课程设计选题

每个阶段从6个题目中任选一题！！

1. 德州扑克

介绍

德州扑克是一款风靡全球的扑克游戏，其中体现的概率和博弈的过程吸引着全球各地的玩家，具体规则如德州扑克。本课程作业只需要两人博弈，每人初始1000筹码，一方输光或者对局100局后游戏结束，剩余筹码多者胜。下注规则进行简化，每轮每个玩家只能下注1-10，且一轮只能加注1次。**all in作为拓展选择，需在作业中说明是否添加了该规则**。

阶段一

1. 编写代码实现游戏规则，
   a. 分发扑克，包含每个玩家的手牌和牌面。
   b. 玩家下注，注意下注顺序，加注以后的终止条件变化。如果有all in规则的话all in以后的下注变化，以及all in玩家能够赢得的筹码值变化。
   c. 最后赢输牌的结算。
2. 进行游戏环境可视化（也可利用命令行展示）。

阶段二

1. 演示打牌的过程，可视化需要能动态显示每个玩家当前获胜的概率，当前下注金额，奖金池金额，当前每个玩家剩余金额。（可视化参考，请忽略人）
2. 设计两个玩家，一个可以利用概率和人类玩家的经验，另一个可以进行适当弱化以体现不同智能体性能差异。

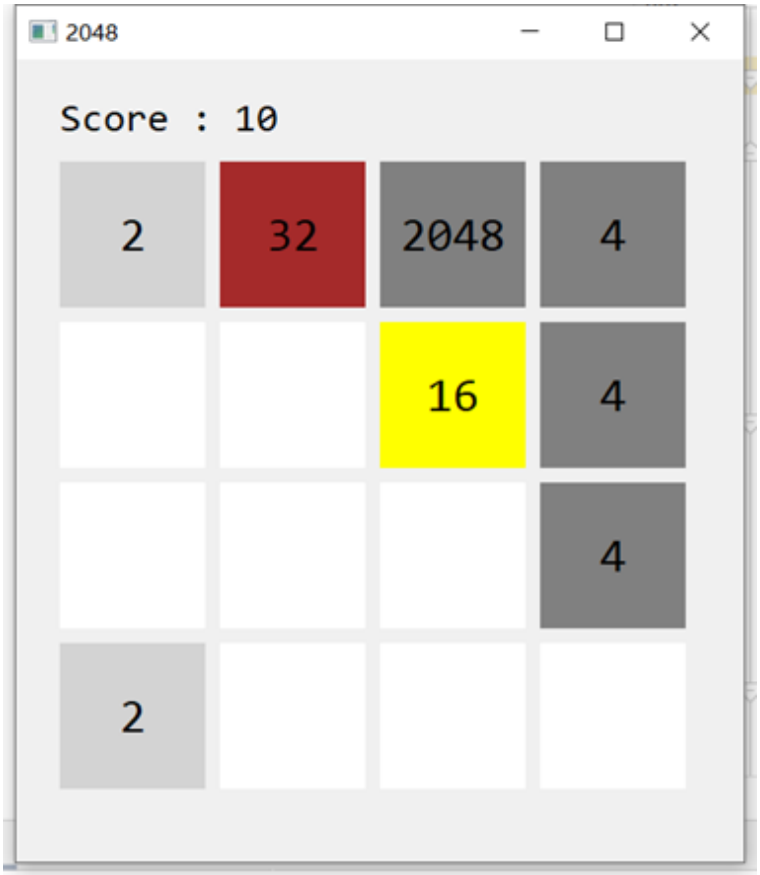
阶段三

1. 设计更好的策略或者利用机器学习的学习思想加强玩家性能，该阶段设计的玩家要能战胜第二阶段的玩家。（参考设计思路）
2. 尝试挑战3个玩家的情况。

2. 2048

介绍

2048是一款曾经火爆过的手机游戏，玩家在一个4x4的格子中进行游戏，玩家需要控制所有方块向同一个方向运动，两个相同数字方块撞在一起之后合并成为他们的和，每次操作之后会随机生成一个2或者4，最终得到一个“2048”的方块就算胜利了。



