School of Informatics

Informatics Research Review
Reinforcement Learning for Video Game Playing

Lukas Schäfer (s1874970)
January 2019

Abstract

Reinforcement Learning is experiencing novel excitement given the opportunities gained by the breakthrough of deep learning. During this research, games and in particular video games like Atari and StarCraft developed to highly interesting challenges. Their multi-agent reinforcement learning tasks and partial observability in particular are currently addressed in order to widen the possibilities of the vast real-world applications for such behaviour learning approaches. This lead, among others, to the re-discovery of intrinsic rewards in the form of curiosity which seems to be a promising technique for exploratory challenges.

Signature: L. Schäfer

Date: 14 January, 2019

Supervisor: Vesko Cholakov

Contents

1 Introduction

Reinforcement learning (RL) is a category of machine learning covering algorithms capable of learning behaviour in a given environment based on provided feedback and experience. Therefore, the agent usually aims to maximise the cumulative reward gained by behaving correctly. This intuition is in many regards comparable to natural learning of humans and to the basic principle of conditioning applied in animal training. E.g. a dog will be rewarded with treats whenever behaving correctly. Through such simple measures many animals will be able to learn the intended actions rather quickly.

Originally, RL was mostly trial-and-error which gradually improved the agent's information about the environment leading to steady progress in its behaviour. Nowadays, reinforcement learning agents are capable of autonomous vehicle [1] and robot control [2], financial decision making [3], and game-playing beyond human performance [4, 5].

This development is largely thanks to machine learning progress specifically in the field of deep learning, i.e. the usage of highly connected neural networks. Such systems are underlying many omnipresent features of modern devices [6] just as image classification [7], recommendation systems [8] and speech recognition [9]. In the context of RL, neural networks are frequently used to learn complex correlations and features of the environment which can be exploited to derive behaviour.

With the rising capabilities of individual RL agents, research is increasingly interested in multi-agent RL, i.e. training multiple agents to intelligently act together. This collaborative learning can be used to teach cooperative or competitive behaviour [10] in which the agents might communicate or analyse each other. This form of RL is a significant challenge due to the non-stationarity introduced by each agent acting in the environment, which makes most single-agent RL approaches impractical as they either become highly unstable or inefficient. While the challenge of multi-agent RL was already identified and addressed early in RL research [11], it becomes increasingly important as more and more automation occurs in real-world applications which mostly involve more than one active component.

In order to research RL approaches fitting to these challenges, games developed to a major evaluation domain covering diverse problems. Initially, research aimed at board games just as Backgammon [12] and more recently Deepmind succeeded against human professionals in Go [13, 4], a computationally highly challenging task [14]. While these games lead to significant progress in the field of RL, they only involve two players. Modern video games however often require the player to control or reason about the behaviour of much larger quantities of units making them interesting challenges for further RL research leaning towards multi-agent problems.

This research review aims to highlight the primary approaches of reinforcement learning and outline the development from single- to multi-agent behaviour learning. Following, section 3 will analyse the challenge of game-playing and why in particular video games are of interest to RL research. In the end, this work will identify recent research challenges and potential approaches for such in section 4 before concluding with a brief summary.

2 Reinforcement Learning

In order to analyse and develop learning strategies for behaviour, a formalisation for the environment, in which RL agents act in, is required. This is usually defined as a *Markov decision process* (MDP) (S, A, P, R), where S describes a finite set of states, A a set of actions, $P(s' \mid s, a)$ is the probability of reaching state s' by applying action a in state s and $R(s, a)$ describes the reward for applying a in s [15]. RL is concerned with the learning process of an agent acting in such an environment. There are multiple variations how to express and interpret learned behaviour of agents.

One possibility is a *policy* π describing the agent's action selection given a state, where $\pi(a \mid s)$ annotates the probability of choosing action a in state s [16]. $\pi(s)$ can be seen as a probability distribution over all actions for state s.

