Mastering the Game of Go without Human Knowledge-hoho

研究现状

AlphaGo网络的监督学习需要人类的行为数据。

这里介绍了一种不需要人类行为数据，只需要游戏规则的只依赖于强化学习的算法——AlphaGo Zero。本文的贡献在于验证了可以不依赖人类的领域知识而获得超越人类表现的算法。

研究方法

AlphaGo Zero只需要一个深度网络，该网络输入当前棋局状态s，输出当前状态s下各种走子（包括弃权）的概率分布$p = Pr(a|s)$和状态价值v：$(p, v) = f_\theta(s)$

网络的输入

棋盘定义：围棋19x19个格子组成的集合

每个时间步下，输入为17个特征：

- 1个我方当前的棋盘X_t，当格子是我方的棋子则置为1，若该格子没落子或为对方棋子则置为0
- 我方前7个历史走子的棋盘$X_{t-1}, X_{t-2}, ..., X_{t-7}$
- 1个对方当前棋盘Y_t，当格子是对方的棋子则置为1，若该格子没落子或为对方棋子则置为0
- 对方前7个历史走子棋盘$Y_{t-1}, Y_{t-1}, ..., Y_{t-7}$
- 另外,还有1个C棋盘：当我方走子时，C的所有格子位1，否则为0

综上，输入数据为$s_t = [X_t, Y_t, X_{t-1}, Y_{t-1}, ..., X_{t-7}, Y_{t-7}, C]$

网络架构

主要是以CNN组成的深度残差网络

算法流程

1. Self-play(自我走子/自我对弈)

首先随机初始化神经网络f_θ的参数，然后进行MCTS搜索，寻找合适的走子策略π_θ

MCT树的每个节点代表当前棋局（状态s），边代表当前状态下的走子（a, s）。

每条边包含4个数据：$\{N(s,a),W(s,a),Q(s,a),P(s,a)\}$，分别代表：

$N(s,a)$：访问当前边的次数

$W(s,a)$：总的动作价值

$Q(s,a)$：平均动作价值

$P(s,a)$：选择这条边的先验概率

MCTS搜索有3个过程：

(1) Select

从根节点s_0开始，计算每条边的$Q(s_t,a)+U(s_t,a)$值，选择$a_t=\arg\max_a(Q(s_t,a)+U(s_t,a))$，作为走子动作，其中：

$U(s,a)=c_{puct}P(s,a)\frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}$，$N(s,b)$为s下搜索的总次数，$c_{puct}$为一个常量。

一直走到达叶节点s_L

(2) Expand与Evaluate

由于叶节点下面没边了，需要扩展。使用神经网络输出s_L下的走子策略：$(p,v)=f_\theta(s_L)$，那么该叶节点下就有多个边(s_L,a)了(论文为$(d_i(p),v)=f_\theta(s_L)$，d_i is a

dihedral reflection or rotation selected uniformly at random from $i \in [1..8]$，这里为了简便先这么写)

初始化每条边为$\{N(s,a)=0, W(s,a)=0, Q(s,a)=0, P(s,a)=p\}$

另外计算P(s,a)时，实际会加上一个噪声(Dirichlet noise)增加尝试所有动作的概率：$P(s,a)=(1-\epsilon)p+\epsilon\eta$，其中$\eta=0.25$

