强化学习应用案例—OpenAI 捉迷藏系统的复现
实验指导书

谢　榕

强化学习应用案例—OpenAI 捉迷藏系统的开发

一、实验目的

通过复现美国 OpenAI 公司开发的机器学习系统-Hide and Seek（捉迷藏）游戏，熟悉机器学习系统的基本结构，包括定义、工作原理、设计方法以及影响系统设计重要因素，了解机器学习系统的开发与应用，更好地理解机器学习的概念以及机器学习基本方法，包括传统学习方法、深度学习和强化学习等。通过案例创新教学方法实践，培养创新型人工智能$^{+}$人才。

二、实验内容及原理

1、强化学习的基本原理

如图 1 所示，强化学习把学习看作试探、评价的过程。Agent（智能体）选择一个动作(action)用于环境，环境接受该动作后，状态(state)发生变化，同时产生一个强化信号，即奖励（reward）反馈给 agent，agent 根据信号和环境的当前状态再选择下一个动作，选择原则是使受到奖励的概率增大。选择动作不仅影响立即强化值，而且影响环境下一时刻状态及最终强化值。

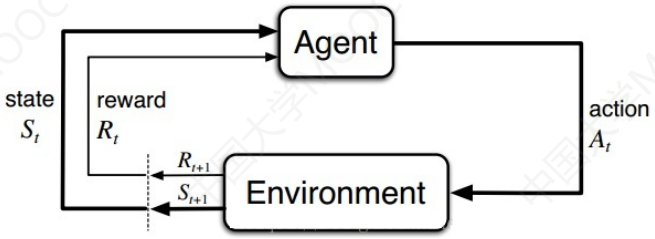


图 1：强化学习基本原理

2、OpenAI 捉迷藏系统的工作原理

（1）OpenAI 捉迷藏系统简介

OpenAI 于 2019 年开发了 Hide and Seek（捉迷藏）系统。一群智能体在一个虚拟环境中玩捉迷藏，它们能够通过不断学习与尝试，自创越来越复杂的作战策略，证明简单游戏规则、多智能体竞争和大规模强化学习算法可以促使智能体在无监督情况下从环境中学习复杂策略与技能。

（2）OpenAI 研究捉迷藏系统的动机

从生物进化角度来看，人类是一个可以不断适应新环境的物种，而人工智能却没有该特性。近年来，机器学习在围棋以及 Dota 2 等复杂游戏中取得显著进步，但这些特定领域的技能并不一定能适应现实场景实际应用。鉴于此，越来越多研究人员希望构建能够在行为、学习和进化等方面远超人类智能的机器智能。

（3）OpenAI 捉迷藏系统的实现目标

多个智能体通过竞争性自我博弈的不断训练，学习如何使用工具并使用类人

技能取得游戏胜利。

（4）OpenAI 捉迷藏系统的基本要素

如图 2 所示，OpenAI 捉迷藏系统的基本要素包括 3D 环境和角色。

3D 环境：一个包含斜坡、墙和盒子的 3D 环境。

角色：在 3D 环境中，智能体以团队为单位进行捉迷藏游戏，由搜索方（红色小人）和隐藏方（蓝色小人）组成。其中，搜索方的任务是紧紧追逐隐藏方，而隐藏方的任务是躲避搜索方的视线。

