如何才能在掷骰子游戏中必赢？

简介

掷骰子猜大小是一个庄家概率必赢游戏。但这个结论只是建立在每次竞猜是独立事件的前提下。如果玩家改变竞猜策略，每次竞猜金额在之前竞猜的结果基础决策，会不会使这个游戏变为玩家必赢游戏？比如每次竞猜金额为之前所有的亏损额再加上自己预期的收益额，那么假设有足够多的本金，猜足够多次，只要能猜中一次，玩家就必然能得到预期的收益额。 此“必赢”竞猜策略要求玩家有足够多的本金，不然可能会存在本金用完之前，还没有赢到预期的收益额的后果。我们能不能有一个比上例更优竞猜策略，使得玩家（在更低风险的限制条件下）必赢？本文通过参考AlphaGo Zero的思路, 利用神经网络和蒙特卡洛树搜索算法（Monte Carlo Tree Search, MCTS），只根据掷骰子的游戏规则，生成最优的竞猜策略，解开掷骰子游戏的玩家必赢的奥秘。本文相关的代码在：<https://github.com/liangfengsid/dicegame>3，欢迎关注。

一. 庄家必赢的掷骰子

大家都玩过或听过掷骰子猜大小的游戏。因为存在三个骰子点数相同时庄家通吃的情况，玩家无论是猜大还是猜小，赢的概率都是小于50%（约为48.6%），这是一个庄家概率必赢游戏，这也是这个游戏存活的保证。因此，在玩家每次等额（或随机数额）竞猜的情况下，长期玩下去，玩家最后的预期都是输的，可以粗略地认为，100元的本金的预期剩余只有约97.2元，或者说，1000个玩家里，预计约486个玩家最终能赢X元，剩余的514个玩家会输X元。

但是概率归概率，任何游戏都有它的技巧。在掷大小游戏中，玩家能不能也通过技巧（策略）来做到必赢或者大概率赢？这个技巧如果从每次竞猜是猜大还是猜小去思考，结果显然是徒劳无功的。因为每次掷骰子出现大或小的情况，相对之前出现大小的情况都是独立事件。不管之前连续出现了100次大，下一次出现大或小的概率都为48.6%，而不是更大概率地结果为小。当然，这是在骰子是正常构造的，且为人为控制其结果的前提下。相反，如果某个骰子由于制造工艺的原因导致出现结果偏向于为6，那么这种情况下，连续出现100次大以后，下一次出现结果为大的概率仍然比小要更高一点。但是这种骰子的差异或人为干涉的情况由于属于非正常游戏且不可估量，就不在本文的分析范围内。

虽然无论猜大还是猜小，玩家都是小概率赢且猜大或小其实无关紧要，但是玩家每次竞猜的数额还是可以控制。如果玩家每次竞猜金额为之前所有的亏损额再加上自己预期的收益额，那么假设有足够多的本金，猜足够多次，只要能猜中一次，玩家就必然能得到预期的收益额。原来有如此好用的竞猜策略，那么如果玩家按照这个策略去玩掷骰子游戏，庄家是不是可以破产了？答案似乎是，但好像又不会有那么便宜的事。这个竞猜策略在下一节中将归纳为“弥亏”策略。在 此竞猜策略假设玩家有足够多的本金，不然可能会存在本金用完之前，还没有赢到预期的收益额的后果。这个假设条件对玩家实际做到“必赢”影响如何，下一节将作分析。

总之，上述的庄家概率必赢结论，是建立在每次竞猜都是相对独立事件的情况下。如果每次竞猜的选择会根据之前的竞猜结果作出调整，上述庄家概率必赢结论未必成立。会不会存在一种竞猜策略，使得玩家必赢呢？我们在下一节来讨论玩家必赢的竞猜策略。

扩展一点关于掷骰子庄家必赢的原理：

一般来说，在此游戏规则的限定下，玩家必输。在常见的掷骰子游戏中，由三个骰子投出，竞猜它们的正面显示数字的情况。每次竞猜由两部分构成，一个是注种选择（包括大、小、三个一，三个二等），一个是竞猜数额。每次根据掷骰子的结果，玩家会得到一定倍率的回报或者失去竞猜数额。值得注意，回报的倍率乘以对应注种出现的概率必然小于1。如注种为小的情况：三个骰子的和为4至10，其中不包括三个骰子数字一样的情况，其出现概率为p=(6\*6\*6 – 6)/ 2/(6\*6\*6) = 48.6%，其倍率为2。投大投小概率都一样，为了简便，在不考虑选择除了大小外的其它注种的情况下，长期竞猜的玩家剩余数额期望为：

