Modeling the Opponent's Action Using Control-Based Reinforcement Learning

Ismael T. Freire[1,3(✉)], Jordi-Ysard Puigbò[1,3], Xerxes D. Arsiwalla[1,2,3], and Paul F. M. J. Verschure[1,3,4]

[1] IBEC, Institute for BioEngineering of Catalonia, Barcelona, Spain
ismaeltito.freire@gmail.com
[2] UPF, Universitat Pompeu Fabra, Barcelona, Spain
[3] BIST, Barcelona Institue of Science and Technology, Barcelona, Spain
[4] ICREA, Catalan Institute for Research and Advanced Studies, Barcelona, Spain

Abstract. In this paper, we propose an alternative to model-free reinforcement learning approaches that recently have demonstrated Theory-of-Mind like behaviors. We propose a game theoretic approach to the problem in which pure RL has demonstrated to perform below the standards of human-human interaction. In this context, we propose alternative learning architectures that complement basic RL models with the ability to predict the other's actions. This architecture is tested in different scenarios where agents equipped with similar or varying capabilities compete in a social game. Our different interaction scenarios suggest that our model-based approaches are especially effective when competing against models of equivalent complexity, in contrast to our previous results with more basic predictive architectures. We conclude that the evolution of mechanisms that allow for the control of other agents provide different kinds of advantages that can become significant when interacting with different kinds of agents. We argue that no single proposed addition to the learning architecture is sufficient to optimize performance in these scenarios, but a combination of the different mechanisms suggested is required to achieve near-optimal performance in any case.

Keywords: Multi-agent models · Cognitive architectures
Theory of mind · Game theory · Social decision-making
Reinforcement learning

1 Introduction

Recent advances in Machine Learning (ML) have shown how model-free Reinforcement Learning (RL) algorithms can solve, apparently, any variety of tasks. This becomes more salient by works from Botvinick, Abeel and collaborators that show how RL agents can learn to collaborate in tasks, develop verbal and non-verbal communication mechanisms and develop Theory-of-Mind (ToM) like

ITF and JP have contributed equally to this work.

V. Vouloutsi et al. (Eds.): Living Machines 2018, LNAI 10928, pp. 179–186, 2018.
https://doi.org/10.1007/978-3-319-95972-6_19

behaviors [1,2]. Nonetheless, our ability to understand how these algorithms generate this set of behaviors is limited and doesn't provide a further understanding of the mechanisms underlying both biological or artificial agents cognitive abilities. To our knowledge, these algorithms behave as an optimization system that converges to a possible, reactive solution, whereas the adaptability to small changes in the environment takes substantially more time than in biological learning or adaptive systems.

In this paper, we propose a methodology to generate cognitive architectures with the objective to understand, particularly, what are the underlying mechanisms of ToM-related behavior. For this reason, we borrow from game theory the Battle of the Exes (BotE) game. We propose a variety of architectures that include the assumption of underlying ToM mechanisms and compare them in order to understand how collaboration and emergence in social interactions can occur. The following section describes the experimental setup and the architectures. We detail the comparative results between a pure RL algorithm and our architectures in Sect. 3. Finally, we discuss the results and provide insights for future advancement in social robot interactions.

2 Methods

2.1 Experimental Setup

To test the interaction between these models, we used as a benchmark the continuous-time version of *Battle of the Exes* [3]. In this version, two agents compete against each other to obtain one of the two possible rewards. One of them (high reward) gives a significantly better outcome than the other one (low reward), but if both agents reach for the same reward, none of them will obtain any. We created two conditions, *"High"* and *"Low"*, to see if the manipulation of the difference between both rewards affects the outcome. As shown in Fig. 1(A), the high reward gives 4 points in the *"High"* condition, and 2 in the *"Low"*.

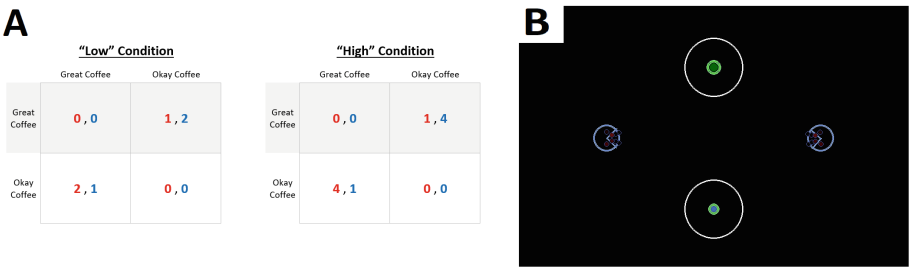

Fig. 1. (A) Payoff matrices of the *Low* (left) and *High* (right) reward conditions used in both experiments. (B) Snapshot of the experimental setup at the start of each round. In blue, two simulated *ePucks*; and in green, the reward spots. The high reward is represented by the bigger spot. The threshold that indicates the tie area is represented by the white circles.