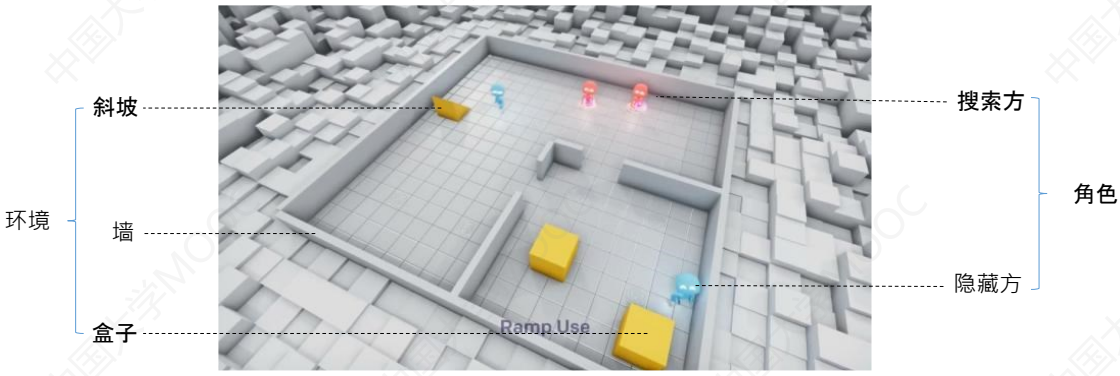


图 2：OpenAI 捉迷藏系统基本要素

（5）OpenAI 捉迷藏系统的奖励策略

赢得游戏的搜索方和隐藏方都会获得奖励，因此他们要不断地更新自己的策略。系统奖励遵循规则 1~规则 3。

规则 1：准备阶段所有智能体的奖励为零。

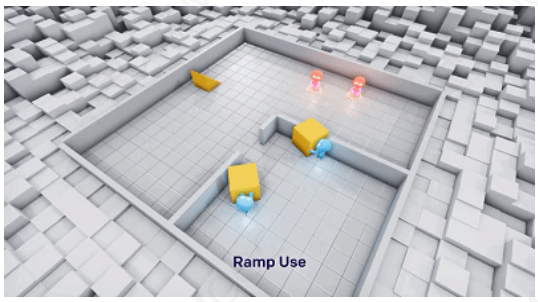
规则 2：隐藏方的奖励定义为：如果所有隐藏方成功隐藏，则奖励+1；相反，如果任何隐藏方被搜索方发现，则奖励-1。

规则 3：搜索方的奖励方式与隐藏方相反，定义为：如果所有隐藏方成功隐藏，则奖励-1；如果任何隐藏方被发现，则奖励+1。

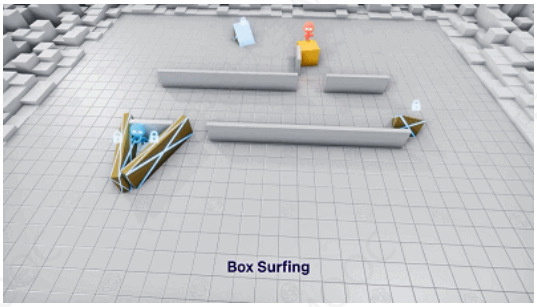
规则 4：智能体行为需要控制在合理的空间。如果超出游戏区域，则会受到惩罚。

（6）OpenAI 捉迷藏系统的学习策略

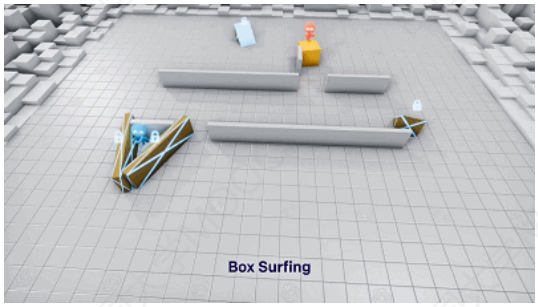
如图 3 所示，经过多轮训练后，智能体学会 6 种策略，即：①奔跑与追逐；②隐藏方学会移动砖块构建堡垒；③搜索方学会移动斜坡以跳进隐藏方的堡垒；④隐藏方学会将所有的斜坡搬进自己的堡垒；⑤搜索方学会跳到盒子上滑行，进而跳到隐藏方的堡垒；⑥隐藏方学会将所有的盒子锁定在适当位置，以防被搜索方利用。



