DouZero+: Improving DouDizhu AI by Opponent Modeling and Coach-guided Learning

Youpeng Zhao, Jian Zhao, Xunhan Hu, Wengang Zhou, Houqiang Li,

Abstract—**Recent years have witnessed the great breakthrough of deep reinforcement learning (DRL) in various perfect and imperfect information games. Among these games, DouDizhu, a popular card game in China, is very challenging due to the imperfect information, large state space, elements of collaboration and a massive number of possible moves from turn to turn. Recently, a DouDizhu AI system called DouZero has been proposed. Trained using traditional Monte Carlo method with deep neural networks and self-play procedure without the abstraction of human prior knowledge, DouZero has outperformed all the existing DouDizhu AI programs. In this work, we propose to enhance DouZero by introducing opponent modeling into DouZero. Besides, we propose a novel coach network to further boost the performance of DouZero and accelerate its training process. With the integration of the above two techniques into DouZero, our DouDizhu AI system achieves better performance and ranks top in the Botzone leaderboard among more than 400 AI agents, including DouZero.**

Index Terms—**DouDizhu, Reinforcement learning, Monte-Carl Method, Opponent Modeling, Coach Network**

I. Introduction

During the development of artificial intelligence, games often serve as an important testbed as they are good abstraction of many real-world problems, and more objective compared to environments specially designed for testing AI since games are developed for humans. In recent years, significant progress has been made in solving perfect-information games such as Go [1]–[3], Shogi (Janpanese chess) [4] and even fighting game [5]. The current research efforts are turning to more challenging imperfect information games (IIG) where agents may cooperate or compete with each other under a partially observable environment. Encouraging achievements have been made from two-player games, such as simple Leduc Hold'em and limit/no-limit Texas Hold'em [6]–[9] to multi-player games, including multi-player Texas Hold'em [10], StarCraft [11], DOTA [12] and Japanese Mahjong [13].

In this work, we are dedicated to designing an AI program for DouDizhu, *a.k.a,* Fighting the Landlord, which is the most popular card game in China with hundreds of millions daily active players. DouDizhu has two interesting characteristics that pose great challenges for AI programs. First, this game involves both cooperation and competition simultaneously in a partially observable environment. To be specific, the two Peasant agents play as a team to fight against the Landlord agent. For example, Figure 1 shows a typical situation where the bottom Peasant can play a small Solo card to help his partner to win. This property makes the popular algorithms for Poker games, such as Counterfactual Regret Minimization (CFR) [14] and its variants not suitable in such a complex three-player setting. Second, DouDizhu has a large and complex state and action space due to the combination of cards and the complex rules compared to other card games. There are thousands of possible combinations of cards where different subsets of these combinations are legal to different hands. Figure 2 exhibits an example of a hand that has 119 legal moves, including Solo, Pair, Trio, Chain of Solo and so on. Unlike Texas Hold'em, the actions in DouDizhu can not be easily abstracted, which makes search computationally expensive and the commonly used reinforcement learning (RL) algorithms less effective. The performance of Deep Q-Learning (DQN) [15] will be greatly affected due to the overestimating issue in large action space [16] while policy gradient methods such as A3C [17] fail to leverage the action features, limiting the capability of generalizing over unseen actions. In this way, previous work has shown that DQN and A3C have a poor performance in DouDizhu, only having less than 20% winning percentage against simple rule-based agents even with twenty days of training [18].

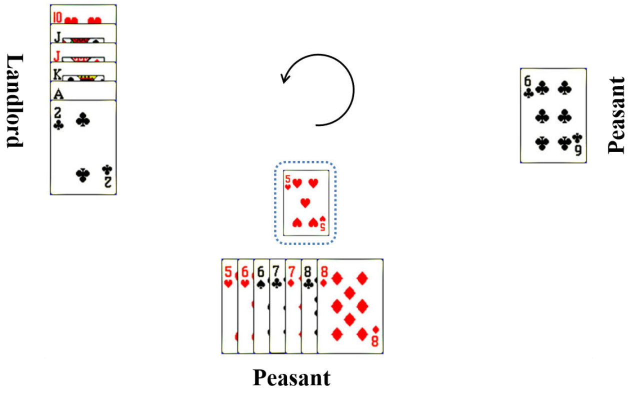


Fig. 1: A case example about cooperation in DouDizhu. If the Peasants learn to cooperate with each other, current player should play small Solo to let the teammate to win the game.

Despite the challenges mentioned above, some achievements have been made in building DouDizhu AI. To deal with the large action space in DouDizhu, Combinatorial Q-Network (CQN) [18] decouples the actions into decomposition selec-

Y. Zhao, J. Zhao, X. Hu, W. Zhou and H. Li are with the CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China.

W. Zhou and H. Li are also with Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

DouZero+: 通过对手建模和教练引导学习改进斗地主AI

赵有鹏, 赵健, 胡迅涵, 周 Wentang, 李厚强,

***Abstract*——近年来，深度强化学习（DRL）在各种完美信息和不完美信息游戏中取得了巨大突破。在这类游戏中，由于不完美信息、庞大的状态空间、合作元素以及每轮可能动作的大量增加，中国流行的纸牌游戏斗地主极具挑战性。最近，提出了一种名为DouZero的斗地主AI系统。DouZero通过传统的蒙特卡洛方法和深度神经网络进行自我对弈训练，不依赖人类先验知识的抽象，已经超越了所有现有的斗地主AI程序。在本文中，我们通过引入对手建模来增强DouZero。此外，我们提出了一种新的教练网络，以进一步提升DouZero的性能并加速其训练过程。通过将上述两种技术整合到DouZero中，我们的斗地主AI系统在Botzone排行榜上表现出色，并在超过400个AI代理中名列前茅，包括DouZero。*

***Index Terms*——斗地主，强化学习，蒙特卡洛方法，对手建模，教练网络**

I. 引言

在人工智能的发展过程中，游戏常常作为重要的试验平台，因为它们是对许多现实世界问题的良好抽象，并且与专门为测试AI设计的环境相比，更加客观，因为游戏是为人类设计的。近年来，在解决完美信息游戏方面取得了显著进展，如围棋[1]-[3]、将棋（日本国际象棋）[4]，甚至格斗游戏[5]。当前的研究重点转向了更具挑战性的不完美信息游戏（IIG），在这种游戏中，智能体可能在部分可观测的环境中相互合作或竞争。从两人游戏，如简单的Leduc Hold’em和限注/不限注德州扑克[6]-[9]，到多人游戏，包括多人德州扑克[10]、StarCraft[11]、DOTA[12]和日本麻将[13]，已经取得了令人鼓舞的成就。

在本项工作中，我们致力于为斗地主设计一个AI程序，*a.k.a*, 拔草 landlord，这是中国最受欢迎的纸牌游戏，每天有数亿活跃玩家。斗地主有两个有趣的特性，给AI程序带来了巨大的挑战。首先，这个游戏在部分可观测的环境中同时包含了合作和竞争。具体来说，两个农民代理作为一个团队与地主代理对抗。

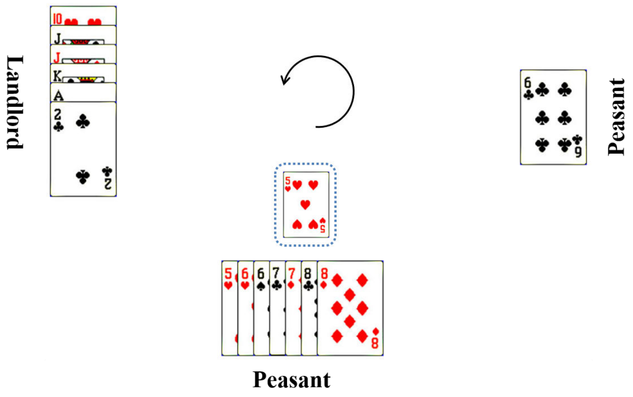


Fig. 1: DouDizhu 中的一个合作案例。如果农民学会相互合作，当前玩家应该玩小单手让队友赢得游戏。

例如，图1显示了一个典型的场景，底牌农民可以通过打出一张小单牌来帮助他的搭档获胜。这一特性使得像假设性遗憾最小化（CFR）[14]及其变体这样的扑克游戏流行算法，在这种复杂的三人设置中并不适用。其次，斗地主由于牌的组合和复杂的规则，拥有一个庞大且复杂的状态和动作空间，与其他纸牌游戏相比更为复杂。存在成千上万种可能的牌组组合，其中不同的子集对不同的手牌来说是合法的。图2展示了拥有119种合法走法的手牌示例，包括单牌、对牌、三张、单牌链等。与德州扑克不同，在斗地主中，动作无法轻易抽象化，这使得搜索计算成本高昂，常用的强化学习（RL）算法效果不佳。由于在大动作空间中存在过度估计问题[16]，深度Q学习（DQN）[15]的表现会受到严重影响，而基于策略梯度的方法如A3C[17]则无法利用动作特征，限制了其在未见过的动作上的泛化能力。因此，以往的工作已经表明，DQN和A3C在斗地主中的表现不佳，即使经过二十天的训练，对战简单的规则基础代理时胜率也低于20%。

尽管存在上述挑战，已在构建 DouDizhu AI 方面取得了一些成就。为了应对 DouDizhu 中庞大的动作空间，组合 Q 网络 (CQN) [18] 将动作分解为分解选择-

Y. Zhao, J. Zhao, X. Hu, W. Zhou and H. Li are with the CAS Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, China.

W. Zhou and H. Li are also with Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China

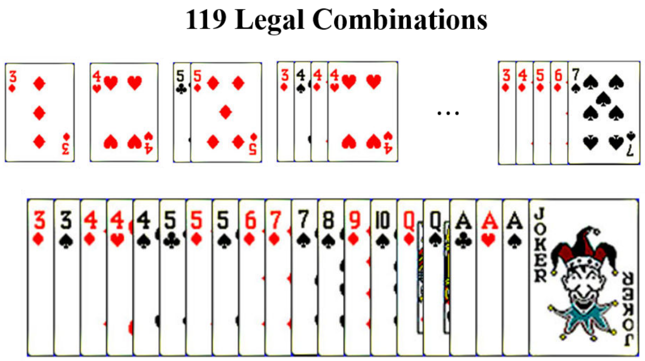


Fig. 2: A hand and its corresponding legal moves.

tion and final move selection. However, the decomposition selection relies on human heuristics and is time-consuming, which limits its performance. In fact, CQN does not have preponderance over the heuristic rule-based model. DeltaDou [19] is the first bot that reaches top human-level performance compared to human experts. It makes use of an AlphaZero-like algorithm, which combines neural networks with Fictitious Play Monte Carlo Tree Search (FPMCTS), and an inference algorithm in a self-play procedure. However, DeltaDou pre-trains a kicker network based on heuristic rules to reduce the action space size, which may have a negative impact on its strength if the output of the kicker network is not optimal. Moreover, the inference and search are so computationally expensive that it takes two months to finish the training of DeltaDou. Recently, DouZero [20] has attracted considerable attention due to its outstanding performance in this complex game. It utilizes self-play deep reinforcement learning without the abstraction of state/action space and human knowledge. It combines classical Monte-Carlo methods [21] with deep neutral networks to handle the large state and action space, which opens another door for such complex and large-scale games.

