AlphaZero

Richard Diehl Martinez and David Golub

Background

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

Published in: Nature, October 18 2017

Authors list: David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, Demis Hassabis

Long-standing History of AI Agents in Board Games

IBM’s DeepBlue vs. Kasparov, 1997

Go vs. Chess

Natural question: why did it take so long to get to superhuman in Go?

IBM’s Deep Blue: superhuman chess player in 1997

- why doesn’t same approach work for go?

Deep Blue

- brute-force minimax search
- could look ahead between 12 and 40 plys (half-moves) parameterized value function for the leaves
- estimate: every additional ply yields 50-70 ELO points

Why did it take so long to get to superhuman for Go?

State space is significantly larger

Significantly better machine learning models (Neural Networks)

Significantly better hardware/compute

Algorithmic improvements over brute force search


AlphaGo Zero
Starting from scratch

defeated AlphaGo Lee by 100 games to 0

AlphaZero

defeated AlphaGo Zero (version with 20 blocks trained for 3 days) by 60 games to 40

AI system that mastered chess, Shogi and Go to "superhuman levels" within less than a day

AlphaZero

defeated AlphaGo Zero (version with 20 blocks trained for 3 days) by 60 games to 40

AlphaZero

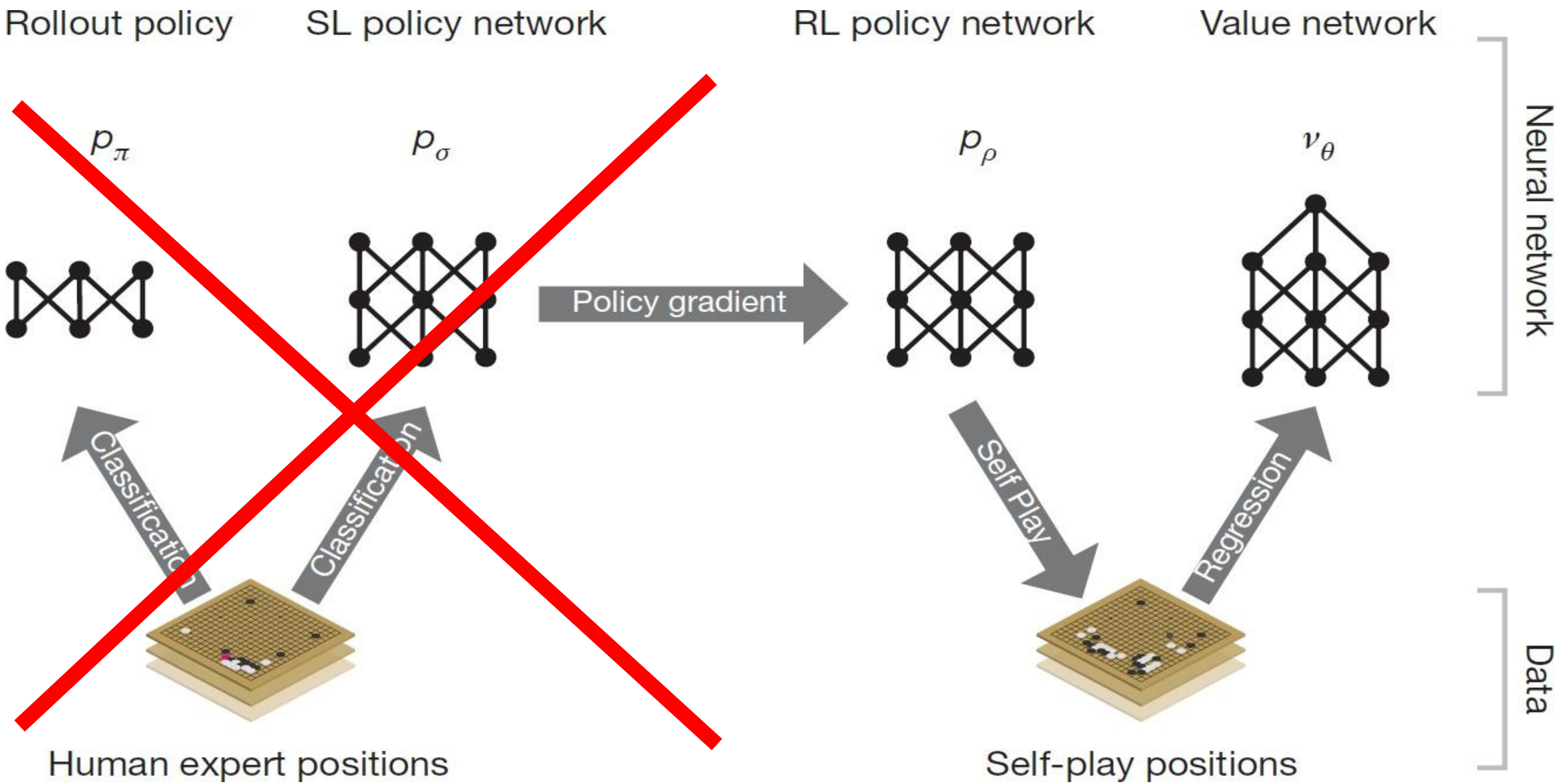
Single Neural Network f_θ that takes in current state s, with two outputs:

$v_\theta(s) \in [-1, 1]$: expected outcome of game (win, lose draw)

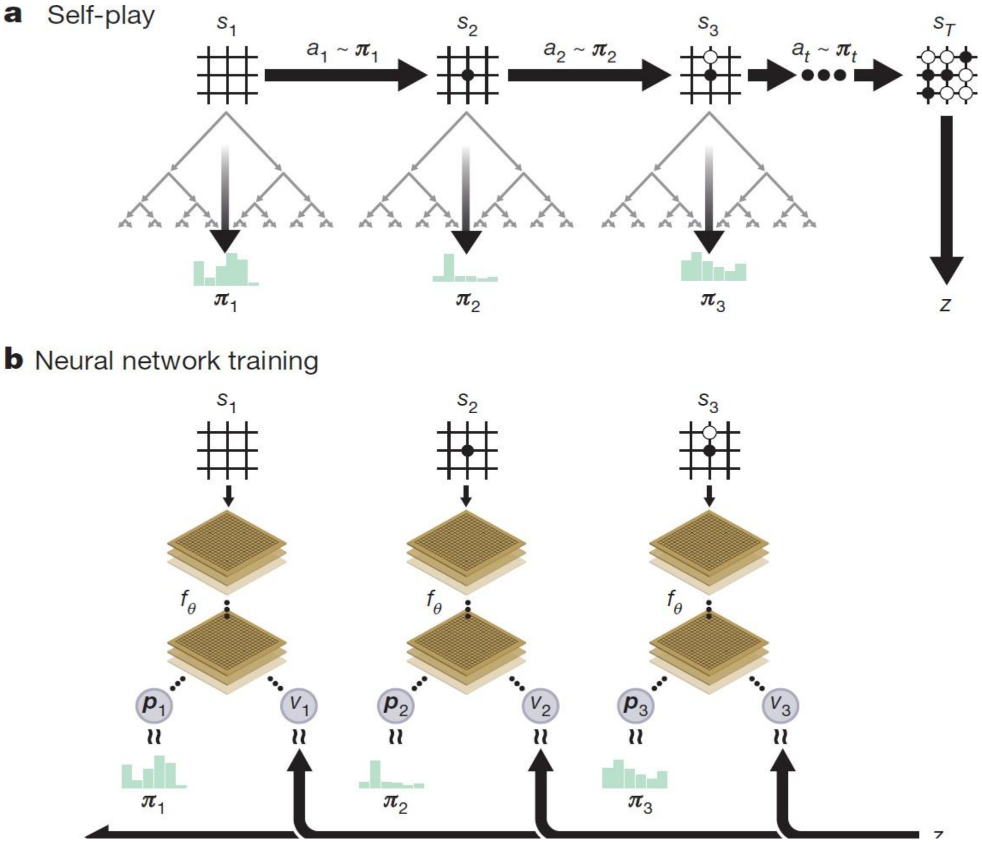
$\vec{p}_\theta(s)$ Policy: probability distribution over actions from state s.

No need for RL! Directly do search to find a better action.

Training the Neural Network



Training the Neural Network



Training Algorithm

High level idea: get training examples in the form $(s_t, \vec{\pi}_t, z_t)$ through self play

s_t is the state, $\vec{\pi}_t$ is a probability distribution over actions, and z_t is the outcome of the game (win/lose).

