

离散与连续环境下的智能体决策系统：基于 DQN 与行为树的坦克 AI 研究

黄鹤翔 高恒斌 都奕宁

信息与电子工程学院

浙江大学, 杭州, 中国

摘要—本报告详细阐述了一个基于深度强化学习 (DQN) 与启发式决策的迷宫自动寻路及对战 AI 系统的设计与实现。项目的核心挑战在于不依赖现有的深度学习框架, 而是仅使用 NumPy 库从底层构建卷积神经网络及优化器。报告首先介绍了迷宫生成算法 (DFS、Prim) 及经典启发式寻路算法 (A*), 随后深入探讨了自定义深度学习框架 simple_nn 的实现细节, 包括 im2col 卷积加速、反向传播算法及 Adam 优化器。在此基础上, 搭建了 DQN 模型, 设计了包含墙壁信息、位置信息及访问记录的 7 通道状态空间, 并通过经验回放和目标网络机制稳定训练过程。同时, 探索了基于遗传算法的神经网络进化策略, 通过锦标赛选择与精英保留机制优化智能体决策。此外, 针对连续物理环境下的复杂对战需求, 设计了基于分层行为树的智能决策系统。该系统引入了基于 Liang-Barsky 算法的通视性与危险势场分析、结合“拉线法”的混合路径规划及动态规避策略, 有效解决了连续空间下的碰撞检测与精准操控难题。最后, 展示了 AI 在迷宫寻路测试及人机对战模式中的表现, 验证了手写框架及决策树 AI 算法的有效性。

Index Terms—深度强化学习, DQN, 卷积神经网络, NumPy, 迷宫生成, A* 算法, 自动寻路, 启发式决策, 行为树, 混合路径规划, 危险势场

I. 引言

项目的设计灵感源于 B 站 up 河南胡季果和 L_Shy_P 的视频, 视频中出现了一个近乎无敌的 AI, 能够追踪敌人躲避子弹。

本项目旨在构建一个具备自主学习能力的坦克对战 AI 系统。与常规项目不同, 本项目的核心要求是“去框架化”, 即不调用成熟的深度学习库, 而是利用 Python 和 NumPy 从零实现神经网络的底层算子及训练机制。同时, 本项目还尝试了遗传算法 (Genetic Algorithm) 作为另一种优化手段, 通过模拟生物进化过程来迭代神经网络参数。这不仅考验对强化学习算法

的理解, 更要求对神经网络反向传播、矩阵运算及优化算法有深入的掌握。

此外, 为了应对更加复杂的连续物理环境对战需求, 本项目还设计了一套基于启发式规则与行为树的智能决策系统。传统的 DQN 模型虽然在离散网格中表现优异, 但在处理连续坐标、任意角度旋转及精细碰撞检测时面临状态空间爆炸的问题。因此, 本项目引入了“Smart AI”架构, 不再依赖强化学习, 而是结合了危险势场分析、混合路径规划及动态规避算法。这种启发式决策的设计, 旨在全面探索不同技术路线在游戏 AI 中的应用潜力, 既实现了离散环境下的端到端学习, 又保证了连续环境下的高水平竞技表现。

II. 设计任务和要求

本项目的具体设计任务和要求如下, 涵盖了从底层框架到上层决策的全栈开发:

1) 迷宫环境构建:

- 实现基于深度优先搜索和随机 Prim 算法的迷宫生成器, 支持生成不同拓扑结构的地图。
- 实现 A* 启发式寻路算法, 作为 AI 路径规划的基准。

2) 底层深度学习框架实现:

- 仅使用 NumPy 库, 从零构建神经网络核心算子, 包括全连接层、卷积层及激活函数。
- 实现 im2col 技术以加速卷积运算, 并推导实现各层的反向传播梯度计算。
- 实现 Adam 自适应优化器及 MSE Loss 损失函数, 支持网络的参数更新。

3) 离散环境强化学习 AI:

- 构建基于 CNN 的 Q 值估计网络，设计包含墙壁、位置及轨迹信息的 7 通道状态空间。
- 实现经验回放与目标网络机制，解决训练不稳定的问题。
- 在离散网格环境中训练智能体，实现从随机探索到最优路径规划的自主学习。

