Learning to Cooperate with Human Evaluative Feedback and Demonstrations

Mehul Verma [a,1] and Erman Acar [a,b]
[a] *Vrije Universiteit Amsterdam, The Netherlands*
[b] *LIACS, Universiteit Leiden, The Netherlands*

Abstract. Cooperation is a widespread phenomenon in nature that has also been a cornerstone in the development of human intelligence. Understanding cooperation, therefore, on matters such as how it emerges, develops, or fails is an important avenue of research, not only in a human context, but also for the advancement of next generation artificial intelligence paradigms which are presumably human-compatible. With this motivation in mind, we study the emergence of cooperative behaviour between two independent deep reinforcement learning (RL) agents provided with human input in a novel game environment. In particular, we investigate whether evaluative human feedback (through interactive RL) and expert demonstration (through inverse RL) can help RL agents to learn to cooperate better. We report two main findings. Firstly, we find that the amount of feedback given has a positive impact on the accumulated reward obtained through cooperation. That is, agents trained with a limited amount of feedback outperform agents trained without any feedback, and the performance increases even further as more feedback is provided. Secondly, we find that expert demonstration also helps agents' performance, although with more modest improvements compared to evaluative feedback. In conclusion, we present a novel game environment to better understand the emergence of cooperative behaviour and show that providing human feedback and demonstrations can accelerate this process.

Keywords. Multiagent Reinforcement Learning, Multiagent Cooperation, Inverse Reinforcement Learning, Interactive Reinforcement learning.

1. Introduction

While artificial intelligence (AI) technologies are playing more important roles in our daily lives than ever, designing intelligent systems which can work with humans more effectively (instead of replacing them) is becoming a central research challenge [1,2,3,4]. This is mostly pronounced as combining human and machine intelligence, aiming to benefit from the strengths of both in solving problems in various scenarios.

Developing such systems requires fundamentally novel solutions to major research problems in AI: It is not secret that the current AI systems outperform humans in many cognitive tasks from pattern recognition [5] or in playing video games [6], yet they fall short when it comes to tasks such as causal modelling, common sense reasoning, and behavioural human capabilities such as explaining its own decisions, adapting to differ-

[1] Corresponding author: Mehul Verma, email address: mehuljan26@gmail.com

ent environments, collaborating with other human agents, etc. A particular challenge in developing such systems is to work with human input which is the main topic of this paper.

Being able to work with human input mandates the essential skill that any such system should have: *cooperation*. Cooperation is a widespread phenomenon in nature that has also been a cornerstone in the development of human intelligence [7,8]. Understanding cooperation, therefore, on matters such as how it emerges, develops, or fails is an important avenue of research, not only in a human context, but also for the advancement of next generation artificial intelligence paradigms which are presumably human-compatible [9]. With this motivation in mind, we study the emergence of cooperative behaviour between two independent deep reinforcement learning (RL) agents provided with human input in a novel game environment. In particular, we investigate whether evaluative human feedback (through interactive RL) and expert demonstration (through inverse RL) can help RL agents to learn to cooperate better.

This work can be considered as a contribution to the multiagent systems with deep reinforcement learning, which is also referred to as Multi-Agent Deep Reinforcement Learning (MADRL). MADRL focuses on the sequential decision-making problem of multiple autonomous agents interacting together with the environment to maximize their long-term returns using a neural network as a function approximator in order to learn in environments with high dimensional state spaces. MADRL has achieved important milestones such as defeating the DOTA II world champion [10] and reaching expert performance in Starcraft II [11].

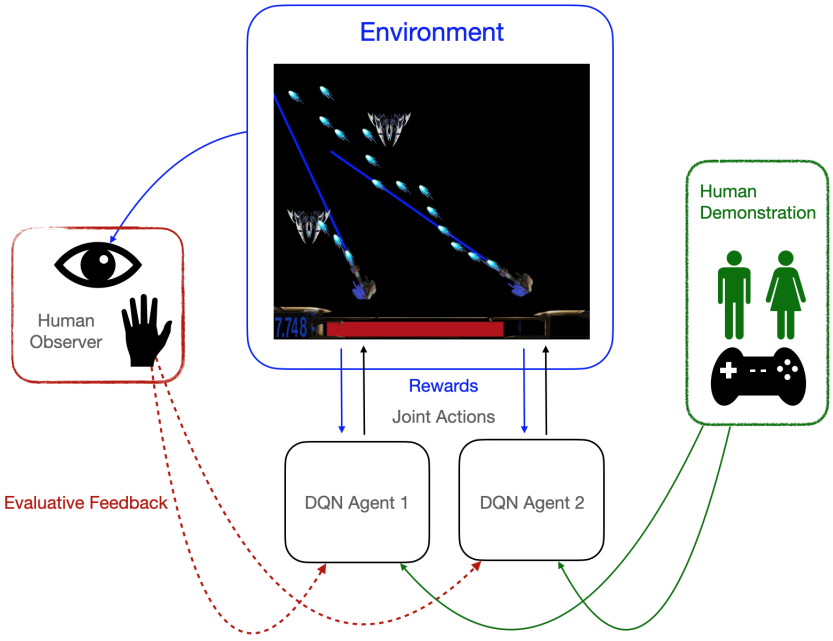

Figure 1. A simplified illustrative summary of the main idea of the paper. Two player real-time environment *Space Cannons* controlled by two DQN agents which are fed with human evaluative feedback (on the left side) and demonstrations (on the right side).

