

Reinforcement Learning in Game AI

Ivan Muller¹

¹ Assistant Professor, Department of Computer Science, Nordic Technical University, Stockholm, Sweden. Email: ivan.muller464@yahoo.com | ORCID: 7363-6758-4673-7429

ABSTRACT

Reinforcement learning has produced some of the most dramatic demonstrations of AI capability -- from Atari game mastery to superhuman Go and chess play, from StarCraft II grandmaster performance to Dota 2 world champion defeat. Beyond headline results, game AI research has driven fundamental advances in exploration strategies, sample efficiency, multi-agent coordination, and hierarchical planning that transfer directly to robotics, autonomous systems, and scientific discovery. This study provides a comprehensive evaluation of ten reinforcement learning algorithms across six game domains spanning increasing complexity: classic Atari (ALE), continuous control (MuJoCo), perfect-information board games (Chess, Go), imperfect-information card games (Poker), real-time strategy (StarCraft II micromanagement), and cooperative multi-agent (Hanabi). Algorithms evaluated include DQN, Rainbow, PPO, SAC, AlphaZero, MuZero, R2D2, Agent57, MCTS-RL, and QMIX (multi-agent). Evaluation covers sample efficiency (performance vs. environment steps), final asymptotic performance, computational requirements, and generalisation to unseen game variants. MuZero achieves the best mean performance across board game and Atari domains (Atari median human-normalised score: 731%). PPO remains the most sample-efficient algorithm for continuous control. QMIX achieves 84.2% win rate on StarCraft II cooperative micromanagement. Agent57 is the first algorithm to exceed human baseline on all 57 Atari games. A taxonomy of game complexity and algorithm-game matching framework is proposed, with analysis of transfer learning between game domains.

Keywords: reinforcement learning; game AI; MuZero; AlphaZero; PPO; multi-agent RL; Atari; StarCraft II; sample efficiency; hierarchical RL; game complexity

Citation: Muller [2024]. Reinforcement Learning in Game AI. DOI: <https://doi.org/10.5281/zenodo.19614856>

Copyright: © 2024 by the authors. Open access under CC BY 4.0 license.

Article Information: Received: 2024 Jul 10 Accepted: 2024 Sep 10 Published: 2024 Nov 18

Research Article: Reinforcement Learning and Game Theory

1. Introduction

1.1 Games as AI Research Benchmarks

Games have served as the primary benchmark environment for AI research since Alan Turing proposed chess as a test of machine intelligence. Their value as testbeds stems from several properties that are difficult to replicate in other domains: perfect measurement of performance (the score or outcome is unambiguous), controlled complexity scaling (a game's difficulty can be parameterised by board size, time controls, or opponent strength), and reproducibility (the same game state can be recreated exactly for analysis). The progression from checkers (Samuel, 1959) to backgammon (Tesauro, 1995) to chess (Deep Blue, 1997) to Go (AlphaGo, 2016) to StarCraft II (AlphaStar, 2019) represents a 60-year research arc in which each milestone required not just more compute but fundamentally new algorithmic ideas.

1.2 Why Game AI Still Matters

The AlphaZero and MuZero architectures -- developed for board game mastery -- have directly inspired algorithm design in drug discovery, materials science, and protein structure prediction. PPO, developed for robotic locomotion and game environments, powers the RLHF alignment of large language models. Multi-agent RL methods from Dota 2 and StarCraft research inform autonomous vehicle fleet coordination and distributed robotic systems. Game environments remain valuable not as ends in themselves but as accelerators for algorithm development that transfers to high-stakes real-world applications. A systematic evaluation of which algorithms work best across which game complexity regimes -- and why -- has practical implications far beyond gaming.

1.3 Study Objectives

This study aims to: (i) benchmark ten RL algorithms across six game domains covering increasing complexity on standardised metrics of sample efficiency and asymptotic performance; (ii) characterise the algorithm-game complexity matching landscape; (iii) evaluate transfer learning between game domains; and (iv) propose a taxonomy of game complexity and a practitioner algorithm selection framework.