In the two experiments carried out in this study, we perform 50 matches between different pairs of agents, with each match consisting of 200 rounds. In experiment one, for each of the three cognitive models described below (check *Control-based Reinforcement Learning Model*, *Opposing Model* and *Predictive Model* sections) we make both agents play against each other using both the same model for each of the payoff conditions, *High* and *Low*. In experiment two we test the new models against the original *Control-based Reinforcement Learning Model*. For that, we follow the same 3×2 experimental paradigm, but in this case, an agent of each pair is endowed with the original model in all conditions.

The experimental setup has been implemented in a 2D robot simulator, *PyRobot2DSim*, that is based on the *PyBox2D* and *PyGame* libraries (see Fig. 1(B) for a visual depiction of the game). All the agents used during the two experiments are embodied and situated in this 2D environment as virtual ePuck robots, so they all have the same number of sensors and actuators. More specifically, they have 3 pairs of sensors, each specialized in detecting one type of object (the high reward, the low reward, and the other agent); and two motors, to control the left and right wheels.

Regarding the rules of the Battle of the Exes, they are implemented as follows: Both agents start at equally distant positions of both rewards (see Fig. 1(B) to see the initial conditions of the game). A round ends at the moment that an agent touches a reward spot. A round ends in a tie if both agents are inside the same white circle that surrounds each reward. In any other case, the agent that first reaches a reward receives the points attached to it, and the other agent immediately receives the points of the opposite reward, as indicated by the payoff matrix. After the reward assignment is done, both agents are automatically transported to their initial positions and a new round starts.

2.2 Original Model

This model is building on the Control-based Reinforcement Learning (CRL) cognitive architecture presented in our previous work [4]. The CRL is a two-layered control architecture that follows the organizing principles of the biologically grounded Distributed Adaptive Control (DAC) theory of mind and brain. It features a low-level *Reactive Layer* for real-time sensorimotor control and an *Adaptive Layer* composed of an actor-critic reinforcement learning algorithm that can learn high-level discrete-time strategies. This hierarchical composition enables both top-down and bottom-up interactions between the reactive and adaptive components of the architecture, which helps to coordinate behavior between both layers.

The *Reactive Layer* is composed of two reactive behaviors: *"attraction towards rewards"* and *"escape from agents"*. The implementation is motivated by Valentino Braitenberg's *Vehicles* [7], where he demonstrates how to generate behavior by directly connecting the sensors and actuators of an agent. Following this approach, the *"attraction towards rewards"* is composed of a direct inhibitory connection and a crossed excitatory connection between the reward

sensors of the agent and its motors. The resulting behavior will make the agent turn in the direction of a reward with a speed proportional to the activation of its reward sensors. Given that the agent has two sets of sensors, each specialized in detecting one type of reward, this behavior is implemented twice, once for the high reward and once for the low reward. For the *"escape from agents"* behavior, the set of connections is the opposite; a crossed inhibitory connection and a direct excitatory connection. The effect of this configuration is an avoidance behavior whose speed is also linked to the level of activation of the sensors specialized to detecting other agents.