Secondly, agents can be defined using *value functions*. While a *state-value function* $V_\pi(s)$ describes the expected cumulated reward by following policy π in state s, an *action-value function* $Q_\pi(s, a)$ is assigned the expected cumulated reward by following policy π after applying action a in state s. In these cases, the goal is to estimate the optimal value function which can be described using the *Bellman optimality equations* [16] below

$$V^*(s) = \max_\pi V_\pi(s) = \max_a R(s, a) + \gamma \sum_{s' \in S} P(s' \mid s, a) \cdot V^*(s')$$

$$Q^*(s, a) = \max_\pi Q_\pi(s, a) = R(s, a) + \gamma \sum_{s' \in S} P(s' \mid s, a) \cdot \max_{a'} Q^*(s', a')$$

where $\gamma \in [0, 1]$ denotes the reward-decay factor allowing to balance short- and long-term

rewards by (potentially) discounting future rewards.
In most interesting applications it is unrealistic to obtain these optimal value functions so we aim for precise approximations which gradually converge towards these solutions. In the following paragraphs a few of the most common algorithms, used to compute and improve value functions or learn policies directly, will be outlined.

2.1 Temporal-Difference Learning

First, *Monte-Carlo methods* were used to steadily improve an estimate for value functions, but these only allowed updates after an entire episode, i.e. after a sequence of actions reaching a terminal state. *Temporal-difference (TD) learning* [17] on the other side makes iterative updates after each action application. The update is based on the actual observed reward $R(s,a)$ and the previously expected reward based on $V(s')$ and $V(s)$ itself.

$$V(s) = V(s) + \alpha(R(s,a) + \gamma V(s') - V(s)) \tag{1}$$

α is a constant parameter representing the step-size referred to as *learning rate*. This iterative update schema is the foundation most current (value-based) RL algorithms are built upon.

2.1.1 Q-Learning

Q-learning [18] is one of the most common RL algorithms based on the TD learning update for Q-values. Its update scheme aims to maximise the Q-value for s' as shown in equation (2)

$$Q(s,a) = Q(s,a) + \alpha(R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)) \tag{2}$$

This property allows Q-learning to estimate the optimal action-value function Q^* and makes the approach *off-policy* as a' is directly chosen from the Q-values disregarding any used policy.

Frequently, the algorithm follows an *ϵ-greedy policy* choosing the most promising action obtained by $\operatorname{argmax}_a Q(s,a)$ with probability $1-\epsilon$ and otherwise applies a random applicable action.

Such randomness is important to balance *exploration* and *exploitation*, a common concept in RL. Exploration means to spread out in the state-space by applying new actions while exploitation refers to using the already obtained knowledge about the quality of some actions, i.e. using actions which were already identified as promising. The latter seems to be the better option at first, but without exploration the agent could be stuck just repeating the so-far best actions without ever discovering better behaviour leading to higher rewards.

2.2 Deep Reinforcement Learning

Deep learning has arguably been one of the most influential fields in AI over the last few years. The term refers to the application and training of neural networks which lead to previously unseen performance in a variety of fields [6].

In the context of game-playing and RL, neural networks are frequently used as estimators for value functions of RL algorithms referred to as *deep reinforcement learning.*

2.2.1 Deep Q-Networks

This motivated *Deep Q-Networks* (DQNs), based on Q-learning and convolutional neural networks, which reached human-level performance on multiple Atari games [19, 5]. Such a result was particularly impressive as the approach did not rely on any game-specific prior knowledge. It exclusively received down- and grey-scaled display frames and in-game scores as inputs with latter being used as rewards indicating progress. The exact same network architecture was used for 49 different Atari games from the Arcade Learning Environment [20].

Stable training of Q-networks is achieved using *experience replay* [21], i.e. each step is saved in a *replay memory* D and random samples are used to minimise the following loss function

$$L_i(\theta_i) = \mathbb{E}_{s,a,r,s' \sim U(D)} \left[(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i))^2 \right] \tag{3}$$

where θ_i refers to the Q-network weights in the i-th iteration. The optimisation is achieved using stochastic gradient-descent [22] based on batches collected from D. Despite not relying on any prior knowledge about the Atari 2600 domain, DQNs outperformed any previous RL approach in 43 out of 49 games while reaching human-like scores on 29 games.

