AlphaGo

2016/06/15

@mabonki0725

AlphaGo 着手先は総合評価

a. Value net
価値関数Vの分布

b. evalation value net
着手価値関数Q（λ=0)

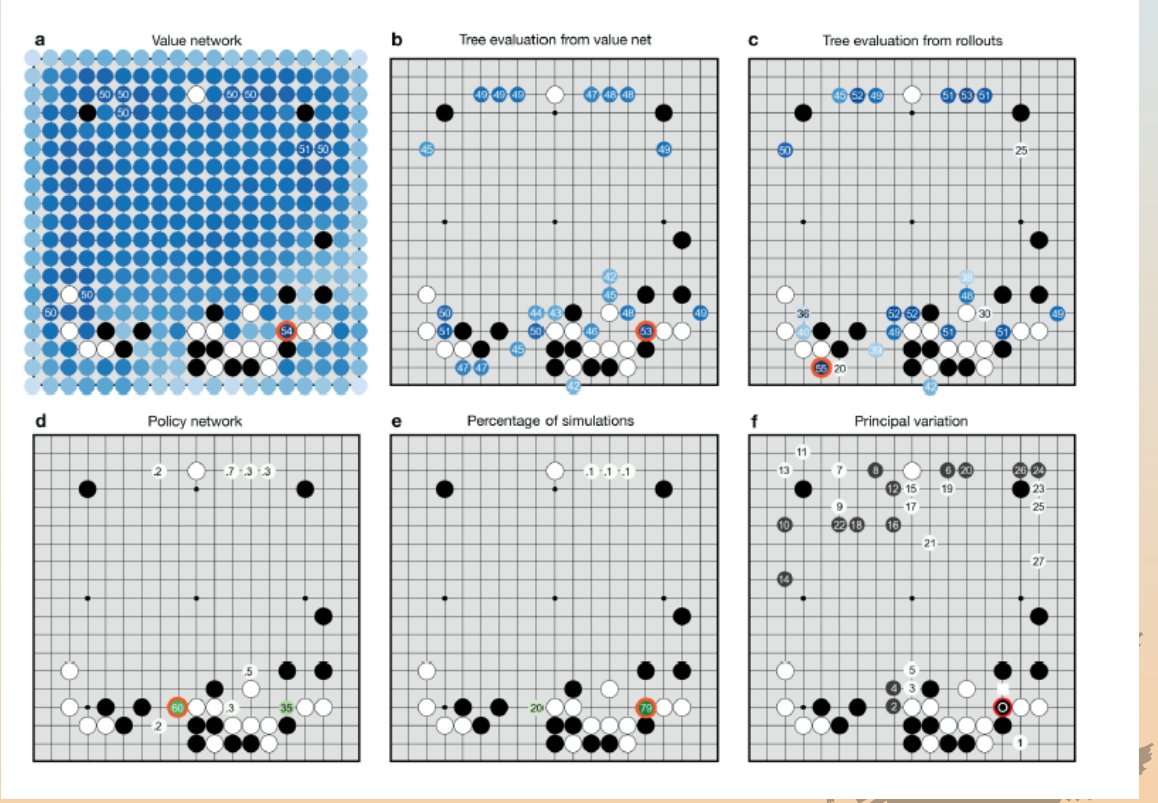
c. evalation rollout
勝率予測率

d. policy net
着手確率Pσ

e. simulation
繰返し対戦で選択された
着手回数(率) N

f. principal valuation
総合的評価に選択された
総合価値関数 Q+d/N

Q = e*{a*(1-λ) + c*λ)
λ=混合割合 (0.5)