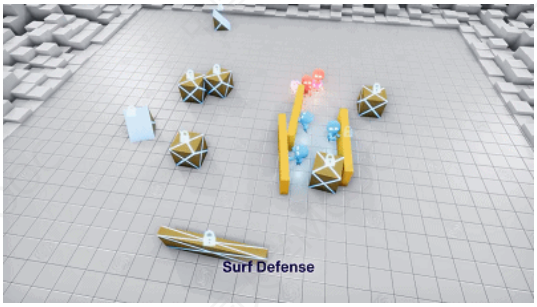
a) 搜索方学会移动斜坡以跳进隐藏方堡垒



b) 搜索方学会跳到盒子上滑行，并跳进隐藏方堡垒



c) 隐藏方学会将所有斜坡搬进自己堡垒



d) 隐藏方学会将所有盒子锁定在适当位置以防被搜索方利用

图 3：OpenAI 捉迷藏系统的学习策略

（7）OpenAI 训练捉迷藏智能体

如图 4 所示，每个智能体都使用自己观察和隐藏的记忆状态独立行动。智能体使用以实体为中心、基于状态的世界表征，也就是对其它目标和智能体是排列不变的。嵌入的每个目标被传递，通过一个 mask 残差自注意块，类似于 Transformer，其中的注意力集中在目标上。不在视线内以及在智能体前面的目标被 mask 掉，以使智能体没有它们的信息。

