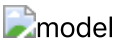
評価関数について

ニューラルネットワークのフレームワークを実装し、下図のようなモデルを使用した。



出力層をシグモイド関数にすることで、ニューラルネットワークの出力が $(0,1)$ に制限され、手番の勝率と対応するようにした。

入力について

局面を表す361次元の数値が入力となっている。局面をベクトルに変換する方法について説明する。

駒のきき(300次元)

手番と非手番の駒の種類ごとに、各マスについていくつのききがあるかを表現した。駒がある場所ではなく、駒のききを表現することで、駒の動きを学習する必要がなくなると考えた。
(例)
手番側の銀が次の位置にいるとき

	A	B	C	D	E
5					
4					
3		全	銀		
2					
1					

入力データは次の二次元グリッドを一次元に変換したものである。

	A	B	C	D	E
5	0	0	0	0	0
4	1	2	2	1	0
3	1	0	1	0	0
2	0	2	0	1	0
1	0	0	0	0	0

駒の有無(50次元)

手番と非手番の、各マスにおける駒の有無をそれぞれ25次元で表現した。

持ち駒の数(10次元)

手番と非手番の持ち駒の枚数を各駒について表現した。

先手か後手か(1次元)

先手か後手かを表現した。千日手が基本的に後手勝利となることから、先手と後手は非対称であり、このような入力を追加した。

強化学習について

強化学習におけるモンテカルロ法を参考にした。
自己対戦で棋譜を生成し、状態が $s_0,s_1,...,s_n$ のように遷移したものとする。ただし、状態は手番から見た形で表現されているものとする。状態 s_i におけるニューラルネットワークの出力を $V(s_i)$ としたとき、添字が大きい順に以下の式で教師データを生成する。

$$ V(s_i) \leftarrow (1-\alpha)V(s_i) + \alpha R_{i} $$

$$ R_i = \left{ \begin{array}{ll} 1 - ((1-\gamma)V(s_{i+1}) + \gamma R_{i+1}) & (i < n) \ 0 & (i = n) \end{array} \right.$$

R_i を更新するときに 1 から前の値を引いているのは、手番から見た形で状態が表現されており、先手と後手が毎回入れ替わるからである。

誤差逆伝播法について

誤差逆伝播法を用いてパラメータを自動調整しているが、勾配消失を防ぐため、出力層のシグモイド関数の微分を常に1としている。

参考文献

斎藤康毅. ゼロから作るDeep Learning ―Pythonで学ぶディープラーニングの理論と実装. オライリー・ジャパン, 2016, 298p.