○は最大値

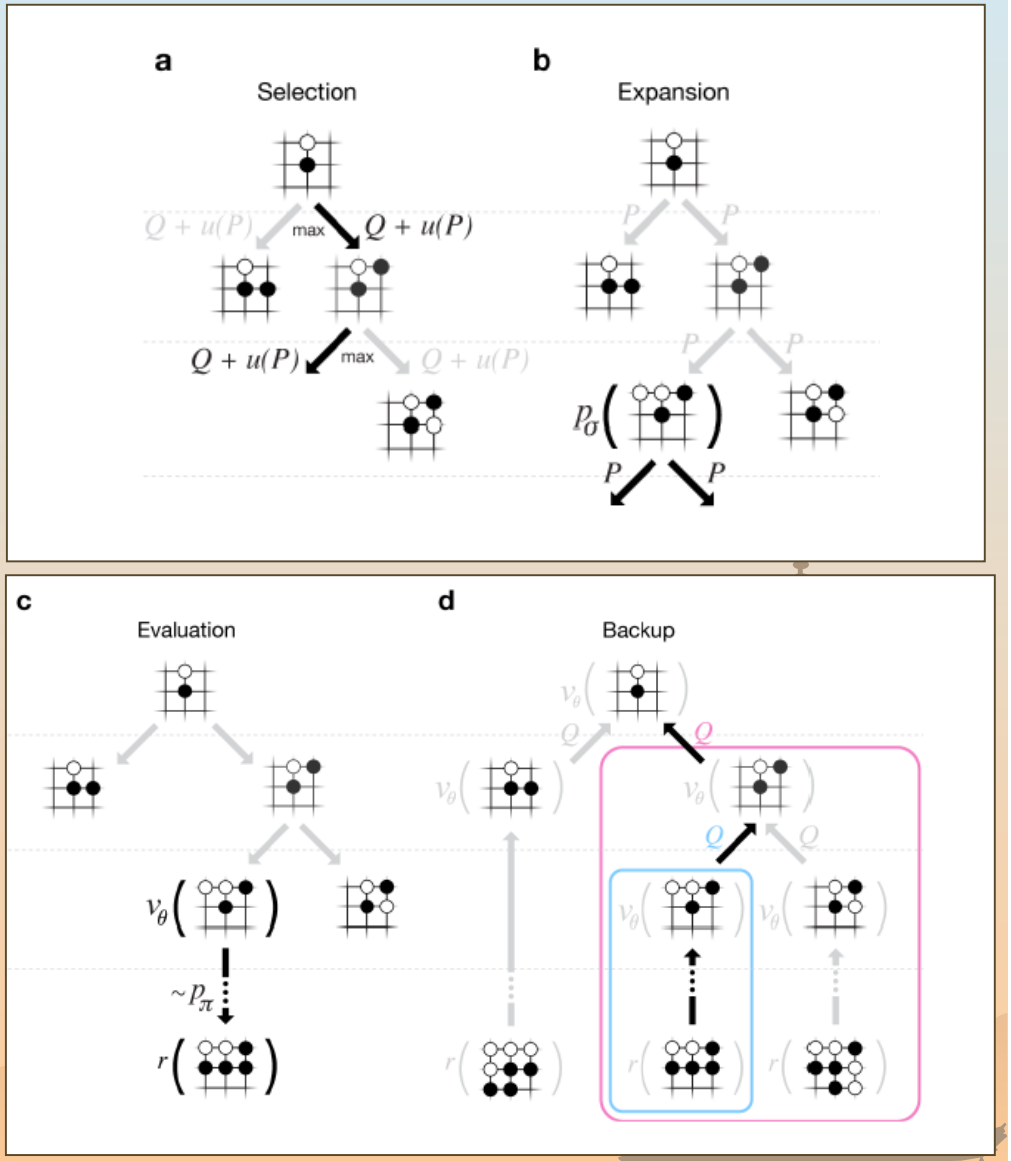
AlphaGoの繰返学習

a. Selection
探査木を辿ってQ+uが最大の手を探す　Q:着手価値
u=着手確率/出現回数

b. Expansion
末端到着すると探索木をPσ範囲で広げる Pσ:着手確率

c. Evaluation
早碁モデルPπで勝負を予測し価値V_θのθを更新

d. Backup
全末端でも勝敗(rollout)計算
V_θとQを更新

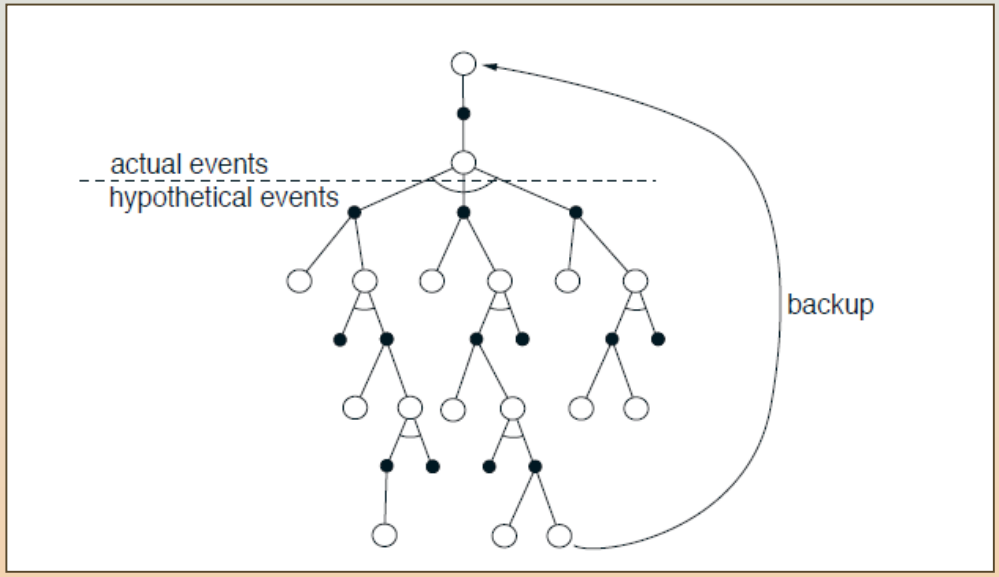


強化学習(前提知識)

現在sの価値関数V(s)は将来の報酬の期待値の合計

報酬:途中の得点あり:野球、ゲーム

報酬:途中の得点なし:碁、将棋、迷路(最後のみ報酬)