，其中ci为第i次的竞猜数额，C为玩家竞猜的总本金。从这个角度看，无论每次的竞猜数额是多少，长期下来玩家的剩余数额都是比本金少的。r

二. 玩家必赢的竞猜策略

下面的讨论过程会涉及数学归纳和概率的知识。

首先我们看一个玩家必赢的充分条件：假设存在一个竞猜策略，当玩家预期的收益额是X，通过这个竞猜策略决定注种和数额，能让玩家在一定竞猜回合内必定能赢得不少于这个预期的收益额的回报。

如果具备这个充分条件，那么玩家就可以在每一轮（包括多个竞猜回合）利用这个竞猜策略赢得不少于X数额时，立刻终止一轮竞猜，然后从头开始，在下一轮中重复使用此竞猜策略。那么，玩家可以通过无限轮这样的竞猜中，赢取无限多X数额的回报。

如果有上述的那么一个竞猜策略，聪明的玩家可以通过掷骰子游戏获取无限的回报。那么，究竟有没有这样一个完美的竞猜策略呢？一个半正确的答案是，有的。在特定的前提下，玩家可以做到必赢。

一个在特定前提下的“弥亏”竞猜策略：假设玩家有足够多的本金，在每回合竞猜前，假设玩家之前所有回合累积亏损数额为Y，玩家每次竞猜的注种默认选“小”，竞猜数额为X+Y。

弥亏策略是上述必赢充分条件的一个例子，而它本身的特定假设就是“玩家有足够多的本金”。例如，玩家的预期收益额为100，第一回合开始时，玩家亏损数额为0，那么根据弥亏策略，玩家第一回合的竞猜金额就为100。如果此回合竞猜结果为小，玩家除本金外收益100，本轮竞猜可以结束，玩家可以开始下一轮竞猜，下一轮竞猜亏损数额重置为0。如果此回合竞猜结果不为小，那么玩家亏损数额100，在第二回合的竞猜金额为100+100=200。如果第二回合玩家竞猜还是不中，玩家累积亏损数额为300，那么在下面的几个回合里，玩家竞猜的金额分别应该是400，800，...，2i-1\*100，直到玩家竞猜中一次，然后结束本轮竞猜。可见，假设第n回合玩家能第一次赢，玩家的本金就必须能覆盖之前n-1回合的累积亏损数额和第n回合的竞猜数额，即本金应满足：，或换句话说，如果在n回合内玩家还没有第一次赢，玩家就会面临不能挽回亏损，甚至全部本金亏损的可能（取决于第n回合竞猜金额不足时，玩家是选择放弃竞猜还是all in）。

那么，在本金有限，可能第n回合余额不足以执行策略，但选择all in，可能亏损全部本金的情况下，玩家使用弥亏策略的期望余额是：

. 具体地，p为0.486，当玩家本金C=10000，期望收益额X=100时，n=7，Br<Bo=9980.01 =0.998C<C。看来，使用弥亏策略，在本金有限的实际前提下，玩家能获得比随机或等额竞猜更高的期望余额，但是相对本金来说，余额仍然是负数的。在本金有限时使用弥亏策略长期玩下去，玩家仍然会输。

换个角度，如果想要在本金有限情况下使用弥亏策略实现概率必赢，那么要求每次竞猜玩家赢的概率p要多少呢？只要让Bo>=C, 在上述的本金和期望收益额下，可以得出p>=0.5。不难证明这也是使用随机数额或等额策略时，想要做到玩家概率必赢的游戏设置前提。本金有限的弥亏策略可以比随机数额或等额策略输得少，但是玩家想要赢，这几个策略对游戏规则的改变的要求程度是一样，即每次竞猜都要有一半的机会赢。

弥亏竞猜策略比随机或等额竞猜策略有更高的期望余额，让我们看到了更好的竞猜策略能在掷大小游戏中获得一定优势。那么，有没有其它的必赢竞猜策略可以比弥亏策略有更宽松的特定条件或者能更快的达到预期收益额呢？使用机器学习方法的人工智能策略也许是一个不错的切入点。我们受AlphaGo Zero1启发，通过神经网络和蒙特卡洛树搜索(MCTS)，根据游戏规则，生成能提供最佳回报的竞猜决策。