We should note that several technical challenges make multiagent learning fundamentally more difficult than the single-agent case, such as the moving target problem (non-stationarity) [12], the curse of dimensionality, multiagent credit assignment [12] and global exploration [13]. Another limitation in further developing these techniques concerns the lack of environments in which cooperation between multiple agents is incorporated. Moreover, the testbeds that do include this element do not properly capture cooperative behaviour as a graded quantity and/or solely employ turn-based systems. As a result, efforts to better understand how cooperation emerges between agents, as well as to enhance it are bounded.

The main contribution of this paper is two-fold. First, a novel environment is introduced which can be used to more precisely measure and analyse cooperation between two deep reinforcement learning agents. Second, two reinforcement learning techniques were used to accelerate the induction of cooperation between agents: namely, interactive and inverse reinforcement learning. Specifically, two independent Deep Q-learning (DQN) agents were supplemented with either positive evaluative feedback from human observers or demonstration data from humans, respectively. We demonstrate that while both techniques are effective at promoting cooperation, particularly interactive reinforcement learning yielded improvements in performance compared to baseline.

2. Background

2.1. Multiagent Markov decision process

The game environment can be expressed as a fully observable [14] *Multiagent Markov Decision Process* (MMDP) [15], where the selected action for any state in the environment consists of individual action components from each of the agents in the MMDP, whom share a common reward function. Formally, it is a tuple $G = \langle N, S, A, R, T, \gamma \rangle$, where $N = \{1,2\}$ is a set of two agents, S is the state space, $A = A_1 \times A_2$ is the joint action space (with A_1 and A_2 representing the action spaces of each agent, respectively), $R : S \times A \to \mathbb{R}$ is the shared reward function, $T : S \times A \times S \to [0,1]$ is a probabilistic transition function and $\gamma \in [0,1]$ is the discount factor for future rewards. The main objective of the agents in our game is to maximise their expected sum of discounted rewards by finding a joint cooperative strategy $\pi = (\pi_2, \pi_2)$, where $\pi : S \to A$, $\pi_1 = S \to A_1$ and $\pi_2 = S \to A_2$.

2.2. Multiagent Deep Reinforcement Learning

The aim of deep reinforcement learning agents is to find an optimal action-selection strategy by maximising a cumulative numerical reward signal through trial and error learning in their respective environments [16]. One of the popular approaches to this in deep RL is DQN [16], which was built upon the popular Q-learning algorithm [17]. In contrast to the Q-learning algorithm, DQN uses a convolutional neural network as a Q-function approximator. The neural network used in the DQN architecture is parameterised by θ and trained on input data to minimise the loss function defined in eq. (1).

$$L(\theta) = \mathbb{E}_{(s_t, a_t, r_t, s_{t+1}) \sim D}[(r + \gamma \max_{a'} Q(s', a' | \theta') - Q(s, a | \theta))^2] \tag{1}$$

where L is the loss function, D is experience replay, r is the reward, γ is the discount factor, $Q(s,a|\theta)$ are the current Q-values with state s and action a and $Q(s',a'|\theta)$ are target Q-values.

To estimate the maximal Q-value, a separate network parameterised as θ' is used. This network is referred to as the target network. The weights of this network are overwritten by the main network weights at a certain interval (every 100 iterations in our implementation). Furthermore, an experience replay D is used to store all experiences (s,a,r,s'), which are then uniformly sampled from by the DQN algorithm to update network weights. DQN also uses a ε-greedy strategy to balance exploration and exploitation in the environment similar to one used in [16]. The value of ε is decremented over time from 0.99 to 0.1 with a decay of 0.999996 in our experiments.

3. Related work

Multiple studies have investigated cooperative behaviour in MARL and the effect of interactive or inverse reinforcement learning on agent behaviour. In this section, we describe the approaches that most closely resemble the current work and discuss how they can be combined.

The article [18] is one of the early approaches to study decentralised multiagent cooperation with deep reinforcement learning. Specifically, the authors study how manipulating the reward function affected the progression of cooperation and competition between independent Q-learners [19]. Another related study [20], shows that agents could learn cooperative strategies in the same two-player pong game using only raw pixel data, even within a non-stationary environment. We base our study on these two papers while focusing only on the cooperative side of the spectrum. Moreover, we extend this work by including the effects on inverse and interactive RL to investigate whether this could improve the sample efficiency of MARL.

