Reinforcement Learning for Dialogue Game Based Argumentation

Sultan Alahmari, Tommy Yuan, and Daniel Kudenko

University of York, Department of Computer Science, Deramore Lane York YO10 5GH, UK
smsa500, tommy.yuan and daniel.kudenko@york.ac.uk

Abstract. Communication between agents is an important feature for intelligent systems. One type of communication is the argumentative dialogue where both agents aim to convert each others' point of view. Argumentative reinforcement learning (RL) agents can learn from experience via trial and error. For an agent learning to argue with other agents, it would be useful to discover some useful argument patterns i.e. state and action pairs, which will enable a RL agent to transfer what has been learned from one domain to another. A dialogue model is needed to manage the evolving dialogue between agents. In this paper we adopt the DE dialogue game and construct an RL agent that can win a debate with the minimum number of moves. The result is promising so far and this enables us to move forward to transfer learning in order to generalise our approach. It is anticipated that our research will contribute to the design and implementation of argumentative learning agents.

Keywords: Multi-agent systems · Argumentation · Dialogue system · Dialogue model· DE dialogue game · Reinforcement learning

CEUR-WS.org/Vol-2346/paper3.pdf

1 Introduction

Recently, research and applications in artificial intelligence and machine learning have been increasing rapidly. Agents or multi-agents are able to via learning from the data [1]- [4]. Reinforcement learning (RL) is a machine learning approach where an agent can learn to map an action to a state in an environment in order to maximise the cumulative reward [11]. In our previous work [1], [2], we have built an RL agent that is able to argue against baseline agents using abstract argumentation systems framework [5]- [13] where the argument game was adopted for reasons of simplicity and flexibility [1]. The results, as reported in [1], are positive when an agent learns in a single argument graph and improves its performance over time. However, when an agent applies what has been learned to a different graph, the result is less encouraging. The difficulty with the abstract argument representation is that argument patterns, i.e. state action pairs are hard to be learned without referencing to the internal argument structure. This motivate us to move to propositional-logic based representation and a richer dialogue model [3], where argument patterns such as argument schemes and source of support, can be learned. An influential logic-based dialogue model - the DE model [16], [14] is used to manage the evolving dialogue.

The remainder of this paper is organised as follows. We will introduce the DE dialogue model in Section 2. Next, we discuss the design of the RL agent and baseline agents in Section 3. Section 4 provides the details for the experiments and discusses the results. Section 5 concludes the paper and give pointers for our intended future work.

2 The "DE" Dialogue Model

The DE dialogue model [16]- [19] was developed based on [9], which is a further version of Mackenzie's DC system [8]. The DE model facilities agents to make moves in an evolving dialogue. The DE model has five different move types are quoted from [14, 16, 19] as follows:

1. *Assertion*: The content of an assertion is a statement P, Q etc. or the truth-functional compounds of statements: $\neg P$, If P then Q, $P \wedge Q$.
2. *Questions*: The question of the statement P is *"Is it the case that P?"*
3. *Challenges*: The challenge of the statement P is *"Why is it supposed that P?"* (or briefly *"Why P?"*).
4. *Withdrawals*: The withdrawal of the statement P is *"no commitment P"*.
5. *Resolution demands*:The resolution demand of the statement P is *"resolve whether P"*.

The DE model specifies one publicly inspectable commitment store (henceforth referred as CS) for each player. A commitment store contains statements that have been stated or accepted by a speaker. A commitment store has an assertion list which holds statements which an agent has explicitly stated and a concession list which contains statements have been implicitly accepted [16]. The commitment rules below are taken from [14, 16, 19]:

1. *Initial commitment,*CR_0: The initial commitment of each participant is null.
2. *Withdrawals,*CR_W: After the withdrawal of P, the statement P is not included in the move.
3. *Statements,*CR_S:After a statement P, unless the preceding event was a challenge, P is included in the move maker's assertion list and the dialogue partner's concession list, and $\neg P$ will be removed from the move maker's concession list if it is there.
4. *Defence,*CR_{YS}:After a statement P, if the preceding event was Why Q?, P and If P then Q are included in the move maker's assertion list and the dialogue partner's concession list, and $\neg P$ and $\neg$(If P then Q) are removed from the move maker's concession list if they are there.
5. *Challenges,*CR_Y: A challenge of P results in P being removed from the store of the move maker's if it is there.

