Mastering the Game of Go Without Human Knowledge

Lead: Liam Hinzman
Facilitators: Tahseen Shabab and Susan Cheng
Overview

- Brief History of AI in Games
- What is Go and Why Should You Care?
- How AlphaGo Zero Works
- Results
- Discussion
Minimax
Heuristics

Reduces search depth
Deep Blue

- 126 million positions per second
- Hand-designed Heuristics
The Game of Go
Go is Incredibly Complex

Go is Hard for Computers

<table>
<thead>
<tr>
<th>GRID SIZE</th>
<th>8 x 8</th>
<th>19 x 19</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVERAGE NUMBER OF MOVE CHOICES PER TURN</td>
<td>35</td>
<td>200—300</td>
</tr>
<tr>
<td>LENGTH OF TYPICAL GAME</td>
<td>60 moves</td>
<td>200 moves</td>
</tr>
<tr>
<td>NUMBER OF POSSIBLE GAME POSITIONS</td>
<td>$10^{44}$</td>
<td>$10^{70}$</td>
</tr>
<tr>
<td>EXPLOSION OF CHOICES</td>
<td>(starting from average game position)</td>
<td></td>
</tr>
<tr>
<td>Move 1</td>
<td></td>
<td>200</td>
</tr>
<tr>
<td>Move 2</td>
<td>1225</td>
<td>40 000</td>
</tr>
<tr>
<td>Move 3</td>
<td>42 675</td>
<td>8 000 000</td>
</tr>
<tr>
<td>Move 4</td>
<td>1 500 625</td>
<td>1 600 000 000</td>
</tr>
</tbody>
</table>
How AlphaGo Zero Works

Monte-Carlo Tree Search

Residual Network

Policy Iteration
Monte-Carlo Tree Search (MCTS)

Selection

Expansion

Tree Policy
Monte-Carlo Tree Search (MCTS)

Sampling

Backpropagation

Default Policy
MCTS: Advantages

- Aheuristic
- Online-search
- Works well on large trees
MCTS: Disadvantages

- Many simulation are required
- No generalization between similar states
- Performance is dependent on “rollout” policy
MCTS in AlphaGo Zero
Upper Confidence Bound for Trees (UCT)

\[ Q(s, a) + c_{puct} \cdot P(s, a) \cdot \frac{\sqrt{\sum_b N(s, b)}}{1 + N(s, a)} \]
Upper Confidence Bound for Trees (UCT)

\[ Q(s, a) + c_{puct} \cdot P(s, a) \cdot \sqrt{\frac{\Sigma_b N(s, b)}{1 + N(s, a)}} \]

- **Exploitation**
- **Exploration**
Upper Confidence Bound for Trees (UCT)

\[ Q(s, a) + c_{puct} \cdot P(s, a) \cdot \frac{\sqrt{\Sigma_b N(s, b)}}{1 + N(s, a)} \]

- \( s \): State
- \( a \): Action
- \( Q(s, a) \): Expected Reward
- \( P(s, a) \): Policy
- \( N(s, a) \): # of state visits
- \( c_{puct} \): Hyperparameter
AlphaGo Zero’s Network Architecture

Policy network |
\[ p_{\alpha/\rho} (a \mid s) \]

Value network |
\[ v_\theta (s') \]

\[ S \]

\[ s' \]
Residual Layer
Dual Heads

The policy head

- Fully connected layer
- Rectifier non-linearity
- Batch normalisation
- 2 convolutional filters (1x1)
- Input

- 19 x 19 + 1 (for pass) move logit probabilities

The value head

- Fully connected layer
- Rectifier non-linearity
- Hidden layer; size 256
- Fully connected layer
- Rectifier non-linearity
- Batch normalisation
- 1 convolutional filter (1x1)
- Input

- tanh non-linearity

- Game value for current player [-1, 1]

- Scalar
Training

Self-play Worker

Training Worker

Evaluator

\[ l = (z - v)^2 - \pi^T \log \rho + c\|\theta\|^2 \]

\[ \pi' > \pi \]
How AlphaGo Zero Chooses a Move

1600 Simulations

\[ \pi \sim N^{1/\tau} \]
Self-Play Workers

The game state  \( \pi \)  The search probabilities  The winner
Training Worker

\[ l = (z - \nu)^2 - \pi^T \log p + c \| \theta \|^2 \]
Evaluator

400 Games  55% Win Rate
MCTS in AlphaGo Zero
5 Minute Break
<table>
<thead>
<tr>
<th>AlphaGo Zero</th>
<th>vs</th>
<th>AlphaGo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entirely self-play</td>
<td></td>
<td>Supervised learning + self-play</td>
</tr>
<tr>
<td>Input is game board</td>
<td></td>
<td>Input is hand-crafted features</td>
</tr>
<tr>
<td>Single network</td>
<td></td>
<td>Two networks</td>
</tr>
<tr>
<td>No rollouts</td>
<td></td>
<td>Rollouts were used</td>
</tr>
</tbody>
</table>
Results
Learning Stages
Ladders
AlphaZero

Chess

AlphaZero vs. Stockfish

- W: 29.8%
- D: 78.6%
- L: 0.4%

AlphaZero vs. Elmo

- W: 98.2%
- D: 0.0%
- L: 1.8%

AlphaZero vs. AG0

- W: 53.7%
- L: 46.3%

Shogi

Go
AlphaGo Zero’s Gift
Discussion
Discussion

How can the AlphaGo Zero algorithm be extended to different games?

How can the sample efficiency of AlphaGo Zero be improved?

A very stable training environment is need for the algorithm. Can this be alleviated to let AlphaZero applied to real-world problems?
Resources

Mastering the Game of Go without Human Knowledge
David Silver 2017 NIPS Talk

ELF OpenGo: An Analysis and Open Reimplementation of AlphaZero

David Silver’s PhD Thesis: Reinforcement Learning and Simulation-Based Search in Computer Go

A Brief History of Game AI Up To AlphaGo - Andrey Kurenkov

AlphaGo Zero Demystified - Dylan Djian