ICLE

doi:1

ring the game of Go without n knowledge

ian Schrittwieser[1]*, Karen Simonyan[1]*, Ioannis Antonoglou[1], Aja Huang[1], Arthur Guez[1], Lucas Baker[1], Matthew Lai[1], Adrian Bolton[1], Yutian Chen[1], Timothy Lillicrap[1], Fan Hui[1], Laurei riessche[1], Thore Graepel[1] & Demis Hassabis[1]

【强化学习算法 29】AlphaGo Zero

张楚珩
清华大学 交叉信息院博士在读

过去整一年了，来讲讲AlphaGo Zero的具体算法细节吧。

原文传送门

Silver, David, et al. "Mastering the game of Go without human knowledge."Nature550.7676 (2017): 354.

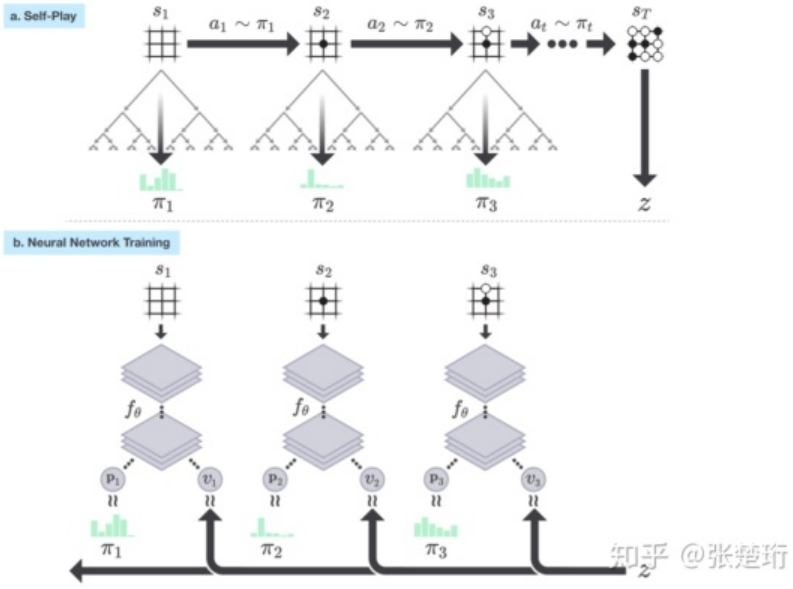
特色

AlphaGo Zero不利用任何的人类经验，完全靠算法自己探索，并且使用了更为优雅而简单的构架来训练，不仅击败了曾经打败过李世乭的AlphaGo Lee，而且打败了在专业在线对战平台上取得60-0胜利的AlphaGo Master。

过程

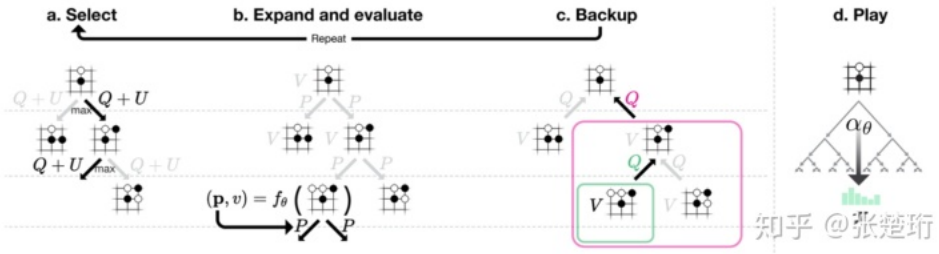
1. 整体构架

整体上来说AlphaGo Zero完全在自我对弈中来学习，如下图所示。



- 该算法一个重要的组成部分是一个把policy network和value network融合在一起的神经网络。神经网络的输入为当前的局面 s，输出为下一步行动的概率 p 和对于当前局面胜率的估计 v，即 $(p, v) = f_\theta(s)$。该神经网络的训练目标是去拟合自我对弈里面产生的真实胜率 z 和下面提到的MCTS产生的落子概率 π，即 $(p, v) \to (\pi, z)$。
- 该算法另一个重要的组成部分是蒙特卡洛树搜索方法（MCTS）。它被使用在自我对弈和最后实际使用的落子选择上。该方法的输入为当前的局面 s 和当前的神经网络 $f_\theta(\cdot)$，输出为落子的概率分布 π。