Dialogue rules that an agent must follow during the dialogue are stated in [14,16,19] as follows:

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

2. $R_{REPSTAT}$: Mutual commitment may not be asserted until answering the question or challenge.
3. R_{QUEST}: The possible answers to 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 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.

There are several advantages for the adoption of the DE model. First, the model leaves enough room for strategic formation, and strategy is essential for an agent to make high quality dialogue contributions. Second, computational agents adopting the DE model have been built with hard coded heuristic strategies and the model has shown advantage over others due to its computational tractability and simple dialogue rules (Yuan et al. 2007 [15], 2008 [16], 2011 [18]). Finally, the DE model is built upon propositional logic. In contrast to abstract argument game, the model is more sophisticated and richer where the dialogue state can be represented using commitment stores and different move types such as questions, statement and challenge. While in an abstract argument game, state representation is restricted to node and arcs. It is our expectation that the DE game can facilitate fruitful learning experience for a computational agent. The design of the RL agent that operate on the DE model is discussed next.

3 Agent Design

The section provides the design details for the RL agent and the baseline agents that have been used to facilitate the DE dialogue.

3.1 Reinforcement Learning Agent

Reinforcement learning is a type of machine learning where an agent interacts with an environment by mapping a state with an action in order to maximise the cumulative reward [11]. Agent needs to explore a state action policy using trial and error. Our RL agent engages in a DE dialogue with other computational agent. The outcome of the the dialogue is one agent persuades the other to give up its original view point. To enable reinforcement learning, we need to design state, action and reward for the agent.

The RL agent needs to observe the environment then decides the kind of actions to undertake. In the DE dialogue model, an important state variable is the commitment store which records what an agent has said or implicitly accepted during the dialogue. Commitment stores are updated via agent actions through the commitment rules. A further state variable is the dialogue history which contains the dialogue situation an

agent is facing (e.g the previous move). Only the previous move is used as a state variable for reasons of simplicity. Since each participant has its own commitment store, we define dialogue state as: $(previousmove \cup CS1 \cup CS2)$ (where CS1 is commitment store for one dialogue participant and CS2 belongs to the dialogue partner) to maximise the possibility that a state is unique in any case.

An action is a move that the RL agent chooses from the available move types in the DE model, which are: *assert*, *question*, *challenge*, *withdraw* or *resolution demand* together with the move content which is a proposition or conjunct of propositions. The agent should map the state with an action which is called *policy* (π).

When a RL agent wins, it will receive a positive reward and a negative reward when loses. We would also like to make an agent to win with the minimum number of moves by using the following reward function formula:

$$R = 100 + \frac{W}{L}$$

where W is the number of moves in a first winning episode (the benchmark), whereas, L is the number of moves in the current episode. When the L is at the minimum, the reward will be increased. We use the following Q-learning algorithm:

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

We offer the immediate reward $r_{t+1} = -0.01$ to encourage the RL agent to choose the minimum moves to win the game.

Our RL agent adopts the knowledge base as shown in Fig. 1 [19].

