Can Chess, with Hexagons?

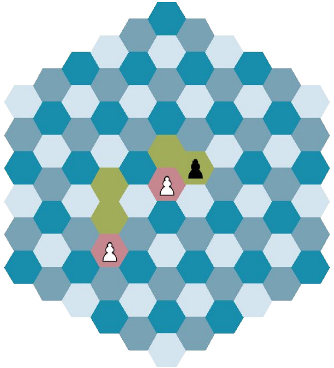
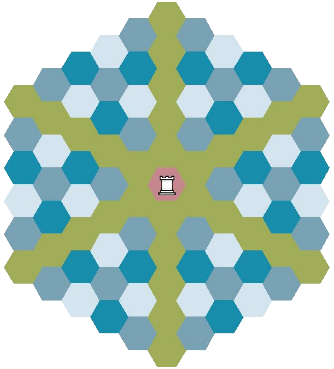
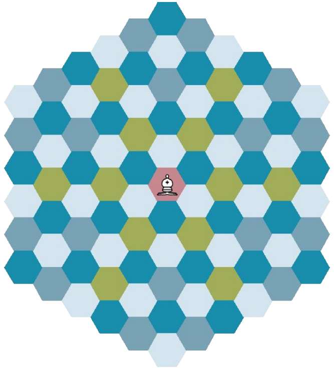
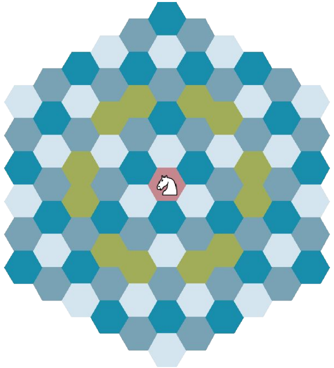
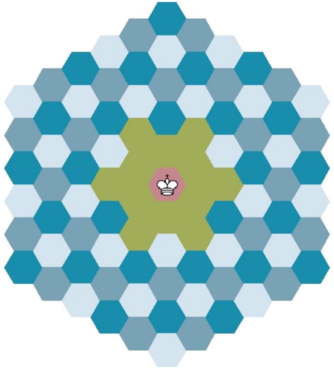
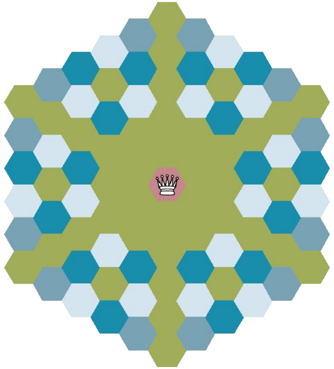
A Reinforcement Learning Exploration.

Flor Sanders
Tawab Safi

Can Chess, with Hexagons?

The Board

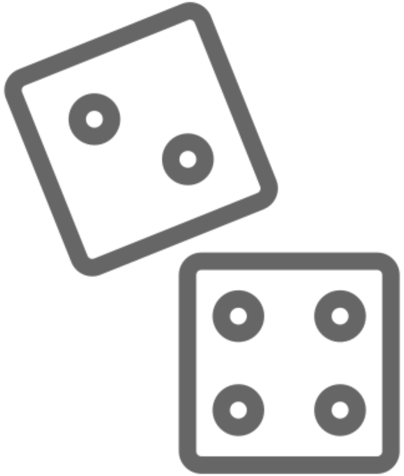
The Rules

The Game

The Engines

Random Player



Greedy Player

Reinforcement Learning

States, Actions & Rewards

State Space → 11 x 11 x 6 array

```
* * * * * * * * * * *
* * * * P * * * * * P
* * * R P * * * * P R
* * N * P * * * P * N
* * * * P * * P * * *
B B B * P * P * B B B
* * * P * * P * * * *
N * P * * * P * N * *
R P * * * * P R * * *
P * * * * * P * * * *
* * * * * * * * * * *
```

Action Space → 91 x 91 moves

- Mask to indicate which are legal

Rewards

- Capture Chess

  Get a reward for each captured piece.
  - Pawn = 1
  - Knight = 3
  - Bishop = 3
  - Rook = 5
  - Queen = 9
  - King = 44 → Sum of all other pieces + 1
- Different goal, but easier to learn

Deep Q Learning

Q Learning Basics

- Bellman Equation

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} P(s,a,s')V(s'))$$

- TD Update

$$TD(a,s) = R(s,a) + \gamma max_{a'} Q(s',a') - Q_{t-1}(s,a)$$

- Policy

$$\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$$

- Approximate Q with Neural Network

Extensions

- Epsilon-Greedy Policy

  Balance exploration and exploitation.
  ϵ decays as 1/n_episodes.

