Reinforcement Learning for Argumentation

SULTAN MOHAMMAD ALAHMARI

Doctor of Philosophy

University of York

Computer Science

JULY 2020

Abstract

Argumentation as a logical reasoning approach plays an important role in improving communication, increasing agree-ability, and resolving conflicts in multi-agent-systems (MAS). The present research aims to explore the effectiveness of argumentation in reinforcement learning of intelligent agents in terms of, outperforming baseline agents, learning transfer between argument graphs, and improving relevance and coherence of dialogue quality.

This research developed '*ARGUMENTO+*' to encourage a reinforcement learning agent (RL agent) playing abstract argument game for improving performance against different baseline agents by using a newly proposed state representation in order to make each state unique. When attempting to generalise this approach to other argumentation graphs, the RL agent was not able to effectively identify the argument patterns that are transferable to other domains.

In order to improve the effectiveness of the RL agent to recognise argument patterns, this research adopted a logic-based dialogue game approach with richer argument representations. In the DE dialogue game, the RL agent played against hard-coded heuristic agents and showed improved performance compared to the baseline agents by using a reward function that encourages the RL agent to win the game with minimum number of moves. This also allowed the RL agent to adopt its own strategy, make moves, and learn to argue.

This thesis also presents a new reward function that makes the RL agent's dialogue more coherent and relevant than its opponents. The RL agent was designed to recognise argument patterns, i.e. argumentation schemes and evidence support sources, which can be related to different domains. The RL agent used a transfer learning method to generalise and transfer experiences and speed up learning.

Acknowledgements

Bismillaah ar-Rahman ar-Raheem, In the name of God the most Compassionate the Most Merciful. All praise and thanks be to Allah, the Lord of the Worlds, and may the peace and blessings be on His Prophet and Messenger Muhammad, on his family and all of his companions. First, I sincerely thank Allah (GOD) Almighty for the blessings he has bestowed upon me and for His guidance during my PhD journey.

Words are not enough express the gratitude I feel for everyone who have stood by me and supported me throughout this research endeavour. Putting his hope and confidence -**after my God**- in what I will add and offer to my great country and the knowledge in general.

I am extremely thankful to my country -**The Kingdom of Saudi Arabia** - for giving me the opportunity to complete my postgraduate masters and PhD in Artificial Intelligence, to help contribute to the development of the country. Special thanks must be given to ***The Custodian of the Two Holy Mosques, King Salman bin Abdulaziz Al Saud***. May Allah protect him. Also, to ***His Royal Highness Prince Mohammad bin Salman bin Abdulaziz, Crown Prince, Deputy Prime Minister and Minister of Defence***. May Allah protect them for all the support and help they provide to all students, especially those studying overseas.

I would also like to thank ***Prince Dr Turki bin Saud, Former president of KACST*** who supported personally and approved my application to study at the University of York. He has all my love, appreciation and respect.

After my God - I wish to thank and appreciate all those people for supporting and encouraging me to carry on with my PhD journey to achieve my goal. These people are very special and have had a beautiful impact on my personal and practical life.

- ***Firstly, my parents, Mohammad Saeed Alahmari and Fatimah Hasan Alahmari*** without their support and prayers - **after my God**- I would not be who I am today. I will be eternally grateful for their unconditional love and support. I will always have deep regards for the enormous sacrifices that they have made to ensure that I receive an outstanding education. Their blessings and kindness have always motivated me to give my best, pursue my goals, and aim higher. Thank you very much from the bottom of my heart.

- A very special, heartfelt thanks to ***my wife Suaad***, who has always supported and motivated me on my scientific journey. She bore the trouble of travelling and alienation for me and has been really patient while I was busy doing my research. I would like to send all my love, appreciation and respect to her.

- ***My brothers (Saeed, Abdullah, Ahmad, Khaled and Hasan)*** and ***my sisters (Zainah, Ayshah and Shatha)*** who encouraged and motivated me during this journey.

- ***My sons, Hatem and Talal***, thank you very much for being my source of motivation. I pray to Allah to bless you and I wish you all the best in your life.

- Thanks to all ***my friends***, those who are in the UK and Saudi Arabia.

- Sincere thanks to ***my supervisors, Dr. Tommy Yuan and Dr Daniel Kudenko***. I will always be grateful for all that I have learnt from them about the AI field. I will never forget how flexible they have been whenever I had questions or needed information. I would just knock on their doors, even interrupting them sometimes while they are busy, but they were always happy to help. So thank you very much for being patient and friendly.

- Thanks also to ***Dr Nick Pears*** who supported me at the end of my PhD.

- Also, I would like to thank ***my colleagues in KACST*** who stood by me during my PhD.

- Many thanks go to ***Former Assistant of the Supervisor of the Administrative Development Management in KACST, Mr Ibrahim Al Munief*** for his help and invaluable advice.

- A special mention of ***my dear friend, Yasir Almukhaizeem*** for his great effort in helping me build the knowledge base for *BREXIT*.

- Last but not least, it would be a failure of my behalf, if I did not thank ***Dr Leila Amgoud, Dr Guillermo Simari, Dr Henry Prakken and Professor Michael Wooldridge***, who were always happy to respond to my emails, to answer my questions and help in general with any queries I had related to my research.

Declaration

I declare that this thesis is a presentation of original work and I am the sole author. This work has not previously been presented for an award at this, or any other, University. All sources are acknowledged as References. In this thesis, some materials appeared in the following published or awaiting publication papers in page vii.

Sultan Mohammad Alahmari
July, 2020

PUBLICATIONS

1. S. Alahmari, T. Yuan, and D. Kudenko, “Reinforcement learning for abstract argumentation: Q-learning approach,” in *Adaptive and Learning Agents workshop (at AAMAS 2017)*, 2017.

2. S. Alahmari, T. Yuan, and D. Kudenko, “Reinforcement learning for argumentation: Describing a PhD research.,” in *CMNA@ ICAIL*, pp. 76–78, 2017.

3. S. Alahmari, T. Yuan, and D. Kudenko, “Policy generalisation in reinforcement learning for abstract argumentation,” in *Proceedings of the 18th Workshop on Computational Models of Natural Argument (CMNA18)*, 2018.

4. S. Alahmari, T. Yuan, and D. Kudenko, “Reinforcement learning for dialogue game based argumentation,” in *Proceedings of the 19th Workshop on Computational Models of Natural Argument (CMNA19)*, 2019.

5. S. Alahmari, T. Yuan, and D. Kudenko, “Reinforcement learning of dialogue coherence and relevance,” in *Proceedings of the 19th Workshop on Computational Models of Natural Argument (CMNA19)*, 2019.

TABLE OF CONTENTS

List of Figures

List of Tables

Abbreviations	**Definition**
AAC	*Arguing Agents Competition*
AAS	*Abstract Argumentation System*
ADF	*Augmented Dickey Fuller Test*
AF	*Argumentation Framework*
AI	*Artificial Intelligence*
BREXIT	*British Exit (The withdrawal of the United Kingdom (UK) from the European Union (EU))*
CoLLeGE	*Computer based Lab for Language Games in Education*
$CS1$	*Commitment store for agent* 1
$CS2$	*Commitment store for agent* 2
DAI	*Distributed Artificial Intelligence*
γ	*Discount factor*
DL	*Dialogue Logic*
DTP	*Decision Theoretic Planning*
R_{t+1}	*Immediate reward*
KB	*Knowledge base*
α	*Learning rate*
MAS	*Multi − Agent Systems*
MDP	*Markov Decision Process*
ML	*Machine Learning*
(π)	Policy
POMDP	*Partially Observable Markov Decision Process*
PPD	*Permissive Persuasion Dialogue*
RL	*Reinforcement Learning*
RL agent	*Reinforcement learning agent*
<S, A, T, R>	$< States, Actions, Transition\ function, Reward\ function >$
TDG	*Toulmin Dialogue Game*
TL	*Transfer learning*
$(V_\pi(S))$	*Value function*

CHAPTER 1

INTRODUCTION

1 Background and motivations

Artificial intelligence (AI) has seen a significant growth in recent times. This growth can be attributed to the ever increasing interest in building machines that mimic human behaviour. Towards this, AI has been broadly defined as - systems that think like humans, think rationally, act like humans and act rationally [9]. In a computational context, Poole and Mackworth have defined AI as "*the field that studies the synthesis and analysis of computational agents that act intelligently*" [10, p. 3].

At the core of an AI system are autonomous computational entities known as 'intelligent agents'. Intelligent agents process inputs received from an environment to perform tasks such that, there is an increased chance of achieving a successful outcome. Therefore, an intelligent agent autonomously performs an action when it interacts with an environment to achieve a goal. On these lines, an intelligent agent is defined by Wooldridge [11] as - a computer system acting in an environment, which is capable of executing an autonomous action to meet its delegated objective [11]. In the last 30 years, agents and AI are said to have become more related [12]. Agents are said to act intelligently when they possess properties such as, appropriateness, flexibility and ability to learn [10]. This implies that, the agents' actions be appropriate to its goals; the agent is flexible and is capable of adapting to the dynamic changes in its environment and goals; and, the agent is capable of learning from experience to make appropriate choices under the existing perceptual and computational limitations [10].

Achieving enhanced problem-solving and AI capabilities can become extremely limited with individual intelligent agents. Therefore, these are typically achieved using multiple interacting intelligent agents to form a Multi-Agent System (MAS) [13]. Along these lines, an important feature of intelligent agents in MAS, is their ability to interact and/or cooperate with other agents for fulfilling a task [14]. While, a MAS is a group of agents interacting in the same environment to achieve their individual goals [15], these individual goals and motivations are likely be varied for each agent. Therefore, for successful interaction in a MAS that will lead to achieving the common collective aim, it is essential that agents are able to effectively communicate by cooperating, coordinating and negotiating as humans do. By allowing to communicate with each other, these interactions are known to bring flexibility to an agents' behaviour which is important for achieving a successful outcome [16]. Therefore, communication between agents in MAS involves mutual sharing of information and resolving conflict scenarios that is required for task performance [17, 18].

In recent times, 'argumentation theory' has motivated the development of effective communication strategies in intelligent agents in AI [19]. According to Wooldridge [11, p. 337], argumentation is defined as - " *providing principle techniques for handling inconsistency, and in particular, for extracting rationally justifiable positions from the inconsistent pool of arguments*". In a MAS, argumentation can be used by intelligent agents to resolve conflicts and arrive at conclusions through logical reasoning. In general, argumentation is a process in which agents attempt to agree on what to believe [11]. Argumentation is of special interest, since each agent provides a rich set of defensive information in an attempt to convince other agents of its view point [17]. In recent years, applying the principle of argumentation to MAS has received increasing attention in the research community [11, 20–23]. This has led to recent developments in the adoption of argumentation models and techniques in the AI field in general and MAS in particular [24]. Rahwan [20] states that, argumentation in MAS has two-fold benefits. On one hand, argumentation based techniques in MAS can be used to specify an autonomous agents' reasoning such as, belief revision and decision making under uncertainty. These on the other hand can also be used as a mechanism to promote multi-agent interaction. This comes from the fact that, argumentation can provide a platform for analysing, designing and implementing sophisticated interaction types between intelligent agents. These high level insights into the advantages of argumentation have led to some of the solid contributions to the multi-agent dialogue research [11, 13, 20–23, 25].

Given the importance of communication in intelligent agents for achieving their goals [11, 25], argumentation theory has provided significant inspiration for exploring different abstract and dialogue game forms of dialogues in MAS [20]. This had been classified by Prakken [26] as: argumentation-based inference and argumentation based dialogue. These two approaches are also termed as abstract argumentation and logic-based dialogue game respectively.

Towards this, Baroni and Giacomin [27] state that an abstract argumentation framework (AF), as introduced in [28], is simply defined as a pair (A,R) where A is the set of elements called arguments and R is the binary relation between A. The attack relation defines which arguments attack each other. Dung's framework [28] states that assessing whether a set of arguments can be accepted as extensions of these arguments can be evaluated through methods such as preferred and grounded extension [28]. Agents can apply this framework to argue with each other, such as playing an argument game, which is considered to be a dialectical context to exchange arguments between agents as outlined in [6, 25, 29].

On the other hand, argumentation can also be applied as a form of dialogue where agents are required to resolve conflicts of opinion by verbal means [26]. Walton and Krabbe [30] present different forms of such dialogues as dialogues which are: seeking information, inquisitive, persuasive, negotiative and deliberative. McBurney and Parsons [31] suggest that the type of dialogue depends on the information available to participants when starting the dialogue, as well as both their individual and shared goals. This research will focus on persuasion dialogue, as in a MAS, persuasion dialogue aims to resolve differing opinions of agents by trying to persuade the other party to change its opinion [32]. Prakken [33] also states that - persuasion dialogue systems regulate how such dialogues can be produced and what their outcome is. Hence, when agents engage in a dialogue system in order to debate with other agents, they are required to follow protocols and rules to maintain fairness while still remaining flexible. McBurney and Parsons [31] support the idea that these rules can make interactions between agents more flexible since each agent makes one move before deferring to the other to argue. Following these rules, the winning and losing agent can be identified based on the utility for each agent. Prakken [32] outlines that a dialogue system must have certain elements: a dialogue goal, at least two agents, a topic language and communication language, a protocol (how to move), effect rules (to specify the effect of utterances on the agents' commitments), outcome rules (identifying the dialogue outcome), turn-taking rules and termination rules. Once these are established, the agents are ready to argue with each other.

The primary objective in this research is to investigate how an agent can learn to argue. Towards this, it is known that, Machine learning (ML) facilitates learning in intelligent agents [34, 35]. In ML, the data is automatically learnt by agents which improves with experience [36]. This fundamental feature of ML plays an important role in many modern day applications such as speech recognition [37].

Among the ML methods, reinforcement learning (RL) allows an agent to independently interact with an environment and learn by experience. RL is a learning paradigm about learning how to regulate within a system in order to achieve a long-term goal [38]. Kapoor states that [39], the grand vision of AI has been to build autonomous agents that can interact with the environment and each other [39]. The formulation of learning in an agent using RL is closer to this vision. The aim of RL is to explore a policy by selecting an action which will maximise the cumulative reward through a trial and error method. Theoretically, since, RL is related to decision in the form of sequence of decisions [40], RL will enhance an agents' effectiveness to engage in either an abstract argument game or dialogue game[1]. This is due to the fact that, abstract argument and dialogue games are based on individual agents taking turns to move until the terminating move, making it a sequential environment. One of the dialogue game that will be considered in this research is the DE dialogue game [7]. The DE dialogue game is developed by Yuan [7]. The DE dialogue game is based on the the DC dialogue which was originally developed by Mackenzie in 1979 [42] and then developed by Moore in 1993 [43] as the underlying model for the debating system. The main aim of the DE dialogue game is to pick out the fallacious arguments and common errors occurring through debate [44]. The reason for adopting this game is firstly, due to the property of the DE model, which allows sufficient room for strategic formation that leads to increased flexibility in agents' while making a move

[1]Dialogue game allows agents to have a dialogue in accordance with rules [41].

[7, 44–46]. Secondly, this model has agents that were built with heuristic strategies and it offers benefits over other models in terms of computational tractability and simple dialogue rules [7, 44–46][2].

This research focuses on reinforcement learning where an agent learns independently through its interactions with the environment. In comparison to supervised learning, which needs to be explicitly told about the correct action that the agent must perform in each situation; RL is based on a rewards paradigm, where an agent learns from the positive and negative consequences of its actions by trial and error [47]. Additionally, Levin *et al.* [48] suggest that supervised learning is not suitable for optimising dialogue strategy and limits dialogue systems from being evaluated against a fixed corpus. Williams and Young [49] support this view by suggesting that, using supervised learning to generate dialogue policy could be problematic since it makes it difficult to collect a suitable training corpus. On the other hand, unsupervised learning deals with identifying the input pattern without the need to be explicit about the output. However, in unsupervised learning, the agent does not need to interact or know the environment. Unsupervised learning is therefore an exploratory approach where patterns are identified by the learning agent. Whereas, reinforcement learning concentrates on mapping an action for each state by interacting with the environment, which therefore allows the agent to observe the state change and learn [38]. Hence, Rieser and Lemon [40] argue that reinforcement learning may be able to develop a dialogue strategy. Thus, an agent could learn to argue, because a dialogue is learned by evaluating feedback so as to increase the long-term reward and exploration which is ideal considering the temporal and dynamic nature of dialogues [40].

In this research, the agents' ability to learn to argue in different argument games (abstract argument game and dialogue game) using reinforcement learning will be studied. Towards this, the agent, as a dialogue participant, requires sophisticated dialogue strategies to make a move in order to generate high quality dialogue contributions. A deep examination of state-of-the-art literature in computerised dialogue

[2]More details will be in section 2 .3.1.1.

systems, such as from Yuan *et al.* ([44, 45]), reveals that, dialogue strategies are devised and hard-wired into the computational agent (i.e. strategic heuristics). However, a key issue with this is that, an agent may not be able to handle new dialogue scenarios. Given the dynamic nature of the environment, this could pose challenging for argumentation. It would be ideal to enable an agent to search for an optimal strategy on its own, e.g. through trial and error, and thus beat the competition by being the agent with the best strategy [50]. RL tends to meet these challenges by allowing the agent to learn dialogue strategies through interactions. This could therefore offer more flexibility for the agent to make arguments (moves) by exploration.

Further to this, the challenges in enabling a learning agent to generalise its approach for adapting to new environments, motivates the choice of RL for this research. This means that the agent may not be able to find patterns in an abstract argumentation system which would allow it to transfer experience into different argumentation graphs rather than learning from the start. Therefore, it may be reasonable to move from abstract argumentation approach to proposition-based argumentation, where the internal structure of an argument is considered in order to identify patterns that would make an agent easily adapt to a new environment. In fact, with this approach, the agent could not only learn to argue but also improve other attributes, such as minimising the number of moves it takes to win and dialogue quality.

Towards this, it is necessary to consider answering the research questions outlined in the following section.

2 Research questions and objectives

Based on the outlook mentioned in the previous sections, the overarching aim of the research presented in this thesis is:

To build an intelligent agent based on reinforcement learning that is able to learn to argue against baseline agents and demonstrate improved performance.

Towards achieving this aim, the present research seeks to answer the following questions:

$\boldsymbol{RQ}_1$: *Is it possible for an RL agent to learn to win an abstract argument game and demonstrate improved performance?*

$\boldsymbol{RQ}_2$: *Do the current features of abstract argumentation systems allow an RL agent to generalise its learning approach to other abstract argumentation graphs?*

$\boldsymbol{RQ}_3$: *Is a RL agent more likely to win a DE dialogue game with minimum number of moves using a reward function to improve its performance than DE heuristic agents?*

$\boldsymbol{RQ}_4$: *Does reshaping the reward function, which takes into account attributes such as the number of contradictions and switches of focus, improve the quality of the RL agent's dialogue contributions and makes the dialogue more coherent and relevant?*

$\boldsymbol{RQ}_5$: *Does using the argument's internal structure, such as argumentation schemes and evidence support sources encourage the RL agent to transfer learning to different domains using a DE model?*

Chapter 3 of this thesis aims to address $\boldsymbol{RQ}_1$ by building new argumentative software *ARGUMENTO+* that will allow the RL agent to play against different hard-wired computerised agents. $\boldsymbol{RQ}_2$ will be reviewed in *Chapter 4* with the aim of introducing some features of abstract argumentation, such as the number of attackers. In *Chapter 5*, $\boldsymbol{RQ}_3$ will be tested by using the DE model to make the agent play against DE baseline agents. $\boldsymbol{RQ}_4$ will be discussed in *Chapter 6*. *Chapter 7* concerns $\boldsymbol{RQ}_5$ and addresses whether the RL agent could use a transfer learning method as a generalisation approach.

In order to address these questions, the research seeks to fulfil the following objectives:

$\boldsymbol{Obj}_1$: To develop an RL agent that improves its performance over time by obtaining greater rewards compared with baseline agents in an abstract argument game.

$\boldsymbol{Obj}_2$: To analyse whether using current features in abstract argumentation allows a RL agent to transfer experience from one abstract argument graph to another.

Obj$_{3a}$: To develop a RL agent that can improve its performance to the point of obtaining more wins than the DE agents.

Obj$_{3b}$: Reshape the reward function to enable a RL agent to outperform the DE agent by winning with the minimum number of moves.

Obj$_{4}$: To reshape the reward function such that the RL agent can improve dialogue quality in terms of increased relevance and coherence in the DE dialogue game.

Obj$_{5}$: To identify argument patterns, such as argument schemes and evidence sources, that can enable a DE-style RL agent to transfer learning between argument domains.

The research questions and objectives will also be reviewed and revisited in the conclusion, in *Chapter 8*, to address how well they have been met.

3 Thesis structure

This section provides a brief overview of the subsequent chapters of this thesis:

Chapter 2 presents a critical review of the literature. It explores multi-agent systems; argumentation theory, which includes abstract argumentation systems, abstract argument games and logic based dialogue games, reinforcement learning and transfer learning. Additionally, related work in reinforcement learning for argumentation is reviewed.

In **Chapter 3** *ARGUMENTO+* is introduced by building an argumentative reinforcement learning agent which plays against different baseline agents. The latter part of this chapter covers the experiments and analysis of this work.

Chapter 4 includes methods to generalise the approach in *ARGUMENTO+* and its evaluation through experiments. It also examines the limitations of using an abstract argumentation system and the necessity to move to a logic-based dialogue game.

Chapter 5 discusses the design of a reinforcement learning agent that plays against baseline agents in a logic-based dialogue game. The chapter also describes the dialogue model that was used and the results.

Chapter 6 contains the design of a reinforcement learning agent that is able to learn to improve the dialogue quality, such as coherence and relevance. The chapter also documents experiments in evaluating whether systematic view of the reward function by dropping one attribute in each can learn to improve the quality of the dialogue.

Chapter 7 provides an approach for generalisation. Since the RL agent is able to learn to argue and improve performance, it was thought beneficial to transfer its experience into a different domain. This chapter discusses approaches to use transfer learning for generalising this approach into a different domain, i.e. *BREXIT*, and how the RL agent performs with and without transfer learning.

Chapter 8 concludes the thesis by summarising the contributions and results to assess how effectively the research questions have been answered and whether the hypothesis stated in this research were or were not rejected. In addition, some suggestions for further work are presented in this chapter.

CHAPTER 2

LITERATURE REVIEW

The present research is primarily concerned with designing an agent that can learn to argue using a reinforcement learning method. In this Chapter, an overview of the current literature related to the questions and objectives of this research are presented. This includes a study of multi-agent system argumentation and discussion on inference and dialogue based argumentation [26].

This chapter consists of four sections covering the relevant literature to address the research goal in a suitably wider context.

I Multi-agent systems and intelligent agents: This section introduces multi-agent systems in general and intelligent agents in particular. It reviews autonomous agents and outlines certain properties of agents' behaviour. It then discusses communication between agents and the main objective of such communication.

II Argumentation: This section discusses the general concept of argumentation and its applications in AI. The motivation behind applying argumentation in intelligent agents is also covered here. Two approaches to argumentation are outlined: an abstract argumentation system, and a logic based dialogue game. A description of how agents argue using the argument game and dialogue model and make moves (strategy) are provided. Knowledge representation is also reviewed towards the latter part of this section.

III Reinforcement learning: In this section, reinforcement learning is introduced with a discussion on the components of reinforcement learning and approaches that are applied to solve reinforcement learning problems.

IV Reinforcement learning for argumentation: This section critically discusses related work on the state-of-the-art process of reinforcement learning for argumentation. It presents research that uses reinforcement learning in argumentation, as well as its limitations.

1 Multi-agent systems and intelligent agents

In the last few decades, theoretical and practical research on intelligent agents and multi-agent systems (MAS) has seen a remarkable growth [11, 25, 51, 52]. Researchers in artificial intelligence (AI) are constantly exploring ways to make computers more intelligent within MAS. Agents are the principal building blocks of a MAS. In a MAS, multiple autonomously acting agents interact with one another to successfully achieve their individual and shared common goal. The state-of-the-art definition of an agent is given as - *"a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its delegated objectives"* [11, p. 21]. In another place, an agent is defined as - *"An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through actuators"* [53, p. 34]. Leading researchers in this area have suggested that using an intelligent agent is the main approach in AI [9]. From the above definitions of the agent it can be summarised that, agents act directly with the environment by doing actions and receiving feedback from the environment. This process of interaction is pictorially presented in Figure 2.1.

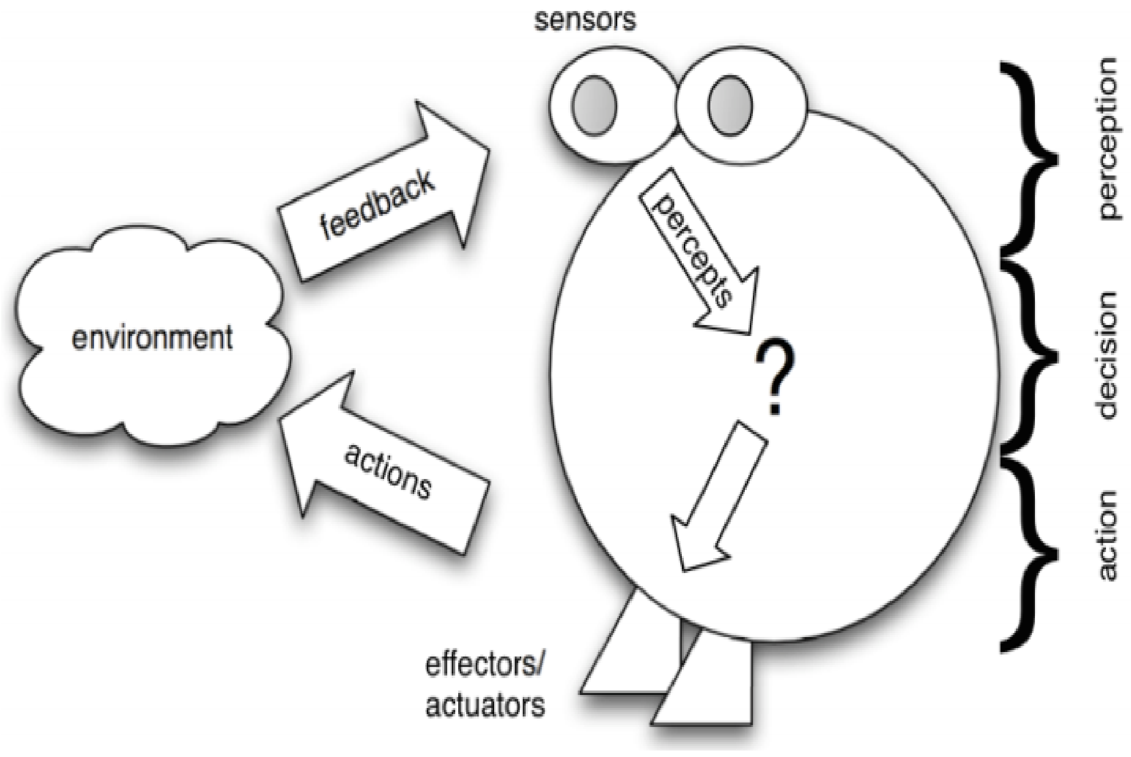


Figure 2.1: Agent and environment

Agents can exist in two forms: (i) physical, such as robots, and (ii) virtual, such as software receiving input from an environment and producing some output [52, 54]. However, a key aspect of agents is their autonomy [11]. An autonomous agent is a computer system that can work autonomously (i.e. independently) within an environment. They are enhanced to make decisions towards achieving a given goal, i.e. when an agent acts within an environment, it should act with the aim of achieving a specific objective [11, 52].

Additionally, agents are expected to demonstrate rational behaviour in the environment. In this case, rationality refers to doing the "right thing", and behaviour refers to the action that is performed over some sensory input or a sequence of sensory inputs that are received by the agent [52]. Therefore, the behaviour of a rational agent has been defined as:

> For each possible precept sequence, a rational agent should select an action that is expected to maximise its performance measure, given the evidence provided by the precept sequence and whatever built in knowledge the agent has [9, p. 37].

From the definition, it can be seen that, an agent's behaviour is affected by it ability to learn the changes in the environment and its decision on the appropriate actions that need to be taken in the given context.

Wooldridge [11] has proposed that three types of behaviours can be conceptualised in intelligent agents, namely: reactive, pro-active and social. In addition to these, other properties of behaviour of intelligent agents are proposed as, learning/adaptive, mobility and flexibility [9, 53, 55, 56]. The types of behaviour of agents are outlined in Table 2.1 [11, 52, 57, 58].

Studying agents is important because they are the key elements in building large-scale distributed systems that need to operate as a part of complex systems [59]. The building of large-scale, distributed and complex systems will therefore require a MAS, i.e. multiple interacting intelligent agents [11]. Successful interaction between

Property	**Definition**
Reactive	Response in time to changes in the environment.
Proactive	Exploration of new possibilities and taking the initiative.
Social	Communicating with other agents.
Learning/ adaptive	Changing behaviour depends on previous experience.
Mobility	Capability to move throughout the network.
Flexibility	Actions not scripted, able to choose which actions to execute and in what sequence to respond to the state of the environment.
Autonomous	Operating without direct intervention of other entities and controlling its own actions.

Table 2.1: Agent property

agents in a MAS is important to support abilities such as coordination and cooperation which play a critical role in enabling to achieve shared goals and decision-making. MAS have been extensively used in AI for dealing with large, complex systems, which is their main distinguishing feature compared to single agents. In a MAS, each agent interacts with the environment and works with other agents to achieve a shared goal. Towards achieving a goal, interaction could involve sharing and exchanging of information, cooperating or coordinating with other agents. The action performed by an agent during the process of achieving a goal could be a reactive or proactive type of action [60]. As can be seen in the Table 2.1, the type of action is dependent on the agents' exploration of the environment and the changes in the environment. Since changes in the environment are dependent on the agents' own beliefs, Wells [60] states that actions allow agents to perform effective reasoning with respect to their beliefs, their environment and the effect their actions have on their environment.

To successfully achieve a goal in a MAS, the ability to communicate in the involved agents is very important. While, it has been shown that agents are able to communicate in some cases [58], several factors need to be taken into consideration for establishing effective inter-agent communication. The considerations of inter-agent communication include: (i) choice of a protocol, (ii) illustration of a domain based on the agents' response in other domains, and, (iii) efficiency of the communication method. The use of protocol in inter-agent communication guarantees that all agents agree on a rule-base that will be followed for sending or receiving a message.

There are a number of approaches to generating high level of interaction protocols that give prominence to the types of messages that are sent during the dialogue. Most interaction protocols are motivated from argumentation theory [30]. Argumentation allows agents to reconcile conflicting information between different agents through communication [11, 61]. The main goal of MAS communication dialogue is for agents to arrive at a mutually acceptable agreement [22]. An agent could use the argumentation method to perform individual reasoning for itself for resolving conflicting evidence, or it can be used to resolve any conflicts in goals it might have by choosing to pursue only one of them [23]. A MAS uses different approaches to argumentation, such as deliberation, persuasion and conflict resolution [62] to clearly visualise the important connections between argumentation and AI, and particularly in a MAS.

2 Argumentation

Argumentation theory is a rich interdisciplinary area of research covering philosophy, linguistics, computer science, communication studies and psychology [63]. According to van Eemeren *et al.* [64, p. 5]:

> *argumentation is a verbal and social activity of reason aimed at increasing (or decreasing) the acceptability of a controversial standpoint for the listener or reader, by putting forward a constellation of propositions intended to justify (or refute) the standpoint before a rational judge.*

Rahwan [20] states that, argumentation can be seen as the rational exchange of different arguments, which are potentially contradictory, in order to achieve a clear conclusion. Therefore, the connection between arguments is perhaps the most crucial aspect of argumentation [20]. Walton expresses argumentation as a sequence of arguments, where one inference's assumption is a basis for the next [24]. Therefore, he defines an argument - as a set of propositions which consists of three parts namely conclusion, a set of premises and an inference from the premises to the conclusion [24].

In recent times, formal argumentation models have become increasingly important in artificial intelligence [19]. Formal argumentation models in argumentation have found a wide range of applications specifying semantics for logic programs, generating natural language text that supports legal reasoning, and facilitating multi-agent dialogue and internet negotiation [21, 23, 61, 63, 65]. For instance, argumentation is used in AI for handling inconsistency in knowledge bases [e.g. 66, 67] and decision making [e.g. 68–70]. Argumentation is seen as *"a promising model for reasoning with inconsistent knowledge, based on the construction and the comparison of arguments"* by Amgoud and Cayrol [71, p. 197]. Argumentation involves identifying the relevant assumptions or premises to analyse a particular problem [66]. According to Falappa *et al.* [72] argumentation is primarily concerned with premise-based evaluation of claims for reaching a conclusion.

In MAS, argumentation allows agents to resolve inconsistent information both within themselves and between agents through communication [20]. These core features of argumentation have led to a significant amount of research exploring its application in AI in general and MAS in particular [11, 19, 20, 73, 74]. Notable work on the application of argumentation in MAS includes work by, McBurney and Parsons [31] in dialogue games, Prakken in persuasion dialogue [33], Amgoud in decision making [75] and Rahwan and Larson on argumentation with game theory [76].

Bench-Capon and Dunne [19] express that, increase in the presence of information about uncertainties is the primary factor that has motivated the significant trend of applying argumentation in AI. This suggests that, argumentation was adapted to support the non-monotonic reasoning approach [19, 28]. In MAS, the primary objective of using argumentation between agents is to resolve conflicts and to inform of an agent's beliefs [11]. Towards this, autonomy [19] and rationality [77], which are the core characteristics of argumentation, have encouraged its use as a technology in developing multi-agent systems. Autonomy allows an agent to act as an individual entity through attempts at cooperating and coordinating with other agents [11] and persuading its opponents. Argumentation allows agents to act rationally. McBurney [77] states that argumentation is a form of rationality where the agent accepts statements which do not have contradictions, which means that an agent can accept an argument based on reasoning. Rationality is therefore considered as a contributing factor to the advantageous effects of communication between different agents [21, 78]. Based on these characteristic features, argumentation is considered as a useful approach for autonomous agents in not only making decisions but also enhance decision making capabilities [11, 19, 79, 80]. One of the main advantages is that an agent can act rationally. McBurney [77] argues that argumentation can be defined as a form of rationality and the agent accepts statements which do not have contradictions, which means that an agent can accept an argument based on reasoning.

State-of-the-art research in this area shows two approaches of using argumentation in AI as, inference based, and dialogue based [26]. Inference based approach deals with inferring conclusions from incomplete or inconsistent information and dialogue based approach to argumentation is aimed at encouraging interactions between agents with the objective of resolving conflicts in point-of-view. In terms of an intermediate idea of argument building, argument attack and argument evaluation, in both these approaches, it can be seen that, argumentation is a non-monotonic notion of logical consequences, in which arguments are seen as constellations of premises and conclusions [26].

In the forthcoming sections (Section 2 .1 and Section 2 .3), we elaborate on argument building with specific focus on abstract and logic-based argumentation systems [28]. Abstract argumentation frameworks take ideas of argument and attack as primitives, with nothing assumed about the internal structure of the argument or the attack relation [26]. Whereas, logic-based argumentation can be seen as a dialogue between agents to resolve a conflict of opinion. Since, a dialogue game must have protocols (dialogue rules that each agent needs to follow), and strategies (a definition of how an agent will make a move) [26, 29, 45, 81], logic based argumentation forms the basis of dialogue game argumentation which is an important aspect of this thesis. Focusing on these two areas will lead our investigation into understanding whether a reinforcement learning agent will be able to learn to argue using both abstract and logic-based argumentation systems, and study their consequences. Before that can be discussed, more analysis of abstract and logic-based argumentation systems is required which is presented in Sections 2 .1 and 2 .3.

2 .1 Abstract argumentation system

One of the most widely accepted abstract argumentation framework available in literature is Dung's framework [28]. In this framework, arguments are evaluated by defining a set of arguments with a binary relation between them (an attack relation in a directed graph). Since its original formulation, Dung's framework is not only well accepted, but is also considered to be enormously influential in making argumentation respectable in AI [26, 70].

In Dung's framework, the abstract argumentation framework (*AF*) consists of a pair (A,R). Where, A is defined as a set of arguments and $R \subseteq A \times A$ is a relation that represents *attacks* or *defeats*. *AF* can be represented as a directed graph, such that nodes are arguments and arcs represent attack relations [45], as shown in Figure 2.2[1]. *AF* is concerned with the high level abstraction of an argument and does not consider its internal structure, e.g. the premises and conclusion. This can be seen as *AF* denoting informal human reasoning in a way that allows a computer to easily perform computation on a collection of arguments. For example, an *AF* can decide whether an argument is acceptable and compute various semantics. Thus, an *AF* to some extent is able to bridge the gap between human and machine reasoning [82].

[1]This materials and definitions are taken from a lecture by Woltran and Brewka available at http://www.informatik.uni-leipzig.de/ brewka/KRlecture/AF.pdf

The AF directed graph, as illustrated in Figure 2.2, shows different arguments within attack relations. The AF in the argumentation graph in Figure 2.2 has a set of arguments: $X = \{a,b,c,d,e\}$, and a set of relations $R = \{(a,b),(c,b),(c,d),(d,c),(d,e),(e,e)\}$, where a attacks b and c attacks d. The *conflict free* set in a given AF, F=(A,R), is a set $S \subseteq A$, if $a,b \in S$, $(a,b) \notin R$. For instance in Figure 2.2 *conflict free* $(F) = \{\{a,c\},\{a,d\},\{b,d\},\{a\},\{b\},\{c\},\{d\},\{\emptyset\}\}$.

Mutual defence [28, 83] means one element in S is defended by another element in S. For instance each $a \in S$ is defended by S in F, $a \in A$ is defended by S in F, if $b \in A$ with $(b,a) \in R$, and there exists a $c \in S$, such that $(c,b) \in R$.

S is admissible [28, 84], if it is both conflict free and mutually defensive. Given an AF F=(A,R), a set $S \subseteq A$, is admissible in F, if S is *conflict free* in F and mutually defensive as mentioned before. In Figure 2.2 admissible $(F) = \{\{a,c\},\{a,d\},\{a\},\{c\},\{d\},\{\emptyset\}\}$.

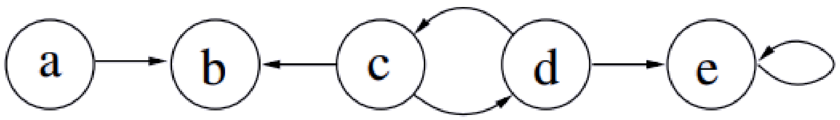
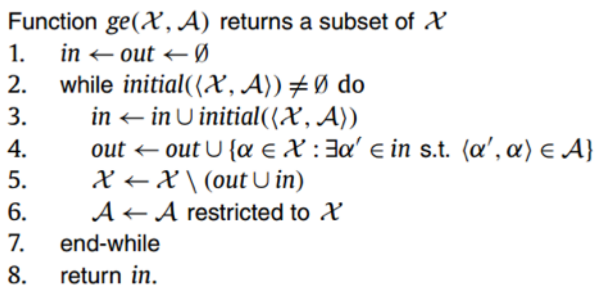


Figure 2.2: Example of arguments and relations

Extensions were introduced, to evaluate whether an argument is acceptable or not [28]. According to leading researchers in this area, S is preferred extension, where it is a maximum admissible set [11, 28, 83–85]. A preferred extension is defined as : given an AF, $F = (A,R)$, a set $S \subseteq A$ is *preferred* in F, if S is admissible in F, and for each $T \subseteq A$ admissible in T,$S \not\subseteq T$ [86]. In Figure 2.2 $pref(F) = \{\{a,c\},\{a,d\}\}$.

A grounded extension is therefore a unique extension, because it has all the arguments that are not attacked, as well as the arguments that are defended by non-attacked arguments [1, 65]. The grounded extension of an AF, $F = (A,R)$, is given by the least fixed point of the operator Γ_F:$2^A \rightarrow 2^A$. Γ_F $(S) = \{a \in A \mid a$ is defended by S in $F\}$ [86]. For instance, in Figure 2.2, $ground(F) = \{\{a\}\}$, a is in a grounded extension if a is guaranteed to be acceptable, which means there is no reason to doubt a.

On these terms, every acceptable argument is IN and every argument that is attacked, but not defended by other arguments, will be OUT. Dunne *et al.* [87, p. 459] identify an algorithm to compute the grounded extension, as shown in Figure 2.3[2] where X is a set of arguments and A is a relation between arguments. Other



```
Function $ge(\mathcal{X}, \mathcal{A})$ returns a subset of $\mathcal{X}$
1.  $in \leftarrow out \leftarrow \emptyset$
2.  while $initial(\langle \mathcal{X}, \mathcal{A} \rangle) \neq \emptyset$ do
3.      $in \leftarrow in \cup initial(\langle \mathcal{X}, \mathcal{A} \rangle)$
4.      $out \leftarrow out \cup \{\alpha \in \mathcal{X} : \exists \alpha' \in in \text{ s.t. } \langle \alpha', \alpha \rangle \in \mathcal{A}\}$
5.      $\mathcal{X} \leftarrow \mathcal{X} \setminus (out \cup in)$
6.      $\mathcal{A} \leftarrow \mathcal{A}$ restricted to $\mathcal{X}$
7.  end-while
8.  return $in$.
```

Figure 2.3: Grounded Extension Algorithm

extensions, such as complete extensions and stable extensions, are also mentioned in the literature [28, 83]. An admissible set S is a complete extension *iff* all arguments defended by S are also in S [88]. In AF, $F = (A, R)$ a set $S \subseteq A$ is *complete* in F, if S is *admissible* in F and each $a \in A$ defended by S in F is contained in S $a \in A$ is defended by S in F, if for each $b \in A$ with $(b, a) \in R$, there exists a $c \in S$, such that $(c, b) \in R$ e.g in Figure 2.2 $complete(F) = \{\{a\}, \{a, c\}, \{a, d\}\}$ [89].

On the other hand, a conflict free set of arguments S is a stable extension *iff* S attacks each argument that does not belong to S. In AF, $F = (A, R)$, a set $S \subseteq A$ is *stable* in F, if S is *conflict free* in F and for each $a \in A \setminus S$, there exists a $b \in S$, such that $(b, a) \in R$. In Figure 2.2, $stable(F) = \{\{a, d\}\}$ [86].

In the context of the present research, we consider using grounded extensions as a reward for the learning agent (discussed in Section 3). The authority of a grounded extension argument cannot be doubted compared to other arguments, as it has not been attacked [11]. This means the extension argument will be a more acceptable argument. In addition to this, there will exist a set of acceptable arguments that have been put forward by the agent [1]. After winning the game in each different episode in reinforcement learning, each agent is interested in maximising the acceptability of its own arguments (discussed in detail below).

[2]initial means an argument is attacked by another argument.

2 .2 Abstract argument game

An argument game is a dialectical context in which agents exchange arguments as game play [6]. The idea of argument games were first introduced in an official study into Argumentation by Vreeswijk in 1993 [26, 90]. Since then, argument games have become a topic of considerable research in AI. This has led to the development of several variants of argument games based on abstract argumentation framework such as, Dung's framework in 1995 [28]. Dung proposed argument games for two logic-programming semantics and defined a framework consisting of argument and attack relations as a pair (A,R) as discussed in previous section. Many formal introductions to argumentation today begin with Dung's abstract argumentation framework [26, 28]. In these, the notions of argument and attack are taken as primitive without assuming the internal structure of the argument or the nature of attacks. However, evaluation is key for considering an argument to be acceptable. Argument evaluation was first studied by Prakken and contributed significantly towards defining argument evaluation [26].

Prakken and Sartor's argument game for logic programming in 1997 [91] . They suggested an argument game for their logic programming based on Dung's grounded semantics. In 1999 Prakken [92] presented the first formal publication on argument games for abstract argumentation semantics. They put forward the game for grounded semantics as an abstraction of the game of Prakken and Sartor in [26, 91]. Wooldridge [25] proposed an argument game which was based on Dung's framework [28] in the area of computational dialectics. Yuan et al [6] argue that this game is simple because it makes it easy for players to follow the game rules. Yuan et al [6, 29] adopt this game and proposed a challenging and entertaining game known as '*ARGUMENTO*' for educational purposes to enable human and agent playing of the game.

Vreeswijk et al. in [85] developed simple and intuitive argument games for both credulous and sceptical reasoning in preferred semantics which was further developed and studied by Dunne and Bench-Capon [93]. A significant contribution for game that was developed by Dunne and Bench-Capon is its positioning of these dialogue games in the existing body of work based on the relative efficiency of propositional proof methods through the definitions of complexity of disputes [19].

Prakken suggests that incorporating a backtracking strategy in an argument game [94] could be make it more flexible and fair for players. This is because the participants can provide an explicit answer structure on dialogue where each move can constitute an attack or a surrender to an earlier move by the opponent. Our present research adopts and develops on the *ARGUMENTO* game. *ARGUMENTO* game previously adopted in Wooldridge game [6, 25, 29] did not allow backtracking for reasons of simplicity [6, 29]. Moreover, Prakken [32] suggested rules for backtracking, but it is argued that these rules were strict for the player and therefore could not allow enough room for making moves. Therefore, allowing backtracking has been left for future work which requires developing new game rules to handle backtracking.

In the present research, due to its simplicity, the game described by Wooldridge (2002) [25] is adopted. This game does not allow backtracking [29, 44]. It is also close to the way humans argue which can be summed up using the saying "try to have the last word" [28]. However, to ensure that the game is fair for both player agents, the game described by Wooldridge [25] adopts rules that each player has to follow. Each argument game has a tuple of: $G =< A, D, R, P >$ where: A is the argumentation system; D is the dialogue history which contains a set of moves made by the players; R is the set of rules that players need to follow to make a move; and P is the set of players, normally 2 denoted as 0, 1.

In [25], Wooldridge identifies six rules (R) that each player (agent) needs to follow in an argument game:

1. The first move in D is made by player$_0$ e.g. $P_0 = 0$.
2. Players take turns making moves (player 1 turn) P_i= P_i mod 2.
3. Players cannot repeat a move.
4. Each move has to attack (defeat) the previous move.
5. The game ends if no further moves are possible.

6. The winner of the game is the player that makes the final move.

Argument games such as *ARGUMENTO* [29] are based on an abstract argumentation system [28]. In an argument game the abstract argumentation (arguments and relations) are shown as a graph with a set of nodes (arguments) and edges (attack relation) (see Figure 2.4) [29]. In the first part of this research, we consider this model to investigate whether an agent is able to learn to argue at an abstract level (discussed in more detail in Chapter 3).

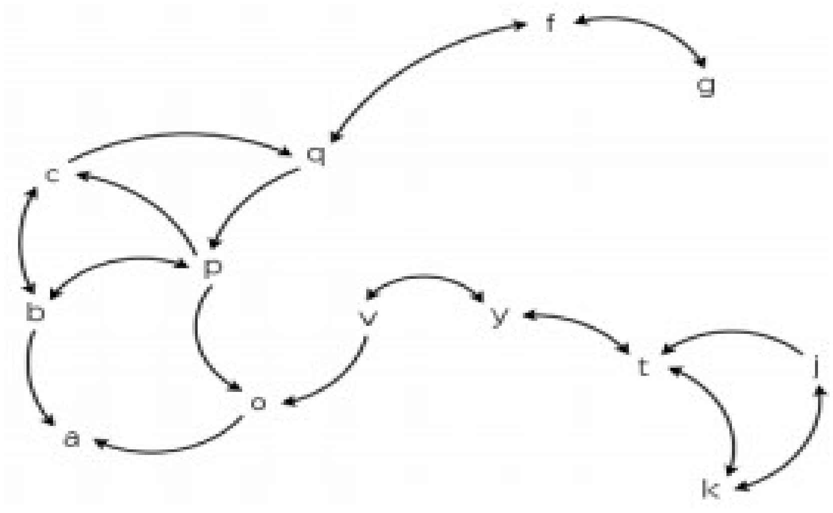
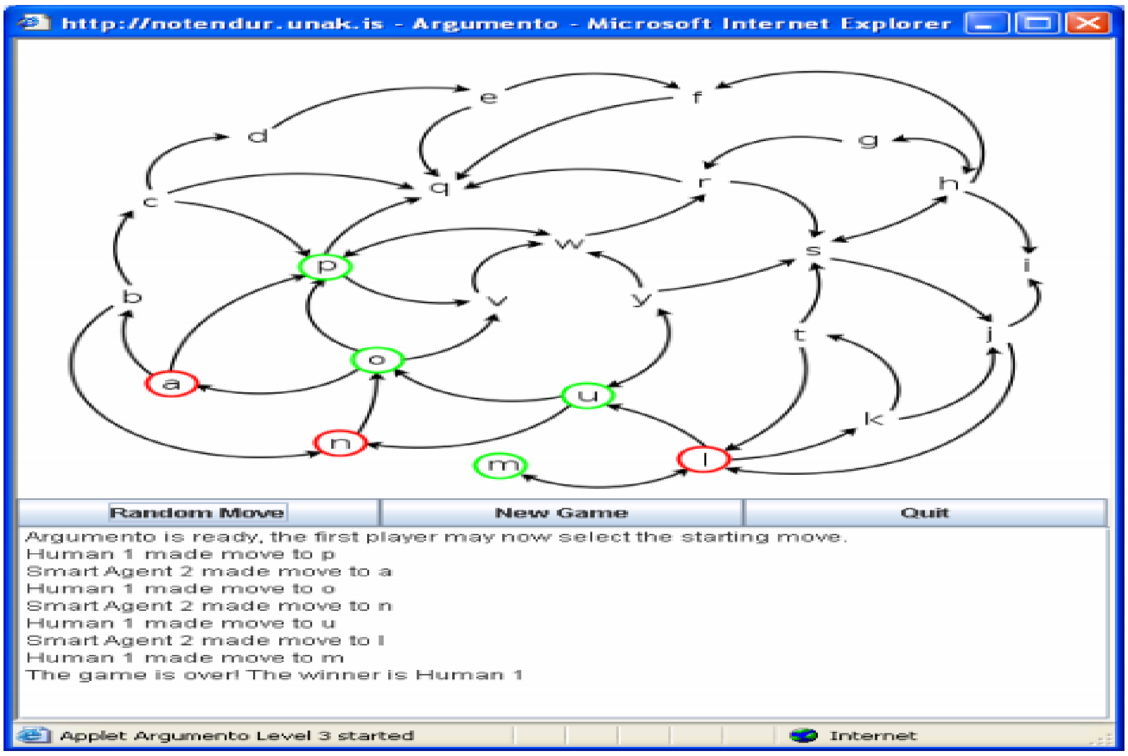


Figure 2.4: Abstract argumentation graph in *ARGUMENTO* game

ARGUMENTO is an example of an implemented argument game, presented in [29]. *ARGUMENTO* is a computer game developed for abstract argumentation [6]. The main purpose of *ARGUMENTO* is to allow agents to have an openly competitive environment to argue with each other for teaching and developing planning skills in students' [6]. An abstract argument game should consider rules (protocols), strategies (how to play), players and evaluation of the game (e.g. challenging, entertaining etc).

In an abstract argument game, each agent tries to use strategies to win the game. Therefore, in order to ensure that the game is played fairly, the agent that makes the first move is allowed to choose an argument from a predetermined set of arguments, rather than a randomly selected agent [29]. This also prevents an agent from choosing a position that does not have an attack, because this could result in the agent winning the game from the first move, which is unfair. *ARGUMENTO* is challenging and entertaining [6], and meets the objectives of an abstract argumentation system. *ARGUMENTO* can be used by students as a tool for studying argument games and abstract argumentation [95]. As the game has different levels based on the complexity of the argumentation system, it can have a motivating effect on students while encouraging them to play and win the game.



http://notendur.unak.is - Argumento - Microsoft Internet Explorer

```
Argumento is ready, the first player may now select the starting move.
Human 1 made move to p
Smart Agent 2 made move to a
Human 1 made move to o
Smart Agent 2 made move to n
Human 1 made move to u
Smart Agent 2 made move to l
Human 1 made move to m
The game is over! The winner is Human 1
```

Applet Argumento Level 3 started

Internet

Figure 2.5: *ARGUMENTO* game

The agent plays the game based on probability, i.e., they play by selecting the move with the highest probability of winning. This process has been shown in the dialogue tree shown in Figure 2.6. In this game, the agent chooses the next move with the highest utility value.[3] For example, in the tree in Figure 2.6, if agent 1 chooses p then the best move for agent 2 to win will be q. This abstract argumentation game was improved in different versions, such as Arguing Agents Competition (AAC) [29, 96]. The over all goal was to develop distributed competition between agents, in which

[3]Utility value means a probability-utility which enables an agent to select the next legal move with a high probability of winning the game [6]. Each node has a computed utility value (values are after /) as shown in Figure 2.6 [6].

they compete with each other using strategies to choose how to move. This is interesting and encourages agents to argue with each other based on choosing strategies to win the game. This aspect of making an agent to learn to argue and learn the optimal moves when playing against an opponent has significantly motivated the present research. This encouraged us to upgrade the *ARGUMENTO* game [29] to *ARGUMENTO+* as described in [1–3]. *ARGUMENTO+* uses reinforcement learning and is discussed in detail in Chapter 3.

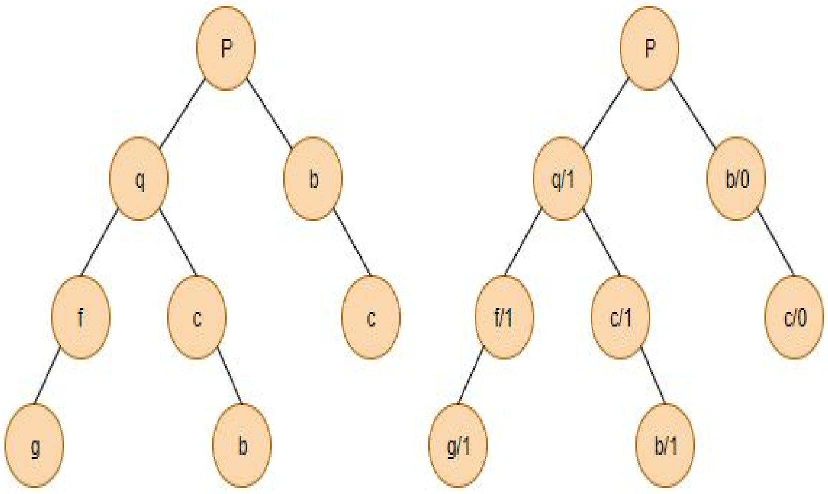


Figure 2.6: Dialogue tree based on probability

2 .3 The logic based dialogue game

The above discussion on abstract argumentation systems, has led us to undertake a closer examination of argumentation-based dialogues. The goal of an agent is to resolve conflicts. However, effective communication, by means of exchange of information between agents (dialogue) is essential for achieving this goal. Communication languages could be based on Speech Act theory [97]. Communication protocol should be fair in order to meet the primary objective of argumentation which is effective resolution of conflicts for all agents [26]. Behaviour which features strategies or tactics for the agent allows the agent to make moves or arguments [98]. In order to engage agents in argumentation, a dialogue model is required to manage the evolving argument. Literature in this area shows types of argumentation-based dialogue, such as communication languages, protocols and agent behaviour. These broadly refer to strategies or rules for deciding how the agent makes moves [26].

Dialogue systems for argumentation have been applied in different fields in AI [99] such as: AI and law [100–103], debating technologies [104–106], intelligent tutoring [42, 43, 45] and multi-agent systems [107–111]. The popular paradigm on formal dialogue in argumentation theory was proposed by Prakken [99] as, *"in argumentation theory, formal dialogue systems have been developed for so-called 'persuasion' or 'critical discussion'"*. Literature in this area shows that, this is the most widely accepted view on formal dialogue in argumentation theory [7, 30, 42, 43, 94]. In *persuasion*, the agents that are initially in conflict aim to resolve the conflict by stating the agent's belief [11]. Establishing communication between agents and allowing agents to work together to achieve this, is one of the goals of multi-agent systems.

Most communication cannot achieve a satisfactory goal with only one message being sent [107]. Also, agents need to exchange a sequence of messages when arguing with each other. This necessitates for a dialogue system similar to the example above. Along these lines, an example on negotiation is also given in [112], and another one on the type of dialogue requiring a sequence of messages to be sent is given in [30].

One of the goals of dialogue based communication in multi- agent systems is to arrive at an agreement[22]. This requires building of a dialogue system to allow interaction between agents with more flexibility and facilitate a constructive argument process. A dialogue game is one of the more flexible approaches to building dialogue systems. For instance, [29, 45, 94, 113] have a useful approach to allowing two computer systems to have a dialogue with each other.

According to the literature, dialogue model, dialogue strategy and domain knowledge representation are identified as the three main elements that underlie a dialogue system. These are important for allowing agents to communicate and debate with each other. The elements of a dialogue model are shown in Figure 2.7.

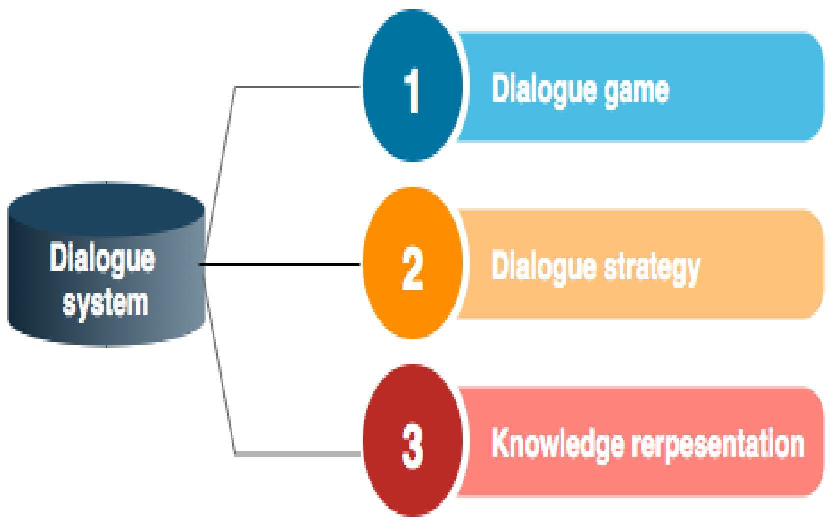


Figure 2.7: Main elements of dialogue system

2 .3.1 Dialogue game

According to McBurney and Parsons [114], a dialogue game is defined as - "interactions between two or more players, where each player "moves by making utterances according to a defined set of rules". Dialogue games are known to have a historic significance and are considered to have been in existence since the middle ages [115]. Early research in this area points to two important publications by Hamblin which mark the start of study of formal dialogue system for argumentation in the 1970s [116, 117]. The topic was originally discussed within the framework of philosophical logic and argumentation [30, 42, 118].

In recent times, constructing dialogue games has become a popular approach in AI [46]. Dialogue games have been applied in a variety of ways. Mackenzie in 1979 [42] provided the "DC" dialogue game which attracted a lot of interest in the AI research community [46]. As shown by, Amgoud *et al.*[119], Moore and Hobbs [120], Maudet and Moore [113], Yuan*et al.* [44, 45] and Reed and Wells [121], the "DC" game [42] is interactive and provides a system that is solid for interactive computer systems applications [46].

Bench-Capon and Dunne [19] state that dialog game approaches feature in many significant contributions dating back to the mid-late 1990s. For instance Gordon's

Pleadings game [100] modelled legal reasoning as a dialogue game. The Pleadings dialogue game was introduced into AI and Law [19], in which pre-trial pleading method was modelled into a structure in which the elements of a case that were agreed and disputed were defined. Bench-Capon and Dunne [19] argue that the main aspect here was that, the dialogue was used to model the legal dispute process, appealing to the notion of procedural justice, whereby a decision derives its legitimacy from the results of a properly conducted proceeding.

Another dialogue game is presented by Loui, who used dialectical approaches to non-monotonic reasoning in which one of the first theoretical limit considerations was introduced [122]. A currently active area in which these ideas have proven to be highly relevant is the exploitation of argumentation in multi-agent systems applications. Another dialogue game by Bench-Capon et al. [123] takes a different approach by providing a formal framework for the syntactic description of a dialogue game in terms of dialogical preconditions and post-conditions for each legal locution [124]. This approach has contributed significantly for automated software design and modelling of legal reasoning [125].

Amgoud et al. [107] studied dialogue games particularly in MAS where dialogue games have been used to identify dialogue rules for persuasion. They believed that it is, in fact, a more general model than the one proposed in [111] and has a specific procedure for the exchange of arguments that the former lacks. As in [111], focuses more on the relationship between beliefs and intentions in a specific type of dialogue negotiation and deliberation.

Mcburney and Parsons [124] provided a logic-based formalism for modelling of the dialogue between intelligent agents. Their approach helps to represent complex dialogue as a sequence of moves within a combination of dialogue games. Additionally, it enables the embedding of dialogues within each other. Parsons et al. [126] studied the outcome of the dialogue game between different agents based on the agent's strategy. A dialogue game for three-party dispute, where the referee has the power to assign the burden of proof to particular allegations against the opponents was suggested by Prakken [127]. Prakken et al. [128] argue that the burden of proof in that game allocations could not be disputed. Prakken in [32] contributed a dialogue game which explicitly concentrates on games for persuasion. Additionally, Yuan et. al, [46] argues that Prakken's dialogue game provided a very useful example dialogue which was applied in other dialogue game models and proved to be more relevant within MAS. Prakken's [32] dialogue game for argumentation approach is a game governed by dialogues in which two participants who disagree about the validity of one or more statements or points attempt to persuade the other party to accept their views.

Later progress in this area has led to a more concrete understanding of dialogue games as - a set of rules that regulate how players make moves in the dialogue [46]. These rules govern permissible sequences of moves, as well as the effect of moves on the commitment stores of participants, which are conceived as records of statements that have been made or accepted.

Argumentation is a way of dialogue between agents that is used to resolve a conflict. When agents make an argument, due to their internal preferences and goals, dialogue issues such as strategic issues could arise [26]. Therefore, research on argumentation-based dialogue focuses on protocols or rules of the game and agent's tactics for making moves to make the game fair and effective for all players. McBurney and Parsons [31] and Medellin-Gasque *et al.*, towards this, [129] present the elements of a dialogue game specification as:

1. Commencement rules: To determine how the dialogue might be initiated.
2. Locutions: Rules indicating which statements are permitted.
3. Rules for combination of locutions: Rules that define the dialogical context under which specific locutions may or may not be permitted or are compulsory or not.
4. Commitments: Rules that define the circumstances under which participants by their utterances engage in dialogical commitments, and thus change the contents of the engagement stores associated with the participants.

5. Rules for combination of commitments: Rules defining the combination or manipulation of commitments in the making of statements which are conflicting or additional commitments are made.

6. Rules for speaker order: Rules that determine the order in which speakers are allowed to make statements.

7. Termination rules: Rules that define the end of the dialogue.

Agents use dialogue games in order to try to convince other agents of their viewpoint [130]. A number of dialogue games have been proposed till date. In Prakken's view [131], a dialogue game is presented as persuasion with dispute. On these lines, a dialogue game is described as a conflict between proponent and opponent, where the proponent seeks to have the opponent concede the claim of the proponent, and the opponent tries to persuade the proponent to withdraw their claim. Prakken [94] expresses that in this mode, a game may not terminate logically leading to confusion and inconsistency in the agent's belief. In addition to this, the possibility of making moves could become limited due to restrictions of protocols.

Toulmin Dialogue Game (*TDG*) was introduced by Bench-Capon [132], which is based on Toulmin's argumentation model [133]. The idea of Toulmin Dialogue Game is based on recording a claim on the stack, then pushing and popping claims from the stack by adding and resolving a claim respectively. Hence, participants seek a claim on which they agree [134]. Additionally, this game also allows backtracking [32]. However, Dunne and Bench-Capon [93] argue that, each player is assumed to have only a partial view of the framework of argumentation, which is extended by elements recognised by its opponent as the dialogue proceeds. Also, if disputes between autonomous agents are considered, it is perhaps unrealistic to expect them to begin with a shared understanding of the overall framework of argumentation [93].

CoLLeGE by Ravenscroft and Pilkington [135] is a dialogue game between a tutor and a student. In this, the tutor, which is a computer, asks questions and the student attempts to answer the questions, while the tutor monitors the answers by resolving any contradictions [7]. Although, the results of *CoLLeGE* showed promise in improving students' understanding of physical motion [136, 137], it was found to have a limited set of alternatives to answer the questions and restricted room for strategies [7, 138].