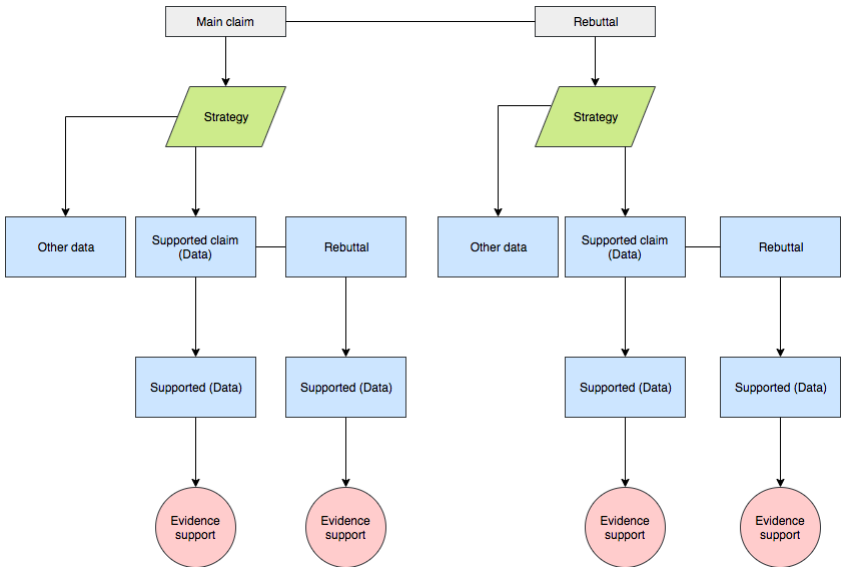

Fig. 1: System knowledge base [19], [6]

Moore argues that the knowledge base should provide statements that can be used to answer questions and support other statements, as well as providing statements to rebut other statements [9], [19]. We also add some features such as argument schemes and evidence source to the knowledge base with the expectation that an RL agent will be able to recognise the argument patterns via these features. An argument scheme (such as argue from consequence [12]) is represented in a parallelogram. An item of evidence (such as newspapers, magazines or scientific papers) is represented in a circle. For example, a RL agent could learn the reliability of a supporting evidence from the environment as seen in Fig. 2.

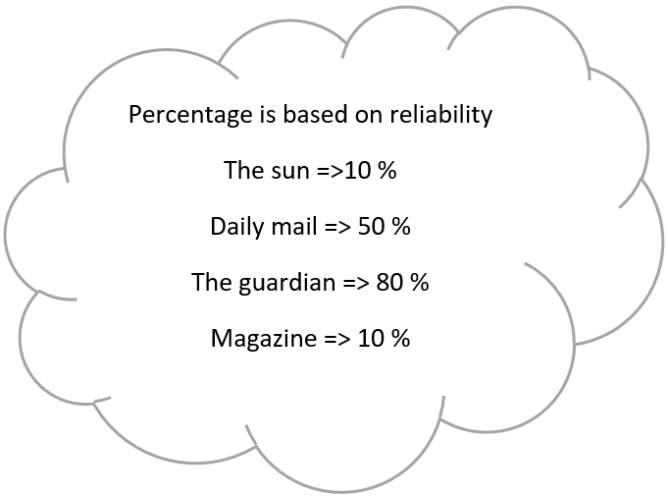

Fig. 2: Sources environment

3.2 Baseline Agents

To facilitate the DE dialogue with the RL agent discussed above, two baseline agents have been built, one with the fixed strategy and one with random strategy [17]. For the fixed strategy agent, the set of heuristics was based on three levels of decisions taken from [9]- [17]:

1. (1) Retain or change the current focus.
2. (2) Build own view or demolish the user's view.
3. (3) Select method to fulfil the objective set at levels (1) and (2) [17].

According to Yuan et al., in [17], they clarify all three levels as *" 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). Details of the level 3 strategies can be found from"*.

A random agent will choose a move randomly from the set of legally available moves in respect of the DE game rules.

4 Experiment and Result

To examine the performance of the RL agent discussed in the previous section, we have conducted a number of experiments. The baseline agents discussed above have been used to play with the RL agent for the evaluation purpose. The discussion topic is capital punishment and this is adapted from [19]. One participant needs to persuade the other to accept its point of view in order to resolve the conflict.

