doi: 10.13413/j.cnki.jdxblxb.2023555

基于深度强化学习的游戏智能引导算法

白 天[1], 吕璐瑶[2], 李 储[1], 何加亮[3]
(1. 吉林大学 计算机科学与技术学院, 长春 130012; 2. 吉林大学 软件学院, 长春 130012;
3. 大连民族大学 信息与通信工程学院, 辽宁 大连 116600)

摘要: 针对传统游戏智能体算法存在模型输入维度大及训练时间长的问题, 提出一种结合状态信息转换与奖励函数塑形技术的新型深度强化学习游戏智能引导算法. 首先, 利用Unity引擎提供的接口直接读取游戏后台信息, 以有效压缩状态空间的维度, 减少输入数据量; 其次, 通过精细化设计奖励机制, 加速模型的收敛过程; 最后, 从主观定性和客观定量两方面对该算法模型与现有方法进行对比实验, 实验结果表明, 该算法不仅显著提高了模型的训练效率, 还大幅度提高了智能体的性能.

关键词: 深度强化学习; 游戏智能体; 奖励函数塑形; 近端策略优化算法
中图分类号: TP391 **文献标志码**: A **文章编号**: 1671-5489(2025)01-0091-08

Game Intelligent Guidance Algorithm Based on Deep Reinforcement Learning

BAI Tian[1], LÜ Luyao[2], LI Chu[1], HE Jialiang[3]
(1. *College of Computer Science and Technology, Jilin University, Changchun* 130012, *China*;
2. *College of Software, Jilin University, Changchun* 130012, *China*; 3. *College of Information and Communication Engineering, Dalian Minzu University, Dalian* 116600, *Liaoning Province, China*)

Abstract: Aiming at the problems of high input dimensionality and long training time in traditional game intelligent algorithm models, we proposed a novel deep reinforcement learning game intelligent guidance algorithm that integrated state information transformation and reward function shaping techniques. Firstly, using the interface provided by the Unity engine to directly read game backend information effectively compressed the dimensionality of the state space and reduced the amount of input data. Secondly, by finely designing the reward mechanism, the convergence process of the model was accelerated. Finally, we conducted comparative experiments between the proposed algorithm model and existing methods from both subjective qualitative and objective quantitative perspectives. The experimental results show that this algorithm not only significantly improves the training efficiency of the model, but also markedly enhances the performance of the agent.

Keywords: deep reinforcement learning; game agent; reward function shaping; proximal policy optimization algorithm

收稿日期: 2023-12-29.

第一作者简介: 白 天(1983—), 男, 汉族, 博士, 教授, 博士生导师, 从事机器学习和医学人工智能的研究, E-mail: baitian@jlu.edu.cn. **通信作者简介**: 何加亮(1977—), 男, 汉族, 博士, 副教授, 从事人工智能、虚拟现实数字艺术及数字健康领域应用的研究, E-mail: 78919121@qq.com.

基金项目: 国家自然科学基金(批准号: U21A20390)和吉林省科技发展计划项目(批准号: 20210509006RQ).

随着人工智能、大数据、云计算等新兴技术的不断涌现，数字经济得到迅速发展[1]，游戏作为数字经济的核心产业之一，在这些新兴技术下也迎来了变革[2]. 游戏玩家对游戏体验的要求不断提高，用户更注重游戏的内在表现及互动效果，在这种发展趋势下，AI(artificial intelligence)算法逐渐被引入到游戏领域. AI算法具有强大的自主学习能力，能通过学习游戏的模式规则自动生成游戏内容[3]，降低传统游戏开发过程中的人工设计成本，可根据玩家的行为数据动态调整游戏难易度及奖励，满足玩家需求，实现游戏的个性化设计. AI算法的使用提升了游戏的开发效率，丰富了游戏内容[4-5]，为玩家提供了更优质的游戏体验[6]，在游戏领域有广阔的发展空间与应用前景[7].

强化学习作为AI领域的一个重要分支，不同于传统的监督学习，其本身所需的数据集均来自于与环境的试错式交互，利用环境的反馈信息不断优化自身决策[8-9]，在实际应用中该方式需消耗大量成本，而在游戏场景中则可极大降低优化成本，因此强化学习是处理游戏AI问题的最佳方法.

强化学习利用深度学习在特征提取以及非线性函数拟合方面的优势，极大提高了传统强化学习算法的泛化能力[10-11]，创造了多种适配复杂场景的高效算法[12-13]，从而使深度强化学习被广泛应用于游戏AI领域. 深度强化学习将游戏作为其学习环境，智能体通过与游戏环境的交互学习最优策略.

