第 3 卷◆第 1 期◆版本 1.0◆2025 年
文章类型：论文|刊号（ISSN）：2972-4236(P) / 2972-4244(O)

Gomoku 的人工智能算法设计及实现

刘畅　邱力军
西京学院机械工程学院
DOI:10.12238/acair.v3i1.11924

[摘　要] 本文集中于对五子棋(Gomoku)AI的运算法则的研究与设计,传统的五子棋AI算法主要是基于搜索和规则的方法,这种方法虽然能够在一定程度上解决问题,但在实际应用中存在着效率低下、易受局限性和难以扩展等问题。近年来,强化学习技术的发展为五子棋AI的研究和应用提供了新的思路和方法,本文设计了一种基于强化学习的五子棋AI系统,该系统使用卷积神经网络和蒙特卡洛树搜索进行训练和预测。具体地,该系统使用双方棋子分布作为输入,预测下一步落子位置的得分,并根据得分进行决策。该系统可以从大量的数据中学习棋局规律,并具有较强的泛化能力,能够应对复杂的棋局和不同的对手。
[关键词] 五子棋；人工智能；卷积神经网络；蒙特卡洛树搜索算法
中图分类号：TP18　**文献标识码**：A

Gomoku's Artificial Intelligence Algorithm Design and Implementation

Chang Liu　Lijun Qiu
School of Mechanical Engineering, Xijing University

[Abstract] This paper focuses on the research and design of Gomoku AI algorithm,the traditional Gomoku AI algorithm is mainly based on search and rule method,although this method can solve the problem to a certain extent, but in the practical application there are inefficient, vulnerable to limitations and difficult to expand and other problems. In recent years, the development of reinforcement learning technology has provided new ideas and methods for the research and application of Gomoku AI. This paper designs a Gomoku Al model based on reinforcement learning, which uses Convolution Neural Network and Monte Carlo Tree Search for training and prediction. Specifically, the model uses the distribution of pieces on both sides as input to predict the score of the next drop position and make decisions based on the score. The system can learn the rules of chess games from a large amount of data, and has a strong generalization ability, can deal with complex chess games and different opponents.
[Key words] Gomoku, Artificial intelligence; Convolution Neural Network; Monte Carlo Tree Search algorith

1 绪论

随着智能化技术在近年来不断发展,人工智能逐渐成为许多行业中十分热门的研究领域。人工智能在许多领域都取得了令人瞩目的成果,例如语音识别领域、自然语言处理以及计算机视觉领域等[1]。感知领域的发展给人们带来便利,例如：指纹解锁、人脸识别等。人工智能领域的公司在国内已经达到数百家。许多国家将人工智能设立为重点研究项目。在人工智能的研究中,神经网络作为一个分支,对整个人工智能领域的发展影响十分深远。目前,围棋与象棋作为全球最流行的两种棋类项目,已经各自诞生了十分优秀的AI系统。该系统也直接改变了整个棋界,从AI需要向人类学习发展到了人类需要向AI学习的地步,这无疑对整个人工智能的研究都有着里程碑的意义。

2 蒙特卡洛模型的构建

2.1定义棋盘及棋盘可视化

对于五子棋游戏,对弈双方都拥有着针对局面的全部信息,这使得五子棋AI的可视化尤为重要,故此系统实现了一个可视化的交互界面,及一整个棋盘界面,使得对弈双方可以完全了解整个棋局：

```
self.width = int(kwargs.get('width', 15)//定义宽度。
self.height=int(kwargs.get('height',15)//定义棋盘高度。
```

2.2蒙特卡洛搜索算法的构建

蒙特卡洛树搜索算法的总体流程是反复进行相同的一套动作。每个动作包括4个步骤[2]：

(1)选择(Selection)：按照一定的方式遍历搜索树,并基于当前的情况决定后续操作。

(2)扩展(Expansion)：为当前节点增加一个子节点,同时选择一个动作到达对应的新状态。

(3)模拟(Simulation)：为新增加的节点计算价值的初试估计值。

(4)反向传播(Back propagation)将搜索树的子节点信息,传播到上层的根节点,并使其再次选择。通过self._Q=0定义了节点在MCTS算法中评估的得分,即价值评估得分。初值为0,训练后会根据公式(2-1)进行更新。

$$Q(s,a)=\frac{W(s,a)}{N(s,a)} \tag{2-1}$$

其中,s是当前的棋盘状态,a是当前的落子动作。W(s,a)是选择当前节点的累积行动价值,N(s,a)是当前节点的总访问次数,两者的更新规则参考公式(2-4)与(2-5)。

