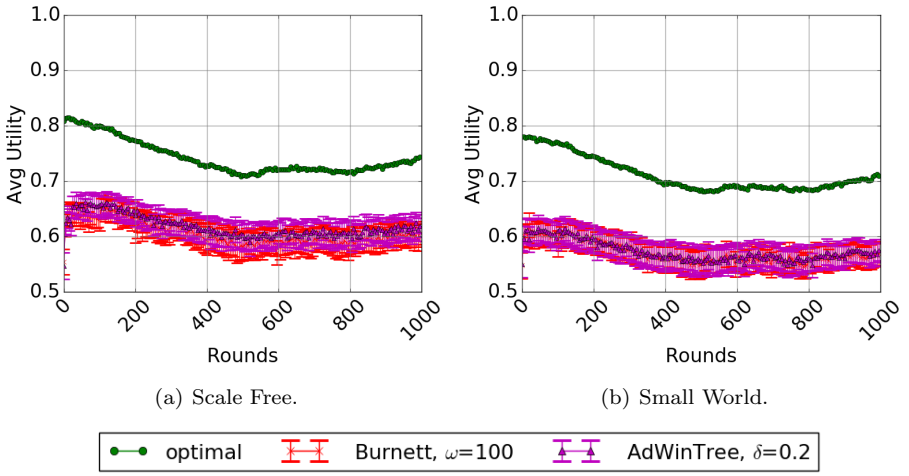
(d) Sudden and gradual dynamic behaviour, $p_{Gr} = 0.005$ $p_{Su} = 0.005$

Figure 5.5: The effect of an increasing dynamic population determined by p_{leave}, on agent performance when agent behaviours are dynamic.

Stereotype models improve an agent's initial assessment of an agent they have never encountered before. It is necessary to evaluate stereotype models in populations where there is a population turnover, because otherwise sufficient information about the agents in the population could be collected for trust and reputation methods to override any stereotypical trust assessment and we would not see the benefit of them. This can be verified in our results from Figure 5.5, when $p_{leave} = 0$ and there is negligible difference in performance between the two models. Figure 5.5(a) shows that when agent behaviour is static, there is an approximately 10% improvement of using AdWin Tree with high levels of turnover. Similarly to results from static behaviour above, AdWin Tree can deal with class imbalance and is a more accurate stereotype model. As dynamic behaviour increases, both models show slightly decreasing performance

as p_{leave} increases. The stereotype models are both learnt with trust values, which become more accurate with repeat interactions. High levels of p_{leave} prevent agents using either model from accurately assessing trust, and therefore prevent a stereotype model from learning accurate correlations between trust and agents' observable features.

Different graph topologies affect how many agents are available to interact with. Most of our results are presented for a fully connected graph because it demonstrates the efficacy of trust and reputation assessment when the trustor has a large choice of partners with sparse data about them. When agents are forced to interact with specific partners because they have a small neighbourhood, as is the case for the majority of agents in scale-free and small-world graphs, it does not matter how effective the trust assessment of them is. AdWin Tree is effective at improving partner selection when trustors face a high amount of uncertainty about their potential partners. When this is not the case, and there is a lot of information about partners, trust and reputation algorithms take precedence to make accurate assessments.



(a) Scale Free.

(b) Small World.

Figure 5.6: Performance of AdWin Tree when the agent population is connected in other graph topologies.

5.4.3 Interaction outcome granularity

Throughout this thesis, we use DRS for trust and reputation assessment. The format of interaction outcomes depends on the application, however, if they are single values, then they can be normalised or binned to be compatible with DRS. DRS has one input parameter, k, defining the number of categories an interaction outcome can fall in. For example, when $k = 2$, interaction outcomes

are perceived as either good or bad. BRS is the specific case of DRS when $k = 2$ and is a common trust algorithm built on by other trust and stereotype algorithms. We present results in Table 5.2 for other values of k to show that our work is applicable, and robust, to the more general DRS. Each cell represents the percentage increase in utility agents receive using AdWin Tree where $\delta = 0.2$ compared to a stereotype model with a fixed size sliding window where $\omega = 100$.

Table 5.2: Percentage improvement of using AdWin Tree over Burnett decision tree stereotype model with increasing values of k in DRS trust assessment method.

		Dynamic Behaviour					
		Static	Gradual		Sudden		Both
		p_{Gr}:0.0 p_{Su} : 0.0	p_{Gr}:0.001 p_{Su} : 0.0	p_{Gr}:0.005 p_{Su} : 0.0	p_{Gr}:0.0 p_{Su} : 0.001	p_{Gr}:0.0 p_{Su} : 0.005	p_{Gr}:0.001 p_{Su} : 0.001
k	2	**7.807 %**	**4.66 %**	5.817 %	**7.36 %**	5.759 %	5.037 %
	3	6.361 %	4.415 %	**6.321 %**	6.734 %	**7.063 %**	**5.59 %**
	4	5.728 %	4.133 %	6.154 %	6.355 %	6.736 %	5.116 %
	5	5.718 %	4.072 %	6.164 %	6.162 %	6.838 %	4.983 %
	10	4.655 %	3.542 %	5.76 %	5.438 %	6.55 %	4.752 %

An interesting trend in these results is that when behaviour is static, a lower value for k is more appropriate. As the level of dynamic behaviour increases, AdWin Tree is better with higher values of k. One explanation for this is identified in Figure 5.7, showing how the variable window changes size. When k is smaller, more data are retained, but as k increases there are fewer data and a smaller standard deviation. This can only have occurred because behaviour change is being detected more frequently with higher values of k, even in static situations. When there are fewer data, smaller values for k make it harder to assess true behaviour values, and so higher values of k will be more accurate.

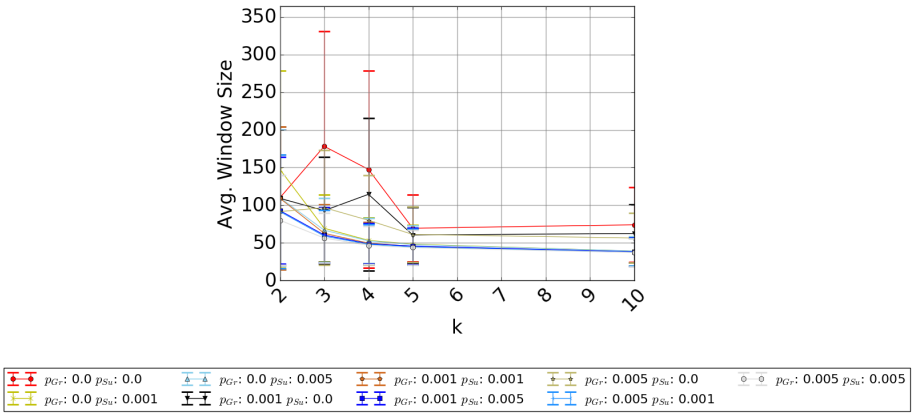
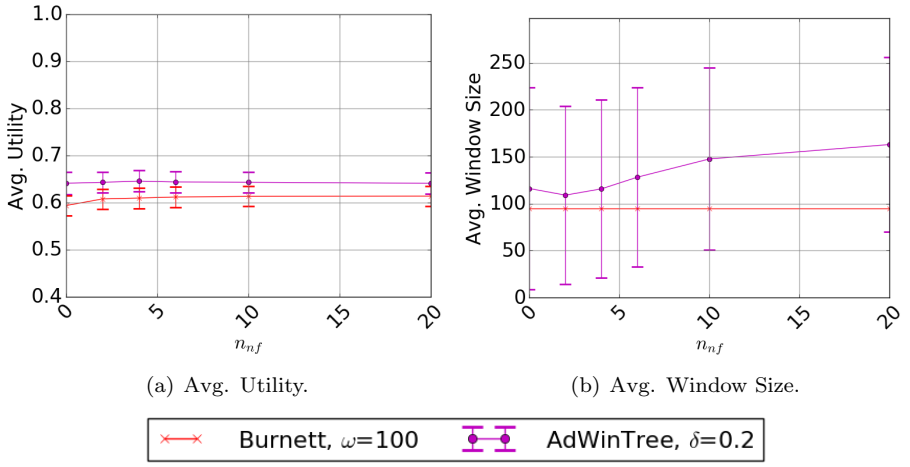


Figure 5.7: Window size changes for different outcome granularities, k.

5.4.4 Noise

A stereotype model correlates agents' observable features with trust. Depending on the application, it is not a realistic assumption that all of an agent's observable features are relevant to their behaviour. Dynamic behaviour means that the class value that the stereotype tree is trying to learn is changing over time, and so identifying true correlations between the feature set and the class is already problematic. Noisy features complicate this task further. When the stereotypes are less accurately identified, there is more noise in the adaptive windows when trying to detect behaviour change in the interaction outcomes from a stereotype.

We have two variables controlling the noise in agents' observable features: the standard deviation in feature values, Θ, and the number of features which do not correlate with agents' behaviour, n_{nf} (in our evaluation agents always have 5 relevant features which do correlate with their behaviour). More detail explaining these variables is provided in Chapter 3. The level of dynamic agent behaviour used when obtaining the results in Figures 5.8, 5.9 and Table 5.3 is $p_{Gr} = 0.001$ and $p_{Su} = 0.001$.

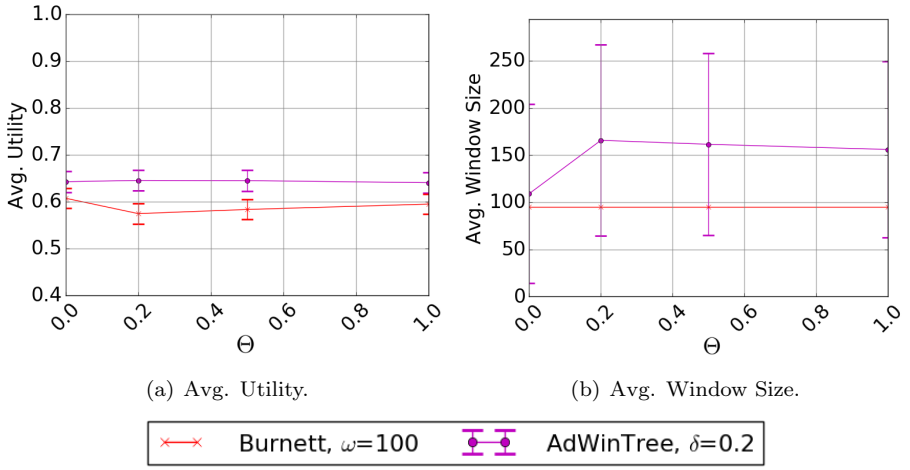


(a) Avg. Utility.

(b) Avg. Window Size.

Figure 5.8: Effect to AdWin Tree of increasing the number of noisy features, n_{nf}, while the noise in the feature values is fixed, $\Theta = 0$.

Figures 5.8(a) and 5.9(a) show that as both n_{nf} and Θ increase respectively, neither model is significantly affected by the added noise. Both the average window size and the standard deviation increase as n_{nf} increases, depicted in Figure 5.8(b). This implies that behaviour change is not as frequently detected but reduces the size of the window by a large proportion when it is detected. Therefore, AdWin Tree is keeping up with crucial behaviour changes and this is reflected by the robust results for average utility in Figure 5.8(a). A small



(a) Avg. Utility. (b) Avg. Window Size.

Figure 5.9: Effect to AdWin Tree of increasing the noise in the feature values, Θ, while the number of noisy features is fixed, $n_{nf} = 2$.

increase in Θ causes a big jump in the average window size of AdWin Tree, but again, this does not negatively impact on performance.

Finally, the increased performance of AdWin Tree over a decision tree stereotype model with a fixed size sliding window in varying degrees of noise is presented in Table 5.3. Looking down the columns, we can see the added benefit of the adaptive window decreases as n_{nf} increases (also seen in Figure 5.8(a), where the average utility converges). The change in Θ, going across the rows, shows an unusual pattern. From Figure 5.9(a), we can see the original stereotype model from Burnett *et al.* dips in performance with a small increase in Θ. This seems to be an anomaly however, as the results revert as Θ increases further. Overall, both models are resilient to a small amount of features variation.

Table 5.3: Percentage improvement of using AdWin Tree over Decision Tree Stereotype model with varying noise.

	Θ			
n_{nf}	0.0	0.2	0.5	1.0
0	7.98%	13.141%	11.156%	8.15%
2	5.798%	12.312%	10.541%	7.673%
4	5.889%	12.054%	10.34%	7.612%
6	5.223%	11.692%	9.521%	7.016%
10	4.873%	9.23%	7.655%	5.447%
20	4.432%	8.794%	6.809%	4.615%

If the trustor could immediately identify the true profile, or group, of a trustee, without the need to learn a stereotype model, an adaptive window could monitor the interaction outcomes from a group of agents with no noise.

We explore this in the next chapter, as well as how to predict changes in group behaviour.