Building on DQNs, further modifications were proposed improving the network's training stability and Q-value estimations leading to state-of-the-art performance on various RL domains:

Double Q-Learning Q-Learning approaches are generally prone to overestimate the expected reward due to the usage of the max operation. *Double Q-Learning* [23] aims to resolve this issue by separating the maximisation in action selection and evaluation.

Dueling Network Architecture It was identified that for many states, the action selection can be neglected as it has no relevant impact on the expected reward. In order to identify these cases, the *Dueling Network architecture* [24] was introduced, splitting the action-value $Q(s, a)$ into state-value $V(s)$ and advantage $A(s, a)$ where latter represents the expected reward gain by choosing a over the average reward by any action in s.

Prioritised Experience Replay Schaul et al. introduced a prioritisation strategy for experience replay to give more importance to some highly relevant experience samples in D without letting them dominate the training entirely causing overfitting and unbalanced behaviour [25].

2.2.2 Actor-Critic Reinforcement Learning

While value function approaches are most common in RL, they have the downside of being designed for finite and discrete action sets. Policy methods are generally preferable for an environment including many or continuous actions but they tend to converge slower. Policies are optimised with respect to some score function. These functions e.g. consider the total accumulated reward during an episode as done in REINFORCE [26].

Actor-critic RL approaches aim to solve this downside of policy RL by using a value function as the foundation for their score function [27]. They involve a policy called *actor*, according to which the agent is behaving, and a *critic*, represented by a value function, providing feedback to update the policy. While both these components require optimisation, actor-critic algorithms proved to be efficient and often more stable than individual value function or policy approaches.

One common actor-critic technique is the *asynchronous advantage actor-critic* (A3C) algorithm in which the policy and value function are both represented and trained as neural networks [28]. During training, multiple agents are run in parallel to all update the central model. This asynchronous optimisation makes experience replay unnecessary and proved to be very efficient and flexible. A3C led to impressive performance on a variety of tasks involving discrete and continuous actions just as Atari 2600 games, a car racing simulation, a continuous action control task and a 3D maze challenge.

3 The Challenge of Game-Playing

Looking at the evaluation domains used throughout research, it becomes apparent that game-playing has developed to a significant testbed and challenge for RL. Humans already play games for thousands of years to intellectually challenge each other. It therefore only seems natural to also impose these challenges on RL to identify its capabilities and still prominent shortcomings. Additionally, there is a large diversity of games with varying properties allowing to address specific RL environments.

3.1 Board Games

Already in 1995, RL research came up with TD-Gammon capable of beating humans at the board game backgammon [12] using a multilayer neural network trained by TD learning.

While this early application of RL was successful in this case, these techniques did not fully manifest themselves in game-playing at the time. Further success was mostly achieved using a combination of hand-tailored features, large databases of human-expert moves to pick from and heuristic state-based search. These methods famously lead to Kasparow to be beaten by IBM's Deep Blue in a game of chess [29] in 1997.

One of the most recent game-playing success story based on RL was Google Deepmind's AlphaGo [13, 4] capable of beating human professional players at the Chinese board game Go. This was immensely impressive and prior seen to be decades from reality due to the computational complexity of the game covering almost $3^{19\times19} \approx 10^{170}$ possible board states making any search among it infeasible [14].

3.2 Video Games

While RL research for many years was primarily concerned with board games, video games rose in popularity over the last decades with claimed global gaming market revenue of 137.9 billion US-dollars in 2018[1]. With the success and progress in game-playing on traditional board games and the rise in popularity of video games, research extended game-playing to these domains. The challenge of video games however is novel and often very different due to their complex visual inputs and variety of actions.