Permissive persuasion dialogue (*PPD*) is a dialogue game by Walton and Krabbe [30]. This game is about one-sided argumentative dialogue with different roles for both the participants involved such that, when one builds its position, the other attacks or challenges. Some of the rules of this game are: locution rules, commitment rules and win and lose rules [7, 30]. In PPD some questions can only be answered using *Yes* or *No*, which is also restricted and it is not able to learn and explore with the purpose of learning how to argue. Yuan [7] argues that PPD does not have the facility to ban the fallacy of question begging, which prevents an agent from learning how to argue by avoid fallacious argument and make some errors during the debate.

Girle [139] introduced *DL3* which is an extension of previous work in *DL2* [140] and *DL* [141]. It is based on a baseline agent exploring issues in joint activity systems [60]. The goal of this game is to model belief revision in AI systems [142]. In *DL3* agents share the knowledge together by telling each other what to do through command dialogue. This approach is based on the Mackenzie-Hamblin system of formal dialogue [143].
Mackenzie [42] presented the formal argumentation system "DC" to investigate the fallacy of question begging, which was also adopted and edited by Moore [43]. Agents are therefore engaged in the argumentation system, but each has access to their own private knowledge base and commitment store. While, DC can be used to help players maintain coherence in their beliefs [107], Walton [144] argues that DC erroneously bans sequences of question begging. In addition, Maudet and Moore [113] assert that there are rules in DC that can prevent players from answering questions in their preferred way. Towards this, Yuan *et al.* [145] expresses that it is not clear if there are other problems related to the rules, and whether DC can prevent all fallacious arguments is questionable. This issue needs to be considered since the main goal of the system is to improve the critical thinking of students and teach them how to avoid errors when debating [146].

As a result, the DE system was designed to avoid the drawbacks of the DC system [145]. Yuan [7] further developed the system, providing a new version, “DE” which was modified for both commitment rules and dialogue rules. Considering these advantages, the present research adopts the DE game to implement an agent which is able to learn to argue based on a dialogue game. The reasons for choosing DE game for this research are therefore listed as follows:

1. This model allows enough room for strategy formation, which means the agent can be more flexible when it makes a move.
2. The DE model agents were built with heuristic strategies and the model has shown benefits over others because of computational tractability and simple dialogue rules [44–46].

Further details of the DE game are presented in-depth in the following section.

2 .3.1.1 DE model

The rules of the DC model are flexible, which means that both dialogue participants are able to make their own positions [147]. However, the DE system considers the weaknesses of the DC system, as outlined below [145]:

1. Fallacious arguments may occur during the conversation.
2. Commitment to rules may cause unnecessary conflict.
3. The rules may prevent answering of questions in the preferred way, as well as answering a challenge using an agreed statement.
4. The absence of a precondition for a challenge may lead to participants attacking a statement, which is not advanced by the other party.

As a result, the DE model was developed to address the issues of the DC model. The DE model is able to pick out fallacious arguments and common errors occurring during a debate [44]. The main point or motivation behind the prototype is to help students to improve their critical thinking skills. Students can debate with the system in a competitive environment. In [145] the same move types as DC are considered, since there are no issues with this in the DC system. The dialogue rules and commitment rules have been modified with a set of rules to prevent illegal moves by participants. In line with [145], the set of DE rules is shown in Table 2.2.

Move types	**Details**
Assertions	Statement, e.g. P, Q or compound statements e.g. P=>Q "if P then Q".
Questions	For instance, "Is it the case that P?" However, the question raised by the agent is an effective attacking technique during dialogue.
Challenges	Such as, "Why P?" As a result, the agent challenges the previous move by the opponent. This challenge technique can provide the agent with a chance to defeat the opponent.
Withdrawals	To withdrawal the statement P or "No commitment P".
Resolution demands	Resolution demand of statement P, "Resolve whether P".

Table 2.2: Rules for DE move types

It can be assumed that the move types in Table 2.2 could handle most human speech behaviour and enhance the agent's ability to simulate human behaviour. Yuan *et al.* [45] state that each party in the DE dialogue model has a commitment store, which records what has been stated and accepted for each participant during the dialogue. The commitment store contains two lists, an assertion list and a concession list. The assertion list is the proposition that the agent has stated, whereas the concession list stores the statements that have been accepted. The rules for updating the commitment store for each party are shown in Table 2.3 [45].

Initial commitment (CR_0)	**Nil value for each participant.**
Withdrawals (CR_W)	If statement P is withdrawn, it will not be included in the move maker's store.
Statements (CR_S)	Assume that agent asserts statement P, which is in the assertion list, ¬ P will be removed from the concession list if it exists.
Defences (CR_{YS})	If the opponent asks "Why Q?" and the agent responds with "P" and "P=> Q" is in the agent's assertion list, then "¬P" and "¬ (P=>Q)" have to be removed from the agent's concession list, if they exist.
Challenges (CR_Y)	If a statement "P" is followed by a challenge "P", then "P" has to be eliminated from the store of the move maker.

Table 2.3: Commitment rules

Likewise, Yuan *et al.* [45] identify dialogue rules for the DE model which each participant has to adopt through use of the model:

1. R_{FROM}: Each participant or agent can make one of the permitted types of move in turn.

2. $R_{REPSTAT}$: Mutual commitment might not be asserted until they have answered the question or challenge.

3. R_{QUEST}: The possible answers for question P can be "P", "¬P" or "No commitment".

4. R_{CHALL}: "Why P?" can be answered by withdrawal of P, a statement to the challenger or resolution demand for any commitments of the challenger which imply P.

5. $R_{RESOLVE}$: A resolution demand can happen only if the opponent has inconsistent statements in the commitment store.

6. $R_{RESOLUTION}$: A resolution demand has to be followed by withdrawal of one of the offending conjuncts or an affirmation of the disputed consequent.

7. $R_{LEGALCHAL}$: The agent can challenge the opponent “Why P?” unless P is on the assertion list of the opponent’s dialogue.

However, the most significant advantage of DE over the DC system is dealing with fallacious arguments and common errors. During the experiment [145], other fallacious issues were not found in the dialogue transcript, for instance, complex questions and appeals to emotion. This is because the knowledge base (KB) for the system was carefully constructed to avoid these kinds of issues [7].

2 .3.2 Strategy

This section discusses about the ways in which an agent can argue or make a move. Some questions are likely to arise when an agent decides to argue with other agents, such as, what argument does the agent put forward in order to persuade the other agent [148]? Therefore it is suggested that an agent should consider collaboration when arguing with other agents to solve problems. Hence, a dialogue system is based on a protocol between agents that allow a set of possible moves for the agent to make [149].

As the agent has different choices when arguing with another agent having a strategy is essential to support decision making. The agent strategy outlined in [150] can have a significant influence in two ways, namely the outcomes of the dialogue, such as who will win, and the dialogue dynamics, such as, whether the agent will end the dialogue in a small number of moves. As a result, strategies are important for an agent to make a high quality argument contribution. Examples of strategies that agents can use during argumentation include three level decisions [45, 151], game theory [150], probability utility [88] and the hiding view strategy [150] which are discussed below.

In [45, p. 10], the three level decisions strategy is used in the dialogue model the levels are identified as:

1. “*Retain or change the current focus*”.

2. “*Build one’s own view or demolish the user’s view*”.

3. *"Select a method to fulfil the objective set at level 1 and 2"*.

The level 1 decision refers to whether there should be a continuation of the attempt to substantiate or undermine a particular proposition [45]. Moore states that the current focus can be retained when answering the previous question [43], but it may be possible to not directly address the user's latest utterance while retaining the focus [45].

The level 2 decision is used to build one's own view or demolish the other view. Building one's own view means having a strategy for more support and evidence; whereas, demolishing is having a strategy to find out how to remove the other's evidence. According to Yuan *et al.* [45, p. 11], demolishing and building can be explained as follows:

1. *"a goal directed plan of questions building the computer's own view might involve removing some unwanted responses from the user"*.

2. *"the computer is using a line of questions to build the case for P in order to attack the user's view ¬P"*.

This brings to the question of whether the user should first build their own view or demolish the other's view? This is critical because it affects the way in which a move is made to complete the dialogue for winning the game. Over all, there seems to be a lack of clear evidence to suggest whether building or demolishing should occur first [43]. However, in the system described in [45], priority is given to a building strategy until the whole knowledge base has been explored. Also, the system should check first whether level 3 is available and can be applied, in which case there is no need for level 1 and level 2; whereas, if there is no level 3 then one can apply level 1 and level 2.

Game theory is another strategy could potentially be used in a debating system. In general, game theory can be defined as - a formal study of conflict and cooperation [152]. The main idea behind this strategy is to undertake precise analysis of the interaction, with the aim of predicting the outcome and mechanism design [150] (i.e., the design rules of the game). This will allow a self-interested agent to behave in the most suitable way. As a result, agents can use game theory to analyse an argument position to choose how to move. A game strategy can be used in argumentation in different ways:

1. The agent can use game theory to analyse the current argument situation to choose the best strategy.
2. Designing the rules by using mechanism design, e.g. argumentation protocols to achieve good argumentation behaviour.

Agents in argumentation play an argument game with each other, trying to win the game by attacking. The winner is the one which maximises the more acceptable arguments. Turning to another strategy, probability utility [6, 29] is used in argument games like *ARGUMENTO* [29]. The main idea of this strategy is based on using probabilistic analysis to select the next move, with a high probability of winning the game.

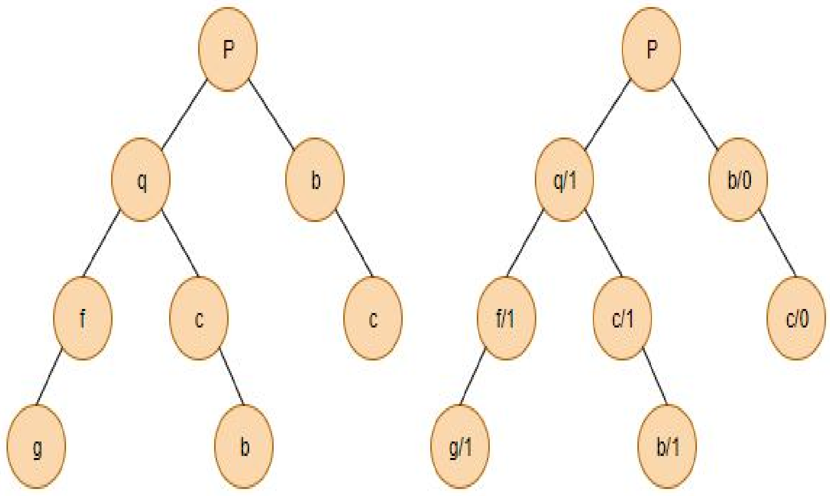


Figure 2.8: Strategy based on probability

In Figure 2.8, the root node “p” is the first move of the argument made by an opponent. Another agent calculates the branches of the sequences of nodes, some of which will be a win (or defeat) by the opponent, and others will be a win (or defeat) by the agent. The probability of each node will be calculated (0,1). If the agent moves to the node that has the highest utility (1) up to the end of the sequence, it will win the game.

Last but not least is the hiding view strategy. In [150] it is claimed that hiding an argument can be beneficial. To prove this let us consider three agents, $A_1 = a_1, a_4, a_5$ $A_2 = a_2$ and $A_3 = a_3$, as in Figure 2.9 of the argumentation graph.

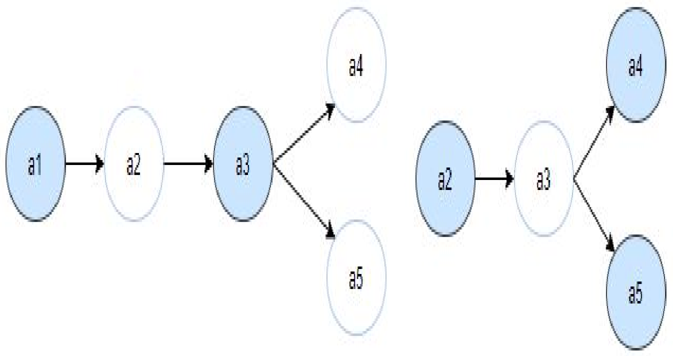


Figure 2.9: Example of argumentation graph

If an agent A_1 hides a_1 then A_1 will get two acceptable arguments, namely a_4 and a_5. This kind of strategy allows an agent to maximise the number of acceptable arguments, such as grounded extensions, which give the agent a greater opportunity to win the game and win the argument. These hidden arguments come from the private knowledge base of each agent, which is the third element of the dialogue system.

Strategy is important for agents to make high quality argument contributions. Based on the state-of-art research in the area, most computerised dialogue systems, such as Yuan *et al.*'s debating system [44, 45], adopt strategies by hard-wiring the debating heuristics into the agent. The main problem with this is that an agent may fail to deal with new dialogue situations that have not been coded and indeed it is an impossible task given the dynamic nature of argumentation. It would therefore be interesting to explore whether an agent could learn to adapt to a new dialogue scenario, which would make the agent more flexible.

2 .3.3 Knowledge representation

The literature has examples of knowledge representation for dialogue systems. Knowledge representation can be defined as expressing domain knowledge in a computer-tractable form so it can be used by an AI agent to perform well [153]. Knowledge representation is the way in which a computer expresses knowledge, either using syntax or semantics. Syntax is the configuration of the components of language constituting a valid sentence [153]; whereas, semantics concerns the facts in the world that the sentence refers to. Toulmin's model [132], as shown in Figure 2.10, is a model of argumentation that divides arguments into six component parts which are claim, grounds, warrant, qualifier, rebuttal, and backing. Yuan [7] adopts a modified version of Toulmin's schema to provide a debating system such as the DE dialogue model, as shown in Figure 2.11.

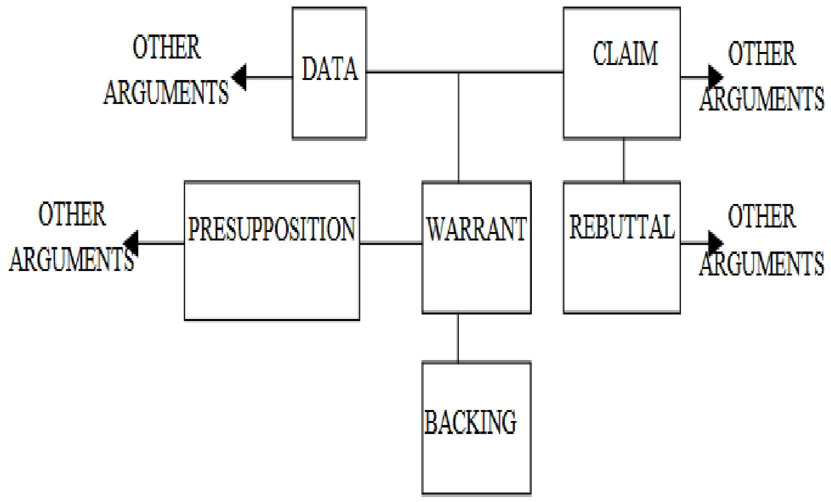


Figure 2.10: Argument schema of Toulmin (1958)

Gordon *et al.* [154] present Carneades, which is a formal, mathematical model of argument structure and evaluation that considers the procedural and dialogical aspects of argumentation. The main advantage of Carneades is that it is designed to be an open integration framework for different kinds of argumentation, using any kind of knowledge representation which is appropriate for the scheme. The structure of Carneades is based on argument graphs which consist of nodes and links representing propositions and inference relations between statements, respectively. The graphs link a set of premises to a conclusion. These premises and conclusions to arguments are statements about the world. Gordon *et al.* [154] argues that the syntax of statements is not important, as they only require the ability to determine if two statements are syntactically equal and in some way indicate the logical complement of a statement. Figure 2.12 gives an example of an argument graph.

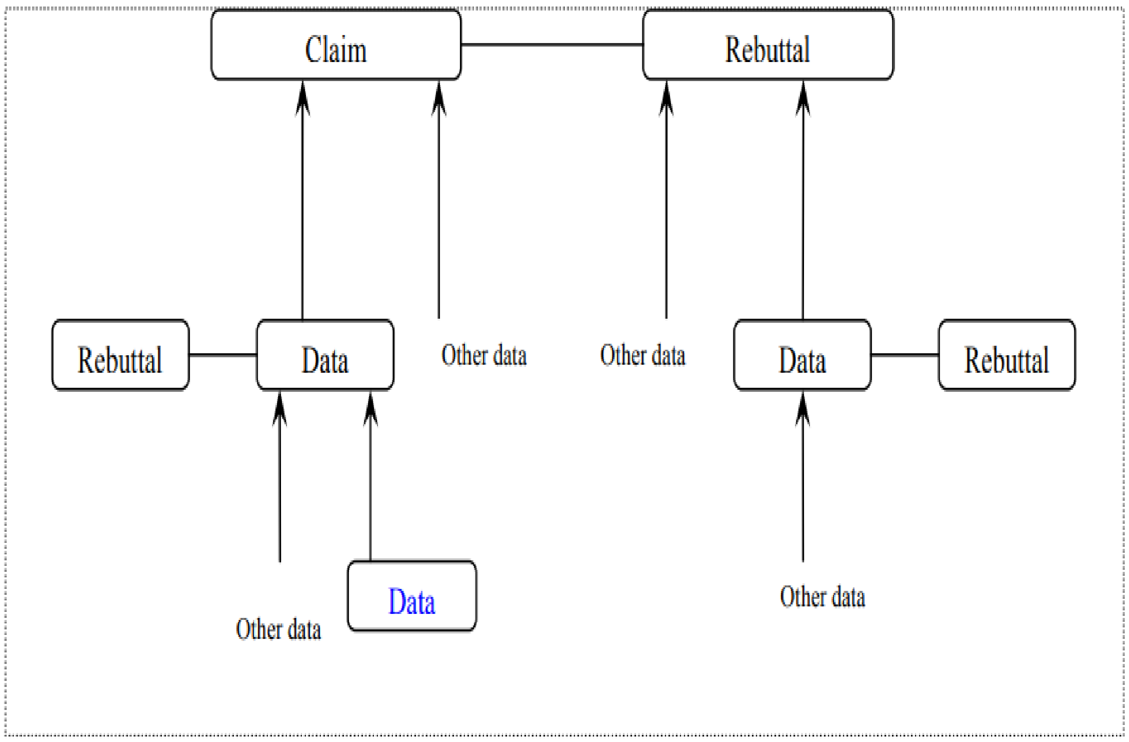


Figure 2.11: DE dialogue model knowledge base architecture

Scheuer *et al.* [155] state that Carneades is aimed at legal argumentation and uses a formal mathematical model to calculate and assign acceptability values to propositions, supporting multiple proof standards. $ArguMed$ has the same implementation for similar decision procedures, so that the argument system focuses on the legal domain [155]. Hence, according to Verheij [156], $ArguMed$ is a system for computer-mediated defeasible argumentation with a template-based interface. The main idea of $ArguMed$ is that the user initialises the argument by filling in a template. The system will keep track of the argument and the justification status of the statement that has been made. Figure 2.13 shows the $ArguMed$ interface. $ArguMed$ has three basic argument moves, making a statement, adding reason and drawing a conclusion, as well as three main data structures, as follows:

1. Statements consisting of sentences.
2. Arguments consisting of a tree of statements, where the child node of each statement node represents the reason.
3. Combining the reason with the conclusion.

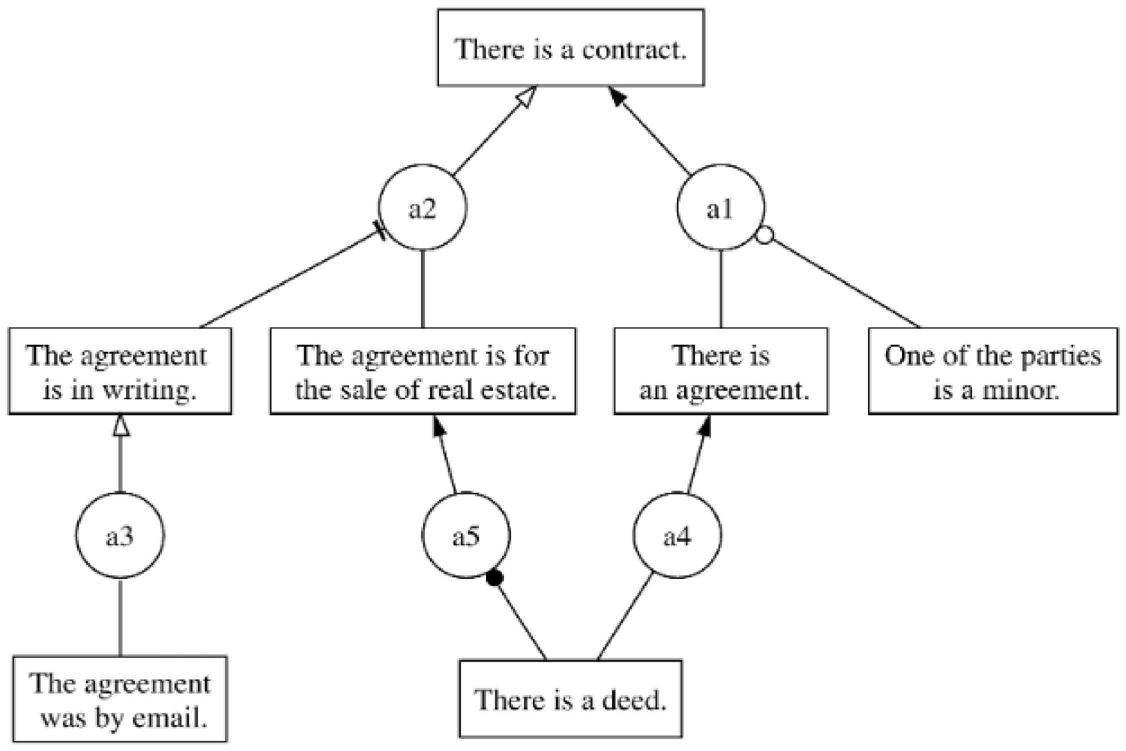

There is a contract.
a2
a1
The agreement is in writing.
The agreement is for the sale of real estate.
There is an agreement.
One of the parties is a minor.
a3
a5
a4
The agreement was by email.
There is a deed.

Figure 2.12: Argument graph

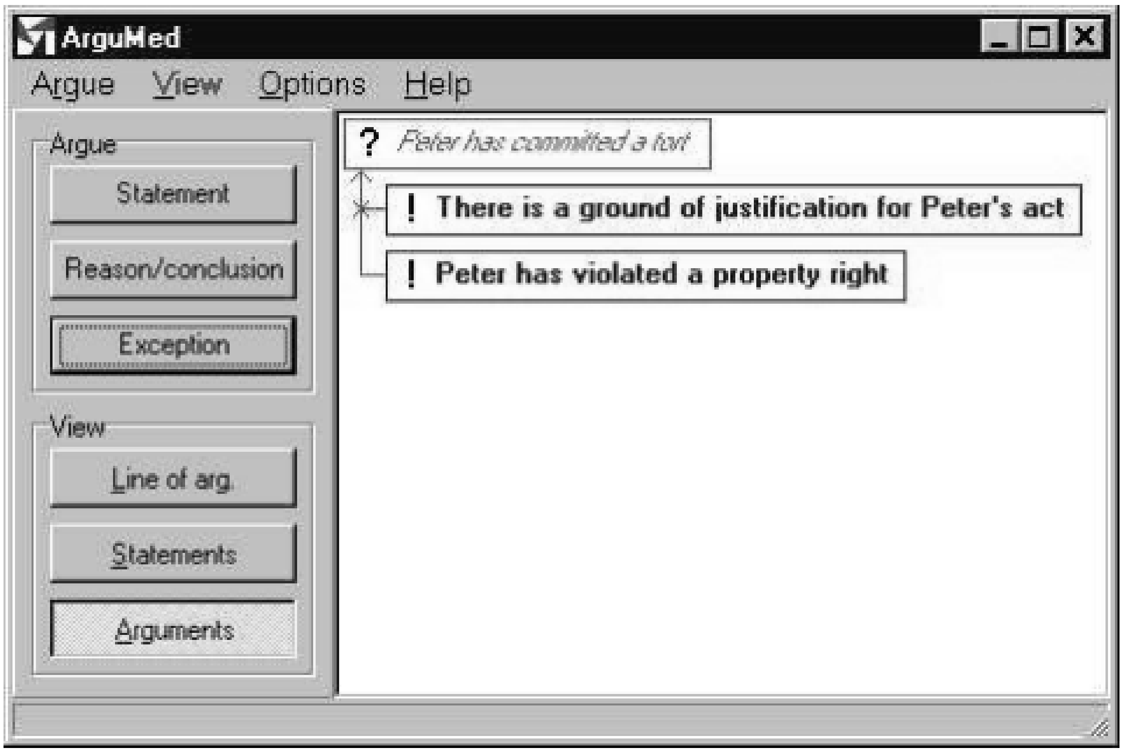
The system is flexible and easy to use [156], because it is based on a visual user interface, which is more usable than a command-line interface.

It should be noted that in this thesis, the logic that is used is a propositional logic, since it deals with a logic-based dialogue game. In argumentation theory, most dialogue games adopt propositional logic [7, 42, 43, 46, 60, 66]. In the context of argumentation, dealing with 1^{st} order logic is more challenging than propositional logic and this is therefore left for future work.

Before going deeper into the main topic of this research, which is agents learning how to argue, it would be fruitful to identify the methods of machine learning which allow agents to learn.

3 Reinforcement learning

Machine learning is becoming increasingly significant as a key technology in a number of engineering applications, as well as in studying scientific questions and



? Peter has committed a tort

! There is a ground of justification for Peter's act

! Peter has violated a property right

Figure 2.13: ArguMed Interface

theoretical problems [157]. Machine learning occurs when a computer agent learns from data. The main aim of machine learning is to improve the performance and behaviour of the agent [35, 36, 158].

Machine learning is classified as supervised learning, unsupervised learning and reinforcement learning. In supervised learning, the machine is given a set of example labelled data points (x, y); it then aims to find a function f, in the allowed class of functions, which maps to the examples [40, 159]. By contrast, unsupervised learning gives the machine a set of unlabelled data x in order to find patterns [158, 160, 161]. The third type, reinforcement learning [38, 162] deals with sequential decision-making problems, where the agent is able to interact with the environment to compute a policy and maximise the cumulative reward [38, 40, 162–165]. There has been an increasing interest in reinforcement learning, especially in the machine learning and artificial intelligence communities [166].

There are a number of ways to allow an agent to learn, and reinforcement learning is a recent development in machine learning [167]. For the learning problem, reinforcement learning allows the agent to learn to control the system to maximise the numerical values which have a long-term reward. Sutton and Barto [168, p. 1] define reinforcement learning as "*what to do – how to map situations to actions – so as to maximise a numerical reward signal*". The agent is not told which actions should be taken from the forms of machine learning, rather that it needs to explore the policy "π" which yields the maximum cumulative reward by trying them out. In reinforcement learning, agents have to interact with the environment by observing states, taking actions and receiving rewards from the environment, as shown in Figure 2.14.[4]

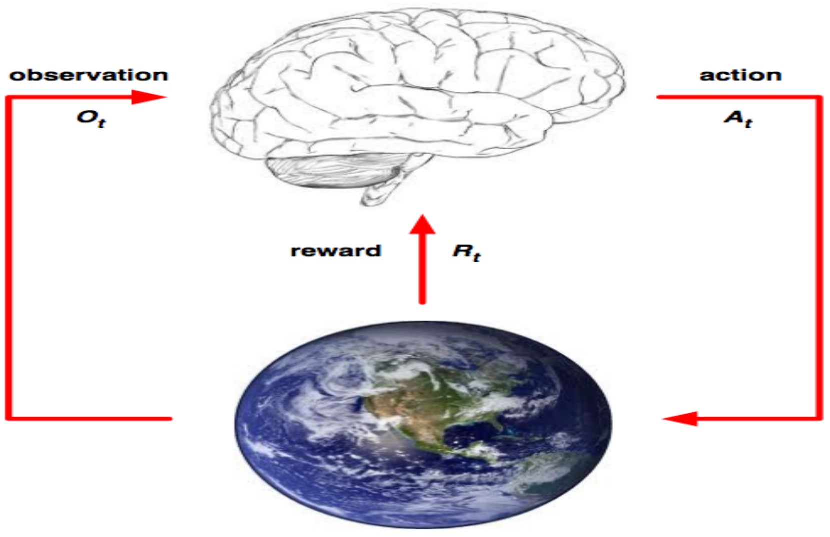


Figure 2.14: The agent-environment interaction in reinforcement learning

The agent in Figure 2.14 interacts with the environment by taking an action "a_t" to change its current state from "s_t" to "s_{t+1}". It receives a reward "r_t" based on whether it is a beneficial or non-beneficial action. The components of reinforcement learning are the policy, the reward function, the value function and the model of the environment, as shown in Table 2.4

The agent interacts with the environment to find the optimal action for each state, with actions made by the agent affecting future states in the environment. Hence, an agent is required to observe the environment while choosing an action for each state, because the agent needs to react through experience [168]. Therefore, the agent's

[4]Figure 2.14 is taken from David Silver's lectures Lecture 1: Introduction to Reinforcement Learning at http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html

Element	**Definition**
Policy	Maps perceived states of the environment to actions to be taken when in those states.
Reward function	Mapping each perceived state (state, action pair) of the environment to a single numerical value, which is a reward that denotes how good or bad a state is for the agent. The main goal of reinforcement learning is to maximise the sum of the reward received in the long-term.
Value function	The value of a state is the total amount of reward an agent can expect to accumulate over time starting from that state and following a given policy. It will specify what a good action is for the agent in the long-term. The agent seeks actions that will bring about states of the highest value, not the highest reward, since what is required is actions that will obtain the highest amount of cumulative reward in the long-term. There is a difference between reward and value. A reward comes directly from the environment, whereas values must be estimated and re-estimated again from the sequence of observations an agent makes over its entire lifetime.
Model of the environment	Simulates the behaviour of the environment.

Table 2.4: Reinforcement learning elements

performance in achieving the goal will improve over time. A Markov Decision Process (MDP) [162, 169] is used to represent the environment [162]. An MDP is an intuitive and essential formalism for decision-theoretic planning (DTP) [162, 170, 171], reinforcement learning [162, 168, 172] and other learning problems in stochastic domains.

An MDP is an environment model which has a set of states and actions which control the system's state [162]. Otterlo and Wiering [162] argue that controlling the system's state leads to improved performance, which is one of the goals of reinforcement learning. Indeed, an MDP is standard for learning sequential decision making.

An MDP is used to represent a reinforcement learning problem and show the problem in a tuple <S, A, T, R> [173], as shown in Table 2.5:

S	States	Can be whatever information is available to the agent.
A	Actions	Transfer the environment from one state to another.
T	Transition function	Represents the probability $[0,1]$ to move from s to s$'$ by taking an action a which can be identified as T(s,a,s')
R	Reward function	R(s,a,s$'$) is a numerical reward value when action a is taken, in state s and resulting in the next state s$'$.

Table 2.5: <S, A, T, R> tuple

The main goal of reinforcement learning is to work out the optimal policy (π^*) through mapping states with optimal actions. For each state there has to be an estimation with a value function ($V_\pi(S)$), which captures how beneficial it will be for the agent to apply an action in the current state [25]. The value function equation is represented as [168]:

$$V_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s] \tag{2.1}$$

Where E_π denotes the expected value given the agent follows policy π; t is any time step; and, γ is a discount factor (to determine the present value of future rewards), $\gamma \in [0,1]$. Therefore, the main target of an MDP is to find the optimal policy (π^*) which will lead to receiving the maximum reward.

There are two approaches to learning an optimal policy, model-based and model-free [38]. These models help to identify the optimal policy, which is the main goal of reinforcement learning. This research focuses on a model-free approach, as no specific model or opponent for the reinforcement learning algorithm agent to play against is assumed. Hence, reinforcement learning, with a free model method, provides different algorithms to solve reinforcement learning problems [168].

In this thesis a very wide and robust range of domains will be used, in the form of the Q-learning algorithm [40, 174, 175]. A tabular approach, rather than a function approximation as more common in SARSA, is used [38]. The advantage of a tabular representation of the Q function is that there are theoretical guarantees for convergence to the optimum policy. With function approximation, this is normally not the case. This algorithm allows the agent to learn by updating the expected Q-values. The value is a state action pair to tell the agent which action it should take [40, 168]. It is an implicit method which can update the value after each iteration. The optimal policy can then be determined. A Q-learning method can estimate state and action value functions, as seen in the following equation:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{2.2}$$

Agents use this equation in different iterations in order to achieve an optimal policy. However, the exploration-exploitation trade-off, which is balanced to obtain the optimal policy [162], should also be mentioned. This is one of the challenges in reinforcement learning [168]; should an agent explore more policies to work out the best action or should it reuse experiences to find which belief is the optimal policy (exploit)? One of the approximate solutions is epsilon greedy (1-ϵ) [38]; actions are chosen based on the epsilon greedy method, otherwise the agent chooses a random action.

Since this research considers the generalisation approach after the reinforcement learning agent is able to learn to argue, transfer learning should be discussed. This is done in the following section.

3 .1 Transfer learning

Human learners have inherent methods of transferring knowledge between tasks [176]. Relevant knowledge from previous learning experiences is recognised and applied when new tasks are encountered. Therefore, when a new task is related to the preceding task, it can be handled more easily. Transfer learning can be defined as taking knowledge from past learning assignments (source task) to enhance learning in a new task (target task) [177].

In machine learning, transfer learning methods are often dependent on the machine learning method of the original and target task(s), to the extent that they can usually be considered extensions of those machine learning algorithms [176]. This means transfer learning has recently become a common approach, as it can be used in different tasks in machine learning, instead of addressing tasks independently as previously. Transfer learning is currently developing methods to transfer experience learned in a source task to improve learning in another fresh task [176].

Since reinforcement learning is one of the machine learning paradigms and is used in this research to allow an agent to learn to argue, transfer learning could be used to improve the speed of learning [178]. The main insight behind transfer learning is generalisation between different tasks. Therefore, it would be beneficial to use it to generalise for how an agent learns, so that it can argue in different tasks.

Transfer learning has significant advantages when used in reinforcement learning [178]. These are summarised as follows:

1. It brings significant achievements to challenging assignments which other machine learning methods cannot deal with, for instance TD-Gammon [179], helicopter control [180] and robot soccer keepaway [181].

2. It can be sustained by standard machine learning methods like rule induction and classification.

3. Transfer techniques are powerful tools in accelerating learning [182, 183].

The main challenge in transfer learning concerns the evaluation of the transfer method [178]. This is due to the availability of multiple measurement options and algorithms that can assess the quality of the transferred knowledge. Measurements suggested in the literature to evaluate transfer learning include: *Jumpstart*, *Asymptotic Performance*, *Total Reward*, *Transfer Ratio* and *Time to Threshold* [178, 184, 185]. These measures are defined in more detail below (taken from [178]) (see Figure 2.15):

1. *Jumpstart*: The initial performance of an agent in a target task may be improved by transfer from a source task.

2. *Asymptotic Performance*: The final learned performance of an agent in the target task may be improved via transfer.

3. *Total Reward*: The total reward accumulated by an agent (i.e., the area under the learning curve) may be improved if it uses transfer, compared to learning without transfer.

4. *Transfer Ratio*: The ratio of the total reward accumulated by the transfer learner and the total reward accumulated by the non-transfer learner.

5. *Time to Threshold*: The learning time needed by the agent to achieve a pre-specified performance level may be reduced via knowledge transfer.

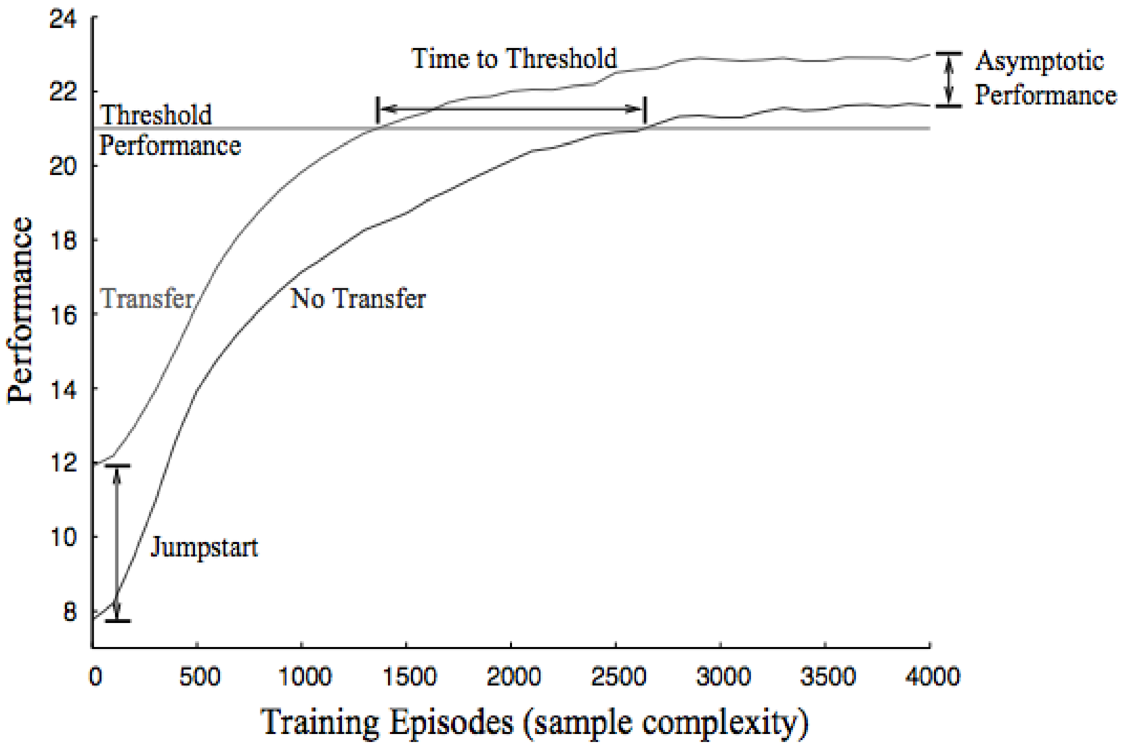


Figure 2.15: Transfer learning

[184, 186]

These metrics are used to evaluate the agent's performance and can be used to compare between with-transfer and without-transfer. Taylor and Stone [187] argue that transfer learning is successful when the above metrics are considered in the following manner:

1. *Jumpstart* is greater than zero.

2. *Asymptotic Performance* and *Total Reward* are increased with transfer.

3. *Transfer Ratio* is greater than one.

4. *Time to Threshold* is reduced through transfer.

However, Taylor and Stone [178] argue that some of these metrics, such as transfer ratio, are related to the reward structure of the target task. Due to this, transfer ratio will not be used in the current work but considered for future work. Other metrics, however, such as *Jumpstart*, *Asymptotic Performance* and *Time to Threshold*that are used for evaluation of transfer learning are suitable for this research and will therefore be considered. These offer simplicity and easy assessment using a graph to compare the reinforcement learning agents‘ performance with transfer learning and without.

Tylor and Stone [178] outline features that would enable an agent to experience successful transfer learning between tasks as follows:

1. Allowed task differences: Transfer learning can occur between tasks that have different states, actions, goal states and reward functions.

2. Source task selection: The transfer learning algorithm should allow the agent to learn from a source task then transfer it to target task. The easiest way to select a source task for a specified target assignment is to suppose that only one source task has been learned by the agent and ensure that it is used by the agent for transfer. Nevertheless, when the source task is selected, it cannot be guaranteed that it will be useful for transfer.

3. Transferred knowledge: Knowledge can be transferred as different types such as low level knowledge i.e state and action or high level knowledge e.g important features for learning. Such information might help the agent learn in the target task.

4. Task mappings: Source and target tasks should have the same states and actions with the same semantic meaning in both tasks. An agent should then be able to learn one task. Agents can learn a task then significantly decrease the time it takes to learn in another semantically similar task [188].

5. Allowed learners: The agent should use the same learning method in both tasks such as temporal difference, a common method in the literature, similarly to Q-learning.

These methods explain how to apply transfer learning between the source task and target task which will be discussed and implemented in Chapter 7.

4 Reinforcement learning for argumentation

For an agent to learn how to argue by using reinforcement learning, the first thing that needs to be identified is the dialogue model which will be used for learning. It would be interesting to engage an agent in an argument game based on an abstract argumentation system such as Dung semantics [11, 28], followed by dialogue game based argumentation [7, 44–46]. Reinforcement learning could lead to a promising paradigm for learning policies in the argumentation domain [189]. This section considers some of the related literature that combines reinforcement learning with argumentation. Exploring how to make an agent learn to argue using reinforcement learning is the main purpose of this research. To achieve this, knowledge of the states, actions and rewards which allow the agent to make decisions is necessary. We will begin this by examining some of the key literature covering research on this subject.

In recent years, there has been an increasing interest to apply reinforcement learning to argumentation. Argumentation and negotiation are different types of dialogue that are used for agent communication [30]. Negotiation between agents is an interaction with conflicting interests that aims to strike a deal; whereas, argumentation is about persuading the opponents' of a point of view in a conflict of opinions scenario. Most research focuses on negotiation [190]. At this juncture, Dung [28] clarification provides a clear distinction between negotiation and argumentation. According to this, negotiation is the operational process used to seek a solution, while argumentation is used to resolve a conflict. As a result, there is no negotiation without argumentation. In other words, argumentation is an integral part of negotiation [28].

Georgila and Traum [189] apply reinforcement learning in the agent to learn negotiation and discover the policy in different situations. The main idea of their system is to find agreement between the agent and the user in regards to a farm, aiming to agree on the most suitable temperature for the farm. Georgila and Traum [189] used a simulated user to train on a spoken dialogue corpus in the negotiation domain and then tweaked across specific cultural norms using manual rules, because the corpus does not contain culturally specific information. After evaluation, the results were consistent and the policy had been learned.

Likewise, models of negotiation based on a Partially Observable Markov Decision Process (POMDP) between a seller and a buyer are presented in [191]. Here, the main advantages and disadvantages of applying reinforcement learning in a negotiation between the seller's agent and the buyer's agent are discussed and presented as:

1. The approach is decentralised, each agent solves their own POMDP model while maintaining a belief about another agent.

2. POMDPs provide a natural method to capture the sequential nature of the process, while incorporating the new observed data, e.g. another agent's action. In addition, POMDPs allow methods to improve an agent's belief about other agents.

3. POMDPs can incorporate the effect of cultural factors in a natural way.

By contrast, another challenge [191] is how to identify the tuple $\{S, A, T, \Omega, O, R\}$ where S is the state, A is the action, T is the transition function, Ω is a finite set of observations, O is the observational function and R is the reward function. In [192] non-cooperative sides in a trading dialogue are studied, whereas in [193] it is related to reinforcement learning for learning negotiation policy in multi-agent systems, compared to other research which applies only two agents. Georgila *et al.* [193] express that to build a dialogue policy could be a challenging task, particularly for complex applications. This has motivated them to use machine learning approaches to dialogue management and specifically focus on reinforcement learning paradigm of dialogue policy. They used single-agent and multi-agent reinforcement learning for dialogue policy learning, with two agents interacting with each other and learning at the same time. The main strength in this method was that it did not require simulated users to train against or a corpora to learn from. However, Georgila *et al.* [193] used a Q-learning algorithm against a stationary environment and converged only for small state sizes. In addition, it was only tested with another agent. However, testing with a human user may provide a fruitful results. It has been previously shown in [194] that it is possible to learn a full dialogue policy by interaction with a human user.

Reinforcement learning in relation to argumentation has been studied previously. Jia [167] discusses on how the agent can learn argumentation rules in the dialogue model. However, although Jia's [167] study explored the agent learning dialogue rules, they conducted this with regard to only a few rules and not all of them. Moreover, further work is still needed to understand how to make the agent learn a strategy to argue and discover rewards when learning the strategy. Towards this, understanding how an agent learns how to move is essential for making a significant contribution to a high quality of argumentation. Rieser and Lemon [40] claim that reinforcement learning handles the dialogue strategy moves as a sequential optimisation problem. Along these lines, Chakrabarti and Luger [195] suggest that, reinforcement learning is a successful approach to allow agents to learn and adopt the optimal strategy in making a dialogue move [196].

Although most previous work has focused on two participants, agent-agent or agent-user, dialogue games involving more than two participants are studied in [190][64]. This is an example of a multi-agent system. This approach is based on the premise that, negotiations in the real world more commonly involve multiple parties. However, applying reinforcement learning to more than two agents can be complicated. Evidence for this is presented in [197], in which the issues of applying reinforcement learning in a multi-agent system in the same environment are identified. One of the issues expressed here is coordination. Since all agents have the same objective for maximising the reward, this can present challenges to maximising the same reward signals. In addition, an agent could lack knowledge about other agents that could affect collaboration. Further, if agents have opposing goals, an optimal solution may no longer exist that could also affect coordination. A dynamic environment which requires multi-sequential decisions that are complex to apply accurately adds to the challenges. Also, communication efficiency is impacted by agents that may have limited information about other agents and may not observe the actions and rewards of other agents. This could lead to an agent not being aware of the presence of other agents. In view of these challenges involving multiple agents, the present research will focus on interaction between two agents to study the learning of argumentation. The study of multi-agents is left for future work.

Ontanon and Plaz [198] presented an argumentation-based framework for deliberation between multi-agents in case base reasoning. This framework allows agents to argue about the solution to a specific problem using which agents learn capabilities that can be used to generate arguments and counterarguments. However, in this case, the agent typically learns from an example and does not learn from scratch. The example is fed by the framework which the learning agent then shares the experience by building a committee to make decisions. Wardeh *et al.* [199] assert that the framework in [198] is articulated, but it is similar to that proposed in [199]. Plaza *et al.* [200] also state that even though the agent learned only very few examples the accuracy was improved through argumentation.

To sum up, reinforcement learning helps to find an optimal action for each state to solve a problem [40, 196]. By means of learning with reinforcement approach, the agents will be able to become more flexible to adapt to new environments. This can play a significant role in learning agents to argue. Also, by allowing agents to learn the dialogue rules and how to move through the dialogue, the agent becomes more efficient in making arguments through exploration. With respect to other researchers who have studied this field, there are some areas in the literature which still require further work. Such as, Jia's work [167] focuses on learning some of the dialogue rules, but not all of them. Issues in this include how to identify an appropriate state and becoming aware of the type of move and the contents when the agent learns the strategy of the dialogue. Since rewards define the way an agent will win the game, defining the reward function in a strategy is another important issue that needs careful consideration. [5] In addition, motivating the agent to win the dialogue game in the minimum number of moves should also be considered. Oren *et al.* [201] regard confidentiality as a strategic concern. Devereux and Reed [202] support this strategy as relevant in some persuasive dialogues, since sometimes an agent reveals more of its KB than is necessary and such superfluity could be unacceptable in some contexts.

Other models for a dialogue system based on reinforcement learning for negotiation between agents only have been built such as, sellers and buyers [193, 203–205]. Most focus on negotiation behaviour. For example, learning negotiation behaviours are studied for a non-cooperative trading game [192] and [193] uses multi-agent reinforcement learning to learn negotiation rules. In [203], dialogue policies were learned from four negotiation scenarios for an agent aiming for negotiation with humans. However, it needs to work on better estimations for the opponent's persuasion strategy and to employ multi-agent reinforcement learning methods.

It is therefore of interest to study further on how an agent can learn argumentation and explore strategies to argue with other agents. This will contribute towards the development of argumentative learning agents.

5 Summary

This chapter reviews state-of-the art research in argumentation and reinforcement learning. It shows that there is research in the area of argumentation that needs to be developed by further work.

Dialogue systems should be sufficiently flexible to allow smooth interactions between agents when arguing with each other [81]. However, Yuan *et al.* state that lack of model flexibility is one of the biggest limitations to agents making arguments [95]. For instance, most dialogue systems have restrictive bipolar question types (i.e. Yes and No) [132]. This may hinder fruitful arguments between agents in making a

[5]Winning here depends on the game rules, i.e an agent persuades its opponent to accept its point of view.

decision [206].

One issue with argumentation is how to argue effectively, which is related to arguing strategies. A strategy is important for an agent to make a high quality argument contribution. Based on state-of-art research in the area, most computerised dialogue systems e.g. Yuan *et al.*'s debating system [44, 45] adopt strategies by hard-coding the debating heuristics into the agent. The main problem with this is, an agent may fail to deal with new dialogue situations that have not been coded, and indeed it is an impossible task given the dynamic nature of argumentation.

Machine learning has an important role to play in meeting these challenges. One of the popular machine learning approaches which involves agents is reinforcement learning. This is the approach used in this research to allow agents to learn how to argue and make moves. It allows them more flexibility in making an argument through exploration trial and error, which is the core of the reinforcement learning paradigm.

It is believed that learning can make agents more flexible in adapting to new environments and new dialogue situations. However, in this research, we start by using abstract argumentation and studying the consequences, followed by an argumentation-based dialogue game.

CHAPTER 3

REINFORCEMENT LEARNING FOR ABSTRACT ARGUMENTATION BUILDING ON *ARGUMENTO+*

This chapter introduces *ARGUMENTO+*, which is a software based abstract argumentation system that has been built using Q-learning approach [1, 2, 28].

The primary reason behind the development of *ARGUMENTO+* was to enable RL agents to argue with other baseline agents. *ARGUMENTO+* facilitates this and increases the chance of winning more games. However, to improve their performance and gain efficient strategies for winning arguments, RL agents have to engage in]p arguments with other agents in a suitable environment. A study of literature reveals that, this is achieved through agents that are hard-wired to make moves and heuristic strategies are built into them [44, 45]. The main issue with this approach is that, an agent may not be able to deal with new dialogue situations that have not been coded. Therefore, it is of interest in this research to make an agent learn how to argue and to explore the optimal dialogue strategy on its own.

1 ARGUMENTO+

The *ARGUMENTO+* system is a test-bed that was developed to test reinforcement learning (RL) agents. *ARGUMENTO+* is named after its predecessor *ARGUMENTO* [6]. *ARGUMENTO* designs and implements an Arguing Agents Competition (AAC), which is an open competitive environment in which heterogeneous agents are pitched against each other. The aim of *ARGUMENTO* was to promote research into the design and implementation of arguing software agents that could be extended to the public domain such as, educational, legal and social interactions [6]. *ARGUMENTO* adopts the argument game which was presented in Wooldridge [25, p. 153-154] for reasons of simplicity in enabling players to easily follow the game rules [29]. *ARGUMENTO* adopts two levels of strategies - random strategy and probability utility strategy [29]. When developing strategies for computational agents to play with human users, there is a genuine concern that the superior memory of a machine compared to human players may compromise the fairness of the game [144] and lead to users becoming frustrated by being defeated.

In the literature, agents adopt different strategies in computational dialectic systems. For instance, Yuan [7] uses Moore's three level decision making [43] for allowing an agent to be involved in academic debate. Amgoud and Maudet [207] build on Moore's theory, and present a strategy model based on the prudence of an agent. This strategy is particularly applied when it comes to disclosing the arguments and the acceptability rates that are provided to those arguments. For other dialogue types, other strategies are used in the agents [6]. For example, Grasso *et al.* [104] adopt a schema derived from Perelman and Olbrechts-Tyteca [208] in a nutritional advice giving system. They presented an agent 'Daphne' capable of providing advice on a controversial subject using dialectical argumentative techniques [104]. Ravenscroft and Pilkington [135, p. 283] use "*a repertoire of legitimate tactics available for addressing common conceptual difficulties*". Picking the smallest argument, is another strategy that is adopted by Amgoud and Maudet in choosing how to move [207]. Freeman and Farley [209] define the ordering of heuristics as guidelines for choosing moves in an argument. Whereas Oren *et al.* [201] suggest a heuristic for argumentation, based on reducing the cost of information revealed to other participants in the dialogue. Whereas, Oren et al. in [210] address strategic possibilities when the information received by an agent varies in confidentiality. Yuan et al. [29] argue that noting the details could be so highly confidential that the cost of disclosing it in the course of an argumentation dialogue would outweigh the value of winning the dialogue. To which, Oren et al. [210] provided a heuristic strategy in argumentation to enhance an agent by taking into account these costs associated with confidentiality. In *ARGUMENTO* Yuan *et al.* [6, 29] proposed two levels of strategies for a computational agent to make a decision which are a random strategy and probability utility based strategy.

In *ARGUMENTO+* an RL agent with three different opponent baseline agents was designed. These baseline agents are adopted from agents in *ARGUMENTO* that were already built by Yuan et al. [6, 29]. These three agents use three different strategies for making dialogue moves. The names of these agents are Random, Maximum-probability utility and Minimum-probability utility. The RL agent plays against these three baseline agents for evaluation purposes. The RL agent aims to maximise the cumulative reward by winning more games. Hence, if the RL agent wins the game, it receives rewards based on the number of acceptable arguments, namely grounded extensions. The reasons for adopting grounded extensions are discussed in Section 2 .1.

The key things that an RL agent needs to identify are, states, actions, environment and the reward. The classic state representation used in [211] and [11] is adopted, where states are defined by nodes in the argumentation graph and actions are the attack relations. To discuss *ARGUMENTO+* the definition of other agents (opponents) needs to be clear, and these are discussed below.

1 .1 Random agent

The Random Agent is one among the three baseline agents. The Random Agent makes moves based on random choice and uses a pseudo-random number to select an argument from the set of legally available argument moves following the algorithm below.

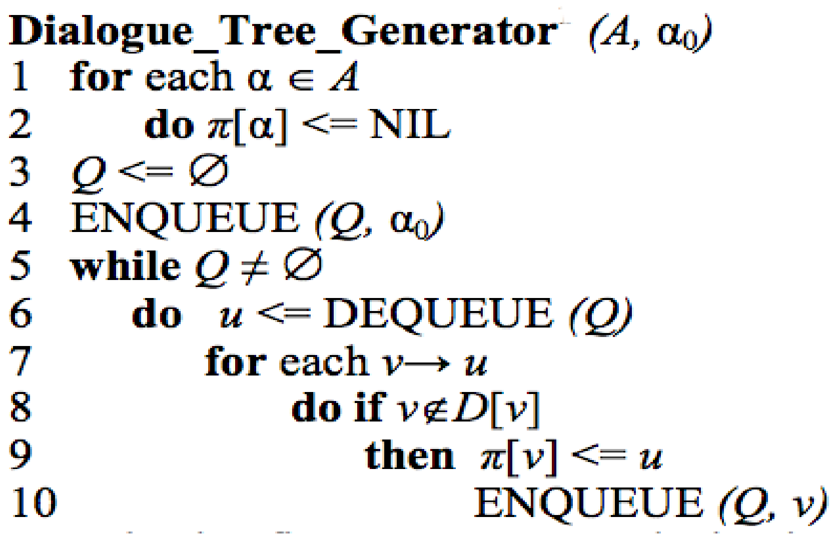
```
chooseRandomAction(argument, gameMoves):
availablemoves = getAvailableMove()
return randomly newArg∈ availablemoves and newArg ∉ gameMoves
```

A legal argument can be defined as an argument α such that: $\alpha \in A \wedge \alpha \rightarrow top[D] \wedge \alpha \notin D$. Where α is an element in the set; A is the argumentation system; D represents the stack of dialogue history; $top[D]$ is the last move in the dialogue history [6]. The relationship above means that an argument α is said to be a valid argument if it attacks the last move contained in the game dialogue history. While also ensuring that the argument itself has not been used in the game and that the game rules have not been broken.

For the Random Agent, the algorithm has to first traverse A to collect all arguments attacking $top[D]$ and then traverse the game dialogue history to ensure that this argument (α) has not been used in the game.

1 .2 Max-probability utility agent

Max-probability utility agent is the other baseline agent in *ARGUMENTO+* that uses the probability-utility strategy. The probability-utility based strategy has been previously used in *ARGUMENTO* [6]. This enables an agent to choose a legal move based on the highest probability of winning an abstract argumentation game. First a dialogue tree T is generated at a_0 using the algorithm in Figure 3.1 taken from [6]:



```
Dialogue_Tree_Generator ($A$, $\alpha_0$)
for each $\alpha \in A$
    do $\pi[\alpha]$ <= NIL
$Q$ <= $\varnothing$
ENQUEUE ($Q$, $\alpha_0$)
while $Q \neq \varnothing$
    do  $u$ <= DEQUEUE ($Q$)
        for each $v \longrightarrow u$
            do if $v \notin D[v]$
                then  $\pi[v]$ <= $u$
                      ENQUEUE ($Q$, $v$)
```

Figure 3.1: A dialogue tree algorithm [6]

Where a_0 is the first argument made by the opponent; A is the argumentation system; $\pi[a]$ refers to the parent of a; Q is the queue data structure; *ENQUEUE* and *DEQUEUE* are queue operations; $D[v]$ refers to the dialogue history up to the point of v; and, $\Leftarrow$ refers to assignment. A dialogue tree is generated (Figure 3.2) after running the algorithm in Figure 3.1 $Dialogue_{T}ree_{G}enerator(A,p)$, where A is the argumentation system and p is the first move made by the opponent.

Each path from the root to the leaf is a possible sequence of dialogues [6], where some sequences lead to a win when the utility value=1 and others lead to a loss when the utility value=0.[1] The agent is designed to choose a move based on the highest probability of winning. The utility in dialogue tree T for each node is then computed using an algorithm taken from Yuan *et al.* [6] (shown in Figure 3.3).

Yuan *et al.* [6] clarify that, P_a is a probability utility of node a; $children[a]$ is the set of children of node a; and, $depth[a]$ is the depth of node a. The utility value for

[1]The sequence of paths leading to a win is still probability, since it is not guaranteed to win based on the strategy of the opponent.

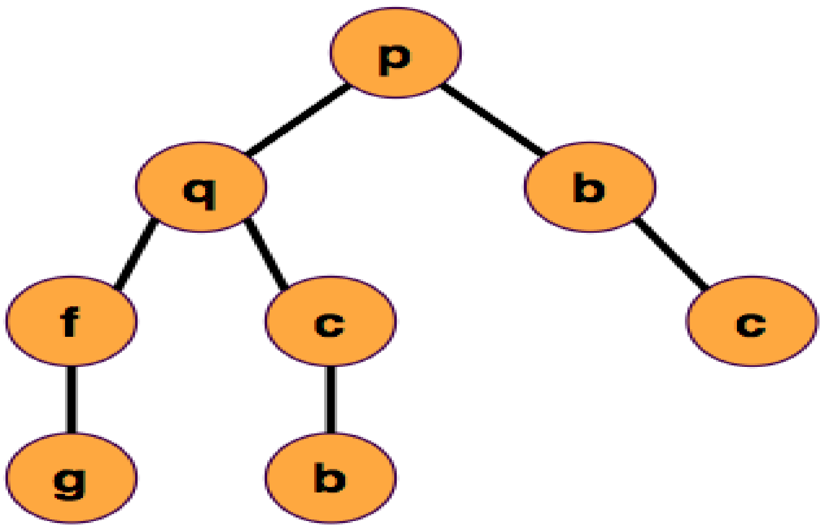


Figure 3.2: A dialogue tree [6]

```
Probability_Utility ($T$, $\alpha$)
1   $P_\alpha$ <= 0
2   if children[$\alpha$] is empty
3     then $P_\alpha$ <= depth[$\alpha$] mod 2
4     else for each $\beta \in$ children[$\alpha$]
5         do $P_\alpha$ <= $P_\alpha$+ Probability-Utility ($T$, $\beta$)
6            $P_\alpha$ <= $P_\alpha$/|childen[$\alpha$]|
```

Figure 3.3: Computing probability utility algorithm [6]

a leaf is computed against its depth (see lines 2 and 3 in Figure 3.3). An internal node's utility value is the sum of its children's utility values. As shown in Figure 3.3, lines 4 and 5 are divided by the number of their children in line 6. The probabilities of its children occurring are equal. The utility values for each node are computed using the probability utility algorithm (Figure 3.3), as shown in Figure 3.4.

Figure 3.4 shows an example of the resulting dialogue tree with utility values for each node. The two branches on the left are winning branches for the agent and the probability utility value is therefore 1. The branch on the right is a losing branch and its value is 0 [6].

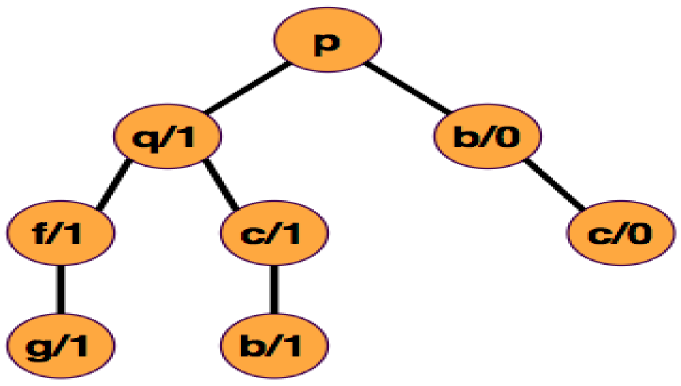


Figure 3.4: A dialogue tree with utility values [6]

1 .3 Min-probability utility agent

This agent aims to choose the minimum probability of the parent of the argument. It uses the same strategy as the max-probability utility agent, but instead of choosing the maximum probability it chooses the minimum probability. This agent was added for evaluation purposes to ensure that the RL agent could learn and outperform in different agent strategies.

2 Reinforcement learning agent

The reinforcement learning agent adopts the widely used reinforcement learning approach and is based on the Q-learning algorithm (Equation 2.2) [38] that has been discussed in Section 3 . The aim of this agent is to learn from experience and map each state with an optimal action by selecting the maximum value from the Q-table, which is updated after each episode. The state, action and reward have to be identified to make an agent learn to argue. In the state representation, state is defined as nodes in the argumentation graph, and action as the attack relation between arguments [11, 211].

The immediate reward in Equation 2.2 r_{t+1} is equal to zero, since a delayed reward is used. As the grounded extensions contain acceptable arguments that can be put forward by the agent, the delayed reward decides the number of acceptable arguments in the grounded extensions. After identifying state, action and reward; experimental validation of whether the RL agent was able to learn to argue with other baseline agents is required. The initial state, as mentioned above, is the current argument [11, 211] and the action will be the possible attacking arguments.

3 Experiments and results

To examine the performance and efficiency of the RL agent, it is necessary to conduct experiments. The baseline agents discussed above can be paired up with an RL agent to facilitate the evaluation. *ARGUMENTO+* has to first generate a text file which contains an argumentation graph with nodes and edges to upload the argumentation graph for simplicity. The argumentation graph in Figure 3.8 is adopted, since it is a common argumentation graph example [11, 25, 85, 212] with a set of arguments defined as, $Arg = \{a,b,c,d,e,f,g,h,i,j,k,l,m,n,p,q\}$ and relations R ={ *(c,d), (c,a), (d,a), (d,e), (d,b), (e,b), (g,d), (g,p), (h,a), (h,e), (h,p), (i,j), (i,n), (i,e), (j,i), (j,n), (k,l), (l,m), (m,k), (m,c), (n,p), (n,f), (p,l), (p,c),(p,q)* } [2]. *ARGUMENTO+* reads the graph from the text file and asks to play the argument game between the RL agent and one of the baseline agents. The discounted factor is required to be configured between 0 and 1, which is believed to have a direct impact on agent behaviour. It then needs to choose which opponent should play against the RL agent and set the simulation running.

The data is saved in two files, a .csv file which contains the final rewards for each agent and a text file in which the reinforcement learning agent stores the final Q-table data. The main objective of the experiment is to evaluate how the RL agent behaves and how effective the learning is. The data is collected from three different game settings:

1. RL agent against Random agent.
2. RL agent against Max-probability utility agent.
3. RL agent against Min-probability utility agent.

The simulation configuration is:

[2] For instance (c,d) means c attacks d

1. For episode 0, epsilon[3] ϵ= 0. Run the game 10 times then take the average of the final rewards. At this stage, all values in the Q-table are zeros.

2. The agent starts the exploration and updates the Q-table accordingly.

3. For every 10th episode of exploration, epsilon is set to 0 to evaluate the agent's policy, and an evaluation is made by running the game 10 times then taking an average of the final rewards for each episode.

30,000 game episodes were run against each of the game settings. To allow the rewards to be displayed in a graph, the average final rewards for every 500 episodes was taken. The performance for both agents in each setting are plotted in Figures 3.5, 3.6 and 3.7 respectively.
The Figure 3.5 shows that the RL agent learns quickly from the beginning to the 1,500th episode. After that, the rate of learning is found to decrease. It outperforms the Max-probability agent at some points as also seen in Figure 3.5. Ultimately, the performance of the Max-probability agent was found to drop, which can also be seen in the the learning curve. However, the mean reward of RL agent was 0.68 and the standard deviation was 0.062 while the mean reward of MaxProb agent was 0.82 and standard deviation was 0.053 indicating that the MaxPRob agent outperformed the RL agent.