Interactive RL, also known as human-centred reinforcement learning [21], uses human evaluative feedback to encourage optimal actions taken by deep RL agents. Interactive RL is inspired by potential based reward shaping [22] and was proposed to solve the inherent problem of sample efficiency in deep RL techniques. The technique used in this paper is that of model-free, reward-based interactive reinforcement learning. Here, human feedback is taken as a numerical reward and is thereby used in a similar manner as traditional rewards in RL systems. One of the earliest and most relevant works on this topic uses clicker training to provide additional positive reward to train a synthetic puppy agent [23]. Another early study uses a similar approach to train an agent in a complex social environment using both positive and negative reward [24]. This is somewhat similar to our approach, but we included only positive evaluative feedback. Furthermore, due to the delayed and sparse nature of rewards from our novel environment, we used feedback from both humans and the environment to speed up exploration and learning [25] in a manner similar to the TAMER + RL framework discussed in [26]. Possibly the most closely related paper to our implementation used DQN-TAMER [27], which is a generalisation of the human-in-a-loop algorithm TAMER and the Deep Q-Learning (DQN) framework. In this study, facial expressions were used as additional feedback for the DQN agent and proved that DQN-TAMER could outperform its constituent algorithms even despite issues such as instability and randomness of delayed feedback. Instead, our implementation only uses human input in the form of a button press.

Inverse RL, introduced in [28], aims to learn the reward function from (expert) human behaviour rather than learning the optimal policy given a reward function as in traditional RL setting, hence the name *inverse*. In our analysis, we employ this very idea with two-players in a cooperative-based game environment that learns from not only from self-exploration of the environment, but also from a dataset containing expert demonstration trajectories.

Literature for learning from demonstrations can be broadly subdivided into two categories: one that assumes the optimality of demonstration data [29,30] and one which does not [31]. This optimality cannot be assumed in real-world scenarios [32]. Moreover, even if it would hold, agents trained purely on demonstration data may not be able to explore the entirety of the state space, since the demonstration data will only contain expert trajectories. Hence, in our approach we assume inherent non-optimality in demonstration data. Perhaps the closest relationship to our work is [6], which proposes deep Q-learning from demonstrations (DQfd). DQfd is an algorithm built upon a DQN model and uses a dual replay buffer to store transition data for expert demonstration and self-exploration. This data is then uniformly sampled from the replay buffer to update network parameters using a combination of 1-step temporal difference (TD) learning, n-step TD, supervised, and regularisation losses. We should note that we take a less complex approach using only TD learning, along with different ratios of sampling from the replay buffer versus self-exploration to investigate the utility of demonstration directly.

4. The Game Environment

We introduce a novel two-players general sum game environment named *Space Cannons*. Space Cannons is a real-time 2D game in which two cannons controlled by two independent deep reinforcement learning agents (see section 2) shoot at enemy spaceships to earn rewards. Hence, both agents receive the same inputs and generate actions that are simultaneously executed in the environment. Since Space Cannons was specifically designed to better understand the emergence of cooperative behaviours between the two agents, enemies are configured to be very difficult to defeat by an individual player itself. Therefore, a cooperative behaviour (i.e., coordinated shooting) between both agents is necessary to shoot down an enemy.

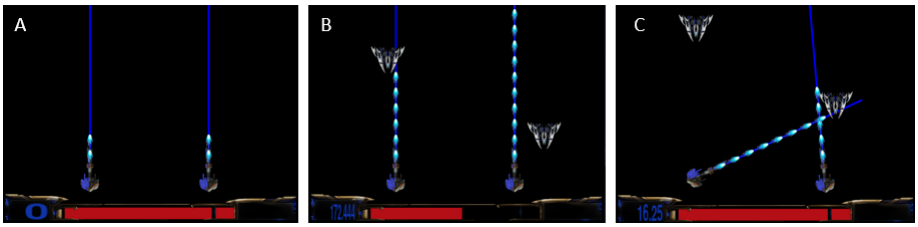

Figure 2. Example frames from the game Space see Cannons. A) shows the state of the game at initialisation, when no enemies have been spawned yet. In B), two enemies are spawned, but the agents are not cooperatively firing at one single enemy. In C), the two agents coordinate their firing and target one enemy; thus displaying cooperation.

Moreover, Space Cannons distributes rewards according to the degree of cooperative behaviour displayed by the agents. Consequently, maximum reward can only be obtained

when both agents contribute equally when executing an enemy. Finally, Space Cannon supports interactive and inverse reinforcement learning settings by incorporating components such as the collection of demonstration data and the provision of positive feedback through human interaction.

4.1. Game components and properties

There are three major components in the game:

1. *Player* objects are the two identical cannons at the bottom of the screen. These cannons are hinged to the surface and may rotate 65 degrees to the left and right to shoot at adversaries approaching from above. The player cannons are also equipped with an aiming laser to facilitate accurate shooting.
2. *Bullet* objects are fired by two identical cannon objects, which upon contact with any enemy will lead to a reduction in enemy health. The bullet objects can be fired using two modes, namely burst mode and normal mode. The former allows the cannons to continuously fire bullets, while the latter offers the player a choice whether to fire or not. For all experiments the burst mode was used.
3. *Enemies* are spaceship objects that spawn at the top of the screen and descend towards cannons at a constant speed. These enemies can cause damage to the player's health when they manage to evade players by passing through the bottom screen.