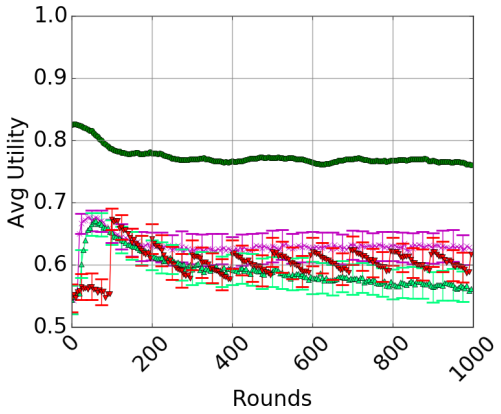
5.4.5 Addressing class imbalance and dynamic behaviour

Section 5.3 described how the class imbalance tool we presented in the previous chapter can be integrated with AdWin Tree. We evaluate if the CIM-AdWin Tree method is robust against both dynamic behaviour and class imbalance. Recall that CIM-CI denotes using just the class imbalance component of CIM, and CIM-CIDB is when CIM also accounts for dynamic behaviour. When using the two methods together, we combine only CIM-CI with AdWin Tree, because AdWin Tree handles dynamic behaviour. This is then compared against AdWin Tree alone, and CIM-CIDB alone.

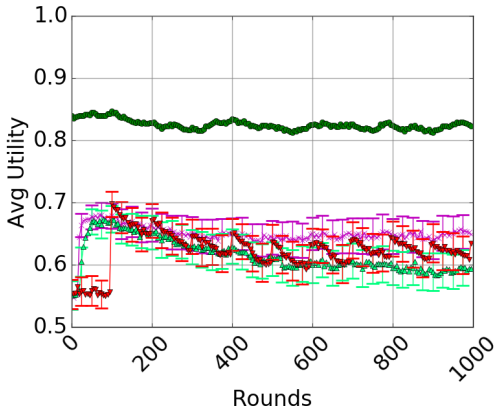
Figures 5.10(a) and 5.10(b) demonstrate that in the context of both sudden and gradual agent behaviour change, AdWin Tree is more successful than when CIM works in conjunction with either AdWin Tree or the original stereotype model. One reason for this is that dynamic agent behaviour reduces the class imbalance problem because agents do not choose to have repeat interactions with the same subset of agents as they do when behaviour is static. Agents learn with the trust assessment model that partners which they used to trust are no longer trustworthy, which encourages them to choose different partners in the future. Without class imbalance, AdWin Tree is much more flexible at handling dynamic behaviour and therefore outperforms CIM. CIM works effectively with the original stereotype model of Burnett *et al.* [Burnett et al., 2010] when it is first built after every L interactions, but CIM uses a forgetting factor which assumes that behaviour has changed unless new information prevents older data from being forgotten. Similarly to sliding windows, the forgetting factor is an inflexible method to manage agents' data.

When behaviour is static though, Figure 5.10(d) shows that CIM on its own performs better than AdWin Tree. When behaviour is static, the class imbalance problem is much more prominent, and AdWin Tree does not address this as well as CIM.

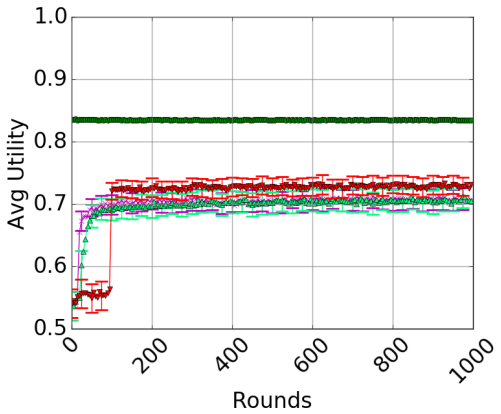
Finally, we can see in Figure 5.10 that AdWin Tree does not perform as well as AdWin-CIM. This is because CIM will sometimes not send data to the tree when it believes there is no other representative data for the interaction partner. This prevents discovering changes in any profile which is not the profile most interacted with. This can be effective in a static setting, but not in a dynamic one. Additionally, the outliers get put in the outlier window, which then gets deleted, whereas AdWin Tree holds on to them to allow us to distinguish between them later on. The outliers are also not being monitored for behaviour



(a) Avg. Utility, gradual dynamic behaviour.



(b) Avg. Utility, sudden dynamic behaviour.



(c) Avg. Utility, static behaviour.

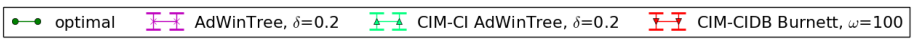


Figure 5.10: Comparing the performance of AdWin, CIM and AdWin-CIM.

change.

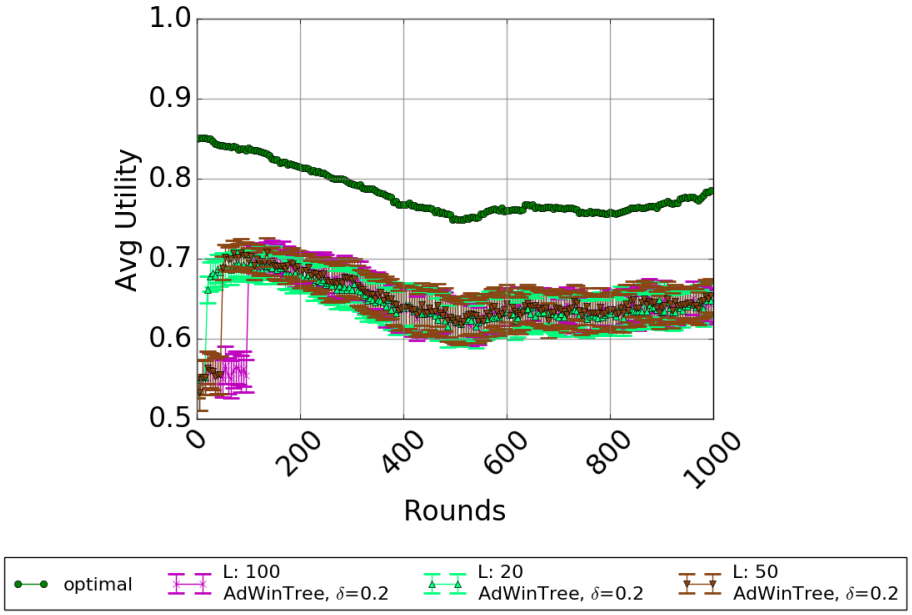


Figure 5.11: Sensitivity of AdWin Tree to learning interval, L.

For AdWin Tree, L is the minimum amount of data required to initialise the stereotype model when it currently does not exist, either because the agent is new or because there are insufficient data from when dynamic behaviour was previously detected and handled. Once sufficient data are available the tree is rebuilt after dynamic behaviour is detected and handled. Larger values of L only delay the stereotype model being initially built, and do not impact on subsequent performance, as seen in Figure 5.11. Therefore, trustors initialise AdWin Tree after 20 interactions. We do not present results for AdWin working in conjunction with Direct Comparative Stereotypes (DCS) because AdWin Tree rarely deletes so many records that the remaining amount of data falls below this value of L and, therefore, DCS is not used.

5.5 Discussion

In this chapter, we presented AdWin Tree, a method to address RO 3, where agents need to make trust and stereotype assessments representative of current agent behaviours, given that these may change over time. The results show that agents using AdWin Tree can more accurately filter out data which no longer represent agents' behaviour compared to using a fixed size sliding window. Fixed window sizes do not account for the possibility that agents will change

their behaviour at different times and speeds, and so a single window size is not applicable to all agents. We demonstrated that in static contexts, AdWin Tree outperforms a fixed size sliding window, because it only removes data if it determines that there has been a change, thereby retaining old but still useful information to distinguish between trustworthy and untrustworthy agents.

AdWin Tree requires that input parameter, δ, which sets the level of confidence necessary to believe there has been a change in the distribution generating the data, indicating there has been a behaviour change. We presented results across a range of values for δ, demonstrating there was little sensitivity to its value.

We evaluated whether AdWin Tree could work in conjunction with CIM to explicitly address both class imbalance and dynamic behaviour and we demonstrated that dynamic behaviour reduces the class imbalance problem. When agent behaviours are static, and the class imbalance problem is more prominent, CIM is the best performing, while if agent behaviours are dynamic AdWin Tree performs best and is effective without CIM. Knowing which approach to take might depend on knowing whether dynamic agent behaviours exist in the environment in advance.

The complexity of AdWin Tree as described in this thesis is $O(n^2)$ because the AdWin algorithm requires splitting the window into two for every combination, and then applying the KS test to that split which is also an $O(n)$ algorithm. However, there an optimisation of the AdWin algorithm to require fewer splits of the window and less frequent tests exist, called AdWin 2 [Bifet and Gavaldà, 2009], which if implemented will reduce the complexity of the AdWin Tree algorithm to $O(nlog(n))$[4]. Increasing dynamic behaviour may speed up behaviour assessment with AdWin Tree because it is dependent on the number of interaction histories it has and AdWin Tree can detect dynamic behaviour and reduce the number of instances. This will especially speed up the reputation process of searching through interaction histories for reputation data and distributing it.

The results showed that the most significant limitation of our work is that it does not offer significant improvements in small-world or scale-free network topologies. While the fully connected graph does not necessarily represent a single example network, we use it to understand how trustors behave when they have sparse data and a large choice of interaction partners. In this instance, AdWin Tree is very successful.

Another limitation of AdWin Tree is there exists a risk that some instances of data will never be deleted. If behaviour change is never detected, potentially a result of not finding accurate stereotypes, or because behaviour is static and

[4]AdWin 1 algorithm was described in this thesis because its simplicity provides a good illustration of the concept drift process.

there is no change to detect, the size of the set of an agent's past experiences will continue to grow. Agents might not have the memory capacity to accommodate this growing set, and the time required to retrieve reputation information will increase. In future work, we might consider enforcing a maximum time limit on data to prevent old information from never being forgotten. This value would only need to be a conservative estimate of how long data are relevant for, as it is unlikely the information has not been removed already, and is therefore unlikely to negatively affect performance. However, it could prevent a build up of noise over a long period of time. One limitation of all existing trust and reputation models, including AdWin Tree, is that once data are forgotten they cannot be reused. We address this in the next chapter, where agents store a summary of old instances that have been deleted, and may use that summary to learn and predict behavioural patterns.

We exploit the stereotype assumption that agents who look the same act the same, however this might not always be the case. Devising a flexible approach to manage information about specific agents instead of groups should be considered in future work. When tracking individual agents, the data can be even more sparse, as we cannot use data from a group of agents to infer the behaviour of one. However, we believe our approach to statistically assess the representative information to use in trust assessment could inspire a solution to that problem. In some contexts, trustors have repeat interactions with favoured trustees, investigating whether agents can monitor those agents individually, and others with our group method, could be beneficial. Another avenue of future work is to detect anomalous agents who are behaving maliciously, and are attacking the system in some way. This could include identifying agents who abuse stereotypical trust by impersonating a particular stereotype by monitoring how their individual level trust deviates from the stereotypical trust.

This chapter aims to show that by adapting the data and methods to the environment or situation, trust, reputation and stereotype assessment models can be improved. Both CIM and AdWin Tree have demonstrated that trustors who can manage information about the agents in their neighbourhood more flexibly are more effective.

CHAPTER 6
Behavioural Patterns

Chapters 4 and 5 demonstrated the importance of using only representative interaction experiences in trust assessment, and proposed methods for selecting appropriate interactions. One limitation of the approaches proposed is they assume only the most recent interactions are representative. However, agents may change their behaviour regularly, or fluctuate their behaviour in a recognisable pattern. In this chapter, we introduce *R*eacting *a*nd *P*redicting *T*rust *a*nd *R*eputation (RaPTaR), a method which estimates the current agent behaviour by both predicting and reacting to behaviour change. In this chapter, we focus on learning complex agent behaviours for groups when those groups are immediately identified. In this chapter, we assume that agents have stereotypical behaviour, such that agents of a group behave the same way, and that these groups are known in advance. This differs from the assumptions made in Chapters 4 and 5 where groups were identified with stereotype functions, however coping with the high level of noise and inaccuracies this can lead to is future work here. Instead, we focus on developing techniques to analyse agent behaviours and exploit as much information about known groups as possible.

6.1 Introduction

An agent is characterised by its autonomy, social ability, reactivity and proactiveness [Wooldridge and Jennings, 1995], however most trust techniques focus on reacting to an agent's perceived behaviour without considering how to be proactive. This chapter discusses how agents can be more proactive in their assessment of other agents behaviour and proposes a method to predict changes in dynamic agent behaviour.