DeepMind团队在2013年提出了DQN(deep Q-network)算法，将深度学习与强化学习首次结合，使用神经网络替代传统的QTable计算[14]，在多款雅达利游戏中的表现达到甚至高于人类玩家水平. 2016年DeepMind团队提出的AlphaGo算法[15]，通过与人类对战，将其动作的监督学习与自我游戏的强化学习紧密结合，在世界围棋大赛中成功获胜，并于2017年提出了相比于AlphaGo更强大的智能体AlphaZero[16]，与Monte Carlo树搜索方法进行结合，实现了最优移动的选择. 2018年DeepMind团队开发的基于深度强化学习的智能体AlphaStar在“星际争霸2”[17]的训练环境下击败了职业玩家，该智能体结合LSTM(long short-term memory)[18]等技术，采用监督学习的方式与人类对战，学习人类玩家的对战策略.

为进一步提升策略的显著性，2015年OpenAI团队提出了TRPO(trust region policy optimization)[19]，采用置信域控制步长，优化了其整体的训练效果，在此基础上提出了策略约束更精简的PPO(proximal policy optimization)算法[20]，并于2018年提出了基于PPO的OpenAI Five. 腾讯公司于2019年将改进的PPO算法与LSTM技术相结合创建了“绝悟”AI[21]，在“王者荣耀”游戏环境下进行了2 100场比赛，其获胜率达99.81%. 上述研究结果表明，基于深度强化学习的游戏AI是当前游戏领域的热点问题，其在多种游戏中均表现了远超人类玩家的水平. 但目前已有的基于深度强化学习的游戏AI算法仍存在一些问题. 为避免将连续状态离散化，大部分游戏AI算法选择将图像作为算法输入，易导致数据处理过于复杂且模型难以收敛，而奖励机制的不合理设计与模型选择不当导致动作价值估计过高的问题均会使模型训练时间过长.

近年来，针对认知障碍人群研发的严肃游戏作为游戏领域的一个新型应用分支开始崭露头角[22]，其中重点训练儿童群体“计划形成”机能的策略性游戏层出不穷，此类游戏场景内各元素彼此间的关联性较强，对儿童能根据当前游戏场景中各元素的关联反映制定出通关策略的能力要求较高，游戏难度较大，大部分年龄较小的玩家无法顺利通过游戏关卡，进而失去耐心，最终失去对游戏的兴趣，游戏体验感差，玩家成就感低，而游戏的体验感是一款游戏能否被长久使用的关键性因素. 该类游戏中缺乏实时的智能AI引导策略，游戏无法根据当前儿童的使用情况制定个性化的智能提示中和游戏难度，无法在游戏场景中适时引导儿童玩家，给予其适当帮助，提升其游戏体验感. 而上述个性化游戏智能AI的整体实现难度较大，因此大部分同类游戏中并未引入该项功能.

针对传统基于深度强化学习的游戏AI算法模型目前存在的问题，本文通过引入经典“青蛙吃蜻蜓”训练计划机能的策略游戏作为模型训练的任务场景，提出一种基于深度强化学习的个性化游戏智能引导算法，能在相对复杂的环境中，根据玩家当前的情况计算出过关的最优解，并给出下一步提示. 该算法通过降低状态信息的输入维度、精细化反馈奖励值等方式，有效提升模型的训练速度与性能. 本文的创新点和贡献如下：

1) 对“青蛙吃蜻蜓”游戏的特点进行深入分析，在此基础上构建了适用于该场景的模型，并分析了

目前所面临的挑战.

2）针对本文模型，引入一种新的模型架构，并提出了改进算法. 通过对奖励函数进行塑形及对状态信息进行有效转换并结合 PPO 算法的裁剪机制，有效提升了模型的训练速度和性能，同时设计并完成算法实验，以验证本文算法模型的有效性.

1　问题描述与建模

1.1　游戏介绍

如图 1 所示，在一个 5×6 大小的方格中，存在青蛙和蜻蜓两类对象，玩家的游戏目标是通过点击的方式控制青蛙，利用多只青蛙的协作“吃掉”所有蜻蜓. 玩家每次可以选择点击一只青蛙，然后青蛙将向其面朝方向吐出舌头，直到第一次碰到蜻蜓、青蛙或者边界，当前青蛙吐出舌头后的描述分为以下 3 种情况：

1）若碰到蜻蜓，则当前青蛙收回舌头，蜻蜓直接消失；

2）若碰到青蛙，则当前青蛙收回舌头，被碰到的青蛙向面朝的方向吐出舌头，与当前青蛙做相同操作和处理，直到收回舌头后，原地旋转 180°；

3）若碰到边界，则当前青蛙收回舌头.