Optimize: $$l = \sum_t (v_\theta(s_t) - z_t)^2 - \vec{\pi}_t \cdot \log(\vec{p}_\theta(s_t))$$

AlphaZero Code in Python

```
def policyIterSP(game):
    nnet = initNNet()                                       # initialise random neural network
    examples = []
    for i in range(numIters):
        for e in range(numEps):
            examples += executeEpisode(game, nnet)          # collect examples from this game
        new_nnet = trainNNet(examples)
        frac_win = pit(new_nnet, nnet)                      # compare new net with previous net
        if frac_win > threshold:
            nnet = new_nnet                                 # replace with new net
    return nnet
```

Training Implementation

- Sensitive to hyperparameters and initial exploration probability: See https://dselsam.github.io/issues-with-alpha-zero/ for more info
- Synchronous stochastic gradient descent with mini-batches of size 4096 for stability

Parameter-server model:

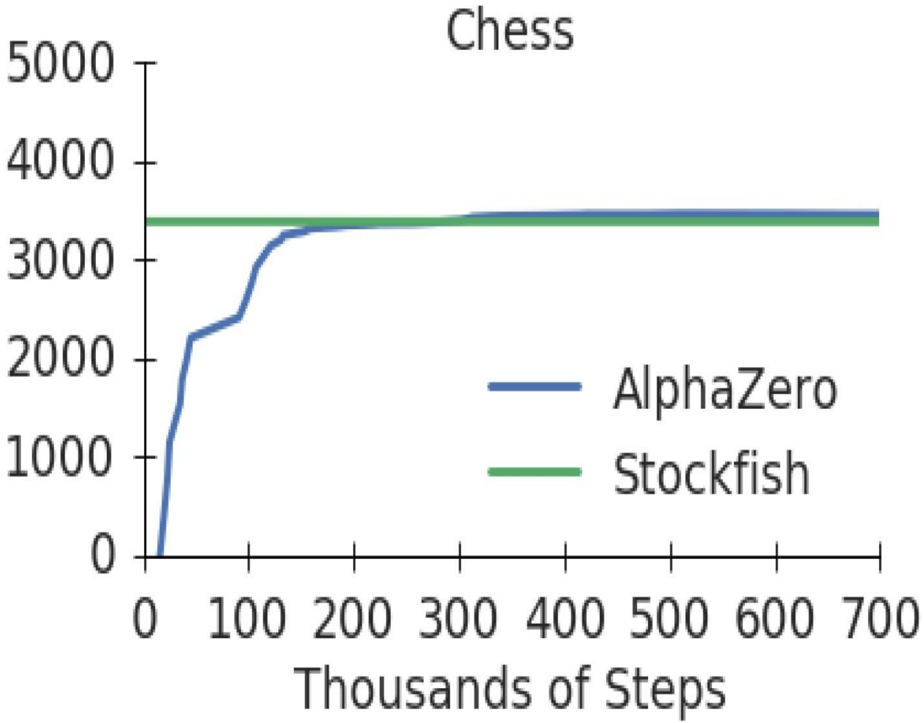
- 5,000 first-generation TPUs to generate self-play games
- 64 first-generation TPUs for parameter updates

Monte-Carlo Tree Search

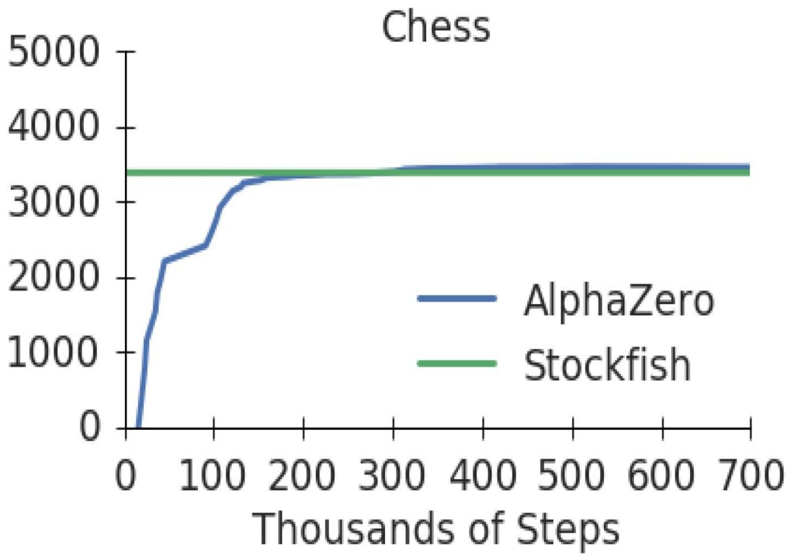
$$U_i = \frac{W_i}{N_i} + cP_i\sqrt{\frac{\ln N_p}{1 + N_i}}$$