$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s\right] \\
&= \mathbb{E}_\pi\left[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s\right] \\
&= \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)\left[r(s,a,s') + \gamma v_\pi(s')\right]
\end{aligned}
$$

Rt:報酬(t時点)

St:状態(t時点)

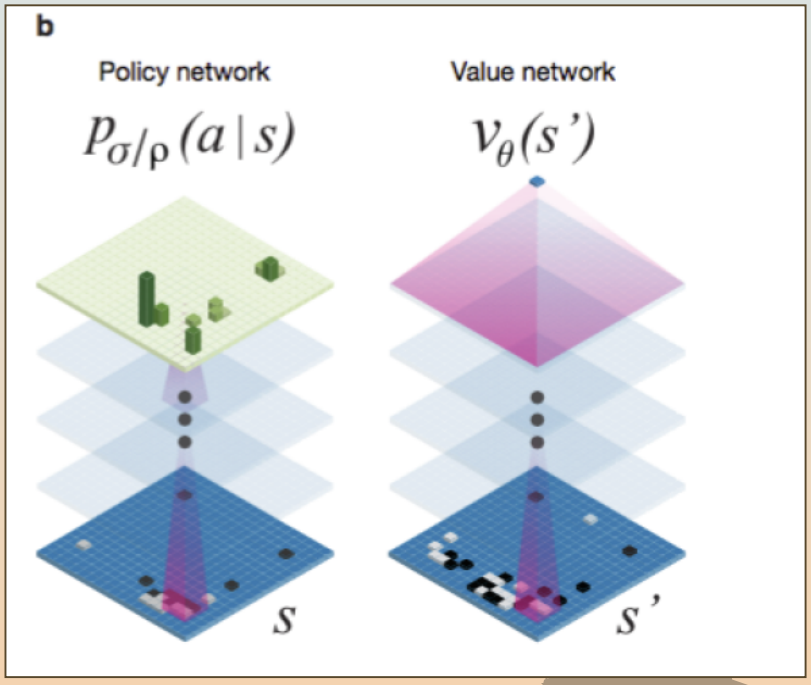
γ:割引率(将来価値は割引)