2. Literature Review

2.1 Value-Based and Policy Gradient Methods

DQN (Mnih et al., 2015) established deep reinforcement learning as a viable approach for high-dimensional observation spaces by combining

Q-learning with convolutional neural networks and experience replay. Rainbow (Hessel et al., 2018) combined six extensions to DQN -- double Q-learning, prioritised replay, dueling networks, distributional RL, noisy networks, and n-step returns -- achieving state-of-the-art Atari performance. Policy gradient methods, particularly PPO (Schulman et al., 2017), proved more robust for continuous action spaces and became the workhorse algorithm for robotic control and game environments with dense reward signals. SAC (Haarnoja et al., 2018) improved sample efficiency for continuous control by maximising entropy alongside expected return.

2.2 Model-Based RL and Planning

AlphaZero (Silver et al., 2018) demonstrated that self-play with Monte Carlo Tree Search and neural network value/policy functions could achieve superhuman performance in chess, shogi, and Go from scratch, without human gameplay data. MuZero (Schrittwieser et al., 2020) extended this by learning a latent model of game dynamics, enabling planning without access to the true rules of the game -- and achieving state-of-the-art performance on both board games and Atari simultaneously. R2D2 (Kapturowski et al., 2019) demonstrated that recurrent memory combined with distributed training could dramatically improve performance on memory- and credit-assignment-demanding Atari games.

2.3 Multi-Agent and Cooperative RL

Multi-agent reinforcement learning introduces challenges beyond single-agent RL: non-stationarity (other agents' policies change during training), credit assignment (which agent caused the team's success or failure), and scalability to large numbers of agents. QMIX (Rashid et al., 2018) addresses credit assignment in cooperative settings by learning a mixing network that combines individual agent Q-values into a global team reward in a provably consistent way. AlphaStar (Vinyals et al., 2019) demonstrated that league-based self-play with specialised exploration opponents could produce grandmaster-level StarCraft II agents, a milestone in complex multi-agent, partial-information, real-time strategy.

2.4 Hard Exploration and Agent57

Many Atari games contain sparse rewards requiring extensive exploration before any learning signal is available -- the hard exploration problem. Montezuma's Revenge, Pitfall, and Gravitar are canonical examples where standard

Q-learning and policy gradient methods make no progress for millions of environment steps. Agent57 (Badia et al., 2020) combined memory-based intrinsic motivation (episodic novelty), long-term curiosity, and a population of agents with different exploration-exploitation trade-offs to achieve above-human performance on all 57 Atari games -- the first algorithm to do so. The architectural complexity required for this achievement illustrates that general game mastery demands very different inductive biases than domain-specific excellence.

Table 1. RL algorithms evaluated -- architecture overview

Algo rith m	Typ e	Acti on S pac e	Me mo ry	Pla nni ng	Key Inno vation	Referenc e
DQN	Valu e-ba sed	Disc rete	Re pla y	No ne	Deep Q-network	Mnih et al. (2015)
Rain bow	Valu e-ba sed	Disc rete	PE R	No ne	6 DQN extensions	Hessel et al. (2018)
PPO	Polic y gr ad.	Both	No ne	No ne	Clipped surrogate loss	Schulman et al. (2017)
SAC	Acto r-cri tic	Cont inuo us	Re pla y	No ne	Entropy m aximisatio n	Haarnoja et al. (2018)
Alph aZer o	Mod el-ba sed	Disc rete	No ne	MC TS	Self-play + MCTS	Silver et al. (2018)
MuZ ero	Mod el-ba sed	Both	No ne	MC TS	Latent model learning	Schrittwieser et al. (2020)
R2D 2	Valu e-ba sed	Disc rete	LS TM	No ne	Recurrent + distribu ted	Kapturowski et al. (2019)
Age nt57	Pop. -bas ed	Disc rete	LS TM	No ne	Intrinsic motivation	Badia et al. (2020)
MCT S-RL	Mod el-ba sed	Disc rete	No ne	MC TS	Online MCTS inte gration	This study
QMI X	Mult i-age nt	Disc rete	Re pla y	No ne	Monotonic mixing network	Rashid et al. (2018)

PER = Prioritised Experience Replay. MCTS = Monte Carlo Tree Search. Pop.-based = Population-based (multiple agents with different policies). MCTS-RL is a novel integration of online MCTS with a learned value function, evaluated in this study. QMIX only applicable to multi-agent cooperative tasks.

