1.

我看了游戏的设定，胜reward = 1，负reward = -1.

2.

**经验回放（experience replay）self-play DQN训练算法：**

初始化超参数：状态空间S（3\*8\*8），

动作空间（0-63 表示下棋的点，65: 无棋可下pas）

Discount rate gamma = 0.9, learning rate alpha = 0.001, epsilon = 0.9

初始化两个Q网络Q\_b, Q\_w分别表示black 和white，参数分别为phi\_black, phi\_white

初始化black和white的经验池D\_b,D\_w, 最大容量 都为10000

for i = 1.. max\_iter:

起始状态: S

For t = 0..终止状态

# **黑棋先下：**

黑棋的初始状态S\_b = 上一步白棋下好后的s\_w’

得到黑棋可以下的点，

执行所有将可下的点a，得到的r\_b, s\_b’加入黑棋的经验池D\_b

根据训练好的Q\_b，采用epsilon-greedy选择action，a

执行a，得到r\_b, s\_b’

**# 白棋后下：**

白棋的初始状态S\_w = s\_b’

得到白棋可以下的点，

执行所有将可下的点a，得到的r\_w, s\_w’加入白棋的经验池D\_w

根据训练好的Q\_w，采用epsilon-greedy选择action，a

执行a，得到r\_w, s\_w’

**当到达终止状态（if done）**：

# **对黑棋的Q网络Q\_w的参数phi\_w进行经验回放更新：**

从黑棋的经验池子D\_b中采样ss\_b, aa\_b, rr\_b, ss’\_b (minibatch个)

若ss’\_b为终止状态，则y = reward\_j

若ss’\_b不是终止状态，则y = reward\_j + gamma \* max\_action' Q(state', action')

损失函数为（y – Q\_b(s,a)）^2， 进行参数更新

# **对白棋的Q网络Q\_b的参数phi\_b进行经验回放更新：**

做法同上

3.

CNN网络设置（全连接）：

3\*8\*8 – 2\*8\*8 – 8\*8

4.

DQN self play训练结果：

Iterate 201次：黑棋胜87，白棋胜113（43.5%，57.5%）

多次尝试都是后下的白棋更好

经验回放DQN和minimax alpha-beta剪枝对抗结果：

黑棋（先手minimax）vs白棋（后手DQN）：

黑棋（先手DQN）vs 白棋（后手minimax）：

6.

黑子（先手）DQN vs Random play训练：

设置一层全连接，reward 0 +/-1，iteration200次

黑胜（DQN）：128，白棋（rnd）：62

Test：

Vs. rand

Vs. ab

Vs. self-play(白)