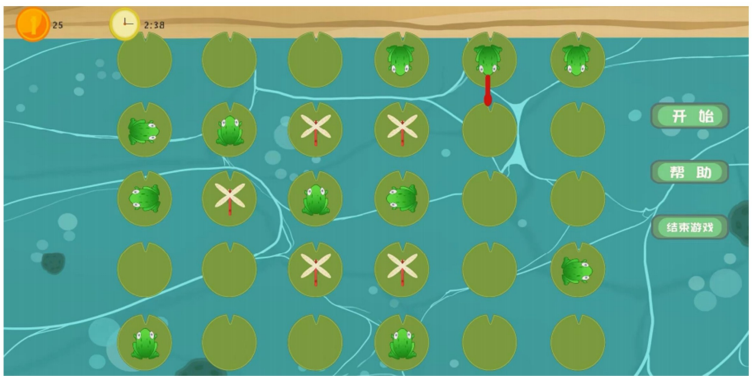

图 1　游戏可视化效果

Fig. 1　Effects of game visualization

在游戏场景中，算法需要在玩家操作游戏过程中在后台进行实时计算，得到当前情况下最优解法中的下一步策略，并在当前关卡停留时长过久时给予玩家个性化智能提示，帮助玩家完成游戏任务.

1.2　问题描述

设图中有 n 行 m 列，共有 $(t+1)$ 个格子，用 $x_{i,j}^{f}\in\{0,1\}$ 表示第 i 行第 j 列是否有青蛙，$d_{i,j}\in\{-1,-2,1,2\}$ 表示青蛙的朝向分别为上、右、下、左，用 $x_{i,j}^{d}\in\{0,1\}$ 表示第 i 行第 j 列是否有蜻蜓. 每个格子只有一个物体，即青蛙或蜻蜓，且青蛙的个数和蜻蜓的个数不大于格子个数，则有

$$\sum_{i=1}^{n}\sum_{j=1}^{m}(x_{i,j}^{f}+x_{i,j}^{d})\leqslant N, \tag{1}$$

$$x_{i,j}^{f}\cdot x_{i,j}^{d}\leqslant 1,\quad \forall i\in[0,n-1],\quad j\in[0,m-1]. \tag{2}$$

1.2.1　玩家操作

令玩家的第 t 个点击操作位置为 (px_t,py_t)，玩家点击的操作集合为 $PL=\{(px_0,py_0),(px_1,py_1),\cdots\}$，在该场景中，用 $x_{i,j}^{d}=1$ 表示玩家在第 t 步点击了方格中的第 i 行第 j 列，且玩家一次只能点击一只青蛙，因此有

$$\sum_{i=1}^{n}\sum_{j=1}^{m}c_{i,j}^{t}=1. \tag{3}$$

1.2.2　规则描述

当玩家点击第 i 行第 j 列的青蛙时，先根据青蛙的朝向进行判断，以青蛙方向朝左为例，其他情况

类似. 从$(j-1)$列开始遍历小于j的格子，令遇到第一个物体的位置为$x^o_{i,j}$，判断是青蛙还是蜻蜓，若是蜻蜓，则蜻蜓直接消失，若是青蛙，则该青蛙改变朝向180°并继续吐舌头，继续遍历该路径上的物体，以此类推，直到该路径上没有任何物体.

在游戏进行过程中场景内所有物体被触发的时刻进行模型建模. 由于玩家一次只能触发一只青蛙，所以触发时刻场景内的舌头数量有且仅有一个，因为只有舌头顶端会触发机制，故用$x^s_{i,j}$表示在第i行第j列有舌头顶端. 对于在第i行第j列的物体$x^o_{i,j}$，在检测到自己被舌头触碰到时，如果自身是蜻蜓，则直接消失，如果是青蛙，则吐出舌头，然后改变自身朝向，该过程可描述为

$$\begin{cases} d_{i,j}=-d_{i,j}, & x^f_{i,f}\cdot x^s_{i,j}=1,\\ x^d_{i,j}=0, & x^d_{i,f}\cdot x^s_{i,j}=1. \end{cases} \tag{4}$$

1.2.3 任务目标

玩家的游戏目标是通过制定青蛙的点击策略，利用多只青蛙彼此间的关联碰撞反应，协作完成“吃掉”当前关卡所有蜻蜓的任务，对完成任务所需步数并无要求，但由于本文算法需要有智能引导作用，因此给出的引导提示应尽可能接近最优解，故任务目标为

$$\begin{aligned} &\mathrm{Minimize}\sum_{t=0}^{T}\sum_{i=0}^{n}\sum_{j=0}^{m}c^t_{i,j},\\ &\text{s.t. 式(1)}\sim\text{式(4)成立}, \end{aligned} \tag{5}$$

其中T为完成任务所用的总步数.

