期中考试 -- 基于卷积神经网络的斗地主AI

问题分析

在最开始的时候，我提出的问题是如何将人工智能应用到**斗地主的策略生成**上。在学习机器学习这门课的时候，我一直在思考这个问题。前面学习的线性模型和决策树我感觉都太简单了，无法真正有效的解决我的问题，因为斗地主需要生成的策略很大，需要精确到出手牌中的哪些牌。而线性模型只能解决二分类问题，出牌的组合可能需要用到上百个普通的线性模型，而决策树一看就不靠谱，我觉得决策树主打一手过拟合，而且输入空间太大，层次划分的先后就会严重的影响模型的性能，而且这么多的出牌组合，我认为不是简单的划分就能解决的。就在我一筹莫展的时候，我们的课程进行到了 **神经网络** 的部分，我认为神经网络应该能够很好的解决我的问题。因为输入的是一个矩阵，所以我无需担心输入空间太大，只要将场上的情况向量化送进模型就行，然后输出的也是一个矩阵，这样我就可以训练模型让它输出的内容就是我需要出的牌的组合，这样就能规避之前模型普通分类的局限性，给出真正有效的输出。后面老师又讲了 `CNN` 卷积神经网络，这个东西通过卷积核提取出输入空间的特征，我觉得可以用它提取出斗地主中 "炸弹" 这样的信息，我觉得给神经网络加上 **卷积** 应该能让模型的准确度更进一步。接下来我将详细的描述我的实现计划，因为今天只带了平板，没有环境和数据进行实验，我不能保证这个方法真的可行，但是我肯定能保证他看上去可行。

数据输入输出的格式

输入格式

对于神经网路来说，最重要的肯定是对输入的数据进行编码，不然无法送进神经网路进行计算，我思考了一下斗地主场上的情况，首先在不考虑花色的情况下（斗地主似乎并不依赖花色进行），一共有15种牌面：A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, 大王, 小王，而且每张手牌最多只有四张，于是我打算将手牌的情况使用一个 15×4 的矩阵进行表示，这是一个 01 矩阵，大概长成这样：

牌的类别	A	2	3	4	5	6	7	8	9	10	J	Q	K	小王	大王
是否出现	1	1	1	0	0	0	1	0	1	1	1	0	1	1	1
是否出现	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
是否出现	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
是否出现	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

上面的矩阵表示的手牌序列是：A, A, 2, 3, 7, 9, 10, J, K, K, 小王, 大王

补直接表示数据而使用 01 矩阵的原因是如果后面要加上对花色的支持，就不用改动输入格式了。而且这样不用在训练过程中进行归一化，而且这样的话输出矩阵就能精确到每张牌出牌的概率。便于模型调整最后输出层的权重。

上面的矩阵只能表示 **手牌** 情况，但是斗地主显然不能只看着自己手上的牌打，我们需要场上的情况才能做出反应。我的解决方案是 **将两个矩阵拼接在一起** 于是 输入空间就是一个 30×4 的矩阵。如果你问我这样做的理由是什么，我只能说我猜的，因为刚学机器学习没多久，我也没有更好的解决方案了。在拼接之后就得到了一个 30×4 的 **输入矩阵**。

写到这我才发现这个模型无法考虑对手的牌数和他们之前出过的牌，只能根据对手当前的出牌和我们的手牌来进行推理。但是我没有想到在网络中记录之前出牌序列的方法，因为输入神经元是固定的，不能每次都将对手的手牌拼接在前面，只能先这样了，我相信随着后面的学习，我应该能找到解决这个问题的方案。

输出格式

输出格式就简单了，我们只需要输出一个 15×4 的矩阵就能完美的表示打出手牌的所有组合了。

数据收集

数据收集是一个比较麻烦的事情，我有几种方案。

方案一：和同学打斗地主，记录我们所有输入矩阵和我打牌的情况作为输出矩阵。但是很显然，这个方案相当低效，而且只限于我和同学打的情况，情况单一且数据量小。斗地主的问题比较复杂，我打算用一个大一点的模型，使用这种方法训练的模型似乎已经将过拟合这三个字写在脸上了。但是这个方法保证能够成功的收集到数据，在其他更好的方法都失败的时候，有一个这样的备用策略不失为一个比较好的选择。

方案二：我亲自游玩类似欢乐斗地主这样的在线对战游戏，如果游戏的数据包没加密的话，可以通过 **抓包** 的方式得到对方出的牌和我的手牌的数据。

方案三：但是抓包的行为可能破坏游戏的 `https` 加密，导致被封号。或者游戏对数据包进行了私有协议的加密，我们还可以用 **OCR** 的方式读取屏幕上的信息。我可以将我玩游戏的录屏拆分成帧，再对每帧进行OCR，来得到输入和输出矩阵。这样的话我还可以通过收集不同的游戏录像来得到更多的样本，我相信像 `BiliBili` 这样的视频网站上应该有不少游戏录像。所以这个方案应该是我的首选方案。

处理方法

经过前面漫长的数据处理流程，我们终于到了最关键也是最激动人心的模型构建时刻了。我的模型方案前面也提到过了，是一个 **卷积神经网络**。

1. 第一层卷积层，我的模型输入是 $1 \times 15 \times 4$ ，我打算在第一层先用 3×3的卷积核处理一下，这里我将使用 50 个卷积核。然后为了保证信息不丢失，这里先不进行 `MaxPooling`，得到一个 $50 \times 12 \times 2$ 的新图。
2. 然后再用 2×2 的20个卷积核再卷一次，得到一个 $20 \times 10 \times 1$ 的新图。也不进行 `MaxPooling`
3. 连接三层全连接网络，每个网络有100个神经元，这三层作为隐藏层。
4. 最后连接输出层，输出层的输入是100，输出应该是一个长度为60的向量，我们最后将它恢复为 15×4 的矩阵，这个就是我们上提到的 **打出手牌的输出矩阵** 。

这就是网络的整体架构，我觉得应该是足够应付我们的需求了。因为数据量比较小，太大的模型容易过拟合，上面的模型参数量应该还好。让后再对网络进行梯度下降和反向传播，最终得到一个模型。

但是这个模型估计还是有点弱，我稍微思考了一下，既然有了模型就不用我亲自生成数据了。我可以将之前OCR收集数据的程序稍作修改，将其改造成一个自动玩游戏的脚本，然和让它与真人进行对战。这样就能在被封号之前生成数不清的数据，我们从总共的数据中挑出**赢下对局** 的数据，再用这些数据对模型进行 **进一步** 训练。经过多次这样的过程，这个模型大概会比我强吧。

总结

我觉得从直觉上来看，这个方案应该能达到不错的效果，当然也可能不如托管的贪心算法。不过我已经尽力了。后面需要改进的可能是标注自己的角色（农民/地主），以及对前面对局出牌的记录。因为只有两个小时边思考边写作，我将我的部分思路历程都直接写上去了。希望这个模型能达到不错的结果。