- Prioritized Experience Replay

  Keep a memory buffer of (s, a, r, s’) used for training the Q-network.
  Sample replay probability is proportional to the model’s TD error.

- Fixed Q-Targets

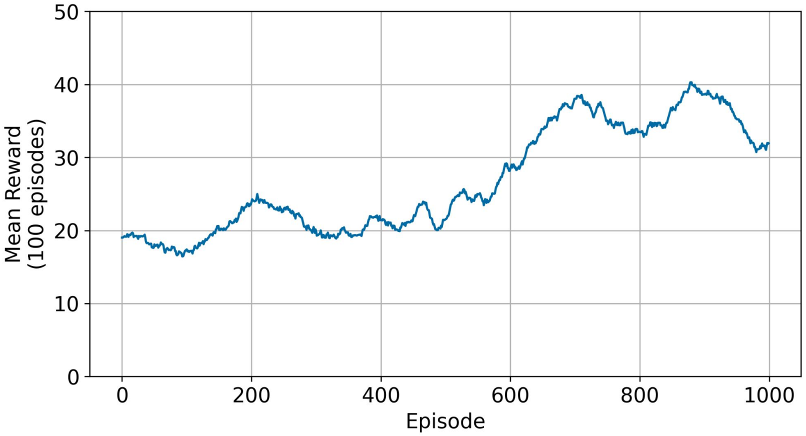
  Use a fixed network to estimate TD error, which is updated every couple of episodes.

Deep Q Learning vs Random Player

Parameters

- ϵ = 0.1
- α = 0.001
- Replay buffer size = 2048
- Minibatch size = 512
- Fixed-Q update period = 10
- NN parameters = 4027
- Nr of episodes = 1000

DQN beats Random Player!

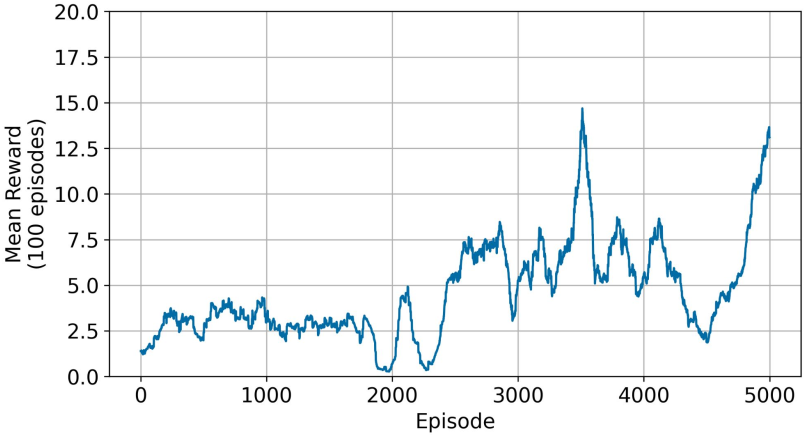


Deep Q Learning vs Greedy Player

Parameters

- ϵ = 0.1
- α = 0.001
- Replay buffer size = 2048
- Minibatch size = 512
- Fixed-Q update period = 10
- NN parameters = 4027
    - Initialize with previous weights
- Nr of episodes = 5000

DQN does not beat Greedy Player…



Simple Actor-Critic

Advantage A2C Basics

- Policy Optimization (Actor Update)

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(a_t|s_t) A^\pi(s_t, a_t)$$

- Critic Update

$$\phi \leftarrow \phi - \beta \nabla_\phi (\delta_t)^2$$

- TD Error = Advantage

$$\delta_t = r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$$

$$A^\pi(s_t, a_t) = \delta_t$$

Extensions

- Shared Feature Extraction Layers

  Both actor & critic share initial convolutional layers which can lead to faster training by learning common features.