探査空間の絞込み戦略

- 局面の着手点の絞込み
  - 深層モデルで着手確率分布 p(a/s)を学習　a:着手点 s:局面
- 局面の遷移の絞込み
  - 出現確率が低い局面の推移は省く
- 着手点の価値関数の近似
  - 着手先の価値関数V_θの近似(DQN)

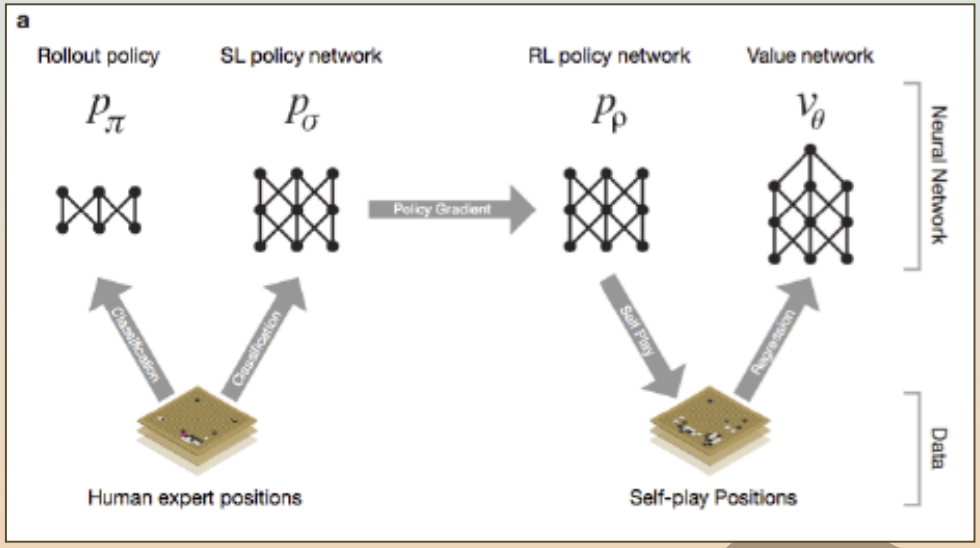
    $V_\theta = \Sigma$ 特徴量$* \theta$　　θ : 重み

    rollout(決着時)の勝負予測に合う様に重みθを学習



パイプライン（段階的精緻化）方式

1. 棋譜による学習
   - Pπ着手確率（早碁モデル）
     勝敗予想に使用
   - Pσ着手確率(SLモデル)
2. Alpha碁対戦
   - Pρ着手確率（RLモデル）
     初期値はPσ
   - V_θ価値関数近似モデル
     勝敗予想より計算



棋譜　SL(Pσ)着手確率モデル

- 13層のCNN型深層学習 重みσをSGDで最適化

$$\Delta\sigma \propto \frac{\partial \log p_\sigma(a|s)}{\partial \sigma}$$

- 入力:局面sの特徴量　出力:次の着手確率Pσ
- プロの着手を予想　57%正解率

Feature	# of planes	Description
Stone colour	3	Player stone / opponent stone / empty
Ones	1	A constant plane filled with 1
Turns since	8	How many turns since a move was played
Liberties	8	Number of liberties (empty adjacent points)
Capture size	8	How many opponent stones would be captured
Self-atari size	8	How many of own stones would be captured
Liberties after move	8	Number of liberties after this move is played
Ladder capture	1	Whether a move at this point is a successful ladder capture
Ladder escape	1	Whether a move at this point is a successful ladder escape
Sensibleness	1	Whether a move is legal and does not fill its own eyes
Zeros	1	A constant plane filled with 0
Player color	1	Whether current player is black

