AlphaZero

Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm

Karel Ha

article by Google DeepMind

AI Seminar, 19th December 2017
Outline

The Alpha* Timeline

AlphaGo

AlphaGo Zero (AG0)

AlphaZero

Conclusion
The Alpha* Timeline
Fan Hui

https://en.wikipedia.org/wiki/Fan_Hui
Fan Hui

■ professional 2 dan

https://en.wikipedia.org/wiki/Fan_Hui
Fan Hui

- professional 2 dan
- European Go Champion in 2013, 2014 and 2015

https://en.wikipedia.org/wiki/Fan_Hui
Fan Hui

- professional 2 dan
- European Go Champion in 2013, 2014 and 2015
- European Professional Go Champion in 2016

https://en.wikipedia.org/wiki/Fan_Hui
AlphaGo (AlphaGo Fan) vs. Fan Hui

AlphaGo won 5:0 in a formal match on October 2015.

[AlphaGo] is very strong and stable, it seems like a wall. ... I know AlphaGo is a computer, but if no one told me, maybe I would think the player was a little strange, but a very strong player, a real person.
AlphaGo (AlphaGo Fan) vs. Fan Hui

AlphaGo won 5:0 in a formal match on October 2015.
AlphaGo (AlphaGo Fan) vs. Fan Hui

AlphaGo won 5:0 in a formal match on October 2015.

[AlphaGo] is very strong and stable, it seems like a wall. ... I know AlphaGo is a computer, but if no one told me, maybe I would think the player was a little strange, but a very strong player, a real person.

Fan Hui
Lee Sedol “The Strong Stone”
Lee Sedol “The Strong Stone”

- professional 9 dan

https://en.wikipedia.org/wiki/Lee_Sedol
Lee Sedol “The Strong Stone”

- professional 9 dan
- the 2\textsuperscript{nd} in international titles

https://en.wikipedia.org/wiki/Lee_Sedol
■ professional 9 dan
■ the 2\textsuperscript{nd} in international titles
■ the 5\textsuperscript{th} youngest to become a professional Go player in South Korean history

https://en.wikipedia.org/wiki/Lee_Sedol
Lee Sedol “The Strong Stone”

- professional 9 dan
- the 2\textsuperscript{nd} in international titles
- the 5\textsuperscript{th} youngest to become a professional Go player in South Korean history
- Lee Sedol would win 97 out of 100 games against Fan Hui.

https://en.wikipedia.org/wiki/Lee_Sedol
Lee Sedol “The Strong Stone”

- professional 9 dan
- the 2\textsuperscript{nd} in international titles
- the 5\textsuperscript{th} youngest to become a professional Go player in South Korean history
- Lee Sedol would win 97 out of 100 games against Fan Hui.
- “Roger Federer” of Go

https://en.wikipedia.org/wiki/Lee_Sedol
I heard Google DeepMind’s AI is surprisingly strong and getting stronger, but I am confident that I can win, at least this time.

Lee Sedol
I heard Google DeepMind’s AI is surprisingly strong and getting stronger, but I am confident that I can win, at least this time.

Lee Sedol

...even beating AlphaGo by 4:1 may allow the Google DeepMind team to claim its de facto victory and the defeat of him [Lee Sedol], or even humankind.

interview in JTBC Newsroom
I heard Google DeepMind’s AI is surprisingly strong and getting stronger, but I am confident that I can win, at least this time.

Lee Sedol

...even beating AlphaGo by 4:1 may allow the Google DeepMind team to claim its de facto victory and the defeat of him [Lee Sedol], or even humankind.

interview in JTBC Newsroom
AlphaGo (AlphaGo Lee) vs. Lee Sedol

In March 2016, AlphaGo won 4:1 against the legendary Lee Sedol. AlphaGo won all but the 4th game; all games were won by resignation.
In March 2016 AlphaGo won 4:1 against the legendary Lee Sedol.
In March 2016 AlphaGo won 4:1 against the legendary Lee Sedol. AlphaGo won all but the 4^{th} game; all games were won by resignation.

https://en.wikipedia.org/wiki/AlphaGo_versus_Lee_Sedol
AlphaGo (AlphaGo Lee) vs. Lee Sedol

In March 2016 AlphaGo won 4:1 against the legendary Lee Sedol. AlphaGo won all but the 4\textsuperscript{th} game; all games were won by resignation.

https://en.wikipedia.org/wiki/AlphaGo_versus_Lee_Sedol
In January 2017, DeepMind revealed that AlphaGo had played a series of unofficial online games against some of the strongest professional Go players under the pseudonyms "Master" and "Magister".