(3) Backup

回溯s_L叶节点到根s_0的路径（边），进行如下更新：

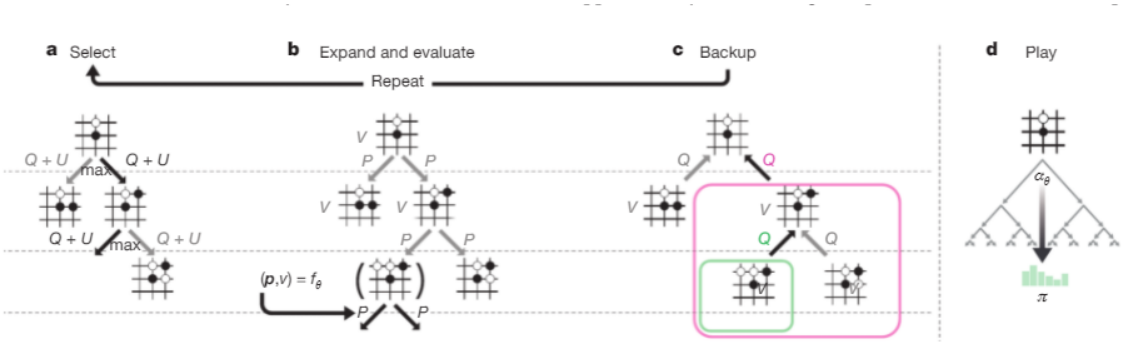
$N(s_t,a_t)=N(s_t,a_t)+1$

$W(s_t,a_t)=W(s_t,a_t)+v$

$Q(s_t,a_t)=\frac{W(s_t,a_t)}{N(s_t,a_t)}$

当搜索完后，MCTS就会产生一个当前状态s下走子策略$\pi(a|s)=\frac{N(s,a)^{\frac{1}{\tau}}}{\sum_b N(s,b)^{\frac{1}{\tau}}}$，其中$\tau$是温度系数，在模拟走子的前30步，$\tau=1$以鼓励探索，往后的模拟走子设置$\tau \to 0$。

整个MTCS过程如下图：



这样每次在真实棋盘下子（Play）之前，算法都会进行1600次的模拟走子，这样每个时间步就会形成一个个样本$\{s_t,\pi_t,z_t\}$，并放到经验回放池，当双方都弃权，或者搜索价值下降到一个阈值，或者时间步超过最大的长度，对战结束，当最终赢了$z_t=+1$，输了$z_t=-1$(hoho: 其余为0？)。

真正进行下子动作的子节点成为新的根节点，这个节点及其下节点的统计数据会保留，然后树的其余部分会删掉。算法也会指定一个阈值v_{resign}，当算出来的根节点的价值和最优子节点的价值低于其时会resign掉（hoho: 啥为resign?）

