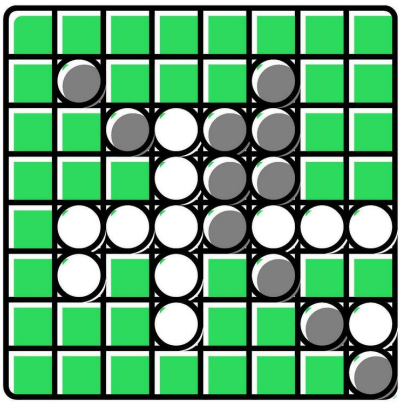
Comparative Analysis of AI Agents for Othello

Mir Mahathir Mohammad
Department of Computer Science
The University of Utah
u1471478@utah.edu

Arman Ashkari
Department of Computer Science
The University of Utah
u1472216@utah.edu



Abstract

Problem: Developing effective AI agents for the game of Othello requires evaluating different algorithmic approaches to determine their relative strengths. **Interest:** This problem is important for understanding how various AI techniques perform in a perfect information, deterministic game with medium complexity. **Solution:** We implement and compare 12 different AI agents ranging from simple heuristics to advanced reinforcement learning. **Evidence:** Through a round-robin tournament with statistical analysis, we demonstrate the relative performance of each approach and identify the most effective strategies for Othello.

1 Introduction

Othello (also known as Reversi) is a classic board game that has served as an important testbed for AI development. As a deterministic, perfect information game with simple rules but complex strategy, it provides an ideal environment for comparing different AI approaches. While simpler than chess or Go, Othello's branching factor (typically 5-15 possible moves) makes it challenging enough to require sophisticated algorithms for strong play.

This project investigates the relative effectiveness of different AI techniques for Othello by implementing and evaluating 12 agents. Our contributions include:

- Implementation of traditional game-playing algorithms (Minimax, MCTS)

Class Project for Advanced AI, Spring 2025.

- Adaptation of reinforcement learning methods (Q-Learning, DQN, Policy Gradient)
- Development of hybrid approaches combining multiple techniques
- Comprehensive evaluation through round-robin tournament

The results provide insights into which algorithms work best for Othello and how different approaches compare in terms of both performance and computational requirements.

2 Related Work

Othello has been studied extensively in AI research. Early work by [4] explored heuristic approaches, while [2] investigated improved search techniques. More recently, [7] demonstrated how deep reinforcement learning could master games like Othello without human knowledge.

Several key papers inform our approach:

- [6] presented early success with heuristic evaluation functions
- [3] showed how to combine search and pattern knowledge
- [5] explored evolutionary approaches to Othello
- [8] demonstrated deep learning applications to board games
- [1] examined hybrid approaches combining multiple techniques

Our work differs by providing a direct comparison of more diverse approaches (including several not previously tested on Othello) under identical conditions.

3 Problem Statement

We formalize the problem as follows:

- **State space:** All possible board configurations (64 squares with 3 states each)
- **Action space:** Valid moves for current player (average 5-15 per turn)
- **Reward:** +1 for win, -1 for loss, 0 for draw
- **Objective:** Develop agents that maximize win probability against diverse opponents

The key challenge is developing evaluation functions and search strategies that can effectively navigate this space to select optimal moves within practical computational limits.

4 Methodology

We implemented the following AI agents:

4.1 Traditional Algorithms

- **Random:** Baseline choosing moves uniformly at random
- **Greedy:** Maximizes immediate piece count difference
- **AlphaBeta:** Minimax with alpha-beta pruning (depths 1,3,4)
- **MCTS:** Monte Carlo Tree Search with UCB1 selection

4.2 Reinforcement Learning

- **QLearning:** Table-based Q-learning with ϵ-greedy exploration
- **DQN:** Deep Q-Network with experience replay
- **PolicyGradient:** Direct policy optimization

4.3 Hybrid Approaches

- **MCTS-MAB:** MCTS with multi-armed bandit node selection
- **MDP-VI:** Markov Decision Process with value iteration
- **Hybrid:** Combines two AlphaBeta agents of different depths in different phases of the game.

Each agent was implemented with consistent interfaces to enable fair comparison. The tournament framework manages game play and records results.

