重庆理工大学学报(自然科学)
Journal of Chongqing University of Technology(Natural Science)
2022 年 第 36 卷 第 12 期
Vol. 36 No. 12 2022
doi: 10.3969/j.issn.1674-8425(z).2022.12.014

“机器博弈”专栏(专栏主编:张小川 重庆理工大学 教授)

# “拱猪”游戏的深度蒙特卡洛博弈算法

吴立成,吴启飞,钟宏鸣,李霞丽

(中央民族大学 信息工程学院, 北京 100081)

**摘 要**:针对现有的“拱猪”卷积模型计算复杂且高度依赖专家知识的问题,提出一种应用于“拱猪”博弈游戏的深度神经网络和蒙特卡洛方法相结合的深度蒙特卡洛算法。采用自对弈的方式进行模拟和评估,使用深度 Q 网络代替 Q 表完成 Q 值的更新,高效地对“拱猪”策略进行探索和利用;采用分布式并行计算的方法提高训练效率,较于传统的蒙特卡洛方法可有效地解决高方差问题。在具有一个 GPU 的单台服务器上训练 24 h 后,所构建的智能代理与“拱猪”卷积模型对弈了 10 000 局。实验结果表明:智能代理胜率可达 78.3%,平均每局可获得 67 分,对具体示例进行分析,进一步验证了该算法的有效性以及智能代理的良好性能。

**关 键 词**:人工智能;拱猪;深度强化学习;蒙特卡洛方法

**中图分类号**:TP183 **文献标识码**:A **文章编号**:1674-8425(2022)12-0121-08

## 0 引言

1997 年,“深蓝”[1] 在国际象棋游戏上以 3.5∶2.5的比分击败了职业选手卡斯帕罗夫;2016 年开始,Google 旗下 DeepMind 公司研制的 AlphaGo[2]与 AlphaZero[3]在围棋游戏上取得了非常优秀的成绩。近年来,机器博弈在非完全信息领域逐渐成为热门研究之一。2015 年,CFR + 算法的提出为解决有限注德州扑克[4-6]游戏中非完全信息博弈提供了新思路[7],不足的是该算法只能解决小型的非完全信息博弈问题;2018 年,由 Brown 等[8]提出的 CFR 算法变体可解决大型非完全信息博弈问题,并在大型扑克游戏中取得了强大的表现;2021 年,张小川等[9]针对提高不同对手的评估效益的问题,提出了一种基于对手牌力的评估方法,实验表明,其所构建的德州扑克智能体能够打败不同风格的对手,总体收益较静态评估方法有所提高;2020 年,由 Li 等[10]开发的麻将人工智能模型 Suphx 在 Tenhou 平台上超过了 99.99% 的玩家水平;2021 年,Zha 等[11]开发的基于深度神经网络的“斗地主”人工智能 DouZero 在 Botzone 的 344 个“斗地主”人工智能模型中取得了第一名

**收稿日期**:2022-09-05
**基金项目**:国家自然科学基金项目(62276285)
**作者简介**:吴立成,男,博士,教授,主要从事智能系统及机器人、计算机博弈研究,E-mail:wulicheng@tsinghua.edu.cn;通讯作者 李霞丽,女,教授,主要从事计算机博弈研究,E-mail:xiaer_li@163.com。

**本文引用格式**:吴立成,吴启飞,钟宏鸣,等.“拱猪”游戏的深度蒙特卡洛博弈算法[J].重庆理工大学学报(自然科学),2022,36(12):121-128.

**Citation format**:WU Licheng, WU Qifei, ZHONG Hongming, et al. Deep Monte Carlo algorithm for Gongzhu game[J]. Journal of Chongqing University of Technology(Natural Science),2022,36(12):121-128.

的成绩。除了传统棋牌类游戏,人工智能在电子游戏上的发展也令人惊叹,由 DeepMind 开发的 AlphaStar[11] 和 OpenAI 开发的 OpenAI Five[12] 分别在星际争霸和 Dota 2 中达到了专业玩家水平;2020 年,Ye 等[13] 开发的“绝悟”人工智能模型击败了顶尖的王者荣耀职业玩家。

现有的“拱猪”人工智能模型较少,且主要采用监督学习方法,计算复杂且高度依赖人类知识。“拱猪”具有 2 个特点。第一:游戏具有极大的反转性,参与游戏的玩家将在仅了解游戏部分信息的条件下进行竞争与合作,竞争和合作的选择决定着最后的游戏结果;第二:游戏玩家与扑克牌可构成巨大的状态空间,同时也拥有多达 52 种动作空间,这使得状态空间的搜索与计算变得非常困难。以上 2 个特点决定了常见的深度学习方法在“拱猪”人工智能模型的构建上很难起到立竿见影的效果。