Likewise, the learning curve is also recognised when playing against the Min-probability agent, as shown in Figure 3.6. The reward stabilises at around 1.3 for the rest of the episodes. After the learning phase, the performance of the opponent agent rapidly decreases to 0.6 indicating that the performance of learning agent was better. The mean reward of RL agent was 1.18 and standard deviation was 0.062 while the mean of MinProb agent was 0.56 and standard deviation was 0.093.

However, when the RL agent played against a Random agent, the rewards of the learning agent dropped from episode 0 (rewards = 1.15) to 1,000 (rewards = 0.8). It then found it hard to learn and the performance was found to consistently reduce

[3]Epsilon ϵ is one of the approximate solutions which calls epsilon greedy (1-ϵ) [38], where an action is chosen based on the epsilon greedy method, otherwise the agent chooses a random action. When ϵ=0 the agent is exploited not exploring.

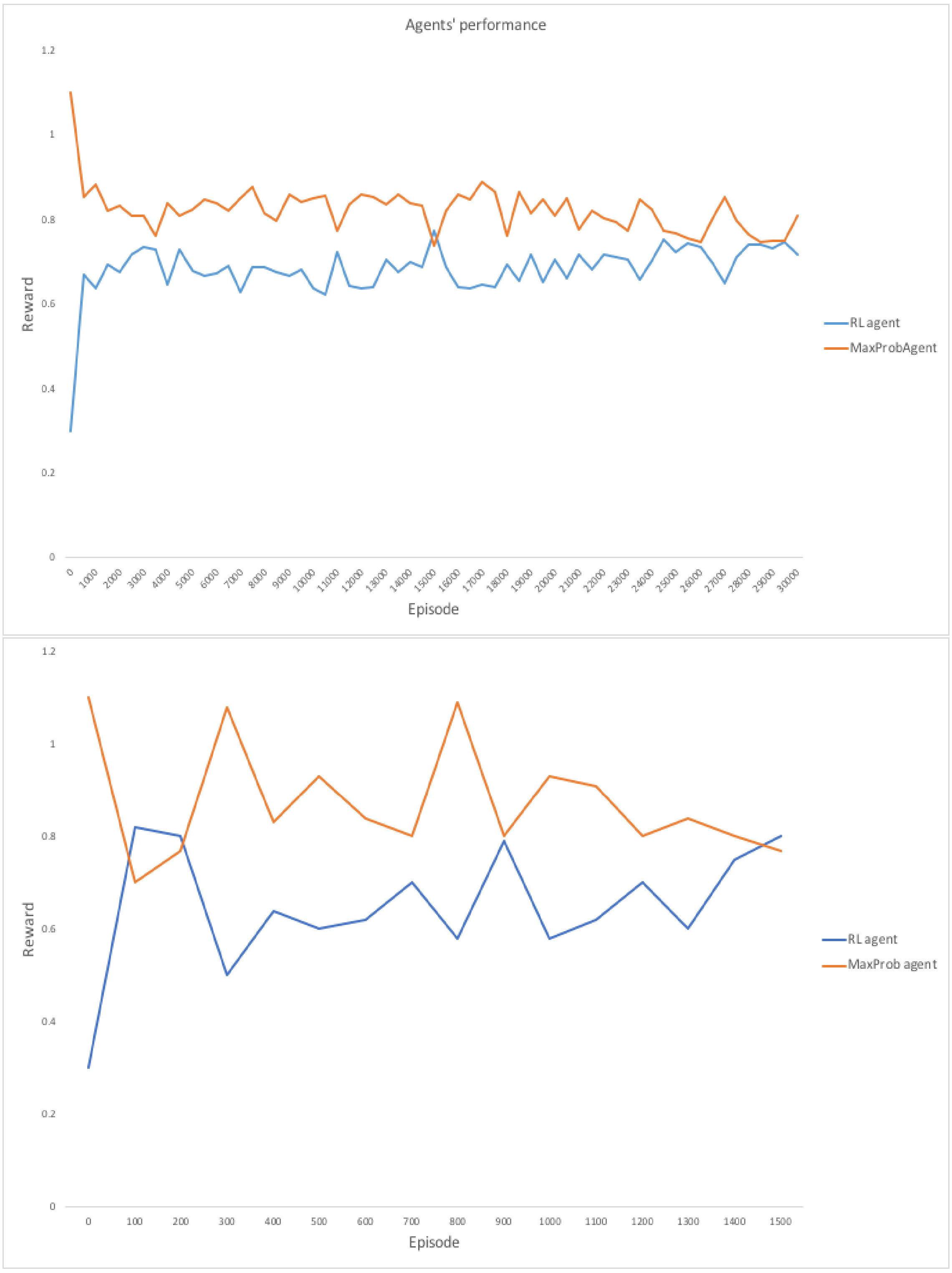


Figure 3.5: RL agent against Max probability agent (1500 episodes in the lower side)

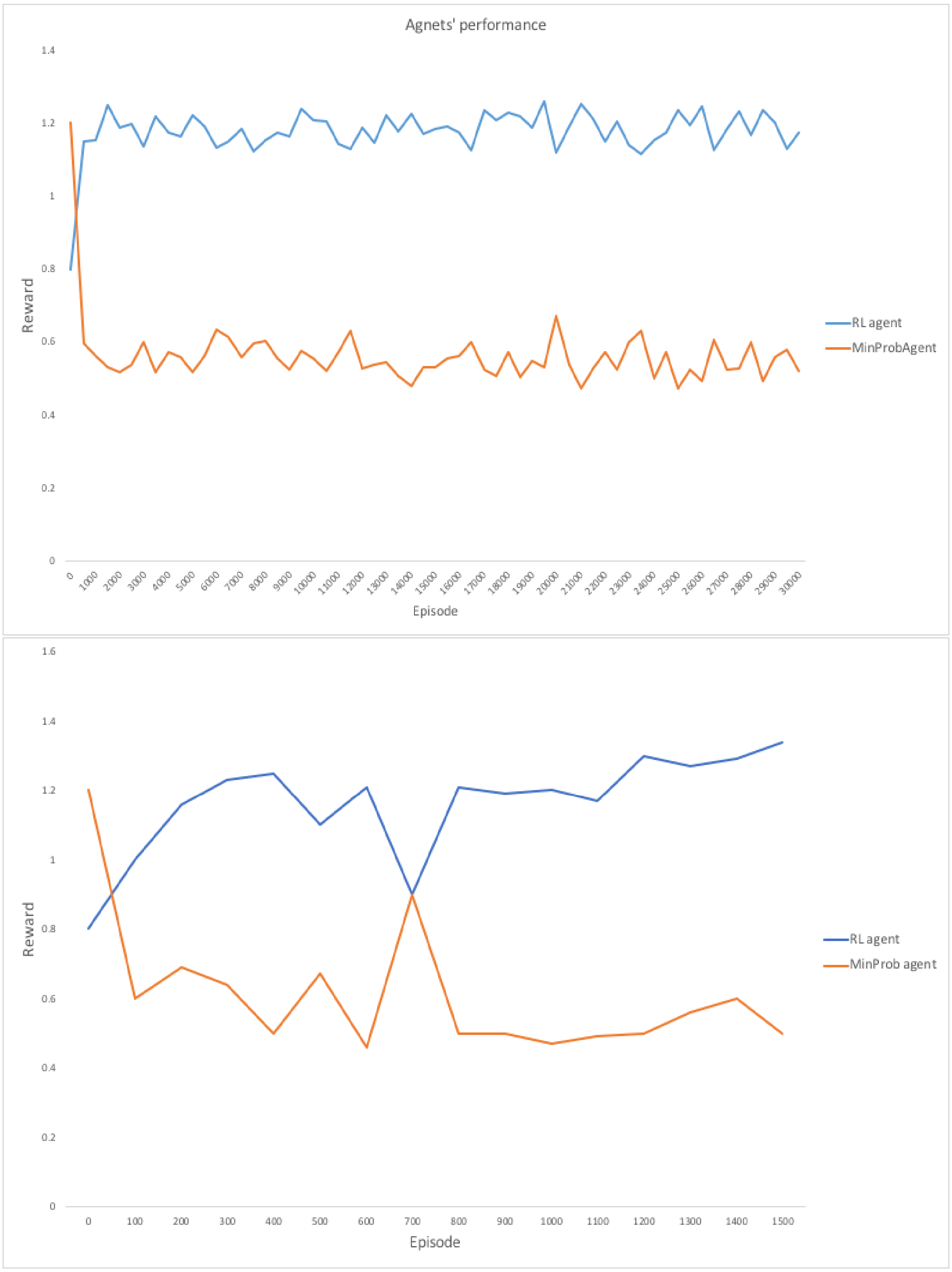


Figure 3.6: RL agent against Min probability agent (1500 episodes in the lower side)

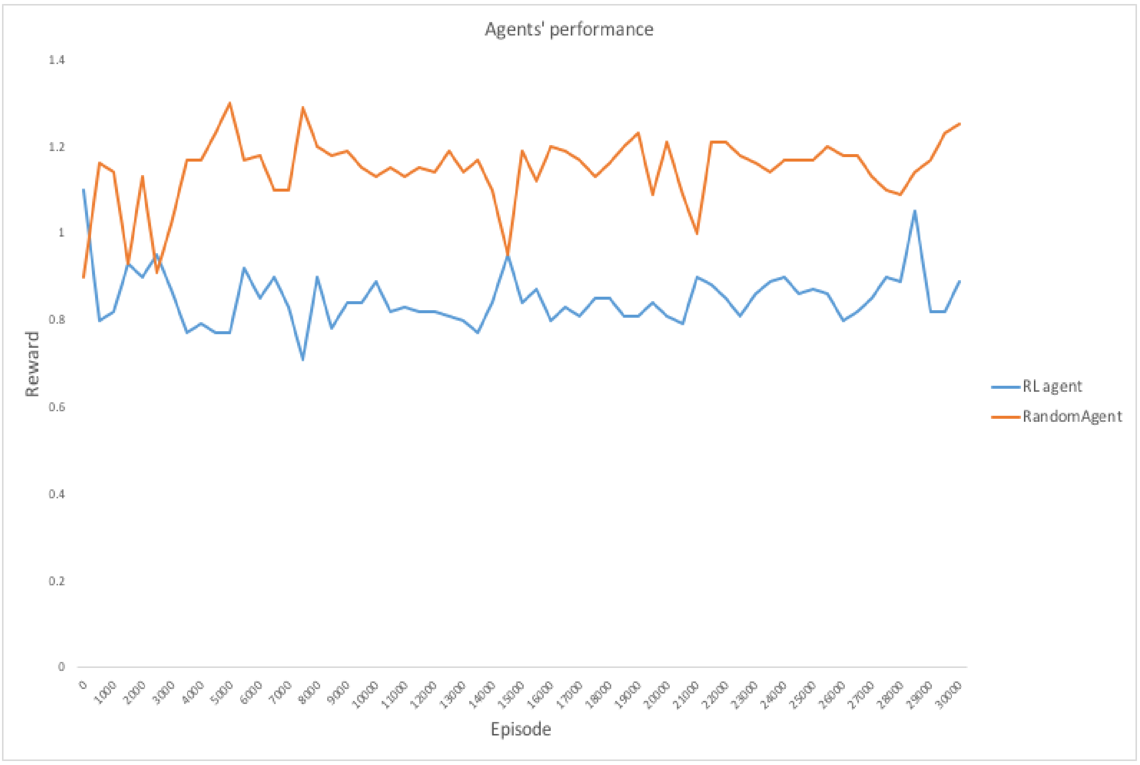


Figure 3.7: RL agent VS Random agent

(see Figure 3.7). The final mean reward of the RL agent was 0.85 with a standard deviation was 0.064, while, the mean reward of Random agent was 1.15 and standard deviation was 0.079.

To investigate this the starting argument and the agent that is meant to start the game was prefixed. Argument "*b*" was then put in the argumentation graph as the initial argument (as shown in Figure 3.8), while the RL agent always started the game.

Unfortunately, the performance of the RL agent against the Random Agent continued to be unsatisfactory, as shown in Figure 3.7. The negative results from the experiment encouraged more analysis of the game scenario. The analysis showed that the problem was related to state representation in the current RL agent's design. For example, Figure 3.9 shows all possible argumentation paths that are rooted at argument "*b*", when the Random agent chooses argument "*e*", the RL agent should choose the argument with the maximum Q-value from "*d*", "*i*" and "*h*" in the Q-table in Figure 3.10. In this case, the Q-values for "*d*", "*i*" and "*h*" are 12.17, 19.38 and

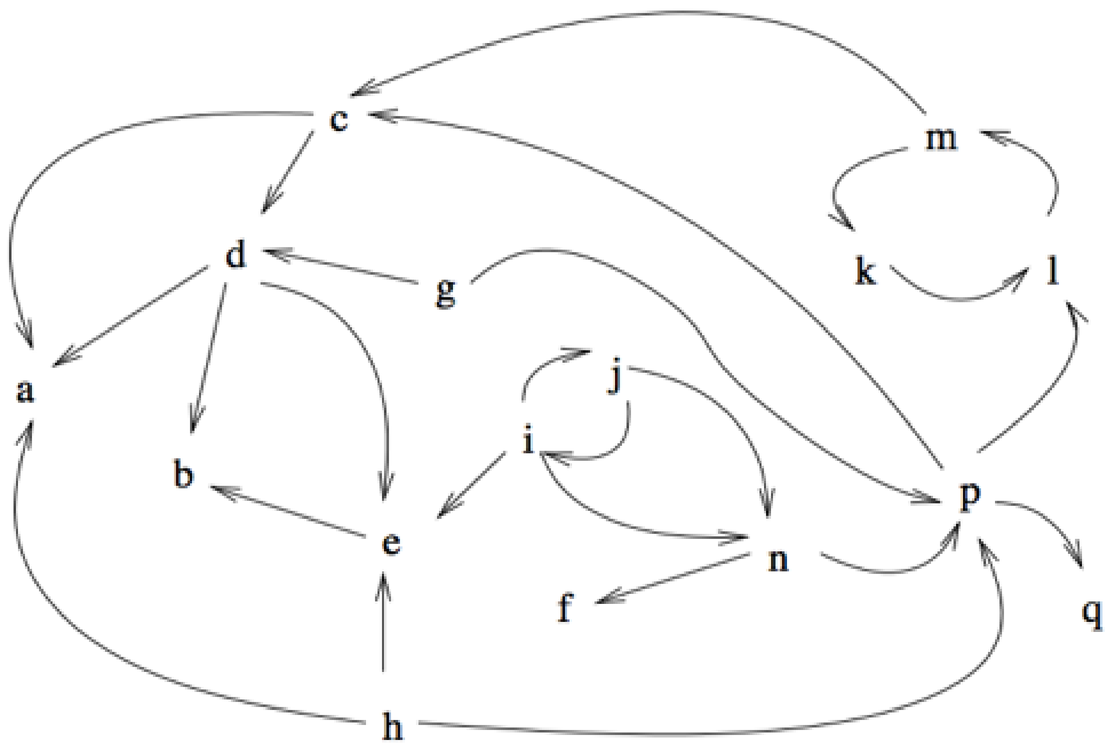


Figure 3.8: Example of abstract argumentation system

2.03 respectively, so argument "i" will be chosen because it contains the highest value. However, argument "i" will cause the RL agent to lose the game because the last move is the Random agent's. Instead, if argument "h", with the lowest Q-value is chosen, it will lead to a win.

Therefore, the question is, why a bad move like picking argument "i" in this case has a high Q-value. The reason for that is that there are some other occurrences of state "i" in the tree which give a much better payoff, and on aggregation, they lift the Q-value for state "i". Fundamentally, the classical approach to state representation [211], which is adopted here by the RL agent, falls short for the argumentation domain.

To address this issue, a more sophisticated dialogue state representation is required that ensures that a state is not repeated. After the previous experiments, it was found that combining the level of the tree with the current and previous arguments was more useful in helping the RL agent to recognise the correct action in different states. In addition, each agent has a unique ID which could also help to identify state representation. Therefore, new state representations were proposed, ($levelOfTree$,

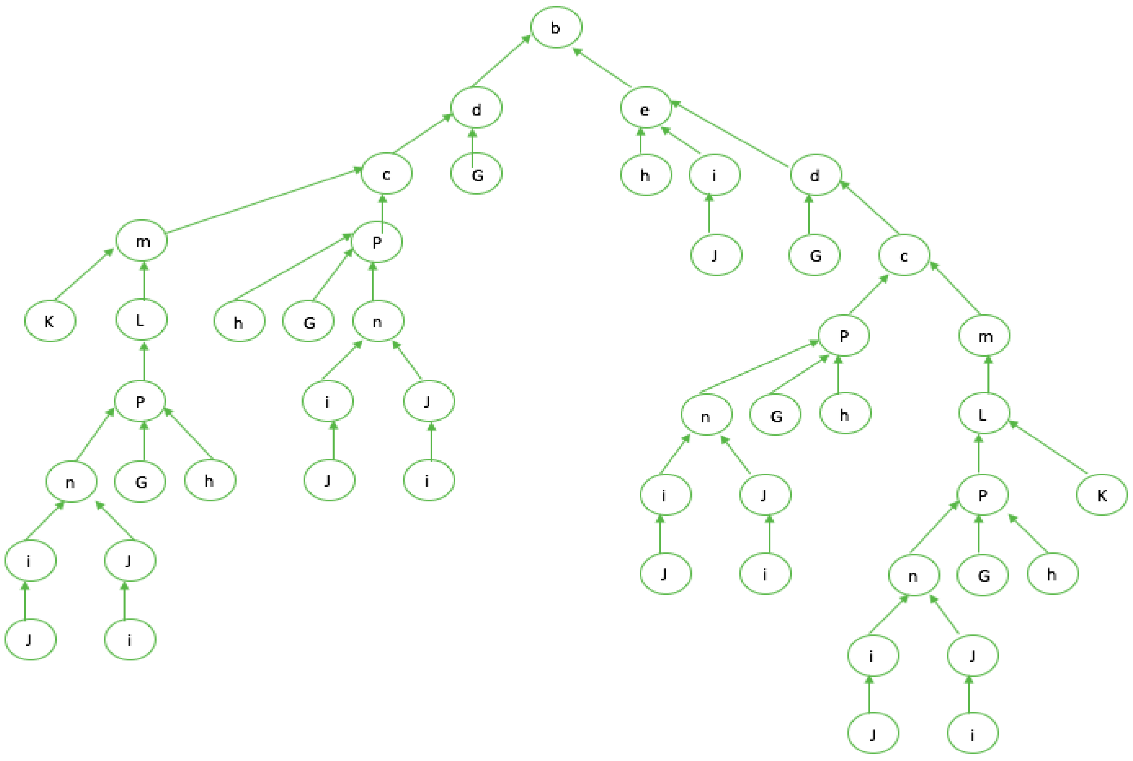
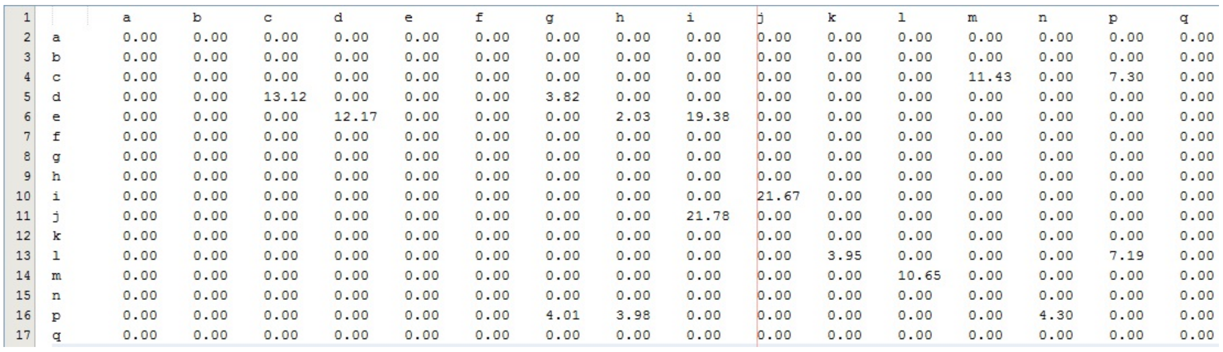


Figure 3.9: Arguments tree rooted at argument "b"



	a	b	c	d	e	f	g	h	i	j	k	l	m	n	p	q
a	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
b	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
c	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	11.43	0.00	7.30	0.00
d	0.00	0.00	13.12	0.00	0.00	0.00	3.82	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
e	0.00	0.00	0.00	12.17	0.00	0.00	0.00	2.03	19.38	0.00	0.00	0.00	0.00	0.00	0.00	0.00
f	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
g	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
h	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
i	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	21.67	0.00	0.00	0.00	0.00	0.00	0.00
j	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	21.78	0.00	0.00	0.00	0.00	0.00	0.00	0.00
k	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
l	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.95	0.00	0.00	0.00	7.19	0.00
m	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	10.65	0.00	0.00	0.00	0.00
n	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
p	0.00	0.00	0.00	0.00	0.00	0.00	4.01	3.98	0.00	0.00	0.00	0.00	0.00	4.30	0.00	0.00
q	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Figure 3.10: The Q-table

agentID, *currentArgument*, *previousArgument*). It was found that these state representations helped the RL agent to improve performance. Hence, the results were promising and it can clearly be seen that the RL agents perform better against different baseline agents. This is demonstrated in Figures 3.11, 3.12 and 3.13 where the performance of the RL agent is shown in blue and the baseline agent shown in orange.

The RL agent performs better and learns better with this new dialogue state representation. In addition, it should be pointed out that we discovered that the learning

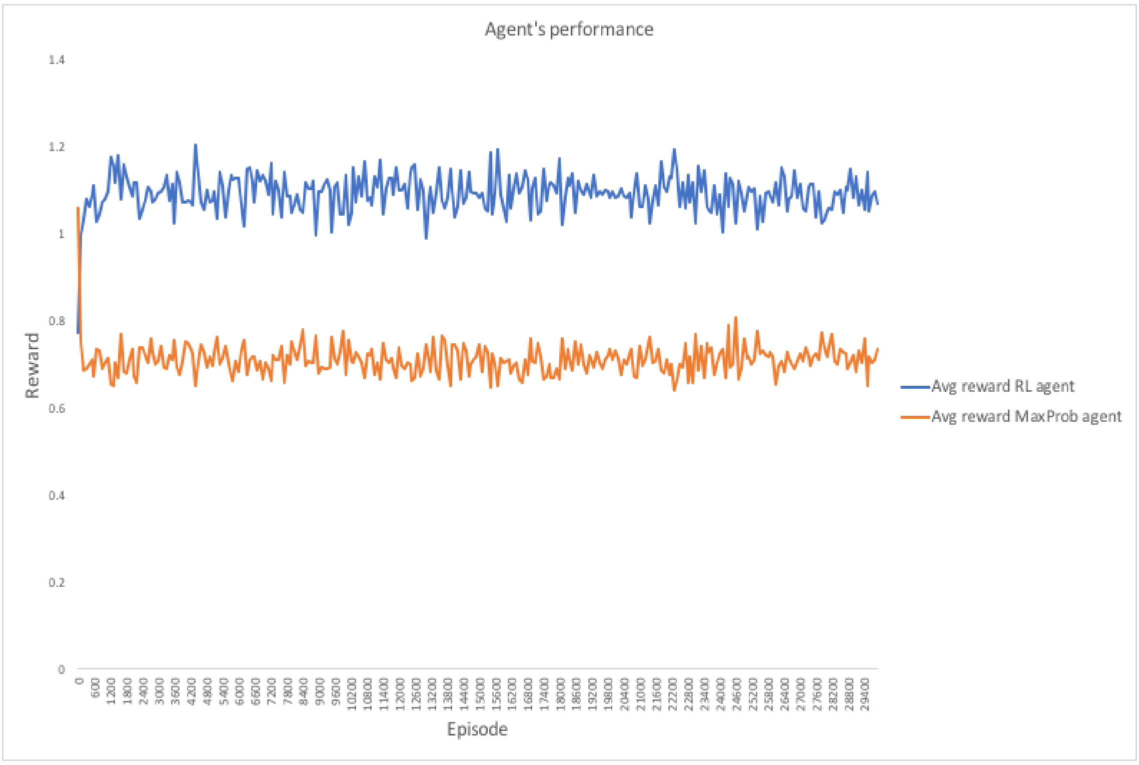


Figure 3.11: RL agent against Max-Probability agent

rate and the discounted factor had a positive relationship. This means that when the learning rate is close to 1, the same as the discounted factor, the RL agent will perform better. This be seen in Figure 3.14. The reason for the positive relationship is due to the agent being interested in the delayed reward to accumulate more acceptable arguments in the longer term.

Based on the results from the RL agent playing the argument game against different baseline agents, there are factors which may impact the RL agent's performance. Initially, the discounted factor was considered to be close to one, as the delayed reward was used. We were interested in using the delayed reward to help the RL agent learn to win more games and consequently learn to argue by maximising the cumulative reward. Another impact on performance is strategy. The RL agent adopts the Q-learning algorithm, as shown in Equation 2.2, choosing an action based on the maximum Q-value in the Q-table. State representation has a significant impact on the RL agent's performance is due to the Q-value. Comparing the classical state representation [11, 211] and the proposed state representation introduced in this chapter, the latter has a significant impact on the RL agent's performance. This is

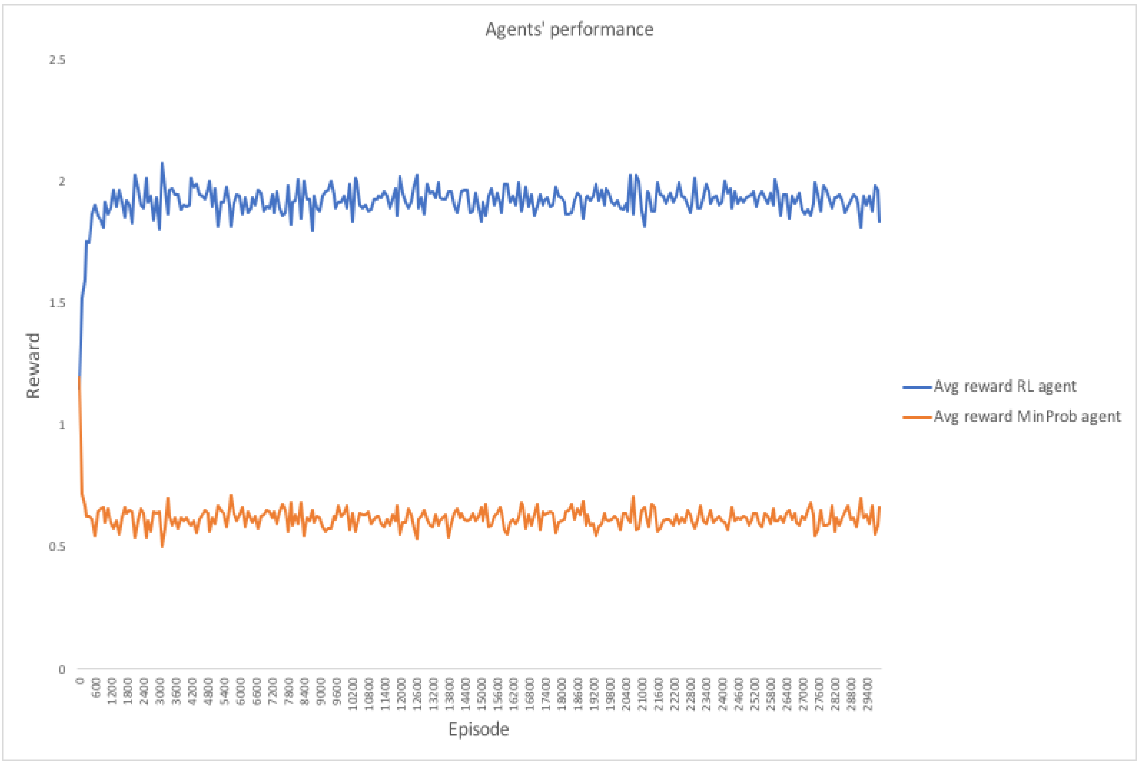


Figure 3.12: RL agent against Min-Probability agent

because the RL agent can distinguish between different states, unlike the classical one.

The Augmented Dickey-Fuller (ADF) test was used to assess whether the findings of the performance of the RL agent with other baseline agents were statistically significant and not due to chance. The ADF test is appropriate since it assesses where the time series is a stationary. The time series is considered stationary if it oscillates around a constant mean. Whereas the time series is considered non-stationary if it has a downward or upward trend. However, for doing further statistical analysis it is necessary to ensure that the time series is stationary.

After doing the stationary test and if the data is found to be stationary, a non-parametric test will be used to test the significance of the findings. In our case, the Mann-Whitney U test is used, which is a non-parametric test and it assumes that the data is not normally distributed. Hence, The Mann-Whitney U test is appropriate if the normality assumption is not valid.

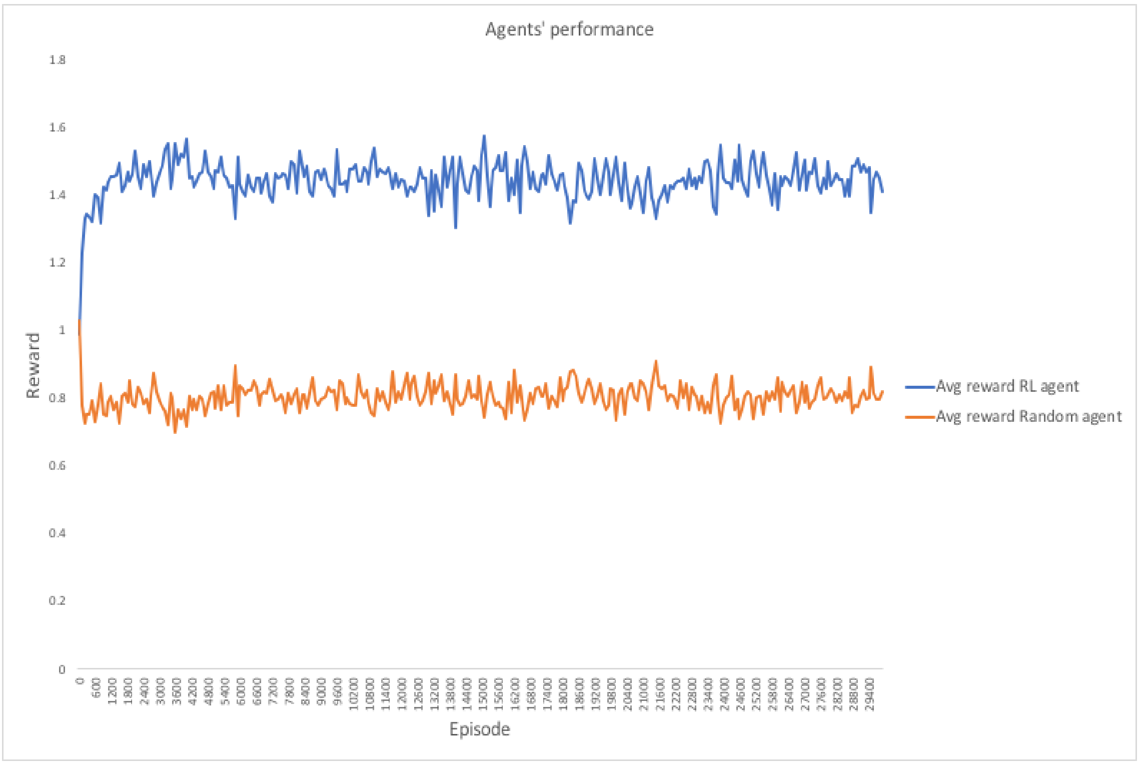


Figure 3.13: RL agent against Random agent

In the following subsections will perform the stationary and non-parametric tests to assess the significance of findings for the RL agent against baseline agents.

3 .1 MaxProb Agent Results

The results from the game between the MaxProb and RL agents from episode 1,000 to 29,900 are presented in Figure 3.11. These results were tested for statistical significance. In Figure 3.11 it can be observed that the rewards of the RL agent increase after a few hundred episodes and then stabilise. Whereas, the rewards of the MaxProb agent decrease after a few hundred episodes then stabilise. The ADF unit root test was used to test whether the reward obtained after episode 1000 was stationary and constant, with the null hypothesis that there exists a unit root against the alternative hypothesis of stationarity which can be described as:

H_0**:** That there exists a unit root (non-stationarity).

H_1**:** There is stationarity.

An ADF test on the RL agent's rewards showed that the null hypothesis of a unit root existing could be rejected ($p - value = 0$ (less than the significance level 0.05)),

Table 3.1: Summary statistics for ADF test

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	1.066	0.087	-51.920	0	-3.433	-2.863	-2.567
MaxProb agent	0.736	0.061	-51.785	0	-3.433	-2.863	-2.567

ADF statistics (-51.920) < critical value (-2.863) so the H_0 is rejected, which means the time series is stationary (see Table 3.1). Therefore, the rewards of the RL agent were stationary.

An ADF test on the MaxProb agent rewards showed the null hypothesis of the existence of a unit root could be rejected ($p-value = 0 <$ (less than the significance level 0.05)), ADF statistics (-51.785) < critical value (-2.863). So the H_0 is rejected, which means the time series is stationary (Table 3.1). Therefore, the rewards of MaxProb agent were also stationary.

Table 3.2: Rewards of RL agent and MaxProb agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	1.09	1.08	1.15	1.03	0.96	1.15	1.17	0.93	1.11	1.01
MaxProb agent	0.75	0.64	0.67	0.77	0.76	0.71	0.69	0.84	0.8	0.73

Table 3.3: Basic statistics RL agent and MaxProb agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	1.068	0.083	1.085
MaxProb agent	0.736	0.061	0.74

In Table 3.3 the higher mean and median suggest the RL agent had higher rewards than the MaxProb agent. A Mann-Whitney U was used to test the null hypothesis that the two groups are homogeneous against the alternative hypothesis that the RL agent had higher rewards than MaxProb agent.

H_0**:** The two groups are homogeneous.

H_1**:** The RL agent had higher rewards than MaxProb agent.

The Mann-Whitney U (two-tailed) test showed that the null hypothesis suggesting the two groups are homogeneous could be rejected with ($p - value = 0$ (less than the significance level 0.05)). Therefore, the RL agent had higher rewards than the MaxProb agent.

3 .2 MinProb Agent Results

The results from the game between the MinProb and RL agents from episode 1,000 to 29,900 are presented in Figure 3.12. These results were tested for statistical significance. In Figure 3.12 it can be observed that the rewards of the RL agent increase after a few hundred episodes then stabilise. Whereas, the rewards of the MinProb agent decrease after a few hundred episodes and then stabilise. The ADF unit root test was used to test whether the reward obtained after episode 1,000 was stationary and constant, with the null hypothesis that there exists a unit root against the alternative hypothesis of stationary.

H_0**:** That there exists a unit root (non-stationarity).

H_1**:** There is stationarity.

Table 3.4: Summary statistics for ADF test

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	1.824	0.129	-53.757	0	-3.433	-2.863	-2.567
MinProb agent	0.663	0.104	-54.179	0	-3.433	-2.863	-2.567

An ADF test on the RL agent rewards showed that the null hypothesis of a unit root existing could be rejected with ($p - value = 0 < p$ significant at 0.05), ADF statistics (-53.757) < critical value (-2.863). The H_0 is rejected, which means the time series is stationary (Table 3.4). Therefore, the RL agent rewards were stationary.

An ADF test on the rewards of the MinProb agent show that the null hypothesis (the existence of a unit root) could be rejected with ($p - value = 0$ (less than the significance level 0.05)), ADF statistics (-54.179) < critical value (-2.863). The H_0

is rejected, which means that the time series is stationary (Table 3.4). Therefore, the rewards of the MinProb agent were stationary.

Table 3.5: Rewards of RL agent and MinProb agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	1.94	2.05	1.7	1.99	1.77	1.78	1.91	1.78	1.68	1.76
MinProb agent	0.64	0.52	0.71	0.54	0.78	0.56	0.67	0.63	0.81	0.77

Table 3.6: Basic statistics RL agent and MinProb agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	1.836	0.127	1.78
MinProb agent	0.663	0.104	0.655

In Table 3.6 the higher mean and median of the RL agent suggest the RL agent had higher rewards. A Mann-Whitney U was used to test the null hypothesis the two groups are homogeneous against the alternative hypothesis that the RL agent had higher rewards than the MinProb agent.

H_0: The two groups are homogeneous.

H_1: The RL agent had higher rewards than the MinProb agent.

A Mann-Whitney U test showed the null hypothesis that the two groups are homogeneous could be rejected with ($p-value = 0$ (less than the significance level 0.05)). Therefore, the RL agent had higher rewards than the MinProb agent.

3 .3 Random Agent Results

The results from the game between the Random and RL agents from episode 1,000 to 29,900 are presented in Figure 3.13. These results were tested for statistical significance. In Figure 3.12 it can be observed that the rewards of the RL agent increase after a few hundred episodes then stabilise. Whereas, the rewards of the MinProb agent decrease after a few hundred episodes then stabilise. The ADF unit root test was used to test whether the reward obtained after episode 1,000 was stationary and constant, with the null hypothesis that there exists a unit root against the alternative hypothesis of stationary.

H_0: That there exists a unit root (non-stationarity).

H_1: There is stationarity.

Table 3.7: Summary statistics for ADF test

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	1.408	0.121	-53.322	0	-3.433	-2.863	-2.567
Random agent	0.812	0.086	-52.245	0	-3.433	-2.863	-2.567

An ADF test on the rewards of the RL agent showed the null hypothesis (existence of a unit root) could be rejected with ($p-value = 0$ (less than the significance level 0.05)), ADF statistics (-53.322) < critical value (-2.863). The H_0 is rejected, which means the time series is stationary (Table 3.7). Therefore, the rewards of the RL agent were stationary.

An ADF test on the rewards of the Random agent showed the null hypothesis of existence of a unit root could be rejected with ($p-value = 0$ (less than the significance level 0.05)). Therefore, the rewards of the Random agent were stationary, ADF statistics (-52.245) < critical value (-2.863). Therefore, the H_0 is rejected, which means the time series is stationary (Table 3.7). Therefore, the rewards of the Random agent were stationary.

Table 3.8: Rewards of RL agent and Random agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	1.45	1.47	1.4	1.4	1.36	1.28	1.71	1.33	1.32	1.36
Random agent	0.87	0.88	0.83	0.77	0.87	0.87	0.62	0.87	0.85	0.75

Table 3.9: Basic statistics RL agent and Random agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	1.408	0.121	1.38
Random agent	0.818	0.083	0.86

In Table 3.9 the higher mean and median suggest the RL agent had higher rewards. A Mann-Whitney U test was used to test the null hypothesis the two groups are homogeneous against the alternative hypothesis the RL agent had higher rewards than Random agent.

H_0**:** The two groups are homogeneous.

H_1**:** The RL agent had higher rewards than the Random agent.

A Mann-Whitney U test showed the null hypothesis the two groups are homogeneous could be rejected with ($p-value = 0$ (less than the significance level 0.05)). Therefore, the RL agent had higher rewards than the Random agent.

So far, the RL agent has learnt and performed in the same argumentation graph. When the RL agent faces a new argument graph, it has to learn from start. It would be ideal if the RL agent could transfer the experience that it has gained to a different argument graph, which would lead to generalising the policy. Transferring experience will speed up learning and improve performance [178]. This will be discussed in the next chapter.

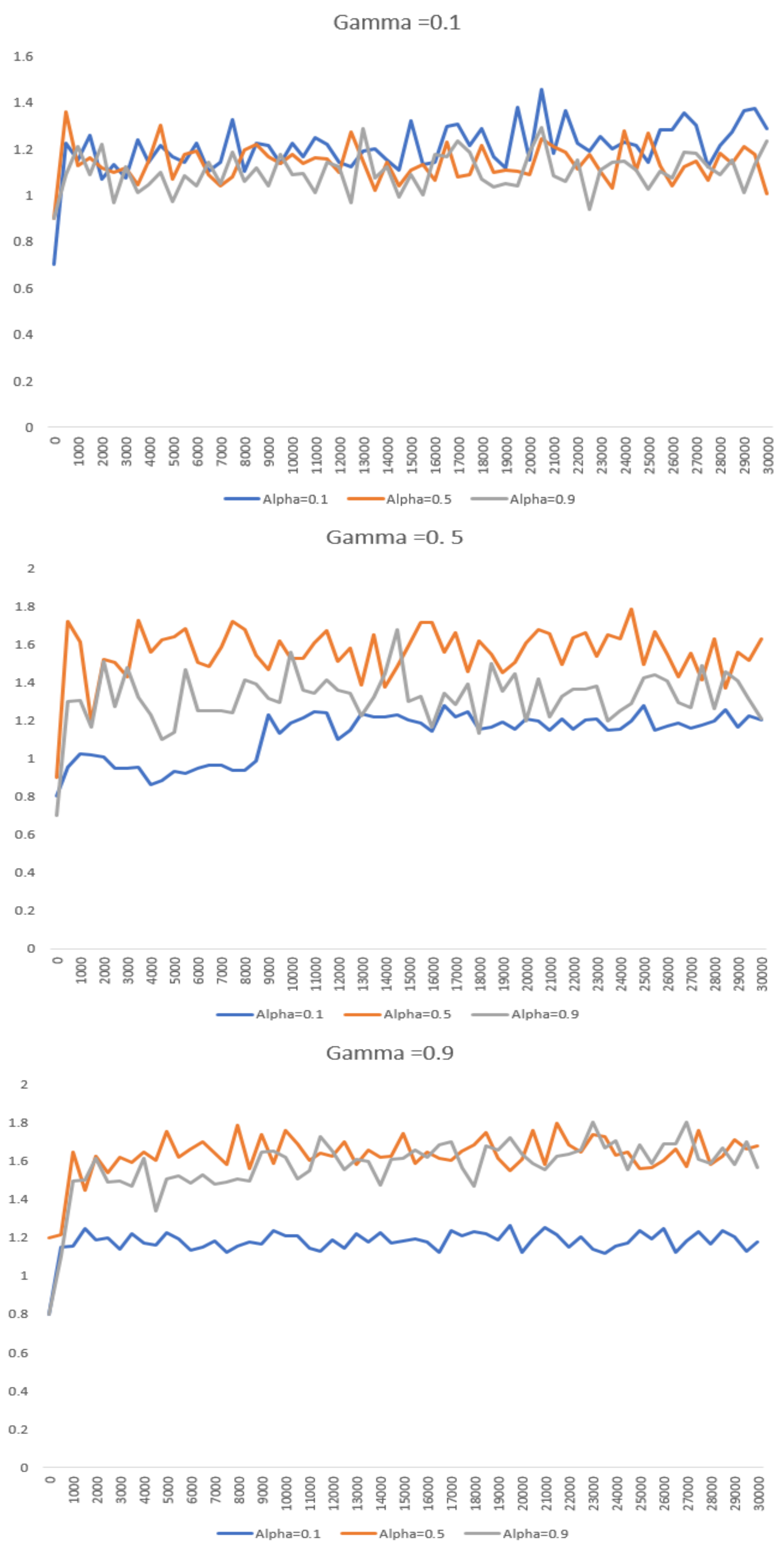


Figure 3.14: Learning rate and discounted factor correlation

4 Summary

In this chapter the design of *ARGUMENTO+* and how the RL agent was able to learn to argue using the Q-learning approach is discussed. Initially, classical state representation was adopted, based on state-of-the-art processes and the number of arguments in the grounded extension used as the long-term reward. A number of experiments were conducted by engaging the RL agent with different baseline agents. Overall, the results have been encouraging and indicate that the RL agent learns well.

The experiments also reveal some challenges in state representation in the argumentation domain. A new state representation was demonstrated, which showed promising results in making each state unique. In addition, it was found that the new state representation helps the RL agent to improve its performance.

Finally, this approach learns and performs in the same argumentation graph. Therefore, if the RL agent plays in a new argumentation graph then it has to learn from the start. Hoever, it would be worthwhile if the RL agent could transfer its experience from one domain to another. This could lead to an insight into generalisation approaches for generalising the policy of Q-value estimates between states and apply them to different argumentation graphs.

CHAPTER 4

POLICY GENERALISATION IN REINFORCEMENT LEARNING FOR ABSTRACT ARGUMENTATION

In the previous chapter, the the ability to learn to argue for an RL agent with two abstract argumentation systems was successfully demonstrated [1, 2]. However, in this case, the agent learnt to argue in a specific argumentation graph. As a step towards generalising the policy [3], we extend on this work to investigate whether the RL agent will be able to transfer knowledge[1] from one domain to another. By not having to learn from scratch each time, this will enable the RL agent to reuse the knowledge and policies learnt in one argumentation graph to apply them to another. Towards this, the current chapter (Chapter 4)) describes our work on policy generalisation and knowledge transfer.

1 Policy generalisation

According to Mitchell [36], a system is known to exhibit learning when its performance tends to improve with experience. Our results described in the previous chapter (Chapter 3) are consistent with this statement where the RL agent demonstrated an ability to learn from experience and showed improvement performance. While this is promising, Watson and Szathmáry [213] suggest that this quality of learning from past experience will have added benefits if it can be generalised and applied to scenarios that are different from where the learning occurred. Along these lines, we extend our current work of learning argumentation in RL agents to generalisation of policy approach.

Policy generalisation refers to generalisation of the state action map which has been previously learnt by the RL agent. This allows the RL agent to adapt the policy learnt in one domain to a different domain. In order to do that, the RL agent needs to identify the argument patterns such as state-action pairs which could be applied within different argumentation graphs and help the agent to transfer experience to different domains.

Since the RL agent is dealing with an abstract argumentation system that only contains arguments and binary relations for attacking, it is important to look into

[1]Transferring knowledge is used to improve a learner in a new domain by transferring information from a related domain [182] (discussed in more detail in Section 3 .1)

the attack relation between arguments to find any useful features that will help to generalise learning from one argumentation graph to another. These features could be useful for argument representation and enable the RL agent to know some of the patterns from a specific domain which could be adapted in other domains.

Along these lines, our work considered the following features: the number of attackers and the number of immediately winning attackers, to represent an argument action [3]. In order to work on these features, the state features and ways to represent the state needed to be identified.

2 State features

Since attack relations have been considered as a useful pattern that allows the RL agent to transfer experience between different argumentation graphs, it is suggested that the following state representation be applied for achieving generalisation:
(*levelOfTree, agentID, numberOfAttackers, numberOfUndefeatedAttackers*)
instead of
(*levelOfTree, agentID, currentArgument, previousArgument*)
In this case, the Q-table will not only deal with arguments, such as “a” or “b”, but also consider variables which are the number of attackers and the number of undefeated attackers in different nodes or arguments.

In the example shown in Figure 4.1, arguments C and D have zero attackers, argument B has one immediately winning attacker, and argument A has two attackers and one immediately winning attacker.

3 Generalisation method

The number of attackers provides the number of possibilities by which an argument can be attacked. The number of immediately winning attackers provides the number of immediately successful attackers. A further feature (currently named category) can be derived by using the formula: (number of immediately winning attackers)/(number of attackers). This number provides a short term view on the

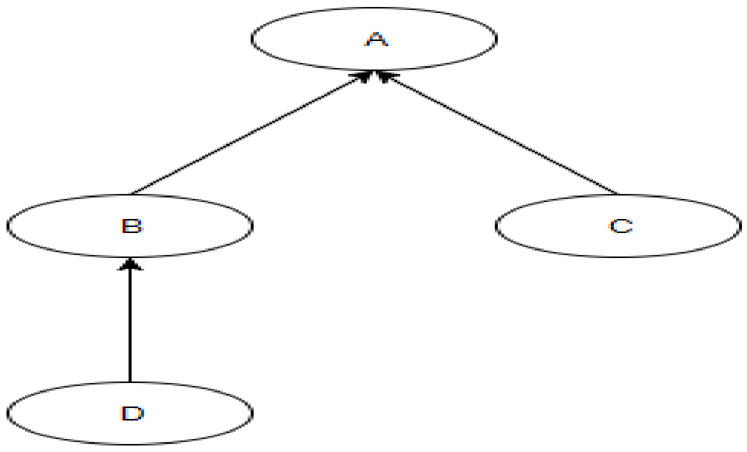


Figure 4.1: Argumentation graph

proportion of winning attackers. The value for category ranges from 0 to 1. It can therefore be further classified into different intervals: {0, (0,0.25], (0.25,0.5), 0.5, (0.5,0.75], (0.75,1), 1}. These intervals are easy for the RL agent to classify, which will lead to wins when transferring experience between different argumentation graphs. The smaller the number is, the more the agent is likely to win in the short-term. Therefore, from a short-term perspective, the categories might be classified as: definite win, highly likely win, likely win, possible win, unlikely win, highly unlikely win, definitely lose.

The number of attackers and the category were applied to represent argument actions and implemented in *ARGUMENTO+*. The following state representation was used to ensure that the state representation is less frequently repeated in a game: (depthOfTree, Argument, Category, NumOfAttackers). Two Q-tables were maintained, one for the current argument game and another general one for use other argumentation graphs. After finishing each argument game, the values in the general Q-table, which contained only (Category, NumOfAttackers), were transferred. If two arguments in the current game had the same category and number of attackers, it would take an average of these two values then transfer that to the general Q-table.

4 Experiment and evaluation

In order to evaluate whether the generalisation method works, a data set needs to be identified that can be used to test the agent's performance. Initially, three different graphs were tested (as a preliminary study) and it was found that the policy converged in episode 50 in all three. However, the performance curve was not able to stailise after convergence. This indicated that more datasets were needed for achieving stability. Therefore, 50 different graphs were randomly generated. The graphs were all connected graphs with the number of nodes ranging from 5 to 10. Leave One Out Cross Validation was chosen in 50 graphs and an average was taken at the end. The experiment was run over 50 games, with each game having 50 graphs. The agent was trained on 49 graphs and then tested with the 50th graph. We decided to encourage the RL agent to take two different approaches based on the number of arguments in the grounded extension.

The RL agent was examined to see whether its interest laid in winning the game with the minimum or maximum number of arguments. Based on the reward formula in the flowchart in Figure 4.2 to establish we aimed to establish whether the RL agent was interested in minimising or maximising the number of acceptable arguments in the grounded extensions. The experiment also had one agent with knowledge and one without. After 50 games, an average of the rewards after every 5 episodes was taken. The results are shown in Figures 4.3 and 4.4.

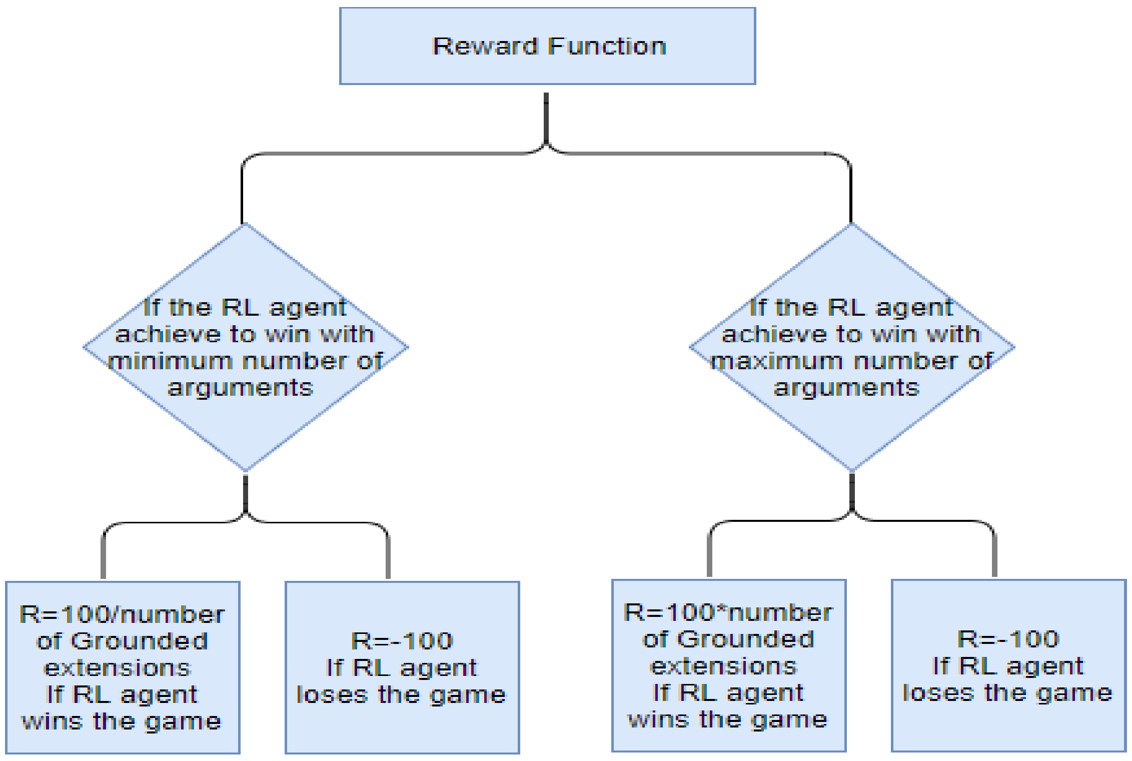

Reward Function
If the RL agent achieve to win with minimum number of arguments
If the RL agent achieve to win with maximum number of arguments
R=100/number of Grounded extensions If RL agent wins the game
R=-100 If RL agent loses the game
R=100*number of Grounded extensions If RL agent wins the game
R=-100 If RL agent loses the game

Figure 4.2: Reward shaping

The simulation configuration used here is as follows:

1. Randomly generate 50 unique argumentation graphs.

2. Perform leave one out cross validation (i.e. in the first iteration, out of the 50 graphs that are generated, graphs 1-49 are used for training the RL agent, and 50^{th} graph for testing; in the second iteration, graphs 1-48 and graph 50 are used for training, and 49^{th} graph for testing; and so on. These steps are repeated until 50 graphs have been used once as testing graphs.

3. Every 5th episode of each testing graph (exploiting not exploring and $\epsilon = 0$) is repeated 10 times.

4. An average of every fifth point of all the 50 testing graphs is taken for showing the results.

In both cases, the learning agent (the orange line in Figures 4.3 and 4.4) demonstrates an advantage for the first few cases. This can be attributed to the previously learnt knowledge. But most of the time that advantage is quickly overtaken by the agent that is learning from scratch (the blue line). This is an unexpected result,

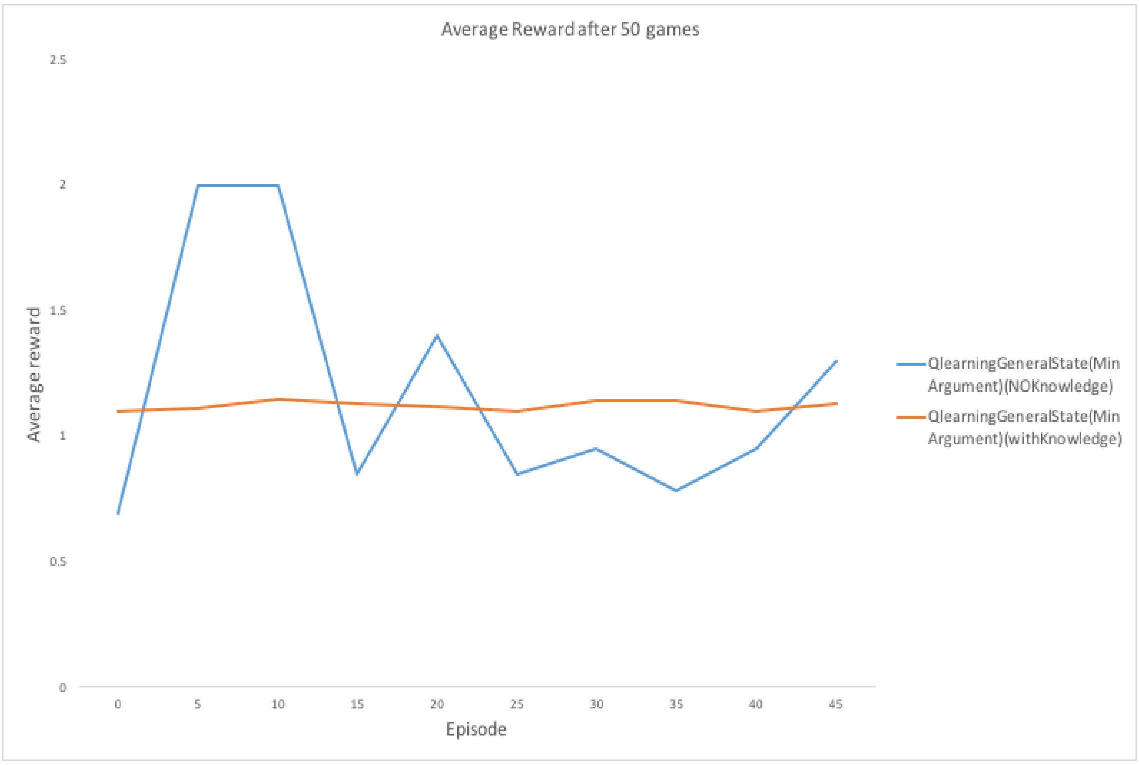


Figure 4.3: Cross validation for RL agent with and without knowledge with minimum numbers of arguments

because the RL agent already has experience from different argumentation graphs and therefore, it should have prior knowledge. However, the usefulness of the learned knowledge is only demonstrated at the start. By comparing both cases in Figures 4.3 and 4.4, the learning agent performs better when trying to win with minimum number of arguments. From the inspection of Q-tables, the only consistent finding that emerged is that, the arguments with no (zero) attackers attract the highest value in winning in the least number of argument scenarios.

Reflecting on the experimental results, the argumentation graph in Figure 4.5 is used as an example to facilitate the analysis. A uniform distribution is assumed, where the winning possibility of an argument is 50/50. For example, the current state of the learning agent is argument A and the agent needs to decide which argument to choose from “*B*”, “*C*” or “*X*”. In our proposal, the agent can see the next level of the tree arguments “*D*”, ”*E*” and “*F*”. Therefore, the chance of winning for arguments “*B*”, “*C*” and “*X*” is 0.25, 0.5 and 1 respectively.

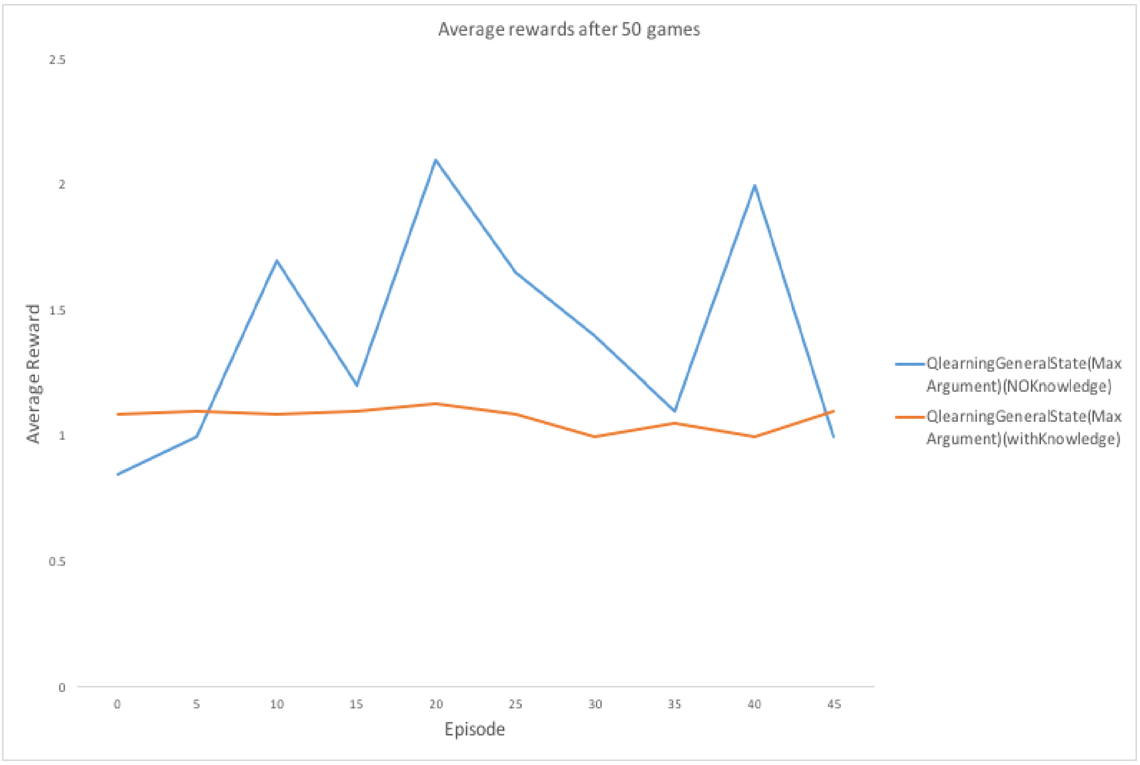


Figure 4.4: Cross validation for RL agent with and without knowledge with maximum numbers of arguments

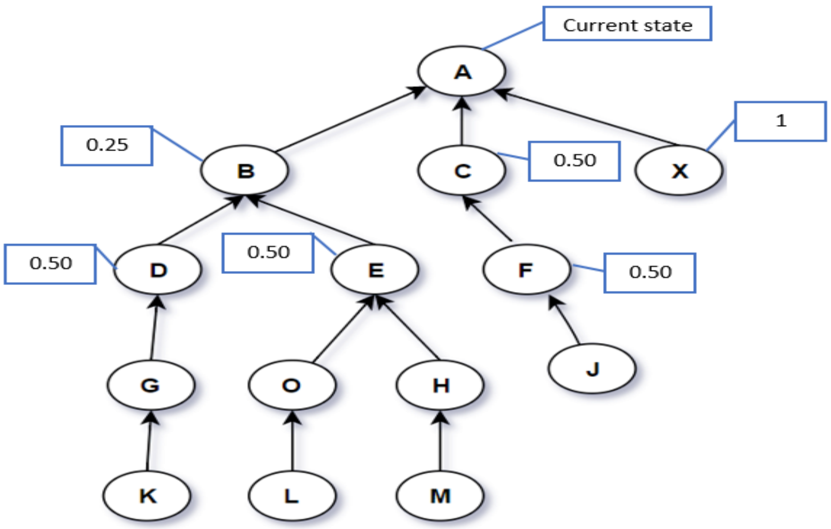


Figure 4.5: Argumentation graph with possibility of winning

Normally, the argument with the least number of attackers would be expected to perform better. This is the case for argument "*X*" (with value 1) in a 'win in the minimum number of arguments' scenario. However, when the learning agent is attempting to maximise the number of grounded extensions, "*C*" (with value 0.5) is the best choice. A further example can be seen from the argumentation graph in Figure

4.6. Although argument "*Q*" (with 0.5) has a higher possibility of winning than "*R*" (with 0.125), the learning agent will choose "*R*" (with a lower value) because it is a higher long-term reward. We believe that this is a potential reason that limits the learning agent from identifying useful patterns which would lead to generalisation of policy for different graphs.

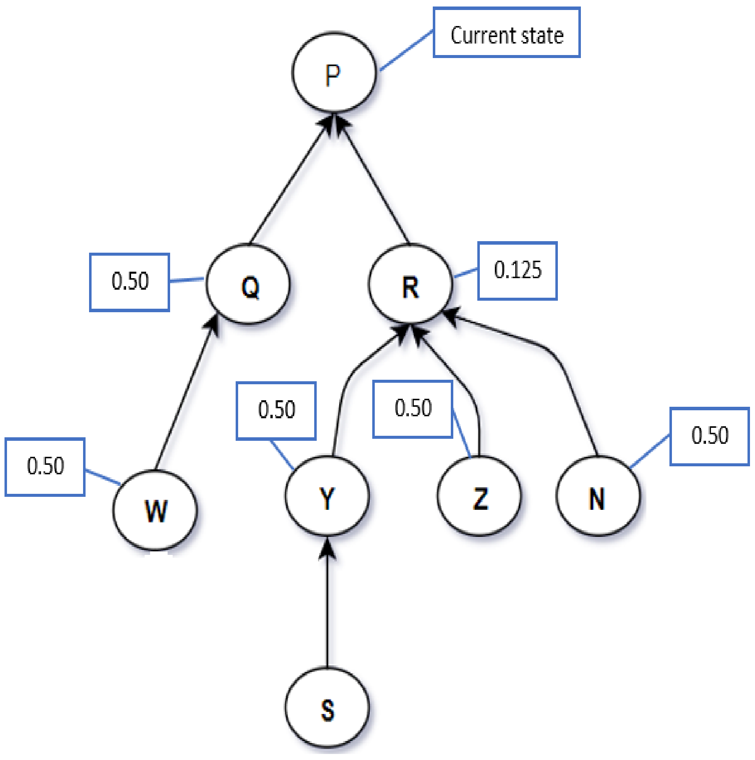


Figure 4.6: Different scenario of argumentation graph

5 Summary

Earlier in this work, an RL agent for abstract argumentation was designed and the agent performed well in a single argument graph. However, some issues in generalising the learning to different argument graphs were encountered. It was concluded that using current features in abstract argumentation face challenges in capturing useful argument patterns that could be reused in different argument graphs.

