

Project Plan

Using Reinforcement Learning and Algorithmic Techniques to Develop Strategies for Tournament-Style Backgammon Matches

Will Farmer supervised by Robert Powell

Project Description

While numerous AI models exist for backgammon games, they are all developed with the idea of winning a single, best-of-one game. The aim of my project is to develop a Reinforcement Learning model, alongside various heuristic and optimisation algorithms, which attempts to win matches that may be first to 5 points, or 7 points, or even longer games. The key differences lies in the doubling cube, victory by gammon and victory by backgammon. The doubling cube can be used by a player at any point in the game. If accepted by the opposing player, the victor of that game will earn as many points as normal multiplied by the face value of the doubling cube (starting at 2, doubling each time it is played). For example, if the value on the doubling cube is 8 and a player wins by backgammon, they will earn 24 points. If a player declines the doubling cube offer, then they resign the game, and the offering player wins 1 point. Hence, a good doubling strategy is paramount in tournament-style matches and thus shall be including the Reinforcement Learning model's training process.

Deliverables

Basic

- A robust game environment
- A functional random agent
- A GUI

Intermediate

- An agent based on a genetic algorithm
- An agent based on the Ant Colony Optimisation algorithm (Novel)
- An agent based on a greedy algorithm

Advanced

- A Reinforcement Learning model that acts as an agent
- A Machine Learning model that acts as an agent, e.g. 1+1 Hill Climbing [1]
- An agent that changes strategy based on the state of the game

Project Timeline

Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Christmas Hols					
14/10/2024	21/10/2024	28/10/2024	04/11/2024	11/11/2024	18/11/2024	25/11/2024	02/12/2024	09/12/2024	16/12/2024	23/12/2024	30/12/2024	06/01/2025		
Initial Research	Game engine dev		GUI		ACO		Presentation Prep		Presentation				Coursework	
	Project Plan + Lit Review		Greedy		Genetic		User Testing		Coursework					
Week 11	Week 12	Week 13	Week 14	Week 15	Week 16	Week 17	Week 18	Week 19	Week 20	Easter hols				
13/01/2025	20/01/2025	27/01/2025	03/02/2025	10/02/2025	17/02/2025	24/02/2025	03/03/2025	10/03/2025	17/03/2025	24/03/2025	31/03/2025	07/04/2025	14/04/2025	21/04/2025
Coursework	Advanced Deliverables		Demo Prep		Demonstration	Remaining Advanced		User Testing	Submit draft to Rob	Make corrections				
Report	Report		Coursework	User Testing	Report					Coursework and Exam Revision				

Following discussion with my supervisor I had determined that this is an ideal but realistic timeline for my project. Based on how long it took me to complete a credit-worth of coursework last year on average I have placed blockers in my timeline so I can dedicate time solely to meeting other modules' deadlines. I note that in the week commencing 28/04/2025 I will be submitting my final report as well as finalising my Viva presentation which will occur in the following week.

Literature Survey

Definitions [2]

- Checker: A piece which is moved around the board by the players. Each player has 15 checkers.
- Point (on the board): A place on the board where a checker may be moved to and occupy
- Points: The score of a player. Given to the victor of a game.
- Home board: The final six points on the board for each player. A player aims to move all of their checker to their home board.
- Bear (past: Borne) Off: When a player has moved all of their checkers into their home board, they may start to bear off pieces. Once all pieces have been borne off, they are the victor.
- Bar: The place to store checkers that have been hit
- Blot: A single checker alone on a point
- Hit: Move a checker onto an opponent's blot, forcing the opponent's checker to the bar with your checker replacing them on the point
- Gammon: Yields double points to the victor. Occurs when losing player has not borne off any checkers.
- Backgammon: Yields triple points to the victor. Occurs when the losing has not borne off any checkers and still has at least one checker on the bar or in the opposition's home board
- Doubling Cube: Used to raise the stakes of the game. Its six faces contain the numbers two, four, eight, sixteen, thirty-two and sixty-four. A player can offer the doubling cube to make the value of the game be multiplied by the number on the cube. If rejected, the game is resigned and the player who offered the cube gains one point. If accepted, the offered player now controls the doubling cube.
- Enter: The act of moving a piece off the bar onto the opposition's home board.
- Count: The sum of distances remaining for each player's checker.
- Doubling Point: The probability of winning when a player plays the Doubling Cube.