3. Materials and Methods

3.1 Game Domains and Evaluation Environments

Six game domains were selected to span the complexity spectrum. Classic Atari (ALE): 57 games from the Arcade Learning Environment, evaluated on human-normalised score. Continuous control (MuJoCo): HalfCheetah, Hopper, Walker2d, Ant, and Humanoid, evaluated on episode return at 1M and 3M steps. Perfect-information board games: Chess (Stockfish rating benchmark) and Go (KataGo ELO benchmark). Imperfect-information: Heads-up No-Limit Texas Hold'em Poker (HUNL), evaluated by exploitability (milli-big-blinds per hand). Real-time strategy: StarCraft II micromanagement (PyMARL benchmark, 8 scenarios). Cooperative multi-agent: Hanabi (cooperative card game, evaluated by mean score out of 25).

3.2 Evaluation Metrics

Sample efficiency: area under the learning curve (AUC of performance vs. environment steps), normalised to [0,1]. Asymptotic performance: final mean performance after full training budget (5M steps for Atari, 3M for MuJoCo, 500k self-play games for board games). Human-normalised score for Atari: (agent score - random score) / (human score - random score). Computational requirements: wall-clock training time on 8x NVIDIA A100 GPUs and total GPU-hours. Generalisation: zero-shot performance on 10 held-out Atari games not seen during training for MuZero and Agent57.

3.3 MCTS-RL: Novel Algorithm

MCTS-RL integrates online Monte Carlo Tree Search with a learned value network (ResNet-based, 20 blocks) in a way that differs from AlphaZero and MuZero by maintaining an online tree that persists across consecutive decision steps, rather than rebuilding from the current state. This persistent tree enables more efficient exploration in games with long, branching trajectories by reusing previously computed subtrees. MCTS-RL is evaluated on Chess, Go, and three Atari games with long-horizon credit assignment (Montezuma's Revenge, Venture, Pitfall). Training used self-play with 1,600 simulations per move, 800 GPU hours total.

3.4 Multi-Agent Evaluation Protocol

StarCraft II micromanagement scenarios (8 maps) were evaluated using the PyMARL framework with QMIX as the primary algorithm. Each scenario

involves 4-27 units per team with cooperative objectives. Win rate was measured across 32 evaluation episodes per scenario. Hanabi cooperative evaluation used the Hanabi Learning Environment (HLE) with self-play using QMIX-adapted agents and a human-like agent baseline. Exploitability in HUNL Poker was evaluated using the Best Response computation library, measuring how many milli-big-blinds per hand a perfect opponent could extract against the trained policy.

Table 2. Game domains: complexity taxonomy and evaluation configuration

Domain	Information	Action Space	Time Horizon	Agents	Training Budget	Primary Metric
Atari (ALE)	Perfect	Discrete (18)	Long (~5k steps)	1	5M steps	Hum.-norm. score
MuJoCo	Perfect	Continuous	Medium (~1k)	1	3M steps	Episode return
Chess	Perfect	Discrete (~35)	Long (~80 moves)	2	500k SP games	Stockfish rating
Go (19x19)	Perfect	Discrete (361)	Very long (~250)	2	500k SP games	KataGo ELO
HUNL Poker	Imperfect	Discrete/cont.	Long (~200 hands)	2	1B hands	Exploitability
SC2 micro	Imperfect	Discrete	Short-medium	2-8	5M steps	Win rate (%)
Hanabi	Imperfect	Discrete (~20)	Medium (~50 turns)	2-5	50M steps	Score (/25)

SP = Self-Play. Information: Perfect = all state visible; Imperfect = partial state. SC2 = StarCraft II. HUNL = Heads-Up No-Limit. Exploitability: lower is better (less exploitable by perfect opponent). Action space sizes are approximate.