For identifying patterns, number of immediately winning attackers and number of attackers were used. Experiments were done by generating different argumentation graphs and leaving one out for cross validation and to test the knowledge for the RL agent. However, the results revealed difficulties for the RL agent to transfer knowledge within different argumentation graphs. Our analysis showed that, based on the pattern identified and the higher long term reward, the RL agent was confused about identifying the next action it needs to take.

Therefore, we propose to move from the abstract argumentation to proposition-based argumentation. In this, the internal structure of an argument is taken into account which could enable the agent to find patterns for generalising the policy. In general, for a persuasive dialogue, the dialogue goal can be specified as converting opponents' viewpoint. Dialogue history, commitment stores and the agent's knowledge base contribute to the formulation of the dialogue state. For example, the commitment store can tell the agent's position. These should be able to provide sufficient information for an agent to take decisions about an action.

Along these lines, this research was moved into dialogue-game-based argumentation, which has a richer argument representation that will potentially help the RL agent to recognise the patterns of the argument. This will be discussed in the next chapter.

CHAPTER 5

REINFORCEMENT LEARNING FOR LOGIC-BASED DIALOGUE GAMES

In the previous chapters (Chapters 3 and 4), the implementation of reinforcement learning in the abstract argumentation framework was described. From the implementation it was observed that , the patterns in the arguments could not be identified by the agent as they dealt with the abstract level of arguments. A method to allow the RL agent to adapt to a new argument graph was attempted. However, as discussed in Chapter 4, the results of this were not very encouraging. It was therefore thought ideal to progress this investigation to dialogue-game-based argumentation that is based on propositional logic. This approach looks at the internal structure of arguments, such as premises and conclusions. These could possibly help the RL agent to find patterns of argument that could not only facilitate the agent to learn how to argue, but also generalise the policy to different domains. The details on the implementation of propositional logic based dialogue game argumentation and its implications on the learning for an RL agent are presented in this chapter (Chapter 5)

1 Introduction

During generalisation of the RL agent policy for the abstract argumentation approach in different graphs, it was observed that the RL agent was not able to transfer experience to different graphs. As can be seen from the results presented in the previous chapter (Chapter 4), the main issue was that the arguments in the abstract level could not identify patterns to enable the learning agent to transfer learning from one graph to another. This suggests that, state action pairs are difficult to learn without reference to the internal argument structure. Since, propositional logic takes into consideration the internal structure of arguments, it is therefore logical to move from the abstract argument game to a propositional-logic based dialogue game for improving learning transfer in an agent [3, 4].

Since an abstract argumentation system deals with pure abstract arguments [11, 25, 28, 85] that are made up of only nodes and attacks, this research has been motivated to move into a logic based dialogue game. Logic based dialogue games take into consideration the internal structure of the arguments such as, argumentation schemes and evidence support sources that could help the RL agent to identify useful patterns that could be mapped between different context domains and facilitate learning transfer[1]. Along these lines, a logic based dialogue game (as discussed in detail in Section 2 .3) is another approach in the argumentation system which would be worthwhile to explore as it allows an RL agent to learn to argue. In this approach, two or more agents have an aim to resolve a conflict of opinion by verbal means [26]. A logic-based dialogue game will allow to study different aspects of this exchange, such as the communication language, protocol and agent behaviour. An essential aspect of protocol for resolving conflicts is to ensure that the communication is fair and effective [26]. Agent behaviour, on the other hand, is concerned with how agents can make moves by strategy. Strategy is part of a logic based dialogue game (as discussed in Section 2 .3) which consists of a dialogue model. The dialogue model allows agents to have a dialogue between themselves for resolving conflicts. A review of literature reveals a number of dialogue games that have been developed in the past [7, 44, 45, 145]. The DE model is one among them that has been adopted in this research. The DE model is discussed in detail in Section 2 .3.1.1. Adopting DE model in this research is motivated by a number of advantages it has to offer such as:

- The DE model deals with fallacious arguments and common errors to avoid [7, 145] when compared with other models such as the DC [43]. According to Yuan [7] this issue concerns the philosophical soundness of the dialogue model (i.e. how best it can avoid fallacious argument) [206]. Therefore, the DE model teaches students to improve critical thinking and debating skills by playing the dialogue game with an intelligent agent that prevents any fallacious arguments for improving debating skills.

- The DE model allows sufficient room for strategy formation [7]. This is considered desirable in a good dialogue model [122]. It is suggested that, a dialogue model provides more freedom for the participants to prepare their debating strategies [7] (such as, distance strategy [206], build and demolish strategy

[1]DE model is dealing with propositional logic, which should lead to the identification of some patterns, which will help and support the RL agent in adapting to different domains for instance in the DE model if the agent assert 'P' opponent may make challenge move 'Why P?' then agent may assert 'R, R→ P' and other heuristic strategy that built in DE dialogue model. This is left for future work.

[43]) to avoid unsuccessful debate. Therefore, an agent's strategy is significant for the agent to make a high quality contribution to the dialogue [4, 98].

- A hard-coded agent adopting the DE model was built with hard-coded heuristic strategies and the model performed to an advantage over others due to its computational tractability and simple dialogue rules [4, 7, 44–46].

- The DE model is built using propositional logic [7]. In comparison to the abstract argument game (described in Chapters 3 and 4), the DE model is richer and has an innovative design in which the dialogue state is represented by means of commitment stores and different types of movement such as questions, statements and challenges [4]. Whereas, in the abstract argument game, state representation is strictly restricted to nodes (arguments) and arcs (attack relations). It was expected that the DE dialogue game would be a useful learning experience for the RL agent.

Our initial implementation using abstract argumentation presented challenges for generalisation. In this, the RL agent encountered difficulty in learning patterns in an abstract argumentation system for transferring experience into different graphs. These limitations could potentially be overcome with the logic-based dialogue game approach as this provides patterns for the RL agent to learn to argue and share experiences in different environments. The present work will adopt the DE model. The knowledge base for this has already been previously built [7] with the architecture shown in Figure 2.11. To this, some features were added and identified as patterns for supporting the RL agent to learn to transfer the knowledge into different contexts. The patterns of arguments are identified here as argument schemes [2] including sources of support for the claim that could be learned by the agent. In the knowledge base, the environment contains the main claim with the rebuttal and each claim is supported by arguments which will be discussed in the coming sections.

[2]it is a way of argument reflecting structure of common types of arguments used in daily discourse, as well as in special contexts such as legal arguments and scientific arguments [63]. This will be refereed in section 2 .3.

2 Agent and environment design

For engaging the RL agent in an argument with different baseline agents, all players and agents that play in the DE game need to be defined. This section presents details of all the agents that take part in the game and discusses their strategies for making moves. The environment and domain in which all participants play is also discussed.

2 .1 RL agent

As discussed in Chapter 2 Section 3 , the RL agent interacts with an environment by observing its state, taking an action, and receiving a reward. The agent uses a Q-learning algorithm that is given in Equation (2.2). To allow the RL agent to learn to argue with opponents, the state, action, and rewards need to be identified. In the RL agent, this occurs by observing the environment and deciding the kind of action that needs to be taken. In the DE dialogue model, the commitment store is a significant state variable which can record what an agent has asserted or implicitly accepted during the dialogue. Therefore, the commitment store updates according to the commitment rules while the agent takes actions.

In addition to the commitment store, dialogue history could also be considered as a state variable. It contains the dialogue situation of an agent, such as it's previous move. Since each agent has its own commitment store, the previous move could be used as a state variable due to it's simplicity. Therefore, the dialogue state can be defined as: $(previousmove \cup CS1 \cup CS2)$, where CS1 is the commitment store for one dialogue participant, and CS2 belongs to the dialogue partner. The state representation will maximise the possibility of a state unreported in any given case.

An action is the decision that the RL agent is able to make from the available move types (such as, *assert*, *question*, *challenge*, *withdraw* or *resolution demand*) in the DE model. Defining these move types allows the agent to choose to reply to the previous move. Additionally, move content, which is a proposition or conjunct of propositions, is stored together with the move type. The RL agent aims to map it's state with an action called *policy* (π).

If the RL agent wins the game, it receives a reward and it is punished when it loses. Likewise, the minimum number of moves will also be considered as the RL agent has a target of winning the game with the minimum number of moves. This strategy could be relevant in some persuasive dialogues because, sometimes an agent is likely to reveal more of it's KB than is necessary and such superfluity could be unacceptable in some contexts [202] (discussed in detail in Chapter 2). The positive reward function for the RL agent is given in the following equation:

$$R = \begin{cases} 100 + \frac{W}{L} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{5.1}$$

Where W is the number of moves in the first winning episode, which becomes the benchmark, and L is the number of moves in the current episode. Hence, when L is at a minimum, the reward will be increased. To apply reinforcement learning, the Q-learning algorithm is used, as discussed in Section 3 of Chapter 2, shown in the following algorithm:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_t,a_{t+1}) - Q(s_t,a_t)]$$

while r_{t+1} is the immediate reward, it is set to -0.01 as a little punishment to motivate the RL agent to choose the minimum number of moves to win the game quickly.

2 .2 Baseline agent

A number of dialogue games that are built with computational agents are reported in the literature. An agent typically adopts a strategy for making moves in a dialogue game . For instance, an agent can adopt information hiding strategies which are discussed in a number of different papers concerning MAS [214, 215]. An agent can be utility based in not only the possible payoff outcomes but also in the information properties of the strategy that the agent uses [214]. For example, Otterloo designs an agent to make the optimal response based on computing the Nash equlibria. Paruchiri et al. [215] develop an agent based on maximising policy randomisation to thwart the opponent's prediction of agent-based behaviour and minimise the opponent's ability to inflict harm. They focused on MAS security where deliberate threats are caused by unseen opponents whose actions and capabilities are unknown, but the opponents may exploit any predictability in the policies of their agent. Amgoud and Maudet [107] presented an idea for an agent which can adopt "meta-preferences" such that, the agent focuses to restrict access to defeaters as a way of pushing the choice of arguments in the context of dialogue by choosing the smallest argument. Rahwan et al. [216] allow agents to interact based on negotiation strategy and identify factors which help an agent to select strategy for negotiating with other autonomous agent.

Wardeh et al. [217] in the PADUA protocol, allow every agent (proponent and opponent which are defined as a dialogical agent) to have its own collection of examples that it can mine to find reasons for and against a classification of a new instance based on association rules. Amgoud and Parsons [218] used standard-based arguments with if then rules, and outlined five classes of agent profiles as follows: (i) agreeable agent (an agent accepts an argument whenever possible) (ii) disagreeable agent (agents which approve only when there is no excuse not to do so) (iii) open-minded agent (agent which can only challenge if necessary) (iv) argumentative agent (agents which make challenge whenever possible) (v) elephant child agent (asks questions whenever possible). Wardeh et al. [217] considered agreeable and disagreeable agents since the agents' behaviour in their context was most appropriate with their approach (PADUA protocol).

The DC system [42, 43] consists of two agents which are used to conduct a debate with each other using DC dialogue game [43]. The DC system allows these agents to debate on a disputed topic, which is *capital punishment*. One agent adopts agree with capital punishment, and another adopts against and visa versa. These agents start debating with these positions until one agent persuades the other with its point of view. However, issues such as, fallacies arguments, and prevented questions begging, were encountered by Yuan [7] and Yuan *et al.* [145] with the proposed DC system under investigation. The DE system was developed to address these issues by means of modified commitment and dialogue rules. Based on these modifications, Yuan developed an agent with heuristic strategies which is ready to play a dialogue game with humans (DE dialogue game). These have been previously discussed in detail in Section 2 .3.1.

Since the DE system is known to improve common errors in the DC system, similar agents have been implemented in DC to further study their behaviour [145]. These agents have been made to debate with each other, wherein, one agent attempts to persuade the opponent to accept it's point of view. These two agents have been built with different strategies one with heuristics and another with randomised strategy.

As discussed in [7], strategy is important for facilitating the DE dialogue with the RL agent. Strategy enables the agents to play a game with high quality dialogue contribution [98, 206, 207, 218]. In the DE model [7, 44–46] there are five different dialogue scenarios which an agent could face. This is defined by the previous move type made by the opponent such as, a challenge, a question, a resolution demand, a statement or a withdrawal [98]. Thus, each baseline agent has to be taken into account in relation to the strategic decisions that are taken.

2 .2.1 DE heuristic strategy

The DE heuristic strategy is an agent built with a hard-coded strategy in the DE model [7]. It depends on the set of heuristics, which was based on three levels of decisions taken from [43, 98]:

1. Retain or change the current focus [3].

2. Build own view or demolish the user's view.

3. Select method to fulfil the objective set at levels (1) and (2) [98].

Yuan *et al.* [98, p. 219] explain the levels as:

[3]Current focus means carry on within the previous move, Yuan [7] clarifies this "The level (1) decision concerns whether to retain the current focus or to change it. The decision, that is, involves whether to continue the attempt to substantiate or undermine a particular proposition."

"*Levels (1) and (2) refer to strategies which apply only when the computer is facing a statement or withdrawal, since in all other cases the computer must respond to the incoming move. Level (3) refers to tactics used to reach the aims fixed at level (1) and (2).*" [4]

Levels 1 and 2 apply only when the agent is facing a statement or withdrawal as a result of it's response to incoming moves in all other cases. Whereas level 3 refers to the tactics which the agent uses to reach the aims fixed at both levels 1 and 2, and are then applied in every game scenario [98].

In the levels taken from Yuan *et al.* [98], level 1 is when an agent decides whether to retain the current focus or change it. In other words, the decision involves whether to continue the attempt to substantiate or undermine a specific proposition. Level 2 is when an agent decides to adopt a build or demolish strategy. A build strategy includes the acceptance of propositions that support the thesis of the proponent, while a demolishing strategy seeks to remove support from the opponent for their thesis. Level 3 can be applied to each of the dialogue situation within different heuristics, which are:

1. A question raised by the opponent.
2. A challenge made by the opponent.
3. A resolution demand made by the opponent.
4. A “no commitment" made by the opponent.
5. A statement made by the opponent.

This is a set of different strategy heuristics which are adopted by the DE agent to play against an opponent. More details of these strategies can be found in Yuan *et al.* [44, 45, 98] (this can also be found in Appendix A taken from [98]).

2 .2.2 DE randomised strategy

A Random agent will choose a move randomly from the set of legally available moves in the DE game rules. This agent is designed to use random arguments based on [7].

[4]Details of the level 3 strategies can be found from [44, 45, 98].

2 .3 Environment

A knowledge base to allow agents to argue with each other is needed to engage agents to play the game. In [7], different ways of representing the knowledge in the literature such as in [43, 99, 132, 133] are reported. It is suggested that the knowledge base should have statements to enable the answering of questions or supporting the argument [7, 43]. Additionally, it should provide statements to rebut other statements. For these reasons, Yuan [7] adopt a modified version of Toulmin's schema [133][5] for developing the DE model and making the knowledge base structure as shown in Figure 2.11.

Some features have been added to the knowledge base (such as, argument schemes and an evidence source), with the expectation that the argument patterns can be recognised by an RL agent through features as shown in Figure 5.1.

According to Walton [63] "*argumentation schemes are forms of argument (structure of inference) that represent structures of common types of arguments used in everyday discourse, as well as in special contexts like those of legal argumentation and scientific argumentation*". Rahwan [20] suggests that these schemes capture stereotypical (deductive or non-deductive) reasoning patterns present in a regular discourse. Walton in [219] has 25 Argumentation schemes which are identified for common forms of presumptive reasoning. Rahwan [20] states that argument schemes offer several useful features for communication with MAS such as in legal reasoning. Atkinson *et al.*[108] use argument schemes for proposing actions to structure their dialogue game protocol. Also, Karunatillake *et al.*[220] use schemes to make a negotiation strategy in the presence of social influence. Hence, it would be interesting to study if the RL agent is able to use argument schemes and evidence support sources to recognise it as patterns.

An argument scheme (such as arguing from consequence [221]) is represented in a parallelogram. The circles in Figure 5.1 represent evidence of the argument (such as newspapers, magazines or scientific papers) as a source of the argument. For

[5]"The modal qualifier and the backing are omitted from Toulmin's original schema"[7]

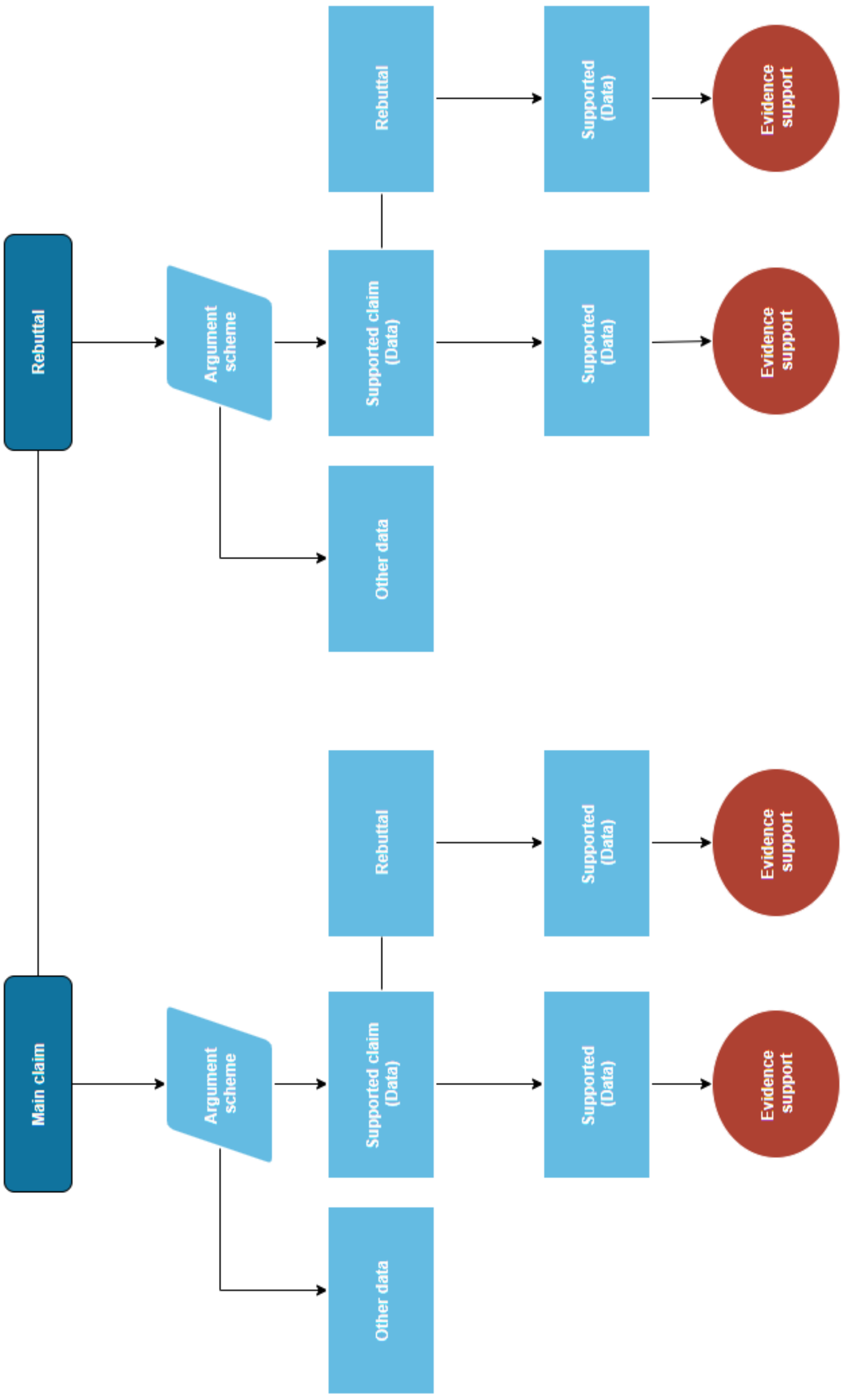


Figure 5.1: System knowledge base [7], [8]

instance, the RL agent could learn the reliability of supporting evidence from the environment [4] as seen in Figure 5.2.

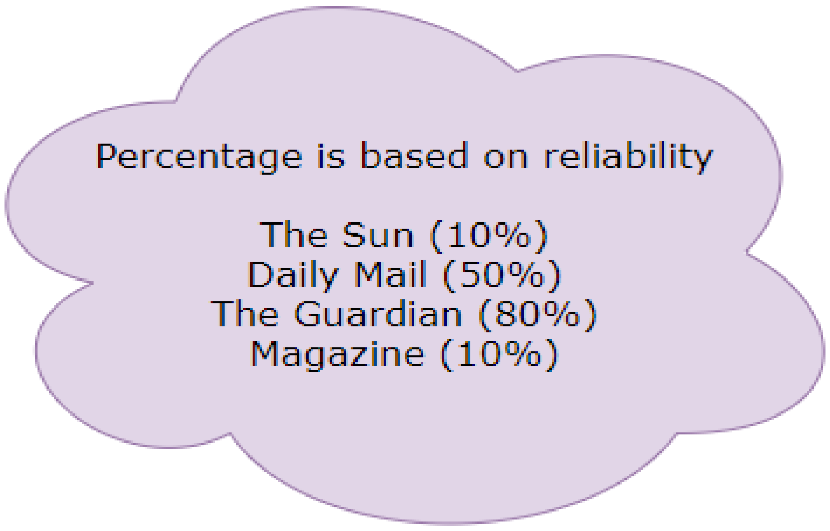


Figure 5.2: Sources environment

The structure of *capital punishment* (CP) knowledge base [7], which was adopted after adding schemes and sources with respect to the practical level is illustrated in Figure 5.3. There are two main claims which are "CP is acceptable" and "CP is not acceptable". The "CP is acceptable" has five main arguments and further five supporting arguments. Whereas, another claim "CP is not acceptable" has five main arguments and six supporting arguments.

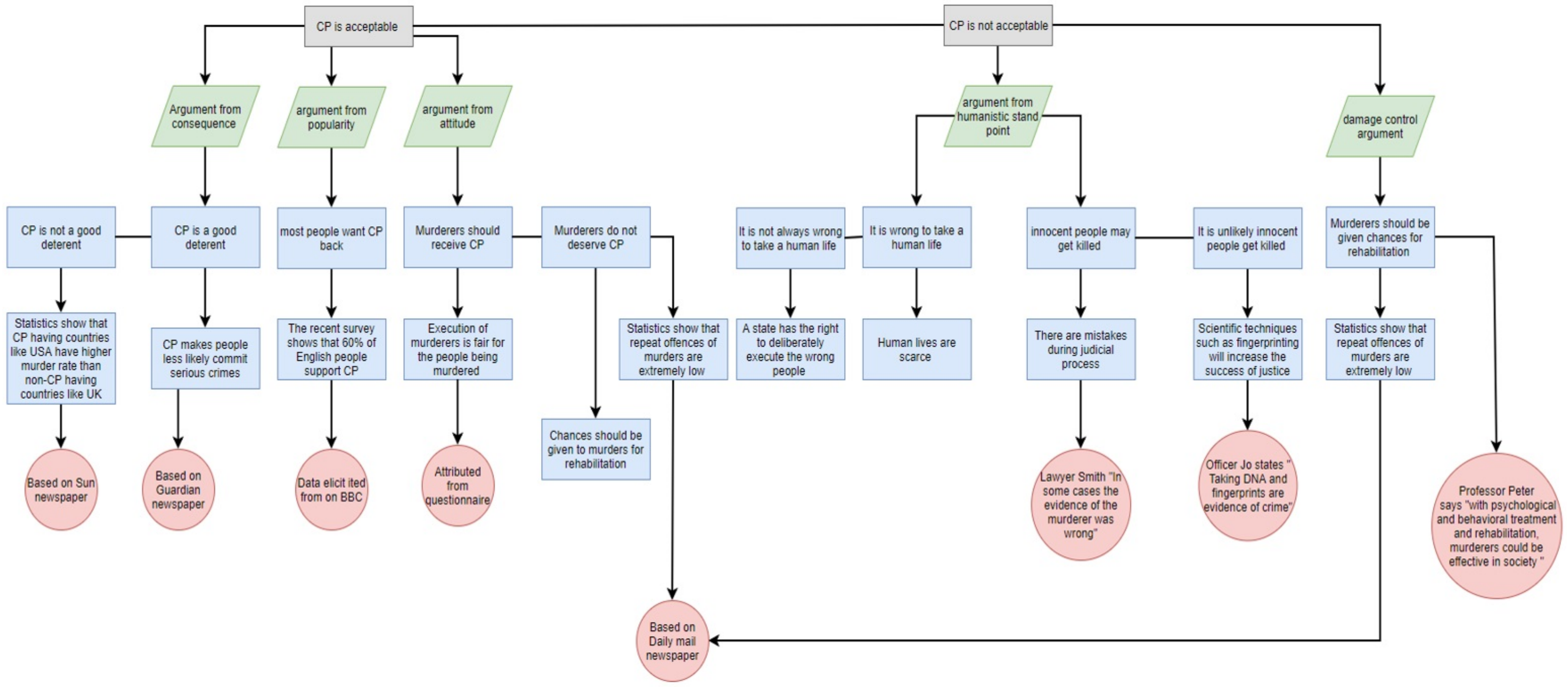

CP is acceptable
CP is not acceptable
Argument from consequence
argument from popularity
argument from attitude
argument from humanistic stand point
damage control argument
CP is not a good deterent
CP is a good deterent
most people want CP back
Murderers should receive CP
Murderers do not deserve CP
It is not always wrong to take a human life
It is wrong to take a human life
innocent people may get killed
It is unlikely innocent people get killed
Murderers should be given chances for rehabilitation
Statistics show that CP having countries like USA have higher murder rate than non-CP having countries like UK
CP makes people less likely commit serious crimes
The recent survey shows that 60% of English people support CP
Execution of murderers is fair for the people being murdered
Statistics show that repeat offences of murders are extremely low
A state has the right to deliberately execute the wrong people
Human lives are scarce
There are mistakes during judicial process
Scientific techniques such as fingerprinting will increase the success of justice
Statistics show that repeat offences of murders are extremely low
Chances should be given to murders for rehabilitation
Based on Sun newspaper
Based on Guardian newspaper
Data elicit ited from on BBC
Attributed from questionnaire
Lawyer Smith "In some cases the evidence of the murderer was wrong"
Officer Jo states " Taking DNA and fingerprints are evidence of crime"
Professor Peter says "with psychological and behavioral treatment and rehabilitation, murderers could be effective in society "
Based on Daily mail newspaper

Figure 5.3: Capital punishment knowledge base

3 Experimental setup

To understand whether the performance of the RL agent is improving, it is needed to engage the RL agent in the DE game to play against baseline agents. For evaluation purposes, conducting a number of experiments is required in which the RL agent needs to play against the DE heuristic strategy agent and the DE randomised agent. The current *capital punishment* discussion topic is shown in Figure 5.3. In this argument game, one player needs to persuade the other party to accept their point of view to resolve conflicts between parties. So, the RL agent should adopt an opinion and then attempt to learn to argue with it's opponent. Baseline agents have already been constructed into the DE model such as in [7]. So all players have a set of moves which are *Assertion, Question, Challenges, Withdrawal* and *Resolution Demands*. Each agent has a commitment store to record what has been stated and accepted during the dialogue. All agent have to follow the DE dialogue rules R_{FROM}, R_{QUEST}, R_{CHALL}, $R_{RESOLVE}$, $R_{RESOLUTION}$ and $R_{LEGALCHAL}$.[6]

To engage the RL agent in the game, state, actions and reward have been justified previously (Equation 5.1). State is defined as: $(previousmove \cup CS1 \cup CS2)$ which is a combination of previous move and commitment store for both agents. The action will be the set of available moves in the DE model. The reward function that is defined in Equation 5.1 and the Q-learning algorithm in Equation 2.2 was implemented in this experiment. In Equation 2.2, $r_{t+1} = -0.01$ is configured to make a little punishment to encourage the RL agent to choose the minimum number of moves to win the game quickly and is associated with the reward function.

Hence, learning rate (α), discounted factor (γ) and epsilon (ϵ)[7] were set up as 0.9, 0.9 and 0.3 respectively which is expected to impact positively in making the RL agent to outperform and improve learning against baseline agents. The game has 4,000 episodes starting from 0 and the RL agent initially starts behaving randomly since it does not have any experience at this stage. Episode 0 is repeated 10 times to avoid a lucky choice, after which the subsequent episodes are executed. Exploration

[6]Discussed in more details in Section 2 .3.1.1

[7]ϵ-Greedy which introduced in Chapter 2

and exploitation need to be traded off to explore more actions space or exploit the knowledge at any given time [8]. The reason for this is to evaluate the experience that the RL agent has learned. Therefore, exploitation is updated every 100 episodes, where it is reset to 0 and is repeated 10 times. The performance was measured by calculating the reward for each agent, RL agent and it's opponent every 100 episodes (exploitation episodes) to observe the RL agent's ability to make decisions based on the values in the Q-table. The aim was to understand the RL agent's performance in terms of rewards gained in the long-term and winning the game with the minimum number of moves.

4 Experimental results

The results against both baseline agents look promising, as shown in Figures 5.4 and 5.5.

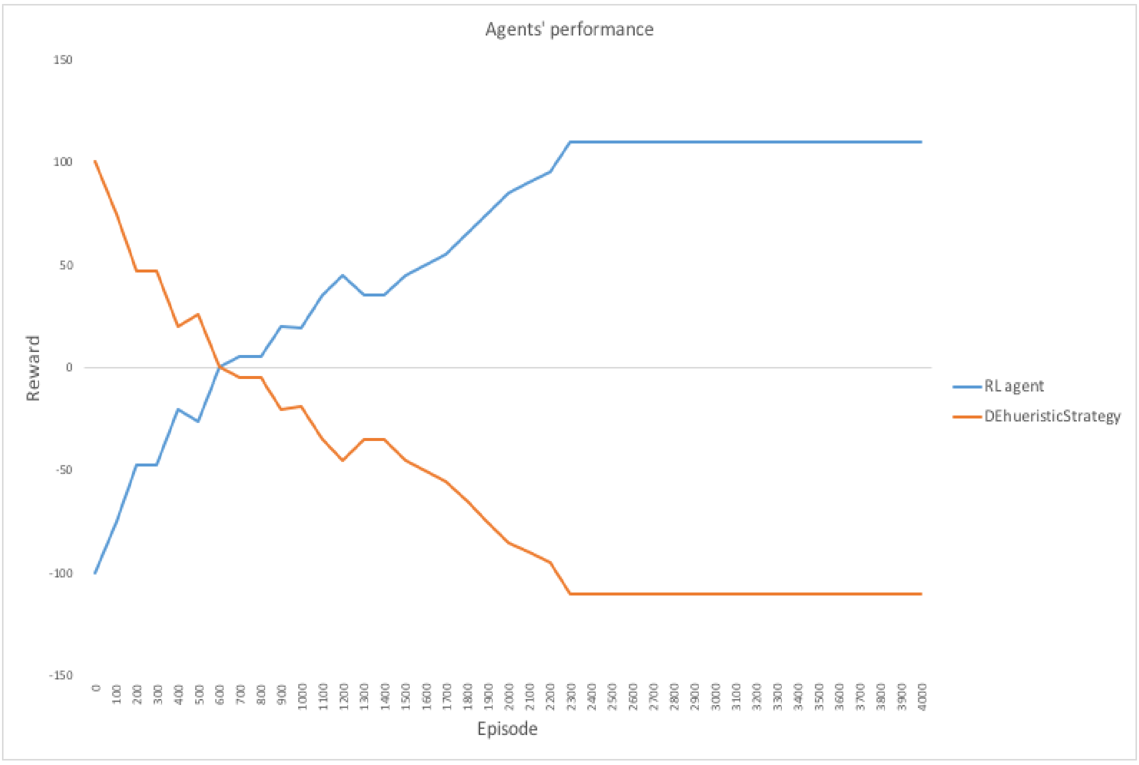


Figure 5.4: RL agent against fixed strategy agent

[8]ϵ-greedy policy has been used which takes action using the greedy strategy with a probability of 1-ϵ and a random action with a probability of ϵ. This strategy makes sure that all actions space are explored [222]

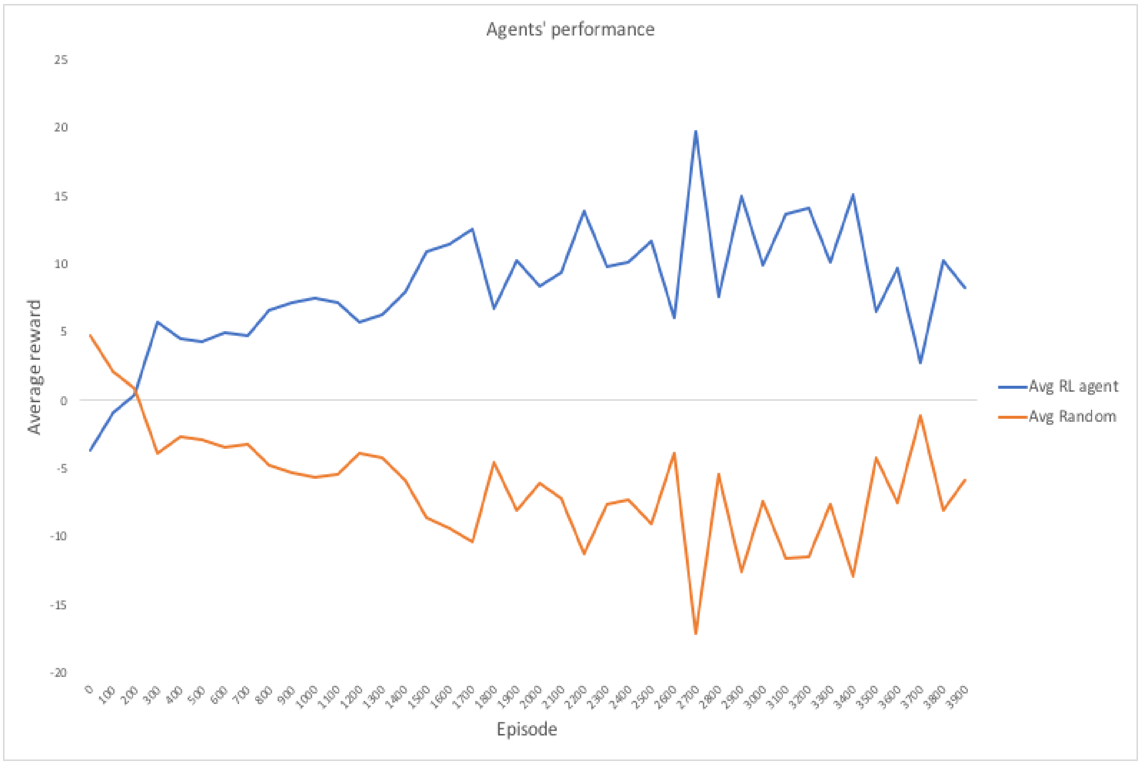


Figure 5.5: RL agent against Random agent

Both Figures 5.4 and 5.5 show that the RL agent curves have a trend to rise with increasing numbers of episodes. This suggests that, the RL agent is able to learn to argue against different DE baseline agents and after converging it wins most games.

The ADF test was used to assess the stationary aspect of a time series. If the data was found stationary a non-parametric test would be used for the chosen episode to assess the significance of finding (The Mann-Whitney U test). The reasons for choosing these tests are mentioned before in Section 3 of Chapter 3.

From Figure 5.5 it can be observed that the rewards of DE Random agent exhibited a downward trend while the rewards of RL agent exhibited an upward trend. The (ADF) unit root test was used to test whether the reward obtained after episode 1000 was stationary and constant. An ADF test was used to test the null hypothesis that, there exists a unit root, against the alternative hypothesis of stationary.

H_0: That there exists a unit root (non-stationarity).

H_1: There is stationarity.

Table 5.1: Summary statistics for ADF test

Agent	Mean	Standard deviation	ADF Statistic	P-value	Critical value 1%	Critical value 5%	Critical value 10%
RL agent	7.352	10.027	-11.082	0	-3.452	-2.871	-2.572
DE Random agent	-5.900	9.058	-11.303	0	-3.452	-2.871	-2.572

An ADF test on the rewards of the RL agent showed that the null hypothesis of existence of a unit root could be rejected with $p - value = 0$ which is lower than the significance level 0.05. ADF statistics $(-11.082) <$ critical value (-2.871) so we reject the H_0 which means the time series is stationary. Therefore the rewards of RL agent were stationary.

An ADF test on rewards of the DE Random agent showed that the null hypothesis of existence of a unit root could be rejected with($p - value = 0$ which is lower than the significance level 0.05). The result of ADF statistics (-11.303) was less than the critical value (-2.871), therefore, we reject the H_0. This means that the time series is stationary. Therefore it can be concluded that the rewards of DE Random agent were stationary.

Table 5.2: Rewards of RL agent and DE Random agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	15.85	12.50	14.97	4.39	3.65	3.47	26.76	-8.51	8.79	0.14
DE Random agent	-14	-10	-13.60	-4.20	2	-2.20	-21.20	9.80	-5.80	0.20

Table 5.3: Basic statistics RL agent and DE Random agent at the last episode

Agent	Mean	Standard deviation	Median
RL agent	8.201	9.829	6.590
DE Random agent	-5.900	9.058	-5

From Table 5.3, the higher mean and median suggest that the RL agent had higher rewards. A Mann-Whitney U test was used to test the null hypothesis and the two groups were found to be homogeneous against the alternative hypothesis as the RL agent had higher rewards than the DE Random agent.

H_0**:** The two groups are homogeneous.

H_1**:** The RL agent had higher rewards than DE Random agent.

A Mann-Whitney U test showed that the null hypothesis could be rejected as the two groups were homogeneous with $p-value = 0.010$ which is lower than the significance level 0.05. Therefore the RL agent had higher rewards than DE Random agent.

Comparing the results obtained by the RL agent (Figures 5.4 and 5.5), it could be stated that the RL agent's performance against the heuristic strategy agent was better than the DE Random agent. This is due to the fact that, the random moves picked up by the DE Random agent could impact on the RL agent's performance in selecting its best move to win the game. This was expected, since it played against an agent with no planned strategy. The RL agent can therefore improve its ability to learn to argue by winning the game against the DE Random agent.

Since it is fruitful for the RL agent to minimise the number of moves or arguments, the RL agent is encouraged to win with minimum number of moves. The reward function motivates the RL agent to win the game with minimum moves as the immediate reward r_{t+1} has been set to -0.01 to make the RL agent win as fast as possible. Hence, it works against both opponents, the DE heuristic strategy agent and the DE Random agent as shown in Figures 5.6 and 5.7 respectively.

The reason to make an RL agent minimise moves is interesting, particularly based on Oren *et al.* [201], who suggest that while an agent tries to win a dispute reveals as least knowledge as possible. They also mention that these kind of tactics can be seen in many real world scenarios such as in [223]. This supports that the agent revealing much information in the current dialogue could effect on the outcome of a dialogue by loosing a game [201].

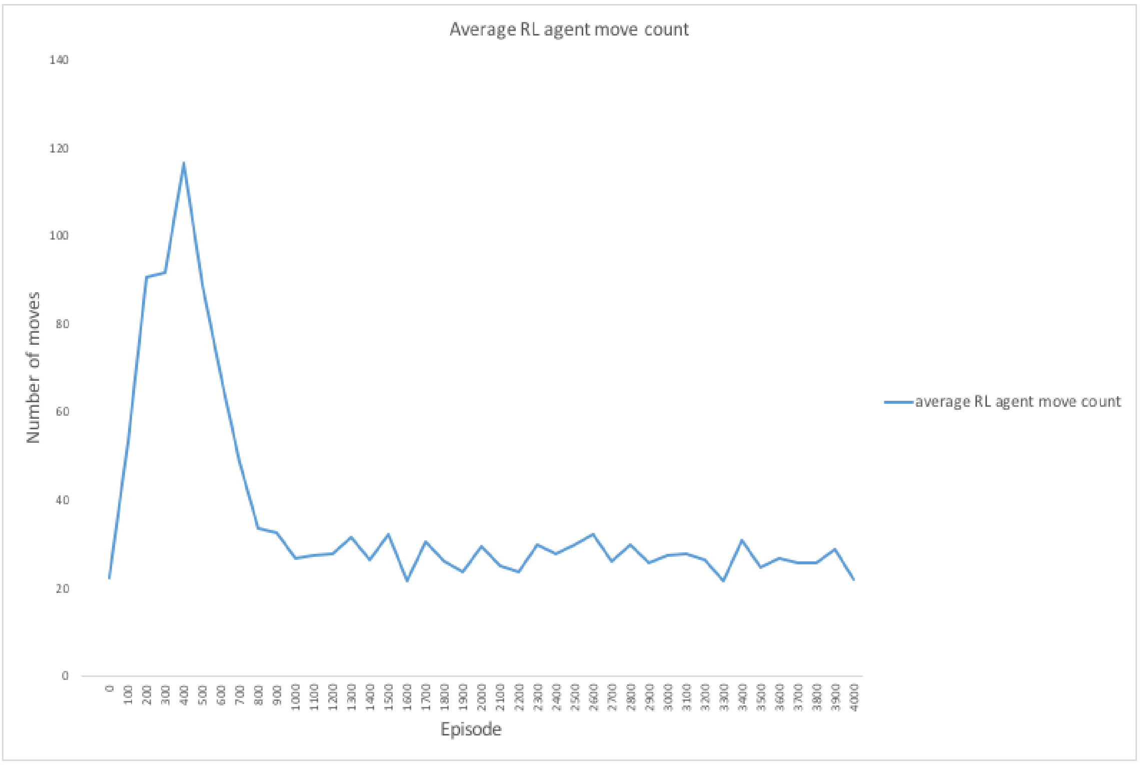


Figure 5.6: Average moves count against Fixed strategy agent

Both Figures 5.6 and 5.7 show that the number of moves converged to decrease after around 750 episodes. So, the RL agent was able to improve to learn to minimise the number of moves and win the games to achieve the task. The reward function has worked successfully to make the RL agent learn to argue with minimum number of argument utterances.

Further statistical tests are needed to to assess whether the data is stationary for both Figures 5.6 and 5.7. From Figure 5.6 it can be observed that the number of RL agent moves increased after a few hundred episodes and sharply declined after the next few episodes before stabilising. The ADF unit root test was used to test whether the mean number of moves after episode 1000 was stationary and constant. The reason for this has been mentioned in Section 3 of Chapter 3. An ADF test was used to test the null hypothesis that there exists a unit root against the alternative hypothesis of stationarity.

H_0**:** That there exists a unit root (non-stationarity).

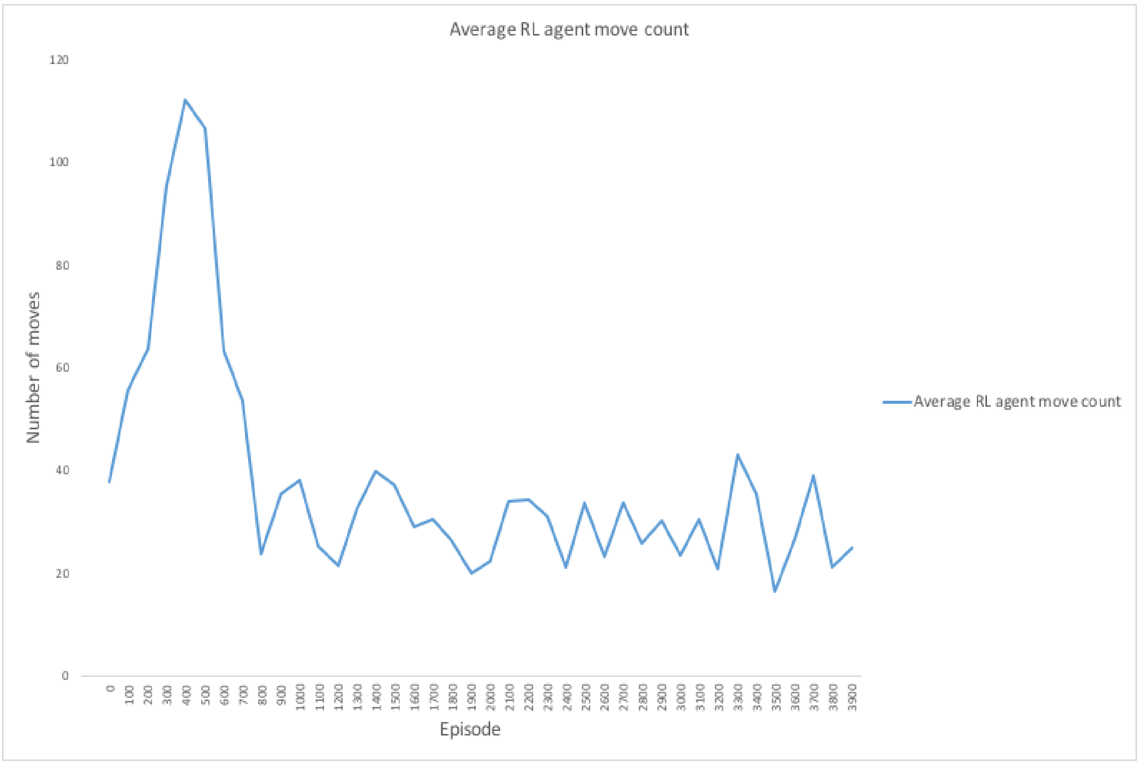


Figure 5.7: Average moves count against DE Random agent

H_1: There is stationarity.

Table 5.4: Summary statistics for ADF test

Agent	Mean	Standard deviation	ADF Statistic	P-value	Critical value 1%	Critical value 5%	Critical value 10%
RL agent	27.155	11.616	-16.213	0	-3.452	-2.871	-2.572

An ADF test on the number of moves from episode 1000 to 4000 showed that the null hypothesis of existence of a unit root could be rejected ($p-value = 0$ which is lower than the significance level 0.05). ADF statistics was $(-16.213) <$ critical value (-2.871), therefore we reject the H_0 which means the time series is stationary (Table 5.4). Therefore the observations from episode 1000 to 4000 were stationary.

Additionally for Figure 5.7 it can be observed the number of RL agents increased for the first few hundred episodes then decreased for the next few episodes before stabilising. The ADF unit root test was used to test whether the mean number of moves after episode 1000 was stationary and constant. The reason for this has been mentioned in Section 3 of Chapter 3. An ADF test was used to test the null hypothesis that there exists a unit root against the alternative hypothesis of stationarity.

H_0**:** That there exists a unit root (non-stationarity).

H_1**:** There is stationarity.

Table 5.5: Summary statistics for ADF test

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	29.090	29.909	-17.366	0	-3.452	-2.871	-2.572

An ADF test on number of moves from episode 1000 to 4000 showed the null hypothesis of existence of a unit root could be rejected with ($p-value = 0$ which is lower than the significance level 0.05). ADF statistics (-17.366) < critical value (-2.871) so we reject the H_0 which means the time series is stationary (Table 5.5). Therefore the observations from episode 1000 to 4000 were stationary.

Indeed, it can be concluded from the experiments that the RL agent is able to learn to argue with different opponents, such as the DE heuristic strategy agent and the DE Random agent using different approaches such as, winning the game or aiming to win the game with minimum number of moves.

5 Alternative reward function

Reward function in Equation 5.1 was introduced which successfully enabled the RL agent to learn to argue and minimising number of moves. In the next step we consider an alternative reward function to study the behaviour of the RL agent and compare its results with with Equation 5.1.

The suggested alternative reward function sharing the desired behaviour is given in the following equation:

$$R = \begin{cases} \frac{W}{L} & \textit{If RL agent wins} \\ -1 & \textit{Otherwise} \end{cases} \tag{5.2}$$

The state and actions remain the same ($previousmove \cup CS1 \cup CS2$) and the set of available moves are as per the DE model. Additionally In Equation 2.2 $r_{t+1} = -0.01$ is configured as a small punishment to make the RL agent choose minimum number of moves and win the game quickly. The game includes 4000 episodes starting from 0. In the initial phase where the RL agent has no explored, it behaves similar to the DE Random agent. Exploration and exploitation trade-off is the same as the previous reward function Equation 5.1 where every 100 episode stopped exploration and exploit experience and repeating this 10 times to avoid lucky choice. This is done to evaluate the experience of the RL agent and understand if it has learnt.

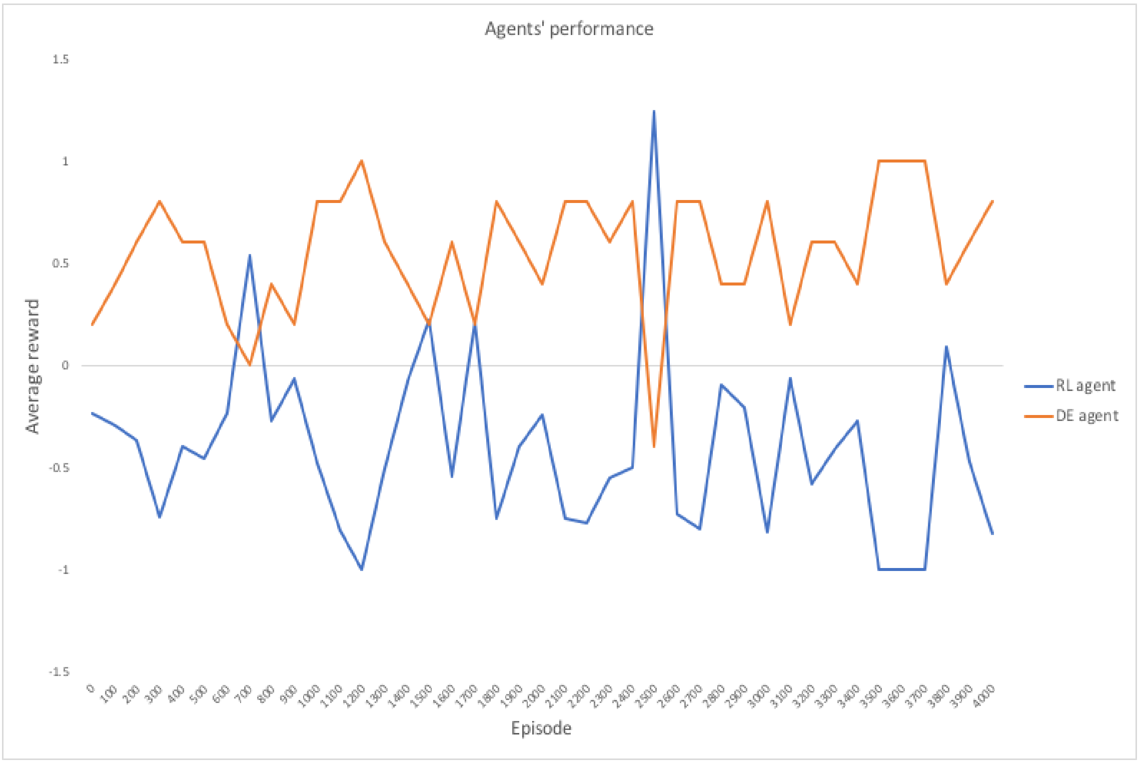


Figure 5.8: RL agent against DE

In Figure 5.8 the RL agent starts from a value -0.2 and then the curve fluctuates. In fact from this figure it is difficult to say that the RL agent is able to learn efficiently since most of the time the DE agent outperforms the RL agent. The average reward of first 1000 is almost zero which is more than average reward of the last 1000 episodes which is almost -0.6. This suggests that, since it is not rewarded enough, the RL agent was able to learn from experience (as in Equation 5.1) by maximising the the cumulative rewards.

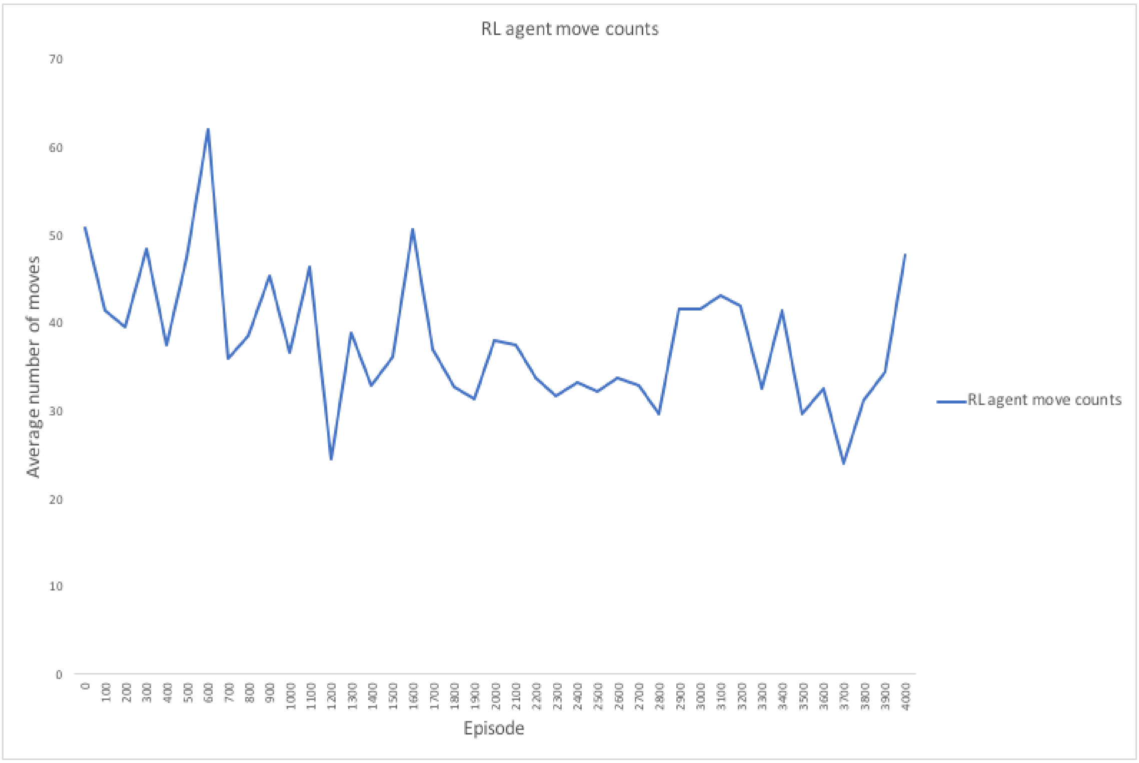


Figure 5.9: RL agent move counts

As seen in Figure 5.9, the RL agent starts from around 51 moves then decreases with small fluctuations as t plays more. The RL agent tries to minimise the moves but in the end achieves approximately the same number of moves as in the first episode. This means that it does not learn effectively due to the reward value not being enough for choosing the optimal action. Additionally, when the RL agent focused on minimising the number of moves, it started to loose the game and vice versa and therefore could not balance. Due to these confusions the RL agent could not behave well to minimise the number of moves.

Generally, the DE agent outperforms the RL agent which means that the RL could not learn to win the game. With respect to the number of moves, the RL agent tries to minimise the number of moves but in the end, it is not able to learn effectively as it achieved in the beginning. This is because, the RL agent was not given enough reward to learn properly by winning more games.

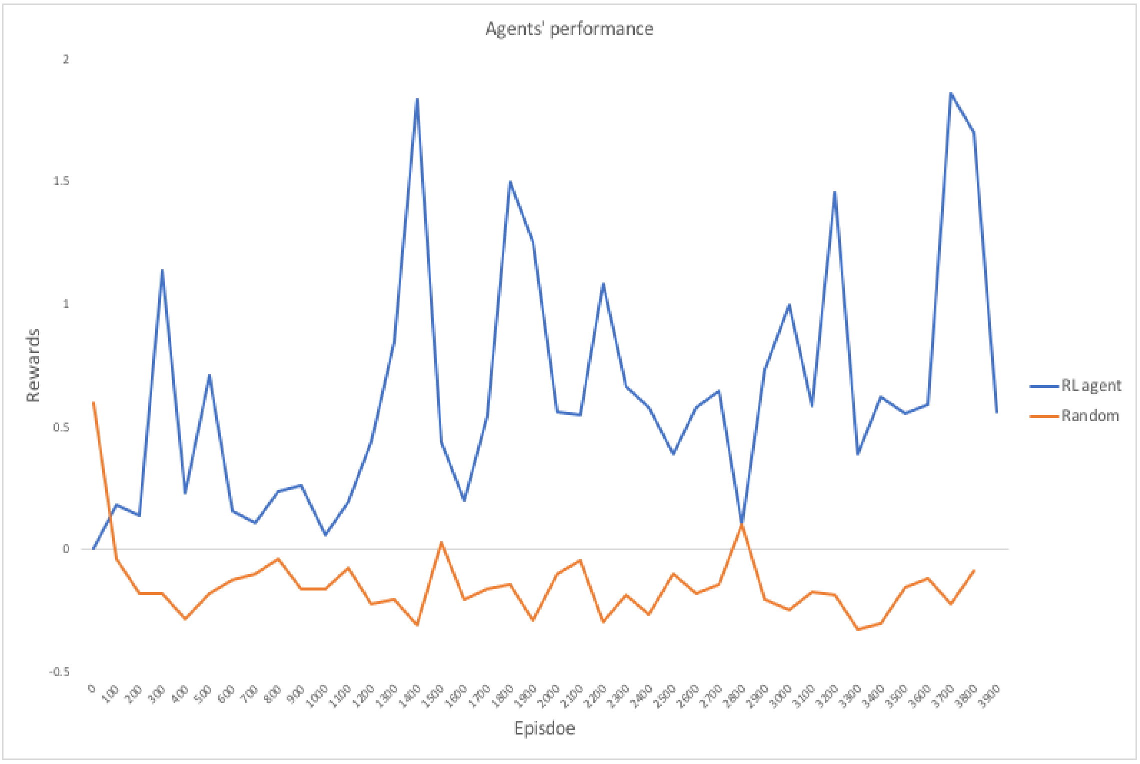


Figure 5.10: RL agent against DE Random

As shown in Figure 5.10 the RL agent played against the DE Random agent. The RL agent starts from a value of 0 and tries to learn with time. However, it is not able to identify an optimal policy to converge and stabilise even as it outperforms the DE Random agent. The reason for this is that, the RL agent defeated the opponent easily but, since it is still exploring, it is not able to perform well.

From the Figure 5.11, the RL agent starts from 25 moves, however its behaviour fluctuates since it did not converge to the optimal policy. Despite trying, the reward function does not motivate the RL agent to minimise the number of moves by exploring to find the optimal action. When the number of moves of the first and the last episode (i.e. 27 moves and 34 moves respectively) are compared, the RL agent did not learn effectively. Hence, the RL agent is not confident with the policy

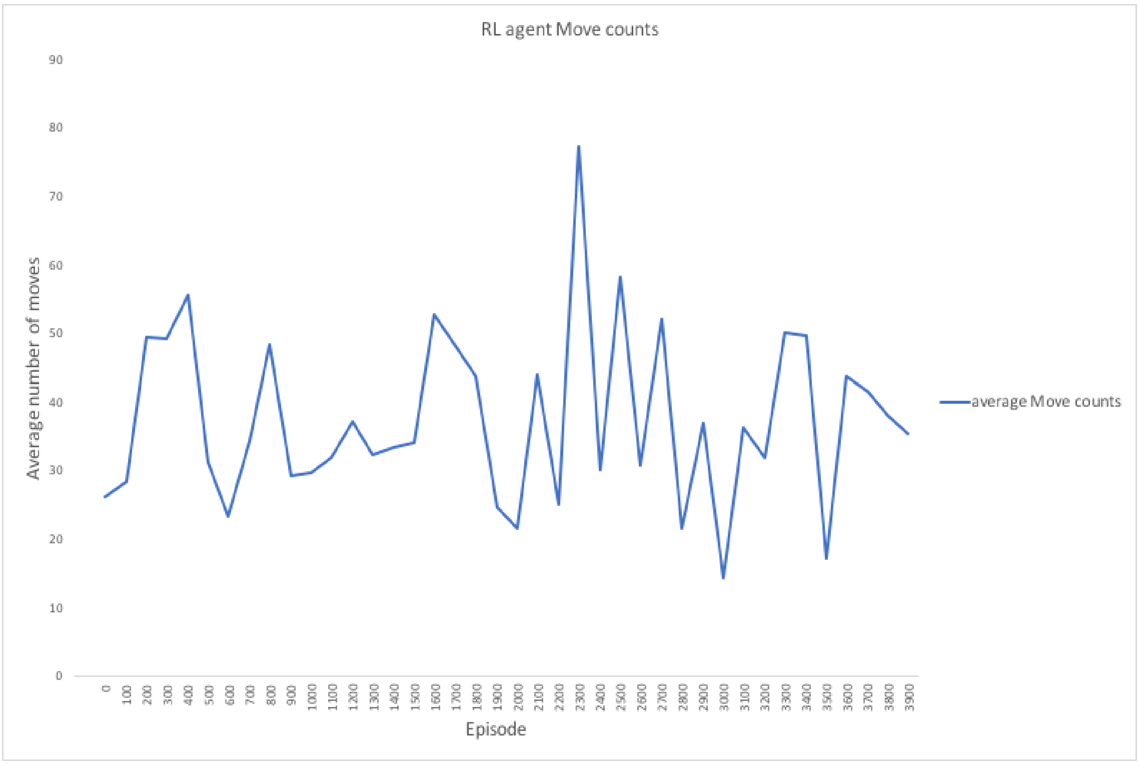


Figure 5.11: RL agent move counts

and therefore cannot predict the next move correctly by minimising number of moves.

By comparing these results with the earlier results of reward function in Equation 5.1, it shows that the initial reward function enhances the RL agent to behave better. When the RL agent played against the DE heuristic agent, the RL agent in Equation 5.1 learns to win with a reward of 100 which is very high compared the reward in Equation 5.2 with respect to the number of moves. The first priority in Equation 5.1 is to win the game then reduce the number of moves. The behaviour of the RL agent in the initial reward function starts from negative and learns and adapts its behaviour to the DE heuristic agent and eventually starts winning by finding accurate moves for all states. However, in the alternative reward function Equation 5.2, the RL agent had very low reward compared to the initial reward for winning and only -1 for losing. Therefore, when the RL agent tried to win with "number of moves" it was not able to follow that, and when the "number of moves" decreased, the agent started to lose the game. Therefore, the RL agent was not able to converge and outperform the DE heuristic agent as given in Equation 5.1.

We compare minimising the number of moves for the RL agent in both reward functions (Equation 5.1 and 5.2) in Figures 5.6 and Figure 5.9 respectively. As in Figure 5.6, the number of moves increase in the beginning and then start decreasing. Whereas, the behaviour of the RL agent in Figure 5.9 shows that the number of moves are increasing and decreasing. In the initial reward function Equation 5.1 the main priority is to win the game which requires to increase the number of moves. Once the agent learns and adapts to winning, it then focuses on reducing the number of moves. This is because, the winning gets high priority with a high reward value. However, in the latter reward function in Equation 5.2, when the RL agent focused on reducing the number of moves, it starts losing the game and when it focuses on winning it starts increasing the number of moves. This confused the RL agent and it was not able to trade off. Hence, it was not able to take a decision leading to the fluctuations. In fact, this seems to be the case for Figures 5.7 and Figure 5.11, when RL agent reduces the number of moves when it played against the DE Random agent.

The behaviour of the RL agent in the earlier reward Equation 5.1 is better than the latter reward function Equation 5.2 when played against DE Random agent. In the first reward function, the RL agent starts from negative value and is able to (acquire or get) positive reward values and outperform the DE Random agent. Whereas in the second reward function there is not much difference between the RL agent and the DE Random agent (as it is aggregate reward so that means sometimes RL agent also lose). This is because, the DE Random agent has a totally random behaviour and it is unpredictable. Therefore, the RL agent tried to learn and not lose the game, however, it is not able to have a high reward as it is still losing from new states. Whereas in the alternative reward function the RL agent cannot learn since it is not able to get enough rewards on winning and agent tries to reduce the number of moves.

6 Summary

In this chapter, the DE dialogue model was used to involve the RL agent in playing an argument game against DE baseline agents in the *capital punishment* domain.

The aim to improve the performance of the RL agent, was achieved. It also successfully enabled the RL agent to win with minimum number of moves. The reward function was reshaped (Equation 5.2) to consider an alternative reward function and studied. However, the RL agent did not behave satisfactorily against both the DE baseline agents. When compared with the initially proposed reward function given in Equation 5.1, the initial reward behaves better because the reward that is given to the RL agent is of high value. This encourages the RL agent to converge to the optimal action. Also, the agent becomes aware of a bad move because the punishment is of very high value (-100). These encouraging results motivated further exploration of learning with improved quality of dialogue with attributes such as coherence and relevance. These will be discussed in the following chapter.