2. 蒙特卡洛树搜索



搜索树的每个节点代表某个状态下采取了某个动作 (s,a) ，存放以下四元组的统计信息

$$\{N(s,a), W(s,a), Q(s,a), P(s,a)\},$$

分别是节点访问次数，总的action-value，平均action-value和做出该选择的先验概率。下面我们分别来讲一下select、expand and evaluate、backup、play这些MCTS的元素在AlphaGo Zero里面是如何做的。

Select

在MCTS里面每一次搜索都会选择一个策略树上的节点来进行，节点的选择一般要平衡探索和利用，即要更多地对于不同的节点进行探索，也要尽量把更多精力花费在比较好的节点上。这里的探索规则是选择 $\arg\max_a Q(s,a)+U(s,a)$ ，其中 $Q(s,a)$ 是已知的平均表现， $U(s,a)$ 表示在未来的探索里面其可能的比现在的平均表现好多少。这里使用了PUCT algorithm，即

$$U(s,a) = c_{\text{puct}} P(s,a) \frac{\sqrt{\sum_b N(s,b)}}{1+N(s,a)}$$

Expand

最开始的时候搜索树只有一个根节点代表当前的状态，当搜索树选择了一个之前没有选择过的动作的时候，这时候就需要拓展一个新节点。新节点会被分配一个初始化的节点统计信息

$$\{N(s_L,a)=0, W(s_L,a)=0, Q(s_L,a)=0, P(s_L,a)=p_a\}$$

其中 $p_a, \_ = f_\theta(s_L)$ 是神经网络输出的落子概率。

Evaluate

搜索树的节点是指数级增长的，因此肯定不可能无限扩展，到达一定深度之后就不会再扩展了。普通的做法就是从这些不再扩展的叶子节点出发，使用蒙特卡洛采样获得多个一直运行到游戏结束的轨迹（rollout），然后把这些rollout的平均成绩作为该节点的得分 $Q(s,a)$ 。这里的做法不同于普通的MCTS，这里直接使用了神经网络的输出 $\_, v = f_\theta(s_L)$ 来作为节点的得分。这种做法避免了复杂而耗时的rollout。

一个细节，考虑到棋局的旋转对称性和镜面对称性，做这些对称变换之后应该不影响节点的得分的，因此这里估计的时候使用了任意一种对称变换 d_i 之后得到的评分 $\_, v = f_\theta(d_i(s_L))$ 。

Backup

叶子节点上面得到的数值需要传到前面的父辈节点上，假设某个搜索轨迹的叶子节点得到评分 v ，那么这条轨迹上的每个节点都进行这样的更新

$$N(s_t, a_t) = N(s_t, a_t) + 1$$
$$W(s_t, a_t) = W(s_t, a_t) + v$$
$$Q(s_t, a_t) = \frac{W(s_t, a_t)}{N(s_t, a_t)}$$

Play

MCTS最后的目的是输出对于根节点来说采取不同行动的概率 $\pi(a|s_0)$ ，这个概率被规定为

$$\pi(a|s_0) = N(s_0, a)^{1/\tau} / \sum_b N(s_0, b)^{1/\tau}$$

其中 τ 是一个温度参数，注意到，当 $\tau \to 0$ 的时候，它就变成了 $argmax$ 。

其他优化

- 为了增加对于开局的探索，在游戏的前30步，设置 $\tau = 1$ ；在之后的行动里面才设置 $\tau \to 0$
- 同样为了增加探索，每个节点初始化的概率设置为

$$P(s, a) = (1 - \epsilon) p_a + \epsilon \eta_a, \text{ where } \boldsymbol{\eta} \sim \text{Dir}(0.03) \text{ and } \epsilon = 0.25$$

- 为了搜索的效率，当某个节点被拓展出来的时候其得分评估低于某个阈值 v_{resign} 的时候就会放弃继续探索这个节点，而这个阈值的设定是保证因为丢弃这个节点而错过胜局的概率不超过5%。