This AlphaGo was an improved version of the AlphaGo that played Lee Sedol in 2016.

Over one week, AlphaGo played 60 online fast time-control games. AlphaGo won this series of games 60:0.

https://deepmind.com/research/alphago/match-archive/master/
In January 2017, DeepMind revealed that AlphaGo had played a series of unofficial online games against some of the strongest professional Go players under the pseudonyms “Master” and “Magister”.

https://deepmind.com/research/alphago/match-archive/master/
In January 2017, DeepMind revealed that AlphaGo had played a series of unofficial online games against some of the strongest professional Go players under the pseudonyms “Master” and “Magister”.

This AlphaGo was an improved version of the AlphaGo that played Lee Sedol in 2016.

https://deepmind.com/research/alphago/match-archive/master/
In January 2017, DeepMind revealed that AlphaGo had played a series of unofficial online games against some of the strongest professional Go players under the pseudonyms “Master” and “Magister”.

This AlphaGo was an improved version of the AlphaGo that played Lee Sedol in 2016.

Over one week, AlphaGo played 60 online fast time-control games.
In January 2017, DeepMind revealed that AlphaGo had played a series of unofficial online games against some of the strongest professional Go players under the pseudonyms “Master” and ”Magister”.

This AlphaGo was an improved version of the AlphaGo that played Lee Sedol in 2016.

Over one week, AlphaGo played 60 online fast time-control games. **AlphaGo won this series of games 60:0.**

https://deepmind.com/research/alphago/match-archive/master/
中国乌镇围棋峰会 顶尖棋手 + DeepMind AlphaGo 共创棋妙未来
The Future of Go Summit in Wuzhen  Legendary players and DeepMind's AlphaGo explore the mysteries of Go
中国乌镇 围棋峰会
The Future of Go Summit in Wuzhen

https://events.google.com/alphago2017/
The Future of Go Summit in Wuzhen

1:1 match vs Ke Jie

Team Go

Pair Go

23 May - 27 May 2017 in Wuzhen, China

https://events.google.com/alphago2017/
23 May - 27 May 2017 in Wuzhen, China

Team Go vs. AlphaGo 0:1

https://events.google.com/alphago2017/
23 May - 27 May 2017 in Wuzhen, China

Team Go vs. AlphaGo 0:1

AlphaGo vs. world champion Ke Jie 3:0

https://events.google.com/alphago2017/
defeated AlphaGo Lee by 100 games to 0
defeated AlphaGo Lee by 100 games to 0
AI system that mastered chess, Shogi and Go to “superhuman levels” within a handful of hours

AlphaZero
AI system that mastered chess, Shogi and Go to "superhuman levels" within a handful of hours

**AlphaZero**

defeated AlphaGo Zero (version with 20 blocks trained for 3 days) by 60 games to 40
1 AlphaGo Fan
1. AlphaGo Fan
2. AlphaGo Lee
1. AlphaGo Fan
2. AlphaGo Lee
3. AlphaGo Master
1. AlphaGo Fan
2. AlphaGo Lee
3. AlphaGo Master
4. AlphaGo Zero
1. AlphaGo Fan
2. AlphaGo Lee
3. AlphaGo Master
4. AlphaGo Zero
5. AlphaZero
AlphaGo
Policy and Value Networks

Policy network

$P_{o/p}(a|s)$

Value network

$V_{\theta}(s')$
Training the (Deep Convolutional) Neural Networks

Silver et al. 2016
AlphaGo Zero (AG0)
AlphaGo \{Fan, Lee, Master\} × AlphaGo Zero:

- **AlphaGo Zero**: from scratch by self-play reinforcement learning ("tabula rasa")
- **Additional (auxiliary) input features**: only the black and white stones from the board as input features
- **Separate policy and value networks**: single neural network
- **Tree search using also Monte Carlo rollouts**: simpler tree search using only the single neural network to both evaluate positions and sample moves
- **(AlphaGo Lee) distributed machines + 48 tensor processing units (TPUs)**: single machines + 4 TPUs
- **(AlphaGo Lee) several months of training time**: 72 h of training time (outperforming AlphaGo Lee after 36 h)
AG0: Differences Compared to AlphaGo \{Fan, Lee, Master\}

AlphaGo \{Fan, Lee, Master\} $\times$ AlphaGo Zero:

- supervised learning from human expert positions $\times$ from scratch by self-play reinforcement learning ("tabula rasa")
AlphaGo {Fan, Lee, Master} × AlphaGo Zero:

- supervised learning from human expert positions × from scratch by self-play reinforcement learning (“tabula rasa”)
- additional (auxialiy) input features × only the black and white stones from the board as input features

[Silver et al. 2017b]
AG0: Differences Compared to AlphaGo \{Fan, Lee, Master\}

AlphaGo \{Fan, Lee, Master\} × AlphaGo Zero:

- supervised learning from human expert positions × from scratch by self-play reinforcement learning (“tabula rasa”)
- additional (auxiliary) input features × only the black and white stones from the board as input features
- separate policy and value networks × single neural network

[Silver et al. 2017b]
AlphaGo \{Fan, Lee, Master\} × AlphaGo Zero:

- supervised learning from human expert positions × from scratch by self-play reinforcement learning ("tabula rasa")
- additional (auxiliary) input features × only the black and white stones from the board as input features
- separate policy and value networks × single neural network
- tree search using also Monte Carlo rollouts × simpler tree search using only the single neural network to both evaluate positions and sample moves
AlphaGo \{Fan, Lee, Master\} \times \textbf{AlphaGo Zero:}

- supervised learning from human expert positions \times \textbf{from scratch by self-play reinforcement learning (“tabula rasa”)}
- additional (auxiliary) input features \times \textbf{only the black and white stones from the board as input features}
- separate policy and value networks \times \textbf{single neural network}
- tree search using also Monte Carlo rollouts \times \textbf{simpler tree search using only the single neural network to both evaluate positions and sample moves}
- (AlphaGo Lee) distributed machines + 48 tensor processing units (TPUs) \times \textbf{single machines + 4 TPUs}

[Silver et al. 2017b]
AG0: Differences Compared to AlphaGo \{Fan, Lee, Master\}

AlphaGo \{Fan, Lee, Master\} × AlphaGo Zero:

- supervised learning from human expert positions × from scratch by self-play reinforcement learning ("tabula rasa")
- additional (auxiliary) input features × only the black and white stones from the board as input features
- separate policy and value networks × single neural network
- tree search using also Monte Carlo rollouts × simpler tree search using only the single neural network to both evaluate positions and sample moves
- (AlphaGo Lee) distributed machines + 48 tensor processing units (TPUs) × single machines + 4 TPUs
- (AlphaGo Lee) several months of training time × 72 h of training time (outperforming AlphaGo Lee after 36 h)

[Silver et al. 2017b]
AG0 achieves this via

[Silver et al. 2017b]
AG0 achieves this via

- a new reinforcement learning algorithm
AG0 achieves this via

- a new reinforcement learning algorithm
- with lookahead search inside the training loop

[Silver et al. 2017b]
AG0: Self-Play Reinforcement Learning

[Silver et al. 2017b]
AG0: Self-Play Reinforcement Learning

a) Self-play

\[ s_1 \xrightarrow{a_1 \sim \pi_i} s_2 \xrightarrow{a_2 \sim \pi_j} \ldots \xrightarrow{a_t \sim \pi_t} s_T \]

b) Neural network training

\[ s_1 \xrightarrow{f_\theta} \pi_1 \]
\[ s_2 \xrightarrow{f_\theta} \pi_2 \]
\[ s_3 \xrightarrow{f_\theta} \pi_3 \]