CHAPTER 6

LEARNING TO IMPROVE DIALOGUE COHERENCE AND RELEVANCE

In the previous chapter the improved performance of the RL agent winning in minimum number of moves using both heuristic and randomised strategies was demonstrated. As a next step, it is worthwhile to look more deeply into the dialogue itself and investigate the quality of the dialogue. In the literature, measurements such as coherence and fluency are used to understand the quality of the dialogue and examine how the RL agent learns to make high quality dialogue contributions. These will be covered in the present chapter.

1 Introduction

The goal of this research is to make the RL agent learn to argue. In the previous chapter it was showed that the RL agent was able to argue to win with a minimum number of moves. This work is now being extended to other goals, such as improving the quality of the dialogue [224].

Previously, criteria such as, coherence and relevance for persuasion dialogues have been used for measuring dialogue quality [224–226]. Amgoud *et al.* [225] suggest that these measures can be used for different protocols and different strategies within different agents to evaluate the dialogue quality. In the DE game, it is important to implement some of these criteria in the RL agent to investigate the quality of the dialogue [7, 44–46].

In the present research, a knowledge base was built into the DE dialogue model [7] and each agent tried to persuade its opponent to believe its point of view. Indeed each agent attempted not to contradict itself [225, 227, 228], which means that it tried to make its debate coherent [229]. Hence, each agent attempted to stay coherent during the dialogue. Additionally, relevance is considered useful for assessing dialogue quality [109, 225, 226]. Relevance refers to all contributing arguments that are related to the main topic [229]. In [230], Walton states that, relevance is when one speaks of arguments, or makes moves in argumentation, that are considered to be logically or objectively flawed in some sense. Therefore, as arguments or moves that should be subject to criticism. Additionally Kok *et al,* [109] define relevance based on, attacking move and surrounding move whereas attacking move is relevant in a dialogue *iff* it changes the move status of the proposal and surrounding move is relevant *iff* its attacking counterpart is relevant.

However, in the DE dialogue game [7] all arguments in the knowledge base were related to the main topic, which was *capital punishment*. Since one of the strategies of the hard-coded agent in the DE dialogue game was switching-focus, it is therefore necessary to define relevance here as maintaining the same dispute line without switching focus [5] [5, 98].

2 Coherence and relevance

Amgoud *et al.* [225] present three different measures of dialogue quality as: the quality of exchanged arguments; the agent's behaviour (for instance coherence and aggressiveness); and, the quality of the dialogue itself, such as its relevance. Amgoud *et al.* [225] suggest that the dialogue measures are important as they can be used as guidelines for agents making high quality dialogue contributions. Weide [231] supports using the measures in [225] as benchmarks for the agent to make a decision on which dialogue moves to choose. In Chapter 5 it was proposed that the RL agent should learn how to argue. Towards understanding the learning performance of the RL agent, it is worthwhile to consider some of the criteria that are used to measure the dialogue quality of the RL agent, such as coherence and relevance [225]. It is believed that the measures of coherence and relevance could be used to evaluate the RL agent as it learns to argue in minimum number of moves, while also contributing to high quality dialogue against baseline opponents [4].

In Chapter 5 it was shown that the RL agent learnt how to argue against different baseline agents and persuade the other party to believe its point of view. However, in order to understand the effectiveness of the approach, it is required to measure and assess the quality of the dialogue. Coherence is one of the measures for dialogue quality. Coherence can be identified based on the persuasiveness of the dialogue in which an agent tries to defend its point of view or what it believes, and does not contradict itself [94, 225, 232]. Coherence in a dialogue can be measured through incoherence, where less incoherence in the dialogue suggests more coherence [225]. Incoherence, therefore indicates more contradiction throughout the dialogue. In our earlier work [4], a formula 6.1 to measure the percentage of incoherence based on [225] has been proposed as:

$$AgentIncoherent = \frac{C}{M} \tag{6.1}$$

where:

C is the number Of Contradictions.

M is the number Of Moves.

Equation 6.1 assesses the percentage incoherence, which depends on how many times the agent contradicts itself throughout the dialogue divided by the number of moves the agent makes.

On the other hand, relevance for the agent in a dialogue can be defined as - making a move that does not deviate from the topic of the dialogue [226]. Since, all moves made by the agent in the DE dialogue game [7] are related to the main claim [4], a way of measuring the relevance of the agents after allowing the RL agent to learn how to argue is required [4]. One of the strategies in the DE dialogue game [7, 98] is to change the current dialogue focus. Therefore, it is important to minimise the focus change for the players. It can be argued that switching focus will result in less fluent dialogue. Therefore, a new formula is proposed (Equation 6.2) to measure the irrelevance of each agent which is based on the number of times an agent switches focus throughout the dialogue with respect to the number of moves as:

$$AgentIrrelevance = \frac{SF}{M} \tag{6.2}$$

where:

SF is the number of focus switches.

M is the number of moves.

However, the proposed equations for incoherence and irrelevance need to be tested using experiments. The experiments involve, the RL agent playing against two different manually constructed baseline agents that are based on DE heuristic strategy and DE Random strategy [7]. The DE heuristic strategy baseline agent used a fixed heuristic strategy and the DE Random strategy agent used a random strategy. Coherence and relevance of dialogues were estimated by calculating incoherence and irrelevance using Equations 6.1 and 6.2 respectively. Incoherence was calculated by tracking the number of contradictions made by each agent. Whilst irrelevance was calculated as the number of times each agent switched focus. This corresponds to the number of occasions an agent did not address the previous move. Therefore, the lower the number of focus switches, the more focused the agent is in the dialogue and therefore more fluency [4].

As a first step, an experiment was conducted using the RL agent playing against baseline agents to measure the quality of the dialogue using Equations 6.1 and 6.2. To measure incoherence, the RL agent engaged in a game with 4,000 episodes where each episode was considered a debate episode between the RL agent and an opponent (i.e. either DE heuristic strategy or DE random). To trade-off between exploration and exploitation, the RL agent was allowed to test the learned policy every 100 episodes and this test was repeated 10 times to avoid random luck. The incoherence and irrelevance measured in this experiment are shown in Figures 6.1 and 6.2. The dialogue quality measures (based on the above equations) are shown in the results after every 500 episodes for representative purposes.

The x axis in Figure 6.1 represents the number of episodes, while the y axis denotes the incoherence measure for agents that are calculated as per Equation 6.1. In Figure 6.2, the x axis denotes the number of episodes and y axis represents the irrelevance measurement as per formula given in Equation 6.2. The results demonstrate that the RL agent did not improve in performance in either coherence or relevance. The results also showed that the proposed measures for coherence and relevance are independent of the number of moves. These suggest that, the coherence and relevance measurements stabilise and remain at the same level when the RL agent

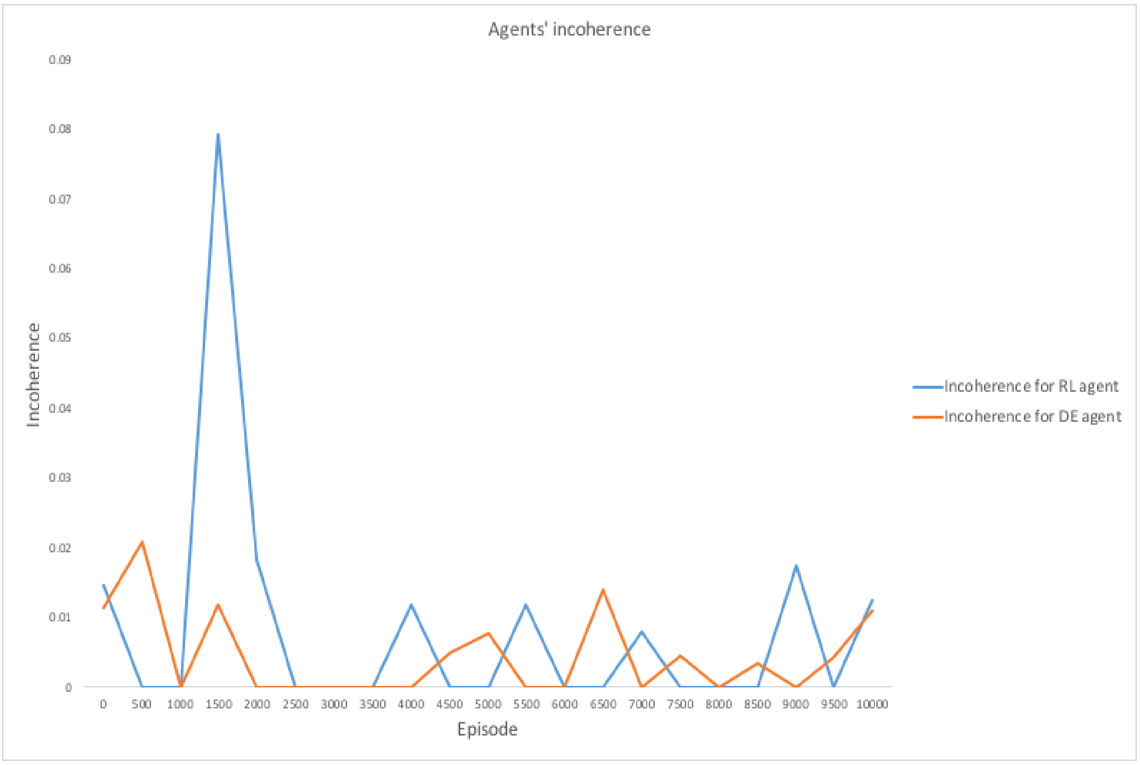


Figure 6.1: Incoherent measuring between RL agent and heuristic agent

wins the game with minimum number of moves.

To investigate whether the RL agent could improve its performance with respect to the coherence and relevance measures, these were incorporated into the reward function using Equation 5.1 described in Chapter 5:

$$R = \begin{cases} 100 + \frac{W}{L} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases}$$

3 Reshaped reward function

The coherence and relevance in Equations 6.1 and 6.2 are independent and not part of the reward function (Formula 5.1). It was shown in the previous section that the RL agent was not able to improve coherence or relevance. In order to overcome this, and improve the performance and dialogue quality of the RL agent, the incoherence and irrelevance equations were incorporated into the reward function. Hence, the

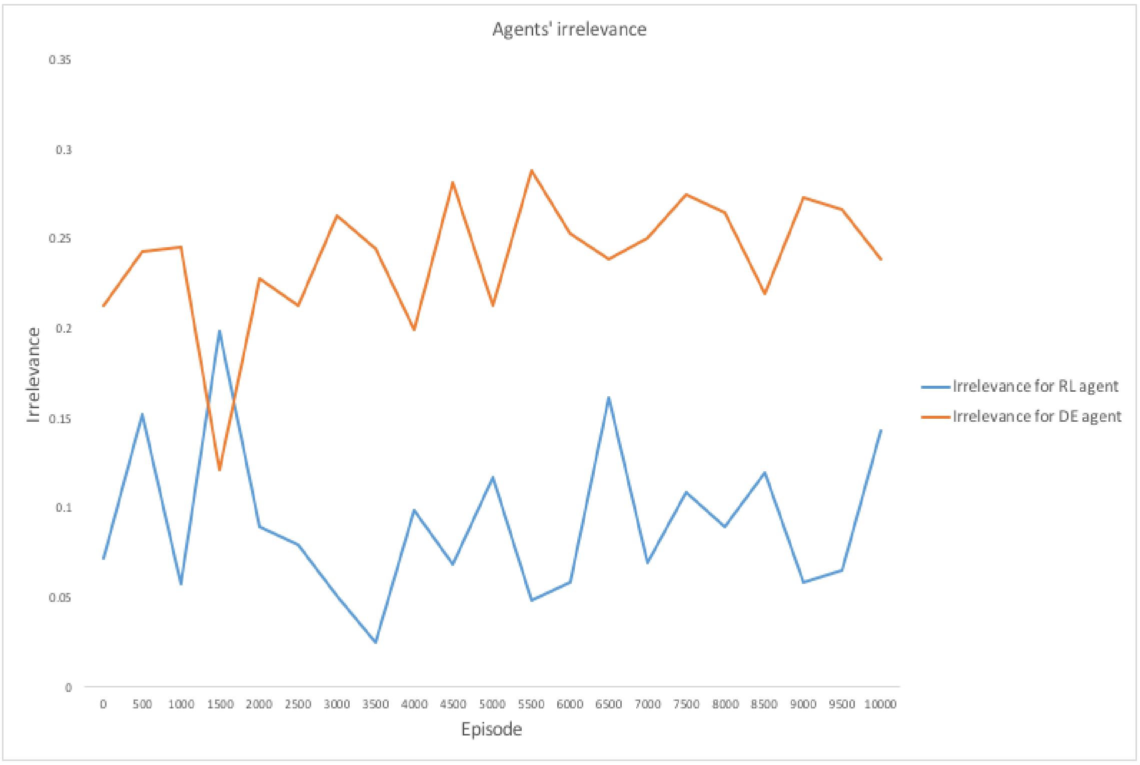


Figure 6.2: Irrelevance measuring between RL agent and heuristic agent

reward function (Formula 5.1) was reshaped, as shown in Equation 6.3

$$R = \begin{cases} 100 + \frac{M_1}{M_n} + \frac{C_1}{C_n} + \frac{SF_1}{SF_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{6.3}$$

where:

R is the reward function.

M_1 is the number of moves in the first episode.

M_n is the number of moves in the current episode.

C_1 is the number of contradictions in the first episode.

C_n is the number of contradictions in the current episode.

SF_1 is the number of switching focus in the first episode.

SF_n is the number of switching focus in the current episode.

Therefore, this reward function (Equation 6.3) was designed to motivate the RL agent to win the game faster, with minimum number of moves, with least number of contradictions, and switching focus.

To evaluate the effectiveness of the new reward function, the RL agent was used in a number of experiments against different DE baseline agents. Initially, the RL agent played against the DE heuristic strategy agent. The results of incoherence and irrelevance are shown in Figures 6.3 and 6.4 respectively.

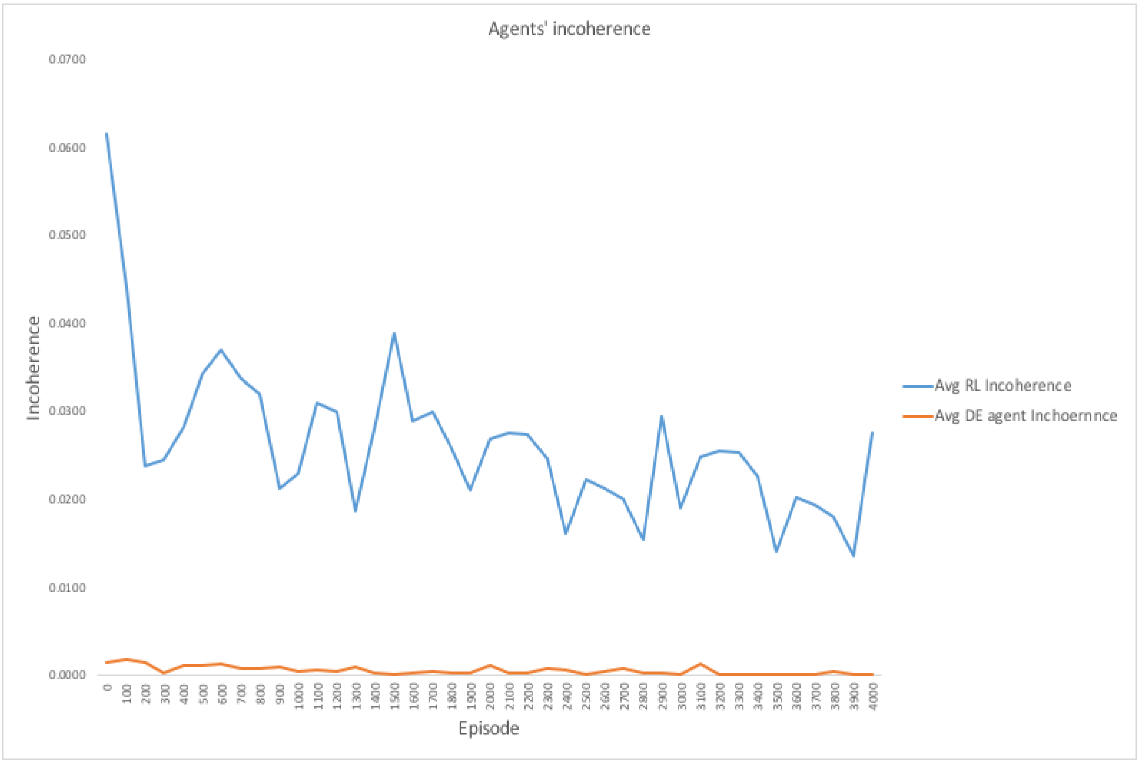


Figure 6.3: Incoherence measuring between RL agent and DE heuristic agent

Figure 6.3 shows the incoherence measurement for both the RL agent and the DE heuristic strategy agent. The gradual decrease in the incoherence measurement indicates improved coherence in the RL agent's performance. This indicates that the suggested reward function in Equation 6.3 supports the RL agent in maximising the coherence in the dialogue by decreasing the number of contradictions, whilst the DE heuristic strategy agent maintains coherence in the dialogue.

The (ADF) unit root test was used to test whether the incoherence obtained after episode 1000 was stationary and constant of the time series data. Non-parametric tests will then be used for the chosen episode to assess the significance of finding

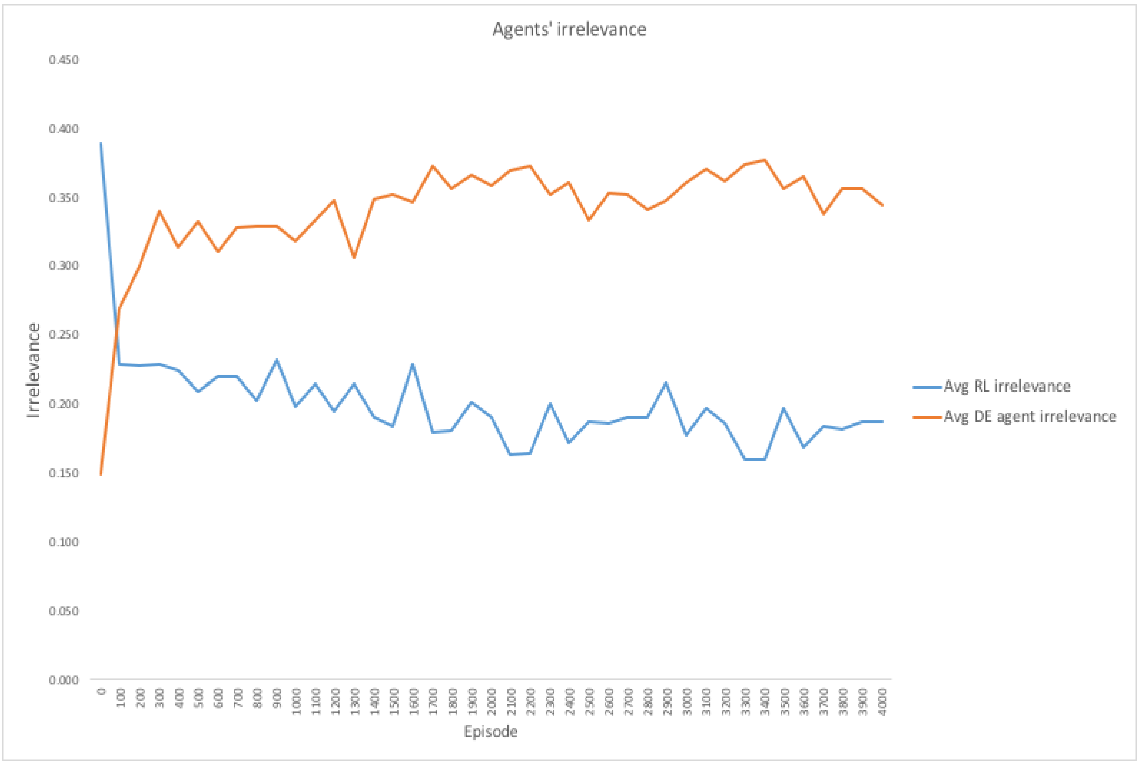


Figure 6.4: Irrelevance measuring between RL agent and DE heuristic agent

(Mann-Whitney U test). The reasons for choosing these tests are mentioned before in Section 3 in Chapter 3.

H_0: That there exists a unit root (non-stationarity).

H_1: There is stationarity.

Table 6.1: Summary statistics for ADF test (Incoherence)

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	0.028	0.010	-3.100	0.02	-3.453	-2.871	-2.572
DE agent	0	0	-16.966	0	-3.453	-2.871	-2.572

An ADF test on incoherence of RL agent showed the null hypothesis of existence of a unit root could be rejected ($p-value = 0.02$ which is less than the significance level of 0.05). ADF statistics $(-3.100) <$ critical value (-2.871) suggests that the H_0 could be rejected and indicates stationarity of the time series data. This leads to the conclusion that the incoherence of the RL agent was stationary.

Similarly, an ADF test on incoherence of DE agent showed the null hypothesis of existence of a unit root could be rejected ($p-value = 0$ which is less than the significance level of 0.05). ADF statistics (-16.966) < critical value (-2.871) indicated that the time series data was stationary and the H_0 could be rejected. These results suggest that the incoherence of the DE agent was also stationary.

Table 6.2: Incoherence values of RL agent and DE agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	0.02	0.04	0.02	0.02	0.01	0.03	0.02	0.03	0.04	0.03
DE agent	0	0	0	0	0	0	0	0	0	0

Table 6.3: Basic statistics RL agent and DE agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	0.026	0.009	0.025
DE agent	0	0	0

From the Table 6.3 the higher mean and median suggest the RL agent had higher incoherence than DE agent. A Mann-Whitney U test was used to test the null hypothesis that the two groups are homogeneous against the alternative hypothesis the RL agent had higher incoherence than DE agent.

H_0: The two groups are homogeneous.

H_1: The RL agent had higher incoherence than DE agent.

A Mann-Whitney U test showed that the null hypothesis of the two groups being homogeneous could be rejected as the ($p-value = 0.0001$ which is less than the significance level of 0.05). This leads to the conclusion that the RL agent had higher incoherence than the DE agent. However, though the DE agent shows better incoherence than RL agent, the RL agent is found to show improvement if first episode is compared with the last episode.

Looking at the relevance measurement in Figure 6.4, it was astonishing to note that in the first appearance the RL agent stayed more relevant than the DE heuristic strategy agent. An inspection of the dialogue transcript revealed the reason for this as, the DE heuristic strategy agent being more aggressive and asking a large number of questions. By passively answering the questions, the learning agent constantly stayed focused.

The (ADF) unit root test was used to test whether the irrelevance obtained after episode 1000 was stationary and constant in the time series data. Non-parametric test will be used for the chosen episode to assess the significance of finding (The Mann-Whitney U test). The reasons for chosen these tests are mentioned before in Section 3 in Chapter 3.

H_0: That there exists a unit root (non-stationarity).

H_1: There is stationarity.

Table 6.4: Summary statistics for ADF test (Irrelevance)

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	0.185	0.041	-10.401	0	-3.452	-2.871	-2.572
DE agent	0.344	0.040	-9.579	0	-3.452	-2.871	-2.572

An ADF test on irrelevance of RL agent showed that the null hypothesis of the existence of a unit root could be rejected with ($p-value = 0$ less than significance level 0.05). ADF statistics (-10.401) < critical value (-2.871), therefore, we reject the H_0 which means the time series is stationary. Therefore the irrelevance of RL agent was stationary.

An ADF test on irrelevance of DE agent showed that the null hypothesis of existence of a unit root could be rejected ($p-value = 0$ less than the significance level 0.05). ADF statistics (-9.579) < critical value (-2.871), therefore, we reject the H_0 which means that the time series is stationary. Therefore, the irrelevance of DE agent were stationary.

From the Table 6.6 the higher mean and median suggest the DE agent had higher

Table 6.5: Irrelevance values of RL agent and DE agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	0.20	0.15	0.23	0.13	0.21	0.16	0.18	0.19	0.26	0.18
DE agent	0.32	0.42	0.29	0.36	0.37	0.35	0.30	0.32	0.33	0.39

Table 6.6: Basic statistics RL agent and DE agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	0.19	0.038	0.19
DE agent	0.35	0.041	0.34

irrelevance. A Mann-Whitney U test was used to test the null hypothesis the two groups were homogeneous against the alternative hypothesis the DE agent had higher irrelevance than RL agent.

H_0**:** The two groups are homogeneous.

H_1**:** The DE agent had higher irrelevance than RL agent.

A Mann-Whitney U test showed the null hypothesis the two groups were homogeneous could be rejected with $p - value = 0.0001$, which is less than significance level (0.05). Therefore the DE agent had higher irrelevance than RL agent.

A subsequent experiment was conducted between the RL agent and DE Random agent. Similar experiments were done between the learning agent and the DE Random agent. The results shown in Figures 6.5 and 6.6 for incoherence and irrelevance respectively. Figure 6.5 confirms the result in Figure 6.3. However, 6.6 shows that the learning agent improved relevance by decreasing irrelevance.

The (ADF) unit root test was used to test whether the incoherence obtained after episode 1000 was stationary and constant in the time series data. Non-parametric test was used for the chosen episode to assess the significance of finding (The Mann-Whitney U test). The reasons for chosen these tests are mentioned before in Section 3 in Chapter 3.

H_0**:** That there exists a unit root (non-stationarity).

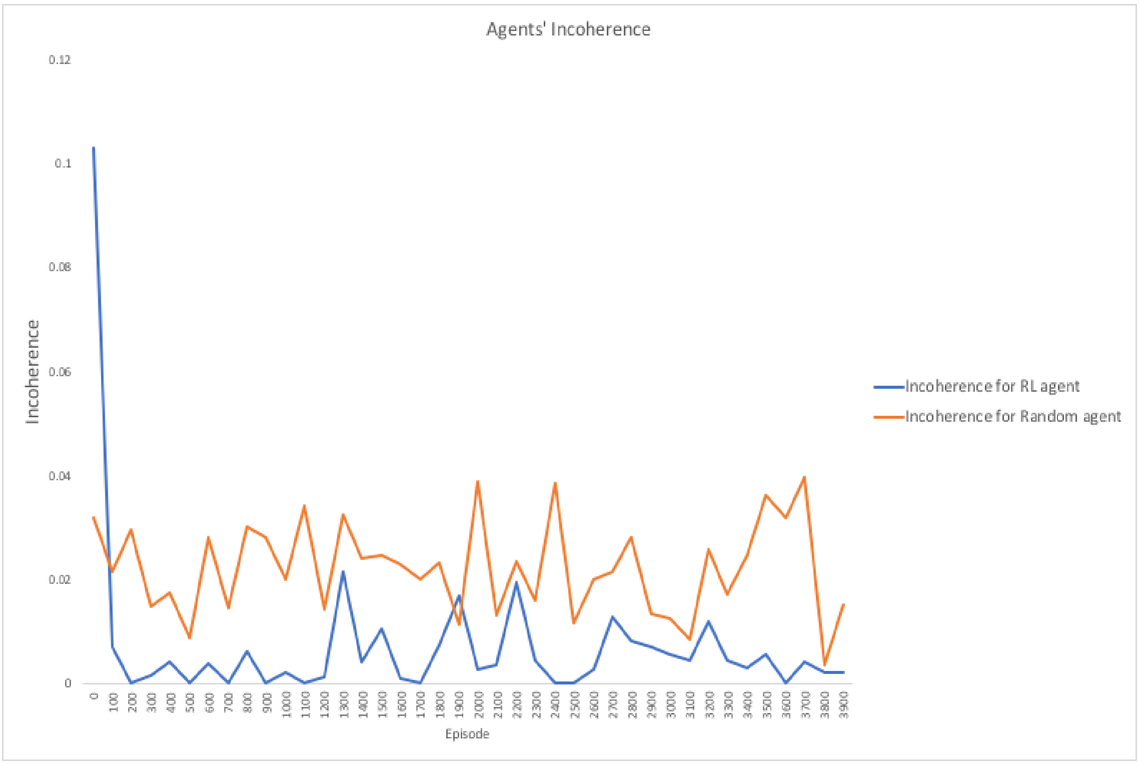


Figure 6.5: Incoherence measuring between RL agent and DE Random agent

H_1**:** There is stationarity.

Table 6.7: Summary statistics for ADF test (Incoherence)

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	0.002	0.007	-4.588	0	-3.452	-2.871	-2.572
DE Random agent	0.015	0.011	-4.937	0	-3.452	-2.871	-2.572

An ADF test on incoherence of RL agent showed that the null hypothesis of the existence of a unit root could be rejected with $p-value = 0$, which is less than significance level (0.05). ADF statistics $(-4.588) <$ critical value (-2.871), therefore, we reject the H_0 which means that the time series is stationary. Therefore the incoherence of RL agent was stationary.

An ADF test on incoherence of DE Random agent showed that the null hypothesis of the existence of a unit root could be rejected with $p-value = 0$, which is less than significance level (0.05). ADF statistics $(-4.937) <$ critical value (-2.871), therefore,

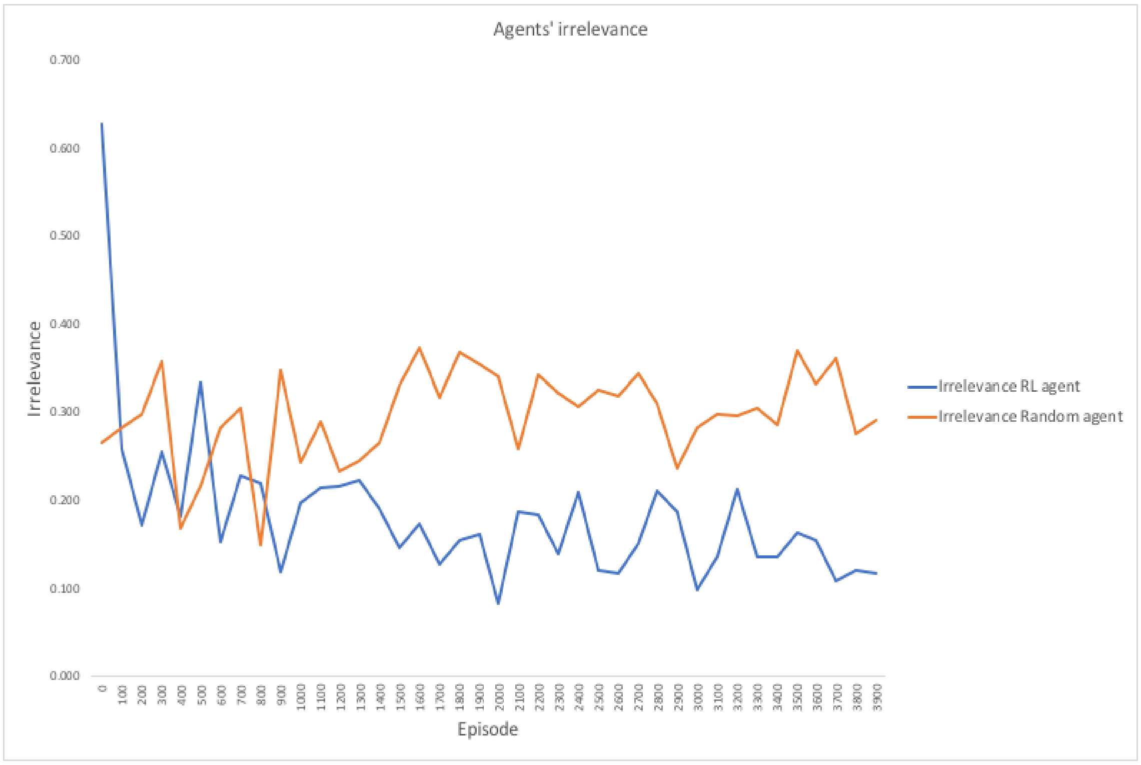


Figure 6.6: Irrelevance measuring between RL agent and randomised based agent

we reject the H_0 which means that the time series is stationary. Therefore the incoherence of DE Random agent were stationary.

Table 6.8: Incoherence values of RL agent and DE Random agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	0	0	0	0	0	0	0	0	0.020	0
DE Random agent	0.025	0	0	0.019	0.023	0.022	0.018	0.024	0.019	0

Table 6.9: Basic statistics RL agent and DE Random agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	0.002	0.006	0
DE Random agent	0.015	0.011	0.019

From the Table 6.9 the higher mean and median suggest that the DE Random agent had higher incoherence. A Mann-Whitney U test was used to test the null hypothesis (the two groups are homogeneous) against the alternative hypothesis (the DE Random agent had higher incoherence than RL agent).

H_0: The two groups are homogeneous.

H_1: The DE Random agent had higher incoherence than RL agent.

A Mann-Whitney U test showed that the null hypothesis (the two groups are homogeneous) could be rejected with $p-value = 0.016$ which is less than significance level (0.05). Therefore, the DE Random agent had higher incoherence than RL agent.

The (ADF) unit root test was used to test whether the irrelevance obtained after episode 1000 was stationary and constant in the time series data. Non-parametric test was be used for the chosen episode to assess the significance of finding (The Mann-Whitney U test). The reasons for choosing these tests have been mentioned previously in Section 3 in Chapter 3.

H_0: That there exists a unit root (non-stationarity).

H_1: There is stationarity.

Table 6.10: Summary statistics for ADF test (Irrelevance)

Agent	**Mean**	**Standard deviation**	**ADF Statistic**	**P-value**	**Critical value 1%**	**Critical value 5%**	**Critical value 10%**
RL agent	0.115	0.008	-4.997	0.00002	-3.452	-2.871	-2.572
DE Random agent	0.297	0.055	-3.073	0.02	-3.452	-2.871	-2.572

An ADF test on irrelevance of RL agent showed that the null hypothesis of the existence of a unit root could be rejected with ($p-value = 0.00002$) which is less than significance level (0.05)). ADF statistics (-4.997) < critical value (-2.871) so we reject the H_0 which means the time series is stationary. Therefore the irrelevance of

RL agent were stationary.

An ADF test on irrelevance of DE Random agent showed the null hypothesis of existence of a unit root could be rejected ($p-value = 0.02 < p$ significant at 0.05). ADF statistics $(-3.073) <$ critical value (-2.871) so we reject the H_0 which means the time series is stationary. Therefore the irrelevance of DE Random agent were stationary.

From Table 6.12 the higher mean and median suggest the DE Random agent

Table 6.11: Irrelevance values of RL agent and DE Random agent at the last episode

	Trials for the last episode									
Agent	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
RL agent	0.13	0.11	0.11	0.11	0.13	0.12	0.11	0.12	0.11	0.13
DE Random agent	0.35	0.29	0.30	0.27	0.40	0.33	0.30	0.20	0.28	0.25

Table 6.12: Basic statistics RL agent and DE Random agent at the last episode

Agent	**Mean**	**Standard deviation**	**Median**
RL agent	0.118	0.009	0.115
DE Random agent	0.297	0.055	0.295

had higher irrelevance. A Mann-Whitney U test was used to test the null hypothesis (the two groups are homogeneous) against the alternative hypothesis (the DE Random agent had higher irrelevance than RL agent).

H_0**:** The two groups are homogeneous.

H_1**:** The DE Random agent had higher irrelevance than RL agent.

A Mann-Whitney U test showed the null hypothesis the two groups were homogeneous could be rejected with $p-value = 0.000085$, which is less than the significance level (0.05). Therefore the DE Random agent had higher irrelevance than RL agent.

Therefore, it can be concluded from the experiments that the RL agent was able to learn how to argue with more fluency and coherence using the suggested reward function in Equation 6.3

4 Additional reshaped reward function

After implementing the reward function in Equation 6.3 and demonstrating promising results with improved dialogue performance, the merits of alternative reward function were systematically reviewed. It is suggested that this alternative reward function can be derived from the original reward function by dropping one from each of its four terms. These reward functions can be seen in the following table:

Table 6.13: Suggesting alternative reward functions

Reward function (Equation)	**Winning**	**Moves count**	**Contradictions**	**Irrelevance**
6.4	0	1	1	1
6.5	1	0	1	1
6.6	1	1	0	1
6.7	1	1	1	0

Equation 6.4

$$R = \begin{cases} \frac{M_1}{M_n} + \frac{C_1}{C_n} + \frac{SF_1}{SF_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{6.4}$$

Equation 6.5

$$R = \begin{cases} 100 + \frac{C_1}{C_n} + \frac{SF_1}{SF_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{6.5}$$

Equation 6.6

$$R = \begin{cases} 100 + \frac{M_1}{M_n} + \frac{SF_1}{SF_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{6.6}$$

Equation 6.7

$$R = \begin{cases} 100 + \frac{M_1}{M_n} + \frac{C_1}{C_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases} \tag{6.7}$$

From these equations (Equation 6.4, 6.5, 6.6 and 6.7) Since in the DE dialogue game, it has two agents one with fixed strategy agent (DE agent) and another agent with randomised strategy so it will consist of two main studies which are described in the following Sections 4 .1 and 4 .2:

4 .1 Experiments against DE heuristic strategy agent

In this experiment, the RL agent will play game against DE with heuristic strategy agent using the alternative reward functions mentioned above.

4 .1.1 Dropping winning

In this section, the RL agent will use Equation 6.4 which is dropped winning part (100) as seen in the original reward function 6.3

I **Performance**

From the Figure 6.7 it can be seen that the RL agent performs worse initially and then starts improving and attempts to outperform DE agent in certain games. However, in most episodes the reward is fluctuating. Overall, the RL agent experienced fluctuations of reward when it encountered a new configuration at the beginning or was not able to learn properly. In this case, the RL agent was not able to completely learn and outperform the DE agent, because (100) rewards encourages the RL agent to win more games against the DE agent within a deterministic environment.

II **Number of moves**

The RL agent in Figure 6.8 starts from around 77 moves count and then decreases. This indicates that the RL agent was able to learn to minimise

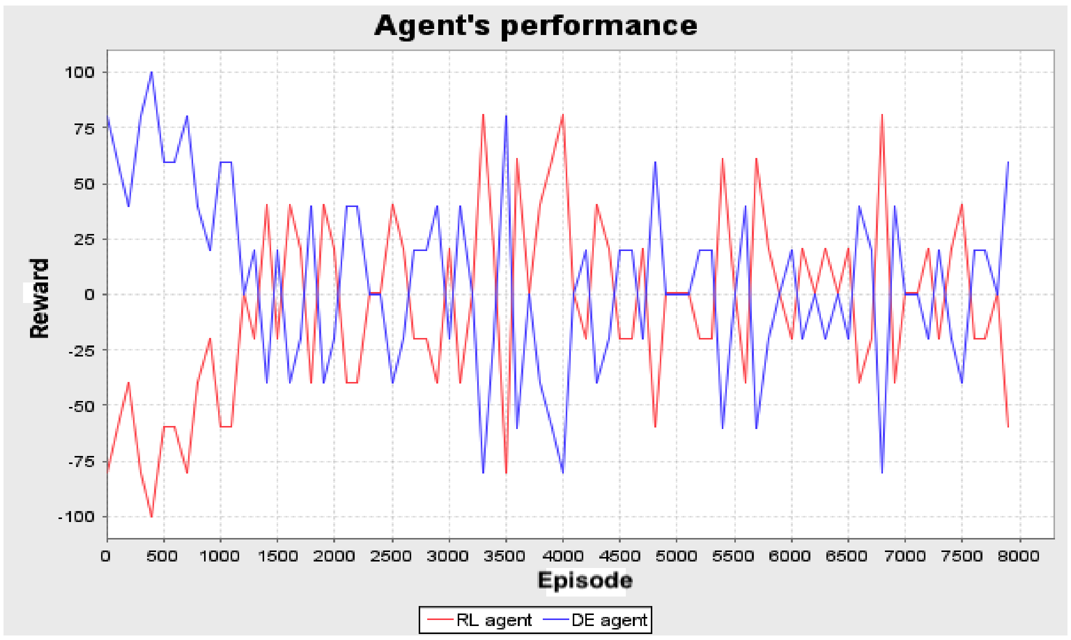


Figure 6.7: Agents' performance

number of moves when playing more until the value decreased to 17. Therefore, the RL agent can minimise number of moves using this reward function since it takes into account the minimum moves factor and hence it can be inferred that it plays with decisive moves.

III Incoherence

The RL agent in Figure 6.9 has a higher value of incoherence than DE agent and shows that RL tries to learn to minimise incoherence after episode 2000. As seen in the graph, incoherence value does not stabilise after it was minimised. Therefore, the RL agent could not learn efficiently to decrease incoherence as expected. The decreasing curve suggests that the RL agent explores and learns as it plays and therefore changes the way it plays with learning. The reason for higher incoherence could be attributed to the limitation of the RL agent to explore much to find a strategy that could minimise incoherence with other attributes.

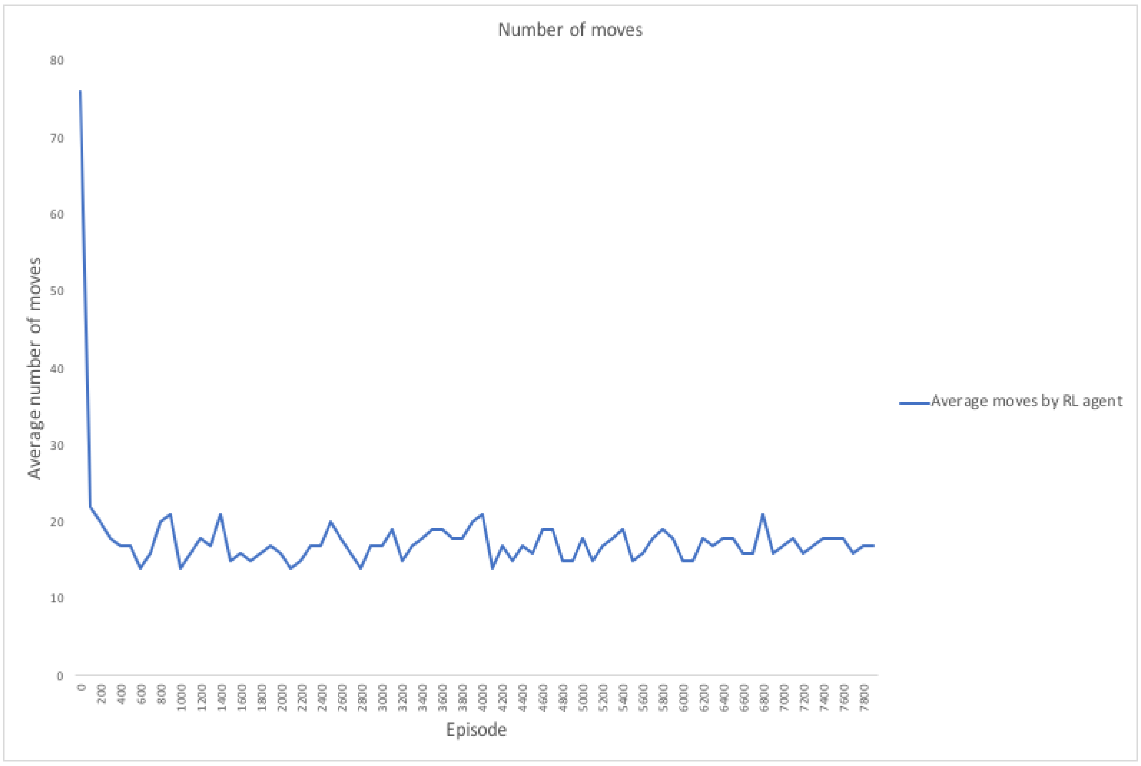


Figure 6.8: RL agent number of moves

IV **Irrelevance**

The performance of RL agent shown in Figure 6.10 demonstrated a decrease in irrelevance with time, while the DE agent showed increasing irrelevance. This suggests that the RL agent stayed focused but due to aggressively asking more questions by switching focus strategy the DE agent's focus was reduced. The value of irrelevance of RL agent was also found to decrease as more games were played. This is an indication of the RL agent continuously learning and improving itself by finding a strategy to maximise reward.

The reward function in Equation 6.4 considered removing the winning part (+100) from the original reward function in Equation 6.3. From the agent performance against DE agent, it can be concluded that the RL agent was not able to learn efficiently after 1500 episode to maximise overall rewards but it was able decrease irrelevance and number of moves. This is because, the RL agent attempts to maximise the cumulative reward function and it is not rewarding much for the agent with respect to winning. Although, the RL agent found a way to increase cumulative reward to decrease the number of moves and irrelevance, it was not able to find a

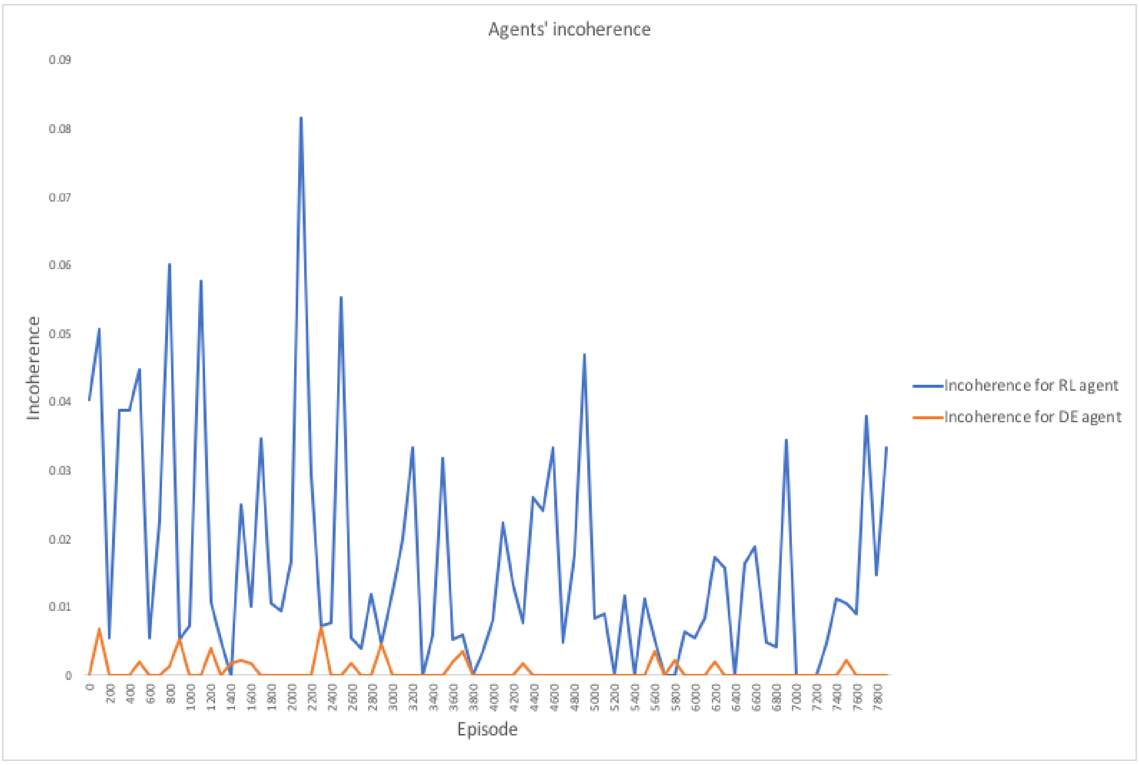


Figure 6.9: Agents' incoherence

good strategy to decrease incoherence. Therefore, the Figures 6.7 and 6.9 showed fluctuations in both incoherence and agent's performance.

4 .1.2 Dropping moves

In this case, the RL agent will use Equation 6.5 which dropped moves count as seen in the original reward function 6.3.

I Performance

In Figure 6.11, it can be seen that, the DE agent initially outperforms the RL agent. However, the performance gradually decreases and the RL agent outperforms the DE agent which stabilises after episode 4000. The RL agent was therefore able to learn to win with time and outperform the DE agent.

II Number of moves

As seen in Figure 6.12, the RL agent initially has a value of around 15 to 20, which then increases to 25. As the RL agent plays more, the performance

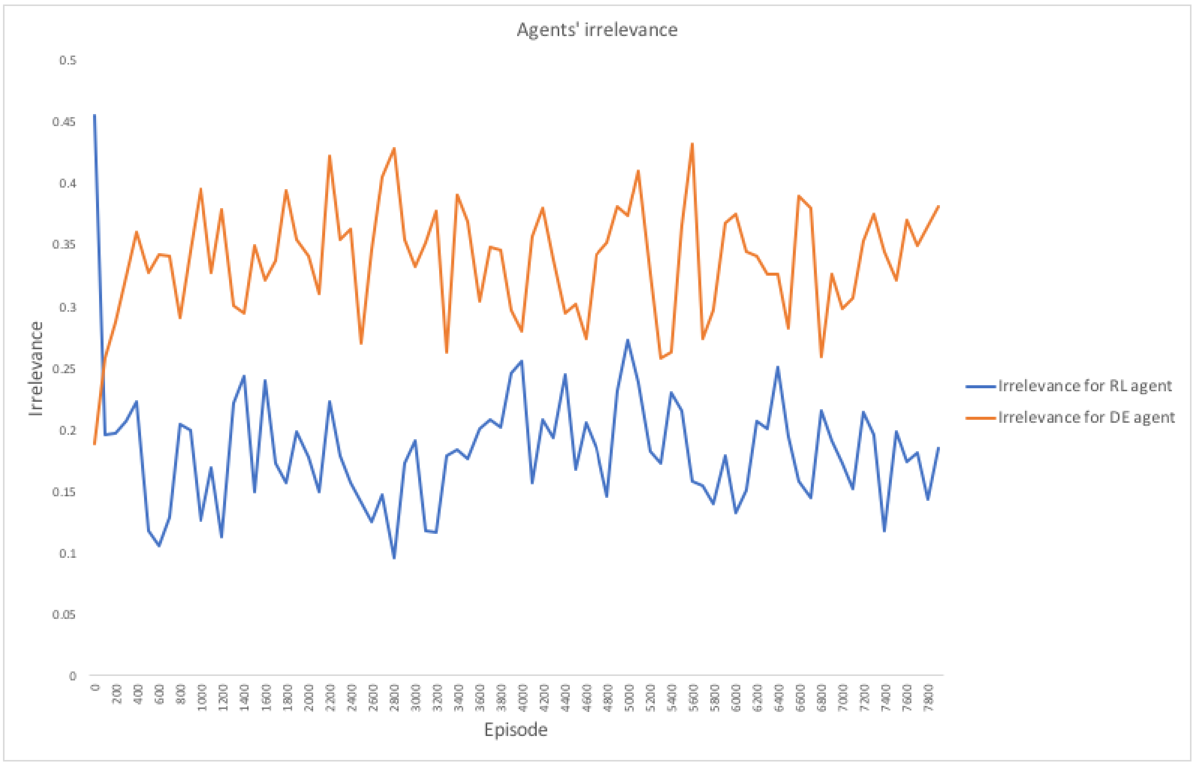


Figure 6.10: Agents' irrelevance

stabilises approximately at 18. Since this reward function Equation 6.5 did not consider the number of moves, there is an effect on the RL agent's behaviour to decrease number of moves and increase cumulative rewards without considering less moves.

III Incoherence

Figure 6.13 shows that the RL agent has higher incoherence values than the DE agent. This is because, the DE agent is deterministic and does not perform explorations. On the other hand, the RL agent performs exploration and learns as it plays which results in changing the way it plays. Additionally, the results from the above graph suggest that, as the number of moves increase, it is more likely that the contradictions will also be increased.

IV Irrelevance

Initially, the RL agent as seen in the Figure 6.14 has a higher value of irrelevance. However, as it plays some games, the irrelevance decreases and is lower

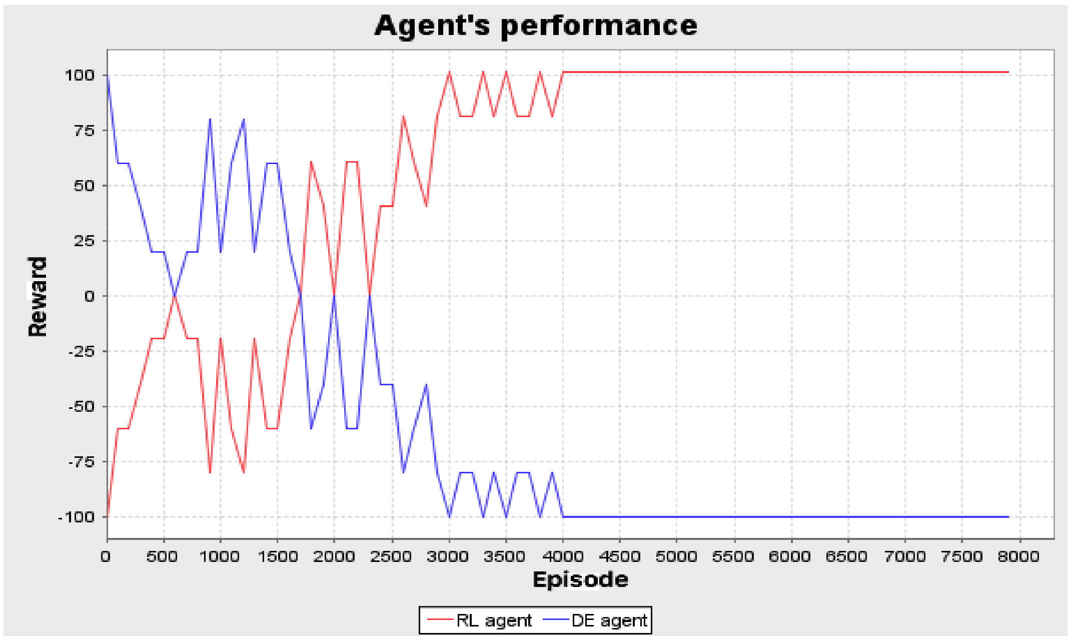


Figure 6.11: Agents' performance

than the DE agent. This suggests that the RL agent did not change focus often. Also, the value of irrelevance of the RL agent decreases as more games are played. This is because, the RL agent is continuously learning and improving itself by finding a strategy to maximise reward and the DE agent behaves aggressively by asking more questions with changing focus.

In this reward function Equation 6.5, the number of moves factor is removed from the reward function. From the agent performance with the DE agent, it can be concluded that the RL agent is learning efficiently and is able to outperform the DE agent after 4000 games. This is because the RL agent tries to maximise the cumulative reward function and that the RL agent was rewarded for winning. However, minimising the number of moves is not able to decrease irrelevance because, the number of moves factor has not been considered. Therefore, this might cause the RL agent to increase moves count. The RL agent found a way to increase cumulative reward by winning almost every game by decreasing irrelevance, however, it was not able to a find strategy for decreasing incoherence. This is possibly due to the increase in number of contradictions with increase in number of moves.

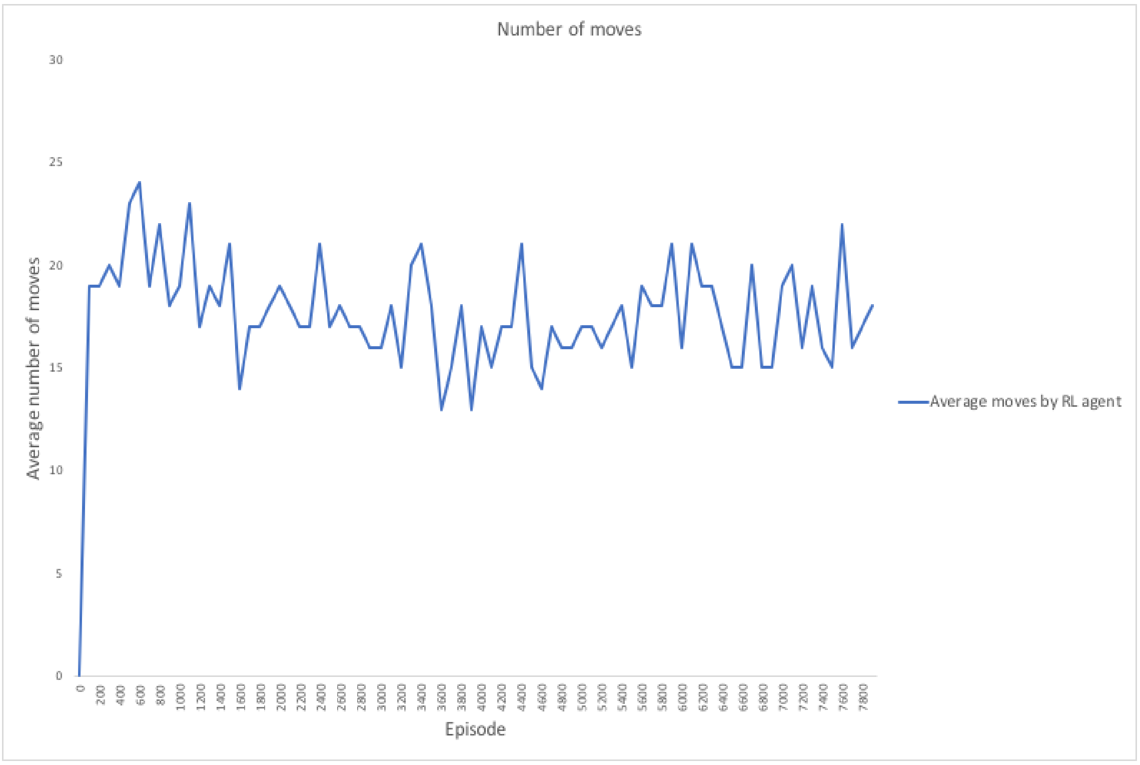


Figure 6.12: RL agent number of moves

4 .1.3 Dropping contradictions

In this section, the RL agent will use Equation 6.6 which has dropped the contradictions part from the original reward function 6.3

I Performance

In the above Figure 6.15, the DE agent was found to outperform the RL agent in the initial stages which then decreased gradually and RL agent took turn and outperformed the DE agent and later stabilised after episode 4000. Hence, the RL agent was able to learn and explore as it played that led to outperforming the DE agent.

II Number of moves

As seen in the Figure 6.16, the RL agent starts with high number of moves (around 78) which then decreases after some small fluctuations. And as it plays more, the number of moves decreases to around 16. This suggests that the RL

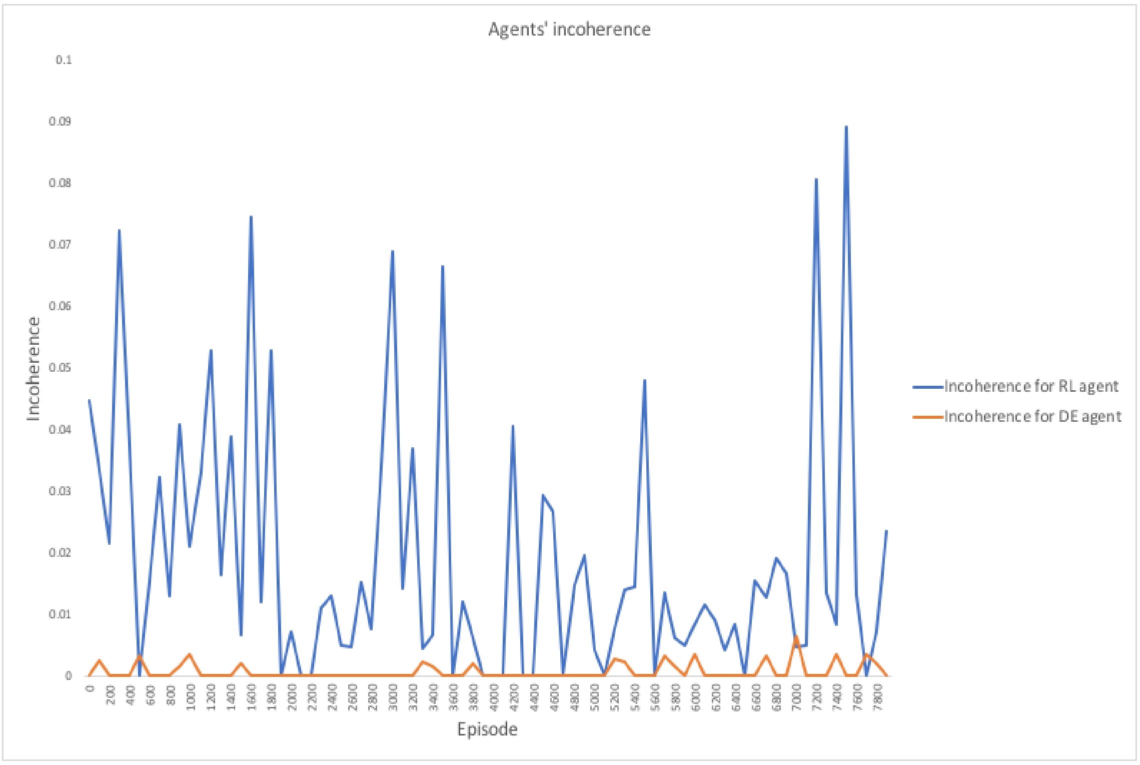


Figure 6.13: Agents' incoherence

agent is able to minimise the number of moves because it was attempting to maximise cumulative rewards.

III Incoherence

The RL agent has a higher value of incoherence than the DE agent. This is because the DE agent is deterministic and does not perform explorations. On the other hand, the RL agent performs exploration and learns as it plays. In addition, the factor of contradiction is not considered in Equation 6.3. Due to the factor of contradiction not being explicitly incorporated in the reward function, the incoherence value is found to be higher. This is the reason that we can see a higher incoherence in RL agent. As seen in Figure 6.17, incoherence starts from a very low value which then starts to increase but never stabilises as RL agent did not have a factor to minimise incoherence.

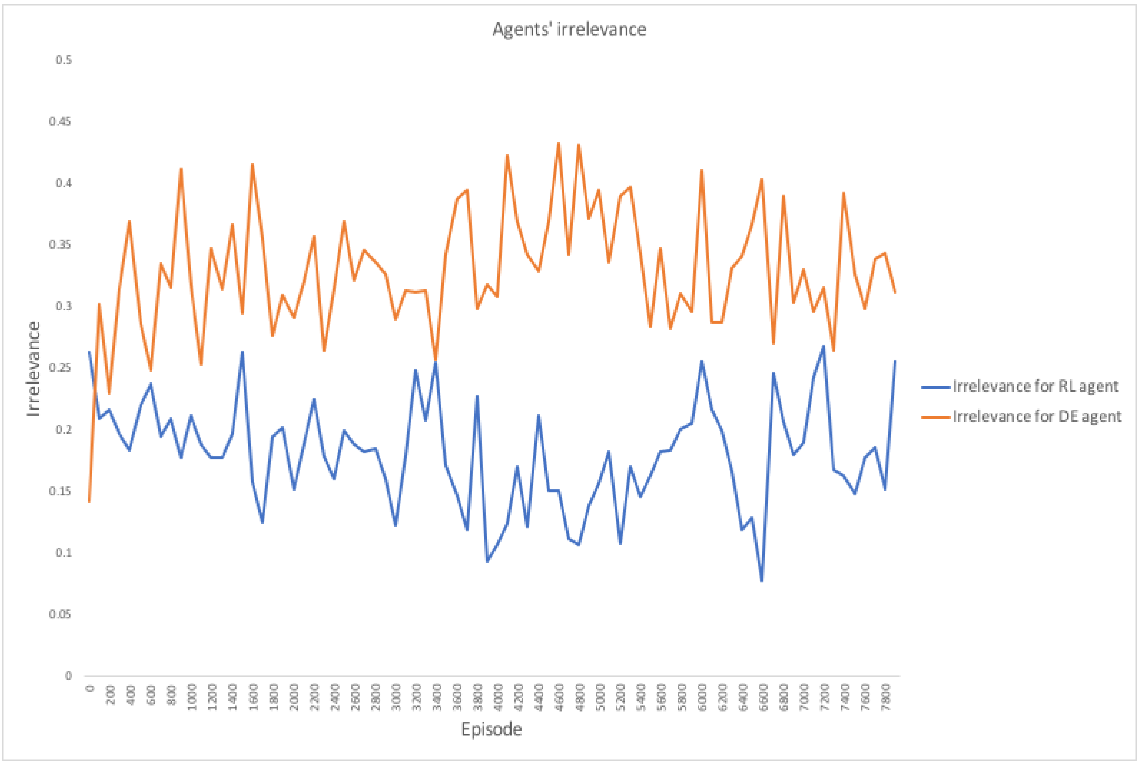


Figure 6.14: Agents' irrelevance

IV **Irrelevance**

In this context, the RL agent shows a decreased irrelevance, suggesting it stayed focus. Whereas, the DE agent has higher irrelevance as it was more aggressive in its attempt to win the game by asking more questions and switching focus strategy. The initial value of irrelevance for the RL agent was high, which then decreased to a 0.1-0.2 range.