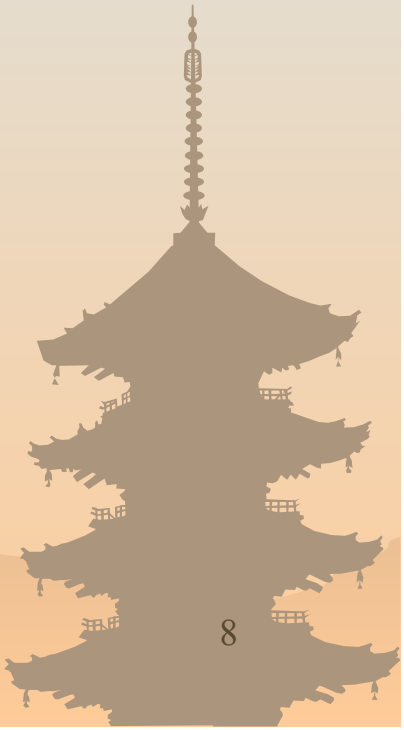
Alpha碁 RL(P ρ) 着手確率モデル

- 深層学習の重み初期値 $\rho = \sigma$ (SLの重み)
- Alpha碁どうしで繰返し対戦させる
- 重み ρ は勝敗予想Zt(1 or -1)を掛けて勝つ方向に傾斜させる

$$\Delta\rho \propto \frac{\partial \log p_\rho(a_t|s_t)}{\partial \rho} z_t$$

傾斜方向　負け予想:減少　勝ち予想:増大
勝敗予想　Pπ(早碁モデル)使用か？

- RLモデルの対戦勝率
  - –SLモデル　勝率80%
  - –Pachi碁ソフト(2段レベル)　勝率 87%



価値関数V(s')近似

- 価値関数V(s'):着手後の価値を計測する
  s':RLモデルのa～Pρ(a|s)による着手後の局面
- 価値関数を勝敗予測に一致する様に近似
  - 線形近似　$v_\theta = \Sigma$ 特徴量$*\theta$　SGD
  - 非線形近似　$v_\theta = \Phi$(特徴量、θ)　深層学習

  z:勝敗予測　Pπ(早碁モデル)で推定
  $z = v_\theta(s)$となる重みθを推定

$$\Delta\theta \propto \frac{\partial v_\theta(s)}{\partial\theta}(z - v_\theta(s))$$

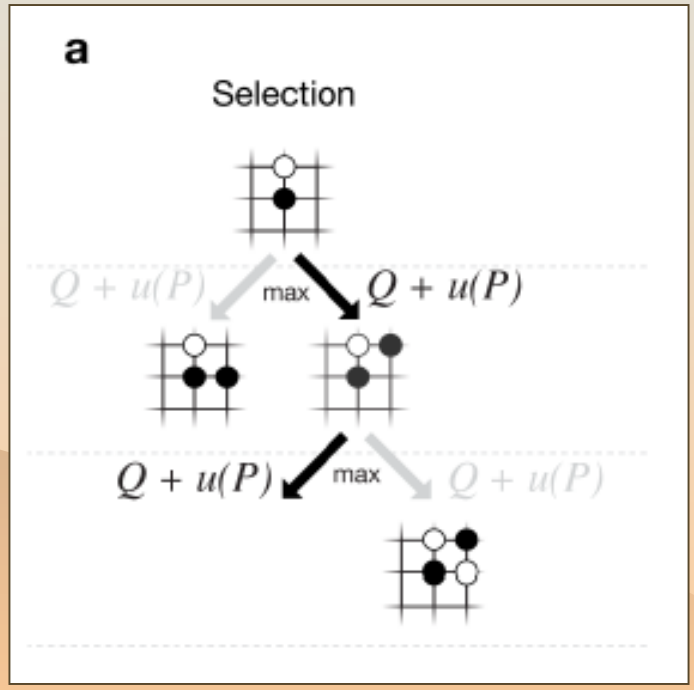
DQNのQ-learningと同じ手法

最適着手価値関数Q(s,a)