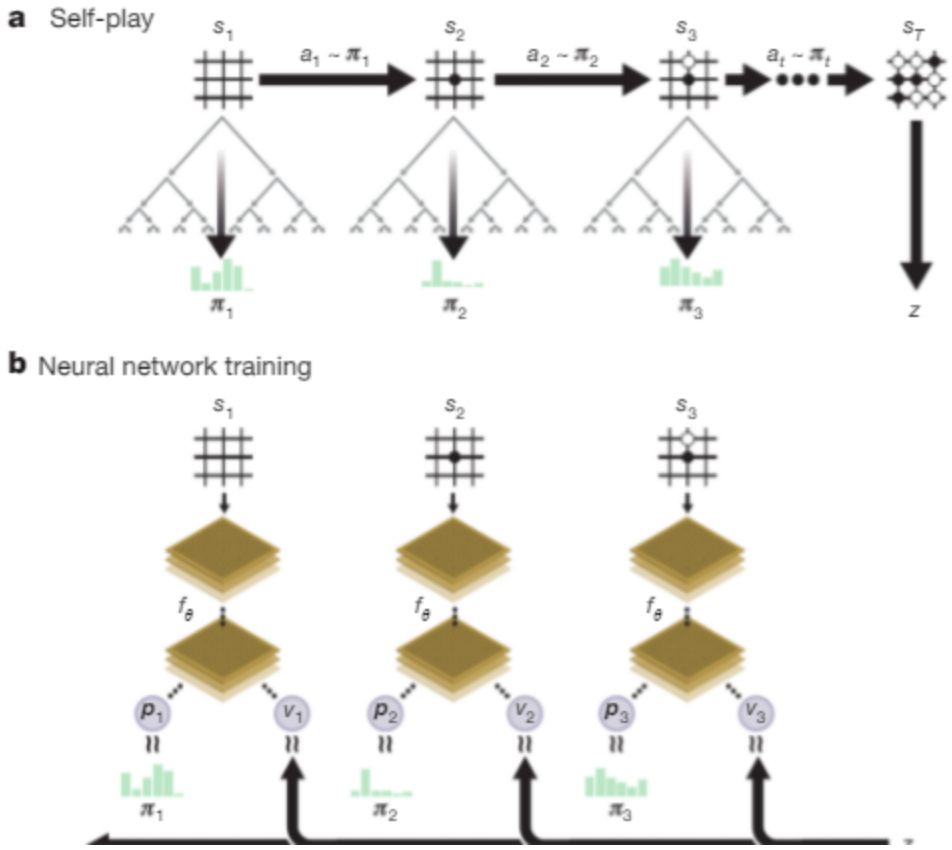
2. 网络更新

当前网络从经验回放池中均匀采样出多个样本$\{s, \pi, z\}$进行训练，损失函数为：

$loss = (z - v)^2 - \pi^T logp + c||\theta||^2$，其中$(p, v) = f_\theta(s)$

训练得到的新网络f_θ^*，会跟当前网络f_θ进行对弈，如果胜率超过55%（wins by a margin of > 55%）就让其成为当前新的网络，继续下一步的Self-Play

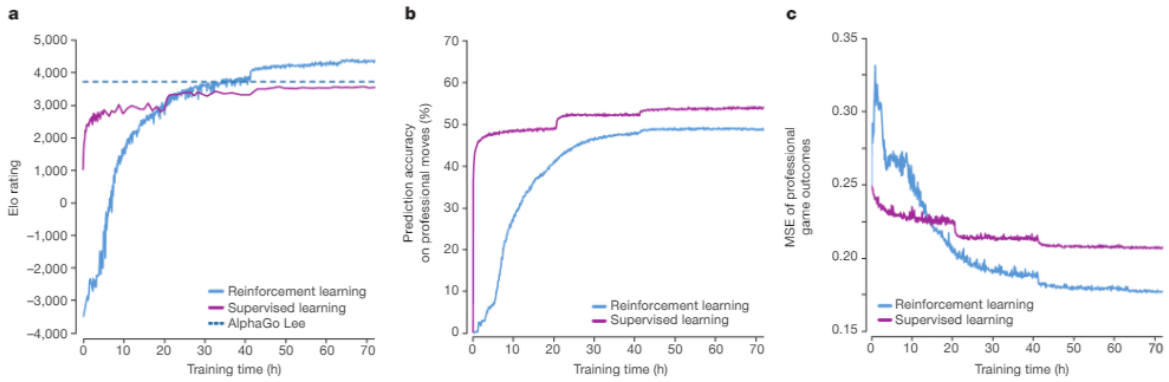
总的流程图如下:



论文中算法会进行大约25000场的Self-play。

研究结论

论文也用同样架构的神经网络通过人类专家数据进行监督训练，并且和AlphaGo Lee一起，跟AlphaGo Zero进行对比，如下:



最开始，用人类专家经验进行监督学习的网络表现较好预测准确度比AlphaGo Zero高，但最终还是被AlphaGo Zero打败。

而对比当初的AlphaGo Lee，AlphaGo Zero将策略网络跟价值网络合并为一个单一的神经网络，虽然稍微降低了预测准确度，但是却减少了价值误差，而且表现飞一般！！！

启发

算法的成功，MCTS作用很大，归功于使用它不断进行策略迭代：

1. MCTS搜索基于网络推荐的策略进行下子，得到的策略又返还给网络进行逼近，从而进行策略改进。
2. Self-play的输出结果进行策略评估，然后又用来逼近网络输出的价值。

附

（暂无）