最大置信上界算法(Upper Confidence Bound Apply to Tree,UCT)是以蒙特卡洛树搜索算法为核心[3],将蒙特卡洛搜索算法与一个U函数相结合的算法。相较于其他算法可以很大程度上的提高博弈树的搜索效率,从而进一步降低空间复杂率。核心公式如公式(2-2)所示。

UCT=Q(s,a)+U (2-2)

通过self._u=0定义公式(2-2)中的U函数。

$$U=cP_t\sqrt{\frac{N}{N_t+1}} \tag{2-3}$$

其中,c是比重系数,控制U函数在公式(2-3)中的比重,Pt是模型落子前的先验概率,由策略价值网络给出,N是父节点的总访问次数,N_t是子节点的总访问次数。蒙特卡洛树搜索模型在遍历过程中会优先选择UCT值最大的节点进行落子。

2.3搜索算法中扩展与模拟步骤的搭建

对扩展出的新节点进行模拟后,该节点的Q值,U值将会更新,并同时更新此节点的父节点。具体更新方式可用公式(2-4)、公式(2-5)、公式(2-6)表示。

Nt(s,a)=Nt(s,a)+1 (2-4)

W(s,a)=W(s,a)+vt (2-5)

$$Q(s,a)=\frac{W(s,a)}{N_t(s,a)} \tag{2-6}$$

其中,其中vt为卷积神经网络输出的评估得分,具体公式可参考公式(3-4)。从公式(2-6)可见,随着模拟次数的增加,动作平均值Q(s,a)会逐渐趋于稳定。

3 五子棋AI系统的实现

3.1通用网络层的构建

(1)第一层卷积层(conv1)：使用32个3×3的卷积核,步长为1,padding方式是same,激活函数使用ReLU,15×15×4的局面维度处理为15×15×32的特征图。公式(3-1)是第一层卷积层的计算公式。

$Z1=ReLU(W_1*X_0+b1)$ (3-1)

其中W1和b1分别表示卷积核权重和偏置项,f表示激活函数,这里使用ReLU函数如公式(3-2)所示。

ReLU(X)=max(0,X) (3-2)

(2)第二层卷积层(conv2)：使用64个3×3的卷积核,步长为1,padding方式是same,激活函数使用ReLU,将15×15×32的局面维度处理为15×15×64的特征图。所用公式与代码描述与第一层卷积层类似,这里不再赘述。

(3)第三层卷积层(conv3)：使用128个3×3的卷积核,步长为1,padding方式是same,激活函数使用ReLU。将15×15×64的局面维度处理为15×15×128的特征图[4]。

3.2动作网络的构建

动作网络的输入来自第三层卷积层(conv3)的输出,构建的第一步是使用4个1×1的卷积核,步长为1,padding方式是same,激活函数使用ReLU,将此输出为15×15×4的TensorFlow特征张量。第二步是将拉平后的TensorFlow张量连接到一个units为225的全连接层(action_fc),使用softmax函数,输出棋盘上每个位置移动的对数概率,即先验概率。此概率影响五子棋AI的最终落子位置,softmax函数如公式(3-3)所示。

$$soft\max(x)=\frac{e^{xi}}{\sum_{i=1}^{n}e^{xi}} \tag{3-3}$$

其中n表示输出层共有n个神经元,xi表示第i个神经元。

3.3评估网络的构建

评估网络的输入同样来自第三层卷积层(conv3)的输出,即15×15×4的特征图,构建过程如下：

(1)使用2个1x1的卷积核和ReLU函数将15×15×4的特征图转换成15×15×2的特征图。

(2)将15×15×2的特征图拉平成一个大小为450的TensorFlow张量。

(3)通过一个包含64个神经元的全连接层(evaluation_fc1),使用ReLU函数来提取更高层次的抽象特征,最终使用公式(3-4)的tanh激活函数,输出当前状态的评估分数(evaluation_fc2),此评估分数代表当前棋局的胜率值,故其范围在[-1,1]之间[5]。

$$\tanh(x)=\frac{1-e^{-2x}}{1+e^{-2x}} \tag{3-4}$$

3.4定义优化器与损失函数

(1)本次构建的卷积神经网络使用Adam优化器[6],其更新规则为：

gt=▽θL(θt-1) (3-5)

mt=β1mt-1+(1-β1)gt (3-6)

