
*EE369 课程大作业*

# 2048游戏项目报告

姓名 段云智 学号 515141910019

2019 年 01 月 12 日

上海交通大學
SHANGHAI JIAO TONG UNIVERSITY

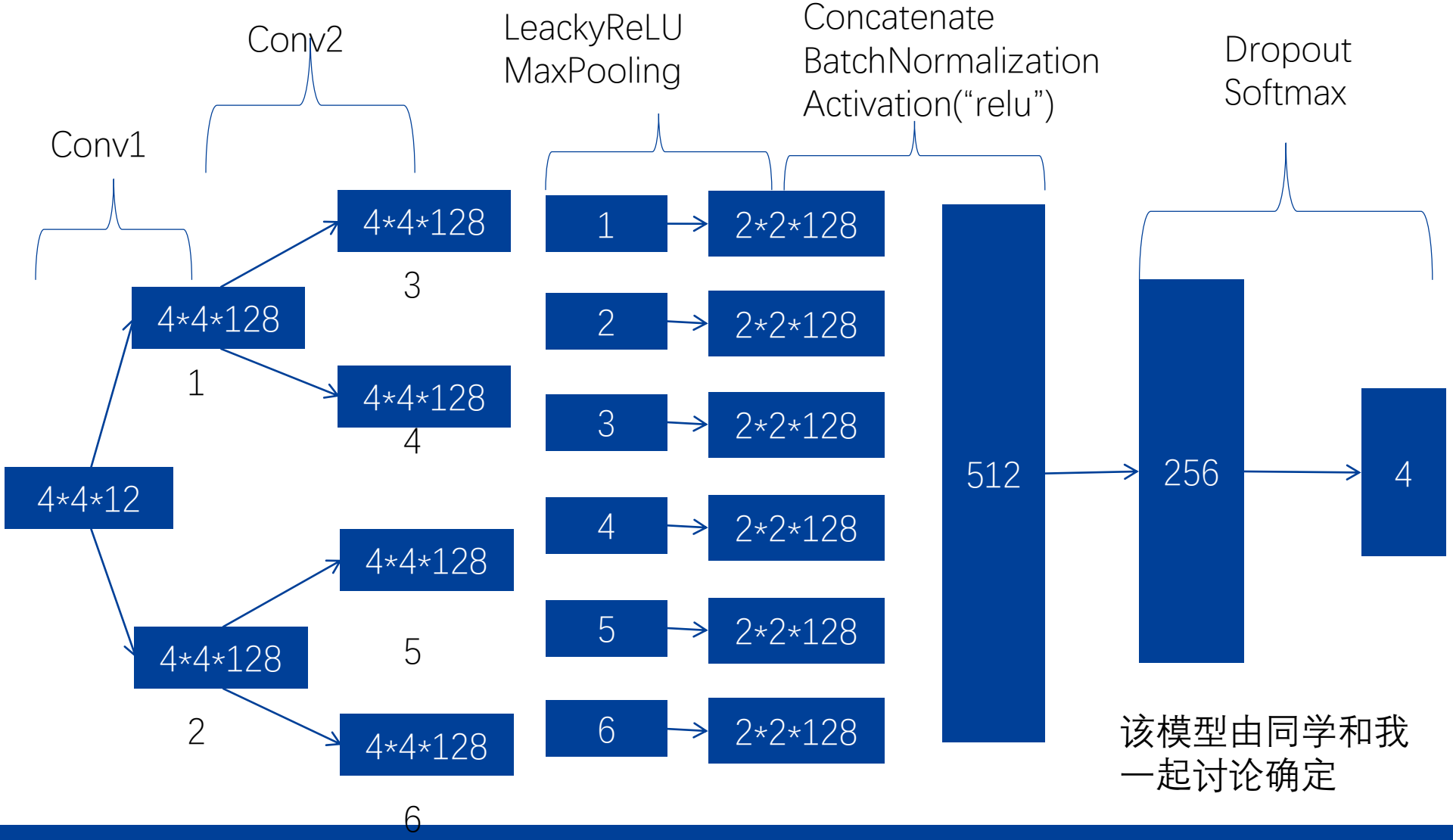
# 项目完成情况


- 50轮平均分数：541.44
- 加分档位：16分档位(基于 Planning 的方法超过1024分)
- 方法简述：采用卷积神经网络，训练模型控制2048游戏达到一定分数
- 代码框架： Keras
- 模型大小（MB）： 20.9MB
- 亮点：
  - 1. 使用深度学习中的卷积神经网络进行监督学习
  - 2. 使用离线+在线学习的方法
  - 3. 代码里实现了分层训练的逻辑
  - 4. 探索表征棋盘的函数，利用搜索树实现了2048的求解
- 代码链接： https://github.com/voldikss/EE369-2048-AI/