Overall, in agents' performance, the RL agent initially performs worse then improves performance and outperforms the DE agent. The factor of contradiction is missing, which impact on incoherence behaviour of the RL agent. The RL agent finds a way to maximise the cumulative reward by winning most of the games as well as decrease moves count and irrelevance.

4 .1.4 Dropping switching focus

In this case, the RL agent will use Equation 6.7, in which the switching focus term from the original reward function 6.3 is dropped.

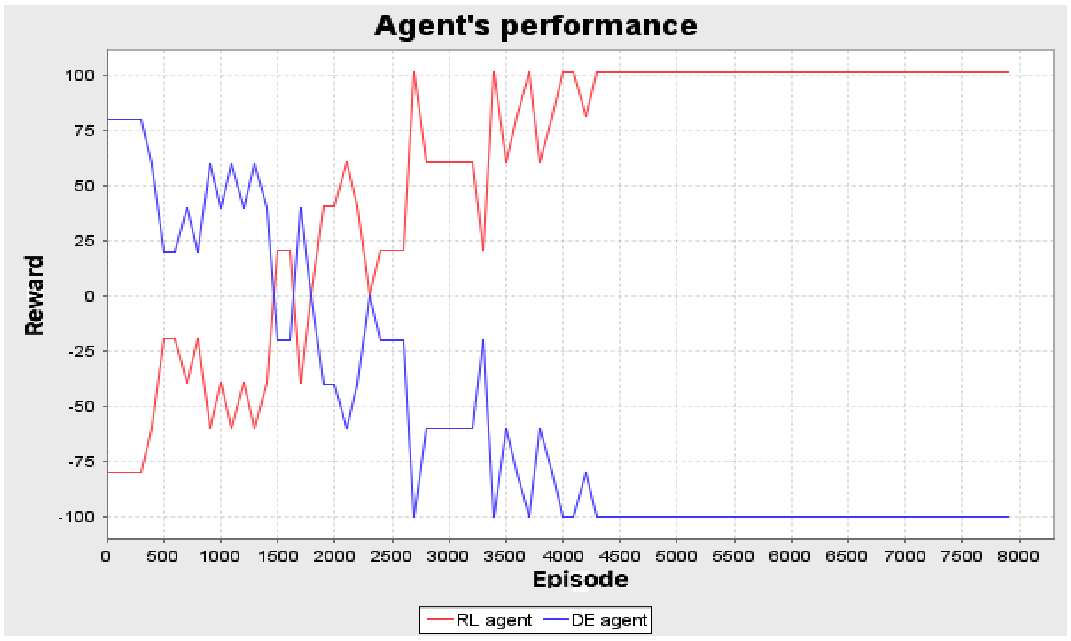


Figure 6.15: Agents' performance

I Performance

The RL agent initially performs poorly. However, it is bale to improve its performance later and outperform the DE agent. This is an expected trend because the RL agent learns to play better as it explores more games and the winning factor (+100) is considered as well. Figure 6.19 shows that the RL agent is able to continuously outperform the DE agent after 4100 episode suggesting that the RL agent wins most of the games.

II Number of moves

The Figure 6.20, shows that the RL agent starts with a value of 50 and decreases to 40. This is larger than the average number of moves in the previous reward functions. This means that the RL agent has not learnt how to finish the game in average less number of moves and hence it can be inferred that it does not perform highly decisive moves. This is because we have dropped the switching focus factor and hence it does not take that into account. However,

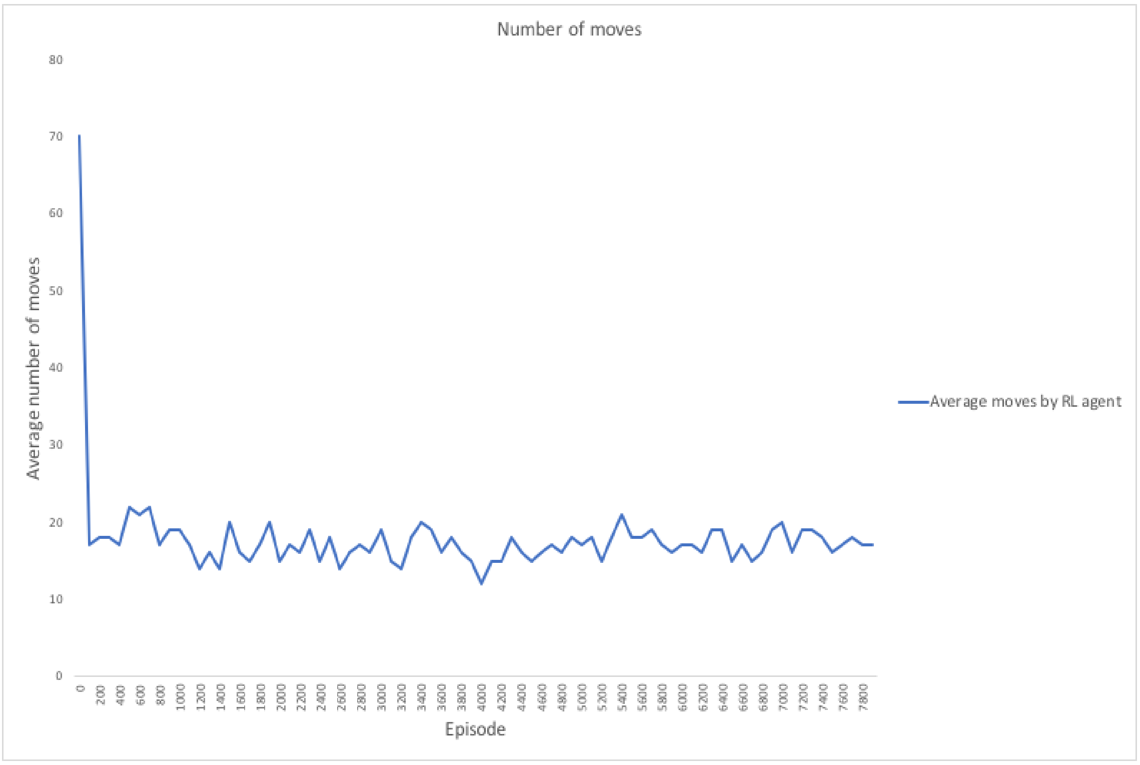


Figure 6.16: RL agent number of moves

the RL agent switched more focus often which led to an increase in the number of moves.

III Incoherence

The RL agent in Figure 6.21 improves incoherence as it is decreasing the incoherence value over the time. The RL agent has a higher value of incoherence than the DE agent. This is because the DE agent is deterministic and does not perform explorations. On the other hand, the RL agent performs exploration and learns how to minimise incoherence as it plays. In this reward function, the RL agent is able to minimise incoherence while also performing better than DE agent.

IV Irrelevance

Since switching focus factor is dropped in this reward function, the RL agent has a higher value of irrelevance than the DE agent as seen in Figure 6.22. The value of irrelevance of RL agent increases as more games are played. It

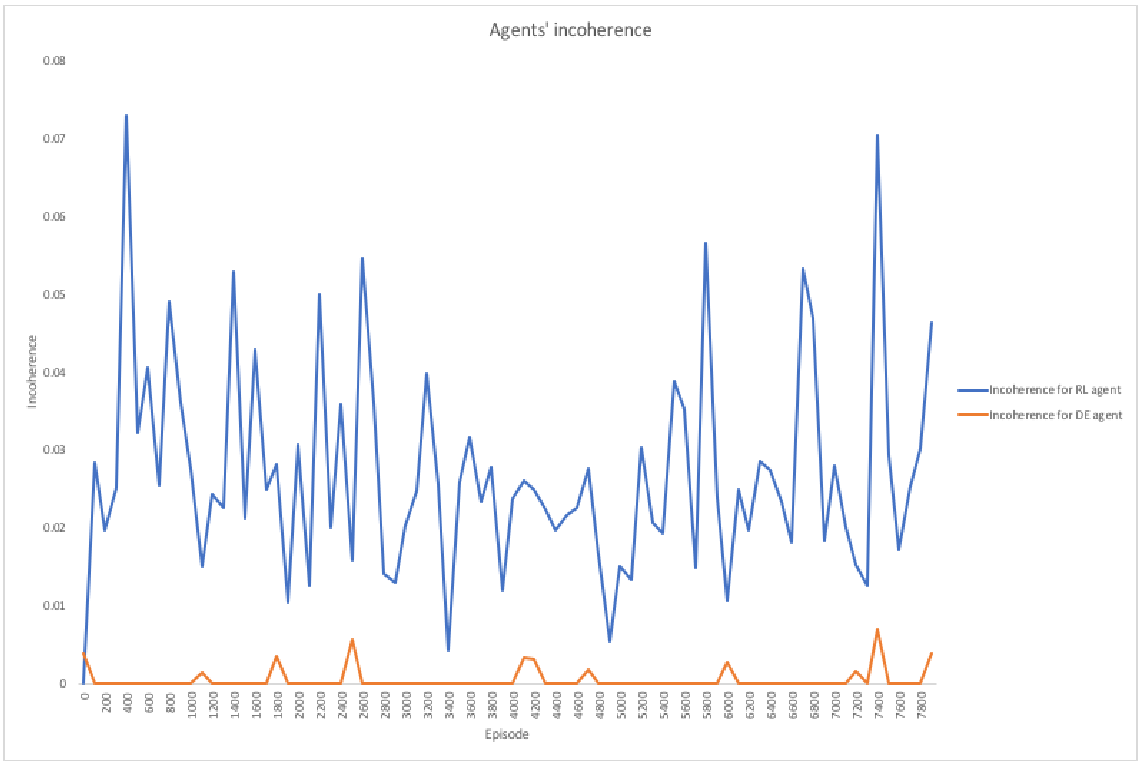


Figure 6.17: Agents' incoherence

means that the RL agent is not learning how to focus appropriately. As a result, this reward function Equation 6.7 has a negative impact on the irrelevance behaviour for the RL agent.

Initially, the RL agent performs poorly, then as it starts to play more games, it is able to learn and improve its performance. This is an expected trend because the RL agent learns to play better as it plays more games by maximising cumulative rewards. Also, the factor of switching focus is not considered in this equation. As it is not explicitly incorporated in the reward function, the irrelevance value is also higher. This also negatively impacts the moves count leading to higher moves count than previous reward functions given in Equation (6.4 and 6.6) suggesting more switching focus has happened. At the start, the initial incoherence value was higher, but as the RL agent played and explored with more games it was able to learn to minimise incoherence.

By comparing these results with the original reward function in Equation 6.3 which takes into account all the attributes, the results are different. The original reward function tries to reward the RL agent as it wins with a high value and gives negative

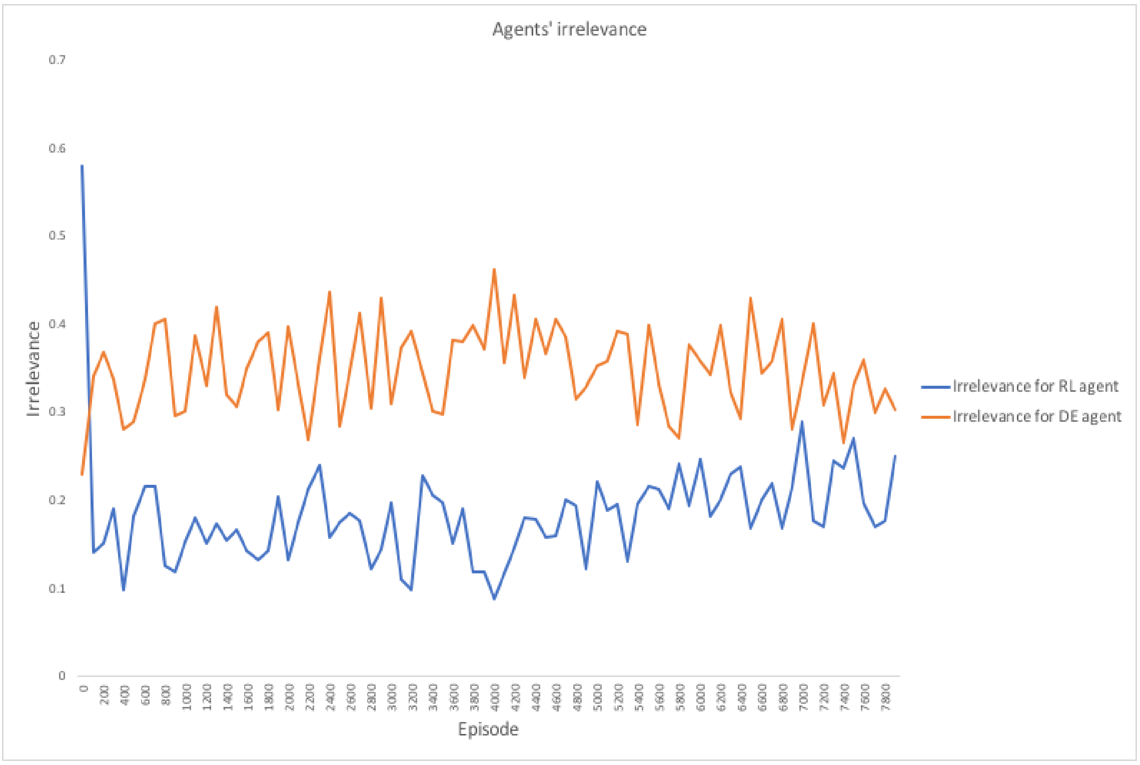


Figure 6.18: Agents' irrelevance

reward as it loses, minimise the number of moves, incoherence and irrelevance by giving appropriate reward. Therefore, the RL agent achieves the aim by maximising the cumulative reward. The original reward function is more robust and takes into account all the factors based on the results.

4 .2 Experiments against DE random strategy agent

In this experiment, the RL agent will play against the DE with random strategy agent using the alternative reward functions which are Equations (6.4,6.5,6.6 and 6.7).

4 .2.1 Dropping winning

In this section, the RL agent will use Equation 6.4 in which the winning part (100) is dropped as seen in the original reward function 6.3.

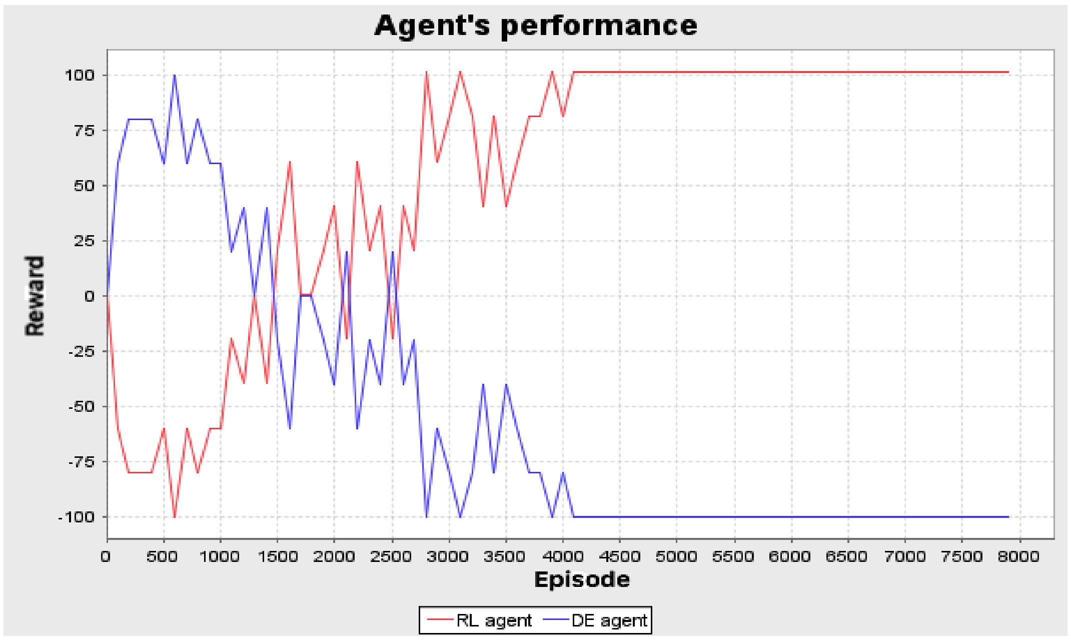


Figure 6.19: Agents' performance

I Performance

Figure 6.23 illustrates that, the RL agent initially performs poorly, and then starts improving and attempts to outperform the DE Random agent. Although, there are many fluctuations and it never stabilises, the RL agent performs better than the DE Random agent. Generally, since reward winning has been dropped and played with agent which has got fixed strategy, the RL agent's performance could not stabilise and converge which impacted its behaviour.

II Number of moves

The above graph (Figure 6.24), shows that the RL agent can improve by minimising number of moves. It started at around 52 moves then decreases gradually as more games are played to around 15 moves. That means that the RL agent learns how to finish the game with minimum number of moves over time which can be inferred that it plays decisive moves as more games are played.

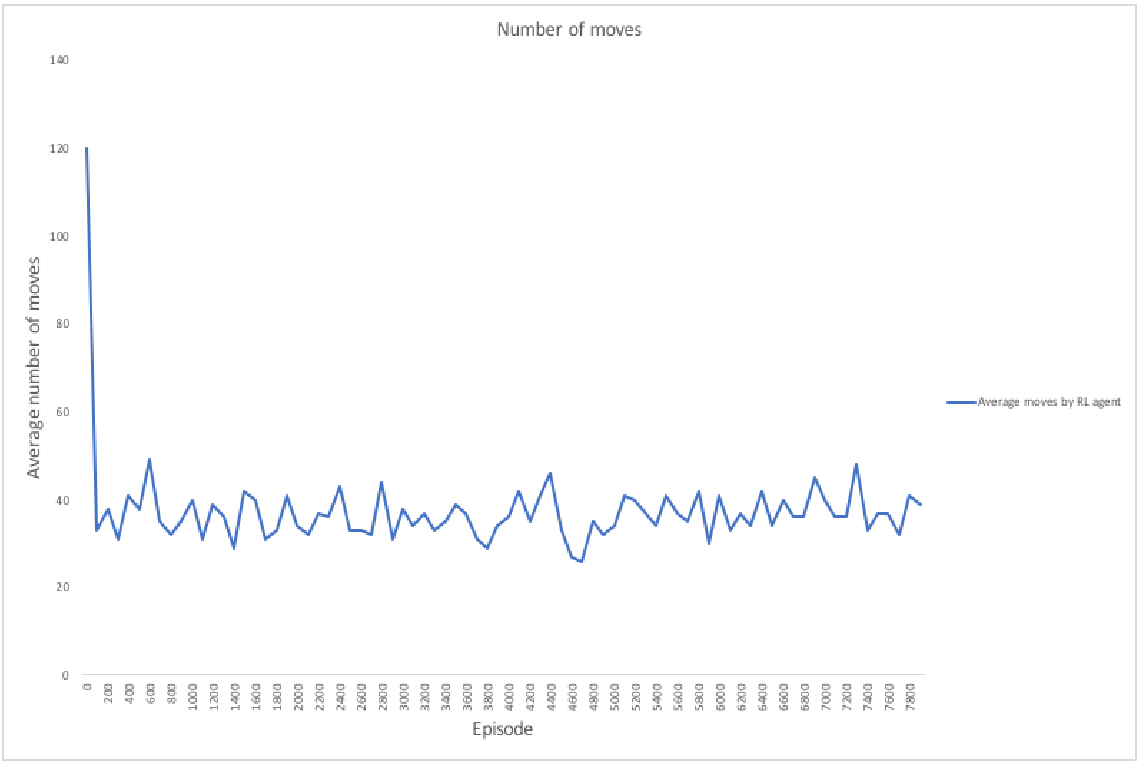


Figure 6.20: RL agent number of moves

III Incoherence

In Figure 6.25, the RL agent shows decreasing value of incoherence than the DE Random agent. Since, contradiction factor was incorporated in the reward function Equation 6.4, the RL agent received more rewards by minimising contradiction. Hence, RL agent is able to learn to minimise incoherence.

IV Irrelevance

Figure 6.26 demonstrates that the RL agent has a lower value of irrelevance than the DE Random agent. This means that the DE Random agent changes focus often than the RL agent and RL agent stayed focused most of the time. Hence, the RL agent is learning how to focus appropriately.

For this reward function, the winning factor of Equation 6.4 has not been considered. In general, it seems that the RL agent performs better in minimising number of moves, incoherence and relevance respectively than the DE Random agent. This is because the RL agent tries to maximise the cumulative reward function with

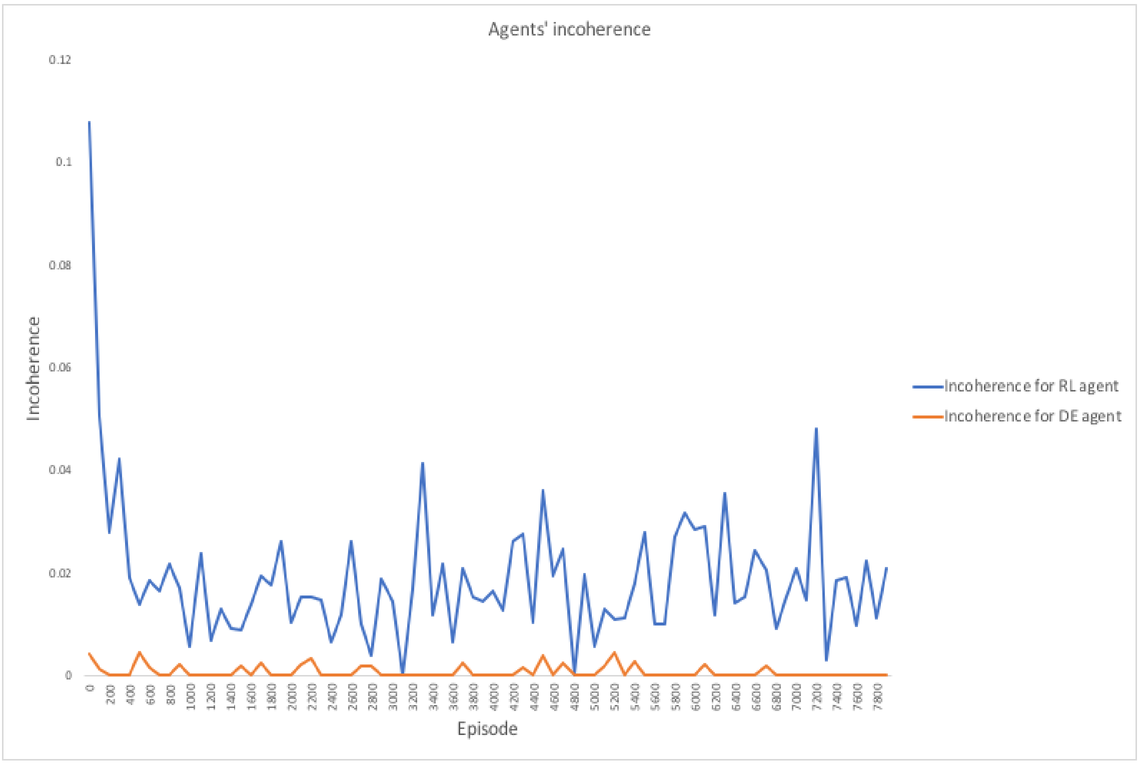


Figure 6.21: Agents' incoherence

regard to these three factors. However, with regards to the performance, the RL agent tries to outperform the DE Random agent with some fluctuations, but never stabilises and converges since this reward function had ignored the winning part. This had an impact on the RL agent's behaviour by not receiving higher reward when it was winning. On the other hand, the opponent is not with deterministic strategy, therefore the RL agent's response and behaviour are impacted. However, the RL agent was able to decrease irrelevance and the number of moves. This is because the RL agent tries to maximise the cumulative reward function and it is not rewarding the agent much on winning.

4 .2.2 Dropping moves

In this case, the RL agent will use Equation 6.5 which is dropped moves count term in the original reward function 6.3

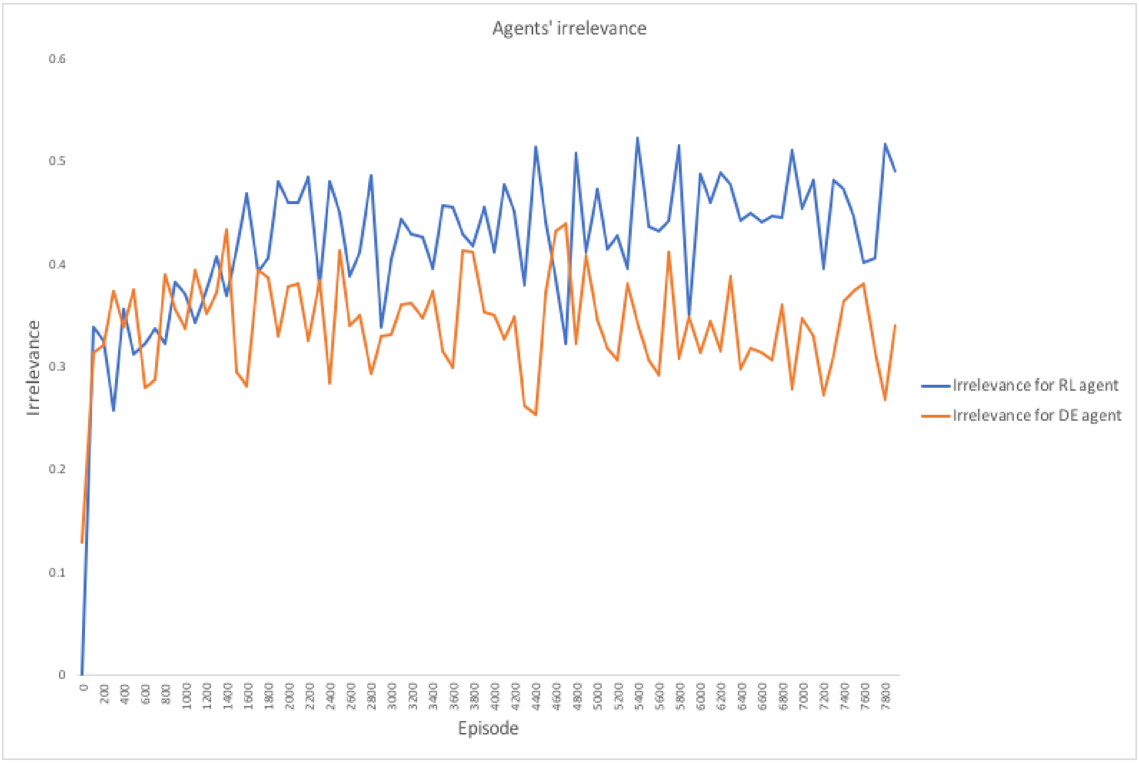


Figure 6.22: Agents' irrelevance

I **Performance**

This reward function Equation 6.5 is not considered number of moves. From Figure 6.27, despite fluctuations in the initial 1000 episodes, the RL agent performs better than the DE Random agent and stabilises after 5000 episodes. Therefore, the RL agent can be able to learn as it plays and it outperforms the DE Random agent. This happens because the RL agent is given positive reward when it wins and tries to maximise the cumulative reward and hence wins more games as it plays.

II **Number of moves**

Figure 6.28, shows that the number of moves is less initially but increases around 25 moves, after which the performance fluctuates frequently as the RL agent plays more games. Although, the RL agent attempts to minimise the number of moves, it increases in the middle because the number of moves in the reward function has been removed and therefore not considered by RL agent while playing the game. Hence, the reward related to the number of

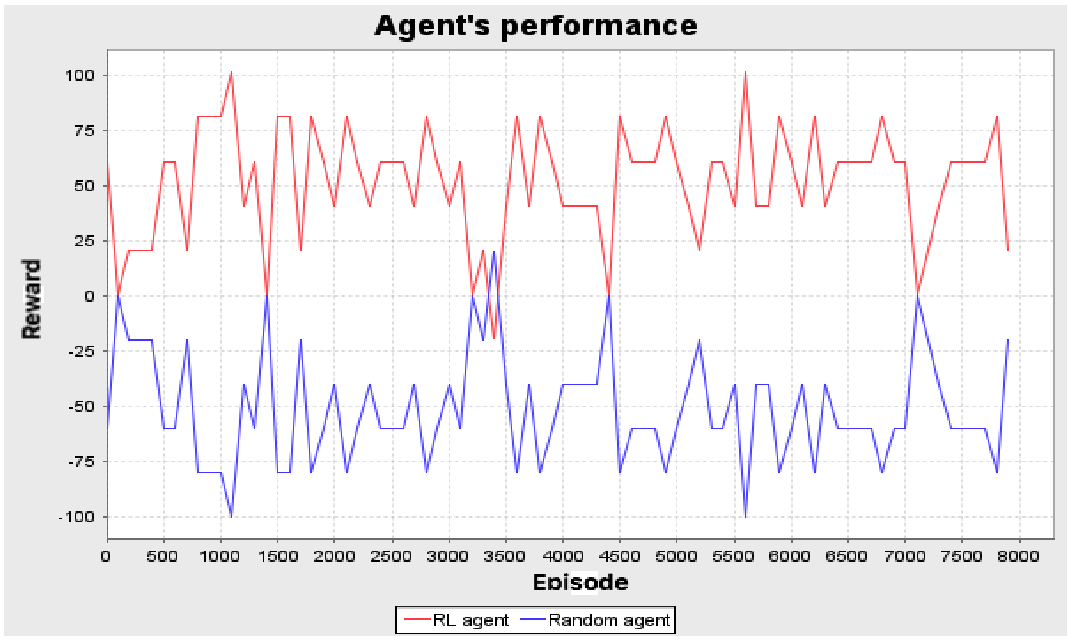


Figure 6.23: Agents' performance

moves is not incorporated explicitly. In fact, the RL Agent tries to maximise the cumulative reward, but in the reward function it is not given enough reward when the number of moves are decreased because the number of move factor is not present and hence the RL agent is not able to decrease the number of moves efficiently.

III Incoherence

From Figure 6.29, it can be seen that the RL agent has a lower value of incoherence than the DE Random agent. Since the RL agent had considered the contradiction factor in the reward function in Equation 6.5, this allows for exploring with minimum incoherence by maximising the cumulative reward. Hence the RL agent is able to improve its learning to minimise incoherence.

IV Irrelevance

Figure 6.30 illustrates that the RL agent had lower value of irrelevance than the DE Random agent. This means that the DE Random agent changed focus

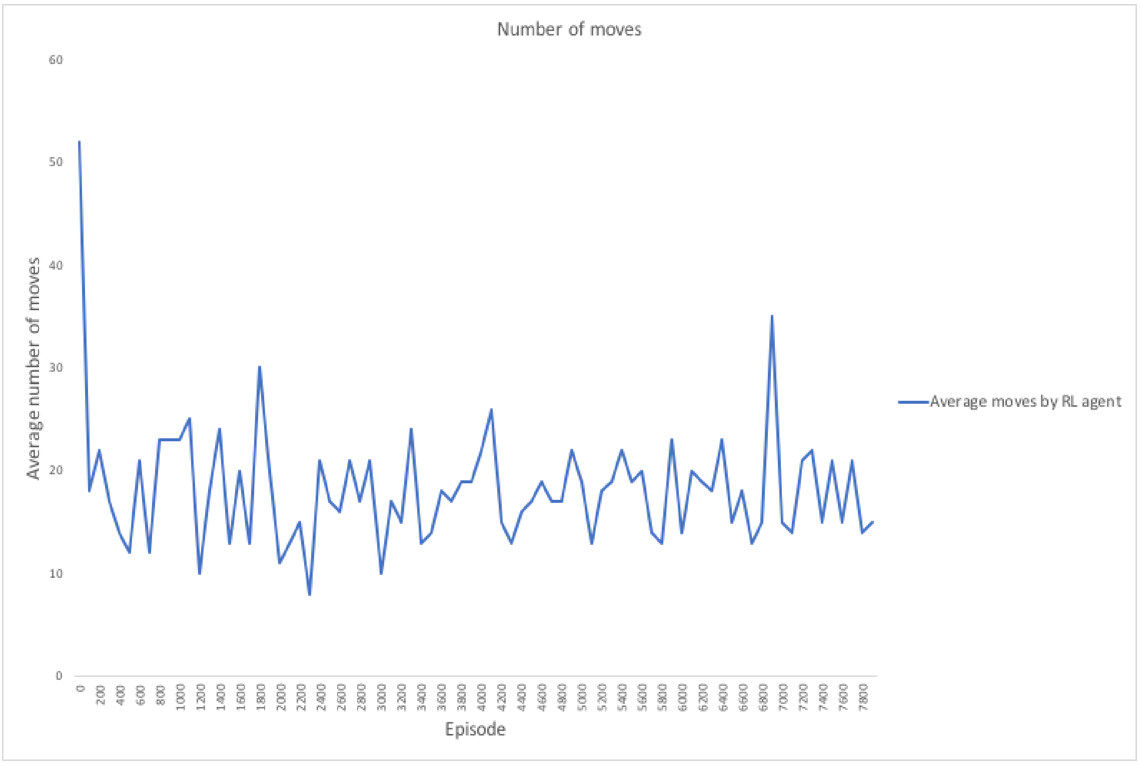


Figure 6.24: RL agent number of moves

to win the game, whereas, the RL agent stayed focused. Since the value of irrelevance of the RL agent decreases with more games being played, the RL agent is able to learn how to focus appropriately and minimise irrelevance.

Therefore, in the reward function given in Equation 6.5 the number of moves factor is dropped. With respect to the agent's performance, it can be concluded that the RL agent is able to learn to argue after 5000 episodes. This is because the RL agent tries to maximise the cumulative reward function and therefore the RL agent was rewarded enough on winning. However, minimising the number of moves did not lead to decrease in the number of moves as the number of moves factor is not considered. Hence, the number of moves related reward is not incorporated in the equation explicitly and it is not given a reward when the number of moves are decreased (because the number of move factor is not present). Therefore, in this case, the RL agent could not decrease the number of moves efficiently. Since this reward function considered contradiction and switching focus factors, the incoherence and irrelevance had decreased and hence the RL agent is able to learn to decrease both incoherence and irrelevance.

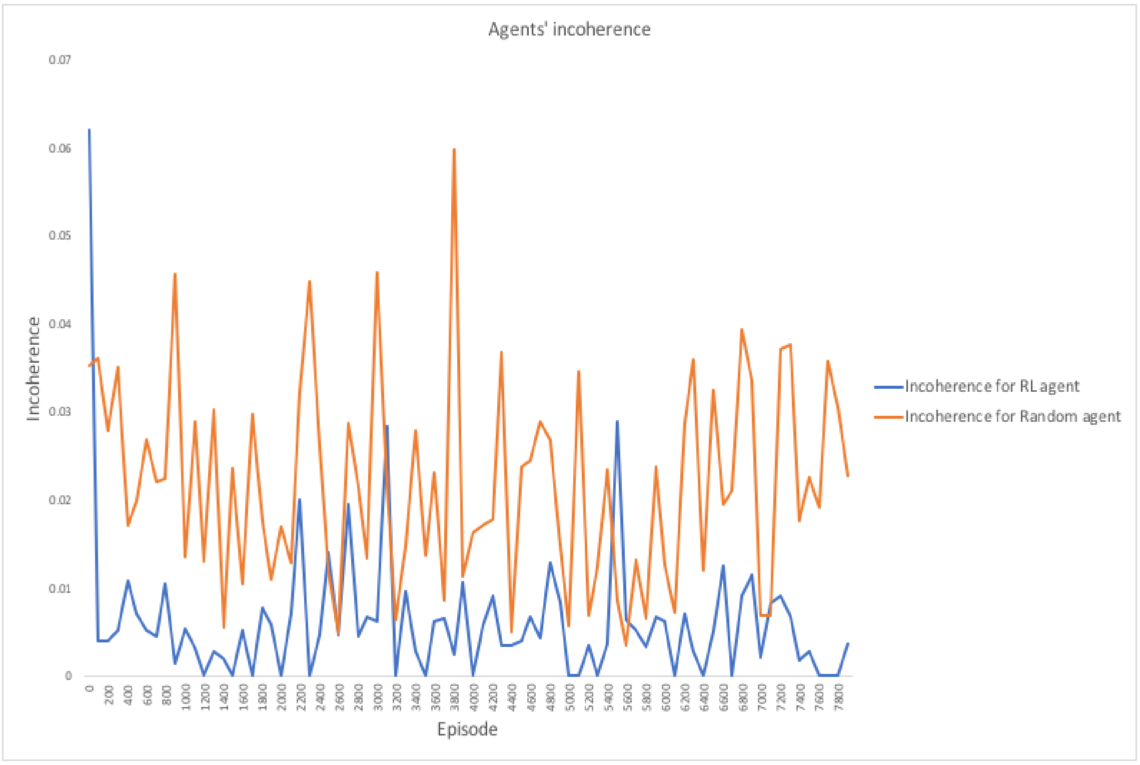


Figure 6.25: Agents' incoherence

4 .2.3 Dropping contradictions

In this section, the RL agent will use Equation 6.6 in which the contradictions term is dropped from the original reward function 6.3.

I Performance

In Figure 6.31, the RL agent is found to perform better than the DE Random agent and stabilise after 5500 episodes. The RL agent was found to attempt to learn and outperform better even in the initial episodes. Hence, the RL agent is able to learn as it plays more games and outperform the DE Random agent. The factor of +100 is present in the reward function (Equation 6.6) therefore it receives higher reward on winning a game. Because the RL agent tries to maximise the cumulative reward, it is able to learn how to win more games.

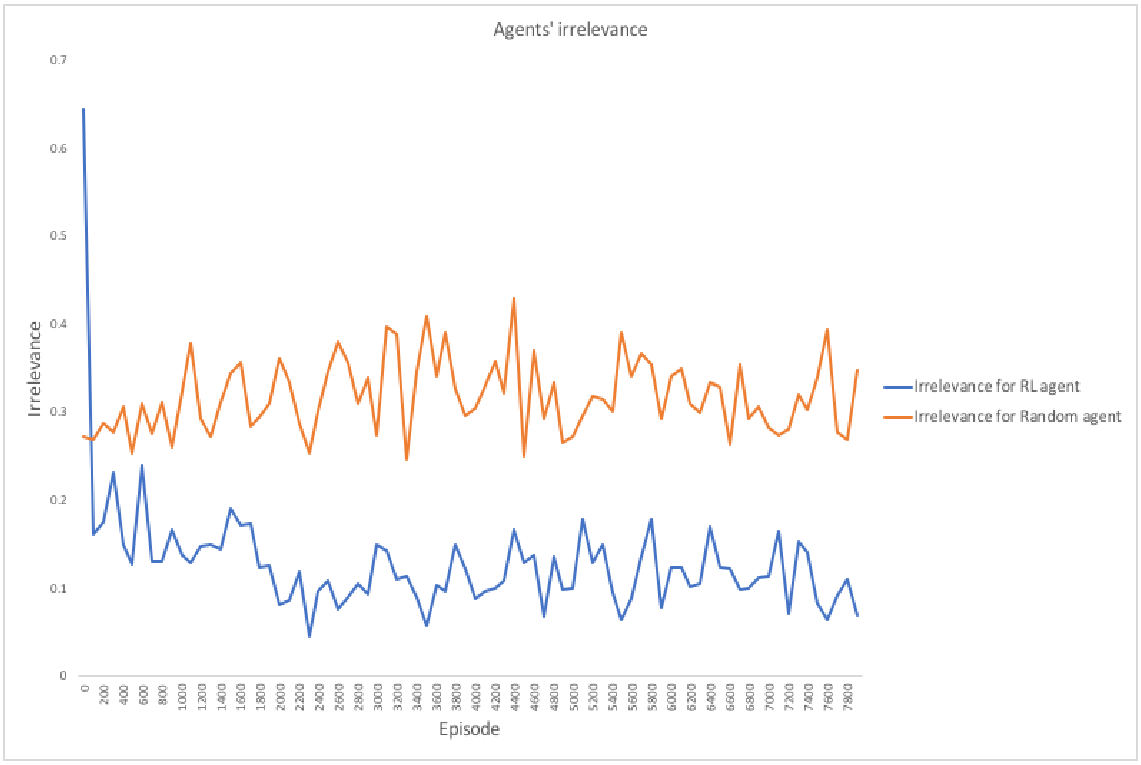


Figure 6.26: Agents' irrelevance

II Number of moves

As seen in Figure 6.32, as the RL agent played more games the number of moves decreased from 47 to 14. This means that the RL agent learns how to finish with less number of moves. Hence, the RL agent is able to learn quickly to win with less number of moves.

III Incoherence

In Figure 6.33, the RL agent had a higher value of incoherence than DE Random agent. This was expected, because the reward function 6.6 did not consider the contradiction factor. This means that, the RL agent does not take into account contradiction, therefore the incoherence is higher. As a result, the RL agent does not learn how to minimise the incoherence. The RL agent tried to maximise the cumulative reward but because the factor of minimising the incoherence was absent in the reward function, it was not able to optimise it.

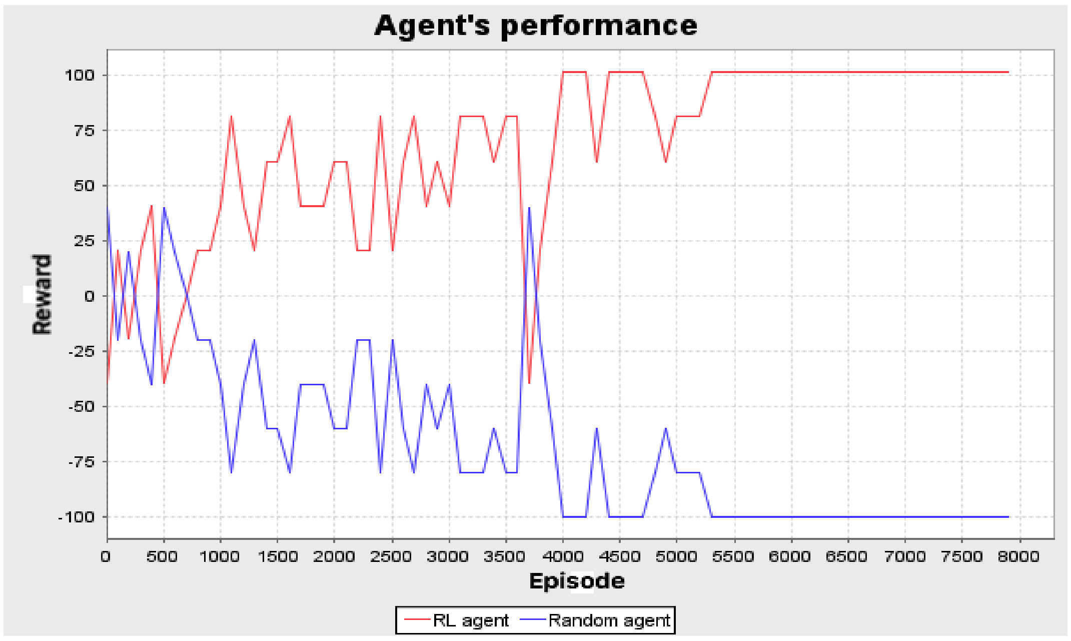


Figure 6.27: Agents' performance

IV **Irrelevance**

The RL agent in Figure 6.34 had a lower irrelevance than the DE Random agent. This means that, the DE Random agent changed focus often than the RL agent. In addition, the value of irrelevance of RL agent decreases as more games are played. The RL agent tried to minimise the irrelevance in order to maximise the cumulative reward. Therefore, this suggests that the RL agent is learning how to focus appropriately.

Regarding the agents' performance, the RL agent initially performs worse, but then, attempts to outperform and improve performance. The factor of contradiction is dropped in this case. As a result the final incoherence is higher than when this factor is present in other reward functions. However, the factors of switching focus and number of moves are present in this reward function. Hence, the number of moves and irrelevance are successfully decreased over the time.

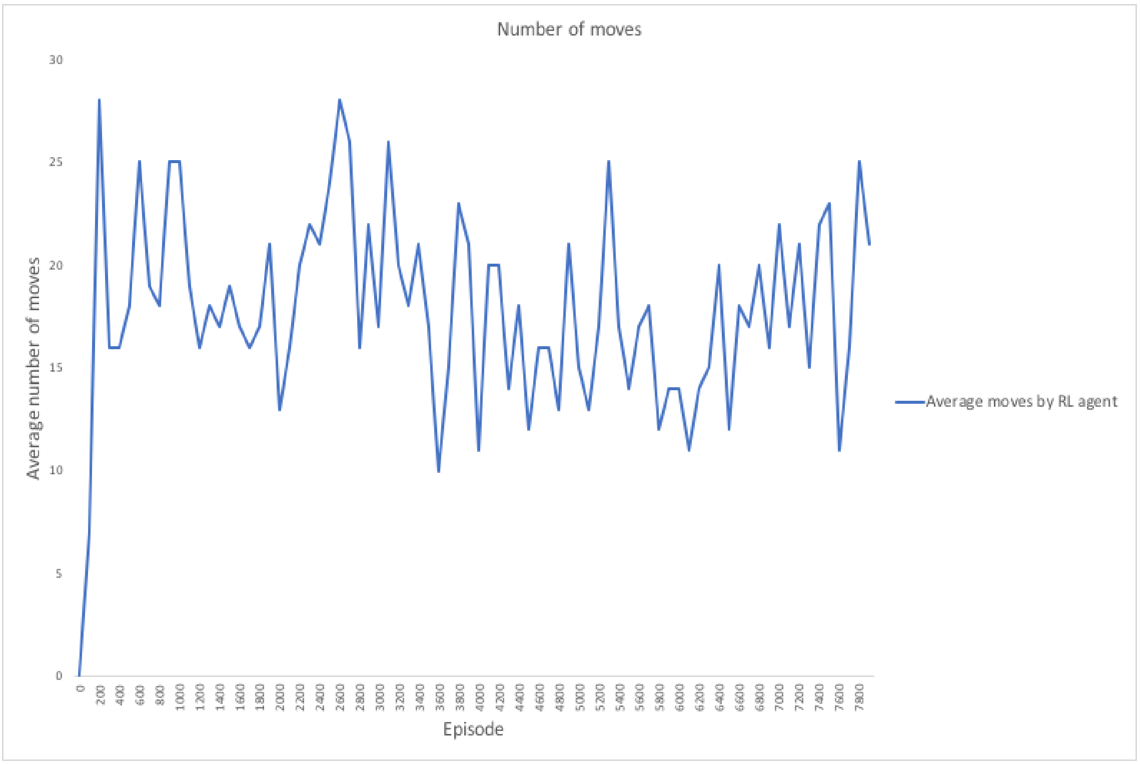


Figure 6.28: RL agent number of moves

4 .2.4 Dropping switching focus

In this section, the RL agent will use Equation 6.7 in which the switching focus is dropped from the original reward function 6.3.

I Performance

As shown in the Figure 6.35, the RL agent outperforms the DE Random agent and stabilises after 5000 episodes. So, the RL agent was able to learn as it played more games and hence outperform the DE Random agent. The RL agent received high reward by considering the winning attribute (+100) to maximise cumulative reward and hence it learns how to win more games as it played.

II Number of moves

In Figure 6.36, a lot of fluctuations can be seen and the number of moves played does not stabilise. This means that, the agent could not learn how to finish the game with less moves. The RL agent takes many moves to win a game. This

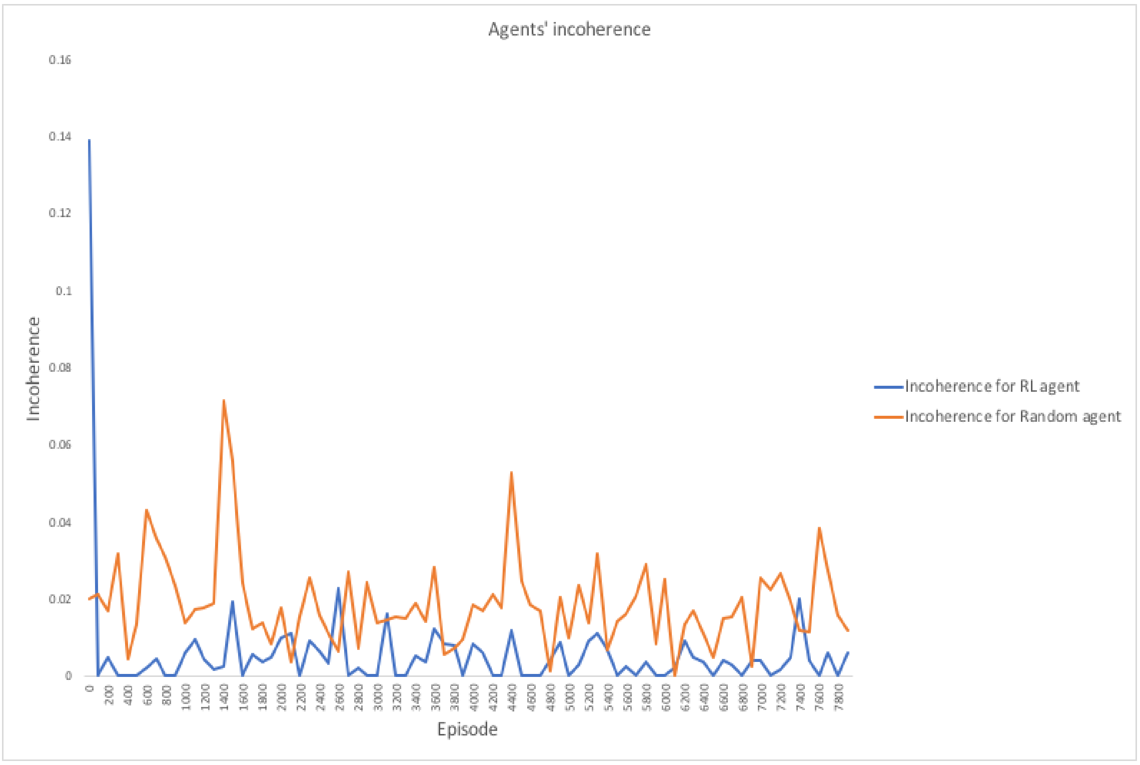


Figure 6.29: Agents' incoherence

is because the switching focus factor was dropped and hence it does not take that into account. The RL agent did not consider to minimise switching focus, therefore it loses focus many times which led to increased number of moves.

III Incoherence

As seen in Figure 6.37, the RL agent could improve incoherence as it is decreasing the incoherence value over time. Since the RL agent had considered the contradiction factor in the reward function in Equation 6.7, this allowed for exploring with minimum incoherence by maximising cumulative reward. The RL agent initially had higher incoherence than the DE Random agent but as the RL agent played more games, and made improvement, it was able to learn how to decrease incoherence.

IV Irrelevance

As seen in Figure 6.38, the RL agent had a similar value of irrelevance as that of the DE Random agent. It means both agents changed focus often for winning

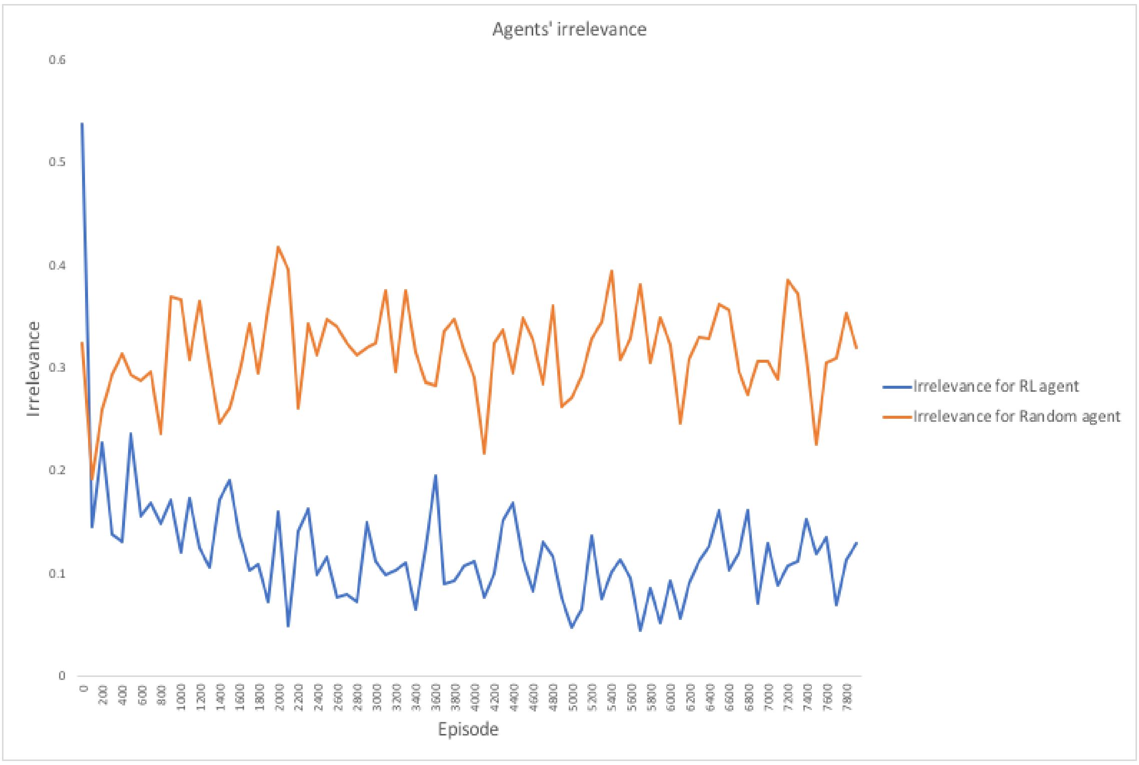


Figure 6.30: Agents' irrelevance

the game. The RL agent could not decrease irrelevance value over time. This indicates that the RL agent could not learn to minimise irrelevance. Since, switching focus factor is factored into the reward function in Equation 6.7, the RL agent aimed to maximise the cumulative reward, but did not minimise switching focus, which caused the irrelevance value to increase over time.

In this reward function Equation 6.7, the RL agent could not decrease the value of irrelevance. This is expected because, the switching focus factor has been dropped. As a result, the RL agent loses focus many times and the number of moves do not minimise over time. Since the factor of irrelevance is absent with other factors being present, the RL agent increases the cumulative reward by optimising the considered factors. As a result, the RL agent outperforms the DE Random agent and thereby improves performance. Also, the RL agent was able to decrease the incoherence by minimising contradictions.

The reward functions described in this chapter aim to make systematic analysis of the RL agent's performance against DE Random agent by dropping one attribute at

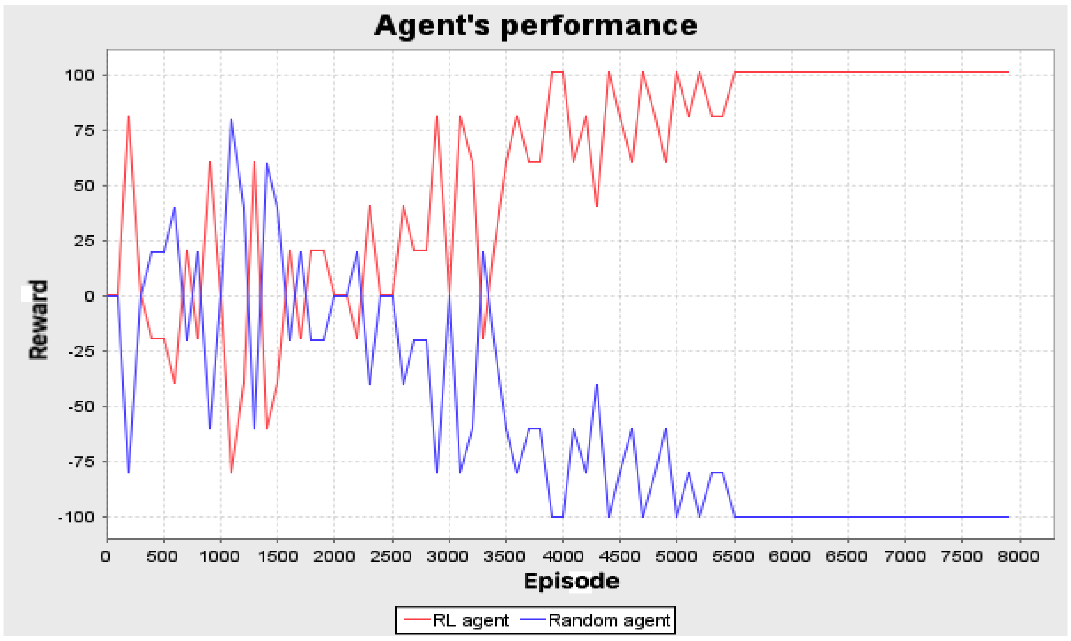


Figure 6.31: Agents' performance

a time. The original reward function Equation 6.3 takes into account all the factors: winning, moves count, contradictions and switching focus respectively. In the original reward function, the RL agent aims to improve performance as well as minimise moves, incoherence and irrelevance which was addressed earlier in this chapter. In this the RL agent played against DE Random agent. The RL agent did not learn how to win the games efficiently if the winning factor (+100) was dropped. Similarly, removing the incoherence factor led to higher incoherence. When the number of moves factor was dropped, the RL agent was able to minimise number of moves. And, removing the irrelevance factor resulted in higher irrelevance.

Therefore, the original reward function seems to perform better compared to these alternative functions as it can trade-off with all attributes and yield more robust and efficient results. The RL agent aims to maximise the cumulative reward, but in these alternative reward function, if one attribute is dropped it will impact on the behaviour of the RL agent and therefore impact on the agent's performance with respect to the dropped attribute. Therefore, it can be concluded that the original reward function Equation 6.3 performs better than alternative reward functions.

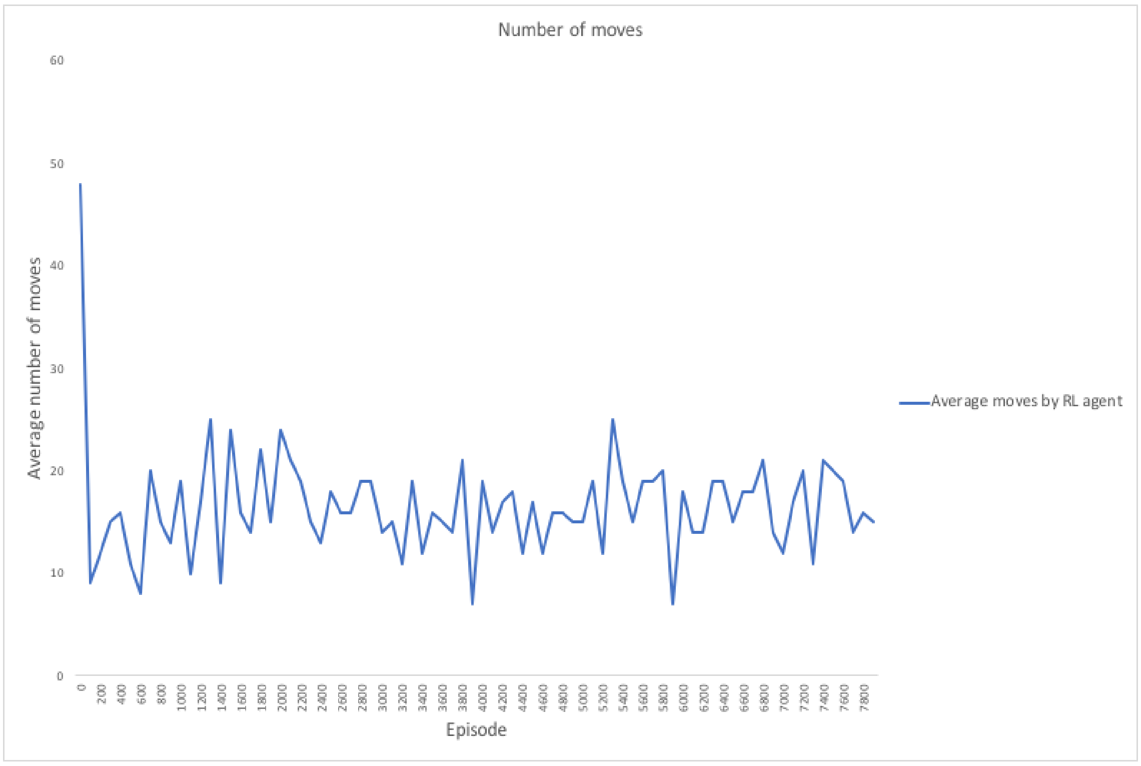


Figure 6.32: RL agent number of moves

5 Summary

To sum up, this chapter proposed quality measures for dialogue between the RL agent and DE baseline agents, i.e. fluency and coherence. The two measurements were incorporated into the reward function and a number of experiments were discussed. From the results, it can be concluded that the RL agent was able to learn to improve its performance, coherence and fluency compared to the DE baseline agents. The DE agent performed better than the RL agent in terms of coherence. This is because the DE agent is deterministic and built with heuristic strategy which makes it contradict less with itself and therefore stay coherent. Hence, it might take a long time for the RL agent to adapt the strategy to be better than or similar to the DE agent.

Alternative reward functions Equation (6.4, 6.5, 6.6 and 6.7) were studied systematically. In each equation, one factor was dropped at a time. The RL agent tries to maximise the cumulative reward but when the factor is dropped, the behaviour of the RL agent is negatively impacted. The reason for this is, when all attributes

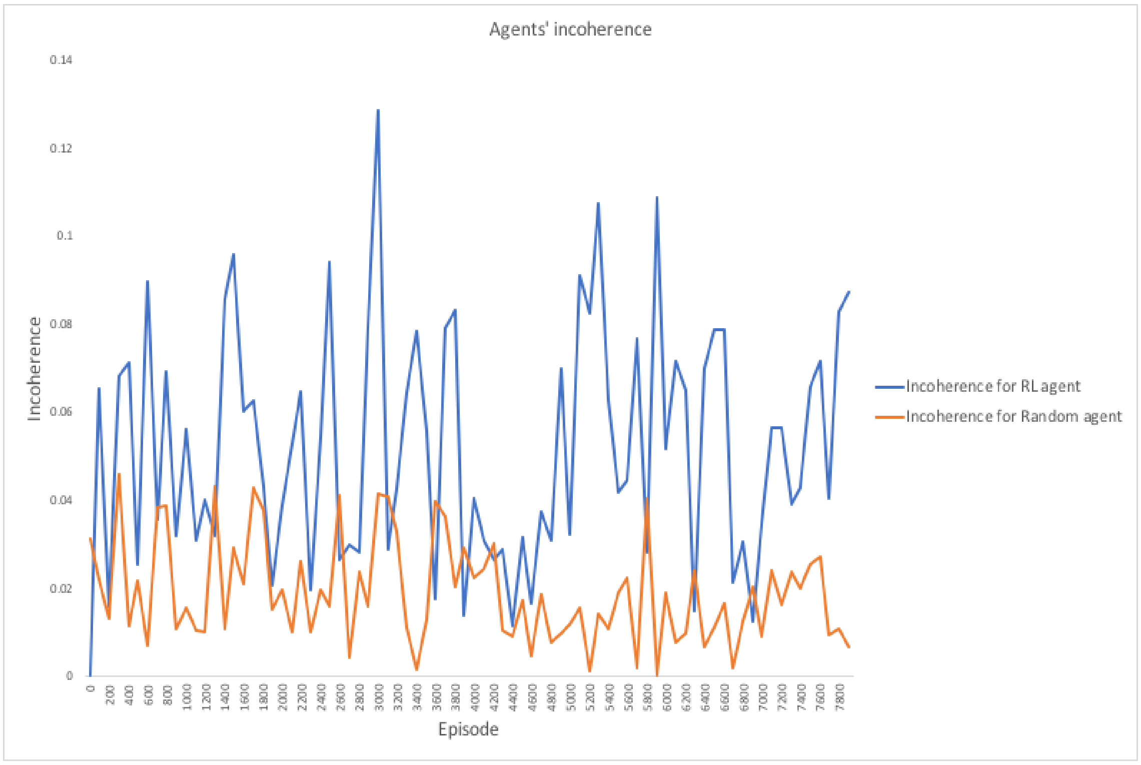


Figure 6.33: Agents' incoherence

are engaged into the reward function, the RL agent is able to trade-off between the attributes and maximise cumulative reward through time. Therefore, the original reward function (Equation 6.3) makes the RL agent behave better than other alternative equations.

This research now moves to generalisation using transfer learning techniques [178] which will be discussed in the next chapter.