In previous chapters we discussed how dynamic behaviour in agents can be context dependent, however we only considered randomly changing behaviour. In this chapter, we specifically consider how dynamic behaviours can follow a particular pattern, and attempt to exploit knowledge learned about those patterns. Examples of where patterned agent behaviour could be exploited include demand shifts in the smart grid at specific times of day and seasonal changes [Tso and Yau, 2007], reduced response from network traffic at peak times [Messina et al., 2017], oscillating behaviour to maliciously abuse good reputations [Hales and Edmonds, 2003; Salehi-Abari and White, 2012; Srivasta et al., 2005] and external, possibly unseen, variable changes triggering specific

agent behaviour responses [Harries et al., 1998]. Additionally, agent behaviours may change as a result of being in a group [Griffiths, 2006; Nguyen and Bai, 2018]. Agents must sometimes delegate subtasks to other members of their group, but their ability to complete the task will reflect on the delegating agent. Therefore, an individual agent's trust may reflect the group's abilities. As groups or coalitions can form dynamically, but also be subject to fluctuating motivations, available resources, a changing global population, or external factors, the trustworthiness of a group and its members can fluctuate [Nguyen, 2017]. A malicious cause of fluctuating behaviour is the oscillating attack, where agents build up a good reputation for a time and then exploit their good reputation until they are no longer trusted, and the process repeats [Srivasta et al., 2005; Yao et al., 2012]. One approach towards overcoming the oscillating attack is for agents to gain trust slowly but lose it quickly. However, the current trust value then does not necessarily reflect an agent's true behaviour. This can be unfair towards agents whose fluctuating behaviour is not malicious, and can prevent fruitful interactions [Liang and Shi, 2005].

In this chapter, we propose RaPTaR, a method which sits above existing trust and reputation methods to detect and learn patterns of behaviour change. By adjusting to those fluctuations in behaviour, RaPTaR can supply the trust and reputation algorithm with only relevant data. Building on the work from Chapter 5, the point of behaviour change is identified by monitoring a variable size window of outcomes from a group of agents, using the Kolmogorov-Smirnov (K-S) statistical test. Transitions between behaviour changes are then recorded to learn patterns in group behaviour for future exploitation. RaPTaR also provides an initial assessment of an agent belonging to a group for any trust assessment model which uses an *a priori* trust estimate, based on both recent behaviour and any learnt behavioural patterns of the group.

RaPTaR offers several contributions to improving trust and reputation assessment in groups. First, RaPTaR reacts to any statistically detected changes without requiring the tuning of parameters in advance. This allows agents to adapt to their situation, which can vary across MAS. Second, inspired by the Reactive Proactive (RePro) concept drift algorithm, agents can learn patterns in behaviour such that future changes can be predicted or expected. An overview of concept drift literature was provided in Chapter 2, and other concept drift algorithms were used in a similar way in the previous chapter. Learning another agent's past behaviour as a summary instead of maintaining all the past interaction records allows an agent to retain useful information while incurring a low memory overhead. We store meta-information about behaviour summaries to help predict whether they will arise again. Third, RaPTaR calculates expected behaviour considering all known behaviours weighted by the probability that

they are active, which addresses a limitation of RePro which can only predict the single most likely behaviour. Fourth, RaPTaR learns how long agents spend acting with a particular behaviour, with the intuition that if the behaviour occurs again, it may have a similar duration, and therefore a change in behaviour can be anticipated. Finally, RaPTaR includes a probability that the agent will switch to an unknown behaviour.

We compare our work to techniques commonly used by trust and reputation assessment models to account for changes in behaviour, namely, sliding windows and forgetting factors. Additionally, we show an analysis of different granularities of outcomes which can be used in DRS, as described in Chapter 3.

6.2 RaPTaR

In this section, we present RaPTaR, which improves existing trust assessment methods in two ways. First, statistically detecting changes in recent outcomes from interactions, and updating the data used in trust assessment accordingly, enables agents to assess trust appropriately for others' behaviour. Second, RaPTaR produces an estimate of an agent's behaviour based on its group by exploiting any learned patterns. An agent uses RaPTaR to monitor the outcomes from a group, G, of agents, $G \subset \mathcal{A}$, where $\mathcal{A}$ is the set of all agents. This accounts for how a group may change its behaviour at different times and speeds compared to other groups. Each identifiable group in the population is monitored separately. Groups may represent some known coalition, however, if groups are not explicitly identifiable, it can be possible to group agents using a stereotype model, as in the previous chapters. An agent stores the outcome of its interactions in a history, $\mathcal{O}$, which is divided into subsets representing the outcomes of interactions with agents from each group they have encountered, $\mathcal{O}_G \subset \mathcal{O}$. RaPTaR can then monitor $\mathcal{O}_G$ for behavioural changes within each group G. In algorithms 6 and 7 we refer to $RaPTaR_G : \langle W, TM \rangle$, where each group is monitored by a RaPTaR model, and that RaPTaR model contains instances of two variables, TM and W. These two variables contain the necessary information for RaPTaR to assess that group's behaviour, and this process is described in detail below. We introduce the notation in Table 6.1 to describe RaPTaR.

RaPTaR has a learning component and a predictive component. When an interaction occurs with a partner, the trustor affects the RaPTaR model for the group that partner belongs to, $RaPTaR_G : \langle TM, W \rangle$. From now on, we refer to those variables as just TM and W, using the assumption that the group has been identified. When the outcome of an interaction is recorded, it will either be assumed to come from the same distribution as other recent outcomes, or from a different distribution if a change in behaviour is detected, which is built

Table 6.1: RaPTaR notation

Notation	Description
$G \subset \mathcal{A}$	An identifiable group of agents assumed to behave similarly, and therefore be monitored using RaPTaR.
$o_t \in [0, 1]$	The outcome of an interaction which occurs between a trustor, tr, and trustee, te, at time t.
W	An adaptive window of recent interaction outcomes that a trustor has had with a group of trustees.
$t_{W[x]}$	The time that the interaction indexed at x in window W occurred.
δ	Input parameter to set the sensitivity of the K-S test used to detect changes in distributions over time, described in Section 6.2.1. The confidence in the decision made by the K-S test is $1 - \delta$.
$\overrightarrow{C}$	A history of learned concepts
$c \in \overrightarrow{C}$	A concept in the list of known concepts for G which have been learnt from interactions with members of G. A concept is a learned estimate of behaviour that was believed to be active for a period of time.
$c_c \in \overrightarrow{C}$	The currently active concept which estimates the behaviour that produced the most recent interaction outcomes currently in W. This might be a concept we already know exists from the concept history if such a concept exists that predicts W well, or a new concept otherwise.
cet	The conceptual equivalence threshold, used to determine if the values in W are similar enough to an already known concept in the concept history.
TM	A transition matrix to maintain statistics about behaviour changes, recorded as concept transitions, which are used for future predictions of agent behaviour.
$c_x, c_y \in \overrightarrow{C}$	Variables to keep track of the two concepts preceding c_c in order to record transition $c_x \rightarrow c_y$ in TM
L_{c_x,c_y}	A list, indexed at $TM[c_x, c_y]$, containing the durations of time concept c_x was active for before transitioning to the next concept, c_y. Such a list exists in every cell of the TM, for every possible transition of the known concepts and may be empty for transitions that have never occurred.
tl_{c_x}	The length of time c_x is believed to have been active before transitioning to c_y, a value stored in the list L_{c_x,c_y}.

on the behaviour change detection method proposed in Chapter 5. For the learning component we use a modified version of the AdWin algorithm [Bifet and Gavaldà, 2007] to detect changes and remove irrelevant data. If a change is detected, the behaviours on either side of the change are learned. Agents record how long a behaviour was believed to be active and which behaviour succeeded it, to learn behavioural patterns and improve predictions of future behaviour changes. Many existing trust and reputation algorithms make use of an *a priori* trust value as input to improve trust estimates when there are few experience data. Similar to stereotype models, the predictive component of RaPTaR produces an *a priori* estimate based on experiences from members

of the group, who are assumed to behave similarly. RaPTaR uses the adaptive window of recent interactions to estimate the current behaviour, how long it has been active, and possible successor behaviours to assess an overall expected utility. We introduce the learning and predictive components of RaPTaR in Sections 6.2.1 and 6.2.2 respectively.

6.2.1 Learning Component

The learning component comprises two parts: the first detects changes in agent behaviour using interaction outcomes, and the second learns patterns in changes of behaviour. RaPTaR's behaviour change detection module is taken from the work presented in Chapter 5, specifically described in Section 5.2.1. Interaction outcomes from each group are monitored in a variable size window W for change using the K-S test. However, RaPTaR handles the data differently once drift is detected.

If there exists a split of W into W_0 and W_1 such that a change in behaviour is detected, RaPTaR stores information about the change for future predictions about behaviour. When a change is detected, the instances in W_0 can be learnt as a concept. The currently active concept, c_c, which best estimates W_1, is still active and may still change, so we cannot record a transition to it yet. Therefore, transitions are recorded between the two concepts which were active prior to c_c. Concept c_y represents the behaviour in W_0, and c_x was the concept learned before c_y, learned from data which have since been removed from W. Concept c_x is initialised to the *unknown concept*, and then updated at the end of this process as $c_x \leftarrow c_y$, ready to record the next transition. The form of the transition matrix is depicted in Table 6.2. The first column of TM, index 0, is for the *unknown* concept, to show how frequently other concepts are followed by a new concept. When a change in behaviour is detected, as described above, the concept c_y which describes the behaviour in W_0 needs to be identified as either a concept seen before, or be learnt as a new concept. The value of all the known concepts in $\overrightarrow{C}$, is compared to the mean of the instances in W_0, μ_{W_0}, and if there exists a concept $c \in \overrightarrow{C}$ such that $|\mu_{W_0} - \mu_c| < cet$, where *cet* is the *conceptual equivalence threshold*, then $c_y \leftarrow c$. The value of *cet* enforces the maximum number of concepts there can be to $\frac{1}{cet}$, therefore also setting the size of the transition matrix. This means the time complexity of searching the transition matrix is linear. If no concept is equivalent, a new concept is learnt with the value μ_{W_0}. If the length of W_0 is less than a stable learning size, s, then it is not considered substantial to record a transition for. The previous concept, c_x, is remembered even though the data for it has since been deleted, and we can now record the transition between c_x and the

newly learned c_y. The list, L_{c_x,c_y}, records the length of time spent at c_x, tl_{cx}, before moving to c_y, for every occurrence of the transition. The time at c_x is initialised to 0 for the first recording. Algorithm 7 describes the reactive and learning components together. As described in the RePro algorithm [Wu and Zhu, 2005], the transition matrix follows the Markov assumption, who's general definition is that a state is dependent on a previous state before it. In this case, a cell in our transition matrix indicates that the probability of a concept becoming active is dependent on whether or not a particular other concept is active right now. As the matrix is 2-dimensional, it only considers one concept previous, when determining the probability a concept follows it. The matrix does not consider whether the concept that was active before that one, or the one before that one, may effect the probability of which concept will become active next[1].

Table 6.2: Transition Matrix.

Previous Concept, c_x	Successor Concept, c_y				
	unknown	1	2	3	...
1		$L_{1,1}$			...
2					...
3					...
...					

Algorithm 6 Overview of Assessing Agents with the Predictive Component of RaPTaR.

```
1: function SIMULATION
2:     for t ∈ T do
3:         for tr ∈ A_tr do
4:             partner ← te|max_{te∈A_te} Behaviour_Assessment(te, O_tr)
5:             o^t ← Interaction(partner)        ▷ depends on partner's behaviour
6:             Learning_Component(partner, o^t)        ▷ RaPTaR updates and learns
7:         end for
8:     end for
9: end function
10: function BEHAVIOUR_ASSESSMENT(te,O_tr)
11:     a ← Predictive_Component(te) ▷ RaPTaR estimates group's behaviour
12:     trust ← trust_alg(O_tr, te, a)
13:     return trust
14: end function
```

[1]This is listed as future work in the original literature [Wu and Zhu, 2005], and could be applied here too.

Algorithm 7 Learning Concepts by Updating TM.

function LEARNING_COMPONENT(te, o^t_{te})
  $G \leftarrow te_G$ ▷ Identify the group te belongs to
  W, $TM \leftarrow RaPTaR_G\langle W, TM\rangle$ ▷ Update W and TM associated with the RaPTaR model for that group
  $W \leftarrow W + o^t_{te}$
  for $(i = |W| - s; i > s; i - 1)$ **do** ▷ iterates backwards through W ▷ where s is the stable learning size
    $W_0 \leftarrow W[0, i]$
    $W_1 \leftarrow W[i, |W|]$
    if $K - STest(W_0, W_1) < \delta$ **then**
      $c_y \leftarrow LearnConcept(W_0)$
      $TM[c_x, c_y, L] \leftarrow TM[c_x, c_y, L + \{tl_{cx}\}]$ ▷ c_x initiated to unknown concept ▷ tl_{cx} initiated to 0
      $c_x \leftarrow c_y$
      $tl_{cx} \leftarrow t_{W_0[|W_0|]} - t_{W_0[0]}$ ▷ Length of time c_x was active
      $W \leftarrow W_1$ ▷ Shrink window to recent relevant data
      return
    end if
  end for
end function
function LEARNCONCEPT(W_0)
  $c_y \leftarrow null$
  $\mu_{W_0} \leftarrow avg(W_0)$
  for $c \in \overrightarrow{C}$ **do**
    if $|\mu_c - \mu_{W_0}| < cet$ **then**
      $c_y \leftarrow c$
    end if
  end for
  if $c_y == null$ **then** ▷ new concept
    $TM[c_x, 0, L] \leftarrow TM[c_x, 0, L + \{tl_{cx}\}]$ ▷ unknown concept is indexed at 0 in TM
    $c_y \leftarrow |TM| + 1$
    $\mu_{c_y} \leftarrow \mu_{W_0}$
    expand TM to include new concept
  end if
  return c_y ▷ After return, transition recorded as normal
end function

6.2.2 Predictive Component

RaPTaR calculates an expected utility (EU) which can be used as an *a priori* for any trust and reputation assessment model as seen in lines 11 and 12 of Algorithm 6. The RaPTaR model used in this description is associated with the group that te is believed to belong to. The group is identified in the same way as in the learning component, when the correct RaPTaR model for a partner's group needs to be identified to be updated with an interaction outcome. RaPTaR calculates that EU based on the value of each concept, $\mu_c \in \overrightarrow{C}$, in the concept history for a group, and the probability that it may currently be active, $p(c)$. The EU draws upon the information recorded in TM by the learning component and is calculated as:

$$EU = \forall_{c \in \overrightarrow{C}} \quad \mu_c \times p(c) \tag{6.1}$$

First, tr identifies whether any of the known concepts, $c \in \overrightarrow{C}$, are likely to be the currently active concept, c_c, by evaluating the difference between the concept values, μ_c, and the mean of the values currently in the window, μ_W:

$$|\mu_{c_c} - \mu_W| < cet$$

The trustor assesses the length of time this concept has likely been active, t_{c_c}, as the current time, t, minus the time the first outcome in W was recorded:

$$t_{c_c} = t - t_{W[0]}$$

The first known occurrence of the currently active concept will be the first element of the adaptive window, $W[0]$, because the reactive component previously assessed a behaviour change which occurred between an element in the window that was deleted, and this element.