本文将在“斗地主”中已成功应用的深度蒙特卡洛算法[11] 移用于“拱猪”游戏中,开发了不依赖人类知识、易于计算、训练效率高的“拱猪”模型。该模型并不会搜索所有的状态空间,而是在自对弈过程中进行逐步探索;基于并行处理的方式可生成更多的训练数据。该模型具有 3 个优点,第一:它不依赖人类专家知识,可以自然地模拟人类行为并进行评估学习,且比传统的深度学习需要更小的计算量;第二:依赖于现代计算机的强大计算能力,模型在训练过程中可以轻易地覆盖人类玩家不常见的状态与动作并进行训练与学习;第三:在大量的自对弈过程中,偶然错误的动作及其评估不会对模型的最终效果产生严重的影响。这 3 个优点使得该模型在训练过程中能够处理“拱猪”巨大且复杂的状态空间,这对于“拱猪”人工智能模型的构建是至关重要的。

本文构建的模型在具有一个 GPU 的单台服务器上训练 24 h 后,表现出来的性能要优于基于卷积神经网络的“拱猪”人工智能模型(以下简称卷积模型)。

## 1 “拱猪”游戏规则和数据格式

### 1.1 “拱猪”游戏规则

“拱猪”是一款风靡于全国及海外华人地区的传统扑克牌游戏,规则简单却易懂难精,由一副去除大小王的 52 张扑克牌组成,参与游戏的玩家共 4 名,每位玩家随机得到 13 张牌。在“拱猪”的游戏规则中,扑克牌被分为两类:有分值的牌和无分值的牌,表 1 展示了“拱猪”中牌与分值的对应关系。

表 1 “拱猪”中牌与分值的对应关系

| 牌 | 黑桃 Q | 方块 J | 红桃 2 ~ 4 | 红桃 5 ~ 10 |
|---|---|---|---|---|
| 分值 | -100 | +100 | 0 | -10 |
| 牌 | 红桃 J | 红桃 Q | 红桃 K | 红桃 A |
| 分值 | -20 | -30 | -40 | -50 |

除此之外,赢得梅花 10 的玩家手里的分牌分值将变为原本的 2 倍,若手里无其他分牌,则玩家获得 +50 分。

“拱猪”分为亮牌阶段和出牌阶段。在亮牌阶段,玩家可以将初始手牌中的黑桃 Q、方块 J、梅花 10 以及红桃 A 亮出,被亮出的牌分值将变为原来的 2 倍,若梅花 10 被亮出,则其翻倍效果由原来的 2 倍变为 4 倍。除了玩家手中仅有一张与首家出牌花色相同的牌之外,亮牌阶段中被亮出的牌在该花色的第一轮不能打出。在出牌阶段,在首轮出牌中,一般由拥有梅花 2 的玩家先出,首家出的牌被称为“首引”,后续玩家出牌的花色必须与“首引”花色相同,若玩家手牌中没有与“首引”花色相同的牌,则可以选择一张其他花色的牌打出,称为垫牌,垫牌在每一轮出牌中规定为最小,相同花色牌中,A 最大,2 最小。在一轮出牌结束后,出牌最大的玩家将赢得本轮出的所有牌,并作为下一轮首位出牌玩家。

在游戏结束后,若某位玩家赢得了所有红桃花色的牌,该玩家将获得 +200 分,如果在亮牌阶段红桃 A 被亮出,则获得 +400 分;若某位玩家赢得了所有分牌与梅花 10,那么该玩家将会获得 +800分。在“拱猪”中,最终得分最多的玩家为赢家。规则详情请参见《中国华牌竞赛规则(试行)》[14]。

### 1.2 数据格式设计

在“拱猪”人工智能模型构建中,亮牌阶段使用 1 ×4 的 one-hot 矩阵对亮牌状态与动作进行编

码;出牌阶段使用 1×52 的 one-hot 矩阵对状态和动作进行编码。“拱猪”规定每位玩家每轮只能出一张牌,1×52 的 one-hot 矩阵可以用不同位置的元素来表示对应牌的不同状态,矩阵位置相对应的元素为 1 表示出此牌,为 0 则不出。可被亮的牌在矩阵中的位置对应关系如表 2 所示,出牌阶段所有牌在矩阵中的位置对应关系如表 3 所示。