[Silver et al. 2017b]
deep neural network $f_\theta$ with parameters $\theta$:
deep neural network $f_\theta$ with parameters $\theta$:

- **input:** raw board representation $s$
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
  - move probabilities $p$

[Silver et al. 2017b]
deep neural network \( f_\theta \) with parameters \( \theta \):

- input: raw board representation \( s \)
- output:
  - move probabilities \( p \)
  - value \( v \) of the board position
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
  - move probabilities $p$
  - value $v$ of the board position
  - $f_\theta(s) = (p, v)$
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
  - move probabilities $p$
  - value $v$ of the board position
  - $f_\theta(s) = (p, v)$

specifics:
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$

- output:
  - move probabilities $p$
  - value $v$ of the board position
  - $f_\theta(s) = (p, v)$

- specifics:
  - (20 or 40) residual blocks (of convolutional layers)
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
  - move probabilities $p$
  - value $v$ of the board position
  - $f_\theta(s) = (p, v)$

specifics:

- (20 or 40) residual blocks (of convolutional layers)
- batch normalization

[Silver et al. 2017b]
deep neural network $f_\theta$ with parameters $\theta$:

- input: raw board representation $s$
- output:
  - move probabilities $p$
  - value $v$ of the board position
  - $f_\theta(s) = (p, v)$

- specifics:
  - (20 or 40) residual blocks (of convolutional layers)
  - batch normalization
  - rectifier non-linearities
AG0: Comparison of Various Neural Network Architectures

![Bar chart showing Elo ratings for different architectures: dual-res, sep-res, dual-conv, sep-conv. The dual-res architecture has the highest Elo rating, followed by sep-res, dual-conv, and sep-conv.](image)

[Silver et al. 2017b]
AG0: Self-Play Reinforcement Learning – Steps

0. random weights $\theta_0$
0. random weights $\theta_0$
1. at each iteration $i > 0$, self-play games are generated:
0. random weights $\theta_0$
1. at each iteration $i > 0$, self-play games are generated:
   i. MCTS samples search probabilities $\pi_t$ based on the neural network from the previous iteration $f_{\theta_{i-1}}$:

   $$\pi_t = \alpha_{\theta_{i-1}}(s_t)$$

   for each time-step $t = 1, 2, \ldots, T$
AG0: Self-Play Reinforcement Learning – Steps

0. random weights \( \theta_0 \)

1. at each iteration \( i > 0 \), self-play games are generated:
   i. MCTS samples search probabilities \( \pi_t \) based on the neural network from the previous iteration \( f_{\theta_{i-1}} \):

\[
\pi_t = \alpha_{\theta_{i-1}}(s_t)
\]

   for each time-step \( t = 1, 2, \ldots, T \)

   ii. move is sampled from \( \pi_t \)
AG0: Self-Play Reinforcement Learning – Steps

0. random weights $\theta_0$
1. at each iteration $i > 0$, self-play games are generated:
   i. MCTS samples search probabilities $\pi_t$ based on the neural network from the previous iteration $f_{\theta_{i-1}}$:

   $$\pi_t = \alpha_{\theta_{i-1}}(s_t)$$

   for each time-step $t = 1, 2, \ldots, T$
   ii. move is sampled from $\pi_t$
   iii. data $(s_t, \pi_t, z_t)$ for each $t$ are stored for later training

[Silver et al. 2017b]
AG0: Self-Play Reinforcement Learning – Steps

0. random weights $\theta_0$

1. at each iteration $i > 0$, self-play games are generated:
   i. MCTS samples search probabilities $\pi_t$ based on the neural network from the previous iteration $f_{\theta_{i-1}}$:
      \[
      \pi_t = \alpha_{\theta_{i-1}}(s_t)
      \]
      for each time-step $t = 1, 2, \ldots, T$
   ii. move is sampled from $\pi_t$
   iii. data $(s_t, \pi_t, z_t)$ for each $t$ are stored for later training
   iv. new neural network $f_{\theta_i}$ is trained in order to minimize the loss
      \[
      l = (z - v)^2 - \pi^\top \log p + c||\theta||^2
      \]