2 模型构建与算法实现

2.1 深度强化学习

强化学习(reinforcement learning)是一个Markov决策过程，可被描述为四元组(S,A,P,R)，其中S为状态空间，A为动作空间，P为状态转移函数，R为奖励函数. 状态转移函数P是一个从$S\times A$到$P(S)$的映射，且有$s_{t+1}\sim p(s|s_t,a_t)$，即状态s_t采取动作a_t后状态转移到s_{t+1}的概率. 奖励函数是一个从$S\times A\times S$到实数域的映射，可记为$r_t=R(s_t,a_t,s_{t+1})$. 强化学习过程中，智能体先处于开始状态s_0，然后选择动作a_0执行，与环境产生交互，获得奖励并转移到状态s_1，通常称智能体经历的状态、动作序列为轨迹$\tau=(s_0,a_0,s_1,a_1,\cdots)$，强化学习的目标是最大化累积奖励：

$$R(\tau)=\max\sum_{t=0}^{T}r_t. \tag{6}$$

深度强化学习模型训练的基本结构如图2所示. 由图2可见，深度强化学习模型分为环境和智能体两部分，环境先将当前状态信息发送给智能体，智能体再根据其策略，输出动作并作用到环境中，使环境的状态发生改变，然后将环境的状态信息继续输入到智能体中，再得到动作，不断循环直到游戏结束.

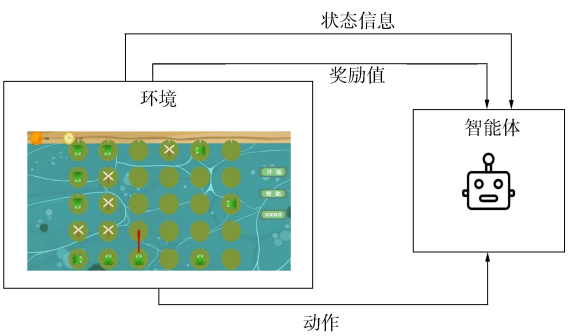

图2 深度强化学习模型训练架构

Fig. 2 Architecture of deep reinforcement learning model training

目前，深度强化学习在游戏智能体领域的实际应用中存在如下问题.

1) 高维输入空间：在一些情况下，输入信息的维度过大或信息量庞大，导致输入状态空间急剧扩大，从而使模型难以在庞大的状态空间中收敛，给模型训练带来挑战.

2) 训练速度缓慢：训练速度相对较慢，原因可能包括奖励机制设计不当等，导致需要耗费大量时间训练模型以达到令人满意的效果，该问题影响了深度强化学习的实际应用效率.

3) 训练效果欠佳：由于算法选择不当等原因，训练效果可能出现不理想的情况，表现为过高估计动作价值等问题，这种情况通常需要较长时间的训练，给实际应用带来一定的困扰.

针对上述问题，本文提出一种创新性的解决方案. 在深度强化学习技术的基础上，重点对模型与算法设计、模型部署和训练等方面进行改进，以提高系统性能和鲁棒性.

1）模型架构优化：引入新的模型架构，通过状态信息转换有效降低状态空间的维度，减少了输入数据量.

2）设计奖励机制：通过精细化设计奖励函数，加速模型的收敛过程.

3）模型部署：将训练好的模型保存为 ONNX 格式，并通过 Unity 的 Barracuda 插件实现模型的部署，确保模型在实际游戏环境中能顺利运行.

2.2　算法模型

2.2.1　模型架构

本文算法模型架构如图 3 所示. 由图 3 可见，在环境初始化后，输出状态信息至智能体，智能体先将状态信息进行转换进而获取到神经网络的输入值，网络模型根据输入值进行计算并输出动作，将状态、奖励和动作存入经验池，每隔一段时间就从经验池中取出样本用于更新网络参数，输出的动作通过转换后，在环境中模拟玩家的操作进行点击，然后由环境调用 API 获取状态信息，通过反馈转换后，再一同输入给智能体，如此循环.