# 问题描述


- 项目背景：2048游戏
- 项目目标：使用监督学习方法训练模型控制2048游戏达到规定的分数。
- 项目优势：
  ① 具有有可以稳定达到2048的程序作为 supervision
  ② 完全 data-driven 的项目，训练数据量无限
- 项目难点：
  ① 棋盘的表示：one hot encoding
  ② 模型的选取：卷积神经网络
  ③ 网络结构：借鉴经典的CNN网络
  ④ FilterSize/ActivationFunction/Dropout/Pooling 的选择

# 模型设计

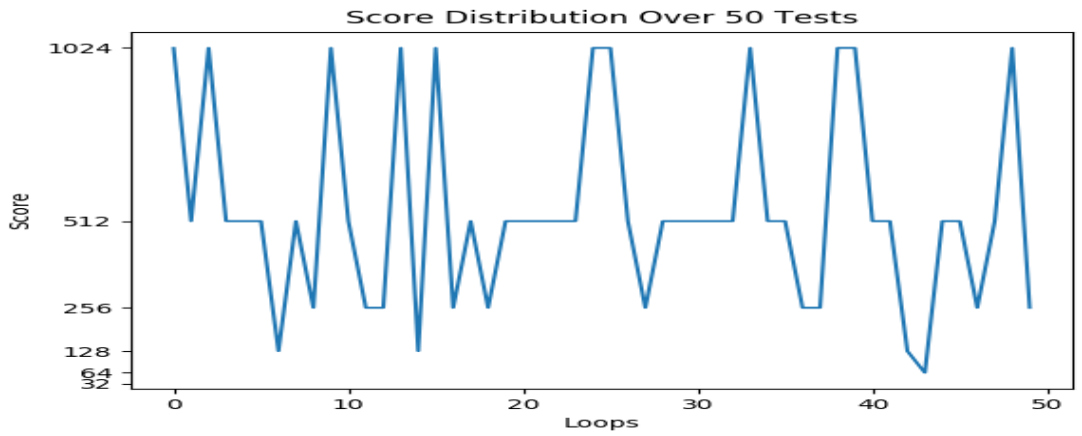



该模型由同学和我一起讨论确定

# 性能分析

- Agent成绩分布

如图




- Agent的稳定性。

可以看到，模型在1024

但分数都在128以上，未出现太低的结果。模型的训练时间过短，如果继续训练的话，预计效果会提升很多。

- 算法的复杂度
  - 单步预测时长：0.02s
  - 模型大小：20.9MB

# 技巧设计


- 使用在线学习：由模型作决策，由 ExpectiMaxAgent 的决策作训练数据
- 使用“离线+在线”的方法：先进行离线学习，能看到明显的效果。在后期分数出现瓶颈时再使用在线学习
- 使用分层训练：有点是降低难度，而且训练数据较为集中，便于加快收敛速度。但后期因为效果不太显著而弃用
- 进行 one-hot 编码时，棋盘的第三个维度最大数为12而非16，避免数据过于稀疏，有助于提升效果（待定）。

# 讨论


- 在训练过程中及时保存模型非常重要，必要时要保存超过一个备份，防止因意外而导致模型文件损坏
- 模块化的设计很重要：将训练部分、构建模型、测试模型、细节处理部分等封装成不同的部分，将能提高开发效率和便于后期维护。
- 多用类的继承实现自己需要的功能而不是直接改写原类，有很多好处。
- 必要时读框架源码有助于理解：读Pytorch源码解决了我对于其Y标签是否需要one-hot编码的疑问
- 框架文档很有帮助：Keras两种后端th和tf对于图片通道维的位置不同，文档中由注明这点。
- Python版本和框架的选择很重要，Python 不同目录下相互import需要格外注意。
- 想用强化学习的方法解决这个（使用一定的评价函数作为Reward我认为是更好的做法）