5 Experiments

We evaluated our hypotheses through a round-robin tournament:

5.1 Hypotheses

- H1: Advanced search methods (AlphaBeta, MCTS) will outperform simple heuristics
- H2: Reinforcement learning agents will approach the performance of traditional search

5.2 Experimental Design

- **Domain:** Standard 8x8 Othello with tournament rules
- **Matches:** Each agent plays every other agent 10 times (5 as Black, 5 as White)
- **Evaluation:** Win percentage against each opponent type
- **Metrics:** Overall win rate, head-to-head performance

5.3 Key Findings

We plot the win percentage of different agents in Figure 1. Our tournament revealed several important insights about agent performance in Othello:

5.3.1 Dominance of Search-Based Approaches

- **MCTS-MAB (Multi-Armed Bandit)** emerged as the strongest agent, achieving a perfect 100% win rate against all non-MCTS opponents and 70% against MCTS variants. This suggests that enhanced exploration strategies significantly boost MCTS performance. This is also the algorithm that took the most runtime among all others.
- **Standard MCTS** performed exceptionally well (80-100% wins) against all agents except MCTS-MAB and AlphaBeta4, where its win rates dropped to 20-30%. This indicates that while MCTS is powerful, it can be outperformed by either deeper search (AlphaBeta4) or better exploration (MCTS-MAB).
- **AlphaBeta4** demonstrated particularly strong results against non-MCTS agents (90-100% wins), but showed vulnerability to MCTS approaches (30-70% loss rate). The depth-4 search appears sufficient to outperform most heuristics but struggles with probabilistic approaches.

5.3.2 Reinforcement Learning Insights

- **Q-Learning** showed promising results against weaker opponents (70-90% wins) but failed completely against MCTS variants (0%). This suggests tabular Q-learning may need more sophisticated state representations to compete with search methods.
- **Policy Gradient** exhibited inconsistent performance, doing well against MDP-VI (90%) but poorly against MCTS (0%). The variance suggests the approach may need more training or better reward shaping.
- **DQN** performed moderately (40-50% average), showing neither the reliability of search methods nor the potential highs of other RL approaches in this implementation.