阶段一

1. 设计完成游戏规则
   a. 编写代码实现4x4的游戏环境。
   b. 方向控制，点击方向键以后所有有数字的格子向同一个方向移动，2个相同的格子相加，3个相同的只能合并靠近边缘的两个，4个相同的全部相加。
   c. 4x4格子中空白格子处随机生成2/4。
   d. 判断失败和胜利条件（无处生成2、4则失败）。
2. 可视化游戏（也可利用命令行展示）。

阶段二

1. 尝试使用贪婪搜索进行游戏。
2. 结合人的策略和经验进行游戏，在不停的失败中反思策略的不足，一步步完善策略，争取达到2048。

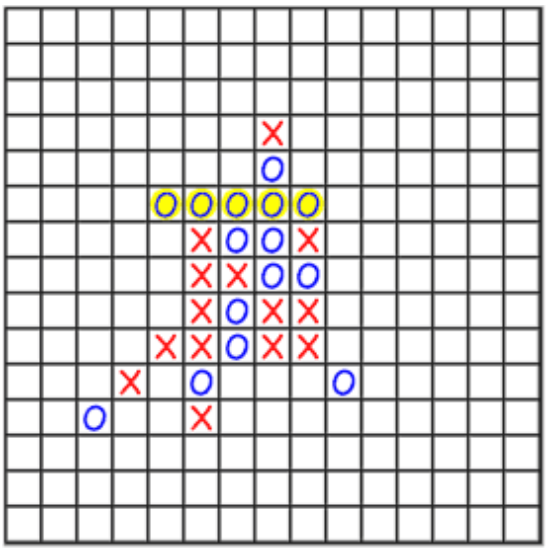
阶段三

1. 建议使用蒙特卡洛树搜索算法等高级搜索算法以更快的速度达到2048。
2. 使用机器学习算法达到4096或者更高（可选）。

3. 五子棋

介绍

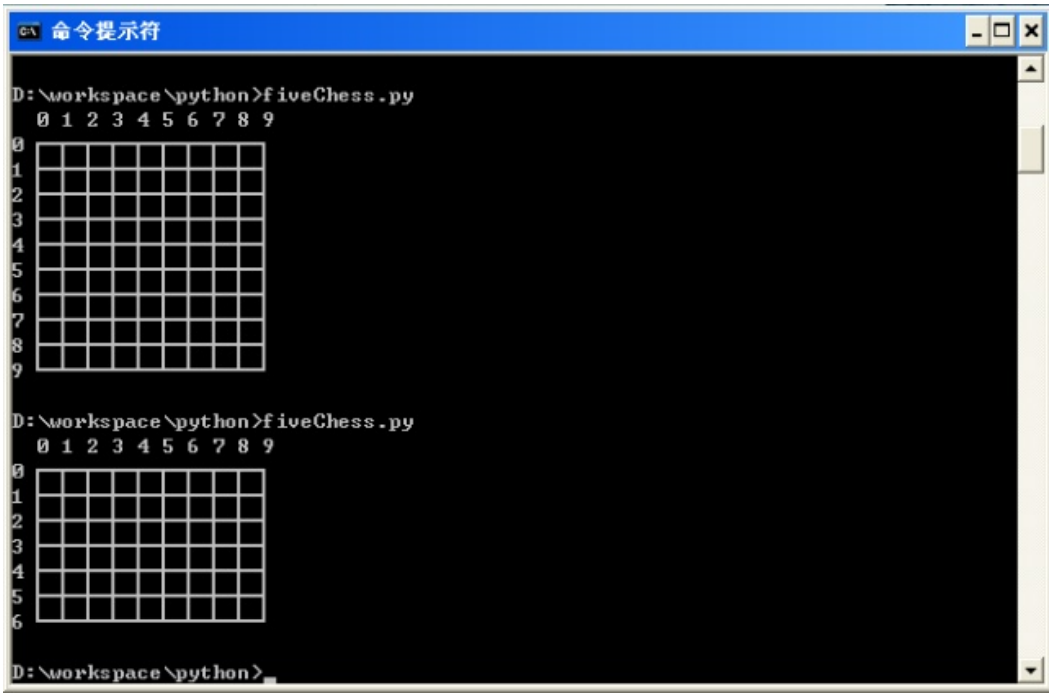
相信大家都玩过五子棋，其规则简单但富有趣味，先将5个相同棋子横向、纵向或对角线方向连接的一方获胜。示意如下：