4.2. Gameplay and reward mechanism

The game begins with the guns facing the top of the screen. After a few frames, a new enemy spaceship spawns every 3 seconds (in terms of learning time) at a random location at the top of the screen and flies to the bottom of the screen. Henceforth, players can rotate the relative positions of their guns in order to fire bullets at the enemies. As these bullets are fired continuously, the players only need to aim at the enemies to defeat them and do not need to learn how and when to shoot. If the enemies are not defeated by the players before they pass through the bottom of the screen, a penalty is incurred and the health of both players is decayed. The game ends when in total 30 enemies have evaded the players. The players need to shoot each enemy with 160 bullets in order to kill them. This requirement was a design choice to prevent individual players from easily being able to defeat enemies by themselves, thereby motivating the agents to collaborate to obtain high scores.

Space Cannons has an intrinsic mechanism to distribute rewards based on the grade of cooperative behaviour displayed by the agents. To do this, a scoring mechanism is initialised whenever an enemy spaceship is eliminated. This scoring mechanism takes the number of hits by the individual players into the account and based on this information calculates the *Individual Contribution* (IC_i) and the *Cooperative Contribution* (α) per enemy. The individual contribution IC_i for each agent i is then calculated by taking the number of hits made by that agent to the enemy and dividing it by the maximum health of the enemy:

$$IC_i = \frac{\text{\#Successful Hits}}{\text{Max Health}} \tag{2}$$

Afterwards, the cooperative contribution is calculated by using the absolute difference of individual contributions from both the players:

$$\alpha = 1 - |IC_1 - IC_2| \tag{3}$$

If an enemy is eliminated with high cooperation between agents, this will result in very similar individual contributions and thereby a high value of cooperative contribution. The cooperative contribution is then used as a threshold to decide if the kill was considered cooperative or not. In our experiment, the value of this threshold was set to 0.5. An increase in this threshold would entail a higher level of cooperation for a kill to be considered cooperative. The individual total reward TR_i for an agent i is then defined as the multiplication of the terms cooperative contribution (α), the individual contribution (IC_i) and the maximum reward (R) for defeating an enemy (in the current experiments, this value was set to 20):

$$TR_i = \alpha \cdot IC_i \cdot R \tag{4}$$

4.3. Agents configuration

In this study, we used the DQN algorithm (see Section 2) as the principal method. DQN was chosen due to its success in solving high-dimensional problems [16]. Since the game Space Cannons generates visually rich game content on a 950×750 display, the frames that are extracted from the game at every iteration may be too complex to process and learn from. Therefore, three preprocessing steps were applied to alleviate this problem. Firstly, information about the scoreboard and enemy health bar was cropped from the frames as they were not required for the model. Moreover, frames were rescaled from $950 \times 750 \times 3$ to $84 \times 84 \times 3$ and normalised by re-scaling them from $0 - 255$ to $0 - 1$. Secondly, colour channels were removed from the frames, thereby converting their dimension from $84 \times 84 \times 3$ to 84×84. Thirdly, to add a sense of motion in our observations, we stacked four consecutive frames as input to the neural network[2], which changed the shape of the frames to $84 \times 84 \times 4$.

The architecture of the DQN used consisted of a two-layer convolutional neural network (CNN) [33], which received input of size $84 \times 84 \times 4$ from the preprocessing function. The first hidden layer was composed of 32 filters of 8×8 with stride 4 and the second layer of 64 filters of 4×4 with stride 2. Both were followed by a non-linear rectifier unit [34]. The output was then passed to a fully connected layer consisting of 512 rectifier units. The output layer was a fully connected linear layer with a single output for each valid action. Finally, RMSprop [35] was used as the loss function for all experiments.

[2]In future implementations, this feature can likely be removed to reduce computational complexity since this game is modelled as an MMDP [15]

5. Experimental Setting

5.1. Baseline: No human input

As the baseline, the two DQN agents were trained for 150 games without any human input. At every iteration, new actions were consequently executed either randomly or from the agents' networks, depending upon the current value of ε in the experiment (as part of the ε-greedy approach).

5.2. Experiment 1: Using interactive reinforcement learning

Our first experiment is involved with using real-time positive feedback from a human observer. A voluntary human participant was instructed to provide a reward signal whenever an optimal behaviour (i.e., cooperation) was observed between the agents.[3]. This reward signal was converted into a positive reward of $+1$ and appended to the current reward signal from the Space Cannons environment. Afterwards, it was stored in the experience replay from which the agents uniformly sampled transition data to update their model parameters.