三. MCTS最优竞猜策略

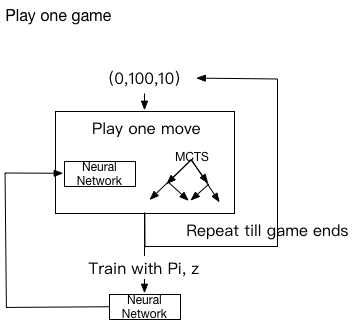
MCTS是人工智能的传统方法，通过搜索游戏状态的可能的不断转变的结果，选择一个最终能赢得游戏或者获取游戏高分的动作。理论上，几乎所有的游戏都可以用MCTS解决。但是由于很多游戏可能的游戏动作太多，状态空间太多，导致搜索树的深度或广度太大，在计算机的计算能力范围内有生之年都不可解决。如围棋就有约1,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000,000（171个零）种游戏状态。要解决这些游戏就要在游戏达到最终状态前，对游戏的某个状态的收益进行估算。不论是AlphaGo2，还是后来更强的AlphaGo Zero都是利用机器学习的神经网络对围棋的游戏状态收益进行预估，结合 MCTS来选择下棋动作的，进行强化学习的。

同样，我们可以借鉴AlphaGo Zero，把机器学习和MCTS结合，仅仅通过规定游戏规则，即规定每个状态下游戏规则允许的动作，让计算机在规则范围内不断地按照神经网络的输出结果进行尝试，根据尝试后的输赢结果对神经网络进行强化学习。在这过程中，计算机将学习到一个能赢得游戏的策略。

三.1. 总体强化训练过程

通过神经网络和MCTS进行强化训练的思路是， 在每个回合根据游戏状态决策一个动作时，通过神经网络输出的动作策略和收益，多次重复地进行模拟玩游戏，再根据多次模拟游戏的结果反馈到MCTS的状态中，决策出最佳动作进行真正的下注行动。在一轮游戏下来，会有一个已执行的实际动作决策和实际的游戏收益，这两个信息又可以用来对神经网络进行训练，并指导下一轮游戏，知道我们认为这个神经网络已经有能力赢得游戏。

下图展示了一轮游戏中神经网络和MCTS训练的大概过程。掷骰子游戏状态只有三个维度：回合数、余额和目标收益。从游戏初始状态（如0，100，10）开始，根据神经网络和MCTS方法的输出结果，决策出一个动作进行操作，游戏状态会转换成下一个状态，记录下此游戏状态下，MCTS的动作策略Pi（即动作的概率分布）。如此不断重复，直到达到规定的游戏结束状态。根据游戏结束状态，计算游戏的实际收益z，如结束时的余额作为收益，把这个收益z作为从初始状态到结束状态所有状态的实际收益。最后，把这些状态各自的动作策略Pi和收益v作为输入，对神经网络进行强化训练。



三.2. 神经网络结构

下面让我们分别介绍上述的神经网络和MCTS的计算过程。首先，神经网络是用于输入掷骰子游戏的状态，生成竞猜的选择和状态预估收益 。考虑到确定游戏本身状态的维度较少，没有必要使用深度神经网络（DNN）或卷积神经网络等较复杂的神经网络，我们使用较简单的Softmax分类器和Sigmoid聚类器来实现。它的结构如下：



它用函数表示为(P,V) = fθ (s)，根据输入的游戏状态s和神经网络参数θ， 输出的是通过Softmax分类器产生的竞猜选择P，它各注种的选择和竞猜数额的组合的概率，和通过Sigmoid聚类器产生的状态预估收益V。它的损失函数为：

.其中z为下文将介绍 的MCTS搜索的实际收益V，π为MCTS对于该游戏状态的实际选择动作的概率，c为常数。

相对围棋，它的状态空间小很多，假设回合数控制在100以内，余额从0%到100%，目标收益在1%到100%，那么状态总数约只有1000000。如输入为（0，100， 1）时，一个输出可能为1%的概率选择猜小，数额为10，0.5%的概率选择猜小，数额为5等等；状态（0，100，1）的预估收益为0.9 。