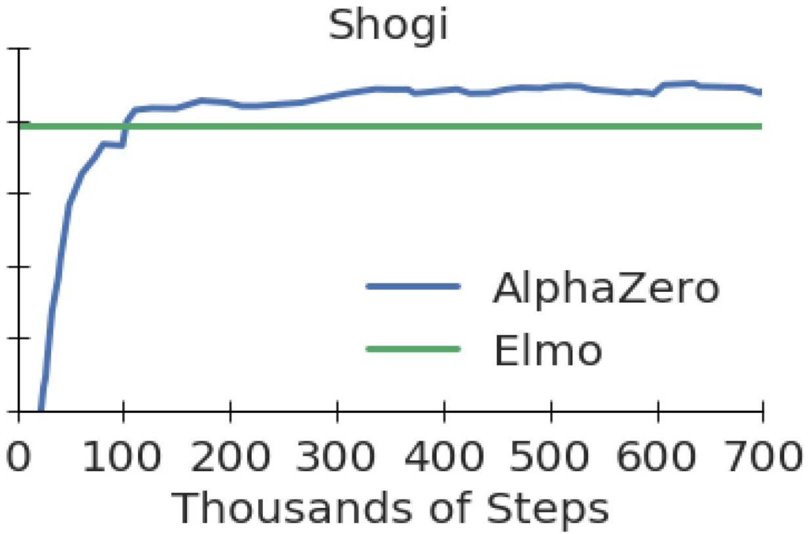
https://web.stanford.edu/~surag/posts/alphazero.html