5.2.1. Handling incorrect human feedback

A common limitation of working with human feedback is that the feedback might not always be accurate due to a variety of systemic factors stemming from limitations in human (cognitive) capacity, such as delays [37] or user fatigue [27], which could degrade the quality of feedback after a certain amount of time. While it is unlikely that faulty feedback can be entirely avoided (after all, this is not our aim either), some techniques can still drastically mitigate such systemic errors and help with the learning of the agents. Various techniques have been proposed to alleviate these problems in the field of *interactive reinforcement learning* [27,38,39,40]. In our paper, we employed some of these techniques. The first of these was to lower the framerate of the game so that observers have enough time to give feedback. Second, the framerate of the game was even further decreased after receiving the first feedback, which allows the human to decide if she/he still thinks that the agents are cooperating. Third, we observed that the human observer typically took some time to stop giving feedback after an enemy was killed. In order to avoid that, we added a delay of 1.5 seconds after every enemy is killed, which gave enough time for the observer to release the feedback key. And as last, to tackle with the user fatigue, we added an option for the observer to freeze the environment and take a break during the experiment.

5.2.2. Quantifying human evaluative feedback

Another challenge in providing human evaluative feedback is that such process can become extremely repetitive, cumbersome and time consuming, since it usually requires a large amount of feedback before we can see any sign of learning. Therefore, it is important to quantify the amount of feedback required to achieve different level of agent performance in our environment. To do this, we provided evaluative feedback for the first 60 out of 150 games during our experiments. The optimiser and network parameters were

[3] In game-theoretic terms, the human participant acts as a *central authority* for the agents to correlate [36]

saved every 10 games until 60 games were completed. These parameters could later be used to continue training the model from the exact timestep when they were saved, in the absence of new evaluative feedback. This allowed for the generation of a performance curve for agents that were trained for the first 10, 20, 30, 40, 50 or 60 games with evaluative feedback. That way, we could quantify the amount of feedback without the need of repeating the feedback experiment too many times (and thereby incurring additional variance).

5.3. Experiment 2: Using Inverse reinforcement learning

Our next experiment investigated whether inverse reinforcement learning through demonstration data could improve the performance and sample efficiency of DQN agents compared to using only self-exploration. In order to do so, the experiment was divided into two phases.

The first phase of this experiment focuses on the collection of demonstration data. In this phase, interaction data between two voluntary human participants and the environment is compiled. After explaining the controls of the game, the participants were instructed to not communicate with one another during the experiment in order to replicate the agents' configuration in our experiment. At every iteration, the pixel data of the state space from the game, actions taken, rewards generated, and pixel information of the next state space was appended to a dataset. The maximum size of this dataset was set to $200k$. After the dataset reached its maximum size, the game was stopped and the dataset was saved. The experiment then progressed to the second phase.

The second phase of the experiment focused on how agents could learn from a combination of self-exploration and the demonstration data. In this phase, the demonstration data from prior phase was stored in a replay buffer which was made available to the DQN agents to sample data and update network parameters from. For convenience, hereafter, we shall refer to this replay buffer object containing demonstration data as *demo buffer*, and the replay buffer created by self-exploration of agents as *regular replay buffer*. It has previously been observed that mixed sampling from these buffers improved performance on Atari games [6]. As it would be interesting to know what the ideal ratio of sampling from each buffer should be, three different sampling ratios were used in the demonstration experiments: namely, 0%:100% (i.e. baseline), 50%:50% and 100%:0% from the demo buffer and regular replay buffer, respectively.

6. Results

The goal of this paper was to investigate whether inverse and interactive RL techniques could affect the rate at which cooperation emerged between two deep reinforcement learning agents. While for interactive reinforcement learning we tested the effect of providing positive feedback from a human observer, in inverse reinforcement learning we focused on how to use expert demonstration data to improve the performance of the agents when used alongside self-exploration. Since the game was designed in such a way that a single agent could nearly not defeat any enemies alone, the average reward obtained by the two agents was used as a metric to evaluate the performance.

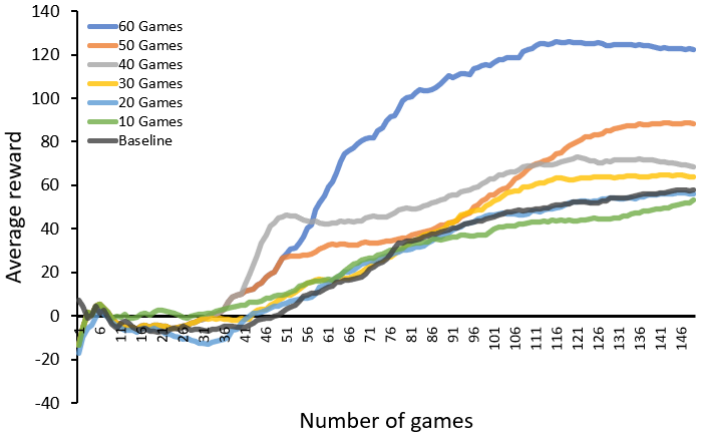

Figure 3. Human evaluative feedback could have a positive effect on the average reward obtained by the two agents on Space Cannons. Curves represent the same model trained without feedback (baseline) or with feedback for the first 10, 20, 30, 40, 50 or 60 games.