We set up the learning rate, discounted factor and epsilon as 0.9, 0.9 and 0.3 respectively. We set a game as 4,000 episodes. Starting from 0 episode where the Q-table values are zeros, the learning agent can behave like a random agent, we ran episode zero 30 times to avoid lucky choice. Hence, we need to trade off between exploration and exploitation to evaluate what has been learned. Exploitation is therefore every 100 episodes and epsilon reset to $\epsilon = 0$ and repeated the run 30 times. The performance was then measured by calculating how much reward each agent gained every 100 episodes, as we were interested in exploitation when the learning agent makes a decision based on the value in the Q-table.

Indeed, as the reward function mentioned above shows, we were interested in long term reward winning with the minimum number of moves. Therefore, we made our learning agent play two games, one against a fixed strategy agent and another against a random agent. The results look promising against both baseline agents, as can be seen in Figs. 3 and 4.

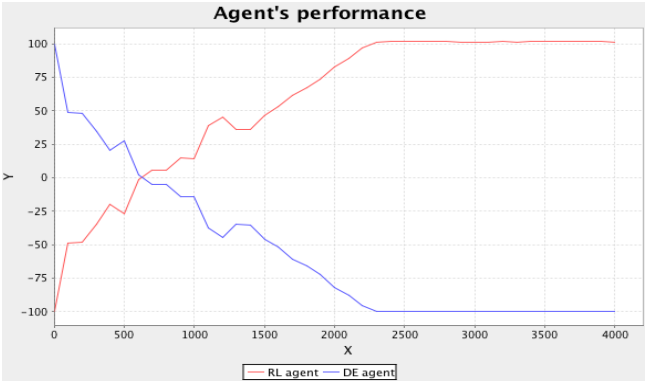

Fig. 3: RL agent against fixed strategy agent

Both figures illustrate that the learning curves significantly rise with the increase of the number of the episodes. This demonstrates that the agent is able to learn to argue against the baseline agents and win most of the games after converging.

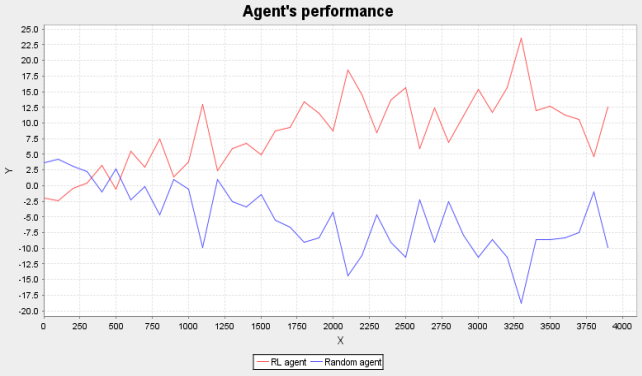

Fig. 4: RL agent against Random agent

Further, the reward function encourages the RL agent to win with the minimum number of moves against the fixed strategy agent as shown in Fig. 5, and likewise against random agent as shown in Fig. 6. Both figures demonstrate that the minimum number of moves converges after around 750 episodes.

Overall, the experiments show that the RL agent has been able to learn to argue against different baseline agents with different argument goals e.g. win or win with minimal number of moves.

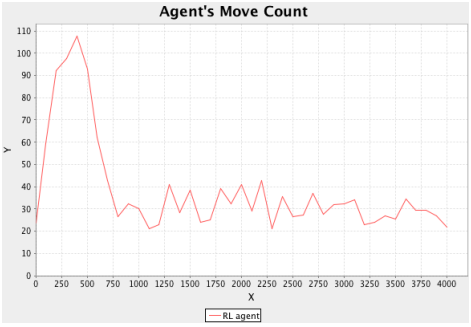

Fig. 5: Average moves count against Fixed strategy agent

5 Conclusion

We have used a DE dialogue model to engage our RL agent playing against baseline agents and learning how to argue. The results are promising given the RL agent's improved performance against the baseline agents. This encourages us to continue with the investigation of transfer learning in order to apply what has been learned from one