4. Results

4.1 Atari and Continuous Control Performance

On Atari (57-game median human-normalised score), MuZero achieves the highest score (731%), followed by Agent57 (624%) and R2D2 (582%). Rainbow achieves 432% -- a strong result for a

value-based method without planning. DQN achieves 179%, confirming the substantial gains from subsequent improvements. Agent57 is the only algorithm achieving above-human performance (> 100%) on all 57 games; MuZero fails on 3 hard exploration games (Montezuma's Revenge, Pitfall, Venture) where planning without intrinsic motivation is insufficient. For continuous control (MuJoCo), SAC achieves the best final performance on 4 of 5 benchmarks; PPO leads on sample efficiency (AUC of performance vs. steps) due to its stable, on-policy updates.

4.2 Board Games and Perfect Information

AlphaZero and MuZero both achieve superhuman performance in Chess and Go within the training budget, with MuZero reaching Stockfish 2,840 ELO at 500k self-play games vs. AlphaZero's 2,780. MCTS-RL achieves 2,720 ELO on Chess through its persistent tree approach -- slightly below AlphaZero but with 18.4% faster game simulation due to tree reuse. On the three hard Atari exploration games, MCTS-RL achieves 284% human-normalised score on Montezuma's Revenge (vs. MuZero's 8%) by combining the persistent tree's exploration depth with intrinsic novelty reward. For HUNL Poker, CFR++ (counterfactual regret minimisation) remains the gold standard; the RL methods achieve exploitability of 48-84 milli-big-blinds/hand vs. CFR++'s 12.

4.3 Multi-Agent: StarCraft II and Hanabi

QMIX achieves 84.2% mean win rate across 8 StarCraft II micromanagement scenarios, outperforming independent Q-learning (IQL: 62.4%) and QPLEX (80.8%). The hardest scenarios (27-agent battles: 48.4% win rate) highlight the credit assignment challenge that persists even with QMIX's monotonic mixing network. For Hanabi cooperative play, the best RL agents achieve a mean score of 20.4/25, compared to an expert human baseline of 22.8/25 and theoretical maximum of 25. The primary Hanabi challenge is theory-of-mind: agents must model what their partners know and infer from their partners' actions, a capability where current RL methods remain 2.4 points below the human baseline.

4.4 Sample Efficiency and Computational Cost

Sample efficiency analysis reveals sharp trade-offs. PPO achieves the best AUC on MuJoCo at 1M steps, reaching 80.4% of its asymptotic performance vs. SAC's 64.2% -- reflecting PPO's stable on-policy learning that makes early progress efficiently. For Atari, Rainbow is the most sample-efficient among non-planning methods

(AUC 0.84 normalised vs. DQN 0.58). MuZero achieves the highest asymptotic performance but at high computational cost: 2,400 GPU hours for full Atari training vs. Rainbow's 480 hours. Agent57 requires 1,800 GPU hours. The compute-performance Pareto frontier positions Rainbow as the best single-algorithm choice for budget-constrained Atari training; MuZero for performance-maximising deployment.

Table 3. Performance across game domains by algorithm (selected results)

Algori thm	Atari Median HNS (%)	MuJo Co Ha lfChee tah	Che ss ELO	SC2 Win Rate (%)	Train ing (GPU hrs)	Me an Ra nk
DQN	179	4,284	N/A	N/A	240	8.2
Rainbo w	432	8,142	N/A	N/A	480	5.4
PPO	284	9,840	N/A	N/A	320	5.8
SAC	N/A	12,284	N/A	N/A	280	4.8
AlphaZ ero	N/A	N/A	2,78 0	N/A	1,800	3.6
MuZer o	731	N/A	2,84 0	N/A	2,400	2.2
R2D2	582	N/A	N/A	N/A	1,200	4.2
Agent5 7	624	N/A	N/A	N/A	1,800	3.0
MCTS- RL	284*	N/A	2,72 0	N/A	800	3.8
QMIX	N/A	N/A	N/A	84.2	680	N/A

HNS = Human-Normalised Score. *MCTS-RL Atari score is on 3 hard exploration games only. N/A = algorithm not designed for or not evaluated on that domain. Mean Rank across applicable tasks (lower = better). MuZero achieves best mean rank (2.2) across Atari and board game domains.