In this work, we improve DouZero by introducing opponent modeling and coach-guided learning. Opponent modeling aims to determine a likely probability distribution for the opponents' hidden cards, which is motivated by the fact that human players will try to predict the opponents' cards to help them determine the policy. Due to the complexity of DouDizhu, a lot of actions may be appropriate when making the decision. In this case, analyzing the opponents' cards will be of great importance because grasping this information helps the bot choose the optimal move. On the other hand, we propose coach-guided learning to fasten the training of the AI. Due to the large information space in this game, the training of the AI program for DouDizhu costs a lot of time. Considering the fact that the outcome of DouDizhu depends heavily on the initial cards of three players, quite a few games are of little value for learning. To this end, we design a novel coach network to evenly pick matched openings so that the models can learn from more valuable data without wasting time to play valueless games, thus accelerating the training process. Through integrating these techniques into DouZero, our DouDizhu AI program achieves a better performance than the original DouZero and ranks the first on the Botzone [22]–[24] leaderboard among more than four hundred agents, including DouZero.

II. Related Work

In this section, we briefly introduce the application of reinforcement learning in imperfect-information games as well as the works about opponent modeling.

A. *Reinforcement Learning for Imperfect-Information Games*

Recent years have witnessed the successful application of reinforcement learning in some complex imperfect-information games. For instance, there are quite a few works about reinforcement learning for poker games [7], [25], [26]. Different from Counterfactual Regret Minimization (CFR) [14] that relies on game-tree traversals, RL is based on sampling so that it can easily generalize to large-scale games. In this way, OpenAI, DeepMind and Tencent have utilized this technique to build their game AI in DOTA [12], StarCraft [11] and Honor of Kings [27], respectively and acquired amazing achievements, proving the effectiveness of reinforcement learning in imperfect-information games. More recently, there are some research efforts that combine reinforcement learning with search and have shown its effectiveness in poker games such as heads-up no-limit Texas Hold'em poker and DouDizhu [19], [28].

However, due to the complexity of DouDizhu, traditional reinforcement learning methods such as DQN [15] and A3C [17] exhibit poor performance in this game as discussed above. Even an improved method, *i.e.* Combinatorial Q-Network, also fails to achieve satisfactory performance. What's more, DeltaDou [19], which infers the hidden information and uses MCTS to combine RL with search, is computationally expensive and depends on human expertise, limiting its practicability and performance. To this end, DouZero [20] utilizes Monte-Carlo methods [21] and manages to defeat all DouDizhu AI programs by now. We note that this technique is also adopted in some other game AIs, such as a modern board game, Kingdomino, and a kind of new chess, Tibetan Jiuqi [29], [30]. But unlike these environments, DouDizhu is a complex imperfect-information game that requires competition and cooperation over the large state and action space. The amazing performance of DouZero reveals the good results of Monte-Carlo methods in such large-scale complex card games, providing new insight into future research on handling complex action space, sparse reward and imperfect information.

B. *Opponent Modeling for Games*

In human practice, gaining an abstract description of the opponent will give the player a clear advantage in games, especially imperfect-information games. As a result, opponent modeling has attracted substantial attention in game AI. For example, Southey *et al.* [31] put forward a Bayesian

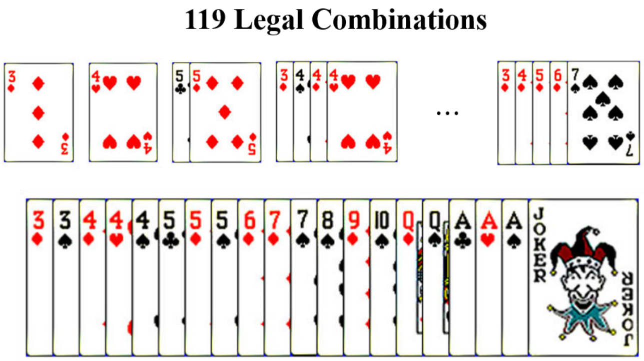


Fig. 2: 一只手及其对应的合法移动。

tion和最终动作选择。然而，分解选择依赖于人工启发式规则并且耗时，这限制了其性能。事实上，CQN 并不比启发式规则模型更具优越性。DeltaDou [19] 是第一个在与人类专家相比达到顶级人类水平性能的机器人。它利用了一种类似于AlphaZero的算法，结合了神经网络与假想博弈蒙特卡洛树搜索（FPMCTS），以及在自我对弈过程中的一种推理算法。然而，DeltaDou 基于启发式规则预训练了一个踢球网络以减少动作空间大小，如果踢球网络的输出不是最优的，这可能对其强度产生负面影响。此外，推理和搜索非常耗费计算资源，使得完成DeltaDou的训练需要两个月时间。最近，DouZero [20] 由于在这一复杂游戏中表现出色而引起了广泛关注。它利用自我对弈深度强化学习，没有抽象的状态/动作空间和人类知识。它结合了经典的蒙特卡洛方法 [21] 与深度神经网络来处理大规模的状态和动作空间，为这类复杂和大规模游戏打开了另一扇门。

在这项工作中，我们通过引入对手建模和教练引导学习来改进DouZero。对手建模旨在确定对手隐藏牌的可能概率分布，这受到人类玩家会尝试预测对手的牌以帮助他们制定策略这一事实的启发。由于DouDizhu的复杂性，在做决策时可能有大量行动是合适的。在这种情况下，分析对手的牌将非常重要，因为掌握这些信息有助于机器人选择最优行动。另一方面，我们提出了教练引导学习来加快AI的训练速度。由于该游戏的信息空间很大，DouDizhu的AI程序训练需要花费大量时间。考虑到DouDizhu的结果很大程度上取决于三个玩家的初始牌，许多游戏对于学习来说价值不大。因此，我们设计了一个新颖的教练网络，以均匀地选择匹配的开局，从而使模型可以从更有价值的数据中学习，而不会浪费时间。

时间来玩无价值的游戏，从而加速训练过程。通过将这些技术整合到 DouZero 中，我们的 DouDizhu AI 程序在超过四百个代理中，包括 DouZero，于 Botzone [22]–[24] 领先榜上排名第一，实现了更好的性能。

II. 相关工作

在本节中，我们简要介绍了强化学习在不完全信息博弈中的应用以及关于对手建模的相关工作。

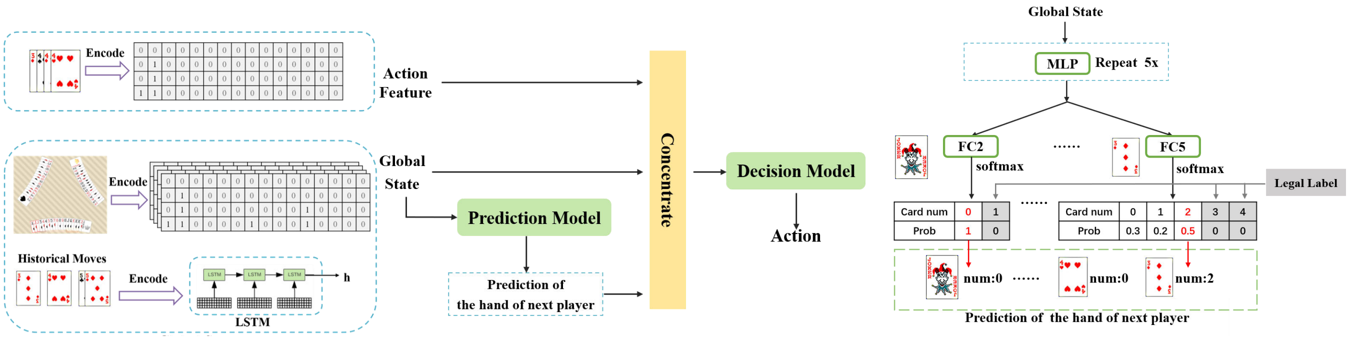
A. Reinforcement Learning for Imperfect-Information Games

近年来，强化学习在一些复杂的不完美信息游戏中得到了成功的应用。例如，有关扑克游戏的强化学习研究相当多，如[7]、[25]、[26]。与依赖于游戏树遍历的Counterfactual Regret Minimization (CFR) [14]不同，强化学习基于采样，因此它可以很容易地应用于大规模游戏。通过这种方式，OpenAI、DeepMind 和腾讯分别利用这一技术在《DOTA》[12]、《星际争霸》[11] 和《王者荣耀》[27] 中构建了其游戏AI，并取得了令人瞩目的成就，证明了强化学习在不完美信息游戏中的有效性。最近，有一些研究努力将强化学习与搜索相结合，并在如限注德州扑克和斗地主[19]、[28]等扑克游戏中展示了其有效性。

然而，由于斗地主的复杂性，传统的强化学习方法如DQN [15] 和A3C [17] 在上述游戏中表现不佳。即使改进的方法*i.e.*组合Q网络也无法达到令人满意的表现。更糟糕的是，DeltaDou [19] 通过推断隐藏信息并使用MCTS将RL与搜索相结合，虽然计算成本高昂且依赖于人类专业知识，限制了其实用性和表现。为此，DouZero [20] 利用蒙特卡洛方法 [21]，现在已经击败了所有斗地主AI程序。我们注意到，这种方法也被其他一些游戏AI所采用，例如现代桌面游戏Kingdomino和一种新型象棋Tibetan Jiuqi [29]、[30]。但是与这些环境不同，斗地主是一种复杂的不完美信息游戏，需要在大状态和动作空间中进行竞争和合作。DouZero 的出色表现揭示了蒙特卡洛方法在如此大规模复杂纸牌游戏中的良好效果，为未来处理复杂动作空间、稀疏奖励和不完美信息的研究提供了新的见解。

B. Opponent Modeling for Games

在人类实践中，获得对手的抽象描述将使玩家在游戏中获得明显的优势，尤其是在不完美信息游戏中。因此，对手建模在游戏AI中引起了广泛关注。例如，Southey *et al.* [31] 提出了一个贝叶斯方法



(a) The overall framework

(b) The details about prediction model

Fig. 3: An overview of the framework that combines opponent modeling with DouZero and the details about the prediction model. The prediction model takes the state information, which is the same as DouZero, as input and outputs the probability of the number of every card in the hand of the next agent. The decision model is trained using deep Monte-Carlo algorithm like DouZero. The prediction result about hand cards of the next player is concatenated with the state features as well as the action features and all these information is input to decision model to decide which action to take. As for the prediction model, it can be viewed as a multi-head classifier, which consists of a layer of LSTM to encode historical moves, five shared layers of MLP and multi-head FC layers to output the probability. We can extract "legal label" from the state information, which represents how many cards of each kind the next player has at most, to help filter out impossible answers.

probabilistic model for poker games which infers a posterior over opponent strategies and makes an appropriate response to that distribution. In another complex imperfect-information game, Mahjong, an AI bot is designed based on opponent modeling and Monte Carlo simulation [32]. In this work, the opponent models are trained with expert game records and the bot decides the move using the prediction results and Monte-Carlo simulation. What's more, Schadd *et al.* [33] propose an approach for opponent modeling in RTS games . It employs hierarchically structured models to classify the strategy of the opponent, where the top-level can distinguish the general style of the opponent and the bottom level can classify the specific strategies that define the opponent's behaviour.