表 2 可被亮的牌在 1×4 的 one-hot 矩阵中的位置

| 牌 | 红桃 A | 黑桃 Q | 梅花 10 | 方块 J |
|---|---|---|---|---|
| 矩阵位置 | 0 | 1 | 2 | 3 |

表 3 不同牌在 1×52 的 one-hot 矩阵中的位置

| 牌 | 红桃 2~A | 黑桃 2~A | 梅花 2~A | 方块 2~A |
|---|---|---|---|---|
| 矩阵位置 | 0~12 | 13~25 | 26~38 | 39~51 |

除手牌外,1×52 的 one-hot 矩阵还可以编码任意玩家赢得的牌、已出的牌、亮牌等信息,这些信息被编码后可以很容易地被神经网络接收并处理,某位玩家的手牌信息编码如图 1 所示。1×52 的 one-hot 矩阵仅仅只能表示位置信息,而某位玩家的出牌顺序信息也至关重要,使用 13×52 的 one-hot 矩阵可编码某位玩家每轮次的出牌信息。玩家历史出牌的信息具有时序性,可以使用 LSTM 来处理此编码后的信息。

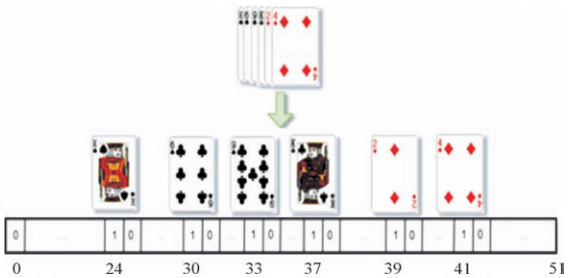


图 1 1×52 的 one-hot 矩阵对某副手牌进行编码

## 2 深度蒙特卡洛算法

深度蒙特卡洛树算法是在生成蒙特卡洛过程中使用深度神经网络代替 Q 学习中的 Q 表来完成 $Q(s,a)$ 的更新,该方法已经在“斗地主”中得到成功应用[11]。本文是将该算法移用于“拱猪”博弈研究中。

### 2.1 传统型蒙特卡洛算法

传统型蒙特卡洛算法属于强化学习算法的一种。强化学习是指智能机器人如何在环境中采取不同的行动以最大程度地提高积累奖励[15],一个典型的强化学习框架由智能代理 Agent、环境 Environment、状态 State、动作 Action、奖励 Reward 以及行动策略 $\pi$ 构成。“拱猪”游戏中,智能代理 Agent在亮牌和出牌阶段分别执行亮牌动作 Action 和出牌动作 Action,当 Agent 执行某个亮牌或者出牌动作后,亮牌或出牌环境 Environment 会转换到一个新的状态 State,即玩家亮牌情况或出牌局面发生变化,对于新的状态,环境会给出奖励信号(正奖励或者负奖励)。智能代理根据新的状态和环境反馈的奖励,按照一定的亮牌或出牌的行动策略 $\pi$ 来执行新的动作。

状态 State 与动作 Action 之间的关系由式(1)决定,其中,$\pi$ 为智能代理的策略。

$$\pi(State) = Action \tag{1}$$

### 2.2 改进型蒙特卡洛算法

传统的蒙特卡洛算法和 Q-learning 算法是通过 $Q(s,a)$ 去决定策略 $\pi$,即智能代理根据 $Q(s,a)$ 来选取获得最大收益的动作。评估 $Q(s,a)$ 的方法为求所有 episode 访问 $(s,a)$ 所得到的回报均值。Q 值的更新函数如式(2)所示。其中 $R$ 为回报函数,$\gamma$ 为折扣因子,$\rho$ 为学习率。

$$Q(s,a) \leftarrow Q(s,a) + \rho(R + \gamma \max_{a'} Q(s',a') - Q(s,a)) \tag{2}$$

Q 值的评估和更新过程可以很自然地与神经网络结合,从而得到改进型的蒙特卡洛算法,即深度蒙特卡洛算法。具体来说,使用神经网络和均方误差来代替 Q 表完成 $Q(s,a)$ 的更新。深度蒙特卡洛算法的伪代码如下所示。

深度蒙特卡洛算法伪代码:

输入:状态-动作对$(s,a)$,学习率$\rho$,折扣因子$\gamma$

输出:新状态$s$

初始化:$Q(s,a)$

for iteration = 1,2,3,…… do (每个 episode)

  Initialize $s$

  for t = 1,2,3,…, $T$ do (episode 每个步骤)

    Choose $a$ from $s$ using policy derived from $Q$

Take action $a$, observe $R, s'$

$Q(s_t, a_t) \leftarrow Q(s_t, a_t) +$

$\rho(R_t + \gamma \max_{a'} Q(s'_{t+1}, a'_{t+1}) - Q(s_t, a_t))$

(使用神经网络和均方误差相结合的方法)

$s \leftarrow s'$

Until $s$ is terminal

end

end

第一个 for 循环中使用 epsilon-greedy 算法来对生成的 episode 进行选择,可以最大程度地提高收益[16],避免模型对预期收益较低的情况进行训练。

第二个 for 循环中的平均回报可通过计算所有$(s,a)$折现的累计奖励获得,改进型蒙特卡洛算法使其能够应用在神经网络中。传统的蒙特卡洛算法依赖 Bootstrapping 的 Q 学习过程繁琐且训练神经网络需要的时间较长[17],而基于改进型蒙特卡洛算法的 Q 学习对于一组$(s,a)$的平均回报能够直接逼近目标 $Q(s,a)$,这将大大缩短模型训练的时间。

传统的蒙特卡洛算法由于高方差的原因,处理非完备信息博弈时效率十分低下。深度蒙特卡洛算法已经在“斗地主”中得到了很好的验证,其算法也同样十分适合“拱猪”游戏。第一:“拱猪”同样是一个回合性的任务游戏,它并不需要处理不完整的回合信息;第二:深度蒙特卡洛算法可以很容易地进行并行计算,每秒可生成更多的样本数据,提高了训练效率,同样可以解决“拱猪”的高方差的问题。第三:“拱猪”智能代理是在没有奖励的情况下对游戏状态和动作进行模拟,这种情况同样会减慢 Q 学习的速度,让模型变得不易收敛,而采用深度蒙特卡洛算法可以减少这种长时间无反馈带来的影响[11]。因此,将深度蒙特卡洛算法应用到“拱猪”游戏中是具有可行性的。

## 3 神经网络结构

“拱猪”有非时序性与时序性 2 种数据,对于非时序性数据,直接使用 one-hot 矩阵进行编码并展平不同特征的矩阵进行连接。对于时序性数据,使用 LSTM 进行相应的处理,一个典型的例子是将某对手玩家的历史出牌数据存储至 13 × 52 的矩阵中,如果玩家尚未出完 13 轮,则使用零矩阵来表示缺失的轮次。相比于其他传统的神经网络结构,LSTM 可以更好地处理时序性数据[18],该算法所需数据较少,能更好适应“拱猪”游戏中的时序性数据。图 2 展示了“拱猪”人工智能模型内部的具体细节,基于蒙特卡洛的 Q 网络由一个 LSTM 和 6 层隐藏维度为 512 的多层感知机组成。这个网络将根据游戏状态与动作来预测给定的 $Q(s,a)$值。

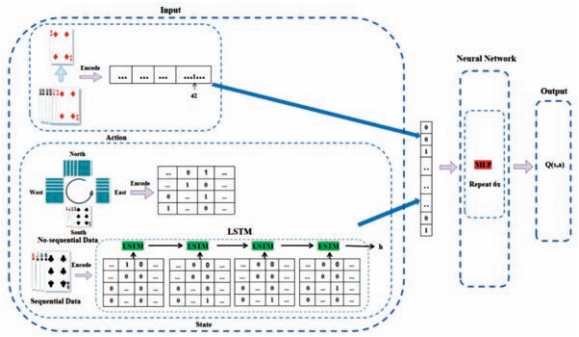


图 2 “拱猪”Q 网络结构模型

“拱猪”是在游戏结束时才能准确计算每位玩家的分数,所以在整个游戏过程中,“拱猪”人工智能模型需要在没有奖励的情况下对游戏状态和行动进行模拟,只有当游戏结束时,才能准确获得这一局游戏的奖励。这种情况会减慢 Q 学习的速度,让模型变得不易收敛,而采用基于深度蒙特卡洛算法的神经网络结构可以有效地解决长时间无反馈所带来的影响。