AlphaZero: Elo Rating Over Training Time

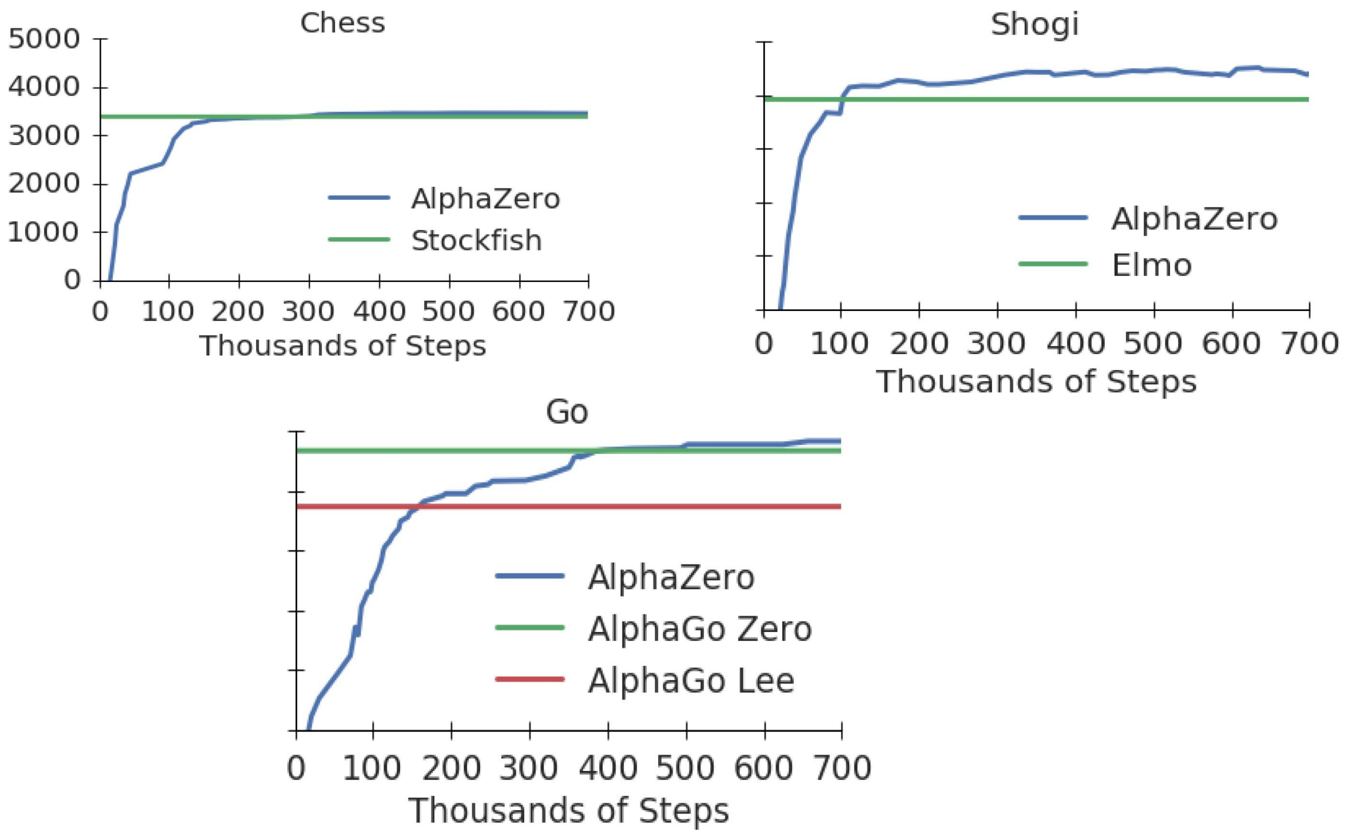


AlphaZero: Elo Rating over Training Time





AlphaZero: Elo Rating Over Training Time

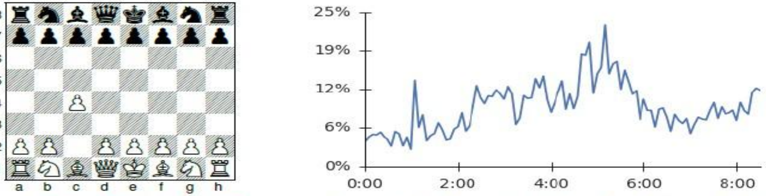


AlphaZero: Tournament between AI Programs

Game	White	Black	Win	Draw	Loss
Chess	*AlphaZero*	*Stockfish*	25	25	0
	Stockfish	*AlphaZero*	3	47	0
Shogi	*AlphaZero*	*Elmo*	43	2	5
	Elmo	*AlphaZero*	47	0	3
Go	*AlphaZero*	*AG0 3-day*	31	–	19
	AG0 3-day	*AlphaZero*	29	–	21

AlphaZero: Openings Discovered by Self-Play (½)