[Silver et al. 2017b]
0. random weights $\theta_0$
1. at each iteration $i > 0$, self-play games are generated:
   i. MCTS samples search probabilities $\pi_t$ based on the neural network from the previous iteration $f_{\theta_{i-1}}$:

   $$\pi_t = \alpha_{\theta_{i-1}}(s_t)$$

   for each time-step $t = 1, 2, \ldots, T$

   ii. move is sampled from $\pi_t$

   iii. data $(s_t, \pi_t, z_t)$ for each $t$ are stored for later training

   iv. new neural network $f_{\theta_i}$ is trained in order to minimize the loss

   $$l = (z - \nu)^2 - \pi^\top \log p + c||\theta||^2$$
0. random weights $\theta_0$

1. at each iteration $i > 0$, self-play games are generated:
   i. MCTS samples search probabilities $\pi_t$ based on the neural network from the previous iteration $f_{\theta_{i-1}}$:
      \[
      \pi_t = \alpha_{\theta_{i-1}}(s_t)
      \]
      for each time-step $t = 1, 2, \ldots, T$
   ii. move is sampled from $\pi_t$
   iii. data $(s_t, \pi_t, z_t)$ for each $t$ are stored for later training
   iv. new neural network $f_{\theta_i}$ is trained in order to minimize the loss
      \[
      l = (z - v)^2 - \pi^T \log p + c\|\theta\|^2
      \]
      Loss $l$ makes $(p, v) = f_{\theta}(s)$ more closely match the improved search probabilities and self-play winner $(\pi, z)$.

[Silver et al. 2017b]
Monte Carlo Tree Search (MCTS) in AG0:

[Silver et al. 2017b]
Monte Carlo Tree Search (MCTS) in AG0:

1. **Select**
   - \( Q + U \) \( \max \)

2. **Expand and evaluate**
   - \( V \)
   - \( P \)
   - \( (p,v) = f_\theta \)

3. **Backup**
   - \( Q \)
   - \( Q \)
   - \( Q \)

[Silver et al. 2017b]
AG0: Monte Carlo Tree Search (2/2)

d Play

[Silver et al. 2017b]
AG0: Self-Play Reinforcement Learning – Review

[Silver et al. 2017b]
AG0: Elo Rating over Training Time (RL vs. SL)

Silver et al. 2017b
AG0: Elo Rating over Training Time (AG0 with 40 blocks)

![Graph showing Elo rating over training time for AlphaGo Zero 40 blocks, AlphaGo Master, and AlphaGo Lee.](image)

[Silver et al. 2017b]
AG0: Tournament between AI Go Programs

[Silver et al. 2017b]
AG0: Discovered Joseki (Corner Sequences)

a. Five human joseki

b. Five novel joseki variants eventually preferred by AG0

[Silver et al. 2017b]
AG0: Discovered Joseki (Corner Sequences)

a five human joseki

[Silver et al. 2017b]
AG0: Discovered Joseki (Corner Sequences)

a five human joseki

b five novel joseki variants eventually preferred by AG0

[Silver et al. 2017b]
AG0: Discovered Playing Styles

Silver et al. 2017b
AG0: Discovered Playing Styles

at 3 h greedy capture of stones

[Silver et al. 2017b]
AG0: Discovered Playing Styles

at 3h greedy capture of stones

at 19h the fundamentals of Go concepts (life-and-death, influence, territory...)

[Silver et al. 2017b]
at 3 h  greedy capture of stones
at 19 h  the fundamentals of Go concepts (life-and-death, influence, territory...)
at 70 h  remarkably balanced game (multiple battles, complicated *ko* fight, a half-point win for white...)

[Silver et al. 2017b]
AlphaZero
To watch such a strong programme like Stockfish, against whom most top players would be happy to win even one game out of a hundred, being completely taken apart is certainly definitive.