Most board games are among others *fully-observable*, i.e. all players have full knowledge about the state of the game, and are often played by two players. Such properties vary considerably among video games. While all Atari games are just for a single player and fully-observable, many video games are designed for multiple players and involve imperfect information. A concept used in many video games is *fog-of-war* where players are unable to determine the exact current state of the game due to limited vision. All these properties make video games highly interesting for further RL research.

Atari 2600 is a collection of Atari games which was one of the first popular video game testbeds for RL research. The games are partly interesting due to their comparably simple structure being single-player games with small visual input and only few possible actions. The

[1] https://newzoo.com (29.11.2018): Newzoo's 2018 Report: Insights Into the $137.9 Billion Global Games Market; `https://newzoo.com/insights/articles/newzoos-2018-report-insights-into-the-137-9-billion-global-games-market/`

collection includes diverse games so that overall convincing performance without game-specific knowledge can be seen as an indication for strong RL performance in general.

However, after reaching human-level performance on most games without any domain-related inputs [5] RL research spread out to more complex and challenging video-games.

Real-Time Strategy (RTS) is a popular gaming genre. These games usually involve gathering resources, resource management, building base structures and finally build up military power to defeat the opposition, usually by destroying their units and buildings [30]. Therefore, an agent requires a variety of skills to properly play such games which are often divided into micro- and macromanagement tasks. While micromanagement refers to low-level control of individual units for short-term rewards, macromanagement is concerned with long-term goals and the high-level strategy of the agent. In order to successfully play such games, a player needs to properly balance its attention to both these tasks.

The games often requires the control of many units, e.g. during combat scenarios. While these can be considered centralised single-agent RL tasks, where a single player controls all units, they can also serve as a domain for multi-agent control learning.

StarCraft: Brood War and StarCraft II are very common RTS video games which received major attention not just in RL but AI research as a whole and from a vast professional, competitive scene [2]. While early research focused on StarCraft based on the BWAPI [3], Deepmind introduced the StarCraft II Learning Environment (SC2LE) in cooperation with the development studio Blizzard [31]. This well-documented interface allows control and access to a variety of information, e.g. multiple feature layer visualisations, making StarCraft II significantly more appealing for future research.

These games are immensely computationally complex. It is estimated that a typical game of StarCraft includes at least 10^{1685} different states making it many magnitudes more complex than the board-game Go [32]. This complexity largely comes from the wide pool of actions available to a player at any point in time. Another challenge involved in StarCraft is the *credit assignment problem* [33], i.e. it is very challenging to identify which choices and actions lead to a win or loss at the end of a game due to their long-term impact. So far, AI agents capable of playing the full game of StarCraft are developed but still heavily rely on domain-specific knowledge and are still far from human professional performance [34].

However, novel RL approaches for multi-agent unit control were recently proposed capable of convincingly beating the in-game AI which is based on domain-specific knowledge. The *counterfactual multi-agent* (COMA) *policy gradients* method [35] seems particularly interesting as it allows to train decentralised policies for multiple agents using a single centralised critic estimating Q-values for joint actions. This critic is represented so that an advantage function for each agent can be efficiently computed in a single forward pass. The credit assignment problem for this critic is also addressed by marginalising out the effect of a single agent's action and fixing the impact of all other agents. Additionally, the approach considers the partial observability included in the game by fog-of-war and limits the vision of each controlled agent.

[2] `https://en.wikipedia.org/wiki/Professional_StarCraft_competition`

[3] The Brood War API, `https://bwapi.github.io/index.html`

4 Research Challenges

While such promising progress in RL research was achieved in a comparably short period of time, there are still many challenges which need to be addressed for AI agents to reach human-level game-playing on complex video games like StarCraft. Particularly the field of multi-agent RL is of interest for future research given its vast set of possible applications.

4.1 Partial observability

As already mentioned, many RL tasks involve partial observability so that an agent is unable to determine the exact state of the environment. This property makes many RL techniques inapplicable, as they rely on the (full) game state, and require the consideration of missing and incomplete information. This becomes especially challenging whenever many agents, which all have different knowledge about the current state, are acting in a multi-agent RL environment.