The next step is to calculate the probability, $p(c)$, that each concept $c \in \overrightarrow{C}$ will succeed c_c given that c_c is assumed to have been active for t_{c_c}. A probability density function (PDF) is built with a Gaussian Kernel Density Estimator [Parzen, 1962] using the lengths of time spent at c_c in the past before it was succeeded by c, i.e., $\forall tl_{c_c,c} \in L_{c_c,c}$. A PDF estimates the probability of all time lengths the concept may be active for given the data in $L_{c_c,c}$. This is advantageous because the reactive component may not identify the time of change perfectly, or infers probabilities for times which were not in the data set but only because we did not see that exact length of time. For example, if the concept was transitioned to several times after 100 times steps, and several times after 110 time steps, it is reasonable to assume the probability of the transition

occurring between these two times is also high. As $|L_{c_c,c}|$ increases, the PDF will become more accurate at estimating the time. The raw probability, $\overline{p(c)}$, that c succeeds c_c after t_{c_c} time at c_c is:

$$\overline{p(c)} = PDF(t_{c_c}) \times |L_{c_c,c}|$$

where $|L_{c_c,c}|$ is the length of the list, and therefore the number of times c has succeeded c_c. Multiplying by the frequency of transitions turns the probability into a frequency distribution estimate, which will give precedence to the concepts which more frequently succeed c_c when the probabilities are normalised.

To normalise the probabilities of each possible next concept (including the possibility that c_c is followed by c_c at this time), the probabilities must sum to one, $\sum_{c \in \overrightarrow{C}} p(c) = 1$. Therefore, $p(c)$ is given as:

$$p(c) = \frac{\overline{p(c)}}{\sum_{c_n \in \overrightarrow{C}} \overline{p(c_n)}} \tag{6.2}$$

If no known concept is active, the currently active concept is assumed to be new and it takes on the value μ_W, and probability 1. No other concept has a probability of being active as we have no information about what or when another behaviour might follow the currently active concept.

6.2.3 Integrating RaPTaR with trust and reputation models

RaPTaR has two points of integration with trust and reputation assessment algorithms. First, RaPTaR detects changes in the behaviour of agents in a group, G, and deletes the interaction records believed to be irrelevant from $\mathcal{O}$. This enables trust and reputation to draw upon records in $\mathcal{O}$ which are believed to be representative of current behaviours. This contrasts with use of a fixed size sliding window that removes the oldest interaction record to accommodate for new interaction records, regardless of the relevance of either the newer or older record. Second, RaPTaR outputs an EU which can be used as an *a priori* estimate to trust and reputation models which can be used when there are no, or few, direct experiences with an agent on which to otherwise base a decision.

6.3 Evaluation

Chapter 3 described the evaluation environment used in this thesis, with a list of the parameters and their possible values in Table 3.2. In this chapter, we also use variable δ, which affects the confidence in the K-S test, first introduced

in the previous chapter. All parameters are explored to understand RaPTaR's dependency on them. While we discuss results for different parameter values, unless otherwise stated, we use the DRS trust assessment model where $k = 5$, and a fully connected graph. All the results presented in this chapter are statistically significant using a paired t-test where $p < 0.001$.

Table 6.3 gives the overall performance of RaPTaR with varying values of δ, compared to sliding windows of varying sizes, ω, and forgetting models with varying factors, λ. The cell values represent the average utility an agent receives per timestep and its standard deviation is given in brackets. All values of δ used in RaPTaR outperform any fixed window size or forgetting factor rate. There is a statistically significant 2-5% improvement from using RaPTaR compared to other models. We examine below how RaPTaR offers much larger improvements during short time intervals of behaviour change. All models perform equally during periods of static behaviour because existing work which is not reactive has sufficient time to adjust. The result is small and robust average improvements over 100 runs showing that RaPTaR consistently improves on existing models. Finally, different δ values perform similarly, and we show that RaPTaR's performance is not sensitive to δ. Different window sizes or forgetting factors can vary more in their performance, so these approaches are more dependent on selecting a good value in advance, which may not be known.

Table 6.3 shows the performance of RaPTaR across different graph topologies. There is less improvement in a scale-free network, where agents have a very low average neighbourhood degree, as seen in Table 3.1, and therefore agents have very few neighbours to choose from. Agents are forced to interact with one of their few neighbours regardless of that partner's behaviour, and therefore also regardless of how accurately the trustor assesses trust. Consequently, in the remainder of this section we focus on fully connected graphs, to emphasise how agents behave when they have a reasonable choice of partner and sparse data about those agents because there are too many to have interacted with and to fully understand. This situation demonstrates the efficacy of the behaviour assessment model.

To visualise why RaPTaR outperforms other models, we analyse an example run. Figure 6.1 shows the average trust estimate of agents from each group at every time point. RaPTaR adjusts to behaviour changes more quickly, and makes more accurate trust assessments of all the groups and not just the group with the best behaviour, compared to a sliding window of a fixed size. Figure 6.2(a) depicts the average outcome that agents receive in this example. At the time intervals 400-550 and 600-750 there are substantial improvements from using RaPTaR. This is not a constant improvement at every time point, because during static periods of behaviour, the sliding window eventually tends towards

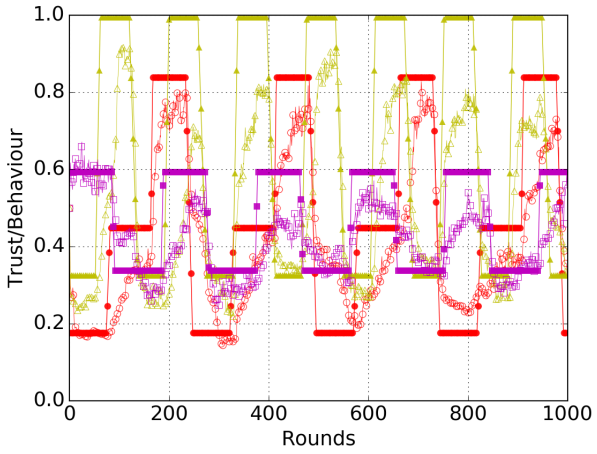
Table 6.3: Comparing RePro with existing methods in 3 graph topologies.

Network Type		Scale Free		Small World		Fully Connected	
$\|\mathcal{A}\|$		100	200	100	196	20	40
RaPTaR, δ	0.05	0.568 (0.009)	0.567 (0.006)	0.593 (0.009)	0.594 (0.006)	0.615 (0.023)	0.614 (0.015)
	0.1	0.569 (0.009)	0.568 (0.006)	0.595 (0.009)	0.595 (0.006)	0.618 (0.023)	0.617 (0.016)
	0.2	0.571 (0.009)	0.57 (0.006)	0.597 (0.009)	0.598 (0.006)	0.622 (0.023)	0.621 (0.015)
	0.3	0.572 (0.009)	0.571 (0.006)	0.599 (0.009)	0.6 (0.006)	0.623 (0.023)	0.623 (0.016)
	0.4	0.573 (0.009)	0.571 (0.006)	0.6 (0.009)	0.601 (0.006)	0.625 (0.023)	0.625 (0.015)
	0.5	0.573 (0.009)	**0.572** (0.006)	0.6 (0.009)	0.601 (0.006)	0.625 (0.023)	0.625 (0.016)
	0.7	**0.574** (0.009)	**0.572** (0.006)	**0.601** (0.009)	**0.602** (0.006)	**0.627** (0.023)	**0.626** (0.016)
	0.9	**0.574** (0.009)	**0.572** (0.006)	**0.601** (0.009)	**0.602** (0.006)	**0.627** (0.023)	**0.626** (0.016)
Fixed Window, ω	50	0.55 (0.009)	0.55 (0.006)	0.569 (0.009)	0.57 (0.006)	0.587 (0.024)	0.586 (0.016)
	100	0.541 (0.009)	0.541 (0.006)	0.557 (0.009)	0.557 (0.007)	0.572 (0.024)	0.572 (0.017)
	200	0.543 (0.009)	0.543 (0.006)	0.559 (0.01)	0.56 (0.007)	0.574 (0.025)	0.574 (0.017)
Forgetting Factor, λ	0.9	0.539 (0.009)	0.539 (0.006)	0.552 (0.009)	0.553 (0.006)	0.568 (0.023)	0.567 (0.016)
	0.95	0.537 (0.009)	0.537 (0.006)	0.55 (0.009)	0.551 (0.006)	0.565 (0.024)	0.564 (0.016)
	1.0	0.543 (0.009)	0.543 (0.006)	0.559 (0.01)	0.56 (0.007)	0.574 (0.025)	0.574 (0.017)

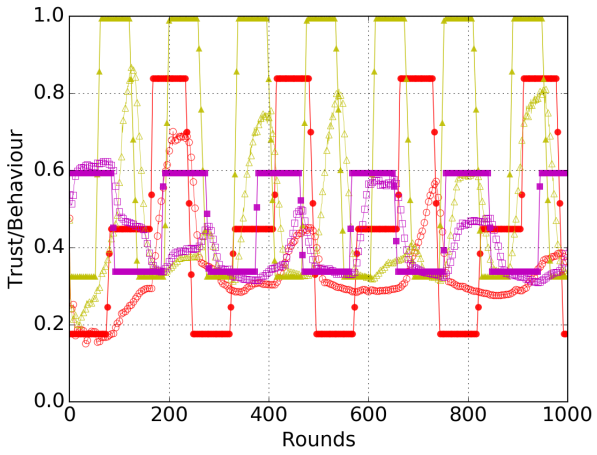
accuracy. However, as soon as the dynamic behaviour of the groups causes the best group to change, RaPTaR's ability to adapt to, and predict, behaviour changes helps agents select a better partner sooner. This analysis can be verified by observing the accuracy of the trust estimates for those time intervals in Figure 6.1.

The example given in Figures 6.1 and 6.2(a) only show how RaPTaR is effective in one example however, agent behaviour can change at different times and speeds. Therefore, Figure 6.2(b) verifies that RaPTaR consistently performs 2-5% better than a fixed window over 100 runs, which is consistent with the results we observed in Table 6.3. Overall, we can see how RaPTaR provides agents with sporadic, large improvements in certain situations, but is also robust in the long term.

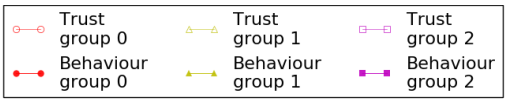
Throughout this thesis, we have demonstrated that the accuracy of the trust



(a) RaPTaR $\delta = 0.2$.



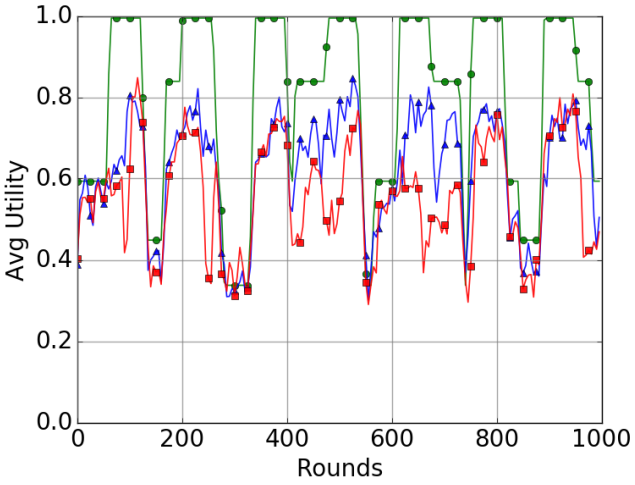
(b) Fixed Window $\omega = 50$.