## 4 模型训练过程

### 4.1 模型特征选择

在参与“拱猪”游戏时,玩家需要关注以下几个方面的信息:自己的手牌、还未出现的牌、自己赢得的牌、其他人赢得的牌、被亮的牌、其余人打过的牌等,这些信息将可决定玩家在出牌时的选择。表 4 为参与游戏的“拱猪”人工智能模型游戏代理所选用的特征参数。

### 4.2 并行训练

在计算资源有限的情况下,采取并行训练可以在单位时间内进行更多次的模型迭代与更新,很大程度上提高了模型的训练效率。

表 4 “拱猪”人工智能模型选用的参数特征

| | 特征 | Size |
|---|---|---|
| 动作 | 游戏代理选择出牌的 one-hot 矩阵 | 52 |
| 状态 | 另 3 位玩家手牌集合的 one-hot 矩阵 | 52 |
| | 游戏代理赢得牌的 one-hot 矩阵 | 52 |
| | 逆时针方向第一位对手玩家赢得的牌 | 52 |
| | 逆时针方向第二位对手玩家赢得的牌 | 52 |
| | 逆时针方向第三位对手玩家赢得的牌 | 52 |
| | 游戏代理亮的牌 | 4 |
| | 逆时针方向第一位对手玩家亮的牌 | 4 |
| | 逆时针方向第二位对手玩家亮的牌 | 4 |
| | 逆时针方向第三位对手玩家亮的牌 | 4 |
| | 逆时针方向第一位对手玩家已出的牌 | 52 |
| | 逆时针方向第二位对手玩家已出的牌 | 52 |
| | 逆时针方向第三位对手玩家已出的牌 | 52 |
| | 当前轮次在游戏代理之前玩家出的牌 | 52 |
| | 本局游戏的历史出牌信息 | 13×208 |

在构建“拱猪”人工智能模型时,将首轮第一个出牌的人工智能模型代理记为 A,逆时针方向上的 3 位玩家分别记为 B、C、D。

进程分为两类:actor 进程与 Learner 进程,actor进程产生训练所需数据,Learner 进程对网络进行训练与更新。“拱猪”的分布式并行训练框架如图 3 所示。

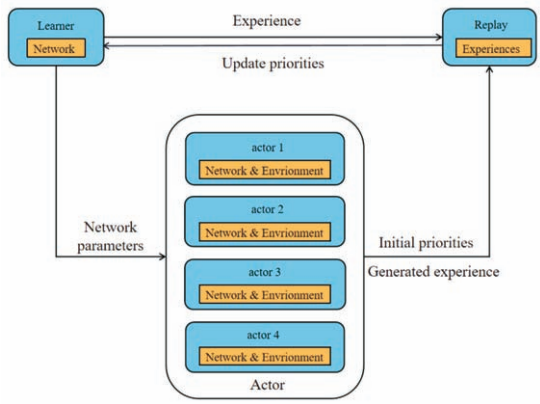


图 3 “拱猪”分布式并行训练框架

本文深度神经网络的训练过程分为 1 个 actor 进程和 1 个 Learner 进程,其中 actor 为总进程,actor 进程包含 4 个玩家的 actor 进程。Learner 进程将为 4 个 actor 进程存储 4 个 Q 网络,这 4 个 Q 网络被称为各个位置的全局网络,这 4 个全局 Q 网络将会根据对应 actor 进程的均方误差进行更新以接近目标值。同时,每个 actor 进程也存储了 4 个 Q 网络,这 4 个 Q 网络被称为各个位置的本地 Q 网络。在整体训练框架中,本地 Q 网络将会在一定训练时间后更新为全局 Q 网络,Actor 进程将从整体游戏环境中获取计算所需的信息并以此更新不同条件下的 $Q(s,a)$。Learner 进程与 Actor 进程之间的通信通过缓冲区进行,每个缓冲区被区分为不同的数据区,不同数据区中存储着游戏运行与模型训练所需的关键数据。

### 4.3 模型训练参数

“拱猪”人工智能模型训练的过程采用分布式并行计算的方式进行,分布式并行计算框架的参数如表 5 所示。

表 5 “拱猪”人工智能模型分布式并行计算参数设置

| 参数名 | 参数值 |
|---|---|
| GPU | NVIDIA RTX 2070 SUPER |
| OS | Ubuntu 18.04 |
| Actor 进程数量 | 1 |
| actor 进程数量 | 4 |
| Learner 进程数量 | 1 |
| 学习率 | 0.000 1 |
| epsilon | 1e-5 |