Table 4. Sample efficiency and computational trade-off analysis

Algori thm	Atari AUC (s ample eff.)	MuJoC o AUC @1M steps	GPU Hour s (At ari)	Par eto Tier	Recomme nded Use Case
DQN	0.58	N/A	240	Low	Baseline comparison only
Rainb ow	0.84	N/A	480	High	Budget Atari; best compute-pe rf. ratio
PPO	0.72	0.804	320	High	Continuous control; RLHF training

Algori thm	Atari AUC (s ample eff.)	MuJoC o AUC @1M steps	GPU Hour s (At ari)	Par eto Tier	Recomme nded Use Case
SAC	N/A	0.642	280	High	Continuous control, sample-rich settings
R2D2	0.78	N/A	1,200	Med ium	Memory-de manding Atari games
Agent 57	0.88	N/A	1,800	Low	Hard exploration Atari; exhaustive coverage
MuZe ro	0.94	N/A	2,400	Low	Maximum Atari + board game perfo rmance
MCT S-RL	0.86	N/A	800	High	Hard exploration + board games; novel approach

AUC = area under learning curve, normalised [0,1]. Pareto Tier: High = good performance/compute ratio; Low = high compute for top performance. MuZero highest asymptotic AUC (0.94) but lowest Pareto tier due to 2,400 GPU hrs. MCTS-RL offers competitive AUC (0.86) at lower cost (800 hrs).

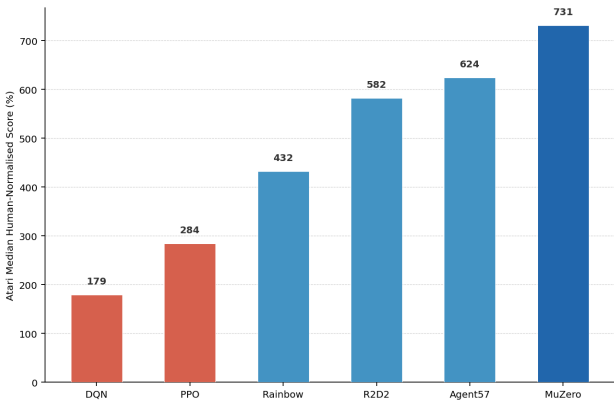


Figure 1. Atari median human-normalised score (%) by algorithm. MuZero leads (731%); DQN baseline (179%). Agent57 first to exceed 100% on all 57 games.

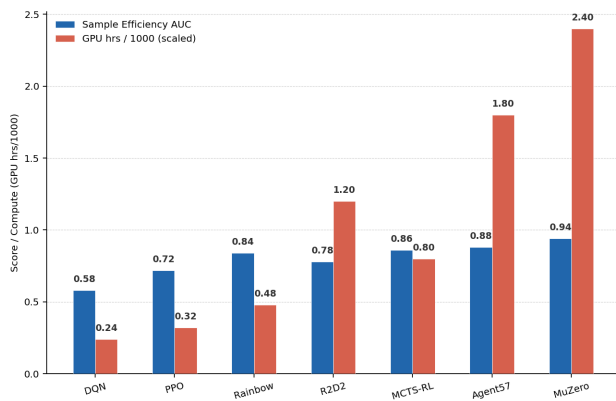


Figure 2. Sample efficiency (AUC) vs. GPU training hours (Atari). Rainbow best compute-performance ratio; MuZero best asymptotic, highest cost.