Recently, inspired by the success of reinforcement learning, many researchers combine opponent modeling with reinforcement learning and have made much progress. In combination with deep Q-learning, opponent modeling achieves superior performance over DQN and its variants in a simulated soccer game and popular trivia game [34]. Knegt *et al.* [35] introduces the opponent modeling technique into an arcade video game using reinforcement learning, which helps the agent predict opponents' actions and significantly improves the agent's performance. In addition, opponent modeling can be adopted in multi-agent reinforcement learning problems where RL agents are designed to consider the learning of other agents in the environment when updating their own policies [36]. Another promising solution is to mimic human players by combining opponent models used by expert players and reinforcement learning [37]. All the above works demonstrate that combining opponent modeling with reinforcement learning is beneficial to achieve performance gain in multi-agent imperfect-information games, which also inspires this work.

III. Preliminary

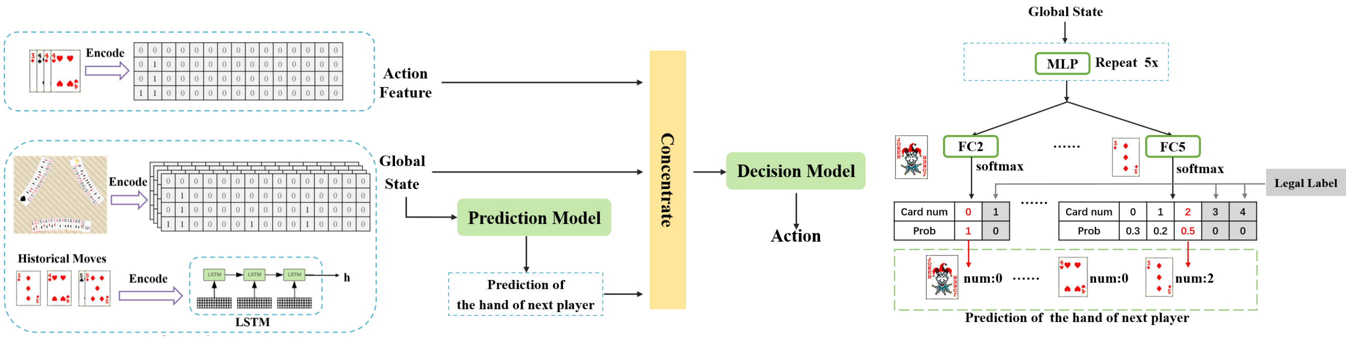
In this section, we first discuss the main algorithm of DouZero, *i.e.* Deep Monte Carlo (DMC), which generalizes Monte Carlo (MC) method with deep neural networks for function approximation. Then, we briefly describe the details of DouZero system.

As a key technique in reinforcement learning, Monte Carlo (MC) method learns value functions and optimal policies from experience, namely, sampling sequences of states, actions and rewards from actual or simulated interactions with the environment [21]. This technique is designed for episodic tasks, where experience can be divided into episodes that eventually terminate, and it updates the value estimation and policy only when an episode is completed. To be specific, after each episode, the observed returns are used for policy evaluation and then the policy can be improved at the visited states in the episode. To optimize a policy π using MC methods, the procedure is intuitively described as follows:

1) Generate an episode using π.
2) For each state-action pair (s, a) visited in the episode, calculate and update $Q(s, a)$ with the average return.
3) For each state s in the episode, update the policy: $\pi(s) \leftarrow argmax_{\mathbf{a} \in A} Q(s, \mathbf{a})$.

When putting MC methods into practice, we can utilize epsilon-greedy to balance between exploration and exploitation in Step 1. Also, the above procedure can be naturally combined with deep neural networks, leading to Deep Monte-Carlo (DMC). In this way, the Q-table $Q(s, a)$ can be replaced by neural networks which can be optimized with mean-square-error (MSE) loss in Step 2.

As DouDizhu is a typical episodic task, MC is naturally suitable for this problem. What's more, DMC requires a large amount of experience for training while it's easy to generate



(a) The overall framework (b) The details about prediction model

Fig. 3: 结合对手建模与DouZero的框架概述及其预测模型的详细信息。预测模型以状态信息为输入，该状态信息与DouZero相同，并输出下一位玩家手中每张牌的数量的概率。决策模型使用类似于DouZero的深度蒙特卡洛算法进行训练。预测结果与手牌信息以及状态特征和动作特征连接后，所有这些信息被输入决策模型以决定采取哪种行动。对于预测模型，可以将其视为一个多头分类器，该分类器包含一层LSTM来编码历史动作，五层共享的MLP和多头FC层来输出概率。我们可以从状态信息中提取“合法标签”，表示下一位玩家最多可能有多少张每种类型的牌，以帮助过滤掉不可能的答案。

一种用于扑克游戏的概率模型，可以推断对手策略的后验分布并据此做出适当的响应。在另一个复杂的不完全信息游戏中，麻将，基于对手建模和蒙特卡洛模拟设计了一个AI机器人[32]。在本工作中，对手模型是通过专家游戏记录训练的，机器人使用预测结果和蒙特卡洛模拟来决定行动。此外，Schadd *et al.* [33] 提出了一种在即时战略游戏中对手建模的方法。该方法使用分层结构的模型来分类对手的战略，其中顶层可以区分对手的一般风格，而底层可以分类定义对手行为的具体策略。

最近，受强化学习成功的启发，许多研究人员将对手建模与强化学习相结合，并取得了显著进展。结合深度Q学习，对手建模在模拟足球游戏和流行 trivia 游戏中优于DQN及其变体 [34]。Knegt *et al.* [35] 使用强化学习将对手建模技术引入街机视频游戏，这有助于代理预测对手的动作，并显著提高了代理的性能。此外，在多智能体强化学习问题中，对手建模可以被采用，其中RL代理在更新自身策略时会考虑环境中的其他代理的学习 [36]。另一种有前途的解决方案是通过结合专家玩家使用的方法和强化学习来模仿人类玩家 [37]。所有上述工作都证明了将对手建模与强化学习结合使用有助于在多智能体不完美信息游戏中实现性能提升，这也启发了本项工作。

III. 预liminary

在本节中，我们首先讨论 DouZero 的主要算法 *i.e.* 深度蒙特卡洛 (DMC)，该算法通过深度神经网络将蒙特卡洛 (MC) 方法扩展到函数逼近。然后，我们简要描述 DouZero 系统的细节。

作为强化学习中的关键技术，蒙特卡洛（MC）方法通过从实际或模拟与环境的交互中采样状态、动作和奖励来学习价值函数和最优策略[21]。该技术适用于阶段性任务，其中经验可以划分为最终会终止的阶段，并且只有在阶段完成时才会更新价值估计和策略。具体来说，在每个阶段结束后，观察到的回报将用于策略评估，然后可以在该阶段访问的状态中改进策略。使用蒙特卡洛方法优化策略{v*}的过程可以直观地描述如下：

1) 使用 π 生成一集。2) 对于该集中访问的每个状态-动作对 (s, a)，计算并更新 $Q(s, a)$ 为平均回报。3) 对于该集中每个状态 s，更新策略：$\pi(s) \leftarrow argmax_{\mathbf{a}\in A} Q(s, \mathrm{a})$ 。

在将MC方法应用于实践中，我们可以在第1步中利用ε-greedy来平衡探索和利用之间的关系。此外，上述过程可以自然地与深度神经网络结合，从而得到深度蒙特卡洛（DMC）。这样，在第2步中，Q表$Q(s, a)$可以被神经网络替代，而神经网络可以通过均方误差（MSE）损失进行优化。

由于斗地主是一个典型的 episodic 任务，MC 自然适合解决这个问题。更重要的是，DMC 训练需要大量的经验，而生成这些经验相对容易。

data efficiently in parallel, which can also alleviate the issue of variance. In addition, adopting DMC in DouDizhu has some clear advantages compared to other reinforcement learning algorithms, such as policy gradient methods and deep Q-learning, which can be referred to in DouZero [20]. Owing to the advantages that DMC has in DouDizhu, DouZero adopts this algorithm and achieves an outstanding performance.

In the implementation of DouZero system, it makes use of a self-play procedure, where the actors play games to generate samples while the learner updates the network using these data. The input of the network consists of state features and action features. The state feature represents the information that is known to the player, while the action feature describes the legal move corresponding to the current state. Specifically, the action in action features is encoded with a one-hot 4×15 card matrix. For the state features, they contain card matrices that represent the hand cards, the union of other players' hand cards, the played cards of other players and the most recent moves and some one-hot vectors that represent that number of cards of other players, and the number of bombs played so far. For the architecture, a layer of LSTM is used to encode historical moves and the output is concatenated with other state/action features. There are six layers of MLP with a hidden size of 512 to produce Q values.

Besides, the system parallelizes DMC with multiple actor processes and one learner process. The learner maintains three global networks for the three positions and updates them to approximate the target values based on data samples generated by actor processes. Each actor maintains three local networks which are synchronized with the global networks periodically. The communication of the learner and actors is implemented with three shared experience buffers. In this way, the system can be trained in an effective self-play procedure.

IV. Method

In this section, we introduce opponent modeling and coach network in our design and describe how they are applied.

A. Opponent Modeling

Opponent modeling studies the problem of constructing models to make predictions about various properties of the modeled agents, *e.g.* actions, goals and so on. Classic methods such as policy reconstruction [38] and plan recognition [39] tend to develop parametric models for agent behaviours. These methods tend to decouple the interactions between the modeled agent and others to simplify the modeling process, which may introduce bias when there exists coupling between agent interactions. In this way, executing opponent modeling when concurrently training all the agents in a self-play procedure is more natural [40] and suitable to the training procedure of DouDizhu AI system. What's more, concurrent learning helps opponent modeling adapt to different levels of the agent as it has witnessed the evolution of the agent's skills during training.

When adopting opponent modeling in DouDizhu, we predict the hand of the player behind current agent so that the model can make decisions accordingly. As for the implementation of opponent modeling, we can naturally take advantage of deep neural networks to make predictions. To avoid confusion with the network that chooses which move to take, we call the part of opponent modeling as "prediction model" and the part that makes decisions as "decision model". Following the practice of DouZero that trains three models for the three players in the game, we also train three prediction models for opponent modeling. The prediction model can be viewed as a multi-head classifier and outputs the probability of the number of every kind of card in the hand of the next agent. To be specific, it has to predict how many Card 3, how many Card 4, *etc*, the next player has in his hand. Since the environment of DouDizhu is easy to realize, we can acquire the true hand of the next player and use it as labels to train the prediction model. What's more, taking Card 3 as an example, we can also know how many card 3 of one kind the next player has at most, which can be calculated by the agent's own hand and how many Card 3 has been played. We call this information "legal label" and this information can be utilized to help the training of prediction models as it can be used to filter out the wrong answers.