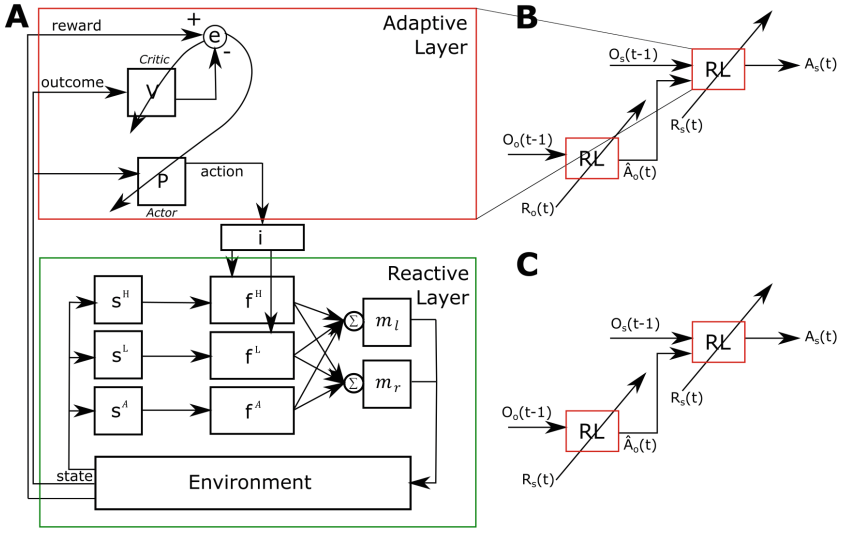

Fig. 2. (A) Schema of the Control-based Reinforcement Learning architecture. The Adaptive Layer (big red box, top) is implemented as a TD-learning algorithm, that in turn is composed of an Actor (P) and a Critic (V). The Critic predicts the expected value of a given state, and based on the outcome of the previous round, it calculates a temporal-difference error -or TD error- (e) that serves to update itself and the Actor. The Actor is in charge of selecting which action to perform in each state. This action is sent down to the inhibitor (i), that will shut down one of the reactive behavior depending on the action chosen. The Reactive Layer (big green box, bottom) is integrated by three set of sensors to detect high and low rewards (s^H, s^L) and other agents (s^A), two motors (m_l, m_r) and three functions that connect the sensors and the motors (f^H, f^L, f^A) to create the reactive behaviors of reward attraction and agent avoidance. (B) Schematic description of the Other's Model. It's composed of two Reinforcement Learning algorithms (RL) such as the one described in A. The first one (left) predicts the action of the other agent based on its outcome, and its updated with the other agent's reward. The second one (right), uses that prediction and its own outcome to choose its own action. (C) Schematic description of the Self Model. It's made of two RL algorithms as in B. The first RL module its also used to predict an action based on the other agent's outcome, but then its updated according to the agent's own reward. (Color figure online)

The *Adaptive Layer* implements a Temporal-Difference reinforcement learning algorithm (TD-learning) [8] whose goal is to learn how to maximize the acquisition of reward. It does it by learning a policy that decides which action to perform for a given state of the environment. In this task, the state of the environment is the outcome of the previous round. When a new round starts, the TD-learning chooses an action based on the previous outcome, and when the round ends, it updates its policy according to the reward received (for a detailed explanation see Fig. 2(A)). In this implementation, there are three possible states - *"high"*, *"low"*, and *"tie"*- that correspond to agent's last result of the game (it got the high reward, the low reward or it tied, respectively). The available actions are: *"to the high"*, *"to the low"*, and *"none"*.

The interaction between the *Reactive* and *Adaptive* layers is modulated by the action selected in each round, and it serves to selectively inhibit those reactive behaviors that are not needed to execute the selected action. In the case that the *Adaptive Layer* has chosen the action *"to the high"*, the *"attraction towards low reward"* reactive behavior will be inhibited. On the contrary, if the *Adaptive Layer* selects *"to the low"*, then the inhibited behavior will be *"attraction towards high reward"*. However, when the selected action is *"none"*, none of the reactive behaviors is inhibited. This capacity to modulate the agent's attention according to its planned action, in combination with the intrinsic agent-avoidance behavior, has been shown crucial for achieving efficient multi-agent interactions in real-time scenarios [4].

2.3 Other's Model

The *Other's Model* architecture implements two RL algorithms in the Adaptive layer, such as the one described in the previous section (TD-learning). The first algorithm is in charge of predicting the action of the other agent. It receives the other agent's outcome to make the prediction, and it keeps track of the other agent's reward to update it's prediction, effectively creating an internal model of the other agent's policy. Then, the action predicted by the first RL serves as an input to the second one, along with the outcome of the previous round, to produce the agent's next action (see Fig. 2(B)). As in the original *Control-Based Reinforcement Learning* model, the second RL is updated according to its own reward.