在单台 GPU 的服务器上训练 24 h 后,初步得到一个“拱猪”人工智能游戏代理。

“拱猪”人工智能模型训练代码及测试文件见 https://github.com/Zhm0715/GZero。

## 5 实验结果及分析

### 5.1 重要指标分析

在“拱猪”人工智能模型游戏代理的训练过程中,actor 进程为游戏中的每个位置(A、B、C、D)都训练了一个自对弈的游戏代理。对于每一个 episode,训练环境将随机生成一组手牌,4 个游戏代理将在非完全信息的条件下进行自对弈。游戏结束后,该局游戏生成的训练数据将被传递给 Learner 进程进行全局 Q 网络的更新,并将这种更新同步至 Actor 进程的本地 Q 网络。图 4 展示了

训练过程中的 Loss 值与训练数据量的关系。

由图 4 可知,4 个方位的游戏代理整体的 Loss 值随着训练数据量的增加而减少,这说明模型在经过训练后逐步趋于局部或整体最优值。对于某方位的游戏代理,在训练过程中会出现 Loss 值突增后再回落的现象,这主要是由于该模型训练过程中采用了 epsilon-greedy 算法来避免 4 个方位的游戏代理同时陷入局部最优值导致训练效果下降。

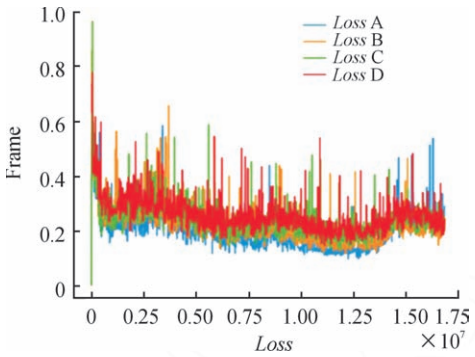


图 4 Loss 值与训练数据量的关系

除了 Loss 值,一局游戏最终的得分也至关重要。图 5 展示了训练过程中的 episode 返回的奖励与训练数据量的关系。为便于计算,此处奖励值为原始分数值的 1/100。

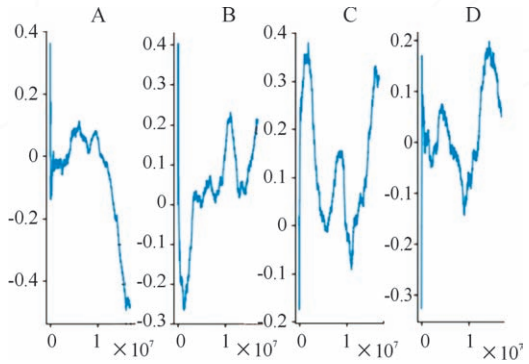


图 5 episode 返回的奖励与训练数据量的关系

由图 5 可知,A 方位玩家的 episode 奖励随着训练次数的增加而下降,而其他方位的玩家则表现出了 episode 奖励不稳定的情况。在训练初期,A 方位的游戏代理表现得比其他方位的游戏代理更好;在训练中后期,B、C、D 方位的游戏代理表现得比 A 方位的游戏代理更好。

在训练完成后,本模型与拟合了 20 000 条真实人类玩家对局数据的卷积模型游戏代理对弈了 10 000 局,该模型获得了 78.3% 的胜率。表 6 为模型评估的关键参数。

表 6 对弈结果关键参数

| 方位 | 游戏代理 | 胜率/% | episode 返回的奖励 |
|---|---|---|---|
| North | 本模型 | 78.3 | 0.67 |
| West | 卷积模型 | 18.8 | -0.19 |
| South | 卷积模型 | 11.4 | -0.27 |
| East | 卷积模型 | 16.5 | -0.21 |

由表 7 可知,该模型返回的平均 episode 激励为 0.67,这表明了在 10 000 局对弈中,该模型游戏代理最终分数为正的局数为 7 830 局,平均每局获得 67 分。除了 North 方位的游戏代理采用基于深度强化学习的“拱猪”人工智能模型外,West、South、East 方位的游戏代理均采用卷积模型,这 3 个方位的游戏代理平均每局获得的分数均为负数。由实验结果可知,该模型在与卷积模型游戏代理进行对弈时能够获得巨大优势。