本项目最终要求完成一个可与人类博弈的五子棋程序。

阶段一

1. 学习五子棋规则，自定义棋盘大小（15 x 15或更大）。
2. 完成可支持两位玩家轮流下棋并使用命令行交互的五子棋程序（可使用坐标描述落子位置）。



3. 在上述基础上添加图形交互界面，支持鼠标点击。

阶段二

1. 使用搜索策略（如min-max算法、alpha-beta剪枝优化等）实现五子棋对弈智能体
2. 程序可与玩家进行博弈

可参考：五子棋入门级AI的设计与实现

阶段三

尝试使用机器学习或深度学习方法训练模型来实现更强的五子棋对弈程序，如使用卷积神经网络、强化学习算法等。

可参考：

1. Learning a five-in-a-row policy using PyTorch
2. Application of Deep Reinforcement Learning in the Board Game

4. 吃豆人

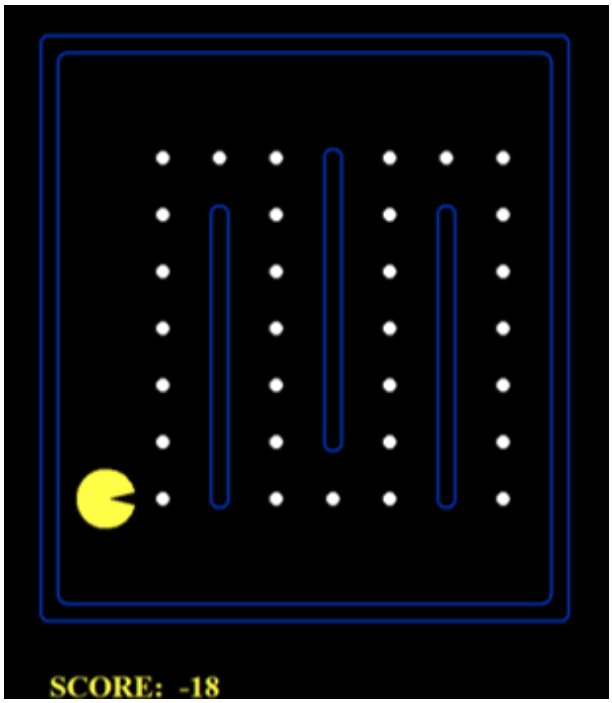
介绍

吃豆人（Pac-Man）是电子游戏历史上的经典街机游戏，由Namco公司的岩谷彻设计并由Midway Games在1980年发行。本项目中我们将复刻该经典游戏，并最终让吃豆人具有躲避幽灵👻的智能。

阶段一

完成游戏的基础功能（该阶段不要求添加幽灵）

1. 地图生成（可随机生成，也可根据设计好的配置文件生成）
2. 键盘控制吃豆人移动并吃掉豆子
3. 分数记录



可参考：

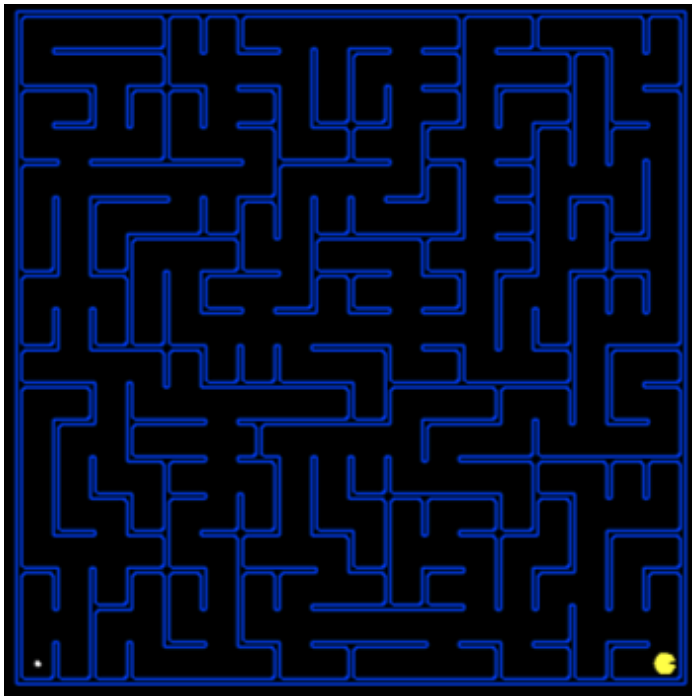
1. Pacman - Free Python Games 2.4.0 documentation