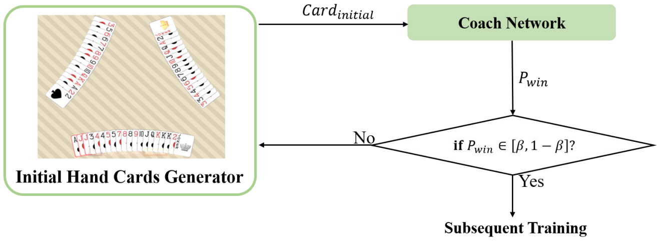

$Card_{initial}$
Coach Network
P_{win}
No
if $P_{win} \in [\beta, 1-\beta]$?
Yes
Initial Hand Cards Generator
Subsequent Training

Fig. 4: The overview of the framework that utilizes coach network. In this figure, we use the $Card_{initial}$, P_{win} and β to represent generated initial hand cards, the predicted probability of winning for Landlord and the threshold value, respectively. The coach network is composed of one embedding layer and several fully connected layers and the model takes $Card_{initial}$ as input and outputs P_{win}. If P_{win} is in the range defined, which is decided by β, the game with such $Card_{initial}$ will be carried on and generates samples for training. Otherwise, another initial hand cards will be generated.

As for the input of prediction models, we make use of the same state features as DouZero. The architecture of prediction models is also similar to DouZero with a layer of LSTM to encode historical moves and five shared layers of MLP. The final layer works as a multi-head classifier where each head corresponds to a fully connected layer and outputs the prediction of one kind of card. This model is trained using cross-entropy loss function. As for the decision model, the features used are also similar to DouZero, except for the prediction of hand cards of the next player in state information. For simplicity, we just concatenate the prediction results as well as original state features for state input of decision models. To sum up, the overview of the framework that combines opponent modeling with DouZero is shown in Figure 3.

数据高效并行处理，这也可以缓解方差问题。此外，与策略梯度方法和深度Q学习等其他强化学习算法相比， Dou Dizhu 中采用 DMC 有一些明显的优点，这些优点可以在 DouZero [20] 中参考。由于 DMC 在 DouDizhu 中的优势，DouZero 采用了该算法并取得了出色的性能。

在 DouZero 系统的实现中，它利用了一种自我对弈过程，其中演员们玩游戏生成样本，而学习者则使用这些数据更新网络。网络的输入包括状态特征和动作特征。状态特征表示玩家已知的信息，而动作特征描述了当前状态下合法的动作。具体来说，动作特征中的动作用一个一热编码的4×15张牌矩阵表示。对于状态特征，它们包含表示手牌的矩阵、其他玩家手牌的并集、其他玩家已打出的牌以及最近的动作和一些表示其他玩家手牌数量的一热向量和已打出的炸弹数量。在架构方面，使用了一层LSTM来编码历史动作，其输出与其他状态/动作特征进行了连接。有六层MLP，隐藏层大小为512，以生成Q值。

此外，该系统通过多个actor进程和一个learner进程并行化DMC。learner维护三个全局网络，用于三个位置，并根据actor进程生成的数据样本更新这些网络以近似目标值。每个actor维护三个局部网络，这些网络会周期性地与全局网络同步。learner和actors之间的通信通过三个共享的经验缓冲区实现。这样，系统可以在有效的自对弈训练过程中进行训练。

IV. 方法

在本节中，我们介绍了对手建模和教练网络在设计中的应用。

A. *Opponent Modeling*

对手建模研究了构建模型以预测所建模代理的各种属性、*e.g.*行为、目标等方面的问题。经典方法如策略重构[38]和计划识别[39]倾向于为代理行为开发参数模型。这些方法倾向于将所建模代理与其他代理之间的交互脱钩以简化建模过程，但在代理交互存在耦合时可能会引入偏差。因此，在自对弈过程中同时训练所有代理执行对手建模更为自然[40]，并且适合 DouDizhu AI 系统的训练过程。此外，同时学习有助于对手建模适应不同水平的代理，因为在训练过程中已经见证了代理技能的演变。

在采用对手建模的 DouDizhu 中，我们预测当前代理后面玩家的手牌，以便模型可以相应地做出决策。至于对手建模的实现，我们可以自然地利用深度神经网络进行预测。为了避免与选择行动的网络混淆，我们将对手建模的部分称为“预测模型”，而做出决策的部分称为“决策模型”。遵循DouZero在游戏中的三个玩家分别训练三个模型的做法，我们也为对手建模训练了三个预测模型。预测模型可以被视为一个多头分类器，并输出下一位玩家手中每种牌的数量的概率。具体来说，它必须预测下一位玩家手中有多少张Card 3，有多少张Card 4，*etc*。由于斗地主的环境易于实现，我们可以获得下一位玩家的真实手牌并将其作为标签来训练预测模型。此外，以Card 3为例，我们还可以知道下一位玩家最多有多少张同花色的Card 3，这可以通过玩家自己的手牌和已经打出的Card 3的数量来计算。我们将这种信息称为“合法标签”，这种信息可以用于帮助预测模型的训练，因为它可以用来过滤掉错误的答案。

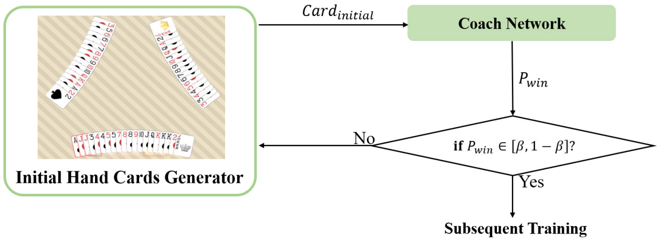

$Card_{initial}$
Coach Network
P_{win}
No
if $P_{win} \in [\beta, 1-\beta]$?
Yes
Initial Hand Cards Generator
Subsequent Training

图 4：利用教练网络的框架概述。在该图中，我们使用 $Card_{initial}$、P_{win} 和 β 分别表示生成的初始手牌、地主的获胜概率和阈值。教练网络由一个嵌入层和若干全连接层组成，模型以 $Card_{initial}$ 作为输入并输出 P_{win}。如果 P_{win} 在由 β 决定的范围内，将进行具有此类 $Card_{initial}$ 的游戏并生成用于训练的样本。否则，将生成另一组初始手牌。

对于预测模型的输入，我们使用与DouZero相同的状态特征。预测模型的架构也类似于DouZero，包含一层LSTM用于编码历史动作以及五层共享的MLP。最后一层作为多头分类器，其中每个头对应一个全连接层并输出一种牌的预测。该模型使用交叉熵损失函数进行训练。对于决策模型，使用的特征也类似于DouZero，除了状态信息中的下一位玩家的手牌预测。为了简化，我们只是将预测结果与原始状态特征进行拼接作为决策模型的状态输入。总之，结合对手建模与DouZero的框架概览如图3所示。

B. Coach-guided Learning

During the training of DouDizhu AI system, we discover that the training process costs a lot of time. To this end, we propose a method to help the agent master the skills faster. In this work, our DouDizhu AI system does not have a bidding phase as the bidding network in DouZero is trained with supervised learning. In other words, the initial hand cards of the three players are fixed at the beginning of the game. However, as a shedding-type game where the players' objective is to empty one's hand of all cards before others, the quality of the initial hand cards has a great impact on the result of this game. If one player gets a very strong hand at the beginning, he can win easily as long as he does not make serious mistakes. In this way, such initial cards are of little value for learning as they can hardly help the agent learn new knowledge. On the other hand, if one player always plays matches where the initial hand cards are relatively balanced, he can learn some skills faster and better as he will lose and receive a negative reward if he makes any unsuitable decision. In the setting of DouZero, we uncover that the initial cards of the three players are generated randomly so that quite a few samples may be not matched in strength. However, the actors still have to play the game using these initial cards that are heavily unbalanced, which also takes much time. If we only allow the actors to generate samples that are based on balanced initial hand cards, the agent can learn faster and form policies that can deal with such situation.

Based on the above discussion, we propose a coach network to identify whether the initial hand cards are balanced in strength. It takes the initial hand cards of the three players as input and outputs the predicted probability of winning for the Landlord in one game, which we call P_{win}. Then we can set a threshold, which is represented with β, to filter out the games whose P_{win} is too small or too big. In this case, there is no need for the actors to play with these initial hand cards, thus setting aside time to carry on more valuable matches.

The input of coach network is the vectors of initial hand cards for Landlord and Peasants, whose dimensions are 20 and 17, respectively. For the architecture of coach network, it consists of an embedding layer to process the input vectors and several layers of fully connected layers to extract representations and make predictions. As our DouDizhu AI system is trained in a self-play manner, the coach network is also concurrently trained with the decision models. The results of self-play games can be used as labels for training the model. Considering that the AI system learns from scratch, the threshold is set to 0 at first and increases through the training process. What's more, we only need to train one coach network for prediction as this module has nothing to do with the positions in DouDizhu. In other words, our coach network only works at the beginning of one game to pick suitable initial data and does not influence the subsequent processes. Therefore such idea can also be transferred into the development of other similar game AIs and benefits the training.

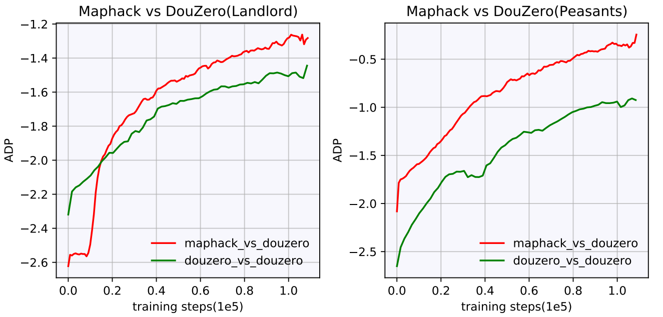


Fig. 5: ADP of "maphack" models, which can see the hand cards of the next player, and DouZero models. Both these models are tested with DouZero baseline that is trained with ADP. "Landlord" means that the models play as Landlord against Peasants of DouZero baseline and the same goes for the reverse.

V. Experiment

In this section, we conduct experiments to demonstrate the effectiveness of the improvement that we introduce to DouZero. To be specific, we first evaluate the performance of opponent modeling and coach network, respectively, and then combine them together. All experiments are conducted on a server with 4 Intel(R) Xeon(R) Gold 6252 CPU @ 2.10GHz and GeForce RTX 2080Ti GPU. Our codes are available at https://github.com/submit-paper/Doudizhu.