# Planning 方法

- **平滑性（smoothness）**

解释：相邻瓦片之间的差值尽量小，便于合并

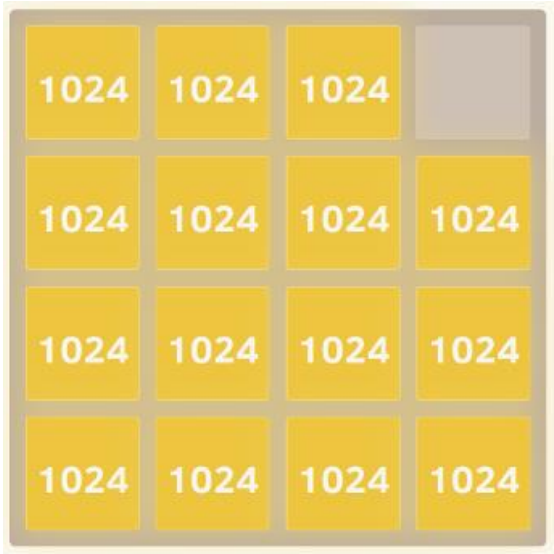
实现：

① 对棋盘求梯度

② 对梯度求绝对值

③ 对绝对值加和

优化：值越小越好




图自
https://stackoverflow.com/questions/22342854/what-is-the-optimal-algorithm-for-the-game-2048

# Planning 方法

- **单调性（monotonicity）**

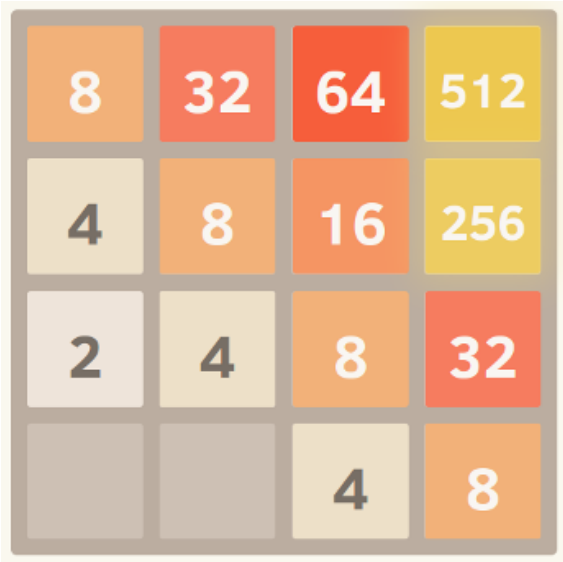
解释：瓦片的分布应具有单调性（经验法则：让最大的瓦片呆在角落）

实现：

① 构造 4*4 权重矩阵，在二维平面内单调分布

② 权重矩阵与棋盘格子点乘相加

优化：值越大越好




图自
https://stackoverflow.com/questions/22342854/what-is-the-optimal-algorithm-for-the-game-2048