2.4 Self Model

The *Self Model* architecture presents a subtle change compared to the *Other's Model* architecture. Conceptually, instead of trying to predict the action of the other agent based on an internal model of the other's strategy, this architecture allows an agent to predict what it would do if it were in the position of the other, and to use this information to chose its own action. Structurally, the first RL algorithm still uses the other agent outcome to make a prediction about the other's future action but is updated using the agent's own information regarding outcome and reward. The second RL functions in the same way as the second RL of the *Other's Model*. So, in this architecture, both RL algorithms are updated

following the agent's own model, without taking into account any information about the other, except for the moment of making the prediction (see Fig. 2(C)).

3 Results

In order to assess performance of different models described above, we use the same metrics as in [4]: *Efficiency*, *Fairness* and *Stability*. *Efficiency* measures the cumulative sum of rewards that agents were able to earn collectively in each round, divided by the total amount of possible rewards. A value of 1 in efficiency is equivalent to both agents obtaining all rewards with no ties at all. *Fairness* quantifies the balance in reward distribution between the two agents. In this case, a fairness value of 1 means that both players earned the higher payoff the same amount of times or, equivalently, that the agents entered a turn-taking kind of convention that fairly distributed rewards among them. Finally, *Stability* measures how predictable the behavior was or, equivalently, whether the agents converged to a common strategy or alternated between non-deterministic states. In other words, stability quantifies how predictable are the outcomes of the following rounds based on previous results by using the information-theoretic measure of surprisal (also known as self-information), which Shannon defined as "the negative logarithm of the probability of an event". Analysis of variance and post-hoc tests were performed for each condition.

Our results compare the different architectures detailed in the previous section in the following manner. First, we compare the performance of each algorithm when interacting with itself (Fig. 3, top panels). Then we extend this comparison by making each architecture interact with the original RL model (Fig. 3, bottom panels). By doing this, we observe how models achieve similar levels of efficiency and fairness when competing against opponents of equal complexity (Fig. 3(A)). In contrast, when both predictive models are paired with the Original model, they tend to engage in more efficient interactions. Nonetheless, these interactions are not necessarily equally fair, observed by the reduced fairness in Fig. 3(B), where projecting self into the other agent tends to lead to more efficient and less fair interactions. We interpret that this behavior is derived from the added complexity in the predictive model, which allows to rapidly converge into dominant interactions.

Overall, we observed that when these models are faced with simpler models (i.e. the original RL model), the benefits of the ToM modules in the architecture provide a significant benefit, reflected by an increase in efficiency in both high (*Original*, $M = 0.88$; versus *Other's Model*, $M = 0.91$, $p < .001$; and versus *Self Model*, $M = 0.92$, $p < .001$) and low (*Original*, $M = 0.86$; versus *Other's Model*, $M = 0.88$, $p < .001$; and versus *Self Model*, $M = 0.89$, $p < .001$) conditions. Our previous work (IROS, under review), showed that Supervised Learning (SL) based predictive algorithms were having a similar effect when facing the Original model. However, in that case, they showed decreased efficiency when facing themselves. We suggest that the difference in the results between RL-based models and SL-models is in the nature of their learning paradigm: while SL algorithms can easily predict systems that are simpler than themselves, RL models seem to be able to improve performance even in the most adverse odds, as observed in [1,2].

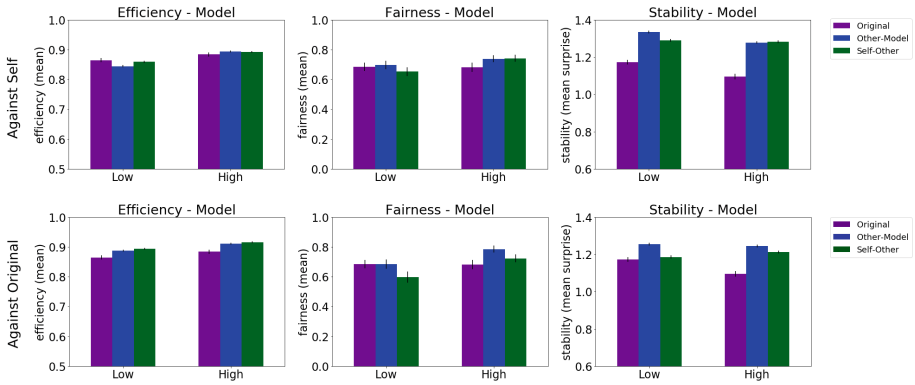

Fig. 3. Predictive models are generally more efficient. Top panel: Results of the models competing against themselves measured by *Efficiency*, *Fairness*, and *Stability*, both in *High* and *Low* payoff difference conditions. Note that *Stability* is measured by the level of surprisal, so a high value in surprisal means lower *Stability*. Bottom panel: Results of the models competing against the original (non-predictive) model, measured with the same metric as in the top panel.