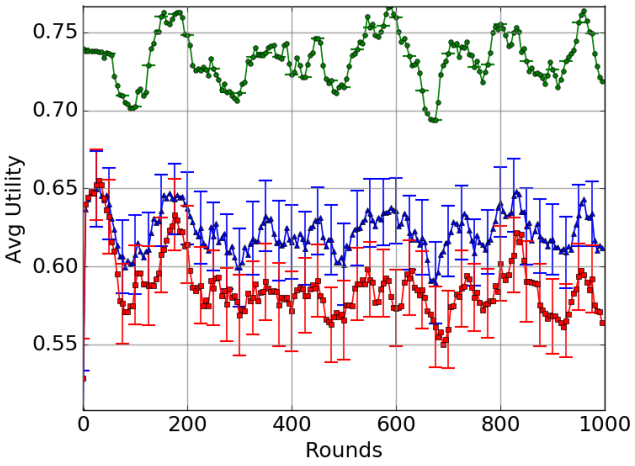
(c) Legend.

Figure 6.1: Comparing trust estimates when using different methods of data selection.

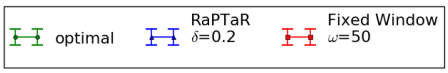
assessment does not guarantee that it will have the best performance. However, a model with perfect accuracy would select the best agents, and in other circumstances might help us avoid bad agents. Therefore, accuracy is still an insightful measure of trust assessment methods. The results in Figure 6.3 show that RaPTaR is significantly more accurate at estimating agent behaviours, and that this could be a contributing factor to its increased performance over the



(a) For a single run.



(b) Averaged over 100 runs.



(c) Legend.

Figure 6.2: Average utility per agent over time in single and multiple runs.

other techniques.

6.3.1 Exploration

In a dynamic environment it is necessary to continually monitor for changes otherwise agents' views of the world may become outdated. However, agents face an exploration-exploitation trade off [March, 1991]. We use the well known ϵ-greedy exploration algorithm, where we define the random probability of ex-

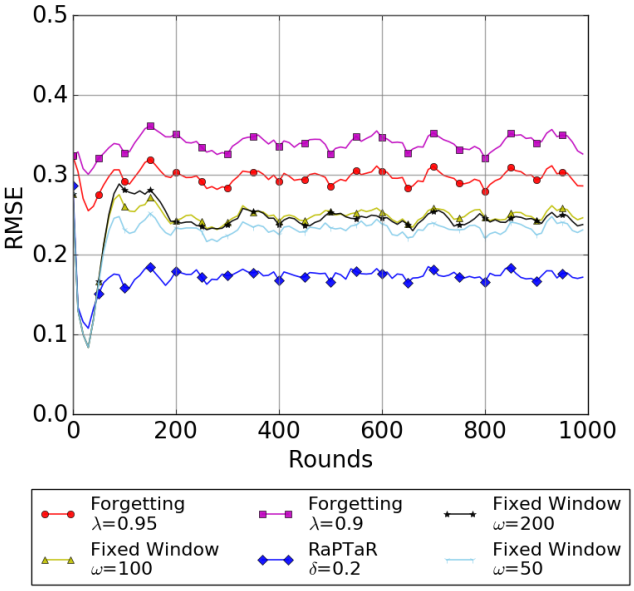


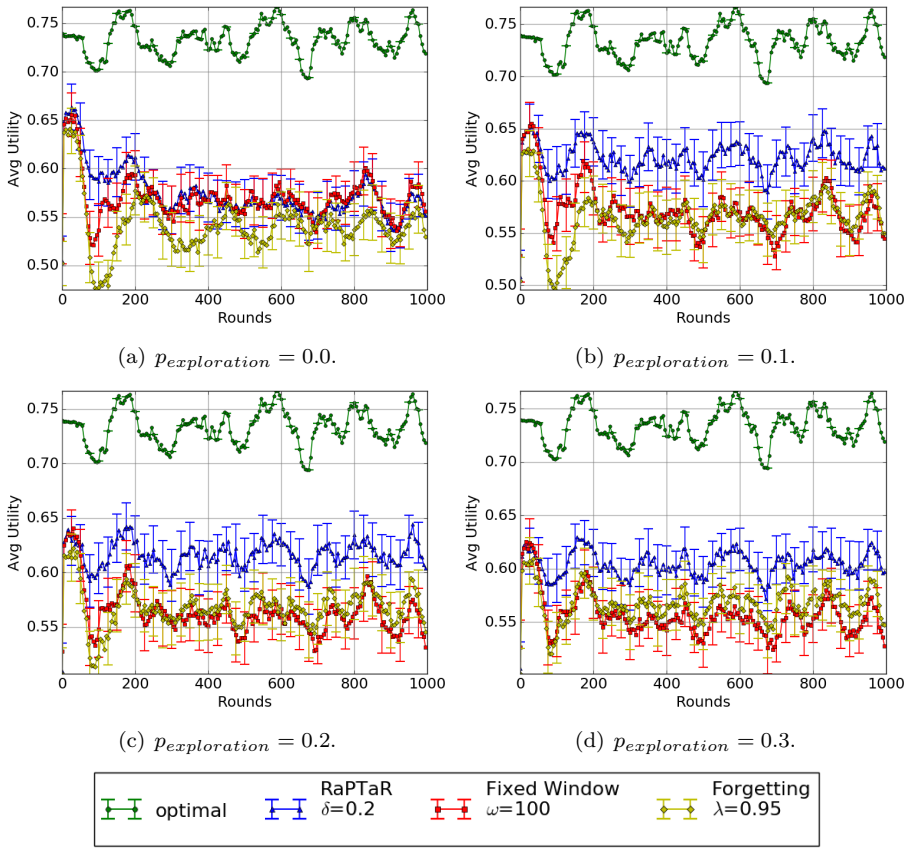
Figure 6.3: RMSE between behaviour and trust.

ploration as interacting with a random partner. We denote this probability as $p_{exploration}$. The necessary amount of exploration for agents to have a correct impression of their environment depends on how volatile the environment is. Figure 6.4 illustrates how different exploration rates affect RaPTaR, sliding windows, and forgetting factors. The results presented are from a fully connected network because agents have a large choice of partners, giving them the opportunity to exploit the knowledge they gain from exploration.

When agents do not explore, as in Figure 6.4(a), all models are equally blind to changes in the environment and perform poorly. As the level of exploration increases, RaPTaR performs the best because it has the capability to learn about the environment and exploit the knowledge it gains from exploration. Once exploration is as high as 30% (Figure 6.4(d)), the forced exploration slightly decreases the average utility of interactions to below 0.6 without gaining additional useful knowledge.

6.3.2 Outcome granularity

DRS is mathematically rigorous given an interaction outcome can be split into k bins for any value of $k > 1$. As k increases, the more specific an interaction outcome result becomes. One advantage of this is that we can select k which best suits the application. For example, some online marketplaces allow you to rate a seller with up to 5 stars, representing $k = 5$, while some recommender



(a) $p_{exploration} = 0.0$.

(b) $p_{exploration} = 0.1$.

(c) $p_{exploration} = 0.2$.

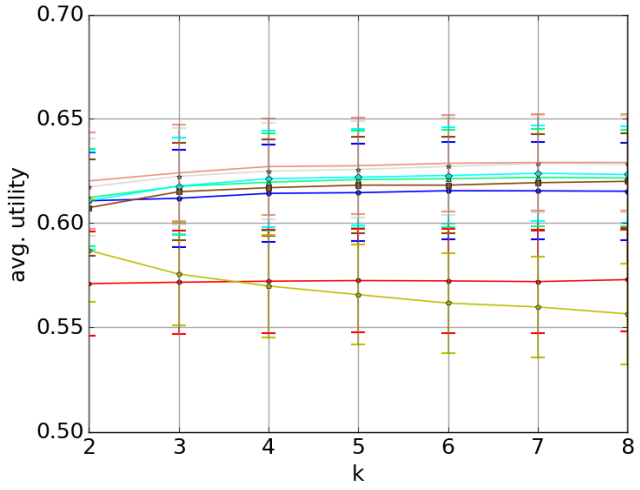
(d) $p_{exploration} = 0.3$.

(e) Legend

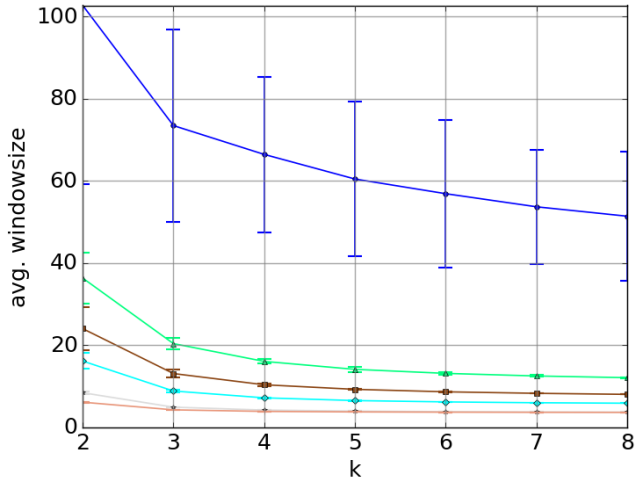
Figure 6.4: Average utility per agent, given different exploration rates. Agents use DRS and are in a fully connected network.

systems use a thumbs up and thumbs down to rate interactions, representing $k = 2$. Figure 6.5(a) demonstrates that RaPTaR marginally improves as k increases.

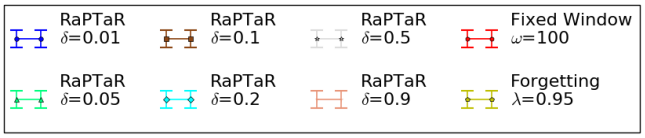
Figure 6.5(b) shows the average window size increases with δ but shrinks as k increases. As interaction outcomes become more precise, fewer instances are needed to detect a change in behaviour because fewer past experiences are needed to accurately represent the agent's behaviour. For example, when $k = 5$, an interaction outcome for an agent with an objectively average behaviour might fall in the third bucket which would accurately represent their true behaviour after just one interaction. When $k = 2$, the interaction outcome will alternate between good and bad, so a trustor needs more interaction outcomes to calculate the agent's true behaviour. When RaPTaR has a higher value for δ, despite causing the retention of fewer interaction outcomes, RaPTaR can still estimate behaviour well using the predictive component and any meta-information it has



(a) Average utility per timestep.



(b) RaPTaR window sizes.



(c) Legend

Figure 6.5: Performance of RaPTaR when interaction outcomes can be perceived into differing k categories for DRS trust assessment.

stored about its partner's behaviour.

Most importantly, as the average window size varies between different values of δ, there is no statistically significant difference in RaPTaR's performance, shown in Figure 6.5(a). As the average window size increases, so does its stan-

dard deviation because when behaviour changes, even low values of δ detect this and the window shrinks to adapt. During static periods of behaviour, the window is more likely to grow larger. When there is a high standard deviation in the window size, we know that the window adjusted to the level of dynamic behaviour it could detect, while a fixed window does not offer this flexibility and ultimately includes too much outdated information. Overall, RaPTaR adapts to behaviour change with all values of δ, demonstrating it is not a highly sensitive parameter.

6.3.3 Random, non-patterned behaviour

Finally, we demonstrate that RaPTaR's predictive component does not overfit to agents' behaviours which do not have pattern that can be learnt and exploited. We present results in Figure 6.6 of how RaPTaR performs on random dynamic behaviour. A group's behaviour can change suddenly or gradually, but it is not repetitive, rendering the predictive component of RaPTaR redundant.

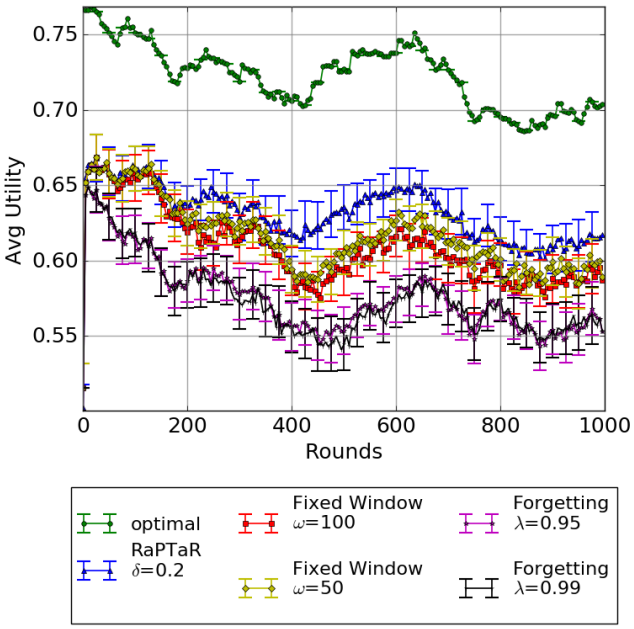


Figure 6.6: RaPTaR performance given random, non-patterned agent behaviour.

