Reinforcement Learning

Lecture 5 Applications

Nan Ye

School of Mathematics and Physics
The University of Queensland

THE UNIVERSITY OF QUEENSLAND AUSTRALIA

AMSI 24 WINTER SCHOOL
24 JUN – 5 JUL 2024 | UQ

Roadmap

- Introduction and overview

  motivation, bandits, big picture
- Classical ideas

  temporal difference methods, policy gradient, ...
- Deep Reinforcement learning

  neural networks, DQN, DDPG, ...
- Advanced techniques

  representation learning, stabilization, few-shot learning
- Applications

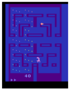
  AlphaGo, AlphaTensor, ...

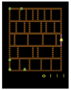
Deep Q-Networks (DQN) for Atari Games

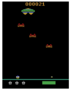
recall...

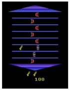

Adventure

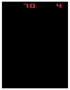

Air Raid


Alien


Amidar


Assault


Asterix


Asteroids


Atlantis

Atari

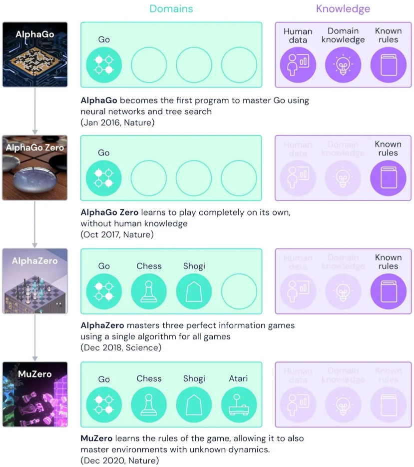
https://gymnasium.farama.org/environments/atari/

AlphaGo

source: DeepMind

first AI to beat a professional Go player, w/o handicap, on 19 $\times$ 19 board

Silver et al., Mastering the game of Go with deep neural networks and tree search, 2016



source: DeepMind

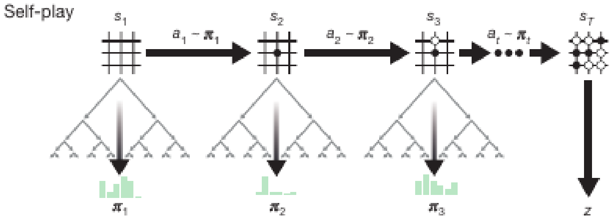
from AlphaGo to MuZero

AlphaGo Zero (Silver et al., 2017)

superhuman GO AI trained by self-play, w game rules, w/o human knowledge

loop(collect experiences by self-play + incremental updates of a policy & value network f_θ)

self-play... of an *improved* version of self



source: (ibid.)

each move given by an improved policy MCTS_{f_θ}

obtained by performing MCTS with the guidance of f_θ

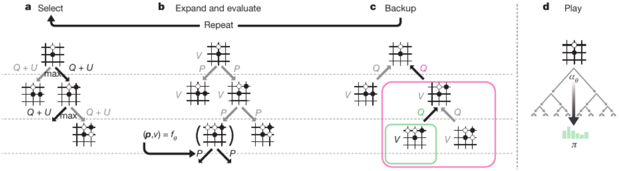
Silver et al., Mastering the game of Go without human knowledge, 2017

policy & value network

$$(p, v) = f_\theta(s),$$

maps a position s to move probabilities p and value v (+: winning; -: losing).

MCTS policy improvement



source: (Silver et al., 2017)

MCTS grows the search tree in an optimistic manner, using f_θ to evaluate leaf nodes

experience collection by self-play
experiences from one game: $(s_1, q_1, z_1), (s_2, q_2, z_2), \ldots$

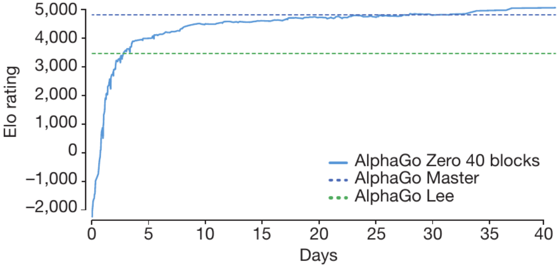
$q_t = \text{MCTS}_{f_\theta}(s_t)$ are the move probabilites given by the improved policy

the move for s_t is sampled from the move distribution q_t

$z_t = -1$ if the player at t lose in the end, $z_t = +1$ otherwise

incremental learning
perform gradient descent minimize loss on random mini-batch of experiences

$$L(\theta, (s, q, z)) = (z - v)^2 - q^\top \ln p + c\|\theta\|_2^2, \quad \text{where } (q, v) = f_\theta(s)$$



source: (Silver et al., 2017)