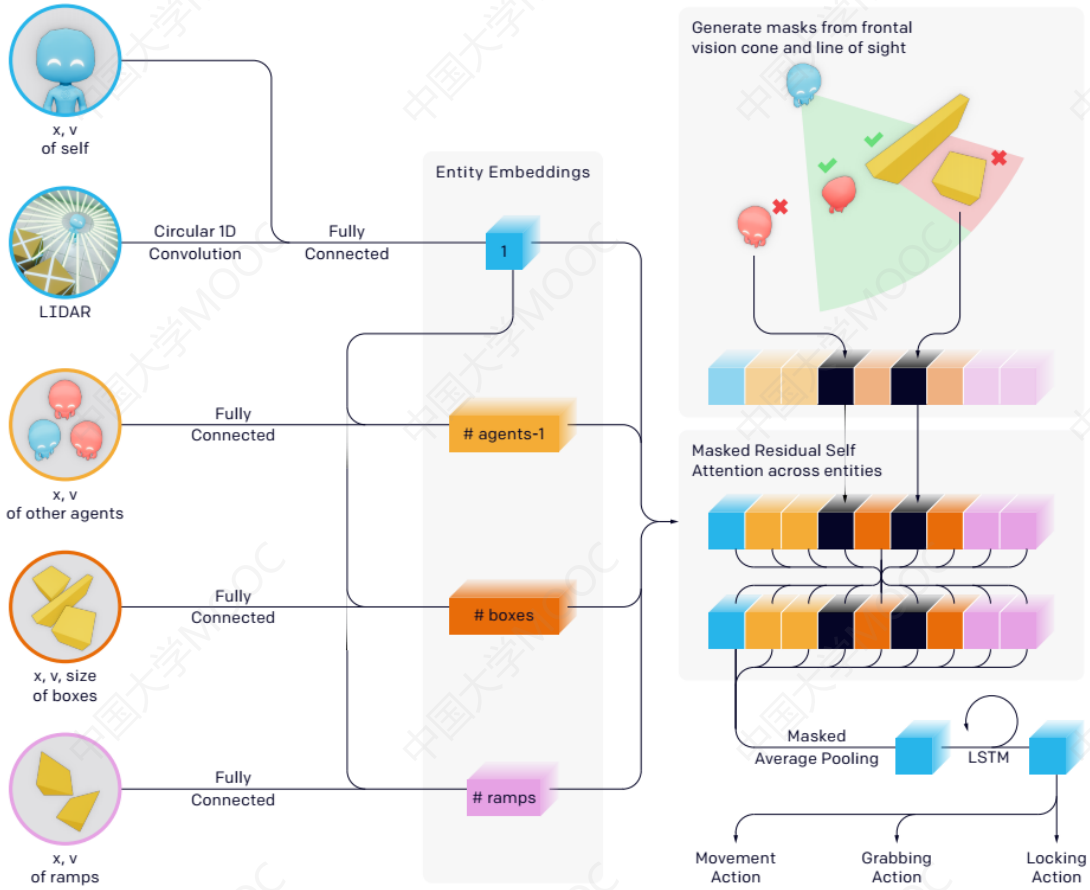


图 4：OpenAI 训练捉迷藏智能体的工作原理图

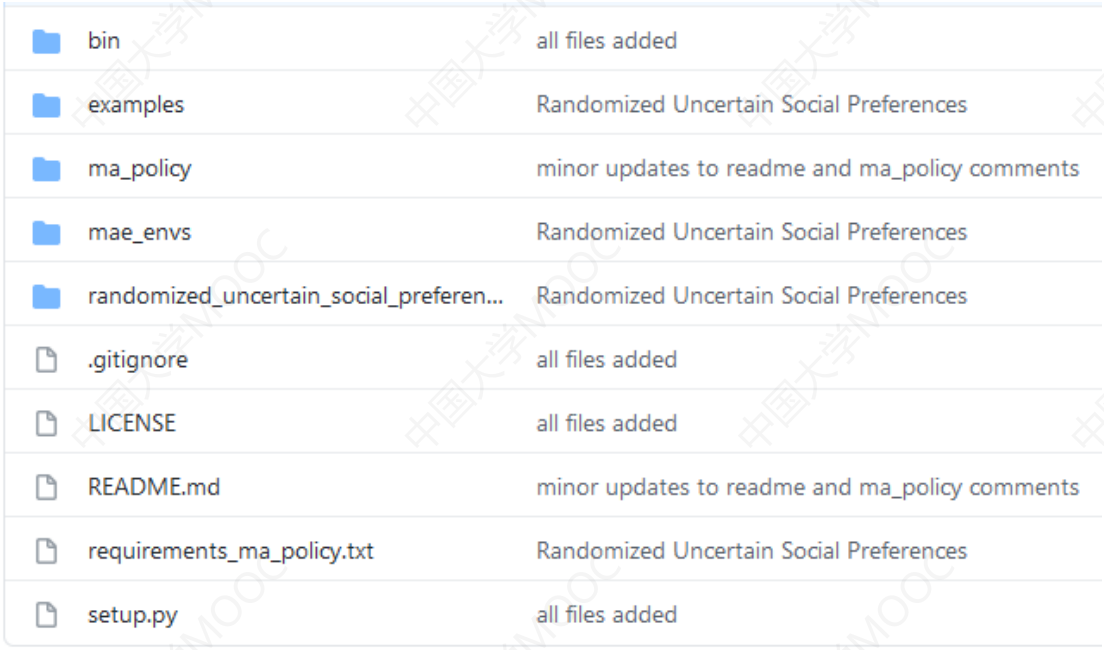
（8）智能体策略架构

通过自我博弈和临近策略优化（Proximal Policy Optimization (https://openai.com/blog/openai-baselines-ppo/)）训练智能体策略。在优化期间，智能体可以在价值函数中使用有关被遮挡目标和其他智能体的特权信息。

三、实验步骤

1、下载开源代码

从 https://github.com/openai/multi-agent-emergence-environments 网站下载系统的开源代码。



2、系统安装

系统存储库依赖于 mujoco-worldgen 包。因此，需要复制 mujoco-worldgen 存储库，安装它以及它的依赖项。

```
pip install -r mujoco-worldgen/requirements.txt
pip install -e mujoco-worldgen/
pip install -e multi-agent-emergence-environments/
```