The improvement from RaPTaR is not as prominent, because there are no patterns to exploit. However, RaPTaR outperforms the other techniques because the reactive element, which uses a modification of AdWin, allows it to react to the detected level of dynamic behaviour. The predictive component is

not influencing the trust assessment enough for the agent to frequently select bad partners.

6.4 Discussion

In this chapter, we presented RaPTaR, a method which identifies representative information for trust assessment given an agent might need to react to, or predict, behaviour changes. Our method exploits agents' repetitive behaviour and is also shown to be effective when agent behaviour is dynamic and changing randomly.

We demonstrated that RaPTaR is successful when the group of agents with the best behaviour changes. During periods of static behaviour, when the ranking of the best agents to interact with remains static, sliding windows and forgetting factors have time to identify the best group and then they perform equally to RaPTaR. However, once dynamic behaviour causes a change in the ranking of agent behaviour, RaPTaR adapts quickly, potentially predicting it, and continues to have good interactions while the other models need time to relearn who the best partners are. This difference is proven to be statistically significant and robust over multiple runs.

Selfish agents repeatedly choose interaction partners they already trust which results in a limited and biased set of interaction outcomes that does not give us information on the rest of the agent population. We addressed a similar problem in Chapter 4, where agents could not accurately assess partners with a stereotype model if they did not have enough data to represent the agent's true group. However, in this chapter we assume that groups are identifiable, and we can easily quantify when there is insufficient data about a specific group, so we do not need to use CIM. However, if we have had few or no interactions with a particular group, the sparse amount of data prevents RaPTaR from learning that group's behaviour patterns. To increase the amount of data about these groups, we have to encourage agents to interact with partners who have been assessed as less trustworthy. These interactions must be forced either through network structure which limits the available partner choice, or through exploration. Exploring to collect data sacrifices a possible higher utility in the short term from exploiting known good agents, however this trade-off can increase agents' knowledge, which leads to improving their partner selection and overall average utility. The extent to which an agent explores affects its performance. In future work, we can consider using exploration algorithms based on an agent's current performance, the quantity of data it currently has and a minimum trust threshold, to improve some of these limitations.

RaPTaR's predictive component will not be effective if behaviour changes are

not repetitive. However, our results show that RaPTaR's ability to detect and react to dynamic behaviour means that it still outperforms sliding windows and forgetting factors. This could be improved in future work by integrating an error detection module with the predictive component. This would prevent an agent from overfitting predictions to behaviour changes which are not reoccurring. RaPTaR could then give more or less precedence to the predictive component depending on how successful it is. An alternative approach to combat overfitting in RaPTaR's predictive component could be to delete the meta-information stored in the cells of the transition matrix, either randomly or based on a probability given how long since the cell was updated. The transition matrix would have to relearn that particular transition if it was true, however, only deleting one cell would have a minimal effect on the overall predicted average utility but have the benefits of removing transitions which are not repetitive, and it would adapt to changing patterns of behaviour.

In this work, we have not considered how detectable traits of the environment, which indicate the context the agent is working in, might affect the prediction of the agent's next behaviour. There are few context-aware trust and stereotype models, however these do not consider dynamic behaviour. Exploring how predictions can be made using contextual information is another area of future work.

As a final consideration for future work in this area, we believe that adapting, or substituting, the RePro algorithm, such that agents can learn more complex or more realistic patterns of dynamic behaviour. Currently, a transition matrix is used to record the changes in agent behaviour which assumes a Markov chain of the first order. However, increasing this to use multiple previous behaviours to predict the next behaviour could improve the probability estimates. Additionally, using Naive Bayes classifiers to assess the probabilities of different behaviours occurring could improve the work, a concise overview of which is provided in the literature [Rish et al., 2001].

The computational complexity of RaPTaR scales with the size of an agent's history of interactions which it must search through after every interaction to statistically detect change using the AdWin algorithm. Therefore, RaPTaR's time complexity is $O(|W|)$ where $|W|$ is the size of the adaptive window of interaction outcomes. The additional computation RaPTaR performs to learn about dynamic behaviours is constant. Even though the AdWin algorithm itself has not made the run time worse, if it results in retaining large quantities of data it might slow down other components of the overall trust assessment process. For example, the most computationally expensive element is to gather reputation information, which becomes increasingly slow as the amount of data increases. As discussed in Chapter 5, enforcing a maximum window size for

the adjustable window can prevent the window from becoming very large and would only require a conservative estimate of the necessary amount of data. This approach is unlikely to negatively affect the performance of RaPTaR as ultimately the size of the window would be adjusted as needed by the AdWin algorithm.

CHAPTER 7
Conclusion

In this thesis, we investigated how to improve partner selection in MAS based on trust where agents have unrepresentative or insufficient information about others. Our contributions address different causes of this problem, and we presented techniques for agents to select relevant information or the appropriate behavioural assessment method to use. One of the main causes of insufficient data to support trust assessments is the selfish nature of agents, as they choose interaction partners they have already determined to be trustworthy. This leaves them unaware of potentially better partners, especially when behaviours are changing over time. Additionally, MAS can be highly dynamic, where agents change their behaviour at different times and speeds. We have investigated the importance of carefully selecting relevant information to use in trust assessment, rather than using the same predefined amount of information for each agent regardless of the circumstance.

Our work primarily extends stereotype literature, a research area which complements trust assessment by correlating trust with agents' observable features. If agents in groups have similar trustworthiness, we can exploit knowledge from one agent to infer trustworthiness about another from the same group. We use a prominent decision tree stereotype model from Burnett *et al.* [Burnett et al., 2010], to illustrate our approach, however, other stereotype models could be substituted. We evaluate the improvement our contributions make to a stereotype model, specifically focusing on how existing methods manage the information used as part of trust assessment. We also consider how agents can learn past agent behaviours as summaries with metadata such as behaviour duration and preceding behaviour. This improves predictions of future behaviour compared to existing trust literature which can only recall recent interaction records. Finally, we introduce a simple but novel stereotype algorithm, DCS, based on tag literature.

7.1 Contributions

In this thesis, we have shown that agents need more flexibility to select the data and methods they use in trust and reputation assessment, allowing them to adapt to the partners in their current environment. Our work has been evaluated in an illustrative multi-agent system where we introduced definitions for sudden and gradual dynamic agent behaviour. Below, we outline how this

thesis makes contributions towards solving our original research objectives.

- Our first contribution is a method to detect and handle class imbalance in a set of interaction outcomes used for stereotype assessment. CIM is presented in Chapter 4 to address RO 1, regarding how agents might have insufficient past interactions to represent each of the agent profiles. This prevents agents from learning accurate stereotypical trust models for all agents, and ultimately can lead to choosing bad interaction partners based on those models. Specifically, CIM is a clustering tool which first identifies if the agent who is choosing a partner has a set of past experiences which represents the agent being assessed, before using the stereotype model to assess them. CIM demonstrates the importance of giving agents the flexibility to decide which assessment method is most appropriate to use based on the current data set and the agent being assessed.

  An important finding from this chapter is that the accuracy of the trust and stereotype models does not guarantee agents will select the best partners for interactions, unless it is perfect. Trust and stereotype models can only accurately assess the small subset of good agents it regularly interacts with, but inaccurately assesses the majority of other agents, leading to a high average error. However, the model ultimately selects good partners because of a few good interactions with them and ignoring everyone else. Therefore, we understand that it is important for trust models to improve their accuracy in general, but we conclude that it is not a useful measure of trust models on its own.

- The second contribution was a novel stereotype assessment method, Direct Comparative Stereotypes, inspired by evolutionary biology. We introduce DCS in Chapter 4 as a simple comparison between two agents' observable features to make an initial estimate about their behaviour. This addressed RO 2 to improve trust assessments when there is no other available information about that agent. We note that the simplicity of DCS encompasses the following limitations. Firstly, agents who look the same may not behave the same. We demonstrated that a high threshold of similarity to infer similar behaviours improves results but this vulnerability still conceptually exists. Secondly, DCS can be exploited by malicious agents who mimic the observable features of their partner. Finally, DCS cannot distinguish between features which do correlate with behaviour and features which do not. DCS is then vulnerable when there is a high proportion of irrelevant features which are similar but should not be used to infer behaviour. Typically, stereotype models are machine learning methods designed to distinguish between relevant and irrelevant features. However,

DCS can improve initial partner selection when there is no information at all. Existing methods use random partner selection in comparison. DCS allows good agents to identify other good agents, and as a result of those initial interactions, good agents' reputations will be spread.

- Contribution 3 statistically assesses if there has been a change in a group's behaviour. We demonstrated in Chapter 5 how dynamic agent behaviour renders past interaction data unrepresentative of current behaviours, addressing RO 3. Existing methods have rigid input parameters fixing the amount of past data used to assess every agent. However, we might not know how much past data will be representative, that amount might change over time, or it might need to vary for different agents. We presented AdWin Tree, showing how adaptive windows monitoring the stereotypes output by a decision tree stereotype model can effectively select the interaction outcomes relevant to each stereotype. Adaptive windows allow an agent to retain representative information for each group, which may span different amounts of time. When dynamic agent behaviour is detected for a stereotype, the interaction records prior to the point of change are removed to not affect the trust and stereotype assessment of agents from that group in the future. We found that when the average size of the adaptive window was similar to the fixed window size, the large standard deviation in the adaptive window demonstrated how it was adapting to the level of dynamic behaviour in the group. Our technique demonstrated that agents with the ability to adapt how much data they use in their trust assessment based on what they assess to be statistically relevant improved their trust assessments and ultimately their partner selection. The level of confidence required to assume agent behaviour change has occurred based on the data is the only parameter to AdWin Tree, and the results were not sensitive to its value.

  In this chapter, it emerged that dynamic agent behaviour gives rise to a series of issues, however, it reduces the class imbalance problem. As agent behaviours change, agents learn through trust techniques which agents are no longer reliable and prompts them to choose other interaction partners. This is a small amount of exploration, and helps to reduce class imbalance. Therefore, our AdWin Tree technique did not need to be paired with CIM, and the results of the two of them together did not show any additional benefit. Overall, if any information about the environment is known in advance, agents should use CIM when agent behaviours are static, and AdWin Tree when agent behaviour is dynamic.

- As part of addressing RO 4, in Chapter 6, we assumed that groups are

identifiable so there is no requirement to learn them using a stereotype model. This meant that agents can accurately discern between interaction outcomes from different groups without noise. Our method RaPTaR, which expands on the adaptive window concept from Chapter 5, learns and exploits agents' behaviour patterns. Existing trust, reputation and stereotype models use recent past interaction data to make a behaviour assessment about an agent and when data are considered too old to be included in the trust assessment they are filtered out indefinitely. However, RaPTaR summarises the different behaviours an agent exhibits to prevent having to relearn them if they reoccur, and learns transition patterns between those behaviours to predict when they might occur in the future. RaPTaR's reactive component is effective enough that RaPTaR performs better than existing work when behaviours are dynamic without a pattern. This is the only chapter where we perform exploration to increase agents' knowledge of a changing environment. When groups are perfectly identifiable, agents can easily only select partners from the group they assess to be the most trustworthy, leaving them with no information about the other groups' behaviour. A small amount of exploration can greatly improve RaPTaR's ability to learn about all the groups' behaviour patterns without compromising its performance. Our exploration technique is quite naïve though as it only selects a random partner in a small proportion of interactions.

7.2 Future Work

In this thesis, we explored how relevant information affects trust, reputation and stereotype assessment. Improving cooperation and coordination amongst agents is a challenge which is always evolving as a result of new technologies and the increasing scale and application of MAS. Some logical next steps which build on our contributions are outlined below.

Firstly, a limitation which exists throughout this thesis is the assumption that all agents behave stereotypically. This assumption allowed us to collate information from multiple agents to improve our understanding of group behaviour. However, agents may deviate from their stereotypes, or not follow a stereotype at all. The adaptive window technique presented in Chapter 5 could form the basis of future work to statistically select relevant information for individual agents. For the evaluation presented in this thesis, there are too little data on individual agents to monitor them this way, however, the scope of this thesis has been to address sparse and limited data. In situations where there is a high volume of data about most of the agents in a neighbourhood, for example

in small-world and scale-free graph types, we can consider tracking individual agents similarly to how AdWin Tree tracks groups. Alternatively, we suggest a hybrid approach to flexibly monitor individual agents or groups depending on the amount of data available for the agent being assessed. Representing the agent population with a Gaussian Mixture Model could also be an interesting approach to stereotyping, where the parameters can allow for different amounts of variance within each group of agents.

The fair division of labour in MAS is not the focus of this work, however a link exists between trust and fairness [Wierzbicki, 2010]. We hypothesise that the more accurately we can assess an agent then the more fairly it can be treated. Trusted AI is gaining more traction, and the ability to describe why agents are being given certain tasks and whether or not that is appropriate is an important ethical consideration of future work.