- Prioritized Experience Replay

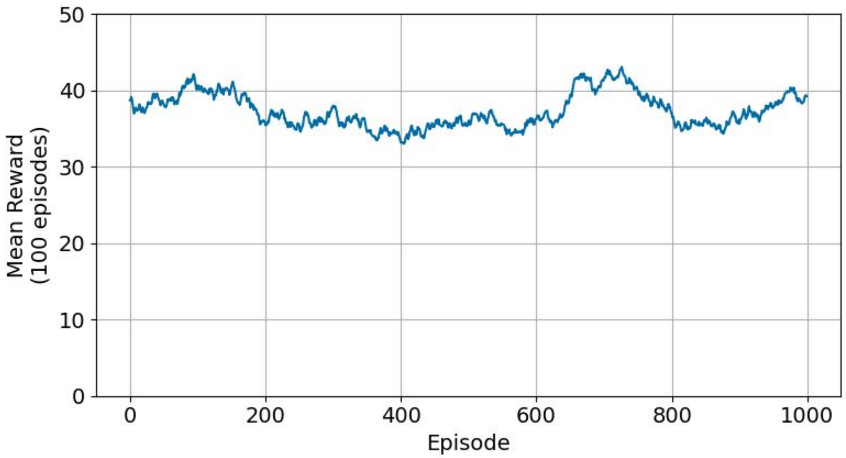
  Keep a memory buffer of (s, a, r, s'),where sample replay probability is proportional to TD error.

Simple Actor-Critic vs Random Player

Parameters

- Shared Feature Extraction = True
- Max Steps per Episode = 150
- Gamma = 0.4
- Learning rate = 0.001
- Replay buffer size = 2,048
- Minibatch size = 512
- NN parameters = 24,406,026
- Nr of episodes = 1,000

A2C vs Random - Training Results:

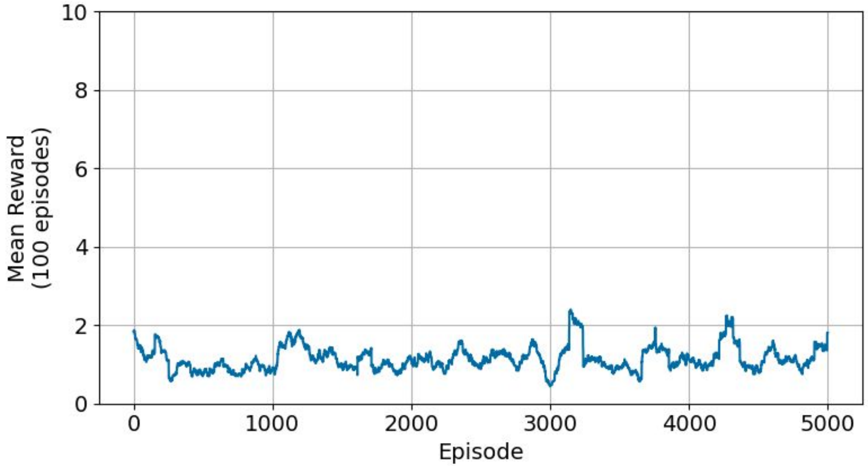


Simple Actor-Critic vs Greedy Player

Parameters

- Shared Feature Extraction = True
- Max Steps per Episode = 150
- Gamma = 0.4
- Learning rate = 0.001
- Replay buffer size = 2,048
- Minibatch size = 512
- NN parameters = 24,406,026
  - Initialize with previous weights
- Nr of episodes = 5,000

A2C vs Greedy - Training Results:



Advanced Actor-Critic

Difference from Simple A2C:

- Proximal Policy Optimization Loss

$$\mathcal{L}_{\theta_k}^{CLIP}(\theta) = \underset{\tau \sim \pi_k}{\mathrm{E}} \left[\sum_{t=0}^{T} \left[\min(r_t(\theta)\hat{A}_t^{\pi_k}, \mathrm{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_t^{\pi_k}) \right] \right]$$

$$r_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_k}(a_t|s_t)$$

- Separate Feature Extraction layer for both actor and critic model.