4) 遗传算法进化策略：

- 构建全连接神经网络作为决策模型，探索非梯度下降类的优化方法。
- 实现基于锦标赛选择、均匀交叉和高斯变异的遗传算法，用于神经网络参数的进化迭代。

5) 连续环境物理引擎适配：

- 升级游戏引擎以支持浮点坐标与任意角度旋转。
- 实现基于分离轴定理或 Liang-Barsky 算法的射线-OBV 碰撞检测，用于子弹判定与视线分析。

6) 连续环境启发式智能 AI：

- **危险感知：**设计危险评分模型，基于物理预测计算未来时间窗口内的碰撞风险。
- **决策系统：**构建分层行为树，实现生存优先、动态规避、战术进攻与路径巡逻的状态切换。
- **运动控制：**设计“网格 BFS + String Pulling 拉线法”的混合路径规划算法，并实现基于 PID 思想的平滑路径跟踪控制。

7) 人机对战系统集成：集成上述模型，实现流畅的实时对战功能，并提供 AI 决策的可视化调试。

- 递归完成后，整个网格形成一棵生成树，对应一个无环连通迷宫。

B. 迷宫生成算法-prim

Prim 算法原本用于求解最小生成树。在迷宫生成中，我们将其改造为随机 Prim 算法：不考虑边权，而是随机选择待扩展的边。

算法维护一个“边界集合”，包含所有与已生成区域相邻但尚未加入的格子。每一步从边界集中随机选取一个格子，将其与已生成区域中的某个邻居连接（打通墙壁），并将该格子的新邻居加入边界集。重复此过程直至覆盖全部格子。

- 将网格视为完全图，每条潜在通道（相邻格子间）是一条边；
- 任选一个起始格子加入生成树；
- 将其所有相邻格子加入“边界集”；
- 从边界集中随机选一格子，随机选择其一个已在生成树中的邻居，打通两者之间的墙，并将该格子纳入生成树，同时将其未访问邻居加入边界集；
- 直至所有格子被纳入。

III. 算法原理

A. 迷宫生成算法-DFS

DFS 迷宫生成算法本质上是一种递归回溯过程。它从起始格子出发，随机选择一个未访问的相邻格子进行“打通”（移除中间墙壁），然后递归地对该新格子执行相同操作。当当前格子的所有邻居均已被访问时，算法回退至上一层，直至遍历完整个网格。

- 将每个网格单元视为图中的一个节点；
- 初始时所有节点标记为“未访问”，所有相邻节点间存在“墙”；
- 从任意起点（如左上角 (0,0)）开始 DFS 遍历；
- 每次移动到新节点时，移除当前节点与目标节点之间的墙，并将目标节点加入连通图；

DFS 生成的迷宫通常具有长而曲折的走廊和较少的分支,Prim 生成的迷宫通常具有更多短分支和较短的主路径。

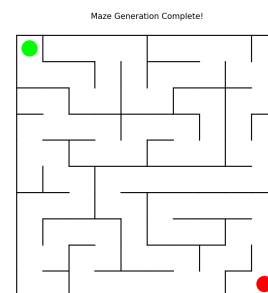


图 1: DFS 生成迷宫

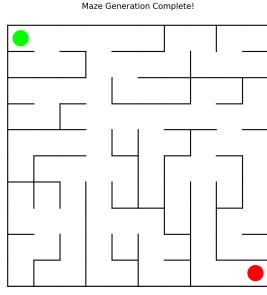


图 2: prim 生成迷宫

C. 解迷宫算法-A*

A* 算法是一种广泛应用于路径规划和图遍历的启发式搜索算法。它结合了 Dijkstra 算法的完备性与贪心最佳优先搜索的效率，能够高效地找到从起点到终点的最短路径。