In this thesis, we have only considered dynamic agents and not the nature of the environment. Context-aware trust models take environmental features into account as well as the interaction partner [Wang et al., 2017]. Understanding the agent environment could improve trust assessments and make agents more adaptable and resilient to environment changes [Hoelz and Rakga, 2015]. Context-aware stereotype models are relatively unstudied however, they can be highly successful at inferring knowledge between different situations [Zhou et al., 2015]. RaPTaR could be applied to learn and exploit environmental features, as well as agents' features, to better understand agents' behaviour.

In this thesis, we have addressed non-threatening dynamic agent behaviour, and we do not identify malicious attackers. Our contribution to statistically detect changes in agent behaviour could be applied to identify specific types of attacks on trust and reputation assessment [Jøsang, 2012], for example, agents who commit the oscillating attack by repeatedly gaining trust over time and then exploiting it [Srivasta et al., 2005; Xiong and Liu, 2004]. Malicious agents might exploit knowledge of which features are trustworthy and then mimic them. Throughout this thesis, we have demonstrated that the stereotype model can be a useful tool for analysing behaviour at all times, even when the *a priori* provided by the stereotype model has little impact on the final trust assessment. Therefore, the stereotype model could be used to identify anomalous agents, who have observable features which appear to fit a stereotype, but their trust assessment does not align with the group's stereotypical trust [Crosbie and Spafford, 1995]. Techniques to identify outliers who are committing criminal activities in centralised e-commerce systems exist in data analysis literature [Breunig et al., 2000].

Finally, we have evaluated this work in simulated conditions. Our environment is comparable to existing work, and uses parameters to define qualities of

real-world scenarios which might vary. The evaluations have shown our work to be robust in contexts where agents have a large choice of potential partners and sparse data. However, we note that real-world networks and data can enlighten us to unforeseen problems, and are not as perfect as a simulated environment. The next steps for the work presented in this thesis should include exploring its efficacy in real world scenarios.

7.3 Final Remarks

In this thesis, we demonstrated the negative impact of unrepresentative data on trust, reputation and stereotype assessment. We identified that the two main causes of unrepresentative data are class imbalance and dynamic agent behaviour. Our techniques either statistically detect whether an agent has representative data of a particular agent type, or whether past experience data are representative of an agent's current behaviour. These techniques can replace existing methods such as fixed sliding windows and forgetting factors, which are limited in MAS because they lack flexibility and granularity to account for the dynamic nature of agents.

The models presented in this thesis are most effective in situations where agents have a wide choice of partner and sparse information, demonstrated by our results from fully connected graphs compared to topologies where agents have smaller neighbourhoods. Making trust assessment in a situation with limited data is important however, the efficacy of the models is significantly reduced when agents are members of smaller neighbourhoods. In more restricted topologies, agents have the opportunity to gather a lot of information about the fewer available agents so their trust assessments are more accurate and take precedence.

The techniques presented here are not as simple or easy to implement as sliding windows and forgetting factors however, they are more appropriate for MAS which are constantly evolving. The uptake of decentralised networks in the real-world means accurately and fairly assessing each agent is becoming increasingly important.

More generally, in the field of MAS, it is important to consider how all methods to achieve communication and cooperation must move towards parameter free techniques. Agents should be adaptable to all facets of the environment, including trust. In this thesis, we have demonstrated how agents who respond to having unrepresentative information depending on their situation are more successful and robust across different dynamic environments compared to existing work in the field of trust and reputation.

Bibliography

R. Axelrod. 1984. The evolution of cooperation. *Basic Books, New York. Econornetrica*, 39:383–396.

R. Axelrod. 1986. An evolutionary approach to norms. *American Political Science Review*, 80(4):1095–1111.

A.-L. Barabási and R. Albert. 1999. Emergence of scaling in random networks. *Science*, 286(5439):509–512.

C. Bicchieri, R. Jeffrey, and B. Skyrms. 1997. The dynamics of norms. *Ethics*.

A. Bifet and R. Gavaldà. 2007. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 443–448.

A. Bifet and R. Gavaldà. 2009. Adaptive learning from evolving data streams. In *International Symposium on Intelligent Data Analysis*. Springer.

D. Boswarthick, O. Elloumi, and O. Hersent. *M2M Communications: A Systems Approach*. John Wiley & Sons, 2012.

M. Bowling and M. Veloso. 2001. Rational and convergent learning in stochastic games. In *International Joint Conference on Artificial Intelligence*, volume 17, pages 1021–1026. Lawrence Erlbaum Associates Ltd.

F. Brazier, B. Dunin-Keplicz, J. Treur, and R. Verbrugge. 1997. Modelling internal dynamic behaviour of BDI agents. In *ModelAge Workshop on Formal Models of Agents*, pages 35–56.

M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. 2000. Lof: identifying density-based local outliers. In *ACM SIGMOD Record*, volume 29, pages 93–104. ACM.

C. Burnett, T. J. Norman, and K. Sycara. 2010. Bootstrapping trust evaluations through stereotypes. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, pages 241–248.

C. Burnett. *Trust Assessment and Decison-making in Dynamic Multi-agent Systems*. PhD thesis, University of Aberdeen, 2011.

C. Burnett, T. J. Norman, and K. Sycara. 2011. Sources of stereotypical trust in multi-agent systems. In *Proceedings of the 14th International Workshop on Trust in Agent Societies*, pages 25–39.

C. Burnett, T. J. Norman, and K. Sycara. 2013. Stereotypical trust and bias in dynamic multiagent systems. *ACM Transactions on Intelligent Systems and Technology*, 4(2):26.

C. Castelfranchi and R. Falcone. 1998. Principles of trust for mas: Cognitive anatomy. social importance, and quantification. In *Multi Agent Systems, 1998. Proceedings. International Conference on*, pages 72–79. IEEE.

F. Cerutti, C. Burnett, and N. Oren. 2015a. Representative agents and the contract negotiation cold start problem. In *International Workshop on Coordination, Organizations, Institutions, Norms and Ethics for Governance of Multi-Agent Systems (COINE) at AAMAS*.

F. Cerutti, N. Oren, and C. Burnett. 2015b. Global approximations for principal agent theory. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 1845–1846.

N. V. Chawla. 2009. Data mining for imbalanced datasets: An overview,. In *Data Mining and Knowledge Discovery Handbook*, pages 875–886. Springer.

R. Chen, J. Guo, and F. Bao. 2016. Trust management for soa-based iot and its applications to service composition. *IEEE Transactions on Services Computing*, 9(3):482–495.

S. Chen, H. He, K. Li, and S. Desai. 2010. Musera: Multiple selectively recursive approach towards imbalanced stream data mining. In *Neural Networks (IJCNN), The 2010 International Joing Conference on*, pages 1–8.

M. Crosbie and E. H. Spafford. Defending a computeur system using autonomous agents. Technical report, Purdue University, 1995.

M. Şensoy, B. Yilmaz, and T. J. Norman. 2016. STAGE: Stereotypical trust assessment through graph extraction. *Computational Intelligence*, 32(1):72–101.

G. D'Angelo, S. Rampone, and F. Palmieri. 2017. Developing a trust model for pervasive computing based on apriori association rules learning and bayesian classification. *Soft Computing*, 21(21):6297–6315.

R. Dawkins. *The Selfish Gene*. Oxford University Press, 1976.

M. Debra, K. E. Weick, and R. M. Kramer. 1995. Swift trust and temporary groups. *Trust in Organizations: Frontiers of Theory and Research*, page 166.

C. Dellarocas. 2000. Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior. In *Proceedings of the 2nd ACM Conference on Electronic Commerce*, pages 150–157.

E. H. Durfee. 2001. Scaling up agent coordination strategies. *Computer*, 34(7): 39–46.

M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Knowledge Data Discovery*, number 34, pages 226–231.

J. Fadul, K. Hopkinson, C. Sheffield, J. Moore, and T. Andel. 2011. Trust management and security in the future communication-based" smart" electric power grid. In *System Sciences (HICSS), 2011 44th Hawaii International Conference on*, pages 1–10. IEEE.

R. Falcone and C. Castelfranchi. 2001. Social trust: a cognitive approach. In *Trust and Deception in Virtual Societies*, pages 55–90. Springer.

J. Gama, I. Žliobait, A. Bifet, M. Pechenizkiy, and A. Bouchachia. 2014. A survey on concept drift adaptation. *ACM Computing Surveys (CSUR)*, 46 (4):44.

D. Gambetta. 2000. Can we trust trust? *Trust: Making and Breaking Cooperative Relations*, 13:213–237.

J. Gao, B. Ding, W. Fan, J. Han, and S. Y. Philip. 2008. Classifying data streams with skewed class distributions and concept drift. *IEEE Internet Computing*, 12(6).

A. Glass, D. L. McGuinness, and M. Wolverton. 2008. Toward establishing trust in adaptive agents. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*, pages 227–236. ACM.

J. Granatyr, V. Botelho, O. R. Lessing, E. Emílio, J.-P. Barthès, and Fabrício. 2015. Trust and reputation models for multiagent systems. *ACM Computing Surveys (CSUR)*, 48(2):27.

E. Gray, C. Jensen, P. O'Connell, S. Weber, J.-M. Seigneur, and Y. Chen. 2006. Trust evolution policies for security in collaborative ad hoc applications. *Electronic Notes in Theoretical Computer Science*, 157(3):95–111.

N. Griffiths. 2005. Task delegation using experience-based multi-dimensional trust. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multi-agent Systems*, pages 489–469. ACM.

N. Griffiths. 2006. A fuzzy approach to reasoning with trust, distrust and insufficient trust. In *International Workshop on Cooperative Information Agents*, pages 360–374. Springer.

N. Griffiths. 2008. Tags and image scoring for robust cooperation. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, volume 2, pages 575–582. International Foundation for Autonomous Agents and Multiagent Systems.

N. Griffiths, K.-M. Chao, and M. Younas. 2006. Fuzzy trust for peer-to-peer systems. In *Distributed Computing Systems Workshops, 2006. ICDCS Workshops 2006. 26th IEEE International Conference on*.

S. J. Grossman and O. D. Hart. 1992. An analysis of the principal-agent problem. In *Foundations of Insurance Economics*, pages 302–340. Springer.

D. Hales and B. Edmonds. 2003. Evolving social rationality for MAS using tags. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems*, pages 497–503. ACM.

W. D. Hamilton. 1964a. The genetical evolution of social behaviour. *Journal of Theoretical Biology*, 7(1):1–16.

W. D. Hamilton. 1964b. The genetical evolution of social behaviour ii. *Journal of Theoretical Biology*, 7(1):17–52.

M. B. Harries, C. Sammut, and K. Horn. 1998. Extracting hidden context. *Machine Learning*, 32(2):101–126.

B. W. Hoelz and C. G. Rakga. 2015. Towards a cognitive meta-model for adaptive trust and reputation in open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 29(6):1125–1156.

B. W. Hoelz and C. G. Ralha. 2015. Towards a cognitive meta-model for adaptive trust and reputation in open multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 29(6):1125–1156.

R. T. Hoens, R. Polikar, and N. V. Chawla. 2012. Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artifical Intelligence*, 1(1):89–101.

L. M. Hogg and N. Jennings. 2001. Socially intelligent reasoning for autonomous agents. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 31(5):381–393.

E. Howley and C. O'Riordan. 2005. The emergence of cooperation among agents using simple fixed bias tagging. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, volume 2, pages 1011–1016. IEEE.

T. D. Huynh and N. Jennings. 2004. FIRE: An integrated trust and reputation model for open multi-agent systems. In *ECAI 2004: 16th European Conference on Artificial Intelligence, August 22-27, 2004, Valencia, Spain: including Prestigious Applicants [sic] of Intelligent Systems (PAIS 2004): proceedings*, volume 110, page 18.

S. L. Jarvenpaa and D. E. Leidner. 1999. Communication and trust in global virtual teams. *Organization Science*, 6(10):791–815.

S. Javanmardi, M. Shojafar, S. Shariatmadari, and S. S. Ahrabi. 2014. Fr trust: a fuzzy reputation–based model for trust management in semantic p2p grids. *International Journal of Grid and Utility Computing*, 6(1):57–66.

N. Jennings. 2001. An agent-based approach for building complex software systems. *Communication of the ACM*, 44(4):35–41.

N. Jennings and J. R. Campos. 1997. Towards a social level characterisation of socially responsible agents. *IEEE Proceedings-Software Engineering*, 144(1): 11–25.

N. R. Jennings. 2000. On agent-based software engineering. *Artificial Intelligence*, 117(2):277–296.

A. Jøsang and R. Ismail. 2002. The beta reputation system. *Proceedings of the 15th Bled Electronic Commerce Conference*, 5:2502–2511.

A. Jøsang. 2012. Robustness of trust and reputation systems: does it matter? In *IFIP International Conference on Trust management*, pages 253–262. Springer.

A. Jøsang and J. Golbeck. 2009. Challenges for robust trust and reputation systems. In *Proceedings of the 5th International Workshop on Security and Trust management (SMT 2009), Saint Malo, France*.

A. Jøsang and J. Haller. 2007. Dirichelet reputation systems. In *Availability, Reliability and Security, 2007. ARES 2007. The Second International Conference on*, pages 112–119. IEEE.