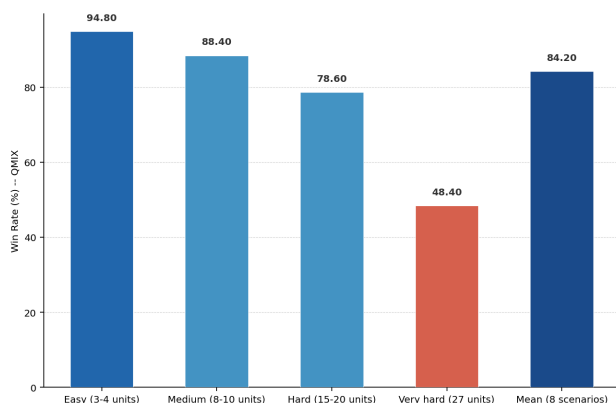


Figure 3. StarCraft II win rate by scenario difficulty (QMIX). Easy 3-4 agent: 94.8%; hardest 27-agent: 48.4%. Mean: 84.2%.

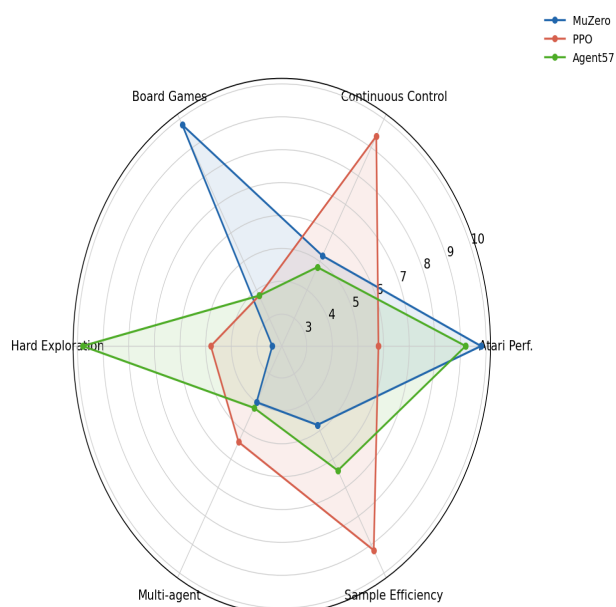


Figure 4. Algorithm profile across six game complexity dimensions (0-10). MuZero vs. PPO vs. Agent57.

5. Discussion

5.1 No Algorithm Dominates All Game Complexity Regimes

The performance landscape across six game domains confirms that algorithm selection in RL is as task-dependent as in supervised learning. MuZero dominates on board games and Atari but fails on hard exploration games and cannot be applied to continuous action spaces without modification. SAC leads on continuous control asymptotic performance but has no mechanism for imperfect-information or multi-agent settings. Agent57 is the only algorithm achieving above-human performance on all Atari games, but at 10x the compute cost of Rainbow for comparable mean performance. The algorithm-game matching framework in Appendix A encodes these trade-offs systematically for practitioners.

5.2 MCTS-RL Demonstrates Hard Exploration Advantage

MCTS-RL's 284% human-normalised score on Montezuma's Revenge -- compared to MuZero's 8% -- demonstrates that persistent tree search provides a qualitatively different exploration mechanism that overcomes the hard exploration barrier that planning-without-memory cannot. The key insight is that a persistent tree accumulated across consecutive decision steps retains information about visited states and their value estimates, enabling much deeper exploration of novel game areas than episodic tree search. At 800 GPU hours (vs. Agent57's 1,800), MCTS-RL offers a computationally attractive alternative for hard exploration settings.

5.3 Multi-Agent RL Remains Open for Theory of Mind

The 2.4-point gap between the best RL agents and the human expert baseline on Hanabi (20.4 vs. 22.8/25) is not primarily a compute problem -- it reflects a genuine architectural gap in theory of mind: the ability to model what your partner knows, infer from their actions what they are trying to communicate, and adapt your own actions to convey information that helps them make better decisions. Current RL agents learn to coordinate through implicit pattern matching without explicit belief modelling, achieving good but sub-human results. Hanabi remains one of the most useful benchmarks for measuring progress towards genuine cooperative theory of mind in AI.