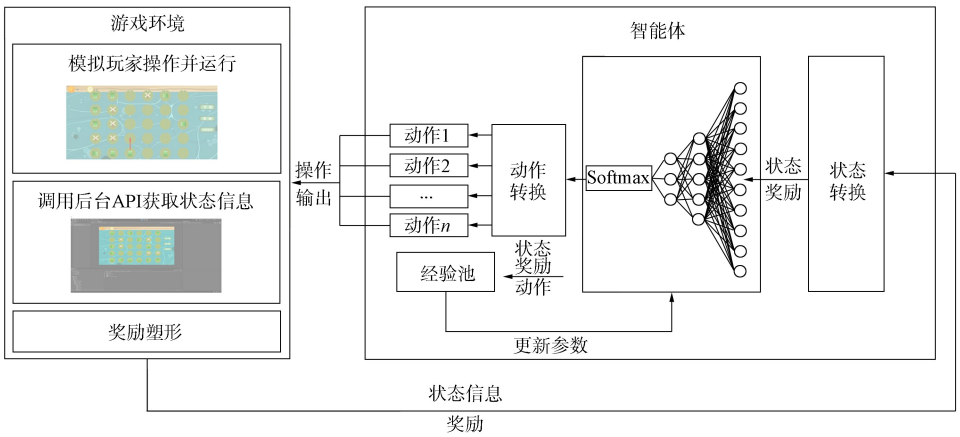

图 3　基于深度强化学习的智能引导算法模型架构

Fig. 3　Architecture of intelligent guidance algorithm model based on deep reinforcement learning

为使模型的收敛效果更好，本文采用 PPO 算法，该算法基于 TRPO 的思想，在其基础上进行改进，主要特点之一是采用了剪切操作，其优化目标如下：

$$\arg\max_{\theta} E_{s\sim v}^{\pi_{\theta_k}} E_{a\sim\pi_{\theta_k}(\cdot\mid s)}\left[\min\left\{\frac{\pi_\theta(a\mid s)}{\pi_{\theta_k}(a\mid s)}A^{\pi_{\theta_k}}(s,a),\operatorname{clip}\left(\frac{\pi_\theta(a\mid s)}{\pi_{\theta_k}(a\mid s)},1-\varepsilon,1+\varepsilon\right)A^{\pi_{\theta_k}}(s,a)\right\}\right], \tag{7}$$

其中 $\operatorname{clip}(x,l,r):=\max\{\min\{x,r\},l\}$，即把 x 限制在$[l,r]$内，如果 $A^{\pi_{\theta_k}}(s,a)>0$，说明这个动作的价值高于平均，最大化该式会增大$\dfrac{\pi_\theta(a\mid s)}{\pi_{\theta_k}(a\mid s)}$，但不会使其超过 $1+\varepsilon$. 通过引入剪切函数限制新旧策略比值的变化范围，用于平衡策略更新的幅度，防止在每次更新中引入过大的变化，从而提高算法的稳定性，更容易收敛到较好的策略.

目前，许多基于深度强化学习的小游戏智能体模型为避免将连续状态离散化，都将图像直接作为模型输入，会导致处理过于复杂，且可能出现模型难以收敛等情况. 针对该问题，本文提出改进的状态信息及精细化的反馈奖励，以提升模型收敛效果，加快模型的训练速度.

2.2.2　状态信息设计

首先将青蛙和蜻蜓的位置信息转换为一维，进行如下转换：

$$\begin{cases} p_{i,m+j}=x_{i,j}^{f}, \\ d_{i,m+j}=x_{i,j}^{d}, \end{cases} \tag{8}$$

其中：青蛙的位置信息表示为集合 $P=\{p_1,p_2,\cdots,p_N\}$，$p_i=\{0,1\}$，$i=\{1,2,\cdots,N\}$表示是否存在青

蛙；蜻蜓的位置表示为集合 $D=\{d_1,d_2,\cdots,d_N\}$，$d_i=\{0,1\}$，$i=\{1,2,\cdots,N\}$表示是否存在蜻蜓. 每只青蛙的朝向表示为 $T=\{t_1,t_2,\cdots,t_N\}$，其中 $t_i=\{0,1,2,3,4\}$，$i=\{1,2,\cdots,N\}$，0 表示当前位置没有青蛙，1,2,3,4 分别表示青蛙朝向是上、右、下、左. 因此，模型在第 t 步的状态信息为

$$S_t=(P_t,T_t,D_t). \tag{9}$$

2.2.3 动作输出设计

动作主要为玩家的操作，玩家共有 N 种操作方式，设玩家在第 t 次的操作方式为 $A_t=\{1,2,\cdots,N\}$，则玩家点击位置的计算方式如下：

$$\mathrm{row}_t=\frac{A_t}{n}, \tag{10}$$

$$\mathrm{col}_t=(A_t+n-1)\bmod m, \tag{11}$$

其中 row_t 表示在第 t 步时点击的行数，col_t 表示在第 t 步时点击的列数.