A. Experiment Settings

Exploitability is a commonly used measure of strategy strength in poker games [41]. However, the huge state and action space in DouDizhu make it intractable to calculate exploitability, not to mention that there are three players in this game, which brings more difficulty in evaluation. In order to evaluate the performance of the model, we launch tournaments that include the two opposite sides of Landlord and Peasants, following what DouZero [20]and Deltadou do [19]. To be specific, for two competing algorithms A and B, they will first play as Landlord and Peasants, respectively, for a given deck. Then we switch the sides, *i.e.* A takes Landlord position and B takes Peasants position, and they play the same deck again. To show the performance of the model in the training process, we execute the test for 10000 episodes every 30 minutes. As our DouDizhu AI is based on DouZero, we just compare the performance between them. We make use of the open-source models of DouZero as the opponent. To demonstrate the improvement, we also realize the original DouZero to intuitively exhibit the performance difference. As for the evaluation metrics, we also follow DouZero and use Winning Percentage (WP) and Average Difference in Points (ADP). Specifically, WP represents the number of games won by algorithm A divided by the total number of games. ADP indicates the average difference of points scored per game between algorithm A and B, where the base point is 1 and each bomb will double the score.

B. *Coach-guided Learning*

在训练 DouDizhu AI 系统的过程中，我们发现训练过程花费了很多时间。为此，我们提出了一种方法来帮助代理更快地掌握技能。在本工作中，我们的 DouDizhu AI 系统没有出价阶段，与 DouZero 中的出价网络通过监督学习训练不同，游戏开始时三位玩家的初始手牌是固定的。然而，作为一种甩牌型游戏，玩家的目标是在他人之前清空手中所有牌，初始手牌的质量对游戏结果有很大影响。如果一名玩家在开始时拿到非常强的手牌，只要不犯严重的错误，他就能轻易获胜。因此，这样的初始手牌对于学习来说几乎没有价值，因为它们几乎不能帮助代理学习新知识。另一方面，如果一名玩家总是玩初始手牌相对平衡的比赛，他可以更快更好地学习一些技能，因为他如果做出任何不合适的选择，就会输掉比赛并获得负奖励。在 DouZero 的设置中，我们发现三位玩家的初始手牌是随机生成的，因此许多样本在强度上可能不匹配。然而，演员们仍然必须使用这些初始手牌进行游戏，这些初始手牌严重不平衡，这也花费了很多时间。如果我们只允许演员生成基于平衡初始手牌的样本，代理可以更快地学习并形成能够处理这种情况的策略。

基于上述讨论，我们提出一个教练网络来识别初始手牌是否平衡。该网络以三位玩家的初始手牌作为输入，并输出该局游戏中地主获胜的预测概率，我们称之为P_{win}。然后我们可以设定一个阈值，用β表示，来过滤掉P_{win}太小或太大的游戏。在这种情况下，无需让演员使用这些初始手牌进行游戏，从而腾出时间进行更有价值的比赛。

教练网络的输入是地主和农民初始手牌的向量，维度分别为20和17。对于教练网络的架构，它包括一个嵌入层来处理输入向量，以及多层全连接层来提取表示和进行预测。由于我们的斗地主AI系统是通过自对弈方式进行训练的，因此教练网络也与决策模型同时进行训练。自对弈游戏的结果可以作为训练模型的标签。考虑到AI系统是从零开始学习的，初始阈值设置为0，并在训练过程中逐渐增加。此外，我们只需要训练一个教练网络来进行预测，因为这个模块与斗地主中的位置无关。换句话说，我们的教练网络仅在游戏开始时用于选择合适的初始数据，并不影响后续过程。因此，这种想法也可以应用于其他类似游戏AI的开发，并有利于训练。

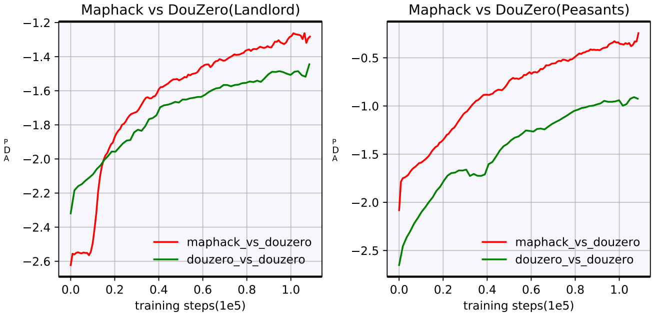


Fig. 5: ADP of “maphack”模型，这些模型可以查看下一位玩家的手牌，以及DouZero模型。这两种模型都使用了基于ADP训练的DouZero基线进行测试。“Landlord”表示这些模型以地主身份与DouZero基线中的农民对战，反之亦然。

V. 实验

在本节中，我们进行实验以展示我们对DouZero引入的改进的有效性。具体来说，我们首先分别评估对手建模和教练网络的性能，然后将它们结合起来。所有实验均在配备4块Intel(R) Xeon(R) Gold 6252 CPU @ 2.10GHz和GeForce RTX 2080Ti GPU的服务器上进行。我们的代码可在https://github.com/submit-paper/Doudizhu获取。

A. *Experiment Settings*

可利用性是扑克游戏中策略强度的一个常用度量标准 [41]。然而，在斗地主中巨大的状态和行动空间使得计算可利用性变得不可行，更不用说游戏中有三个玩家，这增加了评估的难度。为了评估模型的性能，我们按照DouZero [20] 和Deltadou [19] 的做法，启动了包括地主和农民两方的锦标赛。具体来说，对于两个竞争算法 A 和 B，它们将首先分别作为地主和农民，使用给定的牌组进行对战。然后我们交换位置，*i.e.* A 作为地主，B 作为农民，他们再次使用相同的牌组进行对战。为了展示模型在训练过程中的性能，我们每30分钟执行测试10000轮。由于我们的斗地主AI基于DouZero，我们仅比较它们之间的性能。我们利用开源的DouZero模型作为对手。为了展示改进，我们还实现了原始的DouZero，直观地展示性能差异。至于评估指标，我们也遵循DouZero，使用胜率（WP）和平均得分差（ADP）。具体来说，WP 表示算法 A 获胜的游戏数量除以总游戏数量。ADP 表示算法 A 和 B 每场比赛得分的平均差值，基分为1分，每枚炸弹将使得分翻倍。

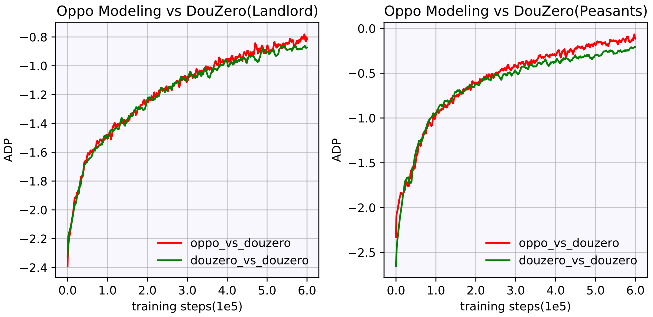


Fig. 6: ADP of models, which combine opponent modeling and DouZero, and DouZero models. Both these models are tested with DouZero baseline that is trained with ADP. “Landlord” means that the models play as Landlord against Peasants of DouZero baseline and the same goes for the reverse.

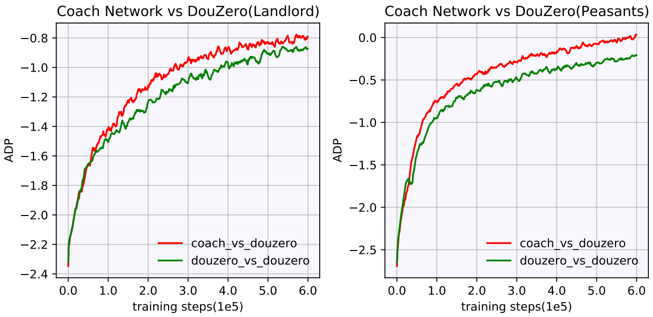


Fig. 7: ADP of models, which combine coach network with DouZero, and DouZero models. Both these models are tested with DouZero baseline that is trained with ADP. “Landlord” means that the models play as Landlord against Peasants of DouZero baseline and the same goes for the reverse.

Our implementation is based on DouZero and training schedules such as the number of actors and training hyper-parameters are kept the same as the default ones. As the DouDizhu environment is realized by ourselves, the reward also needs to be defined. The evaluation metrics of WP and ADP can be utilized when defining the reward. For WP, the agent winning a game is given +1 reward otherwise -1 reward while ADP can be directly used as rewards for ADP settings. DouZero provides two kinds of models which are trained using WP and ADP, respectively. For simplicity, we train our AI system with ADP as objective and compare its performance with the corresponding baseline. Also, we use the metric of ADP when evaluating the performance of the models.

B. Evaluation on Opponent Modeling

In this part, we demonstrate the effectiveness of introducing opponent modeling to DouDizhu. As the state features utilized by DouZero contain all the information that can be known, the information about the hand cards of the next player is included implicitly while the idea of opponent modeling is essentially making such information explicit. In order to investigate whether such an idea helps the agents learn better, we firstly make a pre-experiment where we add the hand cards of the next player into state features directly, whose result is shown in Figure 5. It can be observed that adding the hand cards of the next player into state features indeed boosts the performances of the agents, especially for Peasants. We assume that the obvious improvement of Peasants is attributed to the fact that knowing the hand cards of the next player helps Peasants not only choose cards that the Landlord can’t afford but also cooperate with the teammate better. Whereas for the Landlord, knowing the hand cards of next player indeed helps to make decisions, but if the hand is weak, even having such information can not help a lot. To sum up, the result of the pre-experiment illustrate that introducing explicit representations of the next player’s hand cards improves the performance of DouDizhu AI.

After verifying the validity of our idea, we concurrently train the prediction models as well as the decision models as is discussed in Section IV-A and the result is shown in Figure 6. It reveals that introducing opponent modeling to DouZero mainly improves the performance of models of Peasants, which is corresponding to the analysis above. Although the models perform worse than DouZero at first because the network has to take more features as input and has more neurons, which will slow down learning, they manage to grasp more knowledge after enough training and achieve a performance better than DouZero.

C. Evaluation on Coach Network

Apart from the experiments above, we also conduct experiments to show how coach network” performs in DouDizhu game. The training procedure is discussed in Section IV-B and the upper limit of threshold β is set to be 0.3. The result of the experiment is shown in Figure 7 and the significant improvement proves the effectiveness of this method. It can be observed that the improvement of Peasants is also greater than that of Landlord. Considering that Peasants have an advantage in this game due to cooperation, this phenomenon is acceptable as they can learn more skills in balanced games. Besides, this coach-guided learning strategy just controls the initial state of the game while the results demonstrate the significant improvement it can bring. This fact reveals that the luck factor plays an important role in such kind of imperfect-information games. In other words, our method can be migrated into other environments, helping game AI achieve better performance.