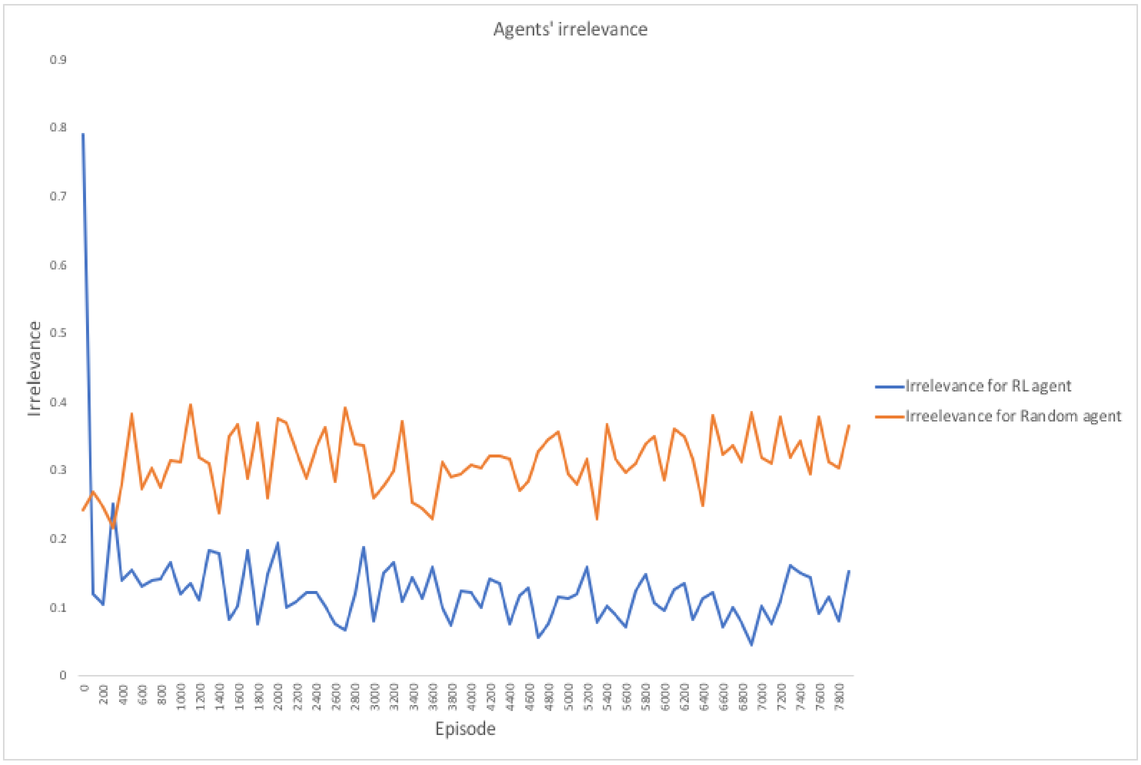


Figure 6.34: Agents' irrelevance

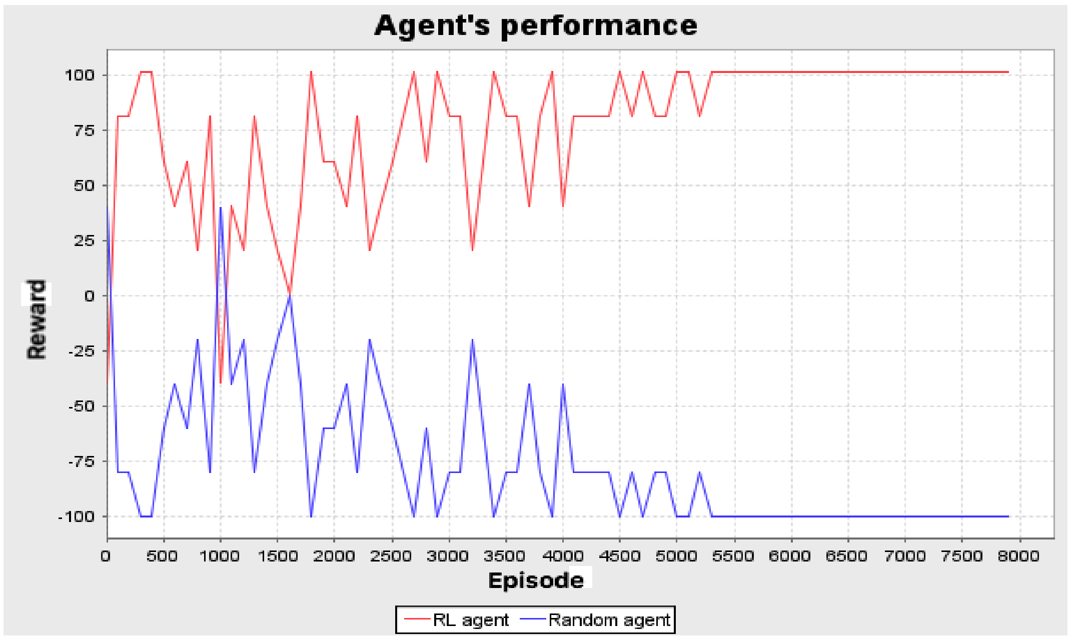


Figure 6.35: Agents' performance

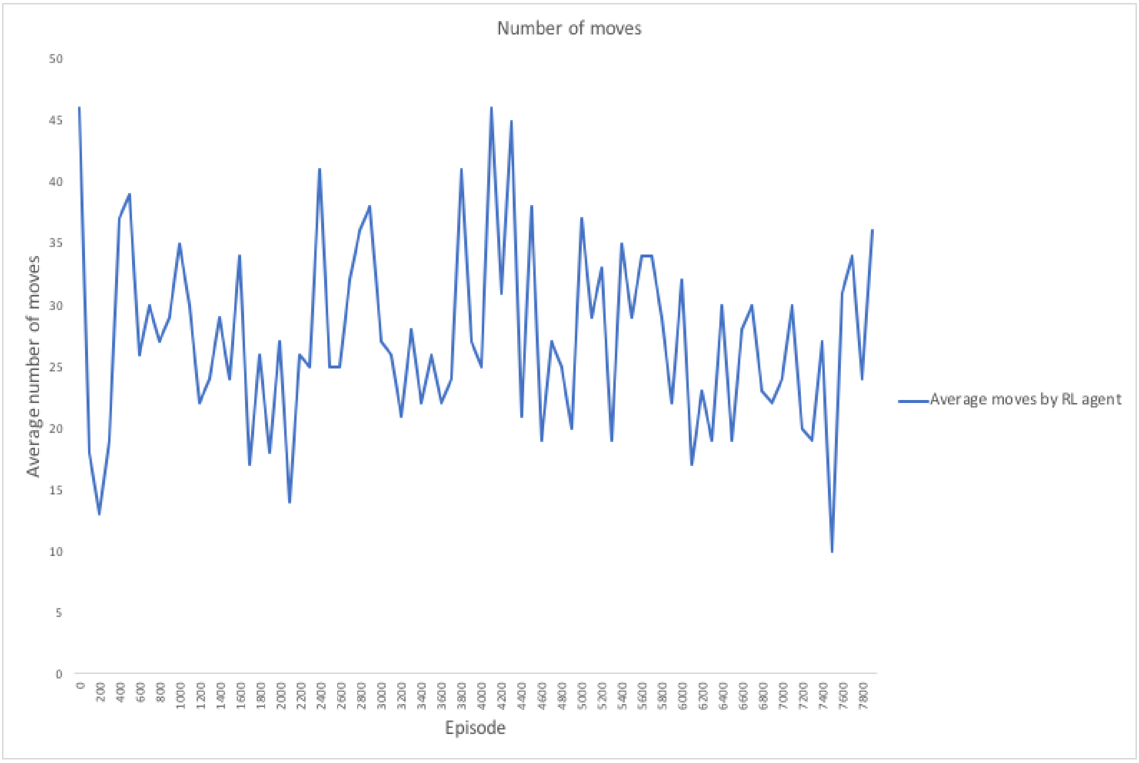


Figure 6.36: RL agent number of moves

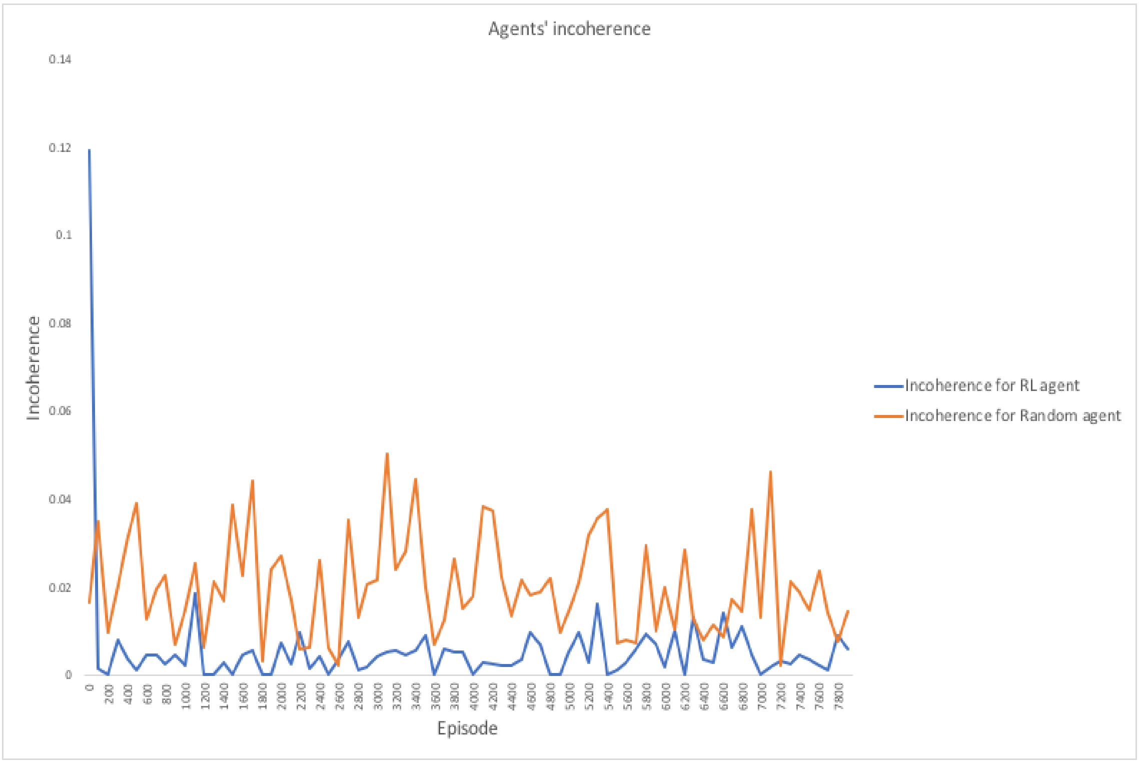


Figure 6.37: Agents' incoherence

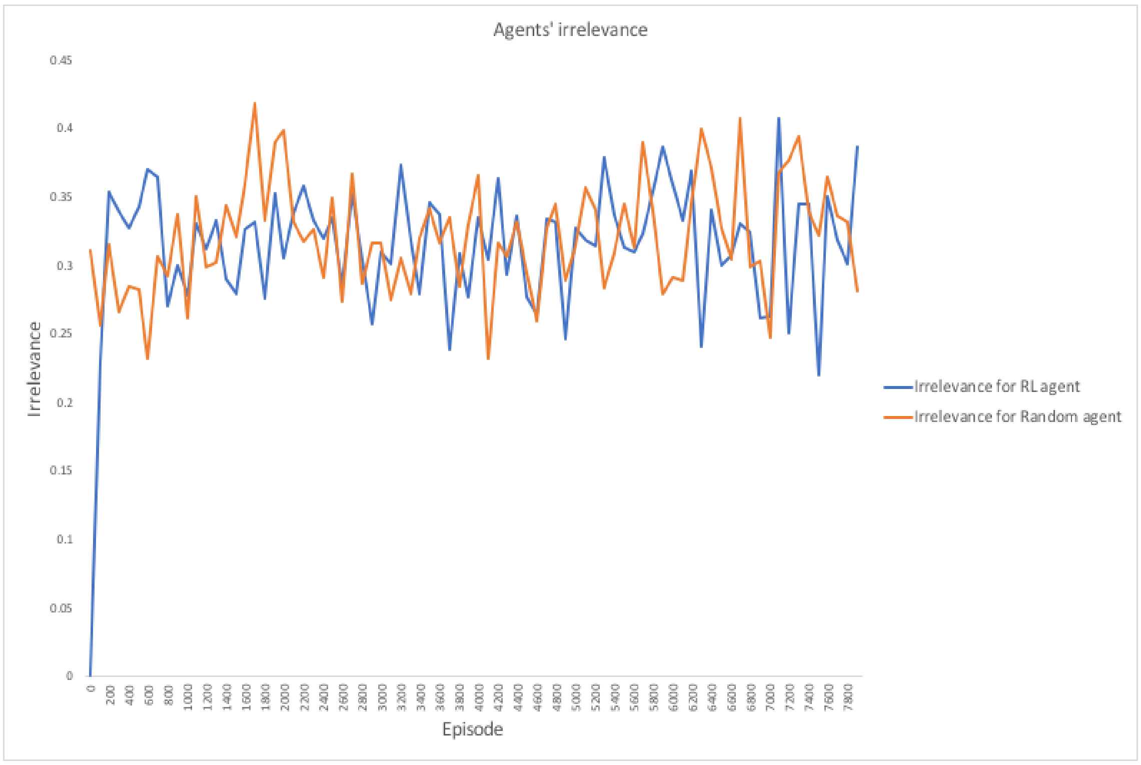


Figure 6.38: Agents' irrelevance

CHAPTER 7

GENERALISATION APPROACH

After allowing the RL agent to learn to argue in a logic based dialogue game against hard-wired DE model agents, it was necessary to generalise the approach to different contexts. This not only saves on the learning time for the agent by not having to learn from scratch each time, but also helps to transfer and reuse the learned experience of one domain to other domains. In this chapter, a generalisation approach to achieve transfer of learning into different tasks is proposed. This uses argumentation schemes and evidence support as patterns that can be recognised by the RL agent in different tasks.

1 Introduction

Generalisation refers to applying previously acquired learning that has occurred in one scenario to other novel scenarios. According to Gluck *et al.* [233], generalisation is the concept where humans or animals can use previous experience from a similar scenario in the current situation. However, generalising any approach requires patterns. Bocconi *et al.* [234] supports generalisation as, being associated with the presence of specific patterns and similarities that can be used to exploit features. Therefore, a new problem scenario will be easier to solve based on previous experience. Hence, generalisation is related to transferring experience between different domains.

In computer science, generalisation refers to a method in which a general approach to solving a class of problems is sought [235]. Some benefits of generalisation through transferable experience are lower learning times and improved performance [178, 236]. This leads to the agent being able to adapt to a new task easily with less experience.

In machine learning, the motivation for transfer learning is based on the need for long-term learning techniques that are aimed at retaining and reusing previously acquired knowledge [237]. Since the RL agent has been able to successfully learn to argue against baseline agents in a specific domain i.e. *capital punishment* using the DE model [7], it is beneficial to generalise the experience that the RL agent has gained and adapt it in a new domain. In the current context, this domain is chosen as *BREXIT*. *BREXIT* covers the United Kingdom's decision to withdraw from the European Union based on the results of a referendum in 2016. *BREXIT* was chosen due to the rich set of arguments that are for and against the decision to withdraw.

This chapter will use the transfer learning approach introduced in Chapter 2 to test whether the RL agent can generalise the policy that it has learnt in one domain to apply to another. Additionally, the kind of patterns that can be learned and transferred to improve performance and speed up learning will be determined.

2 Different context argument domain

A number of methods for representing knowledge in argumentation systems are available in the literature [7]. For instance, Yuan [7], Ravenscroft and Pilkingtons [135], and Bench-Capon [132]. It has been suggested that a knowledge base should enable to provide statements that can be used to answer questions, support other statements and rebut other statements [43]. Yuan [7] asserts that, building the knowledge base in a DE system will enable the agent to provide statements based on answering questions that have been asked and argue with the opponent.

The DE system in [7] adopted a modified version of Toulmin's schema as illustrated in Figure 5.1 of Chapter 5. The *capital punishment* knowledge base was modified by adding argumentation schemes and supporting evidence which are patterns that are assumed to be recognised by the RL agent as knowledge for learning transfer.

Along these lines, a new knowledge base with the same structure as the original *capital punishment* domain introduced in Chapter 5 was built. For this, the highest trending and argument rich topic, that is *BREXIT* was chosen. It refers to the UK withdrawing from European Union (EU) membership. The main reason for selecting *BREXIT* was the availability of arguments that originated from heated debates over whether the UK should leave or remain in the EU. A knowledge base was developed based on the DE dialogue model described previously in Chapter 5.

The *BREXIT* knowledge base focused on two main claims "For *BREXIT*" and "Against *BREXIT*", with some selected topics such as the NHS, taxes and employment. Additionally, the arguments in the dataset were selected from the most debated issues around *BREXIT* from September 2016 to March 2019. The problems of trade with Europe, employment, the NHS and controlling the borders were significant areas of debate and discussion within the British parliament at that time. The importance of these issues could be seen from the perceived future effect of *BREXIT* on each one. There are other important issues, such as education and the impact of *BREXIT* on the cost of living, but these are more concerned with long-term consequences than short-term ones. Figures 7.1 and 7.2 show that financial involvement in the EU, as well as percentage of British jobs represent two arguments from the standpoint of attitude. In addition, control of national borders and spending on the NHS represent arguments based on the consequences.

As *BREXIT* is an ongoing topic of debate, there was some difficulty in finding primary sources for collecting information. Most information was collected from secondary sources, such as newspapers, TV documentaries and journals. As such, a decision was made to select information from the following outlined sources, in order to present a wide range of views on *BREXIT*, and these sources can be found in the following:

1. Reflecting the official link between the UK parliament and the public, presented via e.g. BBC news from October 2016 to March 2019.

2. British newspapers, such as, The Guardian, The Telegraph, The Sun and The Daily Mail.

3. Articles on *BREXIT* published from 2016 to 2019 which were discussed and reviewed critically.

4. NHS policy: The UK's NHS plays an important role in the politics of the UK. Although the public is largely supportive of the service, The King's Fund [238] stated in 2017, that public satisfaction with NHS services was at its lowest

point for ten years. Therefore, any issues with *BREXIT* funding for the NHS rated high on the agenda.

5. BBC news: According to Ofcom [239], the BBC news was rated by participants as highly accurate and reliable. In the annual polling, participants were asked, which news source they trusted most of all those available. Approximately 59% gave the BBC a good rating for trustworthiness, while 54% rated it as the most accurate reporting. However, it was ranked as impartial and unbiased by just 48%.

6. The Guardian: The Guardian newspaper has a peak degree of domain authority for Google and Google news[1], which means it has a high rank of searching for breaking news. The Guardian's editorial code talks strongly about verification and does not claim anything that cannot be confirmed as a fact. According to a 2017 survey conducted by Pew Research Centre [240], the publication has a 50% confidence rating among UK adults, compared with a 22% distrust rating.

7. The Daily Mail: Although the confidence rating of the newspaper is fairly low, as an unreliable source[2], it is still read by adults in the UK [3]. According to Jigsaw Research [241], of the 40% of adults who claim to get news via newspapers, 31% claimed to do so through the Daily Mail. This represented the highest percentage in any publication. (It would be remiss, therefore, to not consider the The Daily Mail as a source, in spite of its lack of perceived credibility, in light of its high level of circulation).

8. Scottish independence question: The question over Scottish independence from the United Kingdom is directly related to the issue of *BREXIT* [4]. In the referendum, Scotland voted closely to remain part of the UK. A number of polls showed close attitudes in Scotland towards leaving the UK post *BREXIT*, for

[1]https://www.theguardian.com/commentisfree/2015/jul/06/how-the-guardian-decides-which-sources-can-be-deemed-trustworthy

[2]https://www.pressgazette.co.uk/facebook-more-trusted-news-daily-star-according-bbc-commissioned-survey/

[3]https://www.statista.com/statistics/380710/daily-mail-the-mail-on-sunday-monthly-reach-by-demographic-uk/

[4]https://www.theguardian.com/uk-news/2020/feb/04/scottish-independence-survey-shows-brexit-has-put-union-at-risk

instance YouGov [5] and Survation [6]. Many in Scotland want to stay in Europe; therefore, this is high on the *BREXIT* agenda with Scottish union and the rest of the UK being potentially at risk.

9. Rejection of no-deal *BREXIT*: Refusal of members of parliament, including many Scottish members, in support of the no deal *BREXIT* is significant because it shows a difference in what people view leaving Europe to mean. The findings of a number of surveys conducted in the second half of 2019 found that nearly 75% of those who voted to leave Europe during the referendum supported a no deal *BREXIT* [7]. Thus, with the 25% who do not support this, it can be supposed that the outcome of the referendum would have been different if this option had been made clear before the voting campaign.

10. Direct and indirect tax implications of leaving Europe, as well as increasing rents: This was a real problem throughout the campaign. According to Deloitte[8], a lot of changes to taxation were not expected, though the UK would need to implement changes in regard to indirect taxation. Even with the possibility that future governments would have more freedom of choice, uncertainty about the direction in which the UK would be taken may be cited as a major factor for those who voted in the referendum. Concerns about rent prices and hence the cost of living were also highly affected [9]. Despite confusion about the matter, many may have seen serious reasons to be concerned by the UK leaving Europe.

11. The Sun: According to the press Gazette[10], in terms of impartiality, the Sun has the lowest rank of news sources.

12. Lack of a viable deal: Post referendum, several rounds of negotiations happened in order to define the relationship between the UK and Europe after *BREXIT*. With any such deal being extremely unlikely to satisfy everyone, the lack of

[5]https://yougov.co.uk/topics/politics/articles-reports/2020/01/30/scottish-independence-yes-leads-remainers-increasi

[6]https://www.survation.com/even-split-between-yes-and-no-in-new-scottish-independence-voting-intention-poll/

[7]https://www.bbc.co.uk/news/uk-politics-49551893

[8]https://www2.deloitte.com/uk/en/pages/tax/articles/uk-leaving-the-eu.html

[9]https://www.leaders.co.uk/advice/brexit-affect-on-rental-market

[10]https://www.pressgazette.co.uk/facebook-more-trusted-news-daily-star-according-bbc-commissioned-survey/

a proposal with a good possibility of a deal in the end proved a problem for *BREXIT* eventually happening [11]. The Prime Minister suggested a *BREXIT* deal around three times, and a lack of resolve to leave with no deal was worth considering.

It took a lot of effort and was very time consuming to collect information from sources. Collecting information from these sources required watching, reading and analysing relevant content, which took significant time to execute.

Structure of the knowledge base: With respect to the practical level, it can be seen in Figures 7.1 and 7.2 there are seven main arguments for "For *BREXIT*" and a further seven supporting arguments "Against *BREXIT*". In the following tables Table 7.1 and 7.2 show arguments and supporting arguments for both claims For and Against.

[11]https://citywire.co.uk/wealth-manager/news/mays-brexit-plan-in-jeopardy-as-mps-deliver-second-historic-defeat/a1209225

Arguments	**Support arguments**
Threat from EU to British sovereignty (and lack of EU accountability).	EU Law overrides UK Law.
More jobs for British people either full or part time.	UK will be for British people and salary will be raised.
Full control of UK borders, waters and commodities.	Only UK terms and regulations will be used with NO exception to EU countries.
Less money spent on UK NHS.	Non-British people with no permission to stay in UK forever will no longer be authorised to use NHS for free.
UK would avoid deeper integration with Eurozone.	Euro has proved to be volatile and staying with pound a good idea.
Threats of possible *BREXIT*(s) are not realistic.	Scottish voices voted "no" to be out of the UK.
Government will control the taxation among companies.	Without auditing prices, living costs will increased evidently.

Table 7.1: For *BREXIT* claim dataset

Arguments	**Support arguments**
Unclear what "sort" of *BREXIT* should be carried out.	Opinions are more varied than just remain or leave - consensus very difficult.
Increase the percentage of taxes on UK taxpayers.	To avoid potential loss, British companies will raise the prices of goods, properties.
Encourage more divisions within UK.	Scottish resistance by asking for another internal *BREXIT*.
Being out of EU will decrease the level of collaboration financially.	Political change lead investors to look for safe and stable place.
More money will be spend on NHS.	This is attributed to the cut of partnership from the EU. Government will be the main resource of funding.
Companies will be not respond immediately to job offers.	Statistically, UK companies intend to shift online commodities to reduce paying salaries (e.g., Waitrose).
British involvement can benefit both EU and Britain - common interests.	UK has very rarely voted against EU policy.

Table 7.2: Against *BREXIT* claim dataset

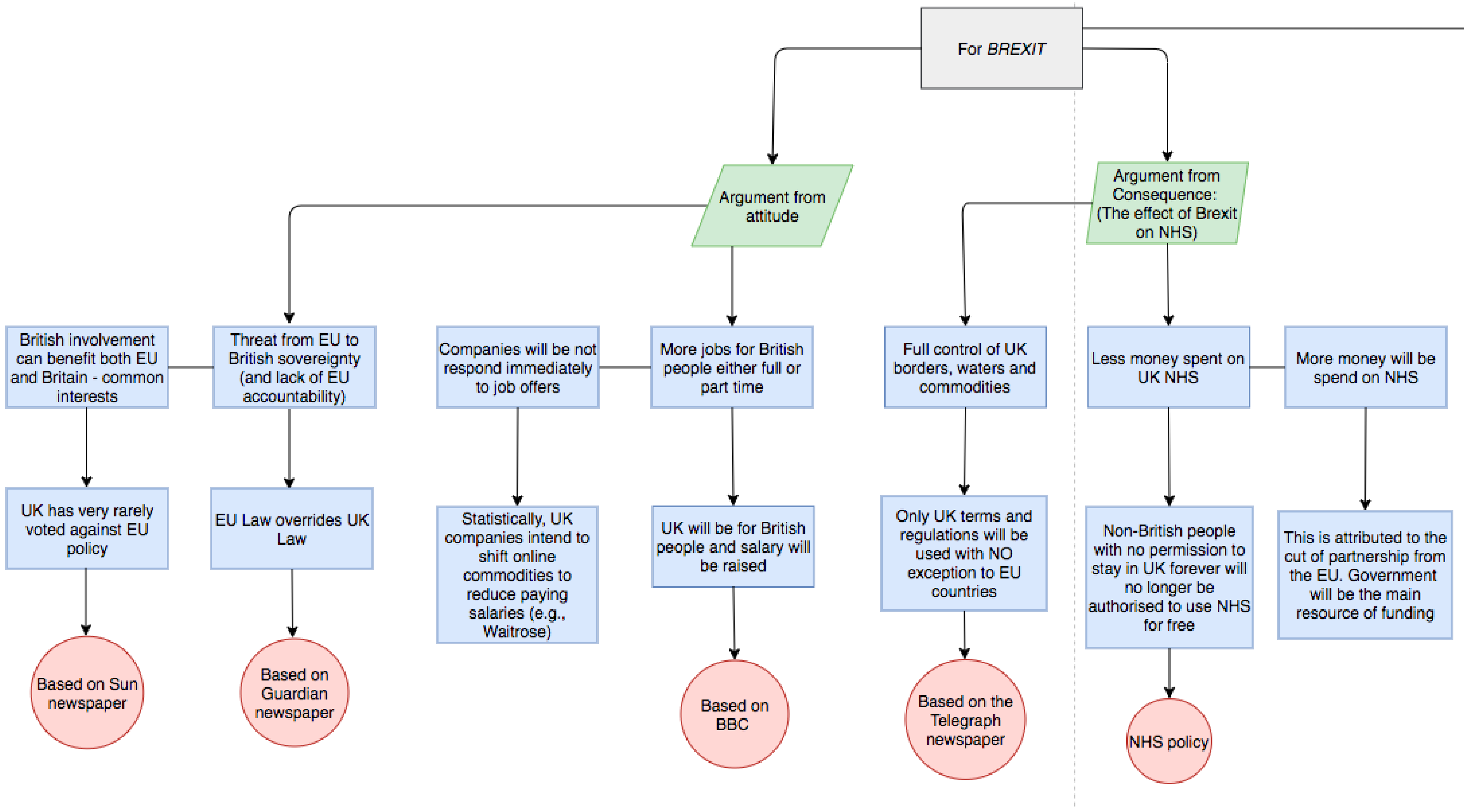

For *BREXIT*
Argument from attitude
Argument from Consequence: (The effect of Brexit on NHS)
British involvement can benefit both EU and Britain - common interests
Threat from EU to British sovereignty (and lack of EU accountability)
Companies will be not respond immediately to job offers
More jobs for British people either full or part time
Full control of UK borders, waters and commodities
Less money spent on UK NHS
More money will be spend on NHS
UK has very rarely voted against EU policy
EU Law overrides UK Law
Statistically, UK companies intend to shift online commodities to reduce paying salaries (e.g., Waitrose)
UK will be for British people and salary will be raised
Only UK terms and regulations will be used with NO exception to EU countries
Non-British people with no permission to stay in UK forever will no longer be authorised to use NHS for free
This is attributed to the cut of partnership from the EU. Government will be the main resource of funding
Based on Sun newspaper
Based on Guardian newspaper
Based on BBC
Based on the Telegraph newspaper
NHS policy

Figure 7.1: For a *BREXIT*

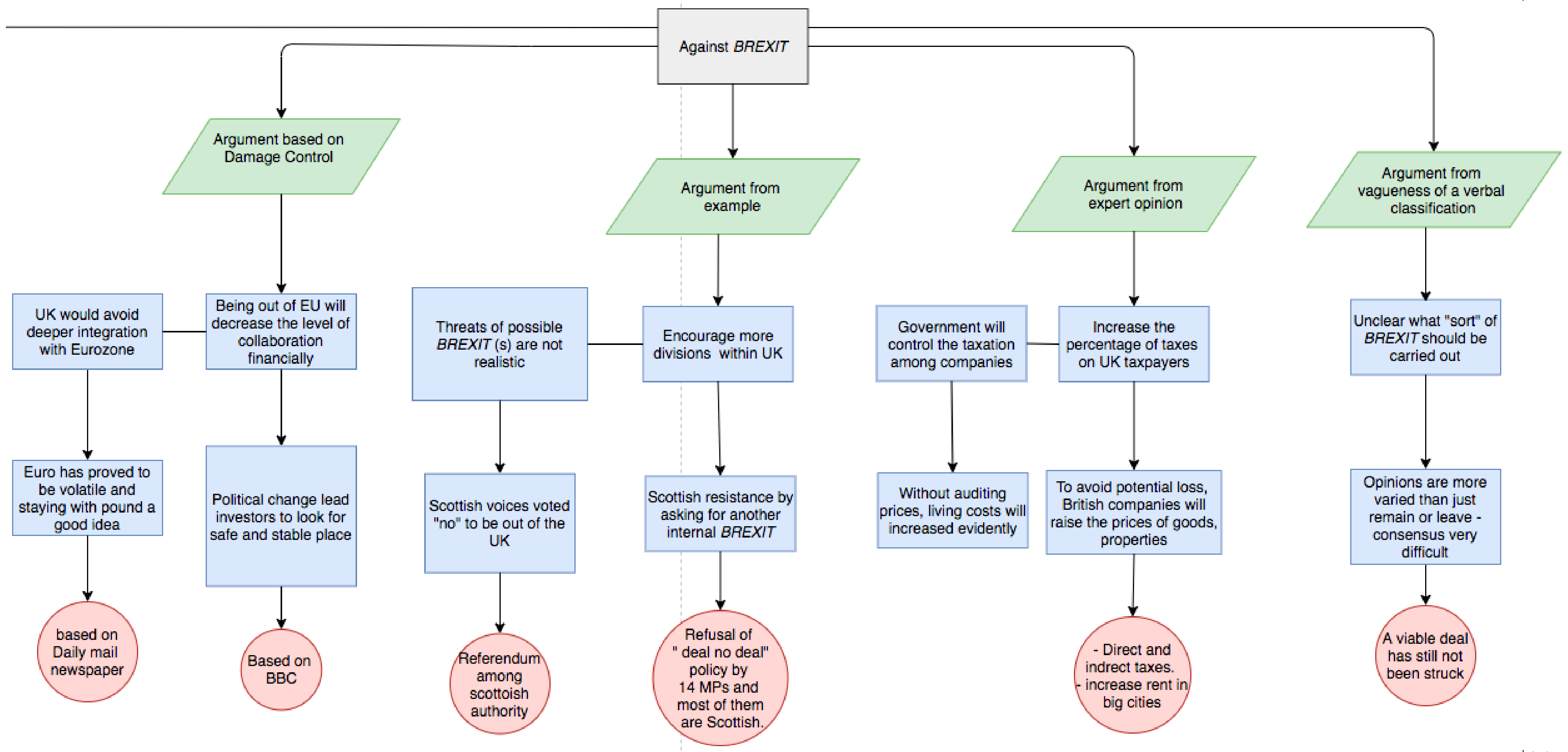

Against *BREXIT*
Argument based on Damage Control
Argument from example
Argument from expert opinion
Argument from vagueness of a verbal classification
UK would avoid deeper integration with Eurozone
Being out of EU will decrease the level of collaboration financially
Threats of possible *BREXIT* (s) are not realistic
Encourage more divisions within UK
Government will control the taxation among companies
Increase the percentage of taxes on UK taxpayers
Unclear what "sort" of *BREXIT* should be carried out
Euro has proved to be volatile and staying with pound a good idea
Political change lead investors to look for safe and stable place
Scottish voices voted "no" to be out of the UK
Scottish resistance by asking for another internal *BREXIT*
Without auditing prices, living costs will increased evidently
To avoid potential loss, British companies will raise the prices of goods, properties
Opinions are more varied than just remain or leave - consensus very difficult
based on Daily mail newspaper
Based on BBC
Referendum among scottoish authority
Refusal of " deal no deal" policy by 14 MPs and most of them are Scottish.
- Direct and indrect taxes. - increase rent in big cities
A viable deal has still not been struck

Figure 7.2: Against *BREXIT*

3 Transferable argument

Since the argumentation schemes and evidence support source patterns were created previously, it can be supposed that these patterns are useful to the RL agent in transferring its experience across different domains. This is due to the fact that these schemes classify arguments and can be shared in different knowledge bases [242]. Various argumentation schemes, such as argument from expert opinion and argument from consequence [221, 242] have been proposed in the literature. These argumentation schemes can be implemented in both the *capital punishment* and *BREXIT* knowledge bases.

Likewise, the sources that are represented in the knowledge base as evidence support for arguments can be applied in different domains. These kinds of sources such as the Guardian newspaper, The Times and BBC news are evaluated based on their reliability. Their reliability scores are established based on how much people generally trust these sources.

Therefore, it is expected that these two patterns can be used by the RL agent to enable transfer learning [176, 178, 184]. These patterns allow mapping between domains and tasks which are the main categories of transfer learning algorithms along with the five dimensions mentioned in [178, p. 1640] (discussed in the next section). Taylor and Stone [178] suggest that, transfer is more likely to be beneficial as the mechanism for selection becomes more related. Taylor *et al.* [243] support the point that transfer data may be reused between different tasks. Having been recorded first in a source task, it can then be transferred into a target task and used for the target task's learner as it builds its model. Lazaric [244] also suggests that, learning algorithms can use the transferred knowledge to solve and improve performance significantly in a target task when the tasks are similar. Hence, these patterns can be useful to transfer between tasks due to their mapping and relation between domains.

4 Method of transfer

The methods that are previously mentioned in Chapter 2 in Section 3 .1 explain how to apply transfer learning between the source task (*capital punishment*) and target task (*BREXIT*). In Table 7.3, the transfer implementation for each method is illustrated.

Method	**Implemented in *capital punishment* and *BREXIT* domains**
Allowed task differences	Each task, even *capital punishment* and *BREXIT* have different states and actions. However, the state shape is still the same $((previousmove \cup CS1 \cup CS2))$ as is the reward function ($R = 100 + \frac{M1}{Mn} + \frac{C1}{Cn} + \frac{SF1}{SFn}$).
Source task selection	First, the RL agent learns how to argue in the source task *capital punishment*. *BREXIT* has been chosen as a target task to test how successful transfer learning can occur.
Transferred knowledge	Since the states and actions will be different between tasks, high level knowledge has been chosen to transfer such as, important features between tasks (patterns) i.e argumentation schemes and evidence support sources.
Task mappings	Both tasks have the same shape of states, actions and reward function. With task mapping, it assumes that transfer learning will decrease the time of learning.
Allowed learners	Both tasks have used the same RL algorithm which is the Q-learning algorithm

Table 7.3: Transfer from *capital punishment* to *BREXIT*

5 Experiment and result

Experiments were carried out to evaluate the performance and efficiency of transfer learning. In order to facilitate the assessment, baseline agents (DE agents) were paired against RL agents. The RL agent first played against one of the baseline agents in the *capital punishment* domain and then played against it again in the *BREXIT* domain.

Transfer learning between the two domains was facilitated using the popular tabular transfer approach. Evidence support sources and argumentation schemes were chosen as patterns and common features between domains. Therefore, two tables were generated to transfer the weights that empowered the RL agent to win the game: one for the evidence support sources and one for the argumentation schemes.

From the source task, tables for both sources and argumentation schemes were generated using the established patterns. These tables were used to help the RL agent in the target task to learn to win the game. Then, weights were calculated by how likely it was for each chosen source and argument scheme to have won, for instance how many times the BBC news source would lead to the RL agent winning the game. In the target task, the agent found statements that are related to the sources and argument schemes which made the RL agent most likely to win the game. Finally, the RL agent chose actions which were related to sources and argument schemes that were more likely to win as they were in the source task. The same reward function as in Equation 6.3 was used:

$$R = \begin{cases} 100 + \frac{M_1}{M_n} + \frac{C_1}{C_n} + \frac{SF_1}{SF_n} & \textit{If RL agent wins} \\ -100 & \textit{Otherwise} \end{cases}$$

The agent was made to play the dialogue game for 4,000 episodes with the same settings with learning rate (α) discounted factor (γ), set to 0.9 and 0.9 respectively in the Q-learning algorithm:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_t,a_{t+1}) - Q(s_t,a_t)]$$

In addition, the immediate reward (r_{t+1}) was set to -0.01 to make the RL agent win faster. The RL agent played against both the DE agent with heuristic strategy or the DE Random agent in the source task. Once the RL agent had learning experience in the *capital punishment* domain the two tables for evidence sources and argumentation schemes were generated. Then, the RL agent repeated playing against baseline agents in the *BREXIT* domain with the same number of episodes

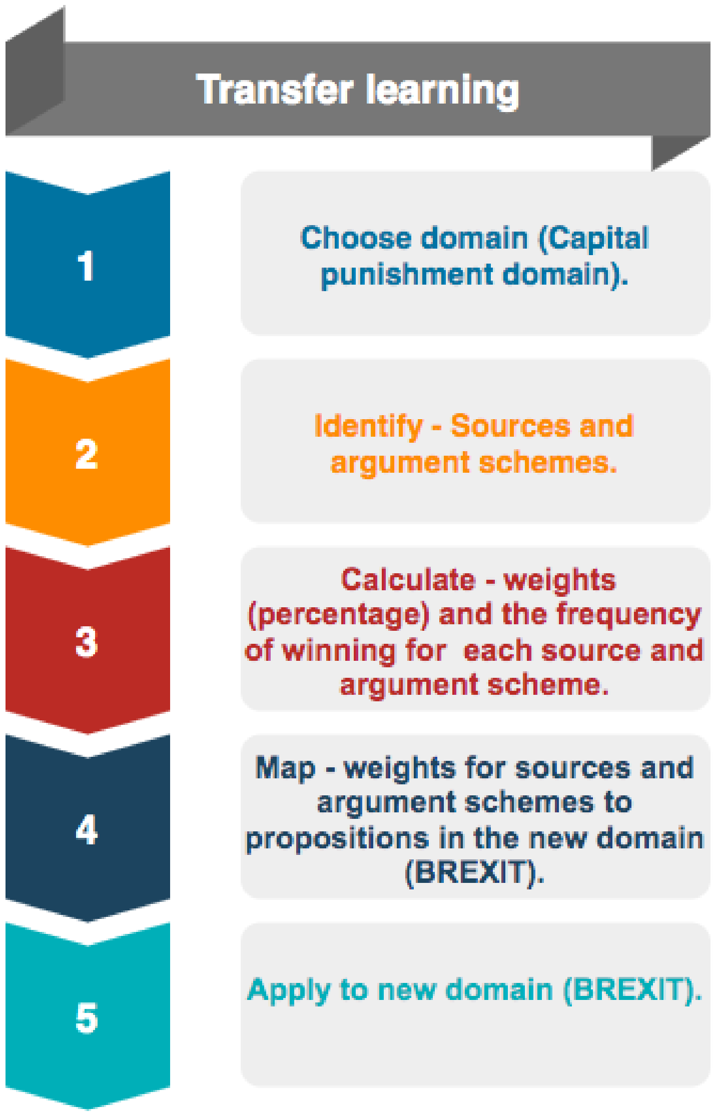

Transfer learning
1
Choose domain (Capital punishment domain).
2
Identify - Sources and argument schemes.
3
Calculate - weights (percentage) and the frequency of winning for each source and argument scheme.
4
Map - weights for sources and argument schemes to propositions in the new domain (BREXIT).
5
Apply to new domain (BREXIT).

Figure 7.3: Transfer learning from source task (*capital punishment*) to target task (*BREXIT*)

and settings. Figures 7.4 – 7.5 show the RL agent's performance with and without transfer learning against baseline agents.

Both figures illustrate that the RL agent was able to perform positively with transfer learning when compared to the one without transfer learning. By evaluating the three transfer learning metrics: jumpstart, time to threshold and asymptotic performance, it is evident that the RL agent successfully transferred experience from the *capital punishment* to the *BREXIT* domain resulting in speeding up of learning with improved learning performance in the target task [243]. Based on the three measurements mentioned earlier, Figure 7.4 demonstrates that the RL agent with transfer performed better than without. It shows that the RL agent with transfer

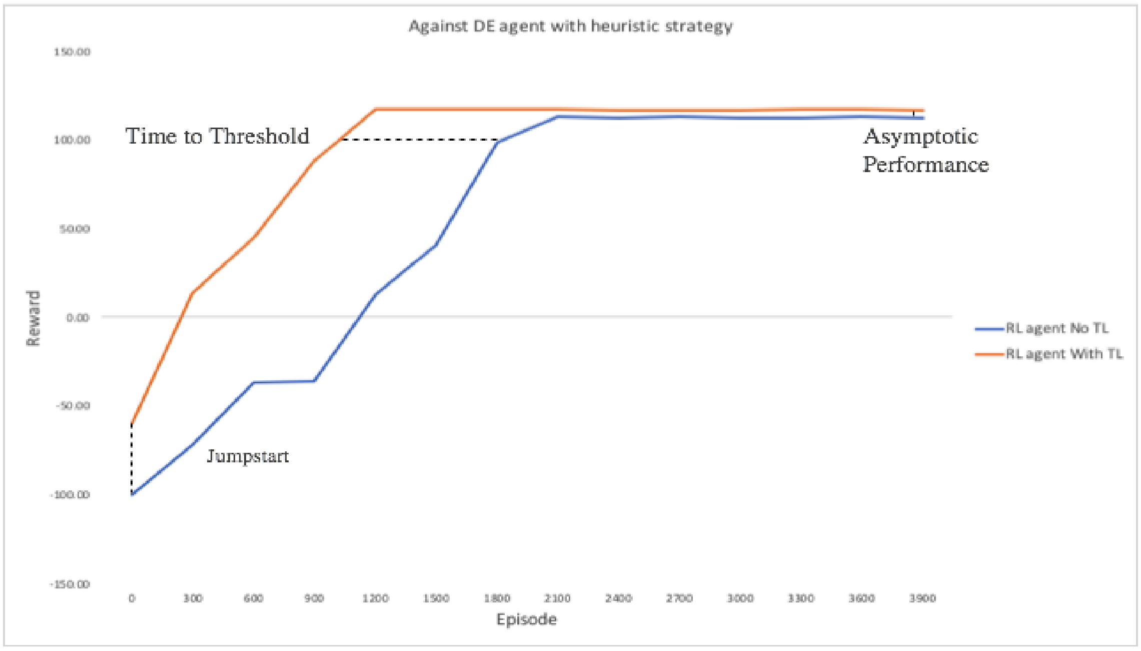


Figure 7.4: RL agent using transfer learning against DE agent with heuristic strategy

initially received a higher reward than without transfer and a significant carry on with higher reward. For the second measure, time to threshold, the RL agent with the transfer was earlier to threshold than without. Asymptotic performance, which is the third measure, shows that the RL agent with transfer is slightly higher than without.

Hence, the RL agent with transfer satisfied all the three evaluation metrics that were used to compare the performance of successful transfer and learning from scratch. This can be seen in Figure 7.5. The results are similar to the DE agents' performance that are shown in Figure 7.4 with minor differences in jumpstart and asymptotic performance. Therefore, by evaluating these three measures, the transfer learning goal was achieved and improvement in learning was demonstrated in the target task by leveraging knowledge acquired from the source task [176, 178].

Additionally, the key insight behind the transfer learning process is generalisation [178]. From the results of our experiments, it can be concluded that argumentation schemes and evidence support sources allow the RL agent to learn and transfer experience from a source domain to a target domain. This can be considered as a

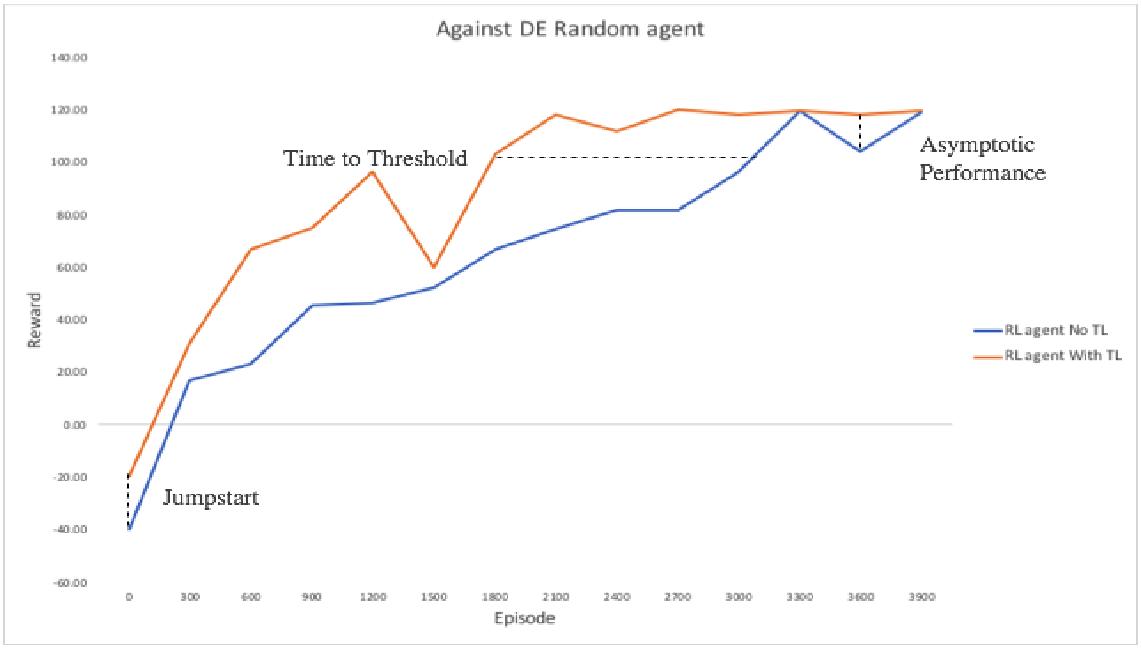


Figure 7.5: RL agent using transfer learning against DE agent with random strategy

generalisation approach for argumentation.

A non-parametric test (Mann-Whitney U test) was used to assess the significance of the findings. The assessment will be the time to threshold for RL agent with TL reached faster than RL agent without TL against DE agent with heuristic strategy. The trials to reach an average reward of 100 (time to threshold). The reason for choosing this test is mentioned before in Section 3 of Chapter 3.

Table 7.4: Basic statistics RL agent with TL and RL agent with NO TL against DE agent with heuristic strategy

Agent	**Mean**	**Standard deviation**	**Median**
RL agent with TL	850	150	900
RL agent with NO TL	1830	155	1800

From Table 7.4 the lower mean and median suggest that the RL agent with TL reach to time to threshold (average 100) faster than the RL agent with NO TL (against DE agent with heuristic strategy). Mann-Whitney U test was used to test the null hypothesis and the two groups were homogeneous against the alternative hypothesis that the time to threshold for the RL agent with transfer learning will be faster than the RL agent without transfer learning.

H_0**:** The two groups are homogeneous.

H_1**:** The time to threshold for the RL agent with transfer learning will be faster than the RL agent without transfer learning.

A Mann-Whitney U test showed the null hypothesis (the two groups are homogeneous) could be rejected with $p-value = 0.00007$ which was less than the significance level at 0.05. Therefore, time to threshold for the RL agent with TL was significantly faster than the RL agent with NO TL.

A non-parametric test (Mann-Whitney U test) was used to assess the significance of finding. The assessment will be the time to threshold for the RL agent with TL reached faster than the RL agent without TL against DE Random agent. Time to threshold is defined as the number of trials to reach average reward of 100. The reason for choosing this test is mentioned before in Section 3 of Chapter 3.

Table 7.5: Basic statistics RL agent with TL and RL agent with NO TL against DE Random agent

Agent	**Mean**	**Standard deviation**	**Median**
RL agent with TL	780	382	850
RL agent with NO TL	1470	691	1750

From Table 7.5 the lower mean and median suggest that the RL agent with TL reached time to threshold (average 100) faster than the RL agent with NO TL (against DE Random agent). Mann-Whitney U test was used to test the null hypothesis that the two groups were homogeneous against the alternative hypothesis that the time to threshold for the RL agent with transfer learning will be faster than the RL agent without transfer learning.

***H_0*:** The two groups are homogeneous.

***H_1*:** The time to threshold for the RL agent with transfer learning will be faster than the RL agent without transfer learning.

A Mann-Whitney U test showed that the null hypothesis (the two groups are homogeneous) could be rejected with $p - value = 0.020$ being less than the significance level of 0.05. Therefore, time to threshold for the RL agent with TL was significantly faster than the RL agent with NO TL.

6 Summary

In this chapter, transfer learning was used to generalise learning within a different domain. A new domain *BREXIT* was built in order to test transfer learning. Patterns (argument schemes and sources) were identified in both *capital punishment* and *BREXIT* domains. A tabular approach was used to implement in the target domain. The results from the experiments are promising. It can be concluded that the RL agent can reuse knowledge learnt in one domain into different domains for speeding up and improving learning.

CHAPTER 8

CONCLUSION AND FURTHER WORK

1 Conclusion

The research reported in this thesis explores how an agent learns to argue using reinforcement learning and applies it to argumentation theory. Towards this, literature on argumentation and reinforcement learning was reviewed. The literature review was particularly focused on applying argumentation theory to abstract argumentation systems and logic-based dialogue systems [6, 7, 11, 25, 26, 28, 29, 32, 44]. These systems, explore how an agent interacts with arguments by playing argument games. Reinforcement learning allows an agent to learn by interaction with the environment. It allows an agent to take actions and receive rewards in an observed state [38]. This research also involves the development of software test-beds for an abstract argument game [6, 29] and a DE dialogue game [7]. By enabling the RL agent to argue with baseline agents, the test-beds provide a platform to compare and assess the agents' performance.

From the results of this research, we are able to conclude that the RL agent was able to learn to argue effectively in abstract argument games and the logic-based dialogue games (these are discussed in Chapters 3 and 5). In both the games, the RL agent was able to win more games and outperform the baseline agents. Additionally, the RL agent was able to learn to adopt strategies such as coherence and relevance which were found to improve the quality of the dialogue (these are discussed in Chapter 6). The RL agent demonstrated an improved performance in the abstract argument game that was based on maximising the final reward for the number of acceptable arguments in grounded extensions. Whereas, in the logic-based dialogue games, the reward function was enhanced by allowing the RL agent to win the DE dialogue game with a minimum number of moves. Coherence and relevance were incorporated as measures in the reward function to encourage the RL agent to learn to improve the dialogue quality.

For the purpose of policy generalisation, it was challenging for the RL agent to find useful features to transfer in a simple argument game. Therefore, the research moved to a logic-based dialogue game with richer argument representations. Argument schemes and source evidence were proposed to transfer the learning. The transfer learning experiments, from the source task *capital punishment* to a new target task *BREXIT*, demonstrated that the RL agent with transfer learning showed a significant advantage over the RL agent without transfer learning.

The aim of this thesis, as stated in Chapter 1 was to fulfil the following:
To build an intelligent agent based on reinforcement learning that is able to learn to argue against baseline agents and demonstrate improved performance.

A variety of issues have been discussed in the preceding chapters that relate to achieving the above aim and responding to the research questions and objectives set out in Chapter 1. The research overview will be outlined below, to show how the following research objectives presented in chapter 1 were addressed.

Obj$_1$: To develop an RL agent that improves its performance over time by obtaining greater rewards compared with baseline agents in an abstract argument game.

Obj$_2$: To analyse whether using current features in abstract argumentation allows a RL agent to transfer experience from one abstract argument graph to another.

Obj$_{3a}$: To develop a RL agent that can improve its performance to the point of obtaining more wins than the DE agents.

Obj$_{3b}$: Reshape the reward function to enable a RL agent to outperform the DE agent by winning with the minimum number of moves.

Obj$_4$: To reshape the reward function such that the RL agent can improve dialogue quality in terms of increased relevance and coherence in the DE dialogue game.

Obj$_5$: To identify argument patterns, such as argument schemes and evidence sources, that can enable a DE-style RL agent to transfer learning between argument domains.

In Chapter 3 *ARGUMENTO+* was built to allow the RL agent to play abstract argument game against different baseline agents. The results show that the RL agent was able to improve its performance and outperform the baseline agents by obtaining more cumulative rewards than opponents. This indicates that the RL agent was able to learn to argue in a abstract argument game, and, therefore the Obj_1 was met.

In Chapter 4 the current features for abstract argumentation framework (i.e. arguments and relations) face challenges in capturing useful argument patterns. These are important for reusing the learnt arguments and relations in different argument graphs. Attack relations, which are the number of attackers and the number of immediately winning attackers, were also found to enable learning. However, the agent found it difficult to transfer experience within different argument graphs. Therefore the above mentioned features were not useful for the RL agent to generalise its policy.

Further, logic-based dialogue model was adopted to explore if this will enable the RL agent to generalise its policy. As presented in Chapter 5 the DE model was used, where the RL agent was allowed to debate with baseline agents. This approach showed promising results, wherein, the RL agent demonstrated improved performance by winning more games than the DE agents. Additionally, reshaping the reward function by incorporating the number of moves factor into the reward function, caused the RL agent to win the DE dialogue game with minimum number of moves. Hence the objectives Obj_{3a} and Obj_{3b} were met.

After objectives Obj_{3a} and Obj_{3b} were successfully achieved, Obj_4 i.e. dialogue quality was addressed. In this, the reward function was reshaped by incorporating other factors such as, coherence and relevance. The RL agent was able to learn to improve the dialogue quality by decreasing incoherence and irrelevance over time against the DE baseline agents. Therefore, the RL agent was able to learn to argue with increasing coherence and relevance in the DE dialogue game.

After learning to win the game with minimum number of moves with improved dialogue quality, generalising of policy was considered. In order to generalise the policy, it is required to identify patterns for transfer learning. It was suggested that argument schemes and evidence sources can be reused within different contexts as patterns for transferring experience. The RL agent was able to use the transfer learning method successfully to transfer experience between different contexts (i.e. Capital Punishment to *BREXIT*). A comparison of RL agent's performance with and without transfer showed promising results. Therefore the fifth objective Obj_5 was successfully addressed.

Towards addressing the objectives and research questions in Chapter 1, experiments were conducted. The results were analysed using appropriate statistical tests to assess the significance of the findings. Tables 8.1 to 8.12 provide a summary of proposed hypotheses of this research and the results of the statistical tests.

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
RL agent Vs. MaxProb Agent (Figure 3.11)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics $(-51.920) <$ critical value (-2.863).	Reject null hypothesis - the rewards for RL agent are stationary.
	MaxProb	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics $(-51.785) <$ critical value (-2.863).	Reject null hypothesis - the rewards for MaxProb agent are stationary.
	RL, MaxProb	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.	Reject null hypothesis - the RL agent had higher rewards than the MaxProb agent.

Table 8.1: Hypotheses test RL agent vs. MaxProb agent. **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent Vs. MinProb Agent(Figure 3.12)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics (-53.757) < critical value (-2.863).	Reject null hypothesis - the rewards for RL agent are stationary.
	MinProb	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics (-54.179) < critical value (-2.863).	Reject null hypothesis - the rewards for MinProb agent are stationary.
	RL, MinProb	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.	Reject null hypothesis - the RL agent had higher rewards than the MinProb agent.

Table 8.2: Hypotheses test RL agent vs. MinProb agent. **The significance level was 0.05.**

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
RL agent Vs. Random Agent (Figure 3.13)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics (-53.322) < critical value (-2.863).	Reject null hypothesis - the rewards for RL agent are stationary.
	Random	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics (-52.245) < critical value (-2.863) .	Reject null hypothesis - the rewards for Random agent are stationary.
	RL, Random	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.	Reject null hypothesis - the RL agent had higher rewards than the Random agent.

Table 8.3: Hypotheses test RL agent vs. Random agent. **The significance level is 0.05.**

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
Figure 5.5 RL agent Vs. DE Random Agent	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0 , ADF statistics $(-11.082) <$ critical value (-2.871).	Reject null hypothesis - the rewards for RL agent are stationary.
	DE Random	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics $-11.303 <$ the critical value -2.871 .	Reject null hypothesis - the rewards for DE Random agent are stationary.
	RL, DE Random	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.010.	Reject null hypothesis - the RL agent had higher rewards than DE Random agent.

Table 8.4: Hypotheses test RL agent vs. DE Random agent. **The significance level is 0.05.**

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
Figure 5.6	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics (-16.213) $<$ critical value (-2.871).	Reject null hypothesis - moves count for RL agent are stationary.

Table 8.5: Hypotheses test RL agent moves count (against DE agent). **The significance level is 0.05.**

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
Figure 5.7	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, the ADF statistics $(-17.366) <$ critical value (-2.871).	Reject null hypothesis - moves count for RL agent is stationary.

Table 8.6: Hypotheses test RL agent moves count (against DE Random agent). **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent Vs. DE heuristic agent (Incoherence) (Figure 6.3)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0.02 , ADF statistics $(-3.100) <$ critical value (-2.871).	Reject null hypothesis - incoherence values for RL agent are stationary.
	DE heuristic	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0, ADF statistics $(-16.966) <$ critical value (-2.871).	Reject null hypothesis - incoherence values for DE heuristic agent are stationary.
	RL, DE heuristic	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.0001.	Reject null hypothesis - RL agent had higher incoherence than DE heuristic agent.

Table 8.7: Hypotheses test RL agent vs. DE heuristic agent (Incoherence). **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent Vs. DE heuristic agent (Irrelevance) (Figure 6.4)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0 , ADF statistics (-10.401) < critical value (-2.871).	Reject null hypothesis - irrelevance values for RL agent are stationary.
	DE heuristic	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0 , ADF statistics (-9.579) < critical value (-2.871).	Reject null hypothesis - irrelevance values for DE heuristic agent are stationary.
	RL, DE heuristic	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.0001.	Reject null hypothesis - the DE agent had higher irrelevance than RL agent.

Table 8.8: Hypotheses test RL agent vs. DE heuristic agent (Irrelevance). **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent Vs. DE Random agent (Incoherence) (Figure 6.5)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0 , ADF statistics $(-4.588) <$ critical value (-2.871).	Reject null hypothesis - the incoherence values for RL agent are stationary.
	DE Random	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0 , ADF statistics $(-4.937) <$ critical value (-2.871).	Reject null hypothesis - the incoherence values for DE Random agent are stationary.
	RL, DE Random	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.016.	Reject null hypothesis - the DE Random agent had higher incoherence than RL agent.

Table 8.9: Hypotheses test RL agent vs. DE Random agent (Incoherence). **The significance level is 0.05.**

Experiment	**Agent**	**Null hypothesis**	**Test**	**Significance**	**Decision**
RL agent Vs. DE Random agent (Irrelevance) (Figure 6.6)	RL	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0.00002 , ADF statistics (-4.997) < critical value (-2.871).	Reject null hypothesis - the irrelevance values for RL agent are stationary.
	DE Random	That there exists a unit root (non-stationarity).	ADF test.	p-value = 0.02, ADF statistics (-3.073) < critical value (-2.871).	Reject null hypothesis - the irrelevance values for DE Random agent are stationary.
	RL, DE Random	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.000085.	Reject null hypothesis - the DE Random agent had higher irrelevance than RL agent.

Table 8.10: Hypotheses test RL agent vs. DE Random agent (Irrelevance). **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent with TL Vs. RL Agent No TL (Against DE heuristic) (Figure 7.4)	RL agent with TL, RL agent No TL	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.00007.	Reject null hypothesis - the time to threshold for the RL agent with transfer learning was faster than the RL agent without transfer learning.

Table 8.11: Hypotheses test RL agent with TL Vs. RL Agent No TL (Against DE heuristic). **The significance level is 0.05.**

Experiment	Agent	Null hypothesis	Test	Significance	Decision
RL agent with TL Vs. RL Agent No TL (Against DE Random) (Figure 7.5)	RL agent with TL, RL agent No TL	The two groups are homogeneous.	Mann-Whitney U test.	p-value = 0.020.	Reject null hypothesis - the time to threshold for the RL agent with transfer learning was faster than the RL agent without transfer learning.

Table 8.12: Hypotheses test RL agent with TL Vs. RL Agent No TL (Against DE Random). **The significance level is 0.05.**

2 Contributions to knowledge

This research is a contribution to the design and implementation of argumentative software agents. With regard to the design, a new sophisticated dialogue state representation was proposed in *ARGUMENTO+* which involves $levelOfTree$, $agentID$, $currentArgument$ and $previousArgument$. It replaces the classic state representation provided in the literature [6, 25, 211] to make each state unique. In the DE dialogue game, a new state representation was proposed for use by the RL agent. In the DE model [7], the agents initially considered previous moves as a dialogue state only. The new state representation, which combines previous moves with commitment stores for both players, is adopted for making the dialogue state as unique as possible for the RL agent. The RL agent performs significantly better than baseline agents by adopting new state representations in both *ARGUMENTO+* and DE dialogue game respectively.

A new reward function was designed by counting the number of acceptable arguments in grounded extensions in *ARGUMENTO+*. It was demonstrated that the RL agent is able to win the games using this reward function and improve its performance. Two new reward functions were proposed for the DE dialogue game. One is for encouraging the RL agent to win the dialogue game with a minimum number of arguments moves, whereas the other reward function is designed to motivate the agent to obtain a greater reward based on four attributes: winning, number of movements, number of contradictions and number of focus switches. Compared with the state-of-the-art in dialogue quality [225, 226], the RL agent demonstrates promising results when using the proposed reward function, as originally developed by Amgoud and de Saint-Cyr [225].

A new method for using argument schemes and evidence sources was designed to transfer learning between tasks in the DE dialogue model. The *BREXIT* knowledge base was designed with a rich set of arguments to establish a testing domain for transfer learning. The knowledge base was built based on a modified version of Toulmin's schema [7, 133]. The knowledge bases for both *capital punishment* and

BREXIT were enhanced by adding argument schemes and evidence support. As such, these patterns could be related between tasks through the transfer learning process [178]. The experiments demonstrate that the RL agent was able to reuse knowledge in different domains for speeding up and improving learning.

With regard to the implementation, the present research can be regarded as an evolution of *ARGUMENTO* (the work by Yuan *et al.* [6, 29]). Along these lines, the *ARGUMENTO+* argument game version was developed. This enables an agent to learn to argue with another agent based on the Q-learning algorithm. The RL agent was able to make moves primarily based on optimal actions to maximise cumulative rewards. The developed strategy was based on the optimal policy. An RL agent was also implemented in the DE dialogue game [7]. The DE RL agent was developed based on the Q-learning algorithm, which performed better than DE baseline agents in terms of number of moves and dialogue quality. Both *ARGUMENTO+* and DE RL agent are built using the object-oriented programming language Java, they can be reused by researchers in the area to test their agents.

In order to determine whether the research questions of this research have been addressed effectively in this thesis and how they are related to the thesis contributions, this section revisits the research questions mentioned in Chapter 1, as follows:

RQ_1: Is it possible for an RL agent to learn to win an abstract argument game and demonstrate improved performance?

RQ_2: Do the current features of abstract argumentation systems allow an RL agent to generalise its learning approach to other abstract argumentation graphs?

RQ_3: Is a RL agent more likely to win a DE dialogue game with minimum number of moves using a reward function to improve its performance than DE heuristic agents?

RQ_4: Does the reshaped reward function, which takes into account attributes such as the number of contradictions and switches of focus, improve the quality of the RL agent's dialogue contributions and makes the dialogue more coherent and relevant?

