How To Solve Go？

Liangqu LONG

Deep Q-Learning

- Represent value function by deep Q-network with weights w

$$Q(s, a, w) \approx Q^{\pi}(s, a)$$

- Define objective function by mean-squared error in Q-values

$$\mathcal{L}(w) = \mathbb{E}\left[\left(\underbrace{r + \gamma \max_{a'} Q(s', a', w)}_{\text{target}} - Q(s, a, w)\right)^2\right]$$

- Leading to the following Q-learning gradient

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \mathbb{E}\left[\left(r + \gamma \max_{a'} Q(s', a', w) - Q(s, a, w)\right) \frac{\partial Q(s, a, w)}{\partial w}\right]$$

- Optimise objective end-to-end by SGD, using $\frac{\partial L(w)}{\partial w}$

DQN

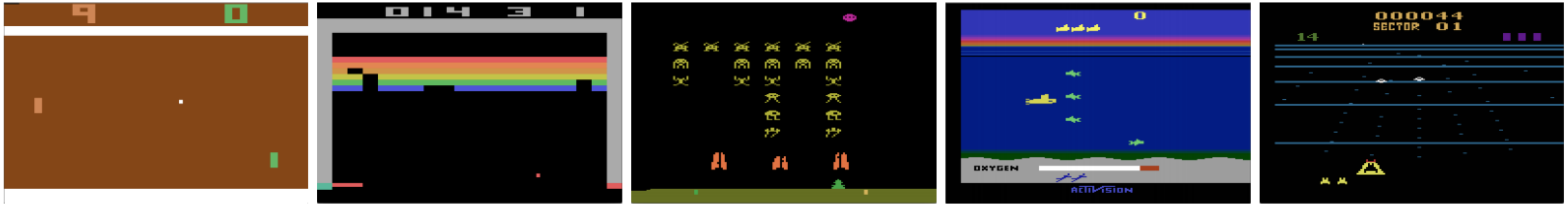


Figure 1: Screen shots from five Atari 2600 Games: (*Left-to-right*) Pong, Breakout, Space Invaders, Seaquest, Beam Rider

Sove Go in DQN
NO....
r(s,a) = Unkown!!!

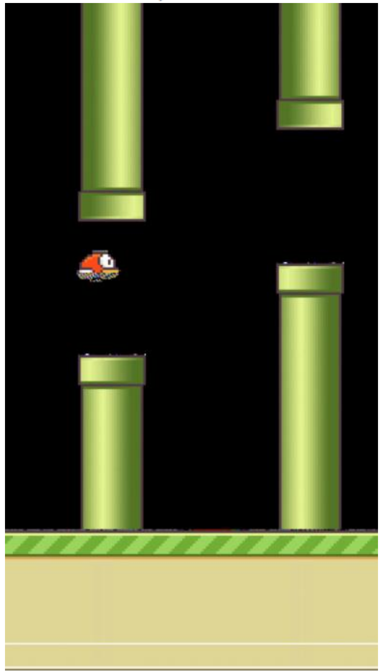
Value Iteration VS Policy Iteration

1. Directly Policy Search, converge faster

   Π <> argmax(Q(s, a))

2. Dimension Explosion

   Go: 19*19 +1 = 362



Policy Gradient Methods: Overview

Problem:

$$\text{maximize}\, E[R \mid \pi_\theta]$$

Intuitions: collect a bunch of trajectories, and ...

1. Make the good trajectories more probable[1]
2. Make the good actions more probable
3. Push the actions towards good actions (DPG[2], SVG[3])

---

[1] R. J. Williams. "Simple statistical gradient-following algorithms for connectionist reinforcement learning". *Machine learning* (1992); R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. "Policy gradient methods for reinforcement learning with function approximation". *NIPS*. MIT Press, 2000.

[2] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, et al. "Deterministic Policy Gradient Algorithms". *ICML*. 2014.

[3] N. Heess, G. Wayne, D. Silver, T. Lillicrap, Y. Tassa, et al. "Learning Continuous Control Policies by Stochastic Value Gradients". *arXiv preprint arXiv:1510.09142* (2015).

Score Function Gradient Estimator

- Consider an expectation $E_{x \sim p(x \mid \theta)}[f(x)]$. Want to compute gradient wrt θ

$$
\begin{aligned}
\nabla_\theta E_x[f(x)] &= \nabla_\theta \int \mathrm{d}x \; p(x \mid \theta) f(x) \\
&= \int \mathrm{d}x \; \nabla_\theta p(x \mid \theta) f(x) \\
&= \int \mathrm{d}x \; p(x \mid \theta) \frac{\nabla_\theta p(x \mid \theta)}{p(x \mid \theta)} f(x) \\
&= \int \mathrm{d}x \; p(x \mid \theta) \nabla_\theta \log p(x \mid \theta) f(x) \\
&= E_x[f(x) \nabla_\theta \log p(x \mid \theta)].
\end{aligned}
$$

- Last expression gives us an unbiased gradient estimator. Just sample $x_i \sim p(x \mid \theta)$, and compute $\hat{g}_i = f(x_i) \nabla_\theta \log p(x_i \mid \theta)$.
- Need to be able to compute and differentiate density $p(x \mid \theta)$ wrt θ

Score Function Gradient Estimator for Policies

- Now random variable x is a whole trajectory $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots, s_{T-1}, a_{T-1}, r_{T-1}, s_T)$

$$\nabla_\theta E_\tau[R(\tau)] = E_\tau[\nabla_\theta \log p(\tau \mid \theta) R(\tau)]$$

- Just need to write out $p(\tau \mid \theta)$:

$$p(\tau \mid \theta) = \mu(s_0) \prod_{t=0}^{T-1} [\pi(a_t \mid s_t, \theta) P(s_{t+1}, r_t \mid s_t, a_t)]$$

$$\log p(\tau \mid \theta) = \log \mu(s_0) + \sum_{t=0}^{T-1} [\log \pi(a_t \mid s_t, \theta) + \log P(s_{t+1}, r_t \mid s_t, a_t)]$$

$$\nabla_\theta \log p(\tau \mid \theta) = \nabla_\theta \sum_{t=0}^{T-1} \log \pi(a_t \mid s_t, \theta)$$

$$\nabla_\theta \mathbb{E}_\tau [R] = \mathbb{E}_\tau \left[R \nabla_\theta \sum_{t=0}^{T-1} \log \pi(a_t \mid s_t, \theta) \right]$$