# Planning 方法

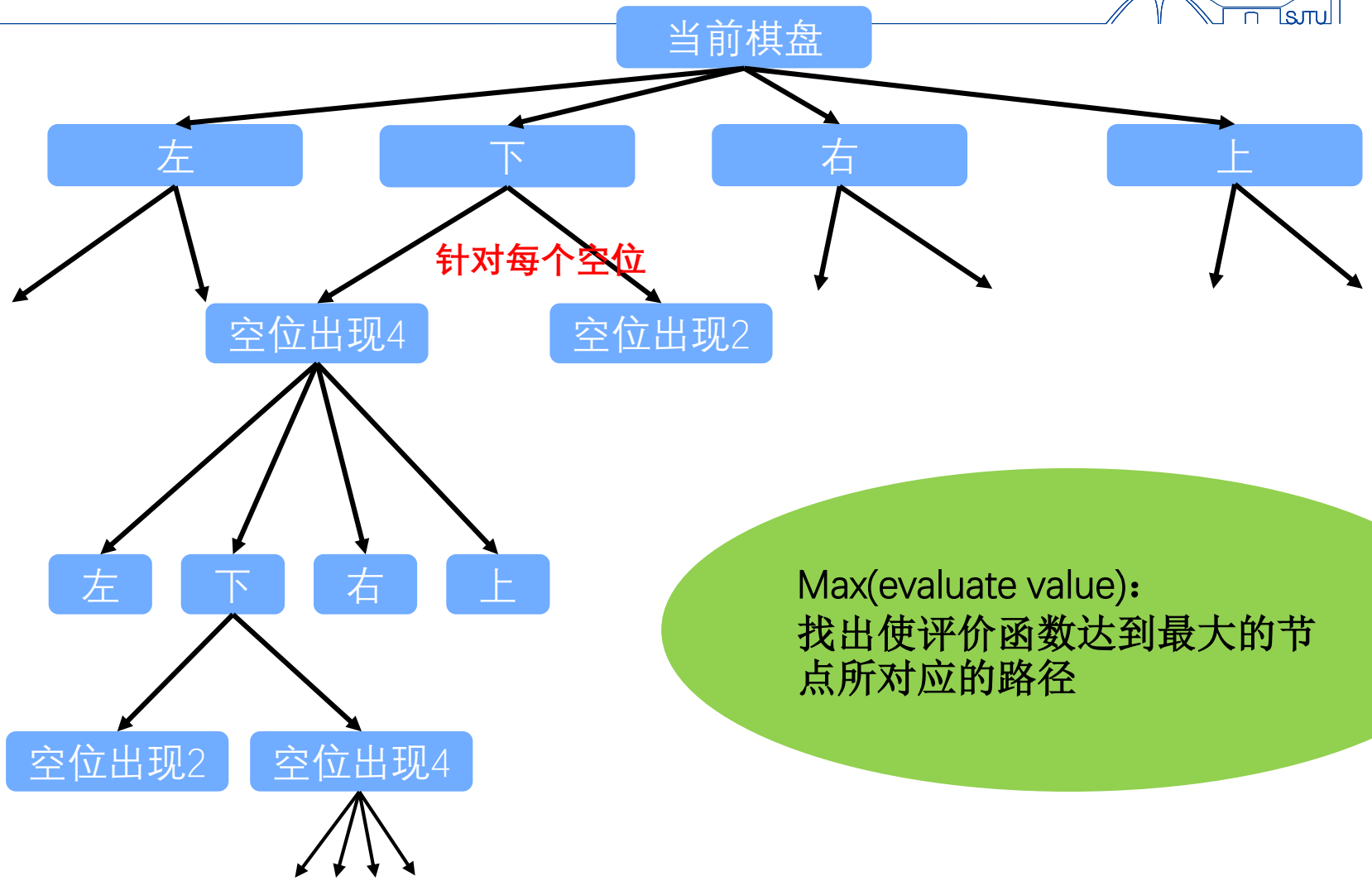
- **评价函数**
- 公式:

$$Alpha = smoothness - \lambda * monotonicity$$

- 实现:
  - 采用树搜索
  - 递归
  - 搜索深度越大，耗时越大，内存占用越大
  - 试验：深度为2时就可以达到1024

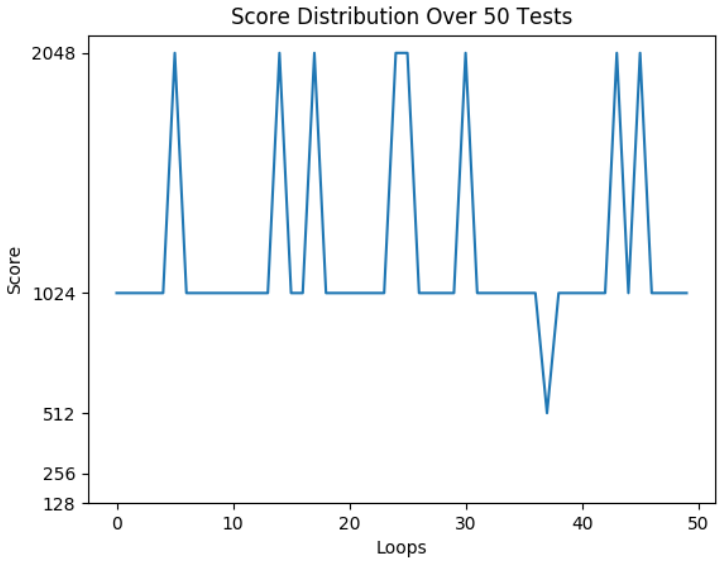
λ：保证 smoothness 和 monotonicity 的影响程度相近，一般取1即可

# Tree Search




Max(evaluate value)：
找出使评价函数达到最大的节点所对应的路径

# 效果及改进




- 如图是搜索深度为4时的结果，

基本可以稳定达到1024有时可达到2048

- 不足及改进
- 算法耗时巨大，深度越大耗时越大。
- 当前算法是无条件展开所有的情况，

可以考虑一些剪枝的算法，减少不必要的展开。

- 由于numpy对于小规模的数据并不能体现出其处理速度上的优势，因此可以把部分用使用numpy 的部分用Python重写。但是性能提升不会很多。
- 采用更好的评价函数等

# Thank You