A* 算法通过评估函数 $f(n)$ 来决定下一个要探索的节点：

$$f(n) = g(n) + h(n)$$

其中：

- $g(n)$: 从起点到当前节点 n 的实际代价。
- $h(n)$: 从当前节点 n 到目标点的启发式估计代价。

在本项目中使用曼哈顿距离 $abs(x_1 - x_2) + abs(y_1 - y_2)$ 进行启发式搜索。我们在初始化时将起点 (0,0) 加入 open 列表中，然后进入循环，直到 open 列表为空或找到终点为止。在每一步迭代中，我们从 open 列表中选择 f 值最小的节点作为当前节点。然后检查当前节点是否为目标节点，如果是则结束循环；如果不是目标节点，则将其标记为已处理并将其相邻的未处理节点加入 open 列表中。同时，将已处理的节点加入 closed 列表中以避免重复处理。

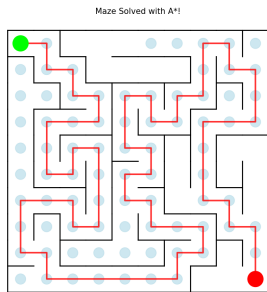


图 3: A* 算法解迷宫

D. 自定义深度学习框架 (simple_nn)

为了替代 PyTorch，我们实现了 `simple_nn.py`。

1) 卷积层实现：卷积操作的核心在于高效地提取局部特征。为了避免低效的多重循环，采用了 `im2col` 技术，将输入特征图展开为矩阵，从而将卷积运算转化为矩阵乘法。

$$Y = W_{col} \times X_{col} + b \quad (1)$$

其中 X_{col} 是展开后的输入矩阵， W_{col} 是重排后的权重矩阵。

2) 优化器：实现了 Adam 优化器，结合了动量法和 RMSProp 的优点，自适应调整学习率。

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \quad (2)$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \quad (3)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t + \epsilon}} \hat{m}_t \quad (4)$$

E. AI 训练-DQN

DQN (Deep Q-Network) 将卷积神经网络与 Q-Learning 结合，解决了高维状态空间的强化学习问题。

1) 核心机制：

- 经验回放：构建一个回放池 D ，存储转移样本 $(s, a, r, s', done)$ 。训练时随机采样一个小批量，打破了数据间的相关性，提高了训练的稳定性。
- 目标网络：引入一个参数滞后的目标网络 \hat{Q} 计算目标值，避免了“自举”导致的目标值震荡。目标网络每一定时间从策略网络中复制参数。

通过贝尔曼最优方程来计算期望 Q 值，即执行该动作后获得的即时奖励 r 加上对未来最大 Q 值的折扣期望

$$Q^*(s, a) = \mathbf{E}_{s' \sim \mathcal{P}} [r + \gamma \max_{a'} Q^*(s', a')]$$

目标函数为最小化均方误差：

$$L(\theta) = \mathbf{E} \left[\left(r + \gamma \max_{a'} \hat{Q}(s', a'; \theta^-) - Q(s, a; \theta) \right)^2 \right] \quad (5)$$

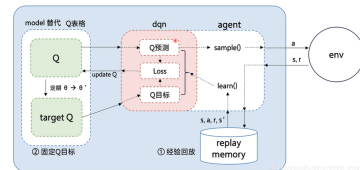


图 4: dqn 算法流程结构

2) Q 值估计网络设计-输入层: 输入层设计为 $7 \times H \times W$ 的张量, 包含 7 个通道:

- 通道 0-3 (墙壁信息): 分别编码每个格子上、下、左、右四个方向的墙壁存在情况
- 通道 4-5 (位置信息): 分别使用 One-hot 编码智能体位置和目标位置
- 通道 6 (历史轨迹): 记录智能体访问过的路径, 赋予智能体记忆

3) Q 值估计网络设计-卷积层: 网络包含 4 层卷积层, 通道数分别为 32, 64, 128, 128, 主要为卷积层与 relu 激活层的循环。网络中没有加入传统 CNN 网络中常见的池化层, 主要是因为池化层的加入会模糊智能体与目标点的位置信息, 影响训练效果。

通过堆叠 4 层卷积层, 深层神经元的感受野逐渐扩大, 能够感知更大范围的迷宫结构, 从而规划长距离路径。通道数逐层增加 ($32 \rightarrow 128$), 使得网络能够组合低级几何特征。

