

2048游戏项目报告

——基于模仿学习训练神经网络控制2048游

XX 杜天宇 516021910343

2018年 1月 13日





项目完成情况

• 50轮平均分数:149.12

■ 是否完成加分?(请注明加分挡位和具体情况):未完成加分

■ 方法简述:基于Keras的模仿学习训练深度神经网络从而控制2048游戏

■ 主要使用的代码框架:基于Keras的深度神经网络框架

■ 模型大小(MB):19.1M





- 亮点:请自己总结2-3点
- 1) 采用CNN提取信息,并且做了5次不同的卷积核
- 2)设置了学习率为0.3,减小训练模型过拟合可能性。
- 3) 代码结构清晰明了,结构简单易懂。

■ 代码链接:https://



问题描述



- 本次作业的主题,是利用机器学习的算法,令程序完成自主学习,学会如何玩2048游戏并尽可能达到更高的分数。
- 主要利用项目代码中提供的ExpectiMaxAgent作为supervision, 进行监督 学习(更确切的说是模仿学习imitation learning)
- 推荐使用的是深度神经网络算法。



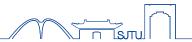
模型设计



- 模型主要是参考助教给出的模型,并加之以适当的修改。
- 基本框架是基于Keras的,其中利用到了Keras层的api,包括:
- 1) Input:用于实例化 Keras 张量。
- 2) Flatten () : 将输入展平。
- 3) concatenate () :连接一个输入张量的列表。
- 4) Dense () :全连接层。
- 5) BatchNormalization():在每个batch上将前一层的激活值重新规范
 化,即使得其输出数据的均值接近0,其标准差接近1
- 6) Activation ():设置激活函数。



模型设计



- 模型首先是对输入(也就是棋盘)利用CNN分别做128个filter的卷积核。 其大小为4*1, 1*4, 2*2, 3*3, 4*4, 5*5。再将这些结果平坦化后连接 成一个列表。并做规范化,设置激活函数为 "relu" (这在课程学习中 有所涉及,即 f(x) = max(0,x)),可以有效的缓解梯度消失问题。将 这一列表作为input layer。
- 然后分别建立含512和128个神经元的全连接层,激活函数仍为relu。
- 最后,建立含4个神经元的全连接层,激活函数为softmax,作为输出。
- 编译并保存模型。



性能分析



- 代码目录的result.py里记录了尝试的5次evaluate的分数。
- 取中位数成绩149.12为例:

Agent的稳定性不足。就以上数据来看,极差有15.04。同时在多次试验时,最少分数出现过109。





算法复杂度:由于是基于学习的agent,所以完成训练后其预测时间复杂度是O(1)。

- 单步预测时长:程序运行总时间15.9504s,共运行50次。每次运行时间 0.3190s
- 空间消耗:模型大小为19.1MB。其中包括input layer一共有4层。层与层之间均采用全连接形式。总参数:1587460,可训练参数:1586180。



技巧设计



■ Train方面也是基于助教的训练代码实现的。设计如下:

■ 1) 通过one-hot编码实现棋盘的表示。

2)模型设计主要考虑多个卷积核后全连接,保证尽可能的提取到棋盘的信息以供学习。

• 3) 训练时设置了学习率,尽量减小过拟合的可能性。





- 可能的算法改进:
- 1) 在训练的时候,适当改变学习率,做到逐层减少。这样或许可以减少模型在测试集里的泛化误差。
- 2)由于游戏是不断变难的,所以对于不同的难度,可以采用不同的模型,使得模型能够得到更好的拟合,而不用强求于对整体的拟合。



讨论



- 心得与感悟:
- 通过这次2048项目的具体实现,我更深的理解了深度神经网络的基本原理与架构。编程中,由于没有python相关经验,自己写起来十分吃力。因此代码主要还是在助教给的基本框架下修改完成的。在参考时,我仔细查询了Keras中文文档,了解到了keras是如何实现深度神经网络的。最终,我的编程能力得到了一定程度的加强。感谢助教让我能够为我提供的帮助!!

Thank You