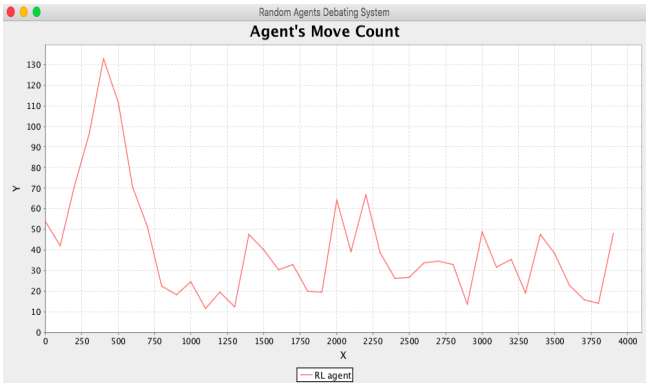

Fig. 6: Average moves count against Random agent

argument domain to another, e.g. Brexit. We are also planning to study the dialogue quality attributes such as coherence and fluency [10]- [7] and incorporate them in the reward function then study the consequences.

References

1. Sultan Alahmari, Tommy Yuan, and Daniel Kudenko. Reinforcement learning for abstract argumentation: Q-learning approach. In *Adaptive and Learning Agents workshop (at AAMAS 2017)*, 2017.
2. Sultan Alahmari, Tommy Yuan, and Daniel Kudenko. Reinforcement learning for argumentation: Describing a phd research. In *Proceedings of the 17th Workshop on Computational Models of Natural Argument (CMNA17)*, 2017.
3. Sultan Alahmari, Tommy Yuan, and Daniel Kudenko. Policy generalisation in reinforcement learning for abstract argumentation. In *Proceedings of the 18th Workshop on Computational Models of Natural Argument (CMNA18)*, 2018.
4. Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.
5. Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial intelligence*, 77(2):321–357, 1995.
6. Tim Kelly and Rob Weaver. The goal structuring notation–a safety argument notation. In *Proceedings of the dependable systems and networks 2004 workshop on assurance cases*, page 6. Citeseer, 2004.
7. Jiwei Li, Will Monroe, Alan Ritter, Michel Galley, Jianfeng Gao, and Dan Jurafsky. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541*, 2016.
8. Jim D Mackenzie. Question-begging in non-cumulative systems. *Journal of philosophical logic*, 8(1):117–133, 1979.
9. David John Moore. *Dialogue game theory for intelligent tutoring systems.* PhD thesis, Leeds Metropolitan University, 1993.
10. Henry Prakken. Coherence and flexibility in dialogue games for argumentation. *Journal of logic and computation*, 15(6):1009–1040, 2005.

11. Richard S Sutton and Andrew G Barto. *Introduction to reinforcement learning*, volume 135. MIT press Cambridge, 1998.
12. Douglas Walton. *Argumentation schemes for presumptive reasoning*. Routledge, 2013.
13. Michael Wooldridge. *An introduction to multiagent systems*. John Wiley & Sons, 2009.
14. Tangming Yuan, David Moore, and Alec Grierson. Computational agents as a test-bed to study the philosophical dialogue model" de": A development of mackenzie's dc. *Informal Logic*, 23(3), 2003.
15. Tangming Yuan, David Moore, and Alec Grierson. A human–computer debating system prototype and its dialogue strategies. *International Journal of Intelligent Systems*, 22(1):133–156, 2007.
16. Tangming Yuan, David Moore, and Alec Grierson. A human-computer dialogue system for educational debate: A computational dialectics approach. *International Journal of Artificial Intelligence in Education*, 18(1):3–26, 2008.
17. Tangming Yuan, David Moore, and Alec Grierson. Assessing debate strategies via computational agents. *Argument and Computation*, 1(3):215–248, 2010.
18. Tangming Yuan, David Moore, Chris Reed, Andrew Ravenscroft, and Nicolas Maudet. Informal logic dialogue games in human–computer dialogue. *The Knowledge Engineering Review*, 26(2):159–174, 2011.
19. Tommy Yuan. *Human-Computer Debate, a Computational Dialectics Approach*. PhD thesis, Unpublished Doctoral Dissertation, Leeds Metropolitan University, 2004.