What’s more, we also show some cases about the prediction of our coach network from games on Botzone platform, which is illustrated in Table I. In case 1, it can be observed that the Landlord is allocated with a very strong hand, which consists of most cards of high rank and cards of low rank that can compose other combinations so that the Landlord can win the game easily. As for case 2, even Landlord has a bomb in his hand, the hand cards of Peasants are also very strong. What’s worse, the Landlord also has quite a few cards of low rank

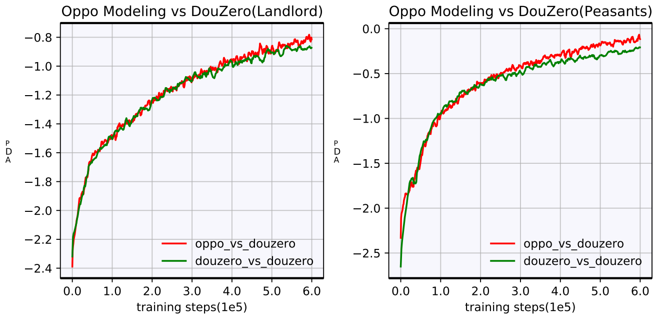


Fig. 6: ADP of 模型，结合了对手建模和DouZero，以及DouZero模型。这两种模型都使用了基于ADP训练的DouZero基线进行测试。“地主”意味着这些模型作为地主与DouZero基线中的农民对战，反之亦然。

我们的实现基于DouZero，训练计划如演员的数量和训练超参数等保持与默认值一致。由于DouDizhu环境是自行实现的，因此奖励也需要定义。在定义奖励时，可以利用WP和ADP的评估指标。对于WP，当智能体赢得一局游戏时，给定+1奖励，否则给定-1奖励；而ADP可以直接作为ADP设置的奖励。DouZero提供了两种模型，分别使用WP和ADP进行训练。为了简化起见，我们以ADP为目标训练我们的AI系统，并将其性能与相应的基线进行比较。此外，在评估模型性能时，我们使用ADP作为指标。

B. *Evaluation on Opponent Modeling*

在这一部分，我们展示了引入对手建模到斗地主中的有效性。由于DouZero所利用的状态特征包含了所有可知的信息，下一个玩家的手牌信息虽然隐含在其中，但对手建模的核心思想是使这些信息变得明确。为了调查这种思想是否有助于智能体更好地学习，我们首先进行了一次预实验，在该实验中，我们将下一个玩家的手牌直接添加到状态特征中，其结果如图5所示。可以观察到，将下一个玩家的手牌添加到状态特征中确实提升了智能体的表现，尤其是对于农民。我们认为农民表现明显提升的原因在于知道下一个玩家的手牌不仅帮助农民选择地主无法承担的牌，还能够更好地与队友合作。而对于地主来说，知道下一个玩家的手牌确实有助于决策，但如果手牌较弱，即使有这些信息也帮助不大。总之，预实验的结果表明引入下一个玩家手牌的明确表示可以提升斗地主AI的表现。

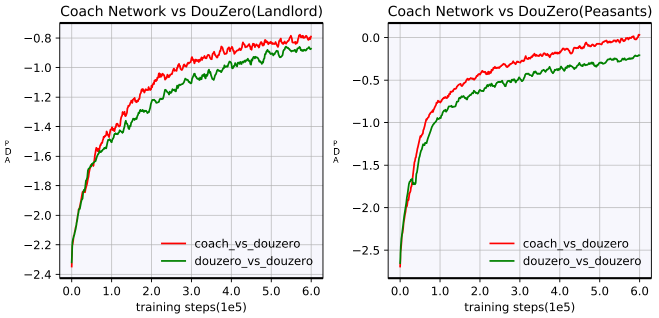


Fig. 7: ADP模型，结合了教练网络与DouZero的模型，以及DouZero模型。这些模型都使用了以ADP训练的DouZero基线进行测试。“地主”表示这些模型以地主身份与DouZero基线中的农民对战，反之亦然。

在验证了我们的想法之后，我们同时训练了预测模型和决策模型，如第四节A部分所述，结果如图6所示。这表明将对手建模引入DouZero主要提高了农民模型的性能，这与上述分析相符。尽管这些模型在最初的表现不如DouZero，因为网络需要输入更多的特征并且有更多的神经元，这会减慢学习速度，但在充分训练后，它们能够掌握更多的知识，并最终达到优于DouZero的性能。

C. *Evaluation on Coach Network*

除了上述实验外，我们还进行了实验以展示“教练网络”在斗地主游戏中的表现。训练过程在第四节B部分讨论，并将阈值 β 的上限设置为0.3。实验结果如图7所示，显著的改进证明了该方法的有效性。可以观察到，农民的改进幅度也大于地主。考虑到农民在这个游戏中由于合作而具有优势，这种现象是可以接受的，因为他们可以在平衡游戏中学到更多的技能。此外，这种教练引导的学习策略只是控制游戏的初始状态，而结果表明它能带来的显著改进。这一事实揭示了在这样的不完美信息游戏中，运气因素起着重要作用。换句话说，我们的方法可以迁移到其他环境中，帮助游戏AI取得更好的性能。

此外，我们还展示了我们在Botzone平台上的一些比赛关于教练网络预测的情况，这些情况在表I中有所说明。在案例1中，可以观察到地主被分配了一个非常强的手牌，其中包括大部分高牌和可以组成其他组合的低牌，使得地主很容易赢得游戏。至于案例2，即使地主手中有一张炸弹，农民的手牌也非常强。更糟糕的是，地主还持有相当多的低牌。

	Landlord	Landlord_down	Landlord_up	Prediction of P_{win} for Landlord	Actual result(Landlord)
Case1	3455677789JQKAAAA22R	334569TTTJJQQQKK2	344566788899TJK2B	0.9932	Win
Case2	45667788889TTTKKA22B	334567TJJJQQQQK22	33445567999JKAAAR	0.1726	Lose
Case3	3455556677799JJQKAAB	3467889TTQKKK222R	33446889TTJJQQAA2	0.5843	Lose

TABLE I: Case study to show the effect of "coach network". It predicts the winning probability of Landlord based on the initial hand cards of the three players. We pick some cases from games from Botzone to show the predicted results of "coach network" and also show the actual result from the view of the Landlord. To be mentioned, T means 10, J means Jack, Q means Queen, K means King, A means Ace, B means Black Joker, and R means Red Joker.

that are difficult to play out. In case 3, the initial hand cards are relatively balanced. However, the Peasant win the game finally, indicating the importance of cooperation. This example illustrates that the balanced samples can indeed help the agents learn cautious policy and cooperation better, thus proving the correctness of our idea.

D. Combination of Two Methods

From the above discussion, it is known that both our improvements can help enhance the performance of DouZero. The result of combining these two methods is shown in Figure 8. As the improvement of "coach network" is more obvious than opponent modeling, to intuitively demonstrate whether the combination of these two techniques brings further improvement, we also add the result of just using "coach network" in the figure. It can be observed the performance is a little worse than just using coach network at first, which is consistent with the discussion of just introducing opponent modeling. To be mentioned, when the performance of the models reaches a certain level, achieving a little improvement is very difficult so the progress that combining the two methods makes is not that apparent. However, further improvement still proves the effectiveness of combination of the two methods.

To comprehensively compare the performance of our DouDizhu AI, we upload our final model to BotZone [23], an online platform with DouDizhu competition. This platform supports more than 20 games apart from DouDizhu, including Go, Mahjong, Chess and so on. There are more than 3500 users on this platform uploading their bot programs to compete with other bots in a selected game. Botzone maintains a leaderboard for each game, which ranks all the bots in the Botzone Elo system by their Elo rating scores. In the Botzone Elo of DouDizhu (named "FightTheLandlord" on the platform), each game is played by two bots, with one acting as the Landlord and the other as Peasants. A pair of games are played simultaneously where the two bots play different roles and the initial hand cards also keep unchanged. Although Elo rating is generally considered as a stable measurement of relative strength, DouDizhu Elo ranking on Botzone suffers from some fluidity due to the nature of high variance of this game. What's more, due to the limit of server resources, Elo rating games are not scheduled very frequently. One bot plays less than 10 Elo rating games on average every day so that it may take a lot of time to achieve a stable ranking. However, keeping a high ranking can still prove the strength of one AI system. Even if DouZero has obvious superiority over other DouDizhu AI systems trained by reinforcement learning, it has ranked about 20th so far on Botzone leaderboard as most bots are realized by strong heuristic rules. Nonetheless, our DouDizhu AI has always ranked top five, even ranked first for several months, proving the effectiveness of the improvements that we make.

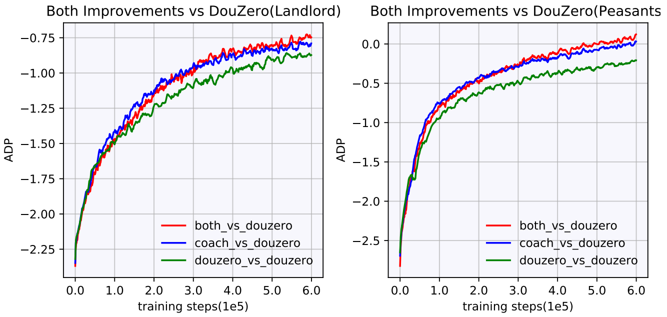


Fig. 8: ADP of models, which combine both improvements with DouZero, and DouZero models. Both these models are tested with DouZero baseline that is trained with ADP. "Landlord" means that the models play as Landlord against Peasants of DouZero baseline and the same goes for the reverse. For comparison, the result of models improved by "coach network" is also included.

VI. Conclusion and Future Work

In this work, we put forward some improvements to the state-of-the-art DouDizhu AI program, DouZero. Inspired by the human player's prediction about opponents' hand cards in practice, we introduce opponent modeling. Based on the nature of high variance of this game, we originally propose a "coach network" to pick valuable samples to accelerate the training. The outstanding performance of our AI on the Botzone platform proves the effectiveness of our improvement.