表 I: 卷积神经网络各层参数配置

	输入通道数	输出通道数	卷积核大小	步长	填充
第一层	7	32	3	1	1
第二层	32	64	3	1	1
第三层	64	128	3	1	1
第四层	128	128	3	1	1

4) Q 值估计网络设计-全连接层: 卷积层输出展平后, 连接 3 层全连接层 ($1024 \rightarrow 512 \rightarrow 4$), 提供了强大的非线性拟合能力, 将提取的迷宫特征转化为上、下、左、右四个离散动作的 Q 值。

F. 超参数设计

```

1 # Hyperparameters
2 WIDTH, HEIGHT = 10, 10
3 EPISODES = 10000
4 BATCH_SIZE = 32
5 GAMMA = 0.9 # 对未来奖励的关心程度, 越大越关心
6 # 使用线性下降的 EPSILON, EPSILON 越大越倾向于随
  机探索
7 EPSILON_START = 1.0
8 EPSILON_END = 0.05
9 EPSILON_DECAY_EPISODES = 4000
10 LR = 0.0001
11 TARGET_UPDATE = 200 # 目标网络的更新频率
12 MEMORY_SIZE = 50000 # 经验池的大小

```

G. 奖励设计

```

# 每一步惩罚使得 ai 寻找最短路径
reward = -0.1
done = False

if (nx, ny) in neighbors:
    self.agent_pos = (nx, ny)

    if self.visited_map[ny, nx] > 0:
        # 格子重复访问惩罚
        reward = -0.5
    else:
        self.visited_map[ny, nx] = 1.0
        # 访问新格子的奖励
        reward = 0.2

    if self.agent_pos == self.goal_pos:
        # 到达目标点的奖励
        reward = 20.0
        done = True
else:
    # 撞墙惩罚
    reward = -1.0

```

H. AI 训练-遗传算法

遗传算法是一种受达尔文生物进化论启发的元启发式优化算法。它模拟了自然界“物竞天择, 适者生存”的过程, 通过在解空间中维护一个种群 (Population), 并利用选择 (Selection)、交叉 (Crossover) 和变异 (Mutation) 等遗传算子, 迭代地演化出越来越优的解决方案。

在这个项目中, 我们使用一个全连接神经网络来对智能体的下一步动作进行决策, 通过遗传算法迭代更新神经网络的参数。同时在训练过程中, 我们借鉴了 DQN 的训练思路, 每隔一段时间将训练目标的参数从智能体中复制下来, 以求获得更加强大的智能体。

- 选择算子: 锦标赛选择
- 交叉算子: 均匀交叉
- 变异算子: 高斯扰动
- 进化策略: 精英保留, 将每一代中适应度最高的前 10% 个体 (精英) 直接复制到下一代, 不经过交叉和变异

I. 连续环境下的智能决策系统

随着游戏环境从离散网格升级为连续物理环境, AI 的决策机制也进行了相应的升级。新的智能体不再局限于上下左右的离散移动, 而是基于连续的坐标体系和角度控制, 实现了更加精准的战术动作。

1) 基于物理场的危险感知模型: 在连续环境中, 子弹的轨迹不再是简单的网格跳跃, 而是具有精确坐标和速度矢量的连续运动。AI 引入了危险评分机制, 通过模拟场上所有子弹在未来时间窗口 $T_w = 1.5s$ 内的运动轨迹, 计算其与自身碰撞的风险。

对于每一个子弹 b_i , 我们首先将其相对速度 v_{rel} 和相对位置投影到坦克的局部坐标系中。利用 Liang-Barsky 算法或分离轴定理, 检测子弹轨迹射线与坦克有向包围盒的相交情况。为了增加安全性, 我们在 OBB 检测中引入了安全边距 $margin = 5$ 像素。若发生碰撞, 则计算预计碰撞时间 t_{impact} 。危险分数 S_{danger} 定义为所有威胁子弹分数的最大值:

$$S_{danger} = \max_{i \in \mathcal{B}_{threat}} \left(\frac{1}{t_{impact}^{(i)} + \epsilon} \right) \quad (6)$$