6.1. Interactive reinforcement learning improves learning: The more the merrier

To understand the effect of positive feedback by human observers on the two DQN agents, we quantified how much feedback was required to teach the agents how to co-operate. Figure 3 describes the average performance of the agents when trained for either the first 10, 20, 30, 40, 50 or 60 games with evaluative feedback, as well as the baseline performance when no evaluative feedback was given at any time. Overall, it can be observed that the agents trained with feedback from a human observer achieved better performance than the baseline. Furthermore, increasing the given amount of feedback also seemed to enhance the performance of the agents. Namely, the performance of agents trained for only the first 10 and 20 games with evaluative feedback was observed to be similar to the baseline experiment. However, when the number was increased to 30 games or above, there seemed to be an increase in the performance of the agents. The agents trained for as many as 60 games with feedback managed to obtain almost twice the amount of average reward compared to the baseline. Therefore, these results suggest that adding human evaluative feedback can aid agents in learning cooperative policies.

6.2. Inverse reinforcement learning improves learning

To test the effect of inverse reinforcement learning, we ran three different experiments for 150 games with different ratios of sampling probabilities from either of the demo or regular replay buffer. Figure 4 describes the average performance of the agents in these three settings. It can be seen that agents trained solely on demonstration data failed to learn a cooperative policy. In contrast, agents trained with equal sampling probability (50:50) from both buffers appeared to achieve better performance than baseline, in which only self-exploration was used. Repeating the experiment with different random initialisations (of the network) yielded similar conclusions. Hence, it can be concluded that providing demonstration data appeared to increase agents' performance, although removing self-exploration altogether resulted in a failure to learn.

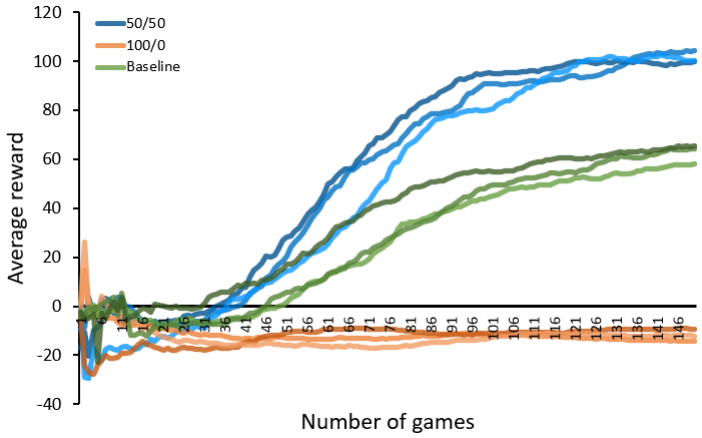

Figure 4. Human demonstration data could increase the average reward obtained by the agents on Space Cannons when sampling equally from the demonstration buffer and regular replay buffer (50/50, respectively), compared to baseline (0/100). Learning appears to fail when sampling exclusively from the demo replay buffer (100/0).

7. Summary and Discussion

This study is aimed to investigate whether inverse and interactive reinforcement learning could be used to improve cooperation between two autonomous agents. We empirically show that this could indeed be the case; that is, both of these approaches enhanced cooperative agent performance on a novel reinforcement learning environment.

In particular, interactive reinforcement learning is shown to have a positive impact on the average reward obtained by the agents compared to baseline. This suggests that interactive reinforcement learning can even be helpful despite the delays and errors inherent in producing evaluative feedback. Such a finding is in line with previous studies using DQN-TAMER [27]. Another interesting finding was that evaluative feedback had to be provided for at least 30 games before an improvement in performance could be observed. Therefore, while a small amount of feedback may not be enough to improve agents performance, increasing its quantity is likely to promote the emergence of cooperative policies between agents.

In the future, it would be interesting to investigate whether negative feedback alongside the positive feedback could further help in accelerating agent learning. Moreover, although several remedies (native to our environment) were employed to reduce the impact of errors and delays in human feedback (see Section 5), it could also be worthwhile to also incorporate extrinsic strategies to counteract these problems. For instance, probabilistic methods could be used to estimate the timing of feedback delay and ensure that it is processed at the appropriate timesteps.

We should note that compared to interactive reinforcement learning, inverse reinforcement learning only resulted in a more modest increase in performance (both relative to baseline). Namely, agents trained with an equal sampling ratio from the demo buffer and regular buffer consistently seemed to outperform agents that only sampled from the regular buffer. In contrast, agents trained exclusively on demonstration data failed to acquire any cooperative policy. A potential explanation for this finding is that these agents

were only able to sample from expert trajectories. Therefore, they might have not gathered enough information about all the other states in the environment to respond to them in an optimal way. Therefore, the results suggest that a proper balance between sampling ratios from the different buffers could be required for optimal performance. Finding the optimal ratio of this is left to be explored in the future work. Additionally, it could be informative to investigate the effect of the loss function used in our experiments. Namely, no specialised loss was used to prevent overfitting on demonstration as in [6], which might explain the relatively small effect of interactive reinforcement learning in our experiments.