- Interpretation: using good trajectories (high R) as supervised examples in classification / regression

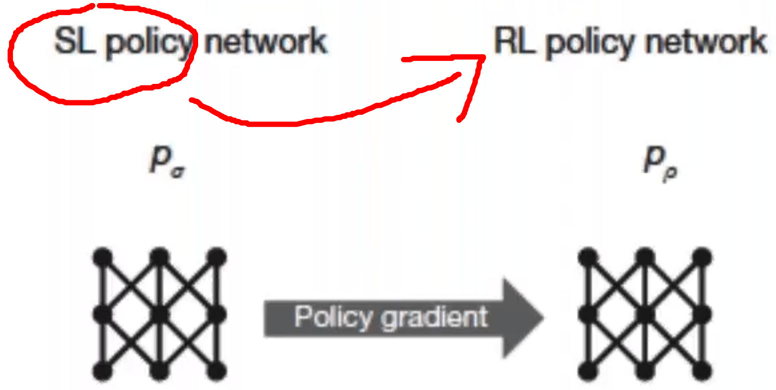
Vanilla Policy Gradient
Not occured yet!

AlphaGo Solution

What is the problem in previous approach?

Learn from both bad and good moves, both winner and loser's moves!

Policy Gradient Method



RL policy network won more than 80% of games against SL policy network.

RL policy network won 85% of games against Pachi (the strongest Go AI before alphago).

Pure RL policy network AI without any tree search!!!
Just like a man with purely Intuition!!!

Policy Gradient: Introduce Baseline

- Further reduce variance by introducing a *baseline* $b(s)$

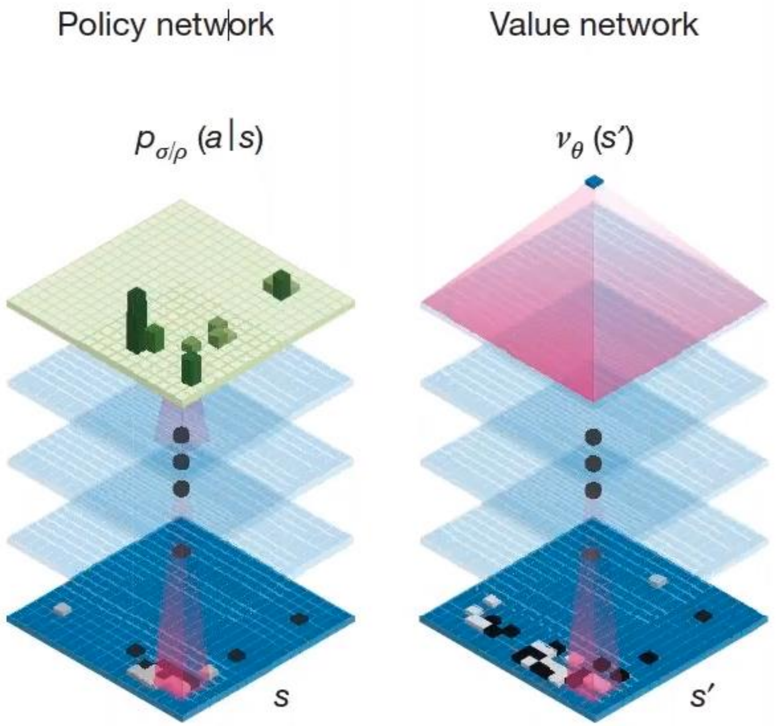
$$\nabla_\theta \mathbb{E}_\tau [R] = \mathbb{E}_\tau \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi(a_t \mid s_t, \theta) \left(\sum_{t'=t}^{T-1} r_{t'} - b(s_t) \right) \right]$$

- For any choice of b, gradient estimator is unbiased.
- Near optimal choice is expected return,
  $b(s_t) \approx \mathbb{E}\left[r_t + r_{t+1} + r_{t+2} + \cdots + r_{T-1}\right]$
- Interpretation: increase logprob of action a_t proportionally to how much returns $\sum_{t'=t}^{T-1} r_{t'}$ are better than expected

Introduce Bias to reduce Variance

AlphaGo Solution

Can we get Evaluation Function? Yes !!! With similar CNN!