2.2.4 奖励函数设计

本文用 R_t 表示奖励值，一般地，如果能提供更精细的奖励反馈，则训练效果和训练速度能得到有效提升. 本文在反复实验对比后，设计精细化奖励如下：

1）如果点击到空白位置，即没有点到青蛙，则表示无效操作，返回−10；

2）如果点击到了青蛙，但青蛙吐出舌头只碰到了边界，则表示低效操作，返回−2；

3）如果点击到了青蛙，且该青蛙吐出舌头碰到了另一只青蛙，由于并不能完全确定是无效操作还是为了使某只青蛙转向，则返回 0；

4）如果点击到了青蛙，且该青蛙吐出舌头碰到了另一只青蛙，还导致另一只青蛙面向蜻蜓，则表示有效操作，返回 2；

5）如果点击到了青蛙，且该青蛙吐出舌头碰到蜻蜓，则表示最有效操作，返回 5.

3 实验结果与分析

3.1 实验环境与设置

为更好展示模型效果，本文实验在自行开发的 Unity 游戏程序上直接训练. 图 4 为 Unity 开发的实际展示效果，通过封装好的 API 获取后台信息，并结合 Socket 通信实现 Python 和 Unity 程序的跨架构通信，游戏被设计为 3 个难度，不同难度所需的点击数会增加，游戏地图在保证一定有解的前提下随机生成. 为分别验证奖励函数塑形和算法模型推理的实际应用效果，本文实验将进行两方面的对比.

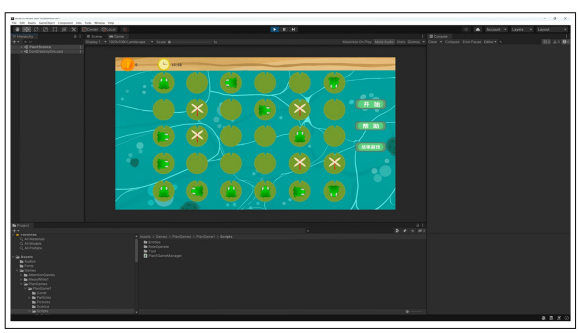

图 4 Unity 开发实际展示效果

Fig. 4 Actual demonstration effect of Unity development

3.1.1 奖励函数塑形与对比

为更清晰地展现奖励函数塑形技术的效果，本文设计一种对比实验，该实验采用一种与蜻蜓数量变化直接正相关的奖励机制. 该奖励机制的核心是通过即时反馈游戏中的关键指标——蜻蜓数量的变化，指导智能体的学习方向. 该奖励机制的计算公式为

$$R_t=A\cdot(|D_t+1|-|D_t|), \tag{12}$$

其中 A 为一正常数，$|D_t|$ 和 $|D_t+1|$ 分别表示在第 t 个操作和在第$(t+1)$个操作时的蜻蜓数量. 每当智能体的执行动作导致游戏中蜻蜓数量增加时，它将获得正向奖励；反之，如果动作导致蜻蜓数量减少，则会受到惩罚. 通过这种方式，该奖励机制直接促进了智能体对能增加蜻蜓数量行为的学习，进而提高了其在游戏中的性能.

通过与本文提出的奖励函数塑形方法进行对比实验，不仅可直观地评估两种不同奖励机制对智能

体学习效率和最终性能的影响，还能进一步验证奖励函数塑形技术在提高模型训练速度和性能方面的优越性. 该对比实验的设计，为探索更高效、更智能的游戏 AI 训练方法提供了有价值的参考.

3.1.2　模型训练与随机策略对比测试

为验证本文模型的推理效果，将其与随机策略进行对比，即随机点击场上某一只青蛙. 深度强化学习部分基于 Python 的 PyTorch 库实现，网络采用 3 层全连接层，隐藏层维度为 128，折扣因子为 0.99，学习率为 0.000 3，并使用 Adam 优化器. 对采用不同方法通关游戏所需的实际步数进行对比.