- 最適着手aはQ+uの最大値で選択

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

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



N(s,a) :対戦の繰返しで出現回数

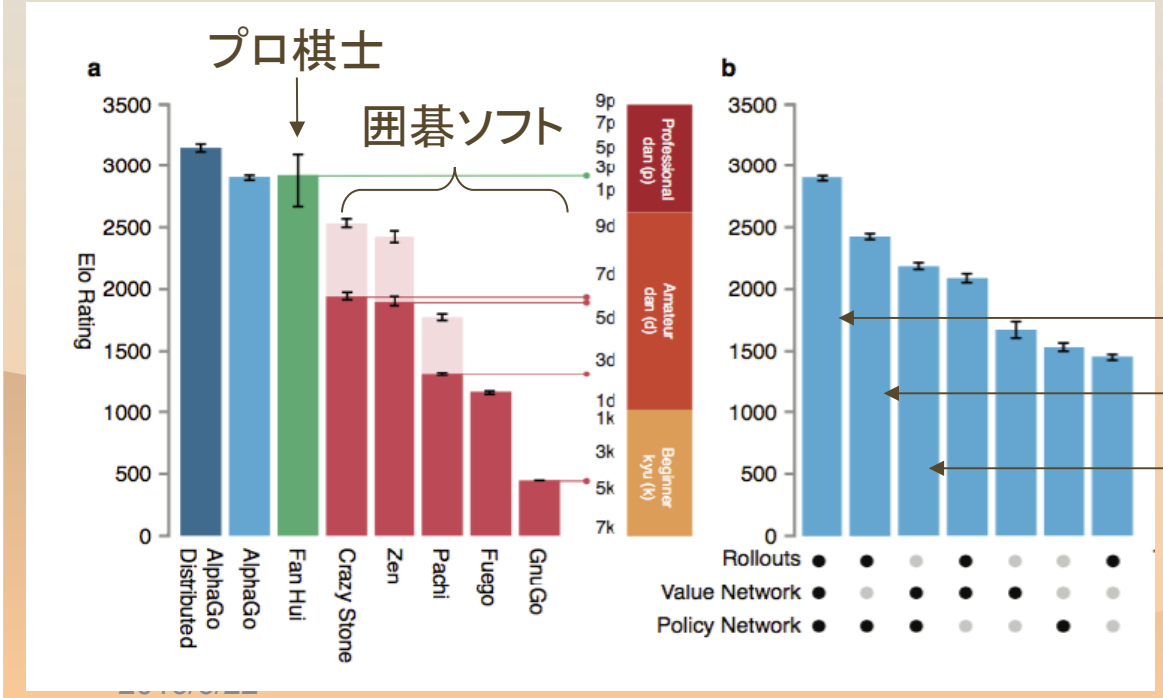
P(s,a): Pσ着手確率

最適着手価値関数Q(s,a)

Q(s,a)関数：対戦繰返しをn回繰返した平均

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

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



zL:勝敗予測

$v_\theta(sL)$:価値関数

λ:混合割合

λ =0.5

λ =1

λ =0

参考文献

- Mastering the Game of Go with Deep Neural Networks and Tree Search *DeepMind* Nature
- Reinforcement Learning *Sutton*
- Playing Atari with Deep Reinforcement Learning *DeepMind*
- 特徴量関数近似による強化学習の汎用化 *M.Nakai*