Viswanathan Anand

It's like chess from another dimension.

Demis Hassabis
To watch such a strong programme like Stockfish, against whom most top players would be happy to win even one game out of a hundred, being completely taken apart is certainly definitive.

Viswanathan Anand
To watch such a strong programme like Stockfish, against whom most top players would be happy to win even one game out of a hundred, being completely taken apart is certainly definitive.

Viswanathan Anand

It’s like chess from another dimension.

Demis Hassabis
AlphaZero: Differences Compared to AlphaGo Zero

AlphaGo Zero × AlphaZero:
AlphaGo Zero × AlphaZero:

- binary outcome (win / loss) × expected outcome (including draws or potentially other outcomes)
AlphaGo Zero × AlphaZero:

- binary outcome (win / loss) × expected outcome (including draws or potentially other outcomes)
- board positions transformed before passing to neural networks (by randomly selected rotation or reflection) × no data augmentation

[Silver et al. 2017a]
AlphaGo Zero × AlphaZero:

- binary outcome (win / loss) × expected outcome (including draws or potentially other outcomes)
- board positions transformed before passing to neural networks (by randomly selected rotation or reflection) × no data augmentation
- games generated by the best player from previous iterations (margin of 55 %) × continual update using the latest parameters (without the evaluation and selection steps)
AlphaZero: Differences Compared to AlphaGo Zero

AlphaGo Zero × AlphaZero:

- binary outcome (win / loss) × expected outcome (including draws or potentially other outcomes)
- board positions transformed before passing to neural networks (by randomly selected rotation or reflection) × no data augmentation
- games generated by the best player from previous iterations (margin of 55 %) × continual update using the latest parameters (without the evaluation and selection steps)
- hyper-parameters tuned by Bayesian optimisation × reused the same hyper-parameters without game-specific tuning

[Silver et al. 2017a]
AlphaZero: Elo Rating over Training Time

[Silver et al. 2017a]
AlphaZero: Elo Rating over Training Time

[Silver et al. 2017a]
AlphaZero: Elo Rating over Training Time

[Silver et al. 2017a]
## AlphaZero: Tournament between AI Programs

<table>
<thead>
<tr>
<th>Game</th>
<th>White</th>
<th>Black</th>
<th>Win</th>
<th>Draw</th>
<th>Loss</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chess</td>
<td>AlphaZero</td>
<td>Stockfish</td>
<td>25</td>
<td>25</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Stockfish</td>
<td>AlphaZero</td>
<td>3</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>Shogi</td>
<td>AlphaZero</td>
<td>Elmo</td>
<td>43</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Elmo</td>
<td>AlphaZero</td>
<td>47</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>Go</td>
<td>AlphaZero</td>
<td>AG0 3-day</td>
<td>31</td>
<td>–</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>AG0 3-day</td>
<td>AlphaZero</td>
<td>29</td>
<td>–</td>
<td>21</td>
</tr>
</tbody>
</table>

(Values are given from AlphaZero’s point of view.)
AlphaZero: Openings Discovered by the Self-Play (1/2)

A10: English Opening

1...e5 g3 d5 cxd5 ♕f6 h2 ♗xd5 ♕f3

w 20/30/0, b 8/40/2

D06: Queens Gambit

w 16/34/0, b 1/47/2

2...c6 ♗c3 ♗f6 ♗f3 a6 g3 c4 a4

A46: Queens Pawn Game

w 24/26/0, b 3/47/0

2...d5 c4 e6 ♗c3 ♗c7 ♗f4 O-O e3

E00: Queens Pawn Game

w 17/33/0, b 5/44/1

3...f3 d5 ♗c3 ♗b4 ♗g5 h6 ♗a4 ♗c6

[Silver et al. 2017a]
AlphaZero: Openings Discovered by the Self-Play (2/2)

B40: Sicilian Defence

B10: Caro-Kann Defence

C60: Ruy Lopez (Spanish Opening)

A05: Reti Opening

[Silver et al. 2017a]
Conclusion
Difficulties of Go

- challenging decision-making

[Silver et al. 2017a]
Difficulties of Go

- challenging decision-making
- intractable search space

[Silver et al. 2017a]
Difficulties of Go

- challenging decision-making
- intractable search space
- complex optimal solution

It appears infeasible to directly approximate using a policy or value function!