$$v^p(s) = \mathbb{E}[z_t | s_t = s,\ a_{t\ldots T} \sim p]$$

Supervised Regression Problem

Input: 19 X 19 matrix with -1, 0, 1

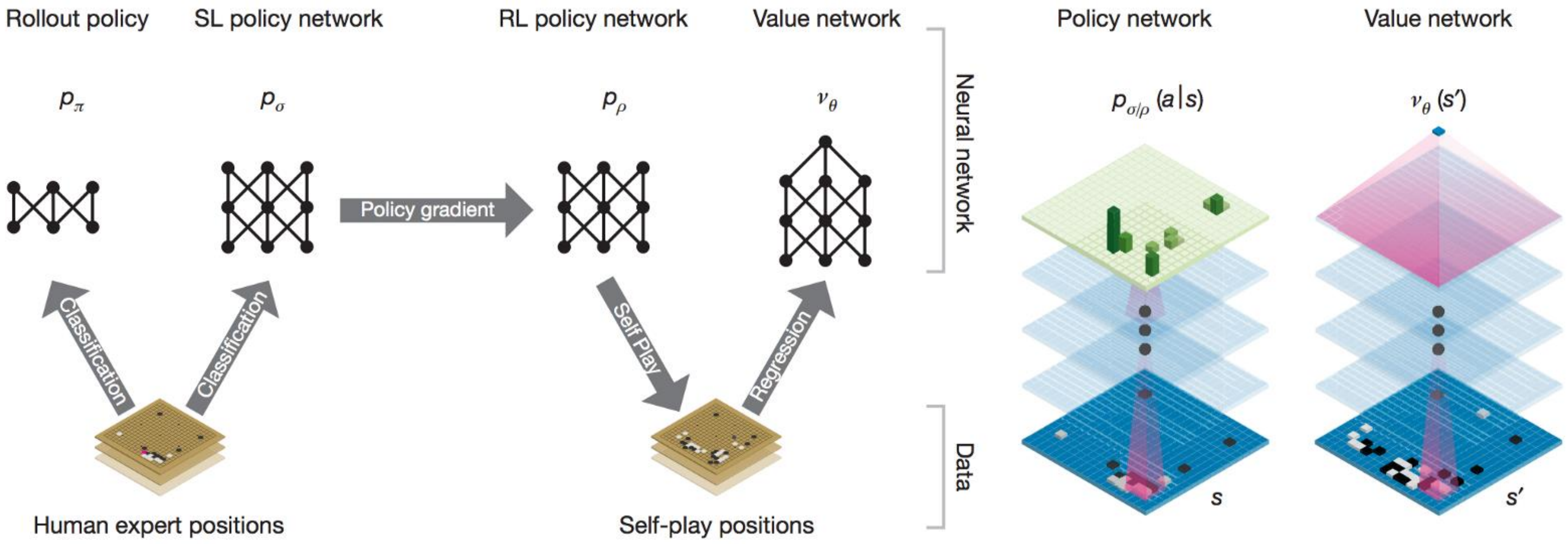
Output: scalar to represent the value of board.

Training Data: state-outcome pairs (s, z) from games played by using policy p for both players.

Pretrained + Fast Rollout + Policy + Value

NO MCTS yet!

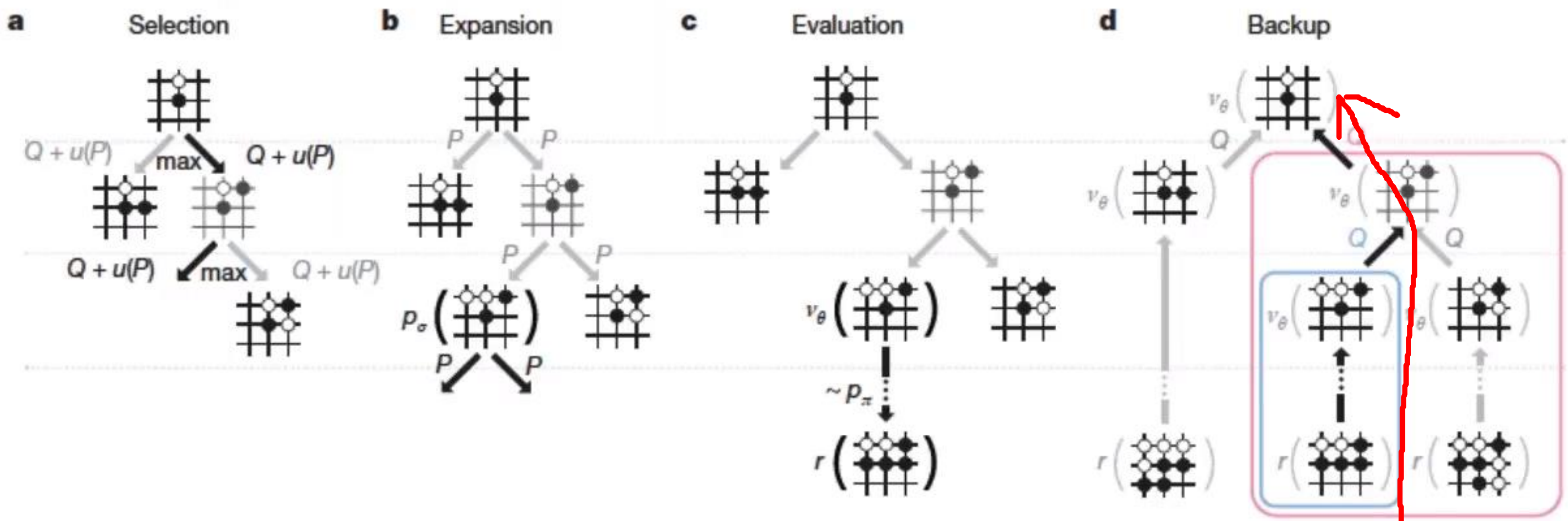
Sove GO with only Actor-Critic:
Policy: sum[R * delta{ logP(s|a) }]
Value: r_t = Z_T