3.2　实验结果与分析

图 5 为不同奖励机制的奖励值变化曲线，其中：蓝色线为本文奖励机制的训练情况；橙色线为平滑曲线，用于观察趋势；绿色线为对比设计的奖励机制；红色线为平滑曲线. 由图 5 可见，本文提出的精细化奖励机制能使模型的训练效果更好，在 4×10^6 个时间步时即趋于稳定，而对比的奖励机制基本在 6×10^6 个时间步时才趋于稳定，训练速度提升约 50%.

图 6 为 3 种不同算法和奖励机制的训练效果对比，其中：绿色为本文基于深度强化学习的智能引导算法，采用精细化奖励机制；橙色为传统深度强化学习算法，采用简单的对比奖励机制；蓝色为随机点击的效果，随机点击场上的青蛙，不使用任何智能引导. 由图 6 可见，前两个均使用深度强化学习模型，效果明显优于随机点击. 实验结果表明，本文采用精细化奖励机制的训练效果优于对比奖励机制.

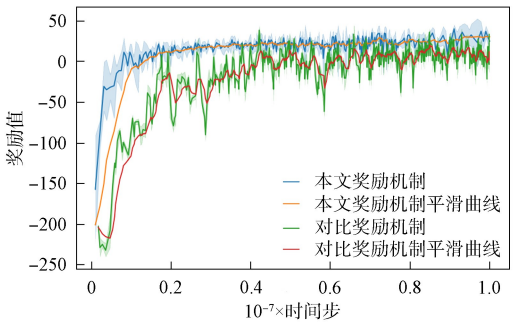

图 5　不同奖励机制的奖励值变化曲线

Fig. 5　Change curves of reward values of different reward mechanisms

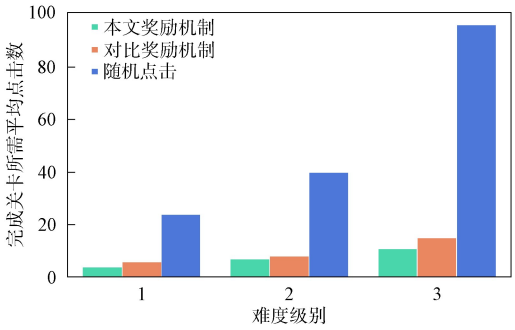

图 6　不同算法和奖励机制训练效果对比

Fig. 6　Comparison of training effects of different algorithms and reward mechanisms

3.3　算法部署

ONNX(open neural network exchange)是一种针对机器学习所设计的开放式文件格式，用于存储训练好的模型，它使不同的人工智能框架(如 PyTorch，MXNet)可以采用相同格式存储模型数据并交互，因此在训练完代码后，为能在 Unity 的游戏中直接使用，本文将训练模型保存为 ONNX 格式，并通过 Unity 的 Barracuda 插件实现打包部署，实际效果与实验效果基本一致.

综上所述，针对传统基于深度强化学习的游戏智能体算法模型输入状态信息维度过大以及训练速度较慢的问题，本文提出了一种基于深度强化学习的游戏智能引导算法. 该算法采用一种新的模型架构，通过状态信息转换，能有效降低状态信息维度，同时精细化反馈奖励值，从而提高模型训练速度和性能. 对比实验结果表明，该算法相比于传统算法，在训练速度上提升了约 50%，在性能上提升了 25%以上，证明了该算法的优越性与有效性.

参　考　文　献

[1] 曹馨月. 数字经济时代中国游戏产业发展面临的机遇、挑战和对策 [J]. 商业观察，2023，9(23)：92-95.(CAO X Y. Opportunities，Challenges and Countermeasures for the Development of China's Game Industry in the Era of Digital Economy [J]. Business Observation，2023，9(23)：92-95.)

[2] 韩东林，李振. 数字技术赋能网络游戏产业高质量发展路径研究 [J]. 合肥师范学院学报，2022，40(5)：74-80.(HAN D L，LI Z. Research on Digital Technology Enabling High-Quality Development Path of Online Game