Review

I first came about the idea of including the Doubling Cube in the reinforcement learning model when reading [3]. The idea that a highly successful model based on reinforcement learning used a heuristic algorithm for deciding when to double surprised me. This led me to read [4], which declared that the optimal time Player A should play the doubling cube is when one has a doubling point of 0.8 and that this should be rejected by B only when B's probability of winning is 0.25 or less. This alters during the phase of bearing off and when the doubling cube has been already played [4], but against an easier opponent it is suggested that doubling earlier is more optimal. It should also be noted that Tesauro did not include backgammons in the output vector due to their incredible rarity. While true that these circumstances are rare, this still surprised me as TD-Gammon 2.1 was trained on approximately 1,500,000 games [3], in which scenario the likelihood of the game ending in a backgammon at least once is

extremely high, and, especially when combined with the doubling cube, can be devastating for an opponent. Hence, in my neural network, I shall have a 6-dimensional vector output representing each colour (black and white) winning, winning by gammon and winning by backgammon. My hope is that this difference, combined with the issue of the doubling cube, will create variation in strategy from other models aimed at playing backgammon, and will hopefully perform better in tournament-style matches as a result of its longer-term 'vision'.

However, it is worth noting that Neurogammon [5] [6] used a neural-network for determining when to double (and a set of 6 other networks for the rest of the game) and yet has been universally less successful than TD-Gammon. While this is true," the consensus of expert opinion at the Olympiad was that Neurogammon's doubling algorithm was probably the strongest part of the program." [6]. The algorithm developed was also only put together very quickly before the Olympiad, using a "crude nine-point scale", similar to that in [4]. If a crude, quickly put-together model could create an algorithm so strong, it is possible that a well-crafted model could produce a more optimal algorithm, as is my goal. Whether I use supervised learning (like Neurogammon) or unsupervised learning for this network is not yet decided, however I am leaning towards using unsupervised learning like the rest of the model.

When making a learning model, it is necessary, of course, that there is material for the algorithm to learn from. In the scenario of backgammon, that is an opponent. In TD-Gammon, the model learnt from self-play [3], which meant that when the model improved in ability, the model would then have a more skilled opponent to play against and learn how to beat. This perpetuates until the model eventually converges. This method of self-play is one of the factors that allowed TD-Gammon to reach such a high level of skill. However, there were hundreds – if not thousands – of games with hundreds or thousands of time-steps, when in a typical human-played games are only comprised of 50-60 time steps. These games were effectively two random agents playing against each other until the model eventually learnt basic strategy.

Conversely, Neurogammon was trained against a sole opponent [6]. This meant that the skill increase at the start was much faster than with TD-Gammon, however the skill ceiling was reached much earlier, and the strategy was very 'inbred' [6] – it is highly possible that the model developed a strategy specifically designed to exploit weaknesses in its opponent's strategy. This is potentially why it does perform at the same level as TD-Gammon.

Curriculum Learning [7] seemed like a good compromise. The model is trained against an 'easy' opponent at the start so that they are not making completely uninformed, random moves for as long. Once it is deemed ready, the model has its opponent changed to a more skilled opponent. This is modelling the education system, where students are taught the basics first, and then throughout the years have their foundations built upon gradually so that a greater depth of understanding can be achieved. In [7], this is done using gradient descent until a 'switch epoch' is reached, when the input data, or, in our case, opponent, is upgraded. However, eventually a skill ceiling is reached again when the toughest opponent is met.

My reasoning is this: curriculum learning shall be used at the start of the learning process. Once defeating the pre-made algorithms, I will switch from curriculum learning to self-play. By this point, the model will have clear strategies and will have played a variety of opponents so should not stick with methods that exploit an individual opponent's weakness. This should hopefully improve learning efficiency at the start and maintain the high skill ceiling from TD-Gammon's learning type.

Bearing off is a part of the game unlike any other. It suddenly becomes less of a game of strategy and instead a race to the finish line. The opponent's checker placement also matters much less as there is nothing you can do to hinder their progress. Because of this, it is possible that Monte Carlo Search Trees (MCSTs) could be used to identify the best move in each situation. Of course, there are still numerous possibilities due to the uncertainty of dice rolls, however there are suitably less than before, and with long simulations using the department supercomputer I may be able to make an exhaustive search tree. MCSTs had extremely good success with the AlphaGo program [8] when combined with Deep Learning.

If it appears like this task is still unfeasible with the processing power at my disposal, then I plan on using one or two networks dedicated to bearing off, similar to the one net used solely for determining when to use the doubling cube.