3. 策略和价值神经网络

神经网络的输入是当前局面状态，输出是行动概率和从当前状态出发的胜率，即 $(p, v) = f_\theta(s)$ 。

输入状态的表示

当前状态的表示是 $19 \times 19 \times 17$ 的二值特征表示

$$s_t = [X_t, Y_t, X_{t-1}, Y_{t-1}, ..., X_{t-7}, Y_{t-7}, C]$$

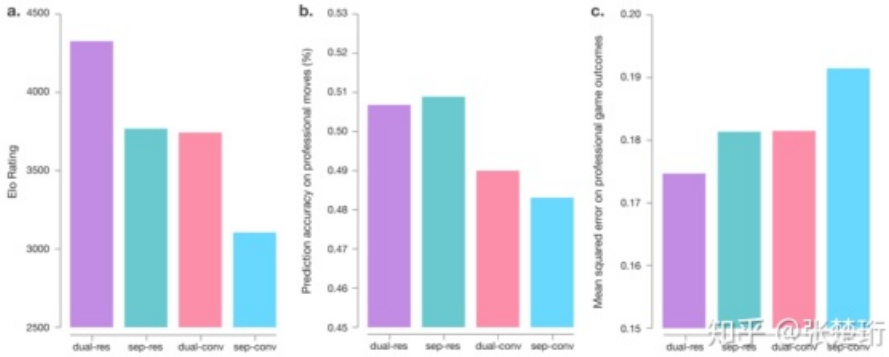
其中 X 表示本方的棋子， Y 表示对方的棋子，由于围棋不能反复征子，所以必须要联合过去的几步才能完全判断当前的局面； C 表示我方棋子的颜色，这是为了表示贴目的信息。

神经网络结构

神经网络的构建参考了图像里面比较先进的做法，使用了共用的19（或39）个residual block，在末尾分成了策略网络和价值网络，它们各使用了一个residual block。总的网络深度为20（或40），总的含参网络层数为39（或79）。

网络结构比较

这里的神经网络主要有两个设计：一是两头输出的神经网络（dual），与之对应的是训练两个神经网络（sep）；二是使用了残差网络（res），与之对应的是使用普通的CNN（conv）。文章对于这些可能的组合做了对比，结果显示文章的这种组合（duel-res）在围棋水平（Elo rating，a图）和对于局面的预测（c图）上也最好，虽然可能在对于人类棋手落子的预测上稍差一些（b图）。



神经网络的训练

神经网络的每一个训练数据形如 (s_t, π_t, z_t) ，其中 s_t 是每一步上棋局的表示， π_t 是MCTS搜索产生的行动概率， $z_t = t_T \in \{-1, +1\}$ 是这一步所在棋局本方的获胜情况。网络训练使用如下损失函数

$$(\mathbf{p}, v) = f_\theta(s), \qquad l = (z - v)^2 - \boldsymbol{\pi}^\top \log \mathbf{p} + c||\theta||^2$$

我们从后面可以看到，这个模型训练了40天，因此也不太可能训练多个模型来找个最好的，如何保证训练的过程中不会“走歪”了呢？他们使用了如下的技巧。

他们在每1000个训练步之后都设置一个checkpoint，把当前的策略 f_{θ_i} 和截至目前位置最优的策略 f_{θ_*} 进行对战。如果新的策略能够以高于55%的胜率战胜之前的最优策略，那么新策略就会被接受，并且取代之前的最优策略；反之，抛弃这1000步的训练，回到之前的最优策略上。

结果

AlphaGo Zero学习了2900万局游戏，使用4块TPU训练了40天，在于AlphaGo Mater的对战中取得了89:11的胜率。

AlphaGo Zero用到了哪些围棋领域的知识？

AlphaGo Zero基本上没有用到人工总结的经验，基本上算作**零和、完全信息**博弈中比较通用的解决方案。仅用到了以下知识

- 所有的围棋规则：这些规则用来进行胜局的判定，并且提供每一步合法的落子位置；
- 围棋的几何构型：由于围棋刚好是 19×19 的格点，并且具有某种程度上的空间局域性和平移不变性，因此使用CNN的结构能够刚好完美处理这样的几何结构。在迁移到其他的游戏上的时候需要着重考虑一下这一点区别。
- 游戏的对称性：游戏的棋局具有旋转和镜像的对称性，在不考虑贴目的情况下，敌我双方也具有对称性，算法也利用了这一个特点。

AlphaGo Zero与之前版本AlphaGo的对比。

- AlphaGo Fan：使用了176个GPU进行训练，其价值函数的目标是直接使用policy network对战得来的。在2015年10月以5:0战胜樊麾。
- AlphaGo Lee：使用了更大的网络结构，并且开始使用被MCTS增强过的策略进行自我对战，使用了48块TPU进行训练。在2016年3月以4:1战胜李世乭。
- AlphaGo Master：使用和Zero类似的网络结构和算法，不过和前面的算法一样，使用了人工设计的特征，并且从人类数据的监督学习中类初始化神经网络。在2017年1月在专业的在线对战平台上取得60:0的胜率。
- AlphaGo Zero：完全没有使用人类经验，使用了4块TPU进行训练。以89:11战胜AlphaGo Master，以100:0战胜AlphaGo Lee。

发布于 2018-11-22

强化学习 (Reinforcement Learning)

文章被以下专栏收录

强化学习前沿
读呀读paper

进入专栏