RQ$_5$: Does using the argument's internal structure, such as argumentation schemes and evidence support sources encourage the RL agent to transfer learning to different domains using a DE model?

For RQ$_1$ it can be clearly seen from the contributions stated above that, the RL agent is able to learn and improve the performance in an abstract argument game. As mentioned in Chapter 3, *ARGUMENTO+* is introduced and the RL agent outperformed baseline agents. This led to wining more games by the RL agent by adapting the optimal policy.

The RQ$_2$ was investigated in Chapter 4. The results showed that the current features of abstract argumentation system are challenging for an RL agent to identify patterns to transfer learning experience from one argumentation graph to another. Although, number of attackers was implemented, the RL agent could not find useful patterns that could be reused in different argumentation graphs. Therefore, it can be inferred that current features in abstract argumentation present challenges to identifying useful argument patterns that could be reused in different argument graphs.

Regarding RQ$_3$, Chapter 5 shows that, the RL agent was able to learn to argue by winning more games in the DE dialogue game. The RL agent outperformed the DE baseline agents with lower number of moves. Reward function was designed to encourage the RL agent to win more games with lesser number of moves.

In order to look deeply into dialogue quality, RQ$_4$ considered the reshaped reward function. Other factors such as contradictions and switching focus were added to reshape the reward function. This reward function led to minimising incoherence and irrelevance in the RL agent. Therefore, findings presented in Chapter 6 suggest that the reshaped reward function caused an improvement in coherence, relevance and dialogue quality.

The final research question RQ_5 concerns whether argument internal structure (i.e. argumentation schemes and evidence support sources) encourage RL agent to transfer learning experience within different context. Our results in Chapter 7 showed that the RL agent allowed to transfer learning successfully. For this, the agent followed the evaluation criteria in transfer learning approach such as jumpstart and time to threshold which was previously mentioned and discussed in Chapter 7. Hence these patterns which are argumentation schemes and evidence support sources support the RL agent for transferring learning successfully.

From the perspective of further work and factors that need to be taken into account for upcoming studies, it is pertinent to mention the limitations of this research. The limitations of this research and ideas for future research will be presented in the next section.

3 Limitations and further work

Our findings suggest that the RL agent can learn to argue against baseline agents. The results show that, the RL agent was able to learn within inference and logic based dialogue game levels of argumentation [26]. These were successfully demonstrated in Chapter 3, Chapter 5 and Chapter 6 along with transfer of experience.

However, the results of this research are bound by certain limitations. The current abstract argument game *ARGUMENTO+* does not implement the backtracking strategy as was previously implemented in [94]. The current game adopted the game in the literature for reasons of simplicity of rules [6, 25, 29]. In literature backtracking is often perceived as a successful approach Yuan *et al.* [6] and Prakken [94]. Prakken [32] argues that, incorporating a backtracking strategy in argument games would be more flexible and fair for the agent by providing an explicit answer structure on dialogue where each move can constitute an attack or a surrender to an earlier move by the opponent. Backtracking is flexible in some respects such as, moving to an earlier branch and postponing replies to move [85]. However, for implementing the backtracking strategy, the simple argument game would need to be enhanced by developing new game rules to handle backtracking. It would be an interesting direction to take and look deeply into how to implement backtracking strategy for agents in *ARGUMENTO+* in the longer term as future work.

This research adopts games involving only two agents which play against each other based on the game rules discussed in Chapter 3 and Chapter 5. Extending this work to multi-agent learning with more than two agents playing the game would be helpful to allow more than two agents playing the game. Playing games with two or more agents in systems has a tradition in game theory [245]. Agents would need to coordinate and collaborate with each other in order to beat other agents [246], which can lead to enhanced competitiveness [56]. This would also open up the question on how to make the rules of the argument game fairer for all players which could make way for future work.

BREXIT knowledge base is implemented in Chapter 7 to make the RL agent transfer learning between different contexts. This knowledge base (*BREXIT*) is the highest and trending collection of rich set of for and against arguments. The motivation behind choosing *BREXIT* was the availability of arguments which are derived from the heated debates on whether the UK should leave or remain in the EU. From the knowledge base some topics from the most debated issues about *BREXIT* from September 2016 to March 2019 were selected such as, NHS, taxes and employment. However, the *BREXIT* knowledge base consists of other topics which have not been covered in this research. Also, within building the knowledge base there were some challenges to find the primary sources for collecting the information. As a result majority of the information used in this research was collected from secondary sources i.e. newspapers, TV documentaries with limited primary sources.

However, the work represents the first step towards creating an intelligent agent that can learn to argue. More study and further work is needed to expand our understanding on this topic and address open questions. Further research can be carried out in a number of interesting directions.

The RL agent currently plays against baseline agents, one future step that can be considered is self-playing; namely, making RL agents play with each other and studying the consequences. In self-playing, each agent aims to maximise its own rewards. Eck and Wazel [247] studied two q-learning agents playing a game based on sequential decisions and found that both agents were able to learn better than agents using a straightforward positional or mobility strategy. Marek in [248], Papahristou and Refanidis in [249], and Tokic and Palm in [250] found that making both learning agents play with each other by implementing reinforcement learning is an effective means to encourage exploration. Given the above arguments, it would be interesting to study two reinforcement learning agents playing a DE dialogue game and then evaluate the results.

Further work may also consider propositional logic based argument patterns in the DE dialogue model, which may be beneficial for learning transfer [7, 98, 178]. This will enable the RL agent to recognise a pattern for use in transfer learning and conduct task mapping between source and target tasks. It is suggested that the heuristic strategies introduced by Yuan *et al.* [98] could be used to show how the agent should argue by using propositional logic when it is faced with such situations. For instance when an agent is facing a statement P it is more likely to use $Q\ if$ $(Q \implies \neg P)$. These heuristic rules may allow the RL agent to learn and use patterns and related maps between tasks [178].

This research considers learning agents which argue against the baseline agents. Further experimental work can be done by playing the RL agent with human users. In the literature the DE dialogue model was developed for an educational purpose by allowing students to argue against DE heuristic agents [7, 44]. It is worthwhile allowing human users to conduct an experiment against the RL agent in the DE dialogue model and evaluate the results. For *ARGUMENTO+*, it would also be interesting to conduct experiments with human users since its precedent *ARGUMENTO* facilitates an intelligent agent playing against human users and the evaluation revealed that the game was both challenging and entertaining [29]. It would be interesting to evaluate whether *ARGUMENTO+* is also challenging and entertaining when playing with human users.

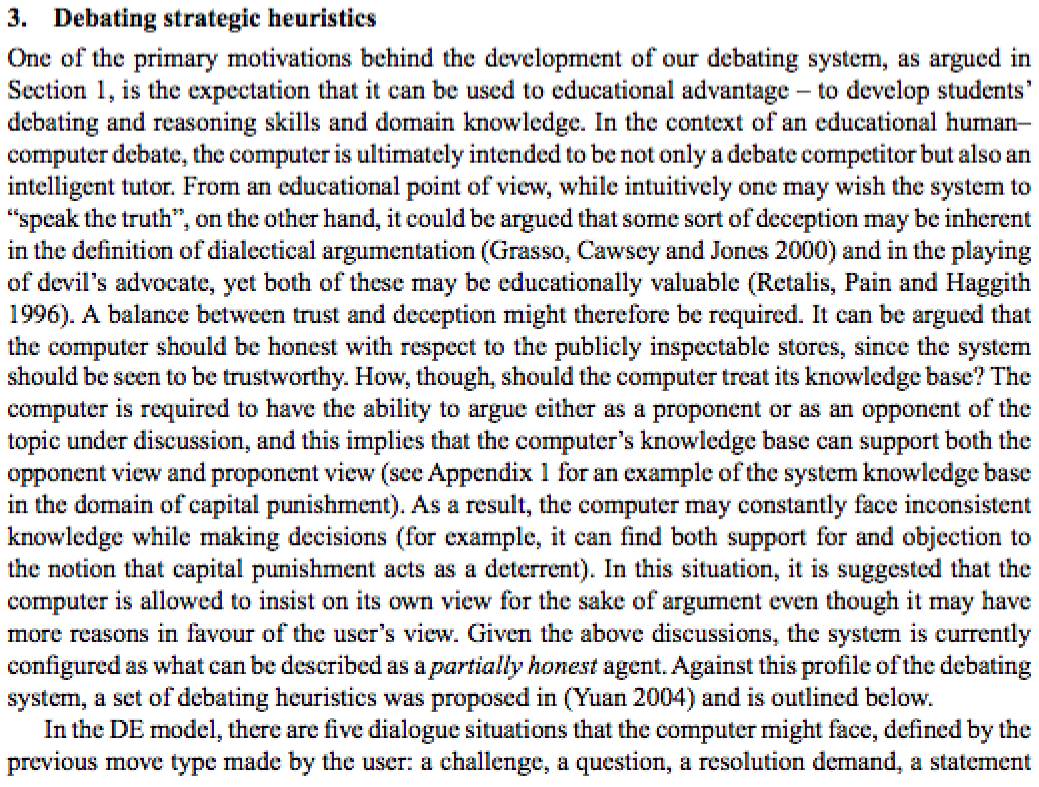
At the end of the PhD journey, this thesis has shown how the RL agent can learn to argue against the baseline agents. Taking into account the expectations from the objectives and research questions set out at the beginning of the research, the results are generally promising. The hypotheses and statistical test shows also significant findings. The research provides grounds for future research. However, this is the start and not the end, as there are many unanswered research questions. It is believed that additional techniques can be explored in the field of argumentation and reinforcement learning to address the current limitations and add useful contributions to this research.

The End

APPENDIX A

Appendix A

These are the debating strategy heuristics for the DE computational agents in the DE dialogue model, taken from Yuan et al. [98, pp.143-147]:



3. Debating strategic heuristics

One of the primary motivations behind the development of our debating system, as argued in Section 1, is the expectation that it can be used to educational advantage – to develop students' debating and reasoning skills and domain knowledge. In the context of an educational human–computer debate, the computer is ultimately intended to be not only a debate competitor but also an intelligent tutor. From an educational point of view, while intuitively one may wish the system to "speak the truth", on the other hand, it could be argued that some sort of deception may be inherent in the definition of dialectical argumentation (Grasso, Cawsey and Jones 2000) and in the playing of devil's advocate, yet both of these may be educationally valuable (Retalis, Pain and Haggith 1996). A balance between trust and deception might therefore be required. It can be argued that the computer should be honest with respect to the publicly inspectable stores, since the system should be seen to be trustworthy. How, though, should the computer treat its knowledge base? The computer is required to have the ability to argue either as a proponent or as an opponent of the topic under discussion, and this implies that the computer's knowledge base can support both the opponent view and proponent view (see Appendix 1 for an example of the system knowledge base in the domain of capital punishment). As a result, the computer may constantly face inconsistent knowledge while making decisions (for example, it can find both support for and objection to the notion that capital punishment acts as a deterrent). In this situation, it is suggested that the computer is allowed to insist on its own view for the sake of argument even though it may have more reasons in favour of the user's view. Given the above discussions, the system is currently configured as what can be described as a *partially honest* agent. Against this profile of the debating system, a set of debating heuristics was proposed in (Yuan 2004) and is outlined below.

In the DE model, there are five dialogue situations that the computer might face, defined by the previous move type made by the user: a challenge, a question, a resolution demand, a statement or a withdrawal. Each therefore needs to be considered in relation to the strategic decisions the computer might need to make. It has been argued (e.g. Moore 1993) that these decisions are best captured at three levels.

(1) Retain or change the current focus.
(2) Build own view or demolish the user's view.
(3) Select method to fulfil the objective set at levels 1 and 2.

Levels (1) and (2) refer to strategies which apply only when the computer is facing a statement or withdrawal, since in all other cases the computer must respond to the incoming move. Level (3) refers to tactics used to reach the aims fixed at levels (1) and (2) and applies in every game situation. These levels of decisions are discussed in turn below.

Level (1) decision concerns whether to retain the current focus or to change it. The decision, that is, involves whether to continue the attempt to substantiate or undermine a particular proposition. Moore (1993) argues that continuing to execute a plan of questions or addressing the previous move will guarantee that the current focus is retained but that it is possible not to directly address the user's latest utterance yet still retain focus. Moore further suggests that there is a presumption in favour of addressing the previous move, but that this presumption may be broken when the line of questioning is deemed a blind alley, or if a successful removal of the user's support has been made, or if, on regaining the initiative after a period without it, a resolution demand can legally be made.

The decision at level (2) considers whether to adopt a build or a demolish strategy. A build strategy involves seeking acceptance of propositions that support the computer's own thesis, while a demolish strategy seeks to remove the user's support for his thesis. The decision is needed only at the beginning of games and when level (1) decision involves a shift in focus. A demolish strategy could possibly be part of a broader build strategy, e.g. a goal-directed plan of questions building the computer's own view might involve removing some unwanted responses from the user. A building attempt might also be part of a broader demolish strategy, e.g. the computer is using a line of questions to build the case for P in order to attack the user's view ¬P. Moore found no evidence to suggest a priority between the build and demolish strategy. In the current debating system, we give priority to a build strategy and make the computer try to open as many subtopics as possible, on the grounds that in an educational debate the aim is to expose the full complexity of the situation and this might be best achieved if the computer seeks to continue until all the knowledge base (henceforth referred as KB) has been explored. Moore also argues that the decisions at levels (1) and (2) heavily depend on the results of level (3) methods. In the current debating system, for example, the computer checks level (3) methods first; if there are level (3) methods available, level (1) and (2) decisions do not need to be applied; however, if there is no level (3) method available, level (1) and (2) decisions will come into play, in that level (1) decision may be to switch the current focus and level (2) decision is to build the computer's thesis if there are build methods available.

The third level of decisions applies to each of the dialogue situations. Level (3) heuristics for each dialogue situation are given in turn below.

3.1. *A question raised by the user*

Questions asked may involve questioning an individual statement, e.g. "Is it the case that P?" or a conditional, e.g. "Is it the case that Q implies P?". In these situations, the computer is allowed by the DE rules to answer *Yes*, *No* or "no commitment". Moore (1993) suggests that the decision must be based on the truth-seeking nature of the game. In the current proposal, the system is required to be a partially honest agent as argued earlier. In addition, Moore suggests one should give an answer in such a way as to avoid unwelcome commitment. Given this, heuristics for a situation in which the computer is facing a question can be proposed as follows (assume the question is "is it the case that P?").

- (Q1) If neither P nor ¬P can be found in its *KB*, then the computer replies with a "no commitment".
- (Q2) If only one of them (P and ¬P) can be found in the *KB*,
  - (Q2a) If the computer has previously uttered "no commitment" to the found statement, then it utters "no commitment" to remain consistent.
  - (Q2b) Else the computer utters the found statement.
- (Q3) If both (P, ¬P) are found in the computer's *KB*, and assuming that one of them (say ¬P) supports the computer's view and the other (say P) supports the user's view.
  - (Q3a) If the computer has an acceptable support for ¬P, then utter ¬P.
  - (Q3b) If the computer has no acceptable support for ¬P, and the computer has not committed to propositions supporting P, the computer should utter "no commitment".
  - (Q3c) If the computer has no acceptable support for ¬P, and the computer has committed to propositions supporting P, then the computer should utter P.

3.2. *A challenge made by the user*

There are three DE legal options available in response to a challenge: a resolution demand, a support or a withdrawal. The first option concerns an inconsistency when the user is challenging a modus ponens consequence of his/her own commitments. From an educational point of view, it can be argued that the computer should point out this inconsistency and make the user aware of this kind of inconsistency in a debate. For the latter two options, Moore's (1993) experimental analysis suggests that one would normally reply with a carefully chosen support if available. In DE, there is no guidance within the rules as to the content of the support. The selection between alternative supports may be influenced by the profile of the agent. Given the definition of the profile of a partially honest agent, the computer should give a support according to its knowledge structure rather than invent one which may not be a suitable support. In addition, it can be suggested that a support which can be further supported is preferred over one which cannot be further supported, since a further challenge might be expected from the user. Given this, the heuristics after a challenge of P can be proposed as follows.

- (C1) If P is a modus ponens consequence of the user's commitment, then pose a resolution demand.
- (C2) Else if there is only one acceptable support available in the *KB*, then state the support.
- (C3) Else if there is more than one acceptable support available, then state the one that can be further supported.
- (C4) Else if all the available acceptable supports are equally supported, then randomly choose one of the supports.
- (C5) Else if no acceptable support is available, then withdraw P.

3.3. *A resolution demand made by the user*

A resolution demand made by the user concerns an allegation that the computer has committed to an inconsistency in its commitment store. In the most likely event, the computer would face a resolution demand of the type "resolve {¬P, P}", in that the computer has committed to both P and ¬P. In this situation, the computer is required to withdraw one of them to keep consistent. Following Moore (1993), the computer should withdraw the statement which has the smaller number of grounding statements in the commitment store at the time the resolution demand is made.

The user might invoke another type of resolution demand (i.e. resolve (Q, Q⊃P, why P) or resolve (Q, Q⊃P, "no commitment" P)) in the event of the computer's challenging or withdrawing a modus ponens consequence of its commitments. In this situation, the computer is required, by the game DE, to withdraw either Q or Q⊃P or affirm P. Moore (1993) argues that the use of such a resolution demand would suggest that, in the user's view at least, the computer has challenged or withdrawn a proposition to which it ought to be committed given the remainder of its commitment store. In such a case, given the partially honest agent profile argued for earlier, the computer takes the option of affirming the disputed consequent P.

3.4. *A "no commitment" made by the user*

After a "no commitment", DE places no restrictions on either move type or contents. The computer's decisions are therefore more open. Following Moore (1993), the heuristics after a "no commitment" are proposed as follows.

(W1) If the computer is facing a "no commitment" to a statement supporting the user's thesis
- (W1a) If the withdrawn statement is a unique support of the user's asserted proposition Q, and Q is not the user's thesis, then challenge Q.
- (W1b) Else check whether the user retains adherence to the thesis.

(W2) If the computer is facing a "no commitment" to a statement supporting the computer's thesis
- (W2a) If the non-committal statement is a modus ponens consequence of the user's commitments, then pose a resolution demand.
- (W2b) Else switch the focus.

3.5. *A statement made by the user*

After a statement, there is no restriction on either move types or move contents in *DE*. Intuitively, one would expect the user to assert a statement which supports his view or opposes the computer's view. However, it is possible that the user may unwisely make a statement which supports the computer's view or goes against his/her own view. The computer may need to deal with these two kinds of statement differently. When the computer is facing a statement (say P) which supports the computer's thesis or militates against the user's view, two heuristics are proposed as follows.

(S1a) If P is a support of the computer's thesis, then use P as the starting point to build a case for the computer's thesis.
(S1b) Else check whether the user still adheres to his/her thesis.

When the computer is facing a statement (say P) which supports the user's view or militates against the computer's view, a set of heuristics is proposed as follows, in line with Moore (1993).

(S2a) If there is an inconsistency (e.g. (P, ¬P)) in the user's commitment store, then ask for resolution.
(S2b) Else if there is a piece of hard evidence in support of ¬P, then state the piece of hard evidence (where a piece of hard evidence is taken to refer to a statistically or scientifically validated fact, as seen by the creator of the KB).
(S2c) Else if there is any support of ¬P and the support (say Q) can be further supported, then state ¬P or state Q if ¬P has been uttered, or form a plan of questions making the user accept ¬P.

(S2d) Else if there is any support of ¬P and the support cannot be further supported, then form a plan of questions making the user accept ¬P.
(S2e) Else if P is challengeable, then challenge it.
(S2f) Else switch the focus.

To decide whether a statement is challengeable, the computer needs to consider the nature of that statement (e.g. whether it is a piece of hard evidence) and the relevant DE dialogue rules. If the computer arrives at option e and the statement in question is not challengeable, the computer reverts to level (1) of the strategic decision-making process.

A further concern is how the plans of questions in heuristics (c) and (d) are organised. Following the Walton, Reed and Macagno (2008) scheme of *argument from gradualism*, the plan can be started by asking a question of a proposition (say A), followed by a series of connected conditionals (say $A \supset B$, $B \supset C$, . . ., $C \supset P$) towards the conclusion (say P). Moore (1993) argues that the computer should hand over the initiative by stating the conclusion P at the end if the plan is executed successfully, with a view to avoiding a one-sided dialogue.

During a plan execution, the user might give unwanted answers (i.e. answers not favourable to the computer's plan). The approach taken here is that the computer tries to remove the obstacles (unwanted answers) and put the plan back on track while the initiative is still held. The plan execution process is as follows.

(P1) If a wanted answer is given, then carry on to execute the plan
(P2) If a non-committal answer is given
  (P2a) If there is an expressed inconsistency in the user's CS, then pose the appropriate resolution demand
    (i) If the user affirms the disputed consequence, then continue the plan
    (ii) Else abandon this line of questions
  (P2b) Else abandon this line of questions
(P3) If an unwanted answer (e.g. ¬P rather than P) is given
  (P3a) If there is an expressed inconsistency in the user's commitment store and the unwanted answer ¬P is an element of the inconsistency, then pose the appropriate resolution demand
    (i) If the unwanted answer is withdrawn, then continue the plan and re-pose the question.
    (ii) Else abandon this line of question
  (P3b) Else if the unwanted statement is challengeable, then challenge the unwanted statement
    (i) If the unwanted answer is withdrawn, then continue the plan to repose the question of P.
    (ii) Else abandon this line of question
  (P3c) Else abandon this line of question

This, then, is the set of strategic heuristics currently adopted by our human–computer debating system. The following sections consider the evaluations of such strategy.

BIBLIOGRAPHY

[1] S. Alahmari, T. Yuan, and D. Kudenko, "Reinforcement learning for abstract argumentation: Q-learning approach," in *Adaptive and Learning Agents workshop (at AAMAS 2017)*, 2017.

[2] S. Alahmari, T. Yuan, and D. Kudenko, "Reinforcement learning for argumentation: Describing a PhD research.," in *CMNA@ ICAIL*, pp. 76–78, 2017.

[3] S. Alahmari, T. Yuan, and D. Kudenko, "Policy generalisation in reinforcement learning for abstract argumentation," in *Proceedings of the 18th Workshop on Computational Models of Natural Argument (CMNA18)*, 2018.

[4] S. Alahmari, T. Yuan, and D. Kudenko, "Reinforcement learning for dialogue game based argumentation," in *Proceedings of the 19th Workshop on Computational Models of Natural Argument (CMNA19)*, 2019.

[5] S. Alahmari, T. Yuan, and D. Kudenko, "Reinforcement learning of dialogue coherence and relevance," in *Proceedings of the 19th Workshop on Computational Models of Natural Argument (CMNA19)*, 2019.

[6] T. Yuan, V. Svansson, D. Moore, and A. Grierson, "A computer game for abstract argumentation," in *Proceedings of the 7th Workshop on Computational Models of Natural Argument (CMNA07)*, 2007.

[7] T. Yuan, *Human-Computer Debate, a Computational Dialectics Approach*. PhD thesis, Unpublished Doctoral Dissertation, Leeds Metropolitan University, 2004.

[8] T. Kelly and R. Weaver, “The goal structuring notation–a safety argument notation,” in *Proceedings of the dependable systems and networks 2004 workshop on assurance cases*, p. 6, Citeseer, 2004.

[9] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[10] D. L. Poole and A. K. Mackworth, *Artificial Intelligence: foundations of computational agents*. Cambridge University Press, 2010.

[11] M. Wooldridge, *An introduction to multiagent systems*. John Wiley & Sons, 2009.

[12] I. Rudowsky, “Intelligent agents,” *Communications of the Association for Information Systems*, vol. 14, no. 1, p. 14, 2004.

[13] Y. Shoham and K. Leyton-Brown, *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.

[14] M.-P. Huget, “Design agent interaction as a service to agents,” in *Communication in Multiagent Systems*, pp. 209–222, Springer, 2003.

[15] V. M. Catterson, E. M. Davidson, and S. D. McArthur, “Practical applications of multi-agent systems in electric power systems,” *European Transactions on Electrical Power*, vol. 22, no. 2, pp. 235–252, 2012.

[16] S. D. Ramchurn, D. Huynh, and N. R. Jennings, “Trust in multi-agent systems,” *The Knowledge Engineering Review*, vol. 19, no. 1, pp. 1–25, 2004.

[17] A. R. Panisson and R. H. Bordini, “Uttering only what is needed: Enthymemes in multi-agent systems,” in *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1670–1672, International Foundation for Autonomous Agents and Multiagent Systems, 2017.

[18] J. Foerster, I. A. Assael, N. de Freitas, and S. Whiteson, “Learning to communicate with deep multi-agent reinforcement learning,” in *Advances in Neural Information Processing Systems*, pp. 2137–2145, 2016.

[19] T. J. Bench-Capon and P. E. Dunne, "Argumentation in artificial intelligence," *Artificial intelligence*, vol. 171, no. 10-15, pp. 619–641, 2007.

[20] I. Rahwan, "Guest editorial: Argumentation in multi-agent systems," *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 2, pp. 115–125, 2005.

[21] S. Parsons and P. McBurney, "Argumentation-based communication between agents," in *Communication in Multiagent Systems*, pp. 164–178, Springer, 2003.

[22] T. J. Norman, D. V. Carbogim, E. C. Krabbe, and D. Walton, "Argument and multi-agent systems," in *Argumentation Machines*, pp. 15–54, Springer, 2003.

[23] P. McBurney, I. Rahwan, and S. D. Parsons, *Argumentation in Multi-Agent Systems: 7th International Workshop, ArgMAS 2010, Toronto, Canada, May 10, 2010, Revised Selected and Invited Papers*, vol. 6614.
Springer, 2012.

[24] D. Walton, "Argumentation theory: A very short introduction," in *Argumentation in artificial intelligence*, pp. 1–22, Springer, 2009.

[25] M. Wooldridge, *An introduction to multiagent systems*.
John Wiley & Sons, 2002.

[26] H. Prakken, "Historical overview of formal argumentation," *IfCoLog Journal of Logics and their Applications*, vol. 4, no. 8, pp. 2183–2262, 2017.

[27] P. Baroni and M. Giacomin, "Semantics of abstract argument systems," in *Argumentation in artificial intelligence*, pp. 25–44, Springer, 2009.

[28] P. M. Dung, "On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games," *Artificial intelligence*, vol. 77, no. 2, pp. 321–357, 1995.

[29] T. Yuan, J. Schulze, J. Devereux, and C. Reed, "Towards an arguing agents competition: Building on argumento," in *Proceedings of IJCAI‚Äô2008 Workshop on Computational Models of Natural Argument*, 2008.

[30] D. Walton and E. C. Krabbe, “Commitment in dialogue,” 1995.

[31] P. McBurney and S. Parsons, “Dialogue games for agent argumentation,” in *Argumentation in artificial intelligence*, pp. 261–280, Springer, 2009.

[32] H. Prakken, “Formal systems for persuasion dialogue,” *The knowledge engineering review*, vol. 21, no. 2, pp. 163–188, 2006.

[33] H. Prakken, “Models of persuasion dialogue,” in *Argumentation in artificial intelligence*, pp. 281–300, Springer, 2009.

[34] M. Možina, J. Žabkar, T. Bench-Capon, and I. Bratko, “Argument based machine learning applied to law,” *Artificial Intelligence and Law*, vol. 13, no. 1, pp. 53–73, 2005.

[35] M. Hall, I. Witten, and E. Frank, “Data mining: Practical machine learning tools and techniques,” *Kaufmann, Burlington*, 2011.

[36] T. M. Mitchell, *Machine Learning*.
USA: McGraw-Hill, Inc., 1 ed., 1997.

[37] O. Cocarascu and F. Toni, “Argumentation for machine learning: A survey.,” in *COMMA*, pp. 219–230, 2016.

[38] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*, vol. 1.
MIT press Cambridge, 1998.

[39] S. Kapoor, “Multi-agent reinforcement learning: A report on challenges and approaches,” *arXiv preprint arXiv:1807.09427*, 2018.

[40] V. Rieser and O. Lemon, *Reinforcement learning for adaptive dialogue systems: a data-driven methodology for dialogue management and natural language generation*.
Springer Science & Business Media, 2011.

[41] N. Maudet and F. Evrard, “A generic framework for dialogue game implementation,” in *Proceedings of the Second Workshop on Formal Semantics and Pragmatics of Dialog*, 1998.

[42] J. D. Mackenzie, “Question-begging in non-cumulative systems,” *Journal of Philosophical Logic*, vol. 8, no. 1, pp. 117–133, 1979.

[43] D. J. Moore, *Dialogue game theory for intelligent tutoring systems.*
PhD thesis, Leeds Metropolitan University, 1993.

[44] T. Yuan, D. Moore, and A. Grierson, “A human–computer debating system prototype and its dialogue strategies,” *International Journal of Intelligent Systems*, vol. 22, no. 1, pp. 133–156, 2007.

[45] T. Yuan, D. Moore, and A. Grierson, “A human-computer dialogue system for educational debate: A computational dialectics approach,” *International Journal of Artificial Intelligence in Education*, vol. 18, no. 1, pp. 3–26, 2008.

[46] T. Yuan, D. Moore, C. Reed, A. Ravenscroft, and N. Maudet, “Informal logic dialogue games in human–computer dialogue,” *The Knowledge Engineering Review*, vol. 26, no. 2, pp. 159–174, 2011.

[47] A. Algarni, *Applying Reinforcement Learning in Cloud Computing (Qualified Dissertation).*
PhD dissertation, Department of Computer Science, University of York, 2014.

[48] E. Levin, R. Pieraccini, and W. Eckert, “A stochastic model of human-machine interaction for learning dialog strategies,” *IEEE Transactions on speech and audio processing*, vol. 8, no. 1, pp. 11–23, 2000.

[49] J. D. Williams and S. Young, “Partially observable markov decision processes for spoken dialog systems,” *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.

[50] P. S. Kośmicki, “A platform for the evaluation of automated argumentation strategies,” in *International Conference on Rough Sets and Current Trends in Computing*, pp. 494–503, Springer, 2010.

[51] G. Weiss, *Multiagent systems: a modern approach to distributed artificial intelligence.*
MIT press, 1999.

[52] M. Glavic, "Agents and multi-agent systems: a short introduction for power engineers," 2006.

[53] S. Russell, P. Norvig, and A. Intelligence, *A modern approach*.
Pearson Education Limited, 1995.

[54] A. Dorri, S. S. Kanhere, and R. Jurdak, "Multi-agent systems: A survey," *IEEE Access*, vol. 6, pp. 28573–28593, 2018.

[55] P. Maes, "Artificial life meets entertainment: lifelike autonomous agents," *Communications of the ACM*, vol. 38, no. 11, pp. 108–114, 1995.

[56] P. Stone and M. Veloso, "Multiagent systems: A survey from a machine learning perspective," *Autonomous Robots*, vol. 8, no. 3, pp. 345–383, 2000.

[57] K. Rabuzin, M. Maleković, and M. Bača, "A survey of the properties of agents," *Journal of Information and Organizational Sciences*, vol. 30, no. 1, pp. 155–170, 2006.

[58] J. Ingham, "What is an agent," *Centre for Software Maintenance University of Durham*, 1997.

[59] R. Kowalczyk, "Intelligent Agent Technology Research." `https://www.swinburne.edu.au/ict/success/research-projects-and-grants/intelligent-agent/`, 2014.
[Online; accessed 06-April-2019].

[60] S. Wells, *Formal Dialectical Games in Multiagent Argumentation*.
PhD thesis, PhD thesis, University of Dundee, 2007.

[61] N. Maudet, S. Parsons, and I. Rahwan, "Argumentation in multi-agent systems: Context and recent developments," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 1–16, Springer, 2006.

[62] S. Ontanón and E. Plaza, "Arguments and counterexamples in case-based joint deliberation," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 36–53, Springer, 2006.

[63] D. Walton, C. Reed, and F. Macagno, *Argumentation schemes*.
Cambridge University Press, 2008.

[64] F. van Eemeren, R. Grootendorst, and F. S. Henkemans, *Fundamentals of Argumentation Theory a Handbook of Historical Backgrounds and Contemporary Developments*.
Lawrence Erlbaum Associates: Hillsdale NJ,USA, 1996.

[65] I. Rahwan, K. Larson, and F. A. Tohmé, "A characterisation of strategy-proofness for grounded argumentation semantics.," in *IJCAI*, pp. 251–256, Citeseer, 2009.

[66] P. Besnard and A. Hunter, *Elements of argumentation*, vol. 47.
MIT press Cambridge, 2008.

[67] G. R. Simari and R. P. Loui, "A mathematical treatment of defeasible reasoning and its implementation," *Artificial intelligence*, vol. 53, no. 2-3, pp. 125–157, 1992.

[68] L. Amgoud and H. Prade, "Using arguments for making and explaining decisions," *Artificial Intelligence*, vol. 173, no. 3-4, pp. 413–436, 2009.

[69] B. Bonet and H. Geffner, "Arguing for decisions: A qualitative model of decision making," in *Proceedings of the Twelfth international conference on Uncertainty in artificial intelligence*, pp. 98–105, Morgan Kaufmann Publishers Inc., 1996.

[70] L. Amgoud and P. Besnard, "Bridging the gap between abstract argumentation systems and logic," in *International Conference on Scalable Uncertainty Management*, pp. 12–27, Springer, 2009.

[71] L. Amgoud and C. Cayrol, "A reasoning model based on the production of acceptable arguments," *Annals of Mathematics and Artificial Intelligence*, vol. 34, no. 1-3, pp. 197–215, 2002.

[72] M. A. Falappa, G. Kern-Isberner, and G. R. Simari, "Belief revision and argumentation theory," in *Argumentation in artificial intelligence*, pp. 341–360, Springer, 2009.

[73] I. Rahwan and G. R. Simari, *Argumentation in artificial intelligence*, vol. 47. Springer, 2009.

[74] P. Besnard and A. Hunter, “Argumentation based on classical logic,” in *Argumentation in Artificial Intelligence*, pp. 133–152, Springer, 2009.

[75] L. Amgoud, “Argumentation for decision making,” in *Argumentation in artificial intelligence*, pp. 301–320, Springer, 2009.

[76] I. Rahwan and K. Larson, “Argumentation and game theory,” in *Argumentation in Artificial Intelligence*, pp. 321–339, Springer, 2009.

[77] P. J. McBurney, *Rational interaction*. PhD thesis, University of Liverpool, 2002.

[78] M.-P. Huget, *Communication in Multiagent Systems: Agent communication languages and conversation policies*, vol. 2650. Springer Science & Business Media, 2003.

[79] A. Kakas and P. Moraitis, “Argumentation based decision making for autonomous agents,” in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pp. 883–890, ACM, 2003.

[80] N. Oren, T. J. Norman, and A. Preece, “Subjective logic and arguing with evidence,” *Artificial Intelligence*, 2007.

[81] R. H. Guttman, A. G. Moukas, and P. Maes, “Agent-mediated electronic commerce: A survey,” *The Knowledge Engineering Review*, vol. 13, no. 2, pp. 147–159, 1998.

[82] S. Modgil, F. Toni, F. Bex, I. Bratko, C. I. Chesnevar, W. Dvořák, M. A. Falappa, X. Fan, S. A. Gaggl, A. J. García, *et al.*, “The added value of argumentation,” in *Agreement technologies*, pp. 357–403, Springer, 2013.

[83] G. Boella, J. Hulstijn, and L. Van Der Torre, “A logic of abstract argumentation,” in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 29–41, Springer, 2005.

[84] P. Baroni, M. Caminada, and M. Giacomin, “An introduction to argumentation semantics,” *The knowledge engineering review*, vol. 26, no. 4, pp. 365–410, 2011.

[85] G. A. Vreeswik and H. Prakken, “Credulous and sceptical argument games for preferred semantics,” in *European Workshop on Logics in Artificial Intelligence*, pp. 239–253, Springer, 2000.

[86] S. Woltran and G. Brewka, “A Short Introduction to Abstract Argumentation Frameworks.” `http://www.informatik.uni-leipzig.de/~brewka/KRlecture/AF.pdf`, 2009.
[Online; accessed 30-January-2020].

[87] P. E. Dunne, A. Hunter, P. McBurney, S. Parsons, and M. Wooldridge, “Weighted argument systems: Basic definitions, algorithms, and complexity results,” *Artificial Intelligence*, vol. 175, no. 2, pp. 457–486, 2011.

[88] I. Rahwan and F. Tohmé, “Collective argument evaluation as judgement aggregation,” in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pp. 417–424, International Foundation for Autonomous Agents and Multiagent Systems, 2010.

[89] S. Gaggl, “Introduction to Formal Argumentation.” `https://iccl.inf.tu-dresden.de/w/images/0/0f/ICCL_SS2017_Argumentation_slides1.pdf`, 2017.
[Online; accessed 30-January-2020].

[90] G. Vreeswijk, “Defeasible dialectics: A controversy-oriented approach towards defeasible argumentation,” *Journal of Logic and Computation*, vol. 3, no. 3, pp. 317–334, 1993.

[91] H. Prakken and G. Sartor, “Argument-based extended logic programming with defeasible priorities,” *Journal of applied non-classical logics*, vol. 7, no. 1-2, pp. 25–75, 1997.

[92] H. Prakken, “Dialectical proof theory for defeasible argumentation with defeasible priorities (preliminary report),” in *ModelAge Workshop on Formal Models of Agents*, pp. 202–215, Springer, 1999.

[93] P. E. Dunne and T. J. Bench-Capon, “Two party immediate response disputes: Properties and efficiency,” *Artificial Intelligence*, vol. 149, no. 2, pp. 221–250, 2003.

[94] H. Prakken, “Coherence and flexibility in dialogue games for argumentation,” *J. Log. Comput.*, vol. 15, no. 6, pp. 1009–1040, 2005.

[95] T. Yuan, D. Moore, C. Reed, A. Ravenscroft, and N. Maudet, “Informal logic dialogue games in human–computer dialogue,” *The Knowledge Engineering Review*, vol. 26, no. 2, pp. 159–174, 2011.

[96] P. S. Kośmicki, “A platform for the evaluation of automated argumentation strategies,” in *International Conference on Rough Sets and Current Trends in Computing*, pp. 494–503, Springer, 2010.

[97] J. L. Austin, *How to do things with words*.
Oxford University Press, 1975.

[98] T. Yuan, D. Moore, and A. Grierson, “Assessing debate strategies via computational agents,” *Argument and Computation*, vol. 1, no. 3, pp. 215–248, 2010.

[99] H. Prakken, “On dialogue systems with speech acts, arguments, and counterarguments,” in *European Workshop on Logics in Artificial Intelligence*, pp. 224–238, Springer, 2000.

[100] T. F. Gordon, “The pleadings game,” in *The Pleadings Game*, pp. 109–169, Springer, 1995.

[101] H. Prakken and G. Sartor, “Law and logic: A review from an argumentation perspective,” *Artificial Intelligence*, vol. 227, pp. 214–245, 2015.

[102] T. J. Bench-Capon, “Hypo’s legacy: introduction to the virtual special issue,” *Artificial Intelligence and Law*, vol. 25, no. 2, pp. 205–250, 2017.

[103] J. C. Hage, R. Leenes, and A. R. Lodder, “Hard cases: a procedural approach,” *Artificial intelligence and law*, vol. 2, no. 2, pp. 113–167, 1993.

[104] F. Grasso, A. Cawsey, and R. Jones, “Dialectical argumentation to solve conflicts in advice giving: a case study in the promotion of healthy nutrition,” *International Journal of Human-Computer Studies*, vol. 53, no. 6, pp. 1077–1115, 2000.

[105] F. Bex, J. Lawrence, M. Snaith, and C. Reed, “Implementing the argument web,” *Communications of the ACM*, vol. 56, no. 10, pp. 66–73, 2013.

[106] M. Snaith and C. Reed, “Toast: Online aspic+ implementation.,” *COMMA*, vol. 245, pp. 509–510, 2012.

[107] L. Amgoud, N. Maudet, and S. Parsons, “Modelling dialogues using argumentation,” in *MultiAgent Systems, 2000. Proceedings. Fourth International Conference on*, pp. 31–38, IEEE, 2000.

[108] K. Atkinson, T. Bench-Capon, and P. Mcburney, “A dialogue game protocol for multi-agent argument over proposals for action,” *Autonomous Agents and Multi-Agent Systems*, vol. 11, no. 2, pp. 153–171, 2005.

[109] E. M. Kok, J.-J. C. Meyer, H. Prakken, and G. A. Vreeswijk, “A formal argumentation framework for deliberation dialogues,” in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 31–48, Springer, 2010.

[110] P. McBurney, D. Hitchcock, and S. Parsons, “The eightfold way of deliberation dialogue,” *International Journal of Intelligent Systems*, vol. 22, no. 1, pp. 95–132, 2007.

[111] S. Parsons, C. Sierra, and N. Jennings, “Agents that reason and negotiate by arguing,” *Journal of Logic and computation*, vol. 8, no. 3, pp. 261–292, 1998.

[112] S. Parsons and N. R. Jennings, “Negotiation through argumentation- a preliminary report,” in *Proceedings of the 2nd international conference On multi agent systems*, pp. 267–274, 1996.

[113] N. Maudet and D. Moore, "Dialogue games as dialogue models for interacting with, and via, computers," *Informal Logic*, vol. 21, no. 3, 2001.

[114] P. McBurney and S. Parsons, "Dialogue games in multi-agent systems," *Informal Logic*, vol. 22, no. 3, 2002.

[115] I. Angelelli, "The techniques of disputation in the history of logic," *The Journal of Philosophy*, vol. 67, no. 20, pp. 800–815, 1970.

[116] C. L. Hamblin, "Fallacies," *Theoria*, 1970.

[117] C. L. Hamblin, "Mathematical models of dialogue1," *Theoria*, vol. 37, no. 2, pp. 130–155, 1971.

[118] J. Mackenzie, "Four dialogue systems," *Studia logica*, vol. 49, no. 4, pp. 567–583, 1990.

[119] L. Amgoud, S. Parsons, and N. Maudet, "Arguments, dialogue, and negotiation," *a a*, vol. 10, no. 11, p. 02, 2000.

[120] D. Moore and D. Hobbes, "Computational uses of philosophical dialogue theories," *Informal Logic*, vol. 18, no. 2, 1996.

[121] C. Reed and S. Wells, "Dialogical argument as an interface to complex debates," *IEEE Intelligent Systems*, vol. 22, no. 6, pp. 60–65, 2007.

[122] R. P. Loui, "Process and policy: Resource-bounded nondemonstrative reasoning," *Computational intelligence*, vol. 14, no. 1, pp. 1–38, 1998.

[123] T. J. M. Bench-Capon, T. Geldard, and P. H. Leng, "A method for the computational modelling of dialectical argument with dialogue games," *Artificial Intelligence and Law*, vol. 8, no. 2, pp. 233–254, 2000.

[124] P. McBurney and S. Parsons, "Games that agents play: A formal framework for dialogues between autonomous agents," *Journal of logic, language and information*, vol. 11, no. 3, pp. 315–334, 2002.

[125] P. McBurney, S. Parsons, and M. Wooldridge, "Desiderata for agent argumentation protocols," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1*, pp. 402–409, 2002.

[126] S. Parsons, M. Wooldridge, and L. Amgoud, "On the outcomes of formal inter-agent dialogues," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pp. 616–623, 2003.

[127] H. Prakken, "Modelling reasoning about evidence in legal procedure," in *Proceedings of the 8th international conference on Artificial intelligence and law*, pp. 119–128, 2001.

[128] H. Prakken, C. Reed, and D. Walton, "Dialogues about the burden of proof," in *Proceedings of the 10th international conference on Artificial intelligence and law*, pp. 115–124, 2005.

[129] R. Medellin-Gasque, K. Atkinson, and T. Bench-Capon, "Dialogue game protocol for co-operative plan proposals," tech. rep., Technical Report ULCS-12-003, Department of Computer Science, University of Liverpool, 2012.

[130] N. Oren, T. J. Norman, and A. Preece, "Information based argumentation heuristics," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 161–174, Springer, 2006.

[131] H. Prakken, "Relating protocols for dynamic dispute with logics for defeasible argumentation," *Synthese*, vol. 127, no. 1, pp. 187–219, 2001.

[132] T. J. Bench-Capon, "Specification and implementation of toulmin dialogue game," in *Proceedings of JURIX*, vol. 98, pp. 5–20, 1998.

[133] S. Toulmin, "The uses of argument," *Cambridge University Press, UK*, 1958.

[134] J. Schneider, T. Groza, and A. Passant, "A review of argumentation for the social semantic web," *Semantic Web*, vol. 4, no. 2, pp. 159–218, 2013.

[135] A. Ravenscroft and R. M. Pilkington, "Investigation by design: developing dialogue models to support reasoning and conceptual change," *International Journal of Artificial Intelligence in Education*, vol. 11, no. 1, pp. 273–298, 2000.

[136] A. Ravenscroft and M. P. Matheson, "Carpe diem: Models and methodologies for designing engaging and interactive e-learning discourse," in *Proceedings*

IEEE international conference on advanced learning technologies, pp. 74–77, IEEE, 2001.

[137] A. Ravenscroft and M. P. Matheson, "Developing and evaluating dialogue games for collaborative e–learning," *Journal of Computer Assisted Learning*, vol. 18, no. 1, pp. 93–101, 2002.

[138] L. Rourke, T. Anderson, D. R. Garrison, and W. Archer, "Methodological issues in the content analysis of computer conference transcripts," *International journal of artificial intelligence in education (IJAIED)*, vol. 12, pp. 8–22, 2001.

[139] R. A. Girle, "Commands in dialogue logic," in *International Conference on Formal and Applied Practical Reasoning*, pp. 246–260, Springer, 1996.

[140] R. A. Girle, "Knowledge organized and disorganized," 1994.

[141] R. A. Girle, "Dialogue and entrenchment," in *Proceedings Of The 6th Florida Artificial Intelligence Research Symposium*, pp. 185–189, 1993.

[142] S. Wells and C. Reed, "Cumulativeness in dialectical games," in *Proceedings of the 6th International Conference on Argumentation (ISSA 2006)*, 2006.

[143] R. A. Girle, "Proof and dialogue in aristotle," *Argumentation*, vol. 30, no. 3, pp. 289–316, 2016.

[144] D. N. Walton, *Logical dialogue-gamES*.
University Press of America, Lanham, Maryland, 1984.

[145] T. Yuan, D. Moore, and A. Grierson, "Computational agents as a test-bed to study the philosophical dialogue model" de": A development of mackenzie's dc," *Informal Logic*, vol. 23, no. 3, 2003.

[146] T. Yuan, D. Moore, and A. Grierson, "Educational human-computer debate: A computational dialectics approach," in *Proceedings of the Workshop on Computational Models of Natural Argument*, 2002.

[147] C. L. Hamblin, "Mathematical models of dialogue 1," *Theoria*, vol. 37, no. 2, pp. 130–155, 1971.

[148] C. D. Emele, T. J. Norman, and S. Parsons, "Argumentation strategies for plan resourcing," in *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 3*, pp. 913–920, International Foundation for Autonomous Agents and Multiagent Systems, 2011.

[149] A. Kakas, N. Maudet, and P. Moraitis, "Layered strategies and protocols for argumentation-based agent interaction," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 64–77, Springer, 2004.

[150] I. Rahwan, "Argumentation among agents," *Multiagent Systems*, p. 177, 2013.

[151] R. M. Pilkington, J. R. Hartley, D. Hintze, and D. J. Moore, "Learning to argue and arguing to learn: An interface for computer-based dialogue games," *Journal of Interactive Learning Research*, vol. 3, no. 3, p. 275, 1992.

[152] T. L. Turocy and B. v. Stengel, "Game theory," *CDAM Research Report (The draft of an introductory survey of game theory, prepared for the Encyclopedia of Information Systems, Academic Press, to appear in 2002)*, 2001.

[153] B. John A., "Knowledge representation." University Lecture, University of Birmingham, Department of Computer Science, 2005.

[154] T. F. Gordon, H. Prakken, and D. Walton, "The carneades model of argument and burden of proof," *Artificial Intelligence*, vol. 171, no. 10-15, pp. 875–896, 2007.

[155] O. Scheuer, F. Loll, N. Pinkwart, and B. M. McLaren, "Computer-supported argumentation: A review of the state of the art," *International Journal of Computer-supported collaborative learning*, vol. 5, no. 1, pp. 43–102, 2010.

[156] B. Verheij, "ArguMed - a template-based argument mediation system for lawyers," *JC Hage et al*, pp. 113–130, 1998.

[157] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, pp. 63–71, Springer, 2003.

[158] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques*.
Morgan Kaufmann, 2016.

[159] Z. Ghahramani, “Unsupervised learning,” in *Advanced lectures on machine learning*, pp. 72–112, Springer, 2004.

[160] D. J. MacKay and D. J. Mac Kay, *Information theory, inference and learning algorithms*.
Cambridge university press, 2003.

[161] C. M. Bishop, “Pattern recognition and machine learning (information science and statistics) springer-verlag new york,” *Inc. Secaucus, NJ, USA*, 2006.

[162] M. van Otterlo and M. Wiering, “Reinforcement learning and markov decision processes,” in *Reinforcement Learning*, pp. 3–42, Springer, 2012.

[163] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T. Graepel, “A unified game-theoretic approach to multiagent reinforcement learning,” in *Advances in Neural Information Processing Systems*, pp. 4190–4203, 2017.

[164] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *International conference on machine learning*, pp. 1928–1937, 2016.

[165] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, p. 529, 2015.

[166] M. I. Jordan and T. M. Mitchell, “Machine learning: Trends, perspectives, and prospects,” *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[167] H. Jia, *Machine learning to argue*.
MSc dissertation, Department of Computer Science, University of York, 2015.

[168] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*.
MIT press, 2018.

[169] M. L. Puterman, *Markov decision processes: discrete stochastic dynamic programming*.
John Wiley & Sons, 2014.

[170] C. Boutilier, T. Dean, and S. Hanks, “Decision-theoretic planning: Structural assumptions and computational leverage,” *Journal of Artificial Intelligence Research*, vol. 11, pp. 1–94, 1999.

[171] C. Boutilier, “Knowledge representation for stochastic decision processes,” in *Artificial intelligence today*, pp. 111–152, Springer, 1999.

[172] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *Journal of artificial intelligence research*, vol. 4, pp. 237–285, 1996.

[173] M. Grounds and D. Kudenko, “Parallel reinforcement learning with linear function approximation,” in *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, pp. 60–74, Springer, 2005.

[174] M. Grounds and D. Kudenko, “Combining reinforcement learning with symbolic planning,” in *Adaptive Agents and Multi-Agent Systems III. Adaptation and Multi-Agent Learning*, pp. 75–86, Springer, 2005.

[175] C. J. C. H. Watkins, “Learning from delayed rewards,” 1989.

[176] L. Torrey and J. Shavlik, “Transfer learning,” in *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*, pp. 242–264, IGI Global, 2010.

[177] A. Argyriou, A. Maurer, and M. Pontil, “An algorithm for transfer learning in a heterogeneous environment,” in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 71–85, Springer, 2008.

[178] M. E. Taylor and P. Stone, “Transfer learning for reinforcement learning domains: A survey,” *Journal of Machine Learning Research*, vol. 10, no. Jul, pp. 1633–1685, 2009.

[179] G. Tesauro, “Td-gammon, a self-teaching backgammon program, achieves master-level play,” *Neural computation*, vol. 6, no. 2, pp. 215–219, 1994.

[180] H. J. Kim, M. I. Jordan, S. Sastry, and A. Y. Ng, “Autonomous helicopter flight via reinforcement learning,” in *Advances in neural information processing systems*, pp. 799–806, 2004.

[181] P. Stone, R. S. Sutton, and G. Kuhlmann, “Reinforcement learning for robocup soccer keepaway,” *Adaptive Behavior*, vol. 13, no. 3, pp. 165–188, 2005.

[182] K. Weiss, T. M. Khoshgoftaar, and D. Wang, “A survey of transfer learning,” *Journal of Big data*, vol. 3, no. 1, p. 9, 2016.

[183] R. Glatt, F. L. Da Silva, R. A. da Costa Bianchi, and A. H. R. Costa, “Decaf: Deep case-based policy inference for knowledge transfer in reinforcement learning,” *Expert Systems with Applications*, p. 113420, 2020.

[184] M. E. Taylor and P. Stone, “Cross-domain transfer for reinforcement learning,” in *Proceedings of the 24th international conference on Machine learning*, pp. 879–886, ACM, 2007.

[185] R. A. Bianchi, L. A. Celiberto Jr, P. E. Santos, J. P. Matsuura, and R. L. de Mantaras, “Transferring knowledge as heuristics in reinforcement learning: A case-based approach,” *Artificial Intelligence*, vol. 226, pp. 102–121, 2015.

[186] M. E. Taylor, “Assisting transfer-enabled machine learning algorithms: Leveraging human knowledge for curriculum design.,” in *AAAI Spring Symposium: Agents that Learn from Human Teachers*, pp. 141–143, 2009.

[187] M. E. Taylor and P. Stone, “An introduction to intertask transfer for reinforcement learning,” *Ai Magazine*, vol. 32, no. 1, pp. 15–15, 2011.

[188] M. E. Taylor, P. Stone, and Y. Liu, “Transfer learning via inter-task mappings for temporal difference learning,” *Journal of Machine Learning Research*, vol. 8, no. Sep, pp. 2125–2167, 2007.

[189] K. Georgila and D. Traum, “Reinforcement learning of argumentation dialogue policies in negotiation,” in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[190] T. Hiraoka, K. Georgila, E. Nouri, D. Traum, and S. Nakamura, “Reinforcement learning in multi-party trading dialog,” in *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, p. 32, 2015.

[191] P. Paruchuri, N. Chakraborty, R. Zivan, K. Sycara, M. Dudik, and G. Gordon, “Pomdp based negotiation modeling,” in *Proc. of the IJCAI Workshop on Modeling Intercultural Collaboration and Negotiation (MICON)*, 2009.

[192] I. Efstathiou and O. Lemon, “Learning non-cooperative dialogue behaviours,” in *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pp. 60–68, 2014.

[193] K. Georgila, C. Nelson, and D. R. Traum, “Single-agent vs. multi-agent techniques for concurrent reinforcement learning of negotiation dialogue policies,” in *ACL (1)*, pp. 500–510, 2014.

[194] M. Gavsić, C. Breslin, M. Henderson, D. Kim, M. Szummer, B. Thomson, P. Tsiakoulis, and S. Young, “On-line policy optimisation of bayesian spoken dialogue systems via human interaction,” in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pp. 8367–8371, IEEE, 2013.

[195] C. Chakrabarti and G. F. Luger, “Artificial conversations for customer service chatter bots: Architecture, algorithms, and evaluation metrics,” *Expert Systems with Applications*, vol. 42, no. 20, pp. 6878–6897, 2015.

[196] A. Schmitt and W. Minker, *Towards Adaptive Spoken Dialog Systems*. Springer Science & Business Media, 2012.

[197] M. Wiering and M. Van Otterlo, “Reinforcement learning,” *Adaptation, learning, and optimization*, vol. 12, p. 51, 2012.

[198] S. Ontañón and E. Plaza, “An argumentation-based framework for deliberation in multi-agent systems,” in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 178–196, Springer, 2007.

[199] M. Wardeh, T. Bench-Capon, and F. Coenen, “Arguing from experience using multiple groups of agents,” *Argument and Computation*, vol. 2, no. 1, pp. 51–76, 2011.

[200] E. Plaza, B. D. Agudo, and E. Golobardes, “Mid-cbr: An integrative framework for developing case-based reasoning systems tin2006-15140-c03,”

[201] N. Oren, T. J. Norman, and A. Preece, "Loose lips sink ships: A heuristic for argumentation," in *Proc. of the 3rd International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2006)*, pp. 121–134, Citeseer, 2006.

[202] J. Devereux and C. Reed, "Strategic argumentation in rigorous persuasion dialogue," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 94–113, Springer, 2009.

[203] A. Papangelis and K. Georgila, "Reinforcement learning of multi-issue negotiation dialogue policies," in *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pp. 154–158, 2015.

[204] R. Laroche and A. Genevay, "The negotiation dialogue game," in *Dialogues with Social Robots*, pp. 403–410, Springer, 2017.

[205] E. Nouri, K. Georgila, and D. Traum, "A cultural decision-making model for negotiation based on inverse reinforcement learning," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 34, 2012.

[206] D. N. Walton, *The new dialectic: Conversational contexts of argument*. University of Toronto Press, 1998.

[207] L. Amgoud and N. Maudet, "Strategical considerations for argumentative agents (preliminary report).," in *NMR*, pp. 399–407, 2002.

[208] C. Perelman and L. Olbrechts-Tyteca, "The new rhetoric: a treatise on argumentation, trans," *John Wilkinson and Purcell Weaver (Notre Dame, IN: University of Notre Dame Press, 1969)*, vol. 190, pp. 411–12, 1969.

[209] K. Freeman and A. M. Farley, "A model of argumentation and its application to legal reasoning," *Artificial Intelligence and Law*, vol. 4, no. 3-4, pp. 163–197, 1996.

[210] N. Oren, T. J. Norman, and A. Preece, "Arguing with confidential information," in *ECAI 2006: 17th European Conference on Artificial Intelligence, August 29-September 1, 2006, Riva Del Garda, Italy: Including Prestigious Applications of Intelligent Systems (PAIS 2006): Proceedings*, vol. 141, p. 280, IOS Press, 2006.

[211] H. Cuayáhuitl, S. Keizer, and O. Lemon, “Strategic dialogue management via deep reinforcement learning,” *arXiv preprint arXiv:1511.08099*, 2015.

[212] H. Prakken, “Abstract Argumentation Proof theory.” `http://www.cs.uu.nl/docs/vakken/mcarg/`, 2016.
[Online; accessed 01-February-2020].

[213] R. A. Watson and E. Szathmáry, “How can evolution learn?,” *Trends in ecology & evolution*, vol. 31, no. 2, pp. 147–157, 2016.

[214] S. Van Otterloo, “The value of privacy: optimal strategies for privacy minded agents,” in *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, pp. 1015–1022, 2005.

[215] P. Paruchuri, M. Tambe, F. Ordóñez, and S. Kraus, “Security in multiagent systems by policy randomization,” in *Proceedings of the Fifth International Joint Conference on Autonomous Agents and Multiagent Systems*, AAMAS ‚Äô06, (New York, NY, USA), p. 273‚Äì280, Association for Computing Machinery, 2006.

[216] I. Rahwan, P. McBurney, and L. Sonenberg, “Towards a theory of negotiation strategy (a preliminary report),” in *Proceedings of the 5th Workshop on Game Theoretic and Decision Theoretic Agents (GTDT-2003)*, pp. 73–80, 2003.

[217] M. Wardeh, T. Bench-Capon, and F. Coenen, “Padua protocol: Strategies and tactics,” in *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, pp. 465–476, Springer, 2007.

[218] L. Amgoud and S. Parsons, “Agent dialogues with conflicting preferences,” in *International Workshop on Agent Theories, Architectures, and Languages*, pp. 190–205, Springer, 2001.

[219] D. Walton, “Argumentation schemes for presumptive reasoning lawrence erlbaum associates mahwah,” *New Jersey*, 1996.

[220] N. C. Karunatillake, N. R. Jennings, I. Rahwan, and T. J. Norman, "Argument-based negotiation in a social context," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 104–121, Springer, 2005.

[221] D. Walton, *Argumentation schemes for presumptive reasoning*. Routledge, 2013.

[222] R. Koppula, "Exploration vs. Exploitation in Reinforcement Learning." `https://www.manifold.ai/exploration-vs-exploitation-in-reinforcement-learning`, 2020. [Online; accessed 02-March-2020].

[223] B. Schneier, "Applied cryptography: Protocols, algorithms, and source code in c," 1993.

[224] T. L. van der Weide, F. Dignum, J.-J. C. Meyer, H. Prakken, and G. Vreeswijk, "Multi-criteria argument selection in persuasion dialogues," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 136–153, Springer, 2011.

[225] L. Amgoud and F. D. De Saint Cyr, "Measures for persuasion dialogs: A preliminary investigation," *Frontiers in Artificial Intelligence and Applications*, vol. 172, p. 13, 2008.

[226] L. Amgoud and F. D. de Saint-Cyr, "On the quality of persuasion dialogs," *Studies in Logic, Grammer and Rheroric*, vol. 12, no. 36, pp. 69–98, 2011.

[227] J. Bentahar, B. Moulin, J.-J. C. Meyer, and B. Chaib-draa, "A modal semantics for an argumentation-based pragmatics for agent communication," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 44–63, Springer, 2004.

[228] J. Bentahar, B. Moulin, and B. Chaib-draa, "Specifying and implementing a persuasion dialogue game using commitments and arguments," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 130–148, Springer, 2004.

[229] M. J. Marcos, M. A. Falappa, and G. R. Simari, "Dynamic argumentation in abstract dialogue frameworks," in *International Workshop on Argumentation in Multi-Agent Systems*, pp. 228–247, Springer, 2010.

[230] D. Walton, "Dialectical relevance in persuasion dialogue," *Informal Logic*, vol. 19, no. 2, 1999.

[231] T. L. van der Weide, *Arguing to motivate decisions*.
PhD thesis, Utrecht University, 2011.

[232] L. Carlson, "Dialogue games: An approach to discourse anaphora," *Dordrecht: Reidel*, 1983.

[233] C. M. Mark A. Gluck, Eduardo Mercado, *Learning and Memory: From Brain to Behavior (International Edition)*.
Worth Publishers, 2013.

[234] S. Bocconi, A. Chioccariello, G. Dettori, A. Ferrari, K. Engelhardt, P. Kampylis, and Y. Punie, "Developing computational thinking in compulsory education," *European Commission, JRC Science for Policy Report*, 2016.

[235] "Computational thinking, how do we think about problems so that computers can help?." `https://community.computingatschool.org.uk/files/8221/original.pdf`.
Accessed: 2019-07-30.

[236] S. Thrun and L. Pratt, *Learning to learn*.
Springer Science & Business Media, 2012.

[237] J. Lu, V. Behbood, P. Hao, H. Zuo, S. Xue, and G. Zhang, "Transfer learning using computational intelligence: a survey," *Knowledge-Based Systems*, vol. 80, pp. 14–23, 2015.

[238] W. Dan, "What does the public think about the NHS?." `https://www.kingsfund.org.uk/publications/what-does-public-think-about-nhs`, 2017.
[Online; accessed 02-March-2020].

[239] ofcom, "Public opinion on the BBC and BBC News." https://www.ofcom.org.uk/__data/assets/pdf_file/0014/58001/bbc-annex2.pdf, 2011. [Online; accessed 02-March-2020].

[240] Bond, "How much does the UK public trust the media?." https://www.bond.org.uk/news/2019/05/how-much-does-the-uk-public-trust-the-media, 2019. [Online; accessed 05-March-2020].

[241] J. Research, "News Consumption in the UK: 2018." https://www.ofcom.org.uk/__data/assets/pdf_file/0024/116529/news-consumption-2018.pdf, 2018. [Online; accessed 10-March-2020].

[242] F. Bex, H. Prakken, C. Reed, and D. Walton, "Towards a formal account of reasoning about evidence: argumentation schemes and generalisations," *Artificial Intelligence and Law*, vol. 11, no. 2-3, pp. 125–165, 2003.

[243] M. E. Taylor, N. K. Jong, and P. Stone, "Transferring instances for model-based reinforcement learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 488–505, Springer, 2008.

[244] A. Lazaric, "Transfer in reinforcement learning: a framework and a survey," in *Reinforcement Learning*, pp. 143–173, Springer, 2012.

[245] A. L. Bazzan, "Opportunities for multiagent systems and multiagent reinforcement learning in traffic control," *Autonomous Agents and Multi-Agent Systems*, vol. 18, no. 3, p. 342, 2009.

[246] L. S. Marcolino, A. X. Jiang, and M. Tambe, "Multi-agent team formation: diversity beats strength?," in *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[247] N. J. van Eck and M. van Wezel, "Application of reinforcement learning to the game of othello," *Computers & Operations Research*, vol. 35, no. 6, pp. 1999–2017, 2008.

[248] M. Grzes, *Improving exploration in reinforcement learning through domain knowledge and parameter analysis*.
PhD thesis, University of York, 2010.

[249] N. Papahristou and I. Refanidis, "Training neural networks to play backgammon variants using reinforcement learning," in *European Conference on the Applications of Evolutionary Computation*, pp. 113–122, Springer, 2011.

[250] M. Tokic and G. Palm, "Adaptive exploration using stochastic neurons," in *International Conference on Artificial Neural Networks*, pp. 42–49, Springer, 2012.

[251] L. Al-Abdulkarim, *Representation of case law for argumentative reasoning.*
PhD thesis, University of Liverpool, 2017.