安装环境：Mac OS X 、Ubuntu 16.04 和 Python 3.6

3、加载源程序

Hide and seek：mae_envs/envs/hide_and_seek.py

Box locking：mae_envs/envs/box_locking.py

Blueprint Construction：mae_envs/envs/blueprint_construction.py

Shelter Construction：mae_envs/envs/shelter_construction.py

4、环境测试

利用 bin/examine script 测试环境。

用法：bin/examine.py base

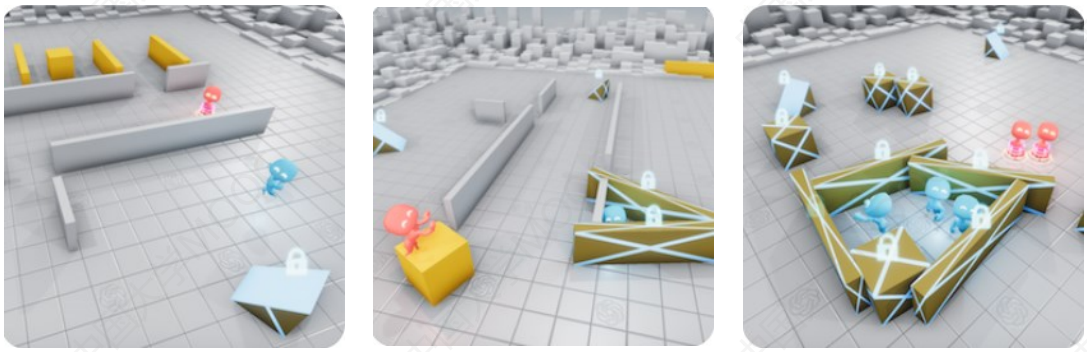
```
bin/examine.py examples/hide_and_seek_quadrant.jsonnet examples/hide_and_seek_quadrant.npz
```

四、实验结果与分析

1、系统功能运行结果

安装 OpenAI 捉迷藏系统，运行系统基本功能，展示如图 5 所示的系统奔跑和追逐、构建堡垒、使用坡道、斜坡防御、箱式冲浪、冲浪防御的基本功能。

a) 奔跑和追逐　　b) 构建堡垒　　c) 使用坡道



d) 斜坡防御　　e) 箱式冲浪　　f) 冲浪防御

图 5：OpenAI 捉迷藏系统的基本功能

2、学习策略所需 episode 和时间分析

分析智能体在各种 batch 大小情况下，例如学会图 3c 的策略（将斜坡搬进自己的堡垒）所需 episode 和时间。观察程序运行时如图 6 所示的以下实验结果。