阶段二

设计自动寻路算法，让吃豆人在尽量少的步数下吃到所有豆子。建议尝试的算法如下：

1. 贪心算法
2. 深度优先线索（DFS）、广度优先搜索（BFS）
3. A-star算法
4. ... 更多的有待探索的算法

展示寻路过程，对比不同算法在不同地图上的特点。



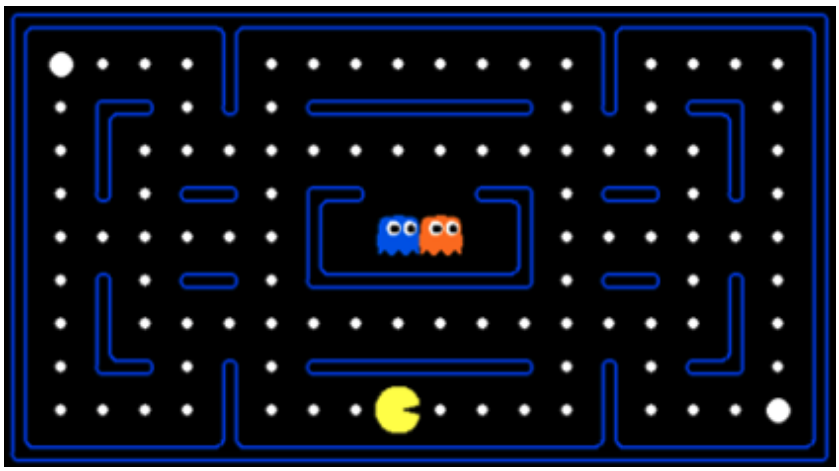
可参考：

1. Projects - CS 188:The Pac-Man Projects

阶段三

在游戏中添加：

1. 幽灵（ghost）：吃豆人碰到就游戏结束，其行动方式不可预测，具有随机性。
2. 胶囊（capsule）：吃豆人碰到可获得强化状态，在一定时间内能吃掉幽灵。



建议使用启发式算法或深度学习算法让吃豆人在环境中生存，并获取更高的分数。

可参考：

1. Reinforcement Learning in Pacman

5. 文本情感分类

介绍

文本情感分类的目标在于判断一段文本中用户表达的情感极性（positive、negative），如“我非常喜欢这部电影”就表达了**正面**（positive）的情感极性。这个任务在生活中有着广泛的应用，如利用评论文本分析商品的好坏、股票预测、舆情监控等。本项目旨在练习利用规则、传统机器学习、深度学习或者多种策略相结合的方式来构建性能优越的文本情感分类模型。

评论数据集和情感词典

下载地址： https://gikt1e63yh.feishu.cn/drive/folder/fldcnXTo1lBlgu1fW5NT7qonNke

说明：评论数据集分为训练集(train.tsv)、验证集(dev.tsv)和测试集(test.tsv)，数据集中每行记录为评论文本和对应的情感极性标签（1表示positive，0表示negative，文本和标签之间用tab符号即'\t'分割）。

阶段一

利用情感词典构建基于规则的文本情感分类模型

1. 熟悉情感分类的性能测试指标准确率（accuracy）及计算方法；
2. 读取数据集并打印评论文本和对应的情感标签；
3. 读取情感词典并打印情感词和对应的情感标签；
4. 利用情感词典来构造基于规则的文本情感分类器，对测试集中的文本进行情感分类并计算准确率。

提示：简单的规则可以是如果文本包含某个情感词就将情感词的情感极性视为文本的情感极性（当然这样可能导致分类错误）。推荐构建规则更复杂、更周到的情感分类系统（可以使用额外的情感词典）。

阶段二

构建基于传统机器学习的文本情感分类模型

1. 利用情感词典构建传统的文本特征表示，如词袋模型；
2. 熟悉朴素贝叶斯算法；
3. 利用朴素贝叶斯或者其它传统机器学习算法构建文本情感分类模型，对测试集中的文本进行情感分类并计算准确率。