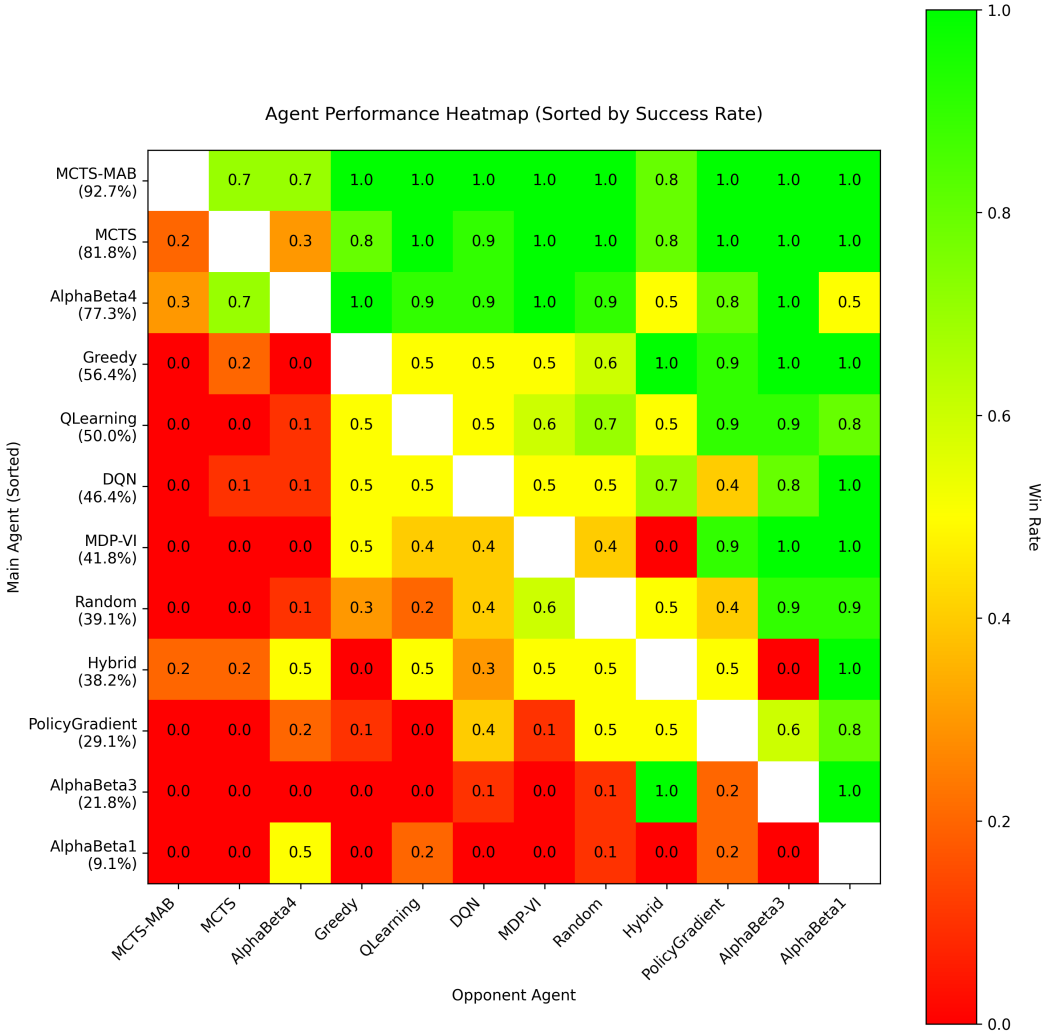


Figure 1: Relative performance of different agent types

5.3.3 Surprising Outcomes

- The **Greedy** approach showed better performance than all of the RL agents. We are assuming that the bad performance of the RL agents is due to ill fine-tuning of architecture, bad hyperparameter adjustment, and low training time.
- The **Hybrid** agent's balanced performance (50% across most matchups) suggests that combining techniques may reduce variance without necessarily achieving peak performance.
- **AlphaBeta1**'s poor showing (0-20% against most non-random opponents) indicates that shallow search depth is insufficient for competent Othello play.
- **MDP-VI**'s strong performance against some RL agents (90-100%) but complete failure against search methods (0%) reveals an interesting dichotomy in approach effectiveness.

6 Limitations and Future Work

The tournament design until now is not fair enough for the RL algorithms because of the following reasons:

- The tree search algorithms consumed more time than the RL agents, having the ability to search under unconstrained time limits. The RL agents perform magnitudes faster than the tree search algorithms such as MCTS and AlphaBeta.
- RL agents needed more training time and a significant number of episodes, which we could not manage due to time constraints.
- The RL agents with the neural networks are simple in architecture with no hyperparameter tuning in our current development. Their poor performance does not fairly represent their genre.

Future directions:

- Incorporate deeper neural networks for better board evaluation with better hyper-parameter tuning.
- Debug RL algorithms to investigate their cause of poor performance and ways to improve.

7 Conclusion

Our primary evaluation of 12 AI agents for Othello demonstrated that traditional search methods (AlphaBeta, MCTS) currently outperform pure reinforcement learning approaches, though RL shows promise. These results provide guidance for developers implementing Othello AIs and suggest directions for future improvement, particularly in combining classical search with learned evaluation functions.

References

[1] Thomas Anthony, Zheng Tian, and David Barber. Thinking fast and slow with deep learning and tree search. *Advances in Neural Information Processing Systems*, 30, 2017.

[2] Michael Buro. Probcut: An effective selective extension of the alpha-beta algorithm. In *ICCA journal*, volume 20, pages 71–76, 1997.

[3] Michael Buro. Improving heuristic mini-max search by supervised learning. *Artificial Intelligence*, 121(1-2):75–96, 2000.

[4] Kai-Fu Lee and Sanjoy Mahajan. The development of a world class othello program. *Artificial Intelligence*, 43(1):21–36, 1986.

[5] David E Moriarty and Risto Miikkulainen. Discovering complex othello strategies through evolutionary neural networks. In *Connection Science*, volume 8, pages 195–210. Taylor & Francis, 1996.

[6] Paul S Rosenbloom. A world-championship-level othello program. *Artificial Intelligence*, 19(3):279–320, 1982.

[7] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[8] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. *AAAI*, 16:2094–2100, 2016.