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining
  - Evaluation by a single neural network

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining
  - Evaluation by a single neural network
  - Monte Carlo tree search

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- effective move selection and position evaluation
  - through deep convolutional neural networks
  - trained by new self-play reinforcement learning algorithm
- new search algorithm combining
  - evaluation by a single neural network
  - Monte Carlo tree search
- more efficient when compared to previous AlphaGo versions

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining
  - Evaluation by a single neural network
  - Monte Carlo tree search
- More efficient when compared to previous AlphaGo versions
  - Single machine

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining
  - Evaluation by a single neural network
  - Monte Carlo tree search
- More efficient when compared to previous AlphaGo versions
  - Single machine
  - 4 TPUs

[Silver et al. 2017a]
AlphaZero: Summary

- Monte Carlo tree search
- Effective move selection and position evaluation
  - Through deep convolutional neural networks
  - Trained by new self-play reinforcement learning algorithm
- New search algorithm combining
  - Evaluation by a single neural network
  - Monte Carlo tree search
- More efficient when compared to previous AlphaGo versions
  - Single machine
  - 4 TPUs
  - Hours rather than months of training time

[Silver et al. 2017a]
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov. It compensated this by:

- selecting those positions more intelligently (the neural network)
- evaluating them more precisely (the same neural network)

Deep Blue relied on a handcrafted evaluation function. AlphaZero was trained tabula rasa from self-play. It used general-purpose learning.

This approach is not specific to the game of Go. The algorithm can be used for much wider class of AI problems!

[Silver et al. 2017a]
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

[Silver et al. 2017a]
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

- selecting those positions more intelligently (the neural network)
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

- selecting those positions more intelligently (the neural network)
- evaluating them more precisely (the same neural network)

[Silver et al. 2017a]
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

- selecting those positions more intelligently (the neural network)
- evaluating them more precisely (the same neural network)

[Silver et al. 2017a]
During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

- selecting those positions more intelligently (the neural network)
- evaluating them more precisely (the same neural network)

Deep Blue relied on a handcrafted evaluation function.
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

■ selecting those positions more intelligently (the neural network)
■ evaluating them more precisely (the same neural network)

Deep Blue relied on a handcrafted evaluation function.

AlphaZero was trained tabula rasa from self-play. It used general-purpose learning.

[Silver et al. 2017a]
Novel approach

During the matches (against Stockfish and Elmo), AlphaZero evaluated thousands of times fewer positions than Deep Blue against Kasparov.

It compensated this by:

- selecting those positions more intelligently (the neural network)
- evaluating them more precisely (the same neural network)

Deep Blue relied on a handcrafted evaluation function.

AlphaZero was trained tabula rasa from self-play. It used general-purpose learning.

This approach is not specific to the game of Go. The algorithm can be used for much wider class of AI problems!

[Silver et al. 2017a]
Thank you!

Questions?
Backup Slides
# Input Features of AlphaZero’s Neural Networks

<table>
<thead>
<tr>
<th>Feature</th>
<th>Go</th>
<th>Chess</th>
<th>Shogi</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Planes</td>
<td>Feature</td>
<td>Planes</td>
</tr>
<tr>
<td>P1 stone</td>
<td>1</td>
<td>P1 piece</td>
<td>6</td>
</tr>
<tr>
<td>P2 stone</td>
<td>1</td>
<td>P2 piece</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Repetitions</td>
<td>2</td>
</tr>
<tr>
<td>P1 stone</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>P2 stone</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Colour</td>
<td>1</td>
<td>Colour</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total move count</td>
<td>1</td>
</tr>
<tr>
<td>P1 castling</td>
<td>1</td>
<td>P1 castling</td>
<td>2</td>
</tr>
<tr>
<td>P2 castling</td>
<td>2</td>
<td>P2 castling</td>
<td>2</td>
</tr>
<tr>
<td>No-progress count</td>
<td>1</td>
<td>No-progress count</td>
<td>1</td>
</tr>
<tr>
<td>Total</td>
<td>17</td>
<td>Total</td>
<td>119</td>
</tr>
</tbody>
</table>