阶段三

构建基于深度学习或者多种策略相结合的文本情感分类模型

1. 利用神经网络（CNN、RNN、Transformer等）构建文本情感分类模型，对测试集中的文本进行情感分类并计算准确率；
2. 在第1步的基础上使用各种方式来优化并提升文本情感分类的准确率。

6. MNIST手写数字识别

介绍

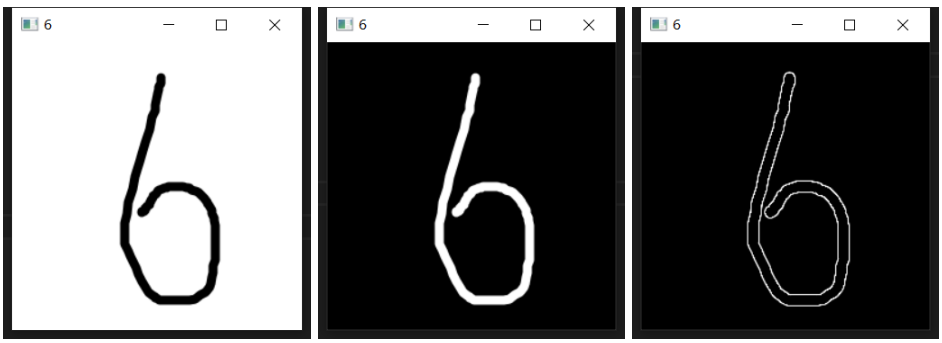
编写一个手写数字识别程序，最终目标是程序以png格式的图片为输入，需要能够自动识别并输出该图片对应的数字（0~9），如果编写了UI界面会有相应的加分，例如界面上可以浏览本地文件或者直接使用鼠标手写。

数据下载地址：MNIST.zip

阶段一

1. 使用任意画图工具手写一个数字并存储为png格式的图片，使用opencv库以**灰度图**形式读取该图片并观察图片在python程序中是如何表示的以及像素值与图片颜色的关系。
2. 将自己手写的白底黑字的图片转化为黑底白字的图片并显示。
3. 使用静态的卷积核实现简单的图像卷积操作（需要使用for循环手动实现，直接调用库函数不给分）。例如使用Laplace算子可以有效的提取出图片中物体的边缘，下方三张图片分别为Laplace卷积核，原图片，黑白反转后的图片和卷积操作之后的图片，可以使用不同算子并比较卷积后的结果有什么不同。

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



提示：

- 使用pip install opencv-python和pip install numpy 命令安装 opencv和numpy包。
- 假设要读取的图片为6.png，python以灰度形式读取该图片并显示的代码为：

```
import cv2
img = cv2.imread('6.png',0)
cv2.imshow('6',img)
cv2.waitKey(0)
```

- opencv中表示图片的方法是8位无符号整数类型的numpy矩阵，其中每个像素点的取值范围为0~255。
- 假设存在一个二维列表a，下列代码是将a转化为numpy矩阵以及一些简单的numpy操作：

```
import numpy as np
```

```
a = [[1,2], [3,4]]
a = np.array(a,dtype=np.uint8) # 将a转化为8位无符号整数类型的numpy矩阵
row, col = a.shape # 获得矩阵a的行数和列数
a[0, 1] = 8 # 将矩阵a中第0行第1列的元素修改为8
```

- 图像卷积操作的资料：https://zhuanlan.zhihu.com/p/43738099
- 卷积操作后的图片要与原图片保持一致，对于位于边缘的像素点如果有些邻域像素不存在可以假设值为0。

阶段二

1. 从下载的数据集文件中读取出图片并存储为三维的numpy矩阵，三个维度为（图片的数量，图片的高，图片的宽）；同时读取出对应数据集的标签并存储为一维的numpy向量；可以利用搜索引擎查阅MNIST数据集的读取方法。
2. 编写程序提取出各个图片的特征向量，自行决定如何构造图片的特征，可以使用图像的像素级特征也可以设计或参考一些图像特征的提取算法。
3. 基于步骤1中提取的特征使用任一种机器学习分类算法完成手写数字识别任务，要求模型在训练集上训练，并汇报模型在测试集上的准确率。
4. 自己手写一些数字并输入模型观察预测结果。

阶段三

1. 使用深度学习模型（如卷积神经网络）或其他更好的算法完成手写数字识别任务，并汇报模型在测试集上的准确率。