Features Added

- Residual Blocks

  Add residual blocks to actor and critic networks to potentially help learning complex patterns.

- Double Critic

  Maintain two separate critics to provide a robust estimate of state values.

- Delayed Critic Update

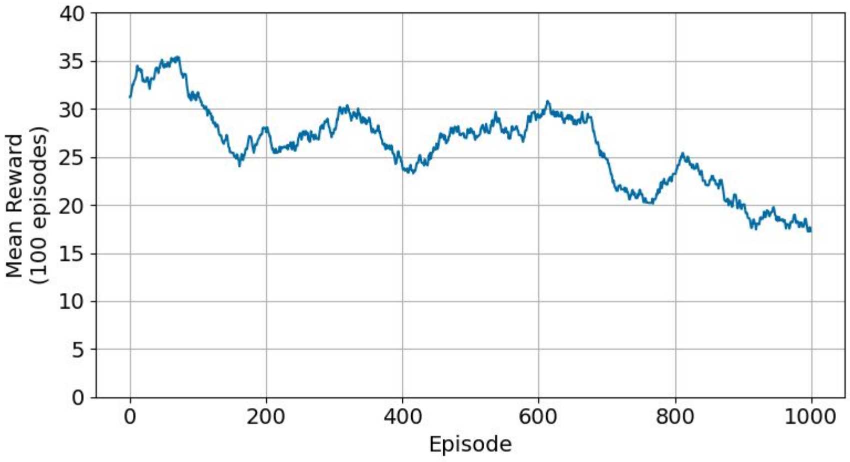
  Use a fixed network to estimate TD error, which is updated every couple of episodes.

Advanced Actor-Critic vs Random Player

Parameters

- Shared Feature Extraction =False
- Max Steps per Episode = 150
- Gamma = 0.4
- Learning rate = 0.001
- Replay buffer size = 2,048
- Minibatch size = 512
- NN parameters = 48,898,445
- Nr of episodes = 1,000

A2C vs Random - Training Results:

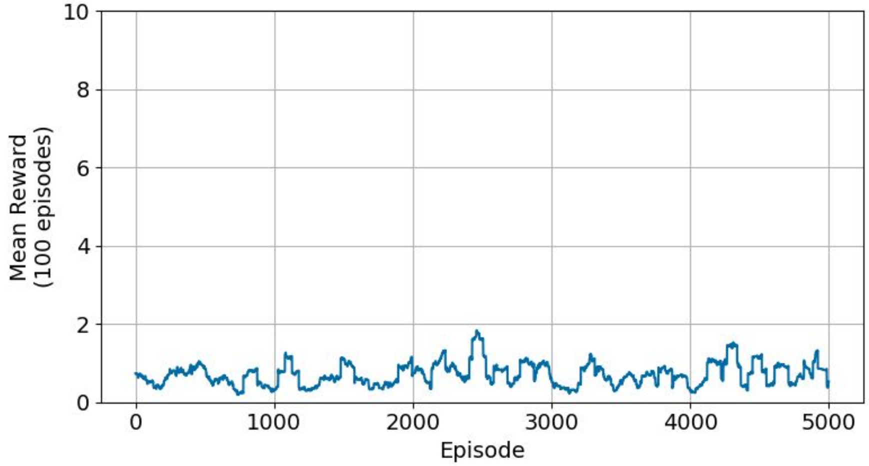


Advanced Actor-Critic vs Greedy Player

Parameters

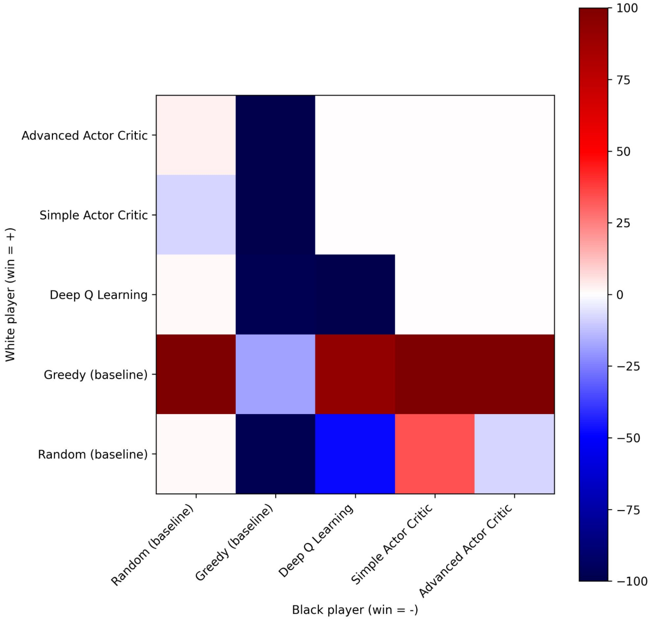
- Shared Feature Extraction =False
- Max Steps per Episode = 150
- Gamma = 0.4
- Learning rate = 0.001
- Replay buffer size = 2,048
- Minibatch size = 512
- NN parameters = 48,898,445
    - Initialize with previous weights
- Nr of episodes = 5,000

A2C vs Random - Training Results:

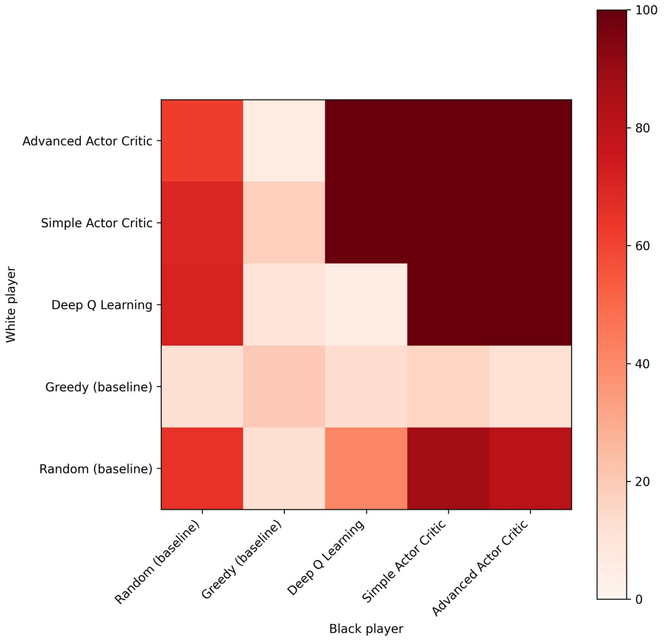


Results

Hex-Chess Engine Competition



Net Wins



Number of Moves

Future Work

Chess Game

- Implement missing features
  - En passant
  - Pawn promotion
  - Check & checkmate
- Make easier to install and play

Reinforcement Learning

- Implement Monte-Carlo Tree Search
- Implement multiple difficulty levels

Conclusion

We built computer engines for Hexagonal Chess

… but they are not very good.

What we learnt:

- Implementing deep Q learning and actor-critic in practice.
- Challenges and limitations of deep learning techniques.

References

- **Our Implementation:** https://github.com/FlorSanders/can_chess_with_hexagons_rl
- Can Chess, with Hexagons (CGP Grey): https://youtu.be/bgR3yESAEVE
- Hexagonal Chess: https://en.wikipedia.org/wiki/Hexagonal_chess
- Hexagonal Game Grids: https://www.redblobgames.com/grids/hexagons/
- RL for Traditional Chess: https://github.com/arjangroen/RLC
- Advantage Actor-Critic: https://arxiv.org/abs/1602.01783
- Soft Actor-Critic: https://arxiv.org/abs/1910.07207 & https://arxiv.org/abs/1801.01290

Student Contributions

- Flor Sanders
  - Implement the Hex-Chess Game Engine & RL Environment
  - Study, implement and evaluate Deep Q Networks
- Tawab Safi
  - Study, implement and evaluate Simple Actor-Critic
  - Study, implement and evaluate Advanced Actor Critic