①当增加 batch 大小时，是否可以提升收敛速度？

②当 batch 大小为 32k 或更高时，采样效率是否会受到影响？

③指定 episode 数量下，batch 大小分别为 8k 和 16k 时，智能体是否能学会图 3c 的策略？

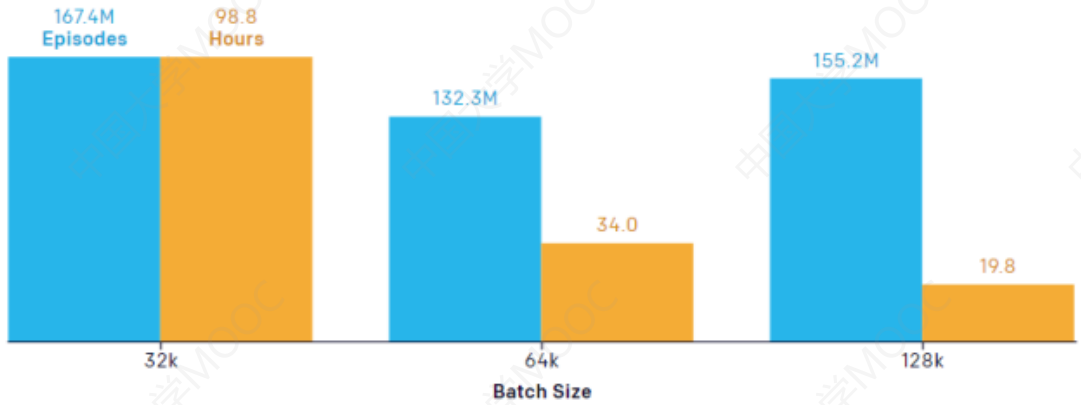


图 6：学习策略所需 episode 和时间分析

3、多智能体训练策略性能评价

比较状态表示的不同变体之间的智能体移动和对象交互的统计数据，评价、衡量多智能体训练策略的性能，观察程序运行时如图 7 所示的以下实验结果。

①单智能体、二维框位置（蓝色）；

②单智能体、盒子位置、旋转和速度（绿色）；

③1~3 个智能体，完整的观察空间（红色）；

④1~3 个具有完整观察空间（紫色）的智能体的 RND

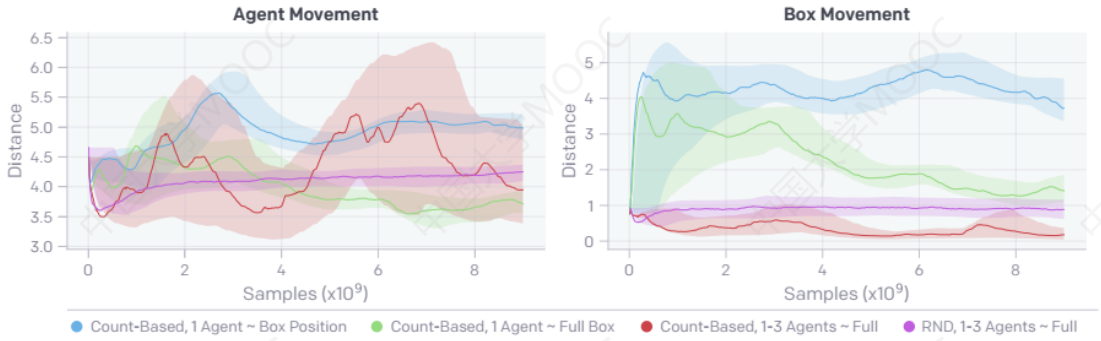


图 7：多智能体训练策略性能评价

4、迁移和微调评估

评价三个种子中的捉迷藏、基于计数和从头开始训练策略的任务套件的性能。观察程序运行时如图 8 所示的以下实验结果。

①Lock and Return、Sequential Lock 和 Construction from Blueprint 中，捉迷藏预训练策略的性能是否略好于基于计数和随机初始化的基线？

②Object Counting 的表现是否比基于计数的基线差？是否比 Shelter Construction 的随机初始化基线学习略慢？

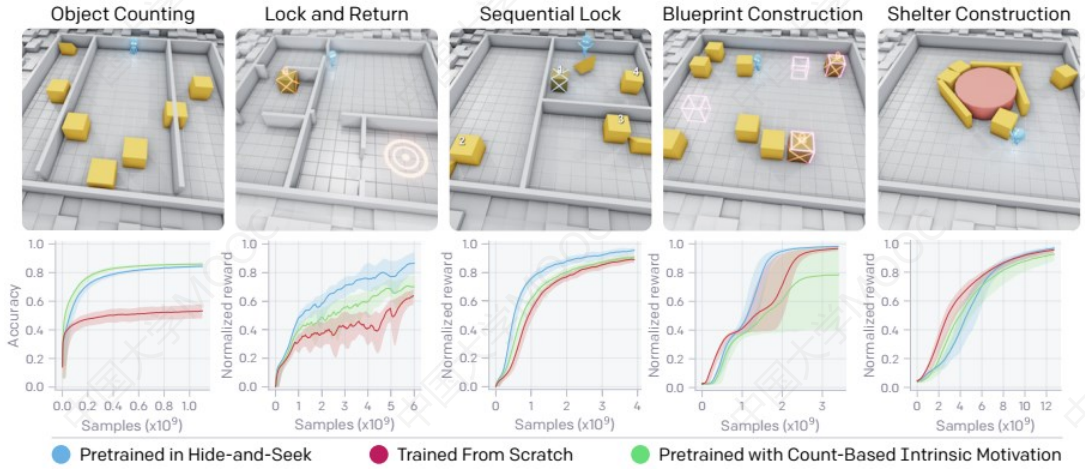


图 8：迁移和微调评估

五、实验要求

提交 OpenAI 捉迷藏系统实现的关键技术报告，实验报告模板见附录 1。

六、思考题

1、这款捉迷藏游戏除了有趣之外，通过这个系统，你还可以学到什么？

2、在该系统捉迷藏游戏功能的基础上，你还可以用 Python 开发哪些功能？

3、这个研究基础上后续还有哪些研究方向？

参考资料

1、论文

Bowen Baker, Ingmar Kanitscheider, Todor Markov, Emergent tool use from multi-agent autocurricula, 2019, https://arxiv.org/abs/1909.07528

2、开源代码

https://github.com/openai/multi-agent-emergence-environments

3、博客

https://openai.com/blog/emergent-tool-use/