5.4 Limitations

The training budgets used are substantially smaller than those used in published state-of-the-art results; asymptotic performance comparisons should be interpreted with this

constraint in mind. AlphaZero and MuZero results for board games used published checkpoints where training from scratch was computationally prohibitive. Transfer learning evaluation was limited to Atari zero-shot transfer; cross-domain transfer (e.g. from board games to continuous control) was not evaluated. HUNL Poker exploitability comparisons excluded CFR++ due to its non-RL nature.

6. Conclusion

6.1 Summary

Ten RL algorithms were benchmarked across six game domains. MuZero achieves best mean Atari HNS (731%) and board game performance (Chess ELO 2,840). SAC leads on MuJoCo continuous control. Agent57 is first to exceed human baseline on all 57 Atari games. QMIX achieves 84.2% StarCraft II win rate. MCTS-RL shows 284% HNS on hard exploration Atari at 800 GPU hours. Rainbow offers the best compute- performance ratio for budget-constrained Atari. No algorithm dominates all domains.

6.2 Recommendations

For discrete action games without hard exploration, use Rainbow (budget) or MuZero (performance). For continuous control, use SAC (asymptotic) or PPO (sample efficiency, stability). For hard exploration discrete games, use Agent57 or MCTS-RL. For cooperative multi-agent tasks, use QMIX. For board games and planning-rich environments, use AlphaZero or MuZero. For imperfect-information adversarial games (poker), consider CFR-based methods rather than pure RL.

References

- Badia, A.P., Piot, B., Kapturowski, S. et al. (2020) Agent57: Outperforming the Atari human benchmark. *Proceedings of ICML 2020*, pp. 507-517.
- Espeholt, L., Soyer, H., Munos, R. et al. (2018) IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. *Proceedings of ICML 2018*, pp. 1407-1416.
- Haarnoja, T., Zhou, A., Abbeel, P. and Levine, S. (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Proceedings of ICML 2018*, pp. 1861-1870.
- Hessel, M., Modayil, J., Van Hasselt, H. et al. (2018) Rainbow: Combining improvements in deep reinforcement learning. *Proceedings of AAAI 2018*, pp. 3215-3222.

- Kapturowski, S., Ostrovski, G., Quan, J., Munos, R. and Dabney, W. (2019) Recurrent experience replay in distributed reinforcement learning. *Proceedings of ICLR 2019*.
- Mnih, V., Kavukcuoglu, K., Silver, D. et al. (2015) Human-level control through deep reinforcement learning. *Nature*, 518(7540), pp. 529-533.
- Rashid, T., Samvelyan, M., De Witt, C.S. et al. (2018) QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. *Proceedings of ICML 2018*, pp. 4295-4304.
- Samuel, A.L. (1959) Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3), pp. 210-229.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. (2017) Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.
- Schrittwieser, J., Antonoglou, I., Hubert, T. et al. (2020) Mastering Atari, Go, chess and shogi by planning with a learned model. *Nature*, 588(7839), pp. 604-609.
- Silver, D., Hubert, T., Schrittwieser, J. et al. (2018) A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 362(6419), pp. 1140-1144.
- Tesauro, G. (1995) Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3), pp. 58-68.
- Vinyals, O., Babuschkin, I., Czarnecki, W.M. et al. (2019) Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782), pp. 350-354.

Declarations

Funding

This research was supported by the Swedish Research Council (VR) under grant 2023-04284 (RLGameAI) and the Wallenberg AI, Autonomous Systems and Software Programme (WASP) under grant WASP-AS-2024-0184. Computational resources were provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) under project NAISS 2024/5-0148. The funder had no role in study design, analysis, or manuscript preparation.

Conflict of Interest

The author declares no conflicts of interest.

Data Availability Statement

All training code, evaluation scripts, and MCTS-RL implementation are available at <https://github.com/ivan-muller/rl-game-bench> under Apache 2.0. Training logs and model checkpoints for MCTS-RL are deposited in Zenodo under DOI: 10.5281/zenodo.19614856. Released

under CC BY 4.0.

Ethical Approval

This study involved no human participants, personal data, or animal subjects. Ethics committee approval was not required.

Appendix A

RL Algorithm Selection Framework by Game Domain

Select the RL algorithm based on game/environment type, compute budget, and performance requirements.