Industry [J]. Journal of Hefei Normal University, 2022, 40(5): 74-80.)
[3] 陈勇. 人工智能技术在计算机游戏软件中的应用 [J]. 软件, 2022, 43(10): 92-94. (CHEN Y. Application of Artificial Intelligence Technology in Computer Game Software [J]. Software, 2022, 43(10): 92-94.)
[4] 宋丕丞. AI赋能游戏开发: 演化脉络与应用趋势 [J]. 北京文化创意, 2024(4): 4-16. (SONG P C. Empowering Game Development with AI: Evolution and Application Trends [J]. Beijing Cultural Creativity, 2024(4): 4-16.)
[5] 王一帆. 游戏化教学策略——AI绘画应用于中小学生计算思维培养的路径探讨 [C]//2024计算思维与STEM教育研讨会暨Bebras中国社区年度工作会议论文集. 北京: 北京师范大学, 2024: 1-13. (WANG Y F. Gamification in Education: Exploring the Path of AI-Powered Drawing for Cultivating Computational Thinking in Primary and Secondary School Students [C]//Proceedings of the 2024 Symposium on Computational Thinking and STEM Education and the Annual Meeting of the Bebras China Community. Beijing: Beijing Normal University, 2024: 1-13.)
[6] 周飞, 李久艳. 人工智能在游戏开发中的应用现状和展望 [J]. 中国管理信息化, 2020, 23(23): 183-185. (ZHOU F, LI J Y. The Application Status and Prospect of Artificial Intelligence in Game Development [J]. China Management Informationization, 2020, 23(23): 183-185.)
[7] KONSTANTINOS S, GEORGE K S, GEORGE A P. Reinforcement Learning in Game Industry—Review, Prospects and Challenges [J]. Applied Sciences, 2023, 13(4): 2443-1-2443-23.
[8] 高阳, 陈世福, 陆鑫. 强化学习研究综述 [J]. 自动化学报, 2004, 30(1): 86-100. (GAO Y, CHEN S F, LU X. Review of Reinforcement Learning Research [J]. Acta Automatica Sinica, 2004, 30(1): 86-100.)
[9] KUMAR A S, GOPINATHA P, SOHOM C. Reinforcement Learning Algorithms: A Brief Survey [J]. Expert Systems with Applications, 2023, 231: 120495-1-120495-32.
[10] LECUN Y, BENGIO Y, HINTON G. Deep Learning [J]. Nature, 2015, 521(7553): 436-444.
[11] KAI A, MARC P D, MILES B, et al. Deep Reinforcement Learning: A Brief Survey [J]. IEEE Signal Processing Magazine, 2017, 34(6): 26-38.
[12] ZHAO D B, SHAO K, ZHU Y H, et al. Review of Deep Reinforcement Learning and Discussions on the Development of Computer Go [J]. Control Theory & Applications, 2016, 33(6): 701-717.
[13] WAN L P, LAN X G, ZHANG H B, et al. A Review of Deep Reinforcement Learning Theory and Application [J]. Pattern Recognition and Artificial Intelligence, 2019, 32(1): 67-81.
[14] MHIN V, KAVUKCUOGLU K, SILVER D, et al. Human-Level Control through Deep Reinforcement Learning [J]. Nature, 2015, 518(7540): 529-533.
[15] 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.
[16] SILVER D, SCHRITTWIESER J, SIMONYAN K, et al. Mastering the Game of Go without Human Knowledge [J]. Nature, 2017, 550(7676): 354-359.
[17] VINYALS O, EWALDS T, BARTUNOV S, et al. Grandmaster Level in StarCraft Ⅱ Using Multi-agent Reinforcement Learning [J]. Nature, 2019, 575(7782): 350-354.
[18] GREFF K, SRIVASTAVA R K, KOUTNIK J, et al. LSTM: A Search Space Odyssey [J]. IEEE Transactions on Neural Networks and Learning Systems, 2016, 28(10): 2222-2232.
[19] SCHULMAN J, LEVINE S, ABBEEL P, et al. Trust Region Policy Optimization [C]//International Conference on Machine Learning. New York: ACM, 2015: 1889-1897.
[20] SCHULMAN J, WOLSKI F, DHARIWAL P, et al. Proximal Policy Optimization Algorithms [EB/OL]. (2017-04-28)[2023-12-28]. https://arxiv.org/abs/1707.06347v2.
[21] YE D, LIU Z, SUN M, et al. Mastering Complex Control in MOBA Games with Deep Reinforcement Learning [C]//Proceedings of the 34th AAAI Conference on Artificial Intelligence. Palo Alto: AAAI, 2020: 6672-6679.
[22] 张薛晴, 宋玉磊, 田萌, 等. 严肃游戏在认知障碍人群中应用的范围综述 [J]. 医学信息, 2023, 36(21): 173-177. (ZHANG X Q, SONG Y L, TIAN M, et al. A Scoping Review of Serious Game in People with Cognitive Impairment [J]. Journal of Medical Information, 2023, 36(21): 173-177.)

(责任编辑: 韩 啸)