$vt=\beta 2vt-1+(1-\beta 2)gt^2$ (3-7)

文章类型：论文|刊号（ISSN）：2972-4236(P) / 2972-4244(O)

$$\hat{m}_1 = \frac{m_1}{1-\beta_q^{\ t}} \tag{3-8}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^{\ t}} \tag{3-9}$$

$$\theta_t = \theta_{t-1} - a\frac{\hat{m}_t}{\sqrt{\hat{v}_t + \varepsilon}} \tag{3-10}$$

其中gt表示当前时间步的梯度,mt和vt分别表示梯度的一阶和二阶矩估计,t和t表示对mt和vt进行偏差校正后的估计值,θ表示模型参数,∂表示学习率,β1和β2分别表示梯度的一阶和二阶矩估计的衰减率,ε表示一个极小常数。

(2)损失函数公式如下:

$$\text{loss} = (z-v)^2 - \pi^{\mathrm{T}}\log(p_t) + c|\theta|^2 \tag{3-11}$$

整个公式由三部分组成:第一项是胜率值误差,第二项是走子概率的交叉熵,第三项是一个正则化参数[7]。

3.5搭建基础框架

首先设定自我训练参数,具体描述如下:

(1)需要一个容量固定的样本池来存放棋谱,同时要保持棋谱总是最新的,即新棋谱进入,旧的棋谱就要被替换,棋谱池容量定义为buffer_size。

(2)策略价值网络间隔多久(每生成多少棋谱后)就开始新一轮的训练,然后更新整个模型,定义 pure_mcts_playout_num 值为棋谱数。

(3)在棋谱的生成过程中,每步棋的蒙特卡洛树搜索模拟次数需要确定。理论上,模拟次数越多,棋谱质量越高,定义n_playoute值为单次落子的模拟次数[8]。但本文需要考虑算力,确定一个平衡点,故本文设定的参数如表所示。

表3-1 自我训练参数设置表

参数定义	参数值
棋谱池容量 buffer_size 每生成C个新棋谱就训练网络并更新一次 蒙特卡洛模型单次落子模拟次数 n_playout	buffer_size=10000 pure_mcts_playout_num= 1000 n_playoute=400

之后,通过self.policy_value_net=PolicyValueNet(self.board_width,self.board_height)调用策略神经网络进行训练。

4 系统调试与性能测试分析

4.1 AI棋力测试

此测试同样通过与五子棋原型系统进行200次对战,以胜率表示各个测试组中AI的棋力,具体参数设置参考表4-1。实验测试结果如表4-1所示。

表4-1 棋力对比结果

测试组	训练次数	胜利次数	获胜机率	平均反应时间
1	300	7	3.5%	1. 1s
2	500	87	43.5%	1.5s
3	1000	136	68%	1.7s
4	3000	179	79.5%	1.8s

由表4-1可以看出训练次数与五子棋AI系统的棋力呈现明显的正相关性,并在训练次数较多时所对应的平均反应时间也会变长,说明了蒙特卡洛搜索算法和卷积神经网络的成功实现。

为解释训练次数在300次到500次的过程中棋力出现了显著提高的现象,这里对学习率进行了监测,看出当前模型在训练次数不足400时学习率很低,此时各个棋盘节点没有完全扩展,无法产生有效的棋盘数据,几乎无法胜利。为改善这一问题,在这里使用基于KL的自适应调节学习率,用于改善训练次数不足时,学习率严重低下的问题。引入后该问题已得到明显改善,监测结果如图4-1所示。