References

- [1] Pollack, J. B., Blair, A. D., & Land, M. (1997). "Coevolution of backgammon strategies using evolutionary programming". Available at <https://www.cse.unr.edu/~sushil/class/gas/papers/pollack96coevolution.pdf> (Accessed 27 October 2024)
- [2] Strato, Michael (2024). Backgammon Glossary. Available at: <https://usbgf.org/backgammon-glossary/> (Accessed 27 October 2024).
- [3] Tesauro, G. (1995). "Temporal Difference Learning and TD-Gammon", *Communications of the ACM*, Vol. 38, No. 3. (Accessed 30 October 2024).
- [4] Keeler, E.B. & Spencer, J. (1975). "Optimal Doubling in Backgammon", *Operations Research*, vol. 23, no. 6, pp 1063-1071. (Accessed 30 October 2024).
- [5] Tesauro, G. (1989). "Neurogammon Wins Computer Olympiad", *Neural Computation*, vol 38, no. 3, pp 55-68. (Accessed 30 October 2024).
- [6] Tesauro, G. (1990). Neurogammon: A Neural-Network Backgammon Program. Available at: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=6905683475edbd3e3e083e4d281bc29e33d4d737> (Accessed 30 October 2024).
- [7] Bengio, Y., Louradour, J. & Weston, J. (2009). "Curriculum Learning", in *Proceedings of the 26th Annual International Conference on Machine Learning*". (Accessed 30 October 2024).

[8] Silver, D., Huang, A., et al. (2016). “Mastering the Game of Go with Deep Neural Networks and Tree Search”, *Nature*, Vol. 529, No. 7587, pp 484-489.

Generative AI Critical Review

Prompt used: "Write a literature review on the topic: Using reinforcement learning to discover new backgammon strategies"

Unmodified output

Literature Review: Using Reinforcement Learning to Discover New Backgammon Strategies

1. **Introduction to Reinforcement Learning and Backgammon**

Reinforcement learning (RL) has been a cornerstone in the development of autonomous agents capable of learning and decision-making through trial and error in complex environments. Backgammon, an ancient two-player board game with elements of luck and strategy, provides an ideal testbed for reinforcement learning algorithms due to its stochastic nature, extensive state space, and long-term strategic planning requirements.

RL's goal in backgammon is to enable the agent to improve its play over time by learning from interactions with the environment, receiving rewards for favorable outcomes, and adjusting strategies accordingly. The most notable success story of RL in backgammon is the development of TD-Gammon by Gerald Tesauro in the early 1990s, which demonstrated the feasibility of training a neural network-based agent using temporal-difference (TD) learning to compete at expert levels.

This literature review explores key developments in reinforcement learning as applied to backgammon, focusing on the algorithms, innovations, and insights derived from their implementation and subsequent impact on the field of game theory and artificial intelligence.

2. **Early Success: TD-Gammon and Temporal-Difference Learning**

The seminal work by Tesauro (1995) introduced TD-Gammon, an RL agent that used TD(λ) learning, a form of temporal-difference learning, to train a neural network to play backgammon. TD-Gammon's architecture consisted of a multilayer perceptron (MLP) trained to predict the value function, which estimates the probability of winning from a given game state.

The neural network was trained by repeatedly playing against itself and updating its weights based on the difference between predicted outcomes and actual rewards received (win or loss) at the end of the game. This self-play paradigm allowed the agent to explore various strategies autonomously without requiring an explicit game-tree search.

The success of TD-Gammon was groundbreaking, as it quickly surpassed amateur-level play and, after further training and refinement, reached performance comparable to human experts, even challenging world champions. Notably, TD-Gammon's discoveries included counterintuitive strategies that had not been widely adopted in human play. Tesauro's work demonstrated that RL could be used not only to mimic human strategies but also to develop novel strategies that push the boundaries of game play (Tesauro, 1995).

3. **Evolutionary Algorithms and Policy Gradient Methods**

Following TD-Gammon, researchers sought to expand on Tesauro's success by employing different RL techniques to discover new backgammon strategies. Evolutionary algorithms, such as genetic algorithms (GA), have been explored as an alternative to TD learning. GAs evolve a population of agents over generations, selecting the best-performing individuals for reproduction, and applying mutations to explore new strategies (Pollack, Blair, & Land, 1997).

Policy gradient methods also emerged as an alternative to value-based methods like TD learning. These approaches optimize a parameterized policy directly by computing the gradient of the expected reward with respect to the policy parameters. Schulman et al. (2017) introduced Proximal Policy Optimization (PPO), a method that has been shown to be effective in complex environments. While PPO has not been extensively applied to backgammon, its success in other domains suggests it holds promise for future exploration.

Both evolutionary algorithms and policy gradient methods offer advantages in exploring strategy spaces, particularly in environments with high variance or sparse rewards, such as backgammon. These methods have the potential to discover more diverse and robust strategies by exploring regions of the strategy space that value-based methods may overlook.

4. **Monte Carlo Tree Search and Deep Reinforcement Learning**

With the advent of deep learning, there has been a resurgence in interest in combining RL with deep neural networks to tackle complex games. One of the most successful applications of deep RL is the combination of Monte Carlo Tree Search (MCTS) with deep learning, as seen in the success of AlphaGo (Silver et al., 2016).

In the context of backgammon, MCTS could provide a powerful way to search the game tree by simulating many possible rollouts and leveraging the outcomes to make informed decisions. By combining MCTS with deep RL, researchers could potentially overcome the limitations of earlier RL models that relied on shallow neural networks.