AlphaGo Lee: defeated Lee Sedol in Mar 2016; AlphaGo Master: defeated top human players by 60-0 in Jan 2017

learning curve for AlphaGo Zero

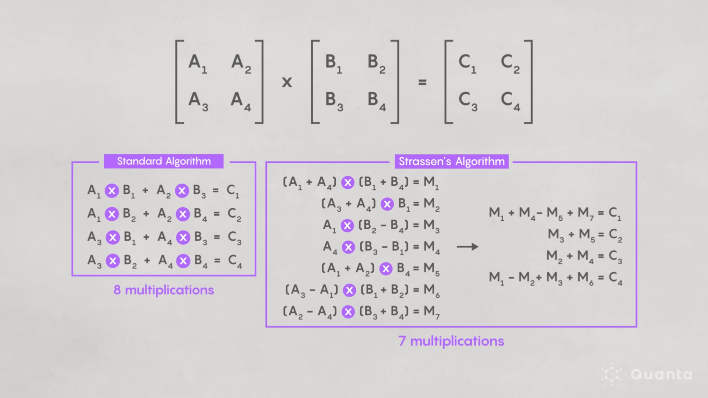
neural network with 84 parameterized layers
29 million games of self-play over 40 days on a single machine with 4 TPUs

what has AlphaGo Zero learned?

> *"AlphaGo Zero discovered a remarkable level of Go knowledge during its self-play training process. This included not only fundamental elements of human Go knowledge, but also non-standard strategies beyond the scope of traditional Go knowledge." (Silver et al., 2017)*

AlphaTensor

recall...



learning fast matrix multiplication

https://www.youtube.com/watch?v=fDAPJ7rvcUw what's the time complexity for multiplying two matrices $A, B \in \mathbb{R}^{n \times n}$?

standard algorithm: $O(n^3)$

Strassen's algorithm: $O(n^{2.8074})$

7 multiplications when $n = 2$, best possible

used in practice for large matrices

exists asymptotically better algorithms (record = $O(n^{2.371552})$), but galactic

AlphaTensor: $O(n^{2.778})$ for matrices in finite filed $\mathbb{Z}_2$ (Fawzi et al., 2022)

outperforms Strassen's algorithm in practice!

Fawzi et al., Discovering faster matrix multiplication algorithms with reinforcement learning, 2022

discovering matrix multiplication as sequential decision making

Step 1. matrix multiplication = a 3D tensor T

Step 2. low-rank decomposition of T = a multiplication algorithm

Step 3. search for a low-rank decomposition $\in$ sequential decision making

Step 1. matrix multiplication = a 3D tensor T

exists an $n^2 \times n^2 \times n^2$ tensor T, for any $n \times n$ matrices A, B, C,

$$C = AB \quad \text{equivalent to} \quad c_k = \sum_{i,j} T_{ijk} a_i b_j, \quad \forall k.$$

intuitively, $T_{ijk} = 1$ if the product of the i-th entry of A and the j-th entry of B is a term in the kth entry of $C = AB$.

e.g., $T_{(1,2),(2,3),(1,3)} = 1$, $T_{(1,2),(1,3),(1,3)} = 0$

Step 2. low-rank decompsition of T = a multiplication algorithm

$$T = \sum_{r=1}^{R} u^{(r)} \otimes v^{(r)} \otimes w^{(r)} \quad \Rightarrow \quad c_k = \sum_{r=1}^{R} w_k^{(r)} A^{(r)} B^{(r)}, \quad \text{where}$$

$$A^{(r)} = \sum_i u_i^{(r)} a_i, \quad B^{(r)} = \sum_j v_j^{(r)} b_j$$

algorithm: compute each $A^{(r)}$, $B^{(r)}$ and $C^{(r)} = A^{(r)} B^{(r)}$ first, then compute each c_k

complexity: $\leq \underbrace{2Rn^2}_{n^2 \text{ for each } A^{(r)},\, B^{(r)}} + \underbrace{R}_{1 \text{ for each } C^{(r)}} + \underbrace{Rn^2}_{R \text{ for each } c_k} = 3Rn^2 + R$ multiplications.