其中 \mathcal{B}_{threat} 为所有在 T_w 内会发生碰撞的子弹集合, $\epsilon = 0.1$ 为防止除零的平滑项。该公式表明, 碰撞时间越短, 危险分数越高, 呈反比关系。

2) 分层行为树与动态规避: 为了在复杂的实时对战中做出最优反应, AI 采用了基于优先级的行为树决策逻辑。系统在每一帧计算当前状态的危险分数, 并根据此动态切换行为模式:

- 1) **生存优先**: 当检测到 $S_{danger} > 0$ 或预判动作会导致危险时, 触发规避逻辑。系统采样动作空间 $\mathcal{A} = \{Forward, Backward, Stop\} \times \{Left, Right, None\}$ 中的 9 种候选动作, 分别模拟执行后的未来状态并计算危险分数, 选择 S_{danger} 最小的动作执行:

$$a^* = \arg \min_{a \in \mathcal{A}} S_{danger}(s', a) \quad (7)$$

- 2) **攻击决策**: 若处于安全状态且满足射击条件, 则进入攻击模式。系统计算目标方位角 θ_{target} 与当前朝向 $\theta_{current}$ 的偏差 $\Delta\theta$, 采用比例控制策略调整

朝向:

$$u_{turn} = \begin{cases} Right, & \text{if } \Delta\theta > \delta \\ Left, & \text{if } \Delta\theta < -\delta \\ None, & \text{otherwise} \end{cases} \quad (8)$$

其中 $\delta = 0.1$ rad 为瞄准死区。当 $|\Delta\theta| < 0.15$ rad 时, AI 将判定为锁定目标并自动开火。

- 3) **路径规划**: 作为默认行为, 规划通向敌人的路径。在路径跟踪控制中, 我们采用了分段控制策略:

$$u_{move} = \begin{cases} Turn, & \text{if } |\Delta\theta| > 0.5 \text{ rad} \\ Forward + Turn, & \text{if } |\Delta\theta| \leq 0.5 \text{ rad} \end{cases} \quad (9)$$

该策略保证了在大幅度转向时原地旋转以避免碰撞, 而在小角度偏差时能够边走边转, 实现平滑的切向移动。

3) 视线检测与通视性分析: 为了判断能否攻击敌人或进行路径平滑, 系统实现了基于几何的通视性检测算法。该算法将连接起点与终点的线段视为射线, 遍历场景中所有的静态墙壁和动态墙壁。

对于每一个墙壁矩形, 将其适度膨胀以预留安全边距, 然后利用 Liang-Barsky 线段裁剪算法检测射线是否与矩形相交。若无任何相交, 则判定两点间通视。该检测被广泛应用于攻击判定和路径优化中。

4) 混合路径规划算法: 针对连续环境下的移动需求, 设计了“Dijkstra 全局规划 + 路径平滑”的混合路径规划算法:

- **全局规划**: 在底层的网格地图上运行 Dijkstra 算法。由于网格图边权均等, 该算法等价于广度优先搜索, 能够保证找到从起点格子到终点格子的最短离散路径 $P_{grid} = \{c_0, c_1, \dots, c_n\}$ 。
 - **路径平滑**: 由于网格路径呈锯齿状, 不符合坦克的运动学特性。系统采用“拉线法”对路径进行后处理: 从路径起点 p_{start} 开始, 贪心地尝试连接后续节点 p_k 。若线段 $\overline{p_{start}p_k}$ 不与任何墙壁相交, 则剔除中间节点, 将路径拉直。
 - **路径跟踪控制**: 智能体实时跟踪平滑后的路径点。控制律设计如下: 若角度偏差 $|\Delta\theta| > 0.5$ rad, 则原地旋转; 若 $|\Delta\theta| \leq 0.5$ rad, 则在调整角度的同时全速前进, 实现平滑的切向移动。
- 连续环境下整体场景如下图所示:

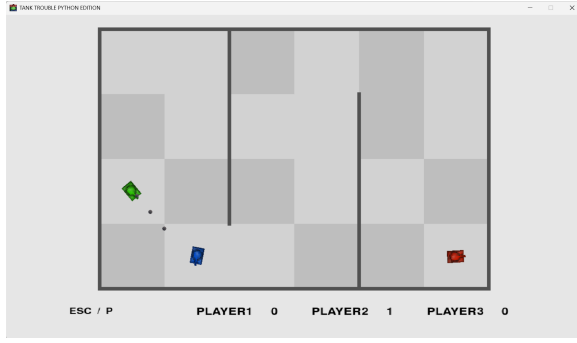


图 5: 连续环境下的坦克对战场景

IV. 主要仪器设备

- **开发语言:** Python 3.10
- **核心库:** NumPy (用于矩阵运算), Matplotlib (用于可视化)
- **硬件环境:** AMD CPU
- **操作系统:** Linux/Windows

V. 设计结果

对于简单环境的 DQN 训练, reward 曲线如下图所示

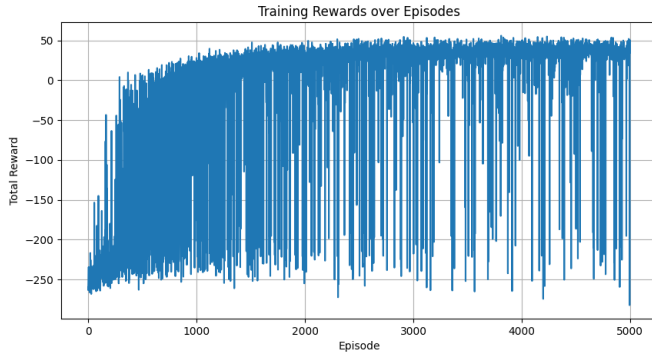


图 6: dqn 奖励变换曲线

更多结果请参见代码压缩包与视频压缩包

VI. 结论

本报告详细阐述了从零构建深度强化学习框架及坦克对战 AI 的全过程。我们首先成功实现了基于 NumPy 的底层神经网络框架, 并通过 DQN 算法在离散迷宫中完成了初步的寻路训练, 验证了“去框架化”深度学习的可行性。

同时, 遗传算法的实验结果表明, 在特定任务下, 基于进化的参数搜索策略也能取得与梯度下降相媲美的效果, 为强化学习提供了有益的补充。

随后, 针对实战中复杂的连续物理环境, 我们突破了离散动作空间的限制, 开发了基于行为树、危险势场及混合路径规划的 Smart AI。该系统成功解决了连续空间下的碰撞检测、动态规避及平滑控制等难题。实验结果表明, Smart AI 在动态规避子弹和战术进攻方面表现优异, 展现了启发式算法在实时对抗类游戏中的鲁棒性与高效性。

本项目不仅加深了对深度学习底层原理的理解, 也探索了将传统游戏 AI 技术与现代物理引擎结合的有效路径, 为后续研究更复杂的强化学习算法奠定了坚实基础。

VII. 项目分工与产出

表 II: 人员分工

姓名	学号	分工	贡献占比
黄鹤翔	3230106231	简单环境迷宫生成、决策 ai、DQN 训练	10
高恒斌	3230105132	连续环境决策 ai	10
都奕宁	3230100948	简单环境决策、连续环境决策	10

本项目的完整实现代码、训练脚本已开源在 GitHub 仓库: <https://github.com/GgbNnD/tankalg>

VIII. 参考文献

- 1) Mnih, V., Kavukcuoglu, K., Silver, D. et al. Human-level control through deep reinforcement learning. Nature 518, 529–533 (2015).
- 2) Hart, P. E., Nilsson, N. J., & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics, 4(2), 100-107.
- 3) Liang, Y. D., & Barsky, B. A. (1984). A new concept and method for Line Clipping. ACM Transactions on Graphics (TOG), 3(1), 1-22.
- 4) Holland, J. H. (1992). Adaptation in natural and artificial systems. MIT press.
- 5) 河南 胡季果, L_Shy_P. (2024). Bilibili Video: <https://www.bilibili.com/video/BV17EswzLEKw/>

IX. 引用

连续环境下游戏引擎参考自 GitHub 开源项目：<https://github.com/mglyn/TANKTROUBLE-pythonedition.git>，其依赖于 numpy/pygame 库