Policy Improvement: MCTS

AlphaGo Solution

Searching with policy and value networks



$$a_t = \operatorname{argmax}_a(Q(s_t, a) + u(s_t, a))$$

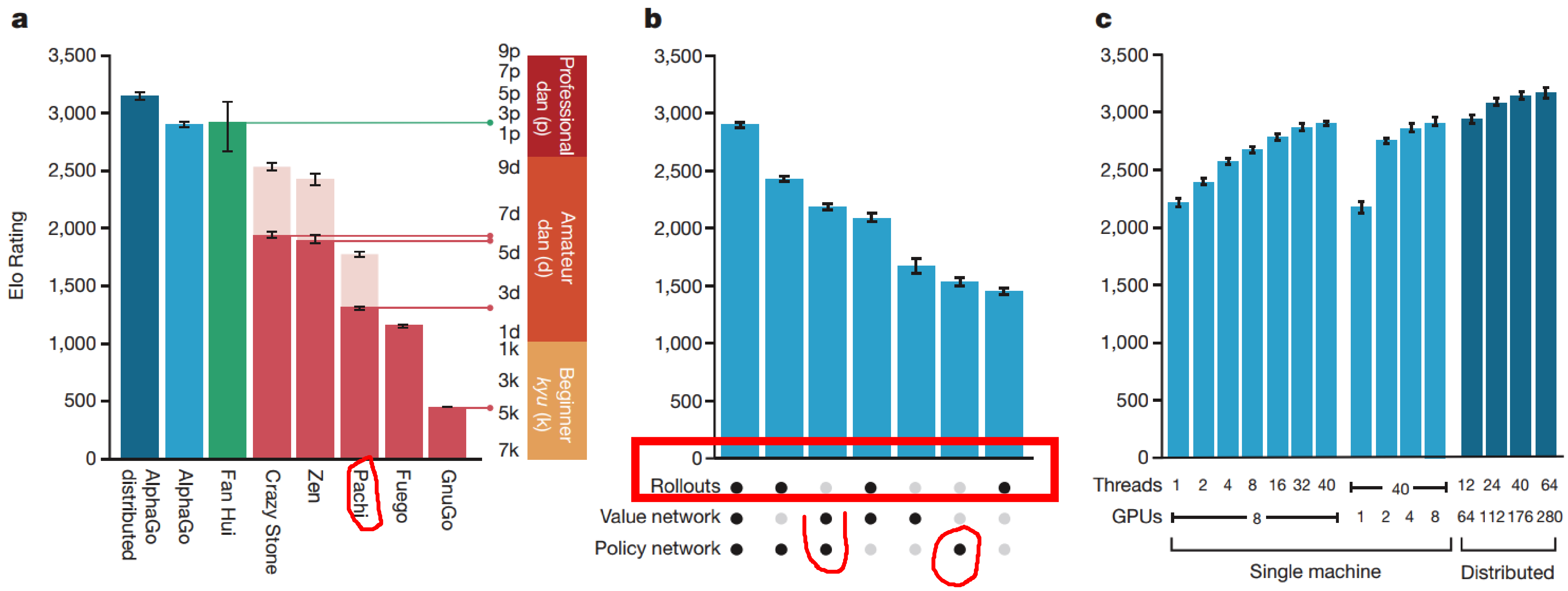
$$u(s,a) \propto \frac{P(s,a)}{1+N(s,a)}$$

$$V(s_L) = (1-\lambda)v_\theta(s_L) + \lambda z_L$$

$$N(s,a) = \sum_{i=1}^{n} 1(s,a,i)$$

$$Q(s,a) = \frac{1}{N(s,a)} \sum_{i=1}^{n} 1(s,a,i) V(s_L^i)$$

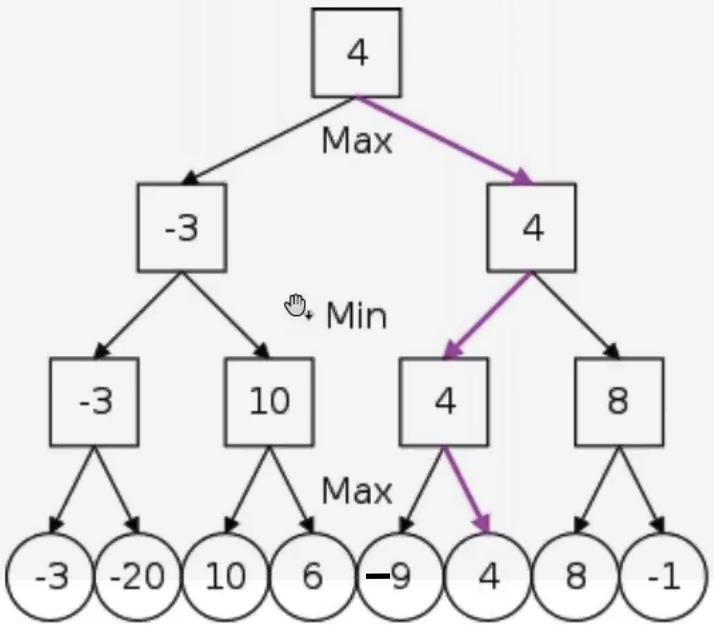
AlphaGo



a
3,500
3,000
2,500
2,000
1,500
1,000
500
0
Elo Rating
AlphaGo distributed
AlphaGo
Fan Hui
Crazy Stone
Zen
Pachi
Fuego
GnuGo
9p
7p
5p
3p
1p
9d
7d
5d
3d
1d
1k
3k
5k
7k
Professional dan (p)
Amateur dan (d)
Beginner kyu (k)
b
Rollouts
Value network
Policy network
c
Threads 1 2 4 8 16 32 40 40 12 24 40 64
GPUs 8 1 2 4 8 64 112 176 280
Single machine
Distributed

How to play Chess Games?

MinMax Tree Search



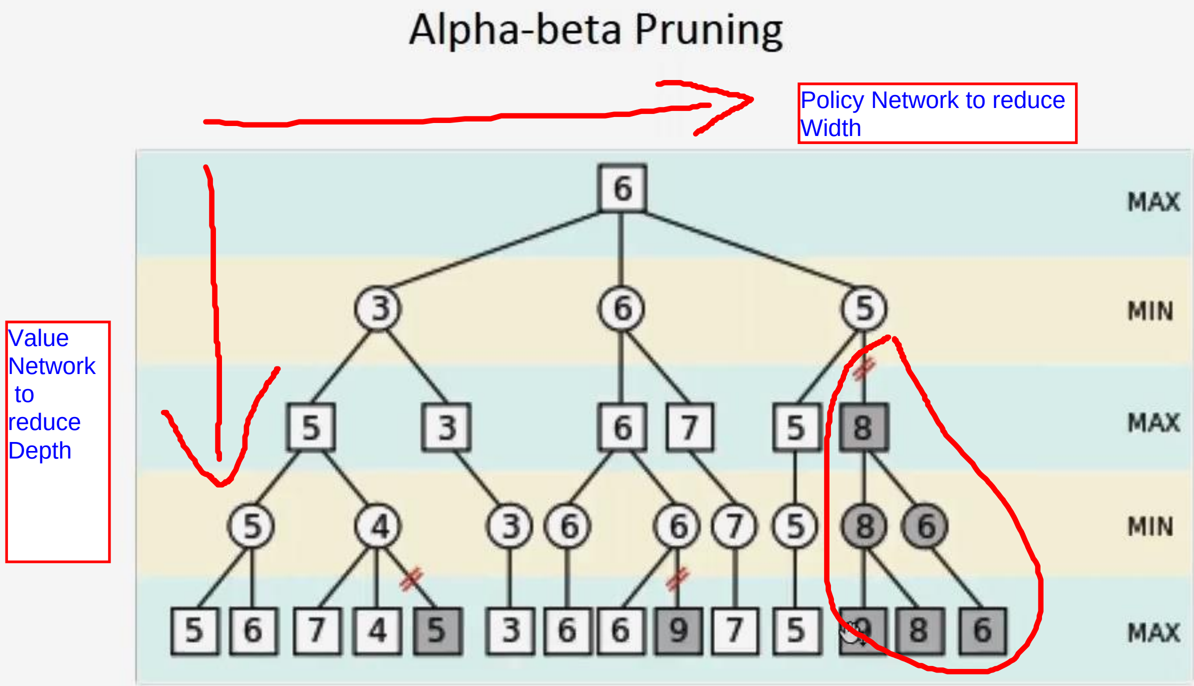
Win or Lose is destined!