Although our DouDizhu AI performs well after adopting these techniques, there is still room for improvement. First, to better show the effect of our improvement, we do not make changes on the architectures of neural networks in DouZero unless necessary. We plan to try other neural networks such as convolutional neural networks like ResNet [42]. Second, we find that there are still some cases where the model can not make good decisions. We hope to combine search with our AI to enhance the performance as search plays an

	Landlord	Landlord_down	Landlord_up	Prediction of P_{win} for Landlord	Actual result(Landlord)
Case1	3455677789JQKAAAA22R	334569TTTJJQQQKK2	344566788899TJK2B	0.9932	Win
Case2	45667788889TTTKKA22B	334567TJJJQQQQK22	33445567999JKAAAR	0.1726	Lose
Case3	3455556677799JJQKAAB	3467889TTQKKK222R	33446889TTJJQQAA2	0.5843	Lose

TABLE I: 案例研究，展示“教练网络”的效果。它基于三名玩家的起始手牌预测房东获胜的概率。我们从Botzone的游戏选取一些案例，展示“教练网络”的预测结果，并从房东的角度展示实际结果。需要注意的是，T 表示10，J 表示杰克，Q 表示皇后，K 表示国王，A 表示 Ace，B 表示黑王，R 表示红王。

这些难以上演。在第3种情况下，初始的手牌相对平衡。然而，农民最终赢得了游戏，表明合作的重要性。这个例子说明了平衡的数据集确实可以帮助智能体学习更加谨慎的策略和更好的合作，从而证明了我们想法的正确性。

D. Combination of Two Methods

从上述讨论可知，我们的改进都可以帮助提升 DouZero 的性能。将这两种方法结合起来的结果如图 8 所示。由于“教练网络”的改进比对手建模更为明显，为了直观地展示这两种技术的结合是否带来了进一步的改进，我们还在图中增加了仅使用“教练网络”的结果。可以观察到，性能在一开始略差于仅使用教练网络，这与仅引入对手建模的讨论一致。值得一提的是，当模型的性能达到一定水平时，取得一点点改进是非常困难的，因此两种方法结合所取得的进步并不明显。然而，进一步的改进仍然证明了这两种方法结合的有效性。

为了全面比较我们的DouDizhu AI的表现，我们将最终模型上传到BotZone [23]，这是一个支持DouDizhu比赛的在线平台。该平台除了支持DouDizhu之外，还支持超过20种游戏，包括围棋、麻将、国际象棋等。该平台上超过3500名用户上传他们的机器人程序，与其他机器人在选定的游戏比赛中竞争。Botzone为每种游戏维护一个排行榜，根据Elo评分排名所有机器人。在Botzone的DouDizhu Elo（平台上的名称为“FightTheLandlord”）中，每场比赛由两台机器人进行，其中一台扮演地主，另一台扮演农民。同时进行两局比赛，两台机器人分别扮演不同的角色，初始手牌也保持不变。虽然Elo评分通常被认为是一个相对稳定的强度衡量标准，但由于这款游戏高方差的性质，Botzone上的DouDizhu Elo排名会受到一些波动。此外，由于服务器资源的限制，Elo评分比赛并不经常安排。一台机器人平均每天参加不到10场Elo评分比赛，因此达到稳定排名可能需要很长时间。然而，保持高排名仍然可以证明一个AI的实力。

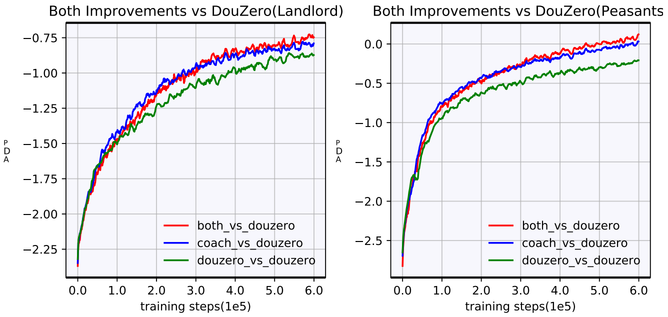


Fig. 8: ADP of 模型，这些模型结合了对 DouZero 的两种改进，以及 DouZero 模型。这些模型都使用了基于 ADP 训练的 DouZero 基准进行测试。“地主”意味着这些模型以地主的身份与 DouZero 基准中的农民对战，反之亦然。为了对比，使用“教练网络”改进的模型的结果也包括在内。

系统。即使DouZero在 reinforcement learning 训练的其他DouDizhu AI系统中表现出明显的优越性，但在Botzone排行榜上它目前仅排名约为第20位，因为大多数机器人是通过强大的启发式规则实现的。然而，我们的DouDizhu AI一直排名前五，甚至有几个月排名第一，这证明了我们所做的改进的有效性。

VI. 结论与未来工作

在本工作中，我们对当前最先进的斗地主AI程序DouZero提出了一些改进。受实际中人类玩家对手牌预测的启发，我们引入了对手建模。鉴于此游戏高方差的性质，我们最初提出了一种“教练网络”来挑选有价值的样本以加速训练。我们的AI在Botzone平台上的出色表现证明了我们改进的有效性。

尽管采用了这些技术，我们的斗地主AI的表现已经不错，但仍有很多改进的空间。首先，为了更好地展示我们改进的效果，除非必要，我们在DouZero中不会对神经网络架构进行修改。我们计划尝试其他类型的神经网络，如ResNet等卷积神经网络[42]。其次，我们发现模型在某些情况下仍然无法做出好的决策。我们希望将搜索技术与AI结合以提升性能，因为搜索在这一过程中发挥着重要作用。

important role and performs well in research about game AI [43], [44]. Finally, we will investigate how to improve the sample efficiency with experiment replay [45] as it still costs a lot of time even utilizing our "coach network". In addition, we will also try to transfer our methods to other games for stronger game AIs.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[3] T. Cazenave, "Improving model and search for computer go," in *IEEE Conference on Games (CoG)*, 2021, pp. 1–8.

[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[5] D.-W. Kim, S. Park, and S.-i. Yang, "Mastering fighting game using deep reinforcement learning with self-play," in *IEEE Conference on Games (CoG)*, 2020, pp. 576–583.

[6] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione, "Regret minimization in games with incomplete information," *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 20, pp. 1729–1736, 2007.

[7] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *arXiv preprint arXiv:1603.01121*, 2016.

[8] M. Moravčík, M. Schmid, N. Burch, V. Lisỳ, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, and M. Bowling, "Deepstack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[9] N. Brown and T. Sandholm, "Superhuman ai for heads-up no-limit poker: Libratus beats top professionals," *Science*, vol. 359, no. 6374, pp. 418–424, 2018.

[10] ——, "Superhuman ai for multiplayer poker," *Science*, vol. 365, no. 6456, pp. 885–890, 2019.

[11] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft II using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[12] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, "Dota 2 with large scale deep reinforcement learning," *arXiv preprint arXiv:1912.06680*, 2019.

[13] J. Li, S. Koyamada, Q. Ye, G. Liu, C. Wang, R. Yang, L. Zhao, T. Qin, T.-Y. Liu, and H.-W. Hon, "Suphx: Mastering mahjong with deep reinforcement learning," *arXiv preprint arXiv:2003.13590*, 2020.

[14] T. W. Neller and M. Lanctot, "An introduction to counterfactual regret minimization," in *Educational Advances in Artificial Intelligence (EAAI)*, vol. 11, 2013.

[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[16] T. Zahavy, M. Haroush, N. Merlis, D. J. Mankowitz, and S. Mannor, "Learn what not to learn: Action elimination with deep reinforcement learning," *arXiv preprint arXiv:1809.02121*, 2018.

[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2016, pp. 1928–1937.

[18] Y. You, L. Li, B. Guo, W. Wang, and C. Lu, "Combinatorial q-learning for dou di zhu," in *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 16, no. 1, 2020, pp. 301–307.

[19] Q. Jiang, K. Li, B. Du, H. Chen, and H. Fang, "Deltadou: Expert-level doudizhu ai through self-play." in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2019, pp. 1265–1271.

[20] D. Zha, J. Xie, W. Ma, S. Zhang, X. Lian, X. Hu, and J. Liu, "Douzero: Mastering doudizhu with self-play deep reinforcement learning," *arXiv preprint arXiv:2106.06135*, 2021.

[21] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[22] H. Zhou, Y. Zhou, H. Zhang, H. Huang, and W. Li, "Botzone: A competitive and interactive platform for game ai education," in *Proceedings of the ACM Turing 50th Celebration Conference-China*, 2017, pp. 1–5.

[23] H. Zhou, H. Zhang, Y. Zhou, X. Wang, and W. Li, "Botzone: an online multi-agent competitive platform for ai education," in *ACM Conference on Innovation and Technology in Computer Science Education*, 2018, pp. 33–38.

[24] H. Zhang, G. Gao, W. Li, C. Zhong, W. Yu, and C. Wang, "Botzone: A game playing system for artificial intelligence education," in *International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 2012, p. 1.

[25] N. Sweeney and D. Sinclair, "Applying reinforcement learning to poker," in *Computer Poker Symposium*, 2012.

[26] M. Lanctot, V. Zambaldi, A. Gruslys, A. Lazaridou, K. Tuyls, J. Pérolat, D. Silver, and T. Graepel, "A unified game-theoretic approach to multiagent reinforcement learning," *arXiv preprint arXiv:1711.00832*, 2017.

[27] D. Ye, Z. Liu, M. Sun, B. Shi, P. Zhao, H. Wu, H. Yu, S. Yang, X. Wu, Q. Guo *et al.*, "Mastering complex control in moba games with deep reinforcement learning." in *AAAI Conference on Artificial Intelligence (AAAI)*, 2020.

[28] N. Brown, A. Bakhtin, A. Lerer, and Q. Gong, "Combining deep reinforcement learning and search for imperfect-information games," *arXiv preprint arXiv:2007.13544*, 2020.

[29] M. Gedda, M. Z. Lagerkvist, and M. Butler, "Monte carlo methods for the game kingdomino," in *IEEE Conference on Computational Intelligence and Games (CIG)*, 2018, pp. 1–8.

[30] J. Zhou, "Design and application of tibetan long chess using monte carlo algorithm and artificial intelligence," in *Journal of Physics: Conference Series*, vol. 1952, no. 4, 2021, p. 042104.

[31] F. Southey, M. P. Bowling, B. Larson, C. Piccione, N. Burch, D. Billings, and C. Rayner, "Bayes' bluff: Opponent modelling in poker," *arXiv preprint arXiv:1207.1411*, 2012.

[32] N. Mizukami and Y. Tsuruoka, "Building a computer mahjong player based on monte carlo simulation and opponent models," in *IEEE Conference on Computational Intelligence and Games (CIG)*, 2015, pp. 275–283.

[33] F. Schadd, S. Bakkes, and P. Spronck, "Opponent modeling in real-time strategy games." in *GAMEON*, 2007, pp. 61–70.

[34] H. He, J. Boyd-Graber, K. Kwok, and H. Daumé III, "Opponent modeling in deep reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2016, pp. 1804–1813.

[35] S. J. Knegt, M. M. Drugan, and M. A. Wiering, "Opponent modelling in the game of tron using reinforcement learning." in *International Conference on Agents and Artificial Intelligence (ICAART)*, 2018, pp. 29–40.

[36] J. N. Foerster, R. Y. Chen, M. Al-Shedivat, S. Whiteson, P. Abbeel, and I. Mordatch, "Learning with opponent-learning awareness," *arXiv preprint arXiv:1709.04326*, 2017.

[37] L. F. Teófilo, N. Passos, L. P. Reis, and H. L. Cardoso, "Adapting strategies to opponent models in incomplete information games: a reinforcement learning approach for poker," in *International Conference on Autonomous and Intelligent Systems*, 2012, pp. 220–227.

[38] D. Carmel and S. Markovitch, "Model-based learning of interaction strategies in multi-agent systems," *Journal of Experimental & Theoretical Artificial Intelligence*, vol. 10, no. 3, pp. 309–332, 1998.

[39] M. Fagan and P. Cunningham, "Case-based plan recognition in computer games," in *International Conference on Case-Based Reasoning*, 2003, pp. 161–170.