Even more interesting than using only interactive or inverse reinforcement learning would be to investigate the impact of using them in conjunction. Particularly, it would be informative to see whether using both approaches simply has an additive, or rather interactive effect.

Finally, the novel game environment Space Cannons we introduce in this paper could also be used as a test bed for more sophisticated sequential social dilemmas. For instance, a Stag-Hunt social dilemma [41] could be implemented by adding multiple types of enemies into the game. These enemies could include both low-health enemies that can be eliminated by a single agent, as well as more high-health enemies that require an increasingly higher level of cooperation between the agents to defeat.

These results suggest the potential of these methods in cooperation-based implementation since they might enhance the sample efficiency of learning. This could be an especially important feature in settings where training data is expensive or difficult to acquire. Lastly, these methods could possibly learn from human strengths to create new regeneration artificial intelligent paradigms that are not just more accurate, but also hopefully more compatible with human applications.

References

[1] Akata Z, Balliet D, de Rijke M, Dignum F, Dignum V, Eiben G, et al. A Research Agenda for Hybrid Intelligence: Augmenting Human Intellect With Collaborative, Adaptive, Responsible, and Explainable Artificial Intelligence. Computer. 2020;53(8):18-28.

[2] Waa Jvd, Diggelen Jv, Cavalcante Siebert L, Neerincx M, Jonker C. Allocation of moral decision-making in human-agent teams: a pattern approach. In: International Conference on Human-Computer Interaction. Springer; 2020. p. 203-20.

[3] Peng A, Nushi B, Kiciman E, Inkpen K, Kamar E. Investigations of Performance and Bias in Human-AI Teamwork in Hiring. Proceedings of 36th AAAI conference on Artificial Intelligence. 2022.

[4] Kamar E. Directions in Hybrid Intelligence: Complementing AI Systems with Human Intelligence. In: IJCAI; 2016. p. 4070-3.

[5] Liu X, Faes L, Kale AU, Wagner SK, Fu DJ, Bruynseels A, et al. A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis. The lancet digital health. 2019;1(6):e271-97.

[6] Hester T, Vecerik M, Pietquin O, Lanctot M, Schaul T, Piot B, et al. Deep Q-learning from Demonstrations. arXiv:170403732 [cs]. 2017 Nov. ArXiv: 1704.03732. Available from: `http://arxiv.org/abs/1704.03732`.

[7] Harari YN. Homo Deus: A brief history of tomorrow. Random House; 2016.

[8] Balliet D, Van Lange PA. Trust, conflict, and cooperation: a meta-analysis. Psychological bulletin. 2013;139(5):1090.

[9] Russell S. Human-compatible artificial intelligence. In: Human-Like Machine Intelligence. Oxford University Press Oxford; 2021. p. 3-23.

[10] OpenAI, Berner C, Brockman G, Chan B, Cheung V, Dębiak P, et al. Dota 2 with Large Scale Deep Reinforcement Learning. arXiv:191206680 [cs, stat]. 2019 Dec. ArXiv: 1912.06680. Available from: http://arxiv.org/abs/1912.06680.

[11] Vinyals O, Babuschkin I, Czarnecki WM, Mathieu M, Dudzik A, Chung J, et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. Nature. 2019 Nov;575(7782):350-4.

[12] Shoham Y, Powers R, Grenager T. If multi-agent learning is the answer, what is the question? Artificial Intelligence. 2007 May;171(7):365-77. Available from: https://www.sciencedirect.com/science/article/pii/S0004370207000495.

[13] Matignon L, Laurent GJ, Le Fort-Piat N. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. Knowledge Engineering Review. 2012 Mar;27(1):1-31. Publisher: Cambridge University Press (CUP). Available from: https://hal.archives-ouvertes.fr/hal-00720669.

[14] Beynier A, Charpillet F, Szer D, Mouaddib AI. DEC-MDP/POMDP. In: Sigaud O, Buffet O, editors. Markov Decision Processes in Artificial Intelligence. Hoboken, NJ USA: John Wiley and Sons, Inc.; 2013. p. 277-318. Available from: https://onlinelibrary.wiley.com/doi/10.1002/9781118557426.ch9.

[15] Boutilier C. Planning, learning and coordination in multiagent decision processes. In: TARK. vol. 96; 1996. p. 195-210.

[16] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, et al. Playing Atari with Deep Reinforcement Learning. arXiv preprint arXiv. 2013.

[17] Watkins CJCH, Dayan P. Q-learning. Machine Learning. 1992 May;8(3):279-92. Available from: https://doi.org/10.1007/BF00992698.