There has been success in playing limited-information games, e.g. in the case of Poker based on counterfactual regret minimization (CFR) which iteratively approximates a Nash equilibrium from repeated self-play [36]. The COMA policy gradients approach previously mentioned even addresses this challenge for the multi-agent context in the form of small-scale combat scenarios in StarCraft. However, these only involved few units and are still far from combat scenarios occurring in human-level games regarding the number of units and strategic complexity.

Another promising approach for coordinated multi-agent behaviour training is to allow for communication. Bidirectionally-coordinated networks (BiCNets) are bi-directional recurrent neural networks trained with a multi-agent actor-critic technique [37] that allows the agents to remember past states and actions and to receive information from the other agents. The approach lead to state-of-the-art performance in multiple cooperative subtasks in StarCraft.

There is still much room and also the necessity for improvement of multi-agent RL approaches in such environments. Due to the fog-of-war concept and its general complexity, StarCraft formed to be a promising domain for further RL research in this exciting field.

4.2 Intrinsic Rewards

Many RL tasks only provide sparse rewards. E.g. for StarCraft, a positive reward could be assigned once a game is won and a negative reward for lost games. Only receiving this feedback after an entire game makes behaviour learning exceptionally difficult. Additionally, straightforward reward functions do not always exist and have to be designed specifically for a domain. This requires complex considerations based on domain-specific knowledge which should be avoided for the purpose of generalisation and flexibility.

This motivates the idea that RL agents should be able to learn and reason about their behaviour without external reward signals leading to *intrinsic rewards*. While this idea is not novel [38, 39], it received new attention with the upcoming of deep reinforcement learning. Recently, a new approach to curiosity-based intrinsic reward was proposed consisting of two neural networks used to learn an efficient state-representation disregarding irrelevant information and to predict the new state s' after applying an action a in state s [40, 41]. The intrinsic reward was defined as the difference of the state representation of s' and its prediction. Therefore, a large intrinsic reward is assigned whenever the agent was unable to predict the new state motivating exploration in the environment. This approach lead to impressive performance in the complex game Montezuma's revenge exceeding average human playing.

5 Future Work

While intrinsic rewards receive increasing research attention for single-agent RL tasks, it did not yet manifest itself for multi-agent RL applications. Given that curiosity is shown to be an efficient approach to learn exploratory behaviour, research could aim to incorporate such rewards in multi-agent tasks involving partial observability. Specifically in cooperative multi-agent domains with limited knowledge, agents could explore the environment based on their curiosity and communicate such progress with other agents to improve collaborative exploration.

6 Conclusion

This review outlined the development of the research field of reinforcement learning from the introduction of temporal difference learning to modern deep reinforcement learning algorithms just as DQNs and A3C. Differences and problems of the approaches with respect to their efficiency, stability and applicable environments were highlighted. Primarily, this report describes the relevance and flexibility of game-playing as a challenging set of domains for RL research focusing on video games which become increasingly important. The rising relevance of these domains is due to their variety of properties. While Atari games are comparably simple tasks, playing modern video games like StarCraft is highly challenging as the game involves partial-observability and a large quantity of actions and agents.

The number of real-world applications for multi-agent RL approaches makes these games interesting for future research. For this purpose, two major challenges with a focus on multi-agent tasks were identified as partial observability and intrinsic rewards. To conclude, curiosity-driven exploration was proposed as a possible approach to these tasks.

References

[1] Markus Kuderer, Shilpa Gulati, and Wolfram Burgard. Learning driving styles for autonomous vehicles from demonstration. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 2641–2646. IEEE, 2015.

[2] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[3] James J Choi, David Laibson, Brigitte C Madrian, and Andrew Metrick. Reinforcement learning and savings behavior. *The Journal of finance*, 64(6):2515–2534, 2009.