A. Jøsang, R. Ismail, and C. Boyd. 2007. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644.

L. Kagal, T. Finin, and A. Joshi. 2001. Trust-based security in pervasive computing environments. *Computer*, pages 154–157.

S. Kalenka and N. R. Jennings. 1999. Socially responsible decision making by autonomous agents. In *Cognition, Agency and Rationality*, pages 135–149. Springer.

S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina. 2003. The eigentrust algorithm for reputation management in p2p networks. In *Proceedings of the 12th International Conference on World Wide Web*, pages 640–651. ACM.

S. N. L. C. Keung and N. Griffiths. 2010. Trust and reputation. *Agent-based Service-oriented Computing*, pages 189–224.

J. E. Kittock. 1993. Emergent conventions and the structure of multi-agent systems. *L. Nadel and D. Stein, eds.*

M. Klusch and A. Gerber. 2002. Dynamic coalition formation among rational agents. *IEEE Intelligent Systems*, (3):42–47.

S. Lee, R. Sherwood, and B. Bhattacharjee. 2003. Cooperative peer groups in NICE. In *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, volume 2, pages 1272–1282.

Z. Liang and W. Shi. 2005. PET: A personalised trust model with reputation and risk evaluation for P2P resource sharing. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences.*

R. Lichtenwalter and N. V. Chawla. 2009. Adaptive methods for classification in arbitrarily imbalanced and drifting data streams. In *PAKDD workshops*, pages 53–75.

X. Liu, A. Datta, K. Rzadca, and E. Lim. 2009. Stereotrust: a group based personalized trust model. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, pages 7–16.

X. Liu, A. Datta, and K. Rzadca. 2013. Trust beyond reputation: A computational trust model based on stereotypes. *Electronic Commerce Research and Applications*, 12(1):24–39.

X. Liu, G. Tredan, and A. Datta. 2014. A generic trust framework for large-scale open systems using machine learning. *Computational Intelligence*, 30 (4):700–721.

G. Lu and J. Lu. 2017. Introduction to the investigating in neural trust and multi agent systems. In *Examining Information Retrieval and Image Processing Paradigms in Multidisciplinary Contexts*, pages 269–273. IGI Global.

N. Luhmann. 2000. Familiarity, confidence, trust: Problems and alternatives. *Trust: Making and Breaking Cooperative Relations*, 6:94–107.

D. W. Manchala. 2000. E-commerce trust metrics and models. *IEEE Internet Computing*, 4(2):36–44.

J. G. March. 1991. Exploration and exploitation in organizational learning. *Organization Science*, 2(1):71–87.

J. Marchant, N. Griffiths, M. Leeke, and H. Franks. 2014. Destabilising conventions using temporary interventions. In *Proceedings of the 17th International Workshop on Coordination, Organizations, Institutions and Norms*.

S. Marsh. *Formalising Trust as a Computational Concept*. PhD thesis, University of Sterling, 1994a.

S. Marsh. 1994b. Optimism and pessimism in trust. In *Proceedings of the Ibero-American Conference on Artificial Intelligence*.

S. Marsh and M. R. Dibben. 2005. Trust, untrust, distrust and mistrust - an exploration of the dark (er) side. In *iTrust*, pages 17–33. Springer.

D. Mayzlin, Y. Dover, and J. Chevalier. 2014. Promotional reviews: An empirical investigation of online review mainpulation. *The American Economic Review*, 104(8):2421–2455.

F. Messina, G. Pappalardo, A. Comi, L. Fotia, D. Rosaci, and G. M. Sarné. 2017. Combining reputation and qos measures to improve cloud service composition. *International Journal of Grid and Utility Computing*, 8(2):142–151.

D. T. Nguyen. *Trust Management for Complex Agent Groups*. PhD thesis, Auckland University of Technology, 2017.

T. D. Nguyen and Q. Bai. 2014. Accountable individual trust from group reputations in multi-agent systems. In *Pacific Rim International Conference on Artificial Intelligence*, pages 1063–1075.

T. D. Nguyen and Q. Bai. 2018. A dynamic bayesian network approach for agent group trust evaluation. *Computers in Human Behavior*.

M. Nowak and K. Sigmund. 1998. Evolution of indirect reciprocity by image scoring. *Nature*, 393(6685):573–577.

E. Parzen. 1962. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics*, 33(3):1065–1076.

C. Player and N. Griffiths. 2017a. Bootstrapping trust and stereotypes with tags. In *Proceedings of the 19th International Workshop on Trust in Agent Societies*.

C. Player and N. Griffiths. 2017b. Using tags to bootstrap stereotypes and trust. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pages 1691–1693. International Foundation for Autonomous Agents and Multiagent Systems.

C. Player and N. Griffiths. 2018a. Addressing concept drift in reputation assessment. In *Proceedings of the 17th International Conference on Autonomous Agents and Multiagent Systems*, pages 2048–2050.

C. Player and N. Griffiths. 2018b. Addressing concept drift in reputation assessment. In *Proceedings of the 10th International Workshop on Adaptive Learning Agents (ALA @ AAMAS)*.

C. Player and N. Griffiths. 2019a. Addressing class imbalance in trust and stereotype assessment. In *In Proceedings of the 21st International Workshop on Trust in Agent Societies*.

C. Player and N. Griffiths. 2019b. Improving trust and reputation assessment with dynamic behaviour. *The Knowledge Engineering Review*.

W. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. 2007. Statistical description of data: Are two distirbutions different. *Numerical Recipes: The Art of Scientific Computing*, pages 730–740.

S. D. Ramchurn, T. D. Huynh, and N. Jennings. 2004a. Trust in multi-agent systems. *The Knowledge Engineering Review*, 19(1):1–25.

S. D. Ramchurn, N. Jennings, C. Sierra, and L. Godo. 2004b. Devising a trust model for multi-agent interactions using confidence and reputation. *Applied Artificial intelligence*, 18(9-10):833–852.

K. Regan, P. Poupart, and R. Cohen. 2006. Bayesian reputation modelling in e-marketplaces sensitive to subjectivity, deception and change. In *Proceedings of the National Conference on Artificial Intelligence*.

P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. 2000. Reputation systems. *Communications of the ACM*, 43(12).

A. Rettinger, M. Nickles, and V. Tresp. 2007. Learning initial trust among interacting agents. In *International Workshop on Cooperative Information Agents*, pages 313–327.

R. L. Riolo, M. Cohen, and R. Axelrod. 2001. Evolution of cooperation without reciprocity. *Nature*, 414(6862):441–443.

I. Rish et al. 2001. An empirical study of the naive bayes classifier. In *IJCAI Workshop on Empirical Methods in Artificial Intelligence*, volume 3, pages 41–46.

J. S. Rosenschein. *Rational Interaction: Cooperation Among Intelligent Agents*. PhD thesis, Stanford University, 1986.

C. Ruiz, E. Menasalvas, and M. Spiliopoulou. 2009. C-denstream: Using domain knowledge on a data stream. In *International Conference on Discovery Science*, pages 287–301. Springer.

S. Ruohomaa and L. Kutvonen. 2010. Trust and distrust in adaptive inter-enterprise collaboration management. *Journal of Theoretical and Applied Electronic Commerce*, 5(2):118–136.

S. Ruohomaa, A. Hankalahti, and L. Kutvonen. 2011. Detecting and reacting to changes in reputation flows. In *IFIPTM*, pages 19–34. Springer.

J. Sabater and C. Sierra. 2001a. REGRET: A reputation model for gregarious societies. In *Fourth Workshop on Deception Fraud and Trust in Agent Societies*, volume 70, pages 61–69.

J. Sabater and C. Sierra. 2001b. REGRET: reputation in gregarious societies. In *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 194–195. ACM.

J. Sabater and C. Sierra. 2002. Reputation and social network analysis in multiagent systems. In *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-agent Systems: part 1*, pages 475–482. ACM.

A. Salehi-Abari and T. White. 2012. DART: a distributed analysis of reputation and trust framework. *Computational Intelligence*, 28(4):642–682.

F. P. Santos. 2017. Social norms of cooperation in multiagent systems. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*.

H. Schmeck, T. Ungerer, and L. Wolf. *Trends in Network and Pervasive Computing-ARCS 2002*. Springer, 2003.

M. Seckler, S. Heinz, S. Forde, A. N. Tuch, and K. Opwis. 2015. Trust and distrust on the web: User experiences and website characteristics. *Computers in Human Behavior*, pages 39–50.

M. Şensoy, J. Zhang, P. Yolum, and R. Cohen. 2009. Poyraz: Context-aware service selection under deception. *Computational Intelligence*, 25(4):335–366.

S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porsini. 2015. Security, privacy and trust in internet of thigns: The road ahead. *Computer Networks*, 76:146–164.

S. Song, K. Hwang, R. Zhou, and Y.-K. Kwok. 2005. Trusted p2p transactions with fuzzy reputation aggregation. *IEEE Internet Computing*, 9(6):24–25.

M. Srivasta, L. Xiong, and L. Liu. 2005. TrustGuard: countering vulnerabilities in reputation management for decentralised overlay networks. In *Proceedings of the 14th International Conference on World Wide Web*, pages 422–431.

P. Taylor, L. Barakat, S. Miles, and N. Griffiths. 2018. Reputation assessment: a review and unifying abstraction. *The Knowledge Engineering Review*, 33.

L. Teacy, J. Patel, N. Jennings, and M. Luck. 2006. Travos: Trust and reputation in the context of inaccurate information sources. *Autonomous Agents and Multi-Agent Systems*, 12(2):183–198.

L. Teacy, M. Luck, A. Rogers, and N. Jennings. 2012. An efficient and versatile approach to trust and reputation using hierarchical bayesian modelling. *Artificial Intelligence*, 193:149–185.

R. L. Trivers. 1971. The evolution of reciprocal altruism. *Quarterly Review of Biology*, pages 35–37.

G. K. Tso and K. K. Yau. 2007. Predicting electricity energy consumption: A comparison of regression analysis, decision tree and neural networks. *Energy*, 32(9):1761–1768.

A. Tsymbal. 2004. The problem of concept drift: definitions and related work. *Computer Science Department, Trinity College Dublin*, 106(2).

A. R. Wagner. 2009. Creating and using matrix representations of social interaction. In *Proceedings of the 4th ACM/IEEE International Conference on Human Robot Interaction*, pages 125–132. ACM.

A. R. Wagner. 2010. Using stereotypes to understand one's partner. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 1445–1446. International Foundation for Autonomous Agents and Multiagent Systems.

A. R. Wagner. 2012. Using cluster-based stereotyping to foster human-robot cooperation. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1615–1622. IEEE.

C. J. C. H. Wakins. 1992. Q-learning. *Machine Learning*, 8(3-4):279–292.

Y. Wang. *Trust-based service management for service-oriented mobile ad hoc networks and its applications to service composition and task assignment with multi-objective optimisation goals*. PhD thesis, Virginia Tech, 2016.

Y. Wang, R. Chen, J.-H. Cho, A. Swami, Y.-C. Lu, C.-T. Lu, and J. Tsai. 2017. Catrust: context-aware trust management for service-oriented ad hoc networks. *IEEE Transactions on Services Computing*.

D. J. Watts and S. H. Strogatz. 1998. Collective dynamics of 'small-world'networks. *Nature*, 393(6684).

G. I. Webb, R. Hyde, H. Cao, H. L. Nguyen, and F. Petitjean. 2016. Characterizing concept drift. *Data Mining and Knowledge Discovery*, 30(4):964–994.

A. Whitby, A. Jøsang, and J. Indulska. 2004. Filtering out unfair ratings in bayesian reputation systems. In *Proceedings of the 7th International Workshop on Trust in Agent Societies*.

A. Wierzbicki. *Trust and fairness in open, distributed systems*. Springer, 2010.

M. Wooldridge and N. R. Jennings. 1995. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115–152.

Y. Y. X. Wu and X. Zhu. 2005. Combining proactive and reactive predictions for data streams. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge Discovery in Data Mining*, pages 710–715. ACM.

L. Xiong and L. Liu. 2004. PeerTrust: Supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843–857.

Y. Yamamoto. 1990. A morality based on trust: Some reflections on japanese morality. *Philosophy East and West*, 40(4):451–469.

Y. Yao, S. Ruohomaa, and F. Xu. 2012. Addressing common vulnerabilities of reputation systems for electronic commerce. *Journal of Theoretical and Applied Electronic Commerce*, 7(1):1–20.

B. Yu and M. P. Singh. 2003. Detecting deception in reputation management. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multi-agent Systems*, pages 73–80. ACM.

J. Zhang and R. Cohen. 2006. A personalized approach to address unfair ratings in multiagent reputation systems. In *Proceedings of the AAMAS Workshop on Trust in Agent Societies*.

P. Zhou, X. Gu, J. Zhang, and M. Fei. 2015. A priori trust inference with context-aware stereotypical deep learning. *Knowledge-based Systems*, 88:97–106.

E. Zupancic and D. Trcek. 2017. QADE: a novel trust and reputation model for handling false trust values in e–commerce environments with subjectivity consideration. *Technological and Economic Development of Economy*, 23(1): 81–110.