### 5.2 算法成功分析

通过基于深度蒙特卡洛算法的模型与基于卷积模型之间的对弈,分析本文所提算法性能的优越性。卷积模型主要采用已有的少量且质量不高的真人玩家的对局数据,通过监督学习的方法,使智能代理能够拟合人类玩家的游戏策略。这种算法高度依赖数据的质量和数量,无法很好地利用先前已学到的知识,更为重要的是,该算法也无法探索已有数据之外的知识,导致智能体的水平很难得到进一步地提升。由对弈结果数据分析可知,基于深度蒙特卡洛算法的智能代理在胜率上要远高于基于卷积算法的智能代理,这是由以下深度蒙特卡洛算法的 3 个优点所决定的。

第一,它不需要高度依赖于训练数据的数量和质量,可以自然地模拟人类的行为并进行评估学习,采用分布式并行计算的方式可以在每秒内产生多组训练数据,大大地提高了训练的效率,可有效解决高方差问题;第二,相较于卷积算法模型,深度蒙特卡洛算法模型可以很好地权衡利用

和探索,在训练过程中不仅可以利用先前已学到的知识,还可以轻易地探索到人类玩家不常见的状态与动作并进行训练与学习,可有效地提高智能代理的游戏水平;第三,在大量的自对弈过程中,偶然的错误动作及其评估不会决定着最终训练的效果,可以减少低质量数据对实验结果的影响。

### 5.3 示例分析

在指标参数之外,“拱猪”人工智能游戏代理在实战中的表现也让人欣喜。图 6 展示了在某局游戏中“拱猪”人工智能游戏代理的对局情况。

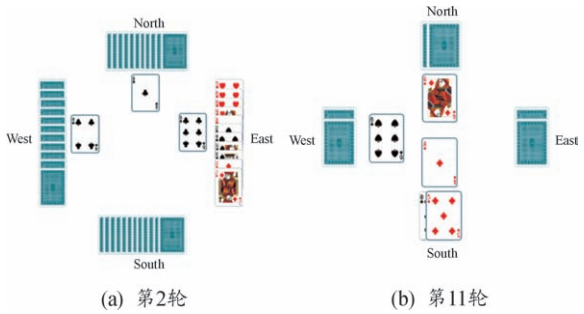


图 6 某局中“拱猪”智能代理出牌动作

由图 6 可知,在本局游戏的第 2 轮中,East 方位的游戏代理首先出牌,因其手牌中含有黑桃 Q 和多张负分较大的红桃牌张,且只有一张梅花花色的牌,故游戏代理选择将惟一一张梅花花色的梅花 6 打出,在后续的游戏过程中,其他玩家出黑桃或者红桃花色牌时,可以迅速将负分牌打出,避免获得较多负分牌而输掉游戏。

在本局游戏的第 11 轮中,North 方位的玩家首先出打出方块 Q,随后 West 方位的玩家打出一张黑桃,说明此时的 West 方位玩家手中已经没有方块花色牌张。结合游戏历史出牌信息,从 South 方位玩家的视角来看,此时未被打出的方块花色牌只有方块 J 和方块 4,且 North 方位玩家先出方块 Q,其目的很可能是想得到方块 J 从而获得正分,所以可以推测出玩家 North 手中没有方块 J 的可能性较大,在这种情况下,East 方位玩家手牌中有且仅有一张方块花色的牌,即为方块 J,此时 South 方位玩家打出方块 A 就有很大的几率得到 East 方位玩家手牌中的方块 J,从而使自己获得正分。

## 6 结论

将在“斗地主”游戏中表现优异的算法移用于“拱猪”博弈研究中,利用蒙特卡洛方法与深度神经网络相结合的方式,构建了一个基于分布式并行计算的自对弈“拱猪”人工智能模型。该模型概念简单,训练高效。通过评估训练的结果可知,该模型比采用深度神经网络拟合人类玩家对局数据的人工智能游戏代理表现出更好的性能。

尽管该模型的人工智能游戏代理的表现让我们欣喜,但仍有很多问题尚待解决。例如是否有其他网络结构比现有的模型训练效果更好?该模型能否根据游戏现有信息对其他玩家手牌进行预测?该模型是否能够在其他类似问题上取得预期的效果?这些问题都将在后续研究中进行探索。

## 参考文献:

[1] HOLMES R J,DANDRADE B W,FORREST S R,et al. Efficient, deep-blue organic electrophosphorescence by guest charge trapping[J]. Applied Physics Letters,2003,83(18):3818 – 3820.

[2] SILVER D,HUANG A,MADDISON C J,et al. Mastering the game of Go with deep neural networks and tree search [J]. Nature,2016,529(7587):484 – 489.