Choose Best Step for both player.

MCTS

Turing

How to play Chess Games?



MCTS

Press Esc to exit full screen

How to play Chess Games?

Why we need Board Evaluation Function?

Search space is about $35^{80} \approx 10^{47}$, can NOT search the full space.

Deep Blue Solution (very brutal):

- Human Experts Engineered Evaluation Function.
- Alpha-Beta Tree Search up to 12 steps.

MCTS

Turing

Why Go game is so hard?

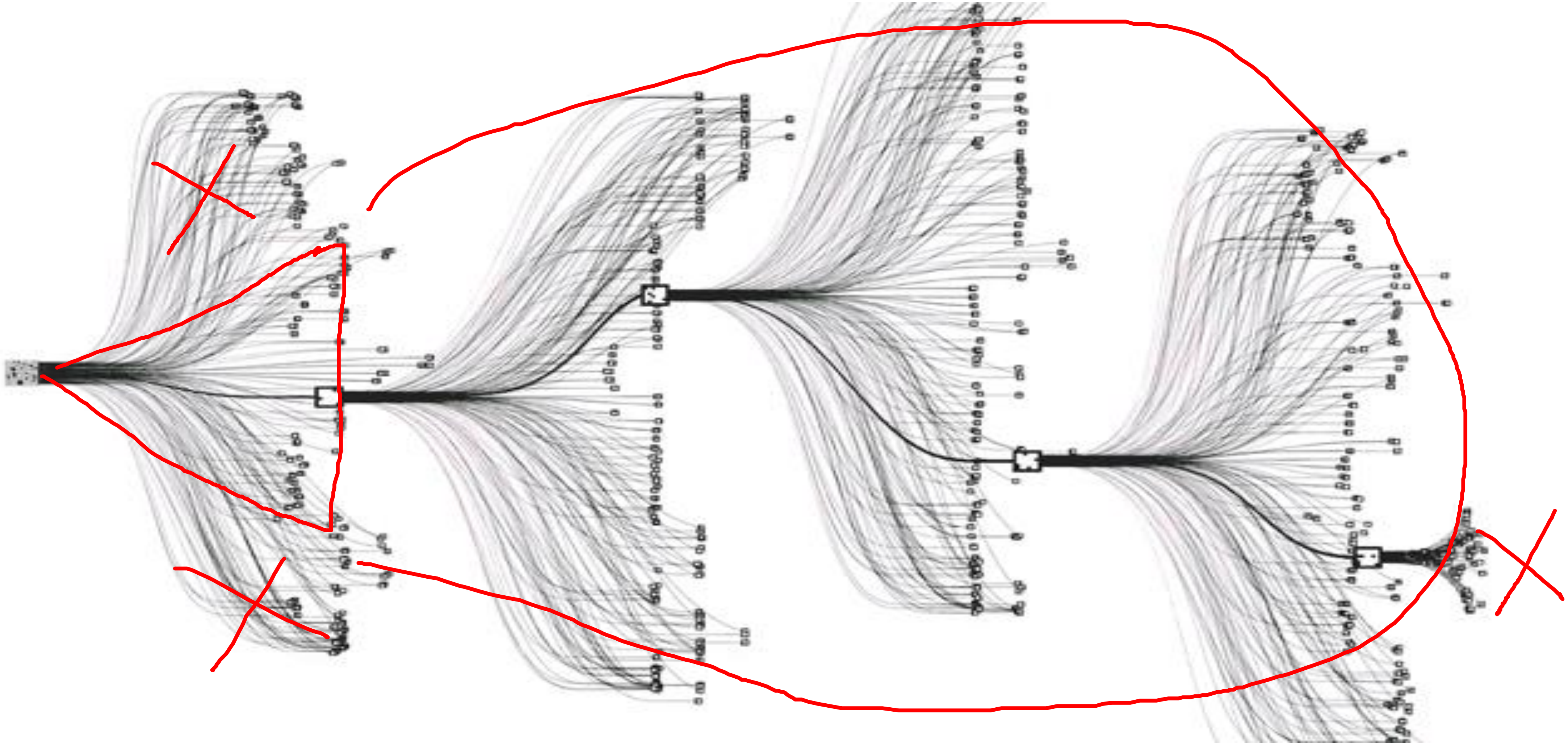
	Go	VS.	Chess
➢ Larger board:	19 X 19		8 X 8
➢ Average branching factor:	250		35
➢ Average depth:	150		80
➢ State Space Complexity:	10^{171}		10^{47}

Impossible to search multiple steps even with alpha-beta pruning!!!

- ➢ Every piece is equal.
- ➢ Global Impact of every single move.

Impossible to build an evaluation function!!!