[40] T. Bansal, J. Pachocki, S. Sidor, I. Sutskever, and I. Mordatch, "Emergent complexity via multi-agent competition," *arXiv preprint arXiv:1710.03748*, 2017.

[41] M. Johanson, K. Waugh, M. Bowling, and M. Zinkevich, "Accelerating best response calculation in large extensive games," in *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2011.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

扮演着重要角色并在关于游戏AI的研究中表现良好 [43], [44]。最后，我们将研究如何通过实验重放 [45] 来提高样本效率，即使利用我们的“教练网络”，这仍然需要大量时间。此外，我们还将尝试将我们的方法应用于其他游戏以获得更强的游戏AI。

REFERENCES参考文献

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, “使用深度神经网络和树搜索掌握围棋游戏,” *Nature*, 第529卷, 第7587期, 第484–489页, 2016年。[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton *et al.*, “无需人类知识掌握围棋游戏,” *Nature*, 第550卷, 第7676期, 第354–359页, 2017年。[3] T. Cazenave, “提高计算机围棋的模型和搜索,” 在 *IEEE Conference on Games (CoG)*, 第1–8页, 2021年。[4] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel *et al.*, “通过自我对弈掌握国际象棋、将棋和围棋的一般强化学习算法,” *Science*, 第362卷, 第6419期, 第1140–1144页, 2018年。[5] D.-W. Kim, S. Park, 和 S.-i. Yang, “使用自我对弈的深度强化学习掌握格斗游戏,” 在 *IEEE Conference on Games (CoG)*, 第576–583页, 2020年。[6] M. Zinkevich, M. Johanson, M. Bowling, 和 C. Piccione, “在不完整信息游戏中最小化遗憾,” *Advances in Neural Information Processing Systems (NeurIPS)*, 第20卷, 第1729–1736页, 2007年。[7] J. Heinrich 和 D. Silver, “在不完整信息游戏中从自我对弈的深度强化学习,” *arXiv preprint arXiv:1603.01121*, 2016年。[8] M. Morav ík, M. Schmid, N. Burch, V. Lisý, D. Morrill, N. Bard, T. Davis, K. Waugh, M. Johanson, 和 M. Bowling, “Deepstack: 头对头无限注扑克的专家级人工智能,” *Science*, 第356卷, 第6337期, 第508–513页, 2017年。[9] N. Brown 和 T. Sandholm, “超人类AI在头对头无限注扑克中的胜利: Libratus击败顶级职业选手,” *Science*, 第359卷, 第6374期, 第418–424页, 2018年。[10] ——, “多玩家扑克中的超人类AI,” *Science*, 第365卷, 第6456期, 第885–890页, 2019年。[11] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, “使用多智能体强化学习在星际争霸II中达到大师级水平,” *Nature*, 第575卷, 第7782期, 第350–354页, 2019年。[12] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse *et al.*, “使用大规模深度强化学习的Dota 2,” *arXiv preprint arXiv:1912.06680*, 2019年。[13] J. Li, S. Koyamada, Q. Ye, G. Liu, C. Wang, R. Yang, L. Zhao, T. Qin, T.-Y. Liu, 和 H.-W. Hon, “使用深度强化学习掌握麻将的Suphx,” *arXiv preprint arXiv:2003.13590*, 2020年。[14] T. W. Neller 和 M. Lanctot, “反事实遗憾最小化简介,” 在 *Educational Advances in Artificial Intelligence (EAAI)*, 第11卷, 2013年。[15] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, “通过深度强化学习达到人类水平的控制,” *Nature*, 第518卷, 第7540期, 第529–533页, 2015年。[16] T. Zahavy, M. Haroush, N. Merlis, D. J. Mankowitz, 和 S. Mannor, “学会不学什么: 深度强化学习中的动作消除,” *arXiv preprint arXiv:1809.02121*, 2018年。[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, 和 K. Kavukcuoglu, “异步方法在深度强化学习中的应用,” 在 *International Conference on Machine Learning (ICML)*, 第16卷, 第1期, 第301–307页, 2020年。[18] Y. You, L. Li, B. Guo, W. Wang, 和 C. Lu, “组合Q学习在斗地主中的应用,” 在 *AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 第16卷, 第1期, 第301–307页, 2020年。[19] Q. Jiang, K. Li, B. Du, H. Chen, 和 H. Fang, “Deltadou: 通过自我对弈掌握斗地主的专家级AI,” 在 *International Joint Conferences on Artificial Intelligence (IJCAI)*, 第1265–1271页, 2019年。

[20] D. 乍, J. 谢, W. 马, S. 张, X. 莲, X. 胡, 和 J. 刘, “Douzero: 使用自玩深度强化学习掌握斗地主,” *arXiv preprint arXiv:2106.06135*, 2021。[21] R. S. 斯图尔特和 A. G. 巴尔托, *Reinforcement learning: An introduction*。MIT印刷, 2018。[22] H. 周, Y. 周, H. 张, H. 黄, 和 W. 李, “Botzone: 游戏AI教育的竞赛和互动平台,” 在 *Proceedings of the ACM Turing 50th Celebration Conference-China*, 2017, 页码. 1-5。[23] H. 周, H. 张, Y. 周, X. 王, 和 W. 李, “Botzone: 用于AI教育的在线多智能体竞赛平台,” 在 *ACM Conference on Innovation and Technology in Computer Science Education*, 2018, 页码. 33-38。[24] H. 张, G. 高, W. 李, C. 中, W. 于, 和 C. 王, “Botzone: 用于人工智能教育的游戏玩系统,” 在 *International Conference on Frontiers in Education: Computer Science and Computer Engineering (FECS)*, 2012, 页码. 1。[25] N. 西文和 D. 西尔文, “将强化学习应用于扑克,” 在 *Computer Poker Symposium*, 2012。[26] M. 拉纳科特, V. 扎姆巴迪, A. 格鲁斯利斯, A. 拉扎里多, K. 图伊尔斯, J. P´埃罗拉特, D. 西尔弗, 和 T. 格雷佩尔, “统一博弈论方法的多智能体强化学习,” *arXiv preprint arXiv:1711.00832*, 2017。[27] D. 叶, Z. 刘, M. 孙, B. 时, P. 赵, H. 吴, H. 于, S. 杨, X. 吴, Q. 郭 *et al.*, “使用深度强化学习在MOBA游戏中掌握复杂控制。” 在 *AAAI Conference on Artificial Intelligence (AAAI)*, 2020。[28] N. 布朗, A. 巴克廷, A. 莱尔, 和 Q. 宫, “结合深度强化学习和搜索的不完美信息博弈,” *arXiv preprint arXiv:2007.13544*, 2020。[29] M. 吉达, M. Z. 拉格克维斯特, 和 M. 布特勒, “王国ino的游戏蒙特卡洛方法,” 在 *IEEE Conference on Computational Intelligence and Games (CIG)*, 2018, 页码. 1-8。[30] J. 周, “使用蒙特卡洛算法和人工智能设计和应用藏族长棋,” 在 *Journal of Physics: Conference Series*, 卷. 1952, 期. 4, 2021, 页码. 042104。[31] F. 南希, M. P. 道林, B. 拉森, C. 比乔内, N. 布尔奇, D. 布林宁, 和 C. 雷纳, “贝叶斯欺骗: 棋牌中的对手建模,” *arXiv preprint arXiv:1207.1411*, 2012。[32] N. 水木和 Y. 辛, “基于蒙特卡洛模拟和对手模型构建计算机麻将玩家,” 在 *IEEE Conference on Computational Intelligence and Games (CIG)*, 2015, 页码. 275-283。[33] F. 施达德, S. 巴克克斯, 和 P. 施普龙克, “实时战略游戏中对手建模。” 在 *GAMEON*, 2007, 页码. 61-70。[34] H. 何, J. 布莱德-格雷伯, K. 柯克, 和 H. 大姆三世, “深度强化学习中的对手建模,” 在 *International Conference on Machine Learning (ICML)*, 2016, 页码. 1804-1813。[35] S. J. 克涅格特, M. M. 德鲁甘, 和 M. A. 维里宁, “使用强化学习在游戏tron中的对手建模。” 在 *International Conference on Agents and Artificial Intelligence (ICAART)*, 2018, 页码. 29-40。[36] J. N. 福斯特, R. Y. 陈, M. 阿尔-谢迪夫特, S. 吉斯蒙, P. 阿比布, 和 I. 摩尔达奇, “对手学习意识下的学习,” *arXiv preprint arXiv:1709.04326*, 2017。[37] L. F. 特奥菲洛, N. 巴索斯, L. P. 雷斯, 和 H. L. 卡尔达索, “根据对手模型调整策略: 不完整信息博弈中的强化学习方法,” 在 *International Conference on Autonomous and Intelligent Systems*, 2012, 页码. 220-227。[38] D. 卡尔梅尔和 S. 马尔科维奇, “多智能体系统中的基于模型的交互策略学习,” *Journal of Experimental & Theo- retical Artificial Intelligence*, 卷. 10, 期. 3, 页码. 309-332, 1998。[39] M. 菲根和 P. 克努森, “计算机游戏中的案例计划识别,” 在 *International Conference on Case-Based Reasoning*, 2003, 页码. 161-170。[40] T. 巴恩斯, J. 帕乔基, S. 西多尔, I. 苏斯克, 和 I. 摩尔达奇, “多智能体竞争中的自发复杂性,” *arXiv preprint arXiv:1710.03748*, 2017。[41] M. 福根, K. 韦, M. 道林, 和 M. 金凯维奇, “在大型扩展博弈中加速最佳响应计算,” 在 *International Joint Conferences on Artificial Intelligence (IJCAI)*, 2011。[42] K. 何, X. 张, S. 任, 和 J. 孙, “深度残差学习在图像识别中的应用,” 在 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, 页码. 770-778。

[43] B. Bouzy, A. Rimbaud, and V. Ventos, “Recursive monte carlo search for bridge card play,” in *IEEE Conference on Games (CoG)*, 2020, pp. 229–236.

[44] S. Ariyurek, A. Betin-Can, and E. Surer, “Enhancing the monte carlo tree search algorithm for video game testing,” in *IEEE Conference on Games (CoG)*, 2020, pp. 25–32.

[45] S. Zhang and R. S. Sutton, “A deeper look at experience replay,” *arXiv preprint arXiv:1712.01275*, 2017.

[43] B. Bouzy, A. Rimbaud, 和 V. Ventos, “桥牌牌局的递归蒙特卡洛搜索,” 于 *IEEE Conference on Games (CoG)*，2020，第229-236页。[44] S. Ariyurek, A. Betin-Can, 和 E. Surer, “增强用于视频游戏测试的蒙特卡洛树搜索算法,” 于 *IEEE Conference on Games (CoG)*，2020，第25-32页。[45] S. Zhang 和 R. S. Sutton, “经验回放的更深入研究,” 于 *arXiv preprint arXiv:1712.01275*，2017。