make it faster:

many zeros or ones in $u^{(r)}$ $\Rightarrow$ computing $A^{(r)}$ takes far fewer than n^2 $\times$

many zeros or ones in $v^{(r)}$ $\Rightarrow$ computing $B^{(r)}$ takes far fewer than n^2 $\times$

zeros or ones in $w^{(r)}$ $\Rightarrow$ computing c_k takes fewer than $2R$ $\times$

$$\begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix} = \begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix} \begin{pmatrix} b_1 & b_2 \\ b_3 & b_4 \end{pmatrix}$$

Strassen's algorithm

r	$A^{(r)}$	$B^{(r)}$
1	$a_1 + a_4$	$b_1 + b_4$
2	$a_3 + a_4$	b_1
3	a_1	$b_2 - b_4$
4	a_4	$b_3 - b_1$
5	$a_1 + a_2$	b_4
6	$a_3 - a_1$	$b_1 + b_2$
7	$a_2 - a_4$	$b_3 + b_4$

$$\begin{aligned} c_1 &= C^{(1)} + C^{(4)} - C^{(5)} + C^{(7)} \\ c_2 &= C^{(3)} + C^{(5)} \\ c_3 &= C^{(2)} + C^{(4)} \\ c_4 &= C^{(1)} - C^{(2)} + C^{(3)} + C^{(6)} \end{aligned}$$

0 $\times$'s for $A^{(r)}$, $B^{(r)}$ and c_k, only 7 $\times$'s for $C^{(r)}$

Step 3. search for a low-rank decomposition $\in$ sequential decision making

search for a low-rank decomposition by subtracting one rank-1 tensor at a time

Algorithm Simulating a policy π_θ for TensorGame

```
Initial state $S_0 = T$
for $t = 0$ to $M$ do
    Sample an action $(u, v, w)$ from a policy $\pi_\theta(S_t)$
    Transition to next state $S_{t+1} = S_t - u \otimes v \otimes w$
    if $S_{t+1} = 0$ then
        break; //an exact decomposition has been found
    Apply a penalty = $\begin{cases} -1, & t < M, \\ -\gamma(S_{t+1}), & t = M, \end{cases}$, where $\gamma(S_{t+1})$ is an upper bound on the rank of $S_{t+1}$
```

note the state, action, transition, and reward

Some tricks in implementation

- Entries of u, v, w vectors are constrained to take values from a small set of integers, so as to guarantee an exact decomposition that can be provably verified.
- A policy and value network: transformer, invariance to permutation of slices.
- Synthetic demonstrations (generated tensor decompositions)
- Random change of basis of T
- Data augmentation (generate new examples by changing order of summation)

These Lectures

Reinforcement Learning (RL)

Goals

- cover mathematical & algorithmic foundation
- in-depth look at a few cool applications
- develop basic practical skills

The End, The Beginning

- Introduction and overview

  origin, applications, regret min for bandits, RL loop, four dimensions, three problems

- Classical ideas

  MDPs, Gym, value iteration, Q-learning, SARSA, REINFORCE, model-based RL

- Deep Reinforcement learning

  ANNs in 5 mins, DQN, Rainbow DQN, DDPG, RL software

- Advanced techniques

  representation learning (CURL, SPR), stabilization (TRPO, PPO), few-shot learning

- Applications

  AlphaGo Zero, AlphaTensor

many more not covered, numerous new things, see NeurIPS, ICML, ICLR, . . .

try the ideas out when it comes to decision-making

References I

Fawzi, A. et al. (2022). Discovering faster matrix multiplication algorithms with reinforcement learning. In: *Nature* 610.7930, pp. 47–53.

Silver, D. et al. (2016). Mastering the game of Go with deep neural networks and tree search. In: *Nature* 529.7587, pp. 484–489.

Silver, D. et al. (2017). Mastering the game of Go without human knowledge. In: *Nature* 550.7676, p. 354.