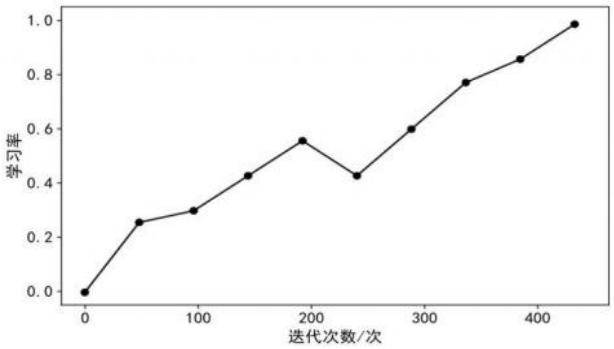

图4-1 改善后学习率曲线

相较于改善前的学习率曲线,图4-1中的学习率曲线更加具有正相关性,这表明该模型在改善之后解决了学习率不足的问题。

4.2策略价值网络表现

损失函数曲线图用以检测卷积神经网络的具体表现,如图4-2所示。

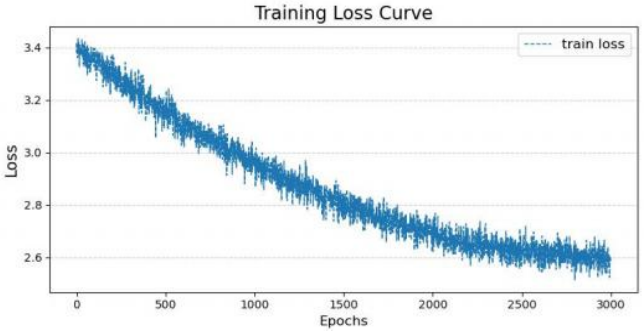

图4-2 训练损失曲线图

从图4-2可以看出该模型随着训练次数的增加,损失值会明显减小,这证明了卷积神经网络对MCTS模拟时提供的选择策略起到了至关重要的作用,而训练次数也直接影响了MCTS的搜索精度。

4.3系统完善

4.3.1增加探索性

在自我对弈中,为了提高五子棋AI的强度和泛化能力,需要增加一定的探索性。具体地,在选择落子位置时,采用了贪心策略,即在一定概率下随机选择一个可行落子位置,以促进模型的探索和创新[9]。

4.3.2引入温度参数

在自我对弈的过程中,引入温度参数,从而使模型在选择落子位置时更具有多样性。具体地,温度参数可以通过对模型的预测结果进行指数运算来计算,如公式(4-1)所示。

$$p_i = \frac{\exp(z_i / \tau)}{\sum_{j=1}^{n} \exp(z_i / \tau)} \tag{4-1}$$

其中Z_i表示模型对于第i落子位置的预测得分,n表示可行落子位置的数量。当温度参数τ较高时,所有落子位置的概率将趋近于相等,模型更容易进行探索；当温度参数τ较低时,模型更倾向于选择得分最高的落子位置,以保证最终胜利的概率[10]。

5 结语

五子棋是一种源自中国的古老棋类,在中国流传到世界之后,其名称和游戏方式都有了很大的改变。目前已存在的大多数五子棋AI系统并不完善,存在许多缺点,其根本原因是没有使用强化学习作为五子棋AI的训练方式。本次设计的基于强化学习的五子棋AI系统则很好的解决了这一问题。该系统使用了蒙特卡洛树搜索算法与卷积神经网络,通过自我对弈产生训练数据,最终通过大量的数据不断更新,最终逐渐提高棋力,完成对整个蒙特卡洛搜索算法的实现。

[参考文献]

[1]王亚杰,邱虹坤,吴燕燕,等.计算机博弈的研究与发展[J].智能系统学报,2017,11(06):789-796.

[2]徐梓荐.蒙特卡洛法在城市污水PPP项目投资风险分析的应用[D].兰州:兰州大学,2018.

[3]Jacoboni C,Lugli P.The Monte Carlo method for semiconductor device simulation[M]. Springer Science & Business Media, 2012.

[4]汪坤兵.六子棋博弈中搜索技术的研究与实现[D].合肥:安徽大学,2016.

[5]李学俊,王小龙,吴蕾.六子棋中基于局部“路”扫描方式的博弈树生成算法[J].智能系统学报,2015,3(2):265-272.

[6]Rubinstein R Y, Kroese D P. Simulation and the Monte Carlo method[M]. John Wiley & Sons,2016.

[7]高友文.基于卷积神经网络的苹果表面伤疤识别研究[D].南京:南京邮电大学,2018.

[8]张璐璐.基于卷积神经网络的人脸面部表情识别方法研究[D].石家庄:河北科技大学,2018.

[9]李玉.基于卷积神经网络的射线图像识别系统的研究[D].北京:北化航天工业学院,2019.

[10]David Silver,Aja Huang.Mastering the game of Go with deep neural networks and tree search[J].Nature,2016, 14(3): 35-68.

作者简介：

刘畅(2000--),男,汉族,内蒙古自治区乌兰察布市凉城县人,全日制本科,研究方向：机械系统设计与状态检测。