**LECTURE 20**

**AlphaGo**

\[
\text{state} = \begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]

\[
S
\]

**Action** $E$

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix} + \text{pass}
\]

19x19 positions

**Policy:** $P(a | s)$

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix} + \text{pass}
\]

19x19

**Value:** $v(s) = \mathbb{E}_{policy} \left[ z | s \right]$

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]

\[
\begin{pmatrix}
\text{average} \\
+1 \\
+1 \\
0 \\
+1 \\
0 \\
-1
\end{pmatrix}
\]

**Deep RL**

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]

\[
\begin{pmatrix}
\circ \\
\circ \\
\circ
\end{pmatrix}
\]
\[
\Delta \theta \propto -\frac{\partial}{\partial \theta} (z - V\theta(s))^2
\]

\[
\Delta \sigma \propto \frac{\partial}{\partial \sigma} \log p_\sigma(\text{als})
\]

Supervised:
\[
\Delta \sigma \propto \frac{\partial}{\partial \sigma} \log p_\sigma(\text{als})
\]

\[
\Delta p \propto \frac{\partial}{\partial p} \log p_p(\text{als}), \quad z \uparrow +1, \quad z \downarrow -1
\]

Stochastic gradient ascent on \( E(z) \)
Try to max expected payout

Policy gradient
\[
\max E_{p_p(\text{als})} \left[ z \right] = \sum_a z \cdot p_p(\text{als})
\]

Average over payoffs of all trajectories

\[
\frac{\partial}{\partial p} E(z) = \sum_a z \frac{\partial}{\partial p} p_p(\text{als})
\]

\[
= \sum_a \left[ z \cdot \frac{\partial}{\partial p} \log p_p(\text{als}) \right] \times p_p(\text{als})
\]

\[
= E_{p_p(\text{als})} \left[ z \cdot \frac{\partial}{\partial p} \log p_p(\text{als}) \right]
\]

Learn \( V\theta(s) \)
play the game

\[ a_1 \sim \nu_p(a|s_1) \quad \text{or} \quad a_1 \rightarrow \max \ V_0(s_2) \quad \text{fast thinking} \]

current state

slow thinking in imagination think ahead

Which one is more accurate?

- \( V_0(s_{20}) \) because closer to the end of game
  - more accurate estimate of average score

monte-carlo tree search

reduce depth

choose best \( a_1 \)

by \( \max Q(s_1, a) \)

or sample \( a \) of \( N(s_1, a) \)

MCTS

step 1: selection: go down a branch

branch

leaf

\[ s_1 \rightarrow s_2 \rightarrow V(s_2) \rightarrow V(s_3) \rightarrow \text{go back to do update} \rightarrow \text{grow the tree} \rightarrow \text{each time choose} \ a_1 \ \text{based on criteria} \rightarrow \text{can't see full game tree, randomly select same branches} \]
Step 2: expand

- Leaf

Step 3: back up

- Go back the branch

Back-up

- \[ N(S,a) = N(S,a) + 1 \] (# of visits)
- \[ w(S,a) = w(S,a) + V \] (total score)
- \[ Q(S,a) = w(S,a) / N(S,a) \]

Categorize over leaf

- \[ S \rightarrow A \]
- \[ 1 \rightarrow \text{average} \]
- \[ 0 \rightarrow \text{average} \]
- \[ -1 \rightarrow \]

Selection

- \[ S \rightarrow A \]
- \[ \text{choose } a \text{ by max } Q(S,a) + \frac{1}{\text{uncertainty}} \]

- Balance between what know best and exploring

- Policy according to supervised learning gives good results
- Policy for unknown states gives randomness
- Reduces breadth

1. Train 2 policy networks - one supervised
2. Train value network - one RL
3. MCTS
Alpha Go Zero

- No human data (CP6)
- Self-play by MCTS

\[ S \xrightarrow{a} S' \xrightarrow{a} S'' \rightarrow \ldots \rightarrow S \]

- Subconscious vs. conscious thinking

RL in general

Markov decision process (MDP)

\[ S_0 \rightarrow \ldots \rightarrow S_t \rightarrow S_{t+1} \rightarrow \ldots \]

World Model

1. Dynamics model:
   - Deterministic: \( S_{t+1} = F(S_t, a_t) \)
   - Stochastic: \( P(S_{t+1} | S_t, a_t) \)

2. Reward model:
   - Deterministic: \( r_t = R(S_t, a_t) \)
   - Stochastic: \( P(r_t | S_t, a_t, S_{t+1}) \)

Reward to go: \( G_t = r_t + r_{t+1} + \ldots \)

Goal: \( \max E(G_t) \)  
  - Policy: PCALS
  - Value: V(S), Q(S, a)

Model-based

- Known (1), (2), play by by MCTS
- Play out in imagination

Model-free

- Do not know (1), (2)
- Play out in real environment, practice in real world

Swimming, riding bicycle