双升AI实验报告

李家成 2100012973

环境使用

本实验基本使用了原有的环境模型,然而,我们的主要贡献在动作生成器上,若将其视为环境的一部分,则可以说对环境进行了部分修改,详见后文.其余环境除修正了部分已向助教反馈的bug外均来自样例代码.

特征处理

同上地,我们基本使用了原有的特征输入,可以将我们对动作生成的精细化视为加入了逻辑先验知识的对次要特征的处理,并以此优化了主要特征option_mat.

算法模型

使用了样例代码提供的三层卷积模型并使用PPO进行训练,尝试调整了模型、算法、超参数,效果均不明显.原因主要包括以下两方面:一者,经过人工处理的动作生成已经能产生相当好的策略;二者,只通过本地1块GTX 1650训练模型实在有些困难,在较长的训练时间下仍然只有较少的迭代次数(而且电脑快烧了..).综合以上两方面,我将重点放在了动作生成器的设计上,并为之后在合适的训练资源下进一步训练提升留下了空间.

动作生成

这是本工作的核心贡献,修改主要集中在mvGen.py中的move_generator类中.通过修改推理或训练过程中的get_action_options()调用的接口即可使用更强大的move_generator生成更好的动作,使用它的bot在班级内部天梯赛和botzone的总榜上均能取得显著优势.

动机

原始的move_generator类逻辑是通过搜索的方法生成全部的可行动作,并将其中前54个处理成为option_mat特征.这带来了多方面的问题:

- 相似动作难以区分:在打小牌时,打2或3通常不会有太大的区别,但多个相似的动作可能会使模型陷入区分它们的困境,而忽略了与之不同却显著重要的其他动作(例如打大牌),需要对相似动作进行聚类剪枝.

- 优秀动作惨遭mask:特征维度限制了每次最多可选动作数为54,在许多时候这大于全部合法动作数量.然而,面临对牌甚至甩牌时,合法动作数量可能远远大于54,而原有的动作生成方案排序使得许多优秀的动作会被忽视,从而模型只能从许多很差的动作中选择一个(例如在对方打副对时垫掉2张较大的主牌),需要获得好的顺序.
- 没有调用历史信息:双升游戏作为多回合的不完全信息博弈,在考虑出牌时,可以通过将全局历史信息结合规则限制和理性假设推出许多重要信息并以此指导出牌,因此合理的出牌方式往往远少于合法的出牌方式(例如上手时先出A).在生成动作时引入历史信息对上面提到的动作剪枝和搜索顺序都有帮助.

综上,我们修改动作生成器,让option_mat特征变得更小且更好.

历史信息

我们在构造move_generator时传入全部历史信息,从而可以通过模拟的方式还原牌局,一方面知晓自己的手牌并预处理分类成单牌、对子、连对,另一方面维护所有已打出的牌,并进一步确定重要信息:其余玩家没有的花色、某门花色最大的牌、某门花色最大的分等.这些都是人类玩家在对局中常常关注,且用于指导决策的重要信息.

hard-生成

即在需要出牌时,基于规则遍历决策树,生成**唯一**的出牌动作.

- 在先手出牌时,只需要选出重要性最高的动作(连对>大牌>队友缺门>某门最后一张>...)
- 在后手出牌时,由于本轮应出牌型已由规则确定,还需要考虑动作的合法性,这要求启发式逻辑要考虑合理和合法两方面.这要求我们在合法的动作的基础上再通过历史信息选出重要性最高的动作.在写了巨量的if-else和经过艰苦的debug后,现在的bot可以在除了极少数corner-case(例如长连对)的情况下做到动作合法,并在几乎所有情况下选择出先验规则认为最合理的动作.

hard-生成的方式模拟了一个冷静且死板的人类玩家的出牌方式,由于双升游戏本身合理动作少的特点,用这种方式生成动作的bot可以在许多对局中有不错的表现,在botzone上的bot中达到了sota.然而,这种生成方式的内在缺陷是,固定逻辑无法在所有情况下做到最优,某些时候需要在常规操作的基础上“灵光一现”达到更好的效果;并且这种生成方式也没有考虑到对队友策略的动态适应.

soft-生成

为了克服上面提到的缺陷,也为了降低编码难度,我们考虑放松对出牌动作的限制,仍然提供少数可选动作并学习进行选择.