[4] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of go without human knowledge. *Nature*, 550:354–, October 2017.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, February 2015.

[6] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

[7] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, pages 1097–1105, USA, 2012. Curran Associates Inc.

[8] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2643–2651. Curran Associates, Inc., 2013.

[9] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[10] Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.

[11] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the tenth international conference on machine learning*, pages 330–337, 1993.

[12] Gerald Tesauro. Temporal difference learning and td-gammon. *Commun. ACM*, 38(3):58–68, March 1995.

[13] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489, January 2016.

[14] Martin Müller. Computer go. *Artificial Intelligence*, 134(1-2):145–179, 2002.

[15] Ronald A Howard. Dynamic programming and markov processes. 1964.

[16] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[17] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[18] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. In *Machine Learning*, pages 279–292, 1992.

[19] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. In *NIPS Deep Learning Workshop*. 2013.

[20] Marc G. Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *J. Artif. Int. Res.*, 47(1):253–279, May 2013.

[21] Long-Ji Lin. Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine learning*, 8(3-4):293–321, 1992.

[22] Herbert Robbins and Sutton Monro. A stochastic approximation method. In *Herbert Robbins Selected Papers*, pages 102–109. Springer, 1985.

[23] Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, pages 2094–2100. AAAI Press, 2016.

[24] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Freitas. Dueling network architectures for deep reinforcement learning. *arXiv preprint arXiv:1511.06581*, 2015.

[25] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[26] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[27] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[28] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[29] Murray Campbell, A. Joseph Hoane Jr., and Feng hsiung Hsu. Deep blue. *Artif. Intell.*, 134(1-2):57–83, January 2002.

[30] Santiago Ontanón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, and Mike Preuss. A survey of real-time strategy game ai research and competition in starcraft. *IEEE Transactions on Computational Intelligence and AI in games*, 5(4):293–311, 2013.

[31] Oriol Vinyals, Timo Ewalds, Sergey Bartunov, Petko Georgiev, Alexander Sasha Vezhnevets, Michelle Yeo, Alireza Makhzani, Heinrich Küttler, John Agapiou, Julian Schrittwieser, John Quan, Stephen Gaffney, Stig Petersen, Karen Simonyan, Tom Schaul, Hado van Hasselt, David Silver, Timothy P. Lillicrap, Kevin Calderone, Paul Keet, Anthony Brunasso, David Lawrence, Anders Ekermo, Jacob Repp, and Rodney Tsing. Starcraft II: A new challenge for reinforcement learning. *CoRR*, abs/1708.04782, 2017.

[32] Nicolas Usunier, Gabriel Synnaeve, Zeming Lin, and Soumith Chintala. Episodic exploration for deep deterministic policies: An application to starcraft micromanagement tasks. *arXiv preprint arXiv:1609.02993*, 2016.

[33] Richard Stuart Sutton. Temporal credit assignment in reinforcement learning. 1984.

[34] Michal Certicky and David Churchill. The current state of starcraft ai competitions and bots. In *AIIDE 2017 Workshop on Artificial Intelligence for Strategy Games*, volume 10, 2017.

[35] Jakob N Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[36] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up limit hold'em poker is solved. *Science*, 347(6218):145–149, 2015.

[37] Peng Peng, Quan Yuan, Ying Wen, Yaodong Yang, Zhenkun Tang, Haitao Long, and Jun Wang. Multiagent bidirectionally-coordinated nets for learning to play starcraft combat games. *arXiv preprint arXiv:1703.10069*, 2017.

[38] Jürgen Schmidhuber. Curious model-building control systems. In *Neural Networks, 1991. 1991 IEEE International Joint Conference on*, pages 1458–1463. IEEE, 1991.

[39] Nuttapong Chentanez, Andrew G Barto, and Satinder P Singh. Intrinsically motivated reinforcement learning. In *Advances in neural information processing systems*, pages 1281–1288, 2005.

[40] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International Conference on Machine Learning (ICML)*, volume 2017, 2017.

[41] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.