[3] SILVER D, HUBERT T, SCHRITTWIESER J, et al. A general reinforcement learning algorithm that masters chess,shogi,and go through self-play[J]. Science,2018,362(6419):1140 – 1144.

[4] LAIR A, SAFFIDINE A. AI surpasses humans at six-player poker[J]. Science,2019,365(6456):864 – 865.

[5] NOAM B,SANDHOLM T. Superhuman AI for heads-up no-limit poker:Libratus beats top professionals[J]. Science,2018,359(6374):418 – 424.

[6] 王亚杰,丁傲冬,祁冰枝,等. 基于预期收益策略与 UCT 的德州扑克算法[J]. 重庆理工大学学报(自然科学),2021,35(3):166 – 173.

[7] BOWLING M,BURCH N,JOHANSON M,et al. Heads-up limit hold’em poker is solved[J]. Science,2015,347(6218):145 – 149.

[8] BROWN N,LERER A,GROSS S,et al. Deep counterfactual regret minimization [J]. Proceedings of Machine Learning Research,2019,97:793 – 802.

[9] 张小川,杜松,赵海璐,等. 一种德州扑克牌力评估方

法[J]. 重庆理工大学学报(自然科学),2021,35(9):130 - 135.

[10] LI Junjie, KOYAMADA S, YE Qiwei, et al. Suphx: Mastering mahjong with deep reinforcement learning[EB/OL]. (2020 - 04 - 01)[2022 - 08 - 09]. https://arxiv.org/pdf/2003.13590.pdf.

[11] ZHA Daochen, XIE Jingru, MA Wenye, et al. DouZero: Mastering Doudizhu with self-play deep reinforcement learning[J]. Proceedings of Machine Learning Research, 2021,139:12333 - 12344.

[12] BERNER C, BROCKMAN G, CHAN B, et al. Dota 2 with large scale deep reinforcement learning[EB/OL]. (2019 - 12 - 13)[2022 - 08 - 10]. https://arxiv.org/pdf/1912.06680.pdf.

[13] YE Deheng, ZHAO Liu, SUN Mingfei, et al. Mastering complex control in MOBA games with deep reinforcement learning[J]. Proceedings of the AAAI Conference on Artificial Intelligence, 2020, 34(4): 6672 - 6679.

[14] 中国华牌竞赛规则编写组. 中国华牌竞赛规则(试行)[M]. 北京:人民体育出版社,2009.

[15] 陆鑫,高阳,李宁,等. 基于神经网络的强化学习算法研究[J]. 计算机研究与发展,2002(8):981 - 985.

[16] BULUT V. Optimal path planning method based on epsilon-greedy Q-learning algorithm[J]. Journal of the Brazilian Society of Mechanical Sciences and Engineering, 2022,44(3):1 - 14.

[17] ZIMMER M, DONCIEUX S. Bootstrapping Q-Learning for robotics from neuro-evolution results[J]. IEEE Transactions on Cognitive and Developmental Systems, 2018, 10(1):102 - 119.

[18] 王霄鹏. 基于 LSTM 改进模型的股票预测研究[D]. 重庆:重庆理工大学,2020.

# Deep Monte Carlo algorithm for Gongzhu game

WU Licheng, WU Qifei, ZHONG Hongming, LI Xiali

(School of Information Engineering, Minzu University of China, Beijing 100081, China)

**Abstract**: The existing convolutional neural network model for Gongzhu game is computationally complex and highly dependent on expert knowledge. In order to solve this problem, a deep Monte Carlo algorithm combining deep neural network and Monte Carlo method is proposed for Gongzhu. This algorithm uses the self-play method to simulate and evaluate actions and states, and uses a deep Q-network to replace Q-table to complete the updating of the Q-value, efficiently exploring and utilizing the strategy for Gongzhu. Besides, this algorithm also uses distributed parallel computing to improve training efficiency. Compared with the traditional Monte Carlo method, the proposed algorithm can effectively solve the problem of high variance. After training on a single server with one GPU for 24 hours, the constructed intelligent agent applied to the proposed algorithm plays 10 000 games against Gongzhu convolutional neural network model. The experimental results show that the intelligent agent has a winning rate of 78.3%, with an average of 67 points per game. The analysis of specific examples further verifies the effectiveness of the algorithm and a good performance of the intelligent agent.

**Key words**: artificial intelligence; Gongzhu; deep reinforcement learning; Monte Carlo method

(责任编辑 王 欢)