While no notable deep RL + MCTS system has been specifically applied to backgammon yet, the success of similar approaches in other board games like Go suggests a promising direction for future research.

5. **Transfer Learning and Curriculum Learning in Backgammon**

Recent developments in reinforcement learning have also explored the potential for transfer learning and curriculum learning to enhance strategy discovery. Transfer learning allows an agent trained in one domain to apply its learned strategies to a related domain, reducing the need for extensive training from scratch (Taylor & Stone, 2009).

Curriculum learning, where an agent is trained on progressively harder tasks, could be particularly useful in backgammon. Early training could focus on simpler endgame scenarios, gradually progressing to full games as the agent's skill improves. This approach could accelerate learning and allow for the discovery of more advanced strategies that may not emerge through self-play alone (Bengio et al., 2009).

In backgammon, curriculum learning could help address the challenge of sparse rewards in early game states by starting with more tractable subproblems, such as learning optimal doubling cube strategies in simplified positions before scaling up to full games.

6. **Challenges and Future Directions**

Despite the successes of RL in backgammon, several challenges remain. One limitation of RL in backgammon is its reliance on extensive computational resources for training. Self-play-based approaches like TD-Gammon require millions of simulated games to converge to effective strategies, which may not be feasible for more complex environments.

Another challenge lies in balancing exploration and exploitation, as RL agents may converge prematurely to suboptimal strategies if they fail to explore the full state space adequately. Techniques such as curiosity-driven exploration (Burda et al., 2018) and intrinsic motivation (Schmidhuber, 2010) offer potential solutions to this problem by encouraging agents to seek out novel states and actions.

Looking forward, hybrid approaches that combine the strengths of different RL techniques (e.g., policy gradient methods with tree search) and the integration of domain-specific knowledge could provide promising avenues for future research. Furthermore, extending RL systems to handle more complex variants of backgammon, such as Hypergammon, which has fewer checkers and more dynamic positions, could lead to the discovery of even more sophisticated strategies.

7. **Conclusion**

Reinforcement learning has significantly advanced our understanding of backgammon strategy, starting with the groundbreaking work on TD-Gammon. Subsequent research has explored various RL methods, from evolutionary algorithms to deep RL, each contributing new insights and strategies to the game. While challenges remain, ongoing advances in RL techniques, combined with increased computational power, promise to drive further discoveries in backgammon and other strategic games.

The potential for hybrid approaches, deep learning, and exploration-based techniques to enhance strategy discovery suggests that the field is ripe for continued innovation. Ultimately, the exploration of new backgammon strategies through RL has implications beyond games, offering valuable insights into decision-making processes in complex, stochastic environments.

References

- Tesauro, G. (1995). Temporal Difference Learning and TD-Gammon. *Communications of the ACM*, 38(3), 58–68.
- Pollack, J. B., Blair, A. D., & Land, M. (1997). Coevolution of backgammon strategies using evolutionary programming. *IEEE Transactions on Evolutionary Computation*, 1(3), 100-112.
- Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms. arXiv preprint arXiv:1707.06347.
- Silver, D., Huang, A., Maddison, C. J., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587), 484-489.
- Taylor, M. E., & Stone, P. (2009). Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10, 1633-1685.
- Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. *Proceedings of the 26th Annual International Conference on Machine Learning* (ICML), 41-48.
- Burda, Y., Edwards, H., Storkey, A., & Klimov, O. (2018). Exploration by Random Network Distillation. arXiv preprint arXiv:1810.12894.
- Schmidhuber, J. (2010). Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3), 230-247.

Critique

Firstly, I found the contents of the review to be very useful in aiding my search for new backgammon material and information, giving me ideas about curriculum learning and genetic algorithms, for example.

However, it contained misinformation. For example, Pollock, Balir and Land's paper was primarily concerning a 1+1 Hill Climbing model in backgammon, which had elements of co-evolutionary learning, rather than a full-blown genetic algorithm. While the paper is still relevant to my research and topic, the AI's output was misleading.

The mention of PPO and a correct reference led me to delve deeper into the possibility of including a PPO implementation in my project as an advanced deliverable. At this point in time it appears out of scope, but I may reconsider after the reinforcement learning module starts in the Epiphany Term.

Monte Carlo Search Trees in AlphaGo led me to read the referenced paper, where the author mentions that they had been used to achieve "Super-human play in Backgammon" and referencing [1]. However, upon reading this paper, I discovered it was unfeasible to recreate, as they had used 2 parallel-RISC Supercomputers.

References

[1] Tesauro, G. & Galperin, G.R. (1996). "On-line policy improvement using Monte-Carlos search", *Advances in Neural Information Processing*, 1068-1074