[Silver et al. 2017a]
### AlphaZero: Statistics of Training

<table>
<thead>
<tr>
<th></th>
<th>Chess</th>
<th>Shogi</th>
<th>Go</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mini-batches</td>
<td>700k</td>
<td>700k</td>
<td>700k</td>
</tr>
<tr>
<td>Training Time</td>
<td>9h</td>
<td>12h</td>
<td>34h</td>
</tr>
<tr>
<td>Training Games</td>
<td>44 million</td>
<td>24 million</td>
<td>21 million</td>
</tr>
<tr>
<td>Thinking Time</td>
<td>800 sims</td>
<td>800 sims</td>
<td>800 sims</td>
</tr>
<tr>
<td></td>
<td>40 ms</td>
<td>80 ms</td>
<td>200 ms</td>
</tr>
</tbody>
</table>

[Silver et al. 2017a]
### AlphaZero: Evaluation Speeds

<table>
<thead>
<tr>
<th>Program</th>
<th>Chess</th>
<th>Shogi</th>
<th>Go</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlphaZero</td>
<td>80k</td>
<td>40k</td>
<td>16k</td>
</tr>
<tr>
<td>Stockfish</td>
<td>70,000k</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elmo</td>
<td></td>
<td>35,000k</td>
<td></td>
</tr>
</tbody>
</table>

[Silver et al. 2017a]
Scalability When Compared to Other Programs

[Silver et al. 2017a]
Further Reading

AlphaGo:

■ **Google Research Blog**

■ an article in **Nature**
  http://www.nature.com/news/google-ai-algorithm-masters-ancient-game-of-go-1.19234

■ a **reddit** article claiming that AlphaGo is even stronger than it appears to be:
  “AlphaGo would rather win by less points, but with higher probability.”
  https://www.reddit.com/r/baduk/comments/49y17z/the_true_strength_of_alphago/

■ a video of how AlphaGo works (put in layman’s terms) https://youtu.be/qWcfiPi9gUU

Articles by Google DeepMind:

■ **Atari player**: a DeepRL system which combines Deep Neural Networks with Reinforcement Learning (Mnih et al. 2015)

■ **Neural Turing Machines** (Graves, Wayne, and Danihelka 2014)

Artificial Intelligence:

■ **Artificial Intelligence course at MIT**
Further Reading II

- **Introduction to Artificial Intelligence at Udacity**
  https://www.udacity.com/course/intro-to-artificial-intelligence--cs271

- **General Game Playing course** https://www.coursera.org/course/ggp

- **Singularity**

- **The Singularity Is Near** (Kurzweil 2005)

Combinatorial Game Theory (founded by John H. Conway to study endgames in Go):

- **Combinatorial Game Theory course**
  https://www.coursera.org/learn/combinatorial-game-theory

- **On Numbers and Games** (Conway 1976)

- **Computer Go as a sum of local games:** an application of combinatorial game theory (Müller 1995)

Chess:

- **Deep Blue beats G. Kasparov in 1997**
  https://youtu.be/NJarxpYyoFI

Machine Learning:

- **Machine Learning course**

- **Reinforcement Learning**
  http://reinforcementlearning.ai-depot.com/

- **Deep Learning** (LeCun, Bengio, and Hinton 2015)
Further Reading III

- **Deep Learning course** [https://www.udacity.com/course/deep-learning--ud730](https://www.udacity.com/course/deep-learning--ud730)
- **Two Minute Papers** [https://www.youtube.com/user/keeroyz](https://www.youtube.com/user/keeroyz)
- **Applications of Deep Learning** [https://youtu.be/hPKJBXkyTKM](https://youtu.be/hPKJBXkyTKM)

Neuroscience:

- [http://www.brainfacts.org/](http://www.brainfacts.org/)