4 Discussion

Our model proposes two alternatives to previous work, by either including a model of the other or a model of the self that are used to predict the other agent's potential actions. With this, we test the architectural properties of some of the underlying processes of Theory Of Mind. These models have demonstrated to be an advantage when tested in a social decision-making task as shown by the increased efficiency of the outcomes. More particularly, our results suggest that having such predictive models of the other benefits the global outcome of the interaction, whereas the model of the other vs. the model of one's self, differ significantly on their ability to collaborate (illustrated by the fairness measure).

We conclude that the evolution of mechanisms that allow for the control of other agents provide different kinds of advantages that can become significant when interacting with other kinds of agents. We argue that no single proposed addition to the learning architecture is sufficient to significantly optimize performance in these scenarios by its own, but a combination of the different mechanisms suggested may achieve near-optimal performance in any case.

Additionally, our framework provides an interesting possibility to model complex social behaviors using multi-agent robotic systems [9]. These behaviors have also been tied to evolutionary dynamics of biological life forms and conscious beings [10], [11], [12], [13]. The study of these social behaviors can also be of great interest for achieving a greater understanding of human social and cultural development and testing these ideas using multi-agent robot platforms.

Acknowledgments. The research leading to these results has received funding from the European Commission's Horizon 2020 socSMC project (socSMC-641321H2020-FETPROACT-2014) and by the European Research Council's CDAC project (ERC-2013-ADG341196).

References

1. Rabinowitz, N.C., Perbet, F., Song, H.F., Zhang, C., Eslami, S.M., Botvinick, M.: Machine Theory of Mind (2018). arXiv preprint arXiv:1802.07740
2. Mordatch, I., Abbeel, P.: Emergence of grounded compositional language in multi-agent populations (2017). arXiv preprint arXiv:1703.04908
3. Hawkins, R.X.D., Goldstone, R.L.: The formation of social conventions in real-time environments. PLoS One **11**, e0151670 (2016)
4. Freire, I.T., Moulin-Frier, C., Sanchez-Fibla, M., Arsiwalla, X.D., Verschure, P.: Modeling the Formation of Social Conventions in Multi-Agent Populations (2018). arXiv preprint arXiv:1802.06108
5. Verschure, P.F.M.J., Voegtlin, T., Douglas, R.J.: Environmentally mediated synergy between perception and behaviour in mobile robots. Nature **425**, 620–624 (2003)
6. Moulin-Frier, C., Arsiwalla, X.D., Puigbo, J.Y., Sanchez-Fibla, M., Duff, A., Verschure, P.F.: Top-down and bottom-up interactions between low-level reactive control and symbolic rule learning in embodied agents. In: CoCo@ NIPS (2016)
7. Braitenberg, V.: Vehicles: Experiments in Synthetic Psychology. MIT Press, Cambridge (1986)
8. Sutton, R.S.: Learning to predict by the methods of temporal differences. Mach. Learn. **3**, 9–44 (1988)
9. Moulin-Frier, C., Puigbo, J.Y., Arsiwalla, X.D., Sanchez-Fibla, M., Verschure, P.F.: Embodied artificial intelligence through distributed adaptive control: An integrated framework (2017). arXiv preprint arXiv:1704.01407
10. Arsiwalla, X.D., Herreros, I., Moulin-Frier, C., Sanchez, M., Verschure, P.F.: Is consciousness a control process? Artificial Intelligence Research and Development, pp. 233–238. IOS Press, Amsterdam (2016)
11. Arsiwalla, X.D., Herreros, I., Verschure, P.: On three categories of conscious machines. In: Conference on Biomimetic and Biohybrid Systems, pp. 389–392 (2016)
12. Arsiwalla, X.D., Herreros, I., Moulin-Frier, C., Verschure, P.: Consciousness as an Evolutionary Game-Theoretic, Strategy, pp. 509–514 (2017)
13. Arsiwalla, X.D., Moulin-Frier, C., Herreros, I., Sanchez-Fibla, M., Verschure, P.: The Morphospace of Consciousness (2017). ArXiv preprint arXiv:1705.11190