[18] Tampuu A, Matiisen T, Kodelja D, Kuzovkin I, Korjus K, Aru J, et al. Multiagent cooperation and competition with deep reinforcement learning. PLOS ONE. 2017 Apr;12(4):e0172395. Available from: https://dx.plos.org/10.1371/journal.pone.0172395.

[19] Tan M. In: Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 1997. p. 487–494.

[20] Diallo EAO, Sugiyama A, Sugawara T. Coordinated behavior of cooperative agents using deep reinforcement learning. Neurocomputing. 2020 Jul;396:230-40. Available from: http://www.scopus.com/inward/record.url?scp=85065027331&partnerID=8YFLogxK.

[21] Li G, Gomez R, Nakamura K, He B. Human-Centered Reinforcement Learning: A Survey. IEEE Transactions on Human-Machine Systems. 2019.

[22] Ng A, Harada D, Russell SJ. Policy Invariance Under Reward Transformations: Theory and Application to Reward Shaping. In: ICML; 1999. p. 278-87.

[23] Buchsbaum D, Blumberg B. Imitation and social intelligence for synthetic characters. In: ACM SIGGRAPH 2004 Sketches. SIGGRAPH '04. New York, NY, USA: Association for Computing Machinery; 2004. p. 98. Available from: https://doi.org/10.1145/1186223.1186346.

[24] Isbell C, Shelton CR, Kearns M, Singh S, Stone P. A social reinforcement learning agent. In: Proceedings of the fifth international conference on Autonomous agents - AGENTS '01. Montreal, Quebec, Canada: ACM Press; 2001. p. 377-84. Available from: http://portal.acm.org/citation.cfm?doid=375735.376334.

[25] Kuhlmann G, Stone P, Mooney R, Shavlik J. Guiding a Reinforcement Learner with Natural Language Advice: Initial Results in RoboCup Soccer. In: The AAAI-2004 workshop on supervisory control of learning and adaptive systems; 2004. p. 6.

[26] Knox WB, Stone P. Combining Manual Feedback with Subsequent MDP Reward Signals for Reinforcement Learning. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 1 - Volume 1. AAMAS '10. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems; 2010. p. 5–12.

[27] Arakawa R, Kobayashi S, Unno Y, Tsuboi Y, Maeda Si. DQN-TAMER: Human-in-the-Loop Reinforcement Learning with Intractable Feedback; 2018.

[28] Russell S. Learning agents for uncertain environments (extended abstract). In: Proceedings of the eleventh annual conference on Computational learning theory - COLT' 98. Madison, Wisconsin, United States: ACM Press; 1998. p. 101-3. Available from: http://portal.acm.org/citation.cfm?doid=279943.279964.

[29] Chernova S, Veloso M. Interactive Policy Learning through Confidence-Based Autonomy. Journal of Artificial Intelligence Research. 2009 Jan;34:1-25. ArXiv: 1401.3439. Available from: http://arxiv.

org/abs/1401.3439.
[30] Argall BD, Browning B, Veloso MM. Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot. Robotics and Autonomous Systems. 2011;59(3-4):243-55.
[31] Ramachandran D, Amir E. Bayesian Inverse Reinforcement Learning. In: IJCAI. vol. 7; 2007. p. 2586-91.
[32] Mourad N, Ezzeddine A, Nadjar Araabi B, Nili Ahmadabadi M. Learning from demonstrations and human evaluative feedbacks: Handling sparsity and imperfection using inverse reinforcement learning approach. Journal of Robotics. 2020;2020.
[33] LeCun Y, Jackel L, Bottou L, Brunot A, Cortes C, Denker J, et al. Comparison of learning algorithms for handwritten digit recognition. In: International conference on artificial neural networks. vol. 60. Perth, Australia; 1995. p. 53-60.
[34] Nair V, Hinton GE. Rectified linear units improve restricted boltzmann machines. In: ICML; 2010. .
[35] Reddy RVK, Rao BS, Raju KP. Handwritten Hindi digits recognition using convolutional neural network with RMSprop optimization. In: 2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE; 2018. p. 45-51.
[36] Zamir S, Maschler M, Solan E, Hellman Z, Borns M. Game Theory. Cambridge University Press; 2013. Available from: https://books.google.nl/books?id=bmtGvwEACAAJ.
[37] Hockley WE. Analysis of response time distributions in the study of cognitive processes. Journal of Experimental Psychology: Learning, Memory, and Cognition. 1984;10(4):598.
[38] Arumugam D, Lee JK, Saskin S, Littman ML. Deep reinforcement learning from policy-dependent human feedback. arXiv preprint arXiv:190204257. 2019.
[39] Thomaz AL, Hoffman G, Breazeal C. Real-time interactive reinforcement learning for robots. In: AAAI 2005 workshop on human comprehensible machine learning; 2005. p. 9-13.
[40] Warnell G, Waytowich N, Lawhern V, Stone P. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32; 2018. .
[41] Skyrms B. The stag hunt. In: Proceedings and Addresses of the American Philosophical Association. vol. 75. JSTOR; 2001. p. 31-41.