Go Tree

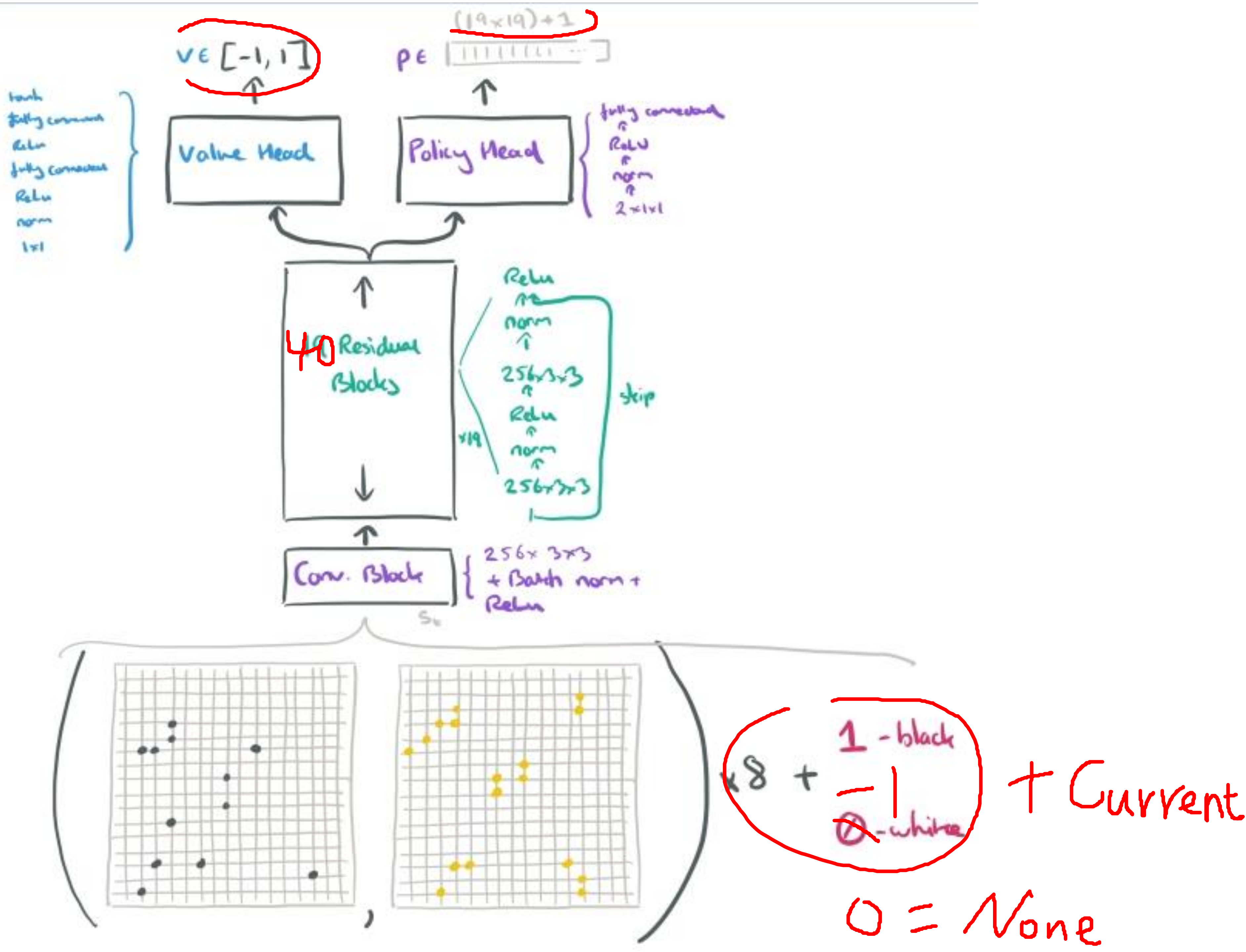


AlphaGo Zero

Without Human Knowledge

Zero vs AlphaGo

1. Self-play vs Play with older system
2. No Rollout
3. No Supevised pre-train
4. ResNet
5. Input maps 48 >> 17

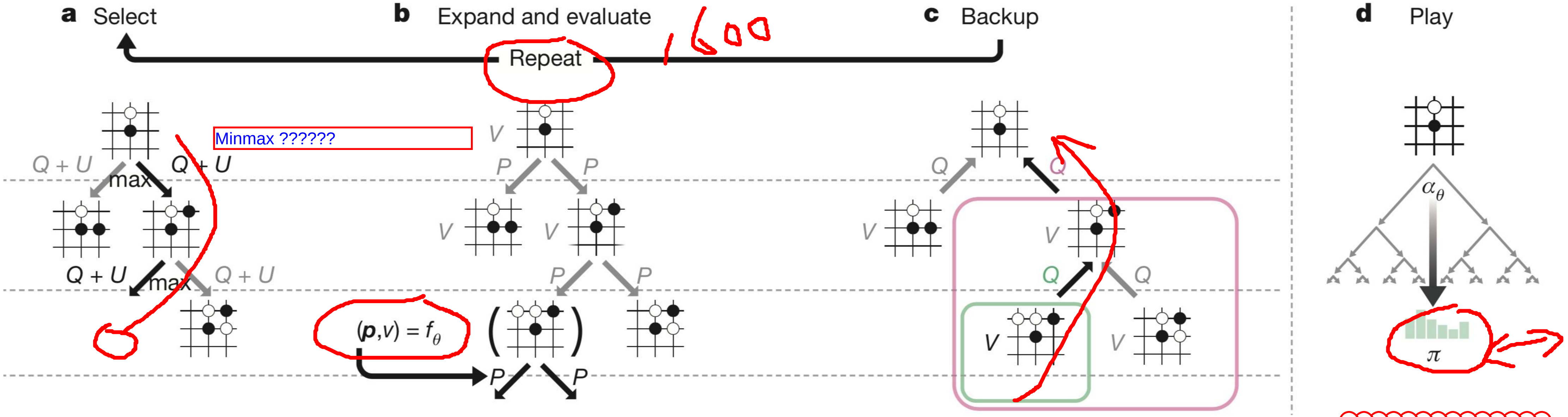


(19×19)+1
v ∈ [-1, 1]
p ∈
tanh
fully connected
Relu
fully connected
Relu
norm
1×1
Value Head
Policy Head
fully connected
ReLU
norm
2×1×1
40 Residual Blocks
Relu
norm
256×3×3
Relu
norm
256×3×3
skip
×19
Conv. Block
256× 3×3
+ Batch norm +
Relu
s_t
×8 +
1 - black
-1
0 - white
+ Current
0 = None