在hard-生成中,我们本质上是在:通过判断函数进行决策树结点转移,在每个结点通过生成函数生成对应的动作.而可能生成的动作事实上只有有限种.于是,我们仍然利用生成函数生成可能的动作,但弃用判断函数,不再生成决策树,相对地,我们将所有可能生成的动作全部作为可选的动作(打大牌、打小牌、打分牌、...).通过之前的分析,我们利用生成函数内部的逻辑在每种分类下选择了代表性动作,生

成的动作可选项是一系列出牌思路下的代表性动作,于是模型可能可以关注学习“该出大牌还是小牌”,而非“该出2还是3”.并且,这样就不用通过写复杂的if-else逻辑选择动作了,而是通过神经网络模型学习进行选择.我们对于自己先手出牌和应单牌的大多数情况进行了这种调整,而一次性出多张牌仍沿用之前的方案.

奖励函数

通过奖励函数指导模型学习是强化学习的核心思想.样例代码提供的奖励函数是本轮的分数(区分正负),但这个奖励函数相对稀疏,且容易使模型具有强大的贪心偏好,经过尝试,我们新增了以下两个奖励:

- 鼓励上手:赋予每轮打出牌最大者微量奖励，即使没有分数牌.
- 鼓励整形：当垫牌使得手牌形状变好，赋予微量奖励.

这些reward的数值需要调整,使得不影响获得尽可能多的全局分数的目标.奖励调整也可以作为未来的工作的一部分.

其他环节

尽管出牌是双升的核心环节,亮牌和埋牌的好坏也对游戏的进行有重要影响.我们考虑将来分别引入单独的模型来学习亮牌和埋牌,学习方式可以是基于后续出牌模型的联合学习,或者是基于人类数据的监督学习.现在我们暂时使用启发式策略.

亮牌

亮牌通常根据自己的手牌和其他玩家的亮牌联合决定.Botzone比赛上的级牌固定为2,且亮主者为庄家,这对于我们的强力模型是有利的.因此我们采用了较为激进的启发式策略.每次摸牌后根据大牌数、对数衡量各花色强度分值,若某花色达到一定的分值且能亮就亮,若被亮到的花色不强就反.思想是若我们在比较早的时间已经在某花色取得了不错的手牌,那么在摸完牌和可能地获得底牌后该花色会更好,以此保证较强的主牌.实战检验效果很好.

我们的策略不会主动打无主,事实上,尽管在设计move_generator时有所考虑,我们仍没有经过无主局的测试,也不能保证无主局时的合法性和合理性。由于无主局的策略应当和有主局有显著的差别,我们考虑之后开发无主局专用的模型.

埋牌

我们的策略尚不能很好地守护底牌,因此我们采用保守的埋牌策略:尽量埋副花色散牌,但不埋分,除非副花色散牌均只剩分.这样我们可以保证在大多数情况下不用由于被抄底失去大量分数.但显然优秀的人类玩家会根据手牌的具体情况适当地选择埋分策略和守底策略,因此这也是未来的改进方向之一.

为了利用已有的分析手牌形状的函数,我们将埋牌函数写在了move_generator中.

总结分析

针对双升游戏,我们提供了强大的move_generator,它可以通过hard或soft的方式生成与历史信息相关的可选动作特征option_mat.使用这种方式建构的bot拥有强势的表现,甚至可以接近人类水平,可以作为未来一段时期的baseline.由于算力原因,我们对训练的探索仍不充分,可以直接以我们的bot为基础进行finetune,或者利用我们提供的move_generator类及其函数构建合适的训练方案.我们的bot还可以快速生成大量质量较高的对局,可以为监督学习提供数据.期待未来能够构建出更强大的bot.