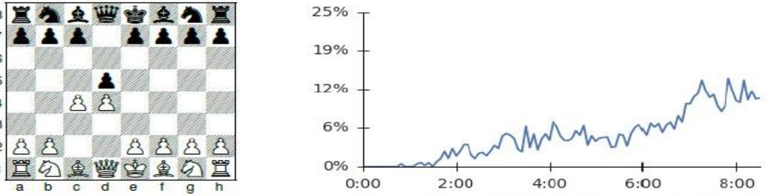
A10: English Opening



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

1...e5 g3 d5 cxd5 ♘f6 ♗g2 ♘xd5 ♘f3

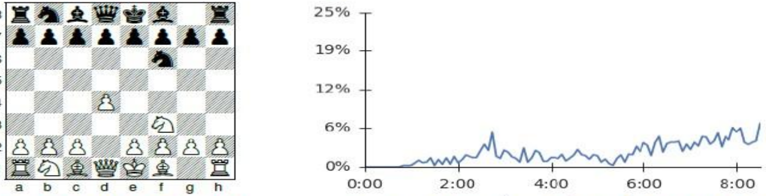
D06: Queens Gambit



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

2...c6 ♘c3 ♘f6 ♘f3 a6 g3 c4 a4

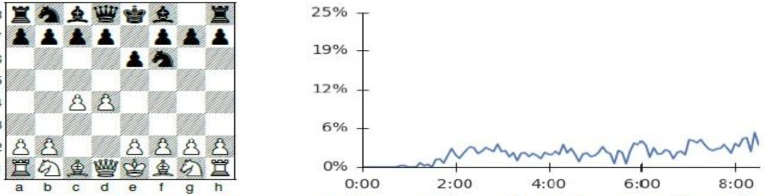
A46: Queens Pawn Game



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

2...d5 c4 e6 ♘c3 ♗e7 ♗f4 O-O e3

E00: Queens Pawn Game

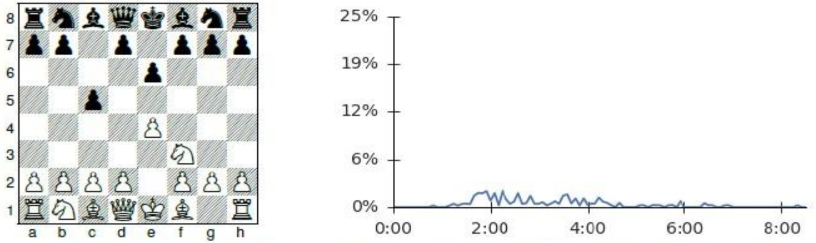


w 17/33/0, b 5/44/1

3.♘f3 d5 ♘c3 ♗b4 ♗g5 h6 ♕a4 ♘c6

AlphaZero: Openings Discovered by Self-Play

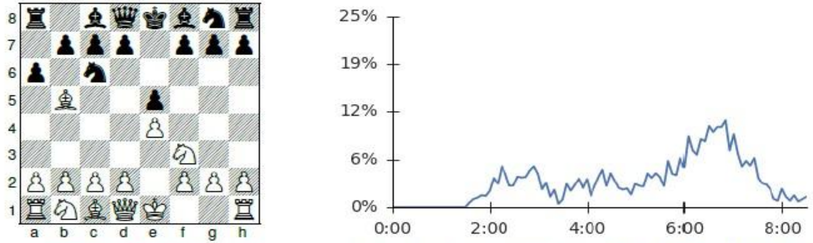
B40: Sicilian Defence



w 17/31/2, b 3/40/7

3.d4 cxd4 ♘xd4 ♘c6 ♘c3 ♕c7 ♗e3 a6

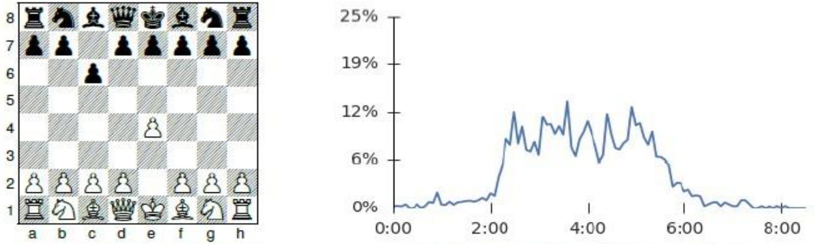
C60: Ruy Lopez (Spanish Opening)



w 27/22/1, b 6/44/0

4.♗a4 ♗e7 O-O ♘f6 ♖e1 b5 ♗b3 O-O

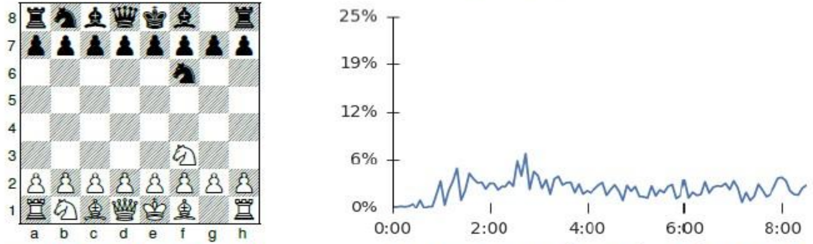
B10: Caro-Kann Defence



w 25/25/0, b 4/45/1

2.d4 d5 e5 ♗f5 ♘f3 e6 ♗e2 a6

A05: Reti Opening



w 13/36/1, b 7/43/0

2.c4 e6 d4 d5 ♘c3 ♗e7 ♗f4 O-O

Conclusion

AlphaZero: new SOTA algorithm for Go, Shogi Chess

Trained solely through self-play + Monte-Carlo Tree Search

Trained using maximum likelihood estimation (MLE) to predict policy and reward, without using reinforcement learning for updates!