Forward

one step

Simulation



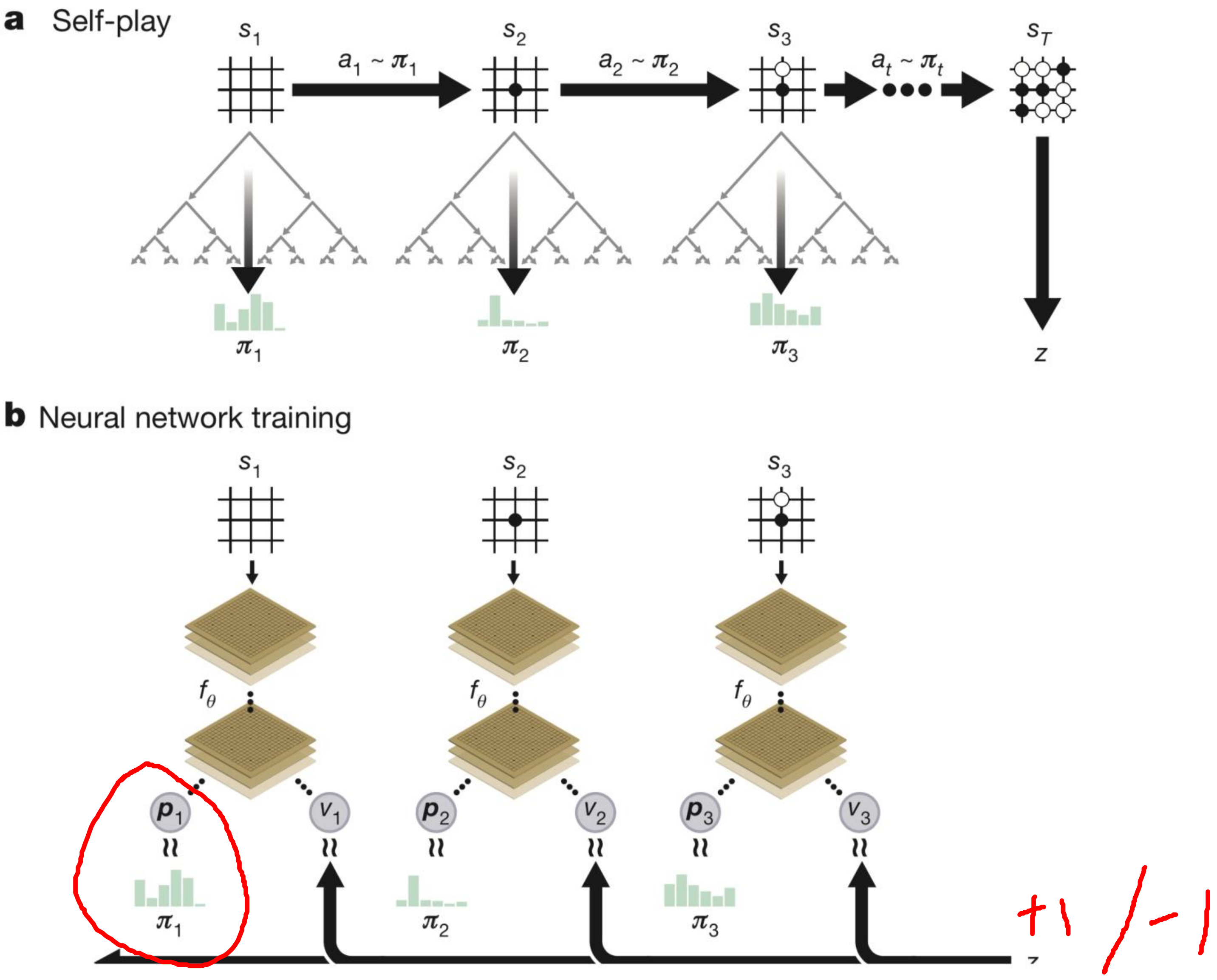
$$U(s,a) = c_{\text{puct}} P(s,a) \frac{\sqrt{\sum_b N(s,b)}}{1 + N(s,a)}$$

$$W(s_t, a_t) = W(s_t, a_t) + v, \; Q(s_t, a_t) = \frac{W(s_t, a_t)}{N(s_t, a_t)}$$

N++

Backward

$$l = (z - v)^2 - \pi^\top \log \mathbf{p} + c\|\theta\|^2$$



step

Simulation ×1600

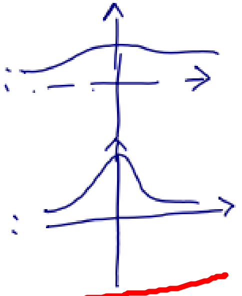
1. Selection: $\max(Q+U)$
2. expand leaf and evalute: $V(\text{leaf})$, $P(\text{next})$
3. BP: $N{+}{+}$, $W+V$, $Q = W/N$

$\downarrow$

play: $\pi \sim \dfrac{N(s,a)^{1/\tau}}{\sum_{cld} N^{1/\tau}}$

$\tau = 1$:

$\tau = 0$:



$$L = (z-v)^2 - \pi^T \log p + c\|\theta\|^2$$

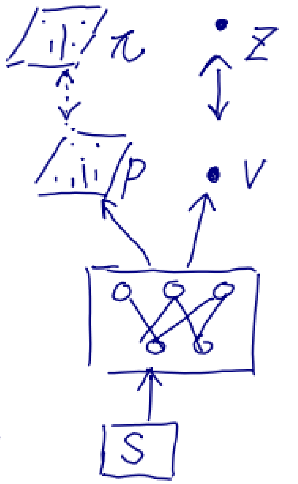
step

episode

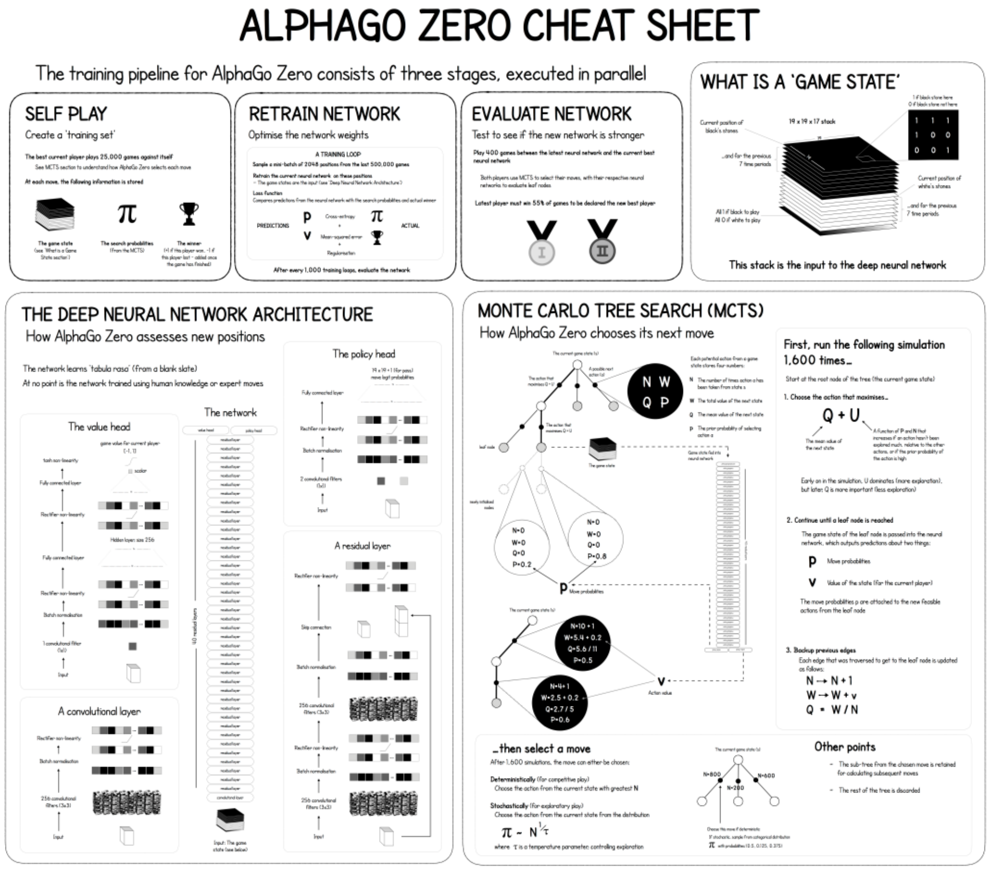
self-play: 25,000 Games

(S, π, z)

500,000 Games (batch: 2048) $\longrightarrow$



AlphaGo Zero In One Diagram

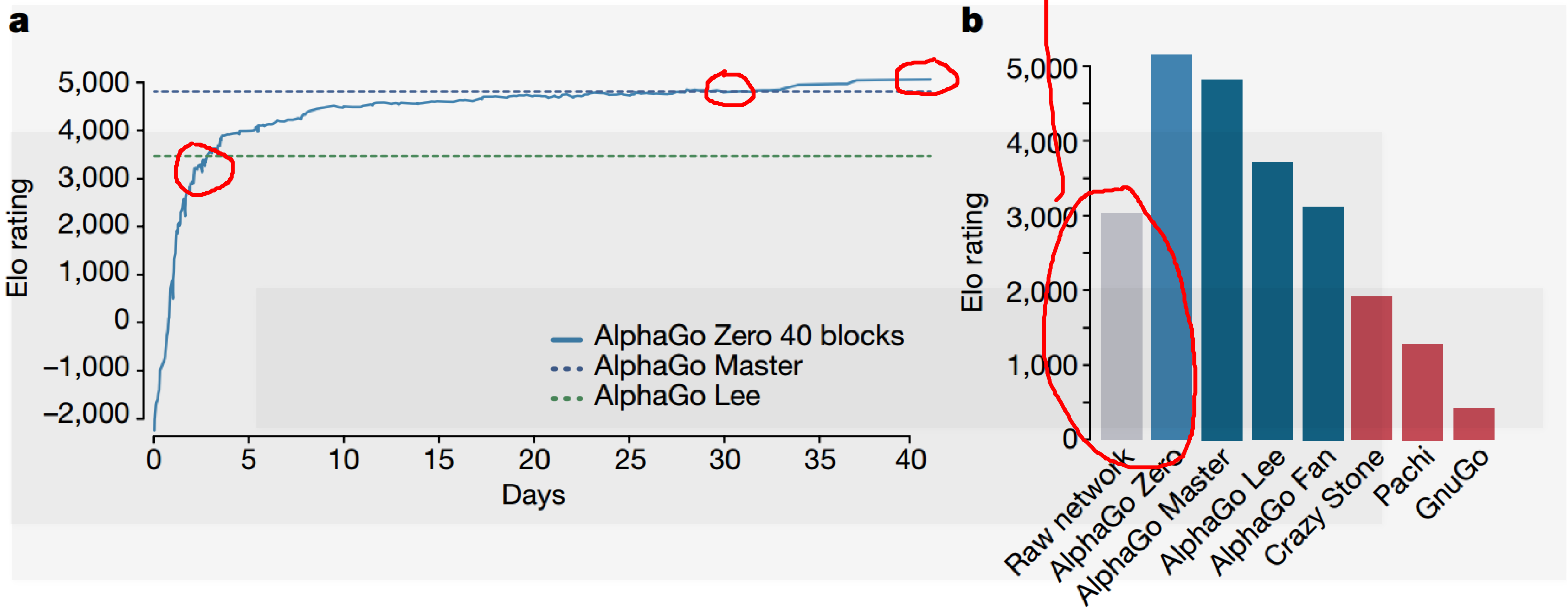


AlphaGo is just a intermediate product!

Ke Jie is defeated in April,
and Zero paper is submitted in Mayday.

AlphaGo Zero

Policy Improvement!



Thanks.

Merry Christmas!