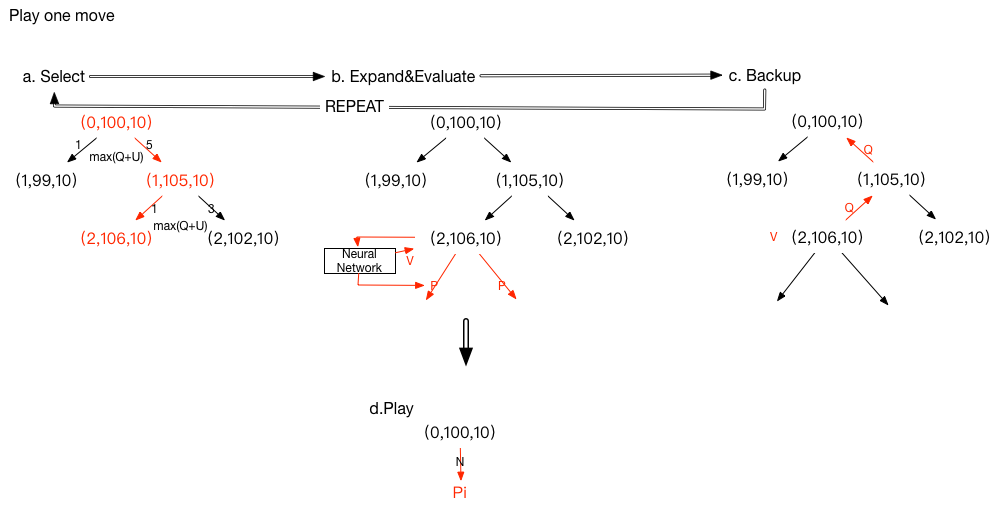
三.3. MCTS状态更新

有了这个神经网络我们就能对MCTS中游戏状态收益进行预估并对游戏动作选择进行指导 。MCTS的树状态可以由四个维度来描述：

{N(s,a), W(s,a), Q(s,a), P(s,a)}.

其中N(s,a)代表游戏状态s执行动作a的访问次数，W(s,a)代表动作收益之和，Q(s,a)代表平均动作收益，P(s,a)代表预定的动作决策概率。

如下图所示，在一次游戏动作决策中，MCTS对于一个游戏状态，它的每一步搜索过程包括如下四个步骤：重复地进行模拟游戏a.选择、b.扩展及预估、c.反推，最后进行d.行动。上述的MCTS树状态{N,W,Q,P}的值将 b.扩展和预估以及c.反推中得到赋值或更新，并指导a.选择和d.行动的执行结果。

a. 选择：MCTS树中，从根状态s0开始，直到叶子节点游戏状态sL的每一步t，分别选择一个动作at，满足, 其中, cpuct为表示影响选择动作时偏向探索的程度的一个常数。如图，MCTS会从根状态(0,100,10)开始，根据a0选择动作，并进入下一个状态，直到状态(2,106,10)这个叶子节点。

b. 扩展及预估：在叶子节点sL，根据神经网络计算(P,V) = fθ (sL)。针对叶子节点sL所有可能的动作a，添加新的边，并对其MCTS树状态进行初始化为N(sL,a)=0, W(sL,a)=0, Q(sL,a)=0, P(sL,a)=Pa。V的值将在下一步反推中用到。

c. 反推：对于选择路径中的每个节点st，对它的树状态进行更新。N(st,a)= N(st,a)+1, W(st,a)= W(st,a)+v, Q(st,a)= W(st,a)/ N(st,a)。

d. 行动：根据树状态，生成根状态下的动作决策，并实际执行动作。，其中称为热度常数。此Pi就是后来用于对神经网络进行强化训练的输入之一。MCTS找到使得Pi(a|s0)最大的a，认为它是最优的动作，并在根状态下执行a。下一次动作决策将从新的根状态s0’开始，MCTS树状态可以复用。

此过程的代码实现可参考：<https://github.com/liangfengsid/dicegame>。

三.4. 回顾

MCTS在游戏过程中，对动作决策起到主要作用。神经网络只是帮助MCTS在此过程中对游戏状态的动作概率和收益进行估算。在这过程中有多个循环。一个是不断重复地玩多轮完整的游戏，然后是每轮游戏需要重复地根据当前游戏状态决策动作进行行动。而每决策回合需要多次重复地进行模拟游戏选择->扩展及预估->反推的过程，此过程需要从树的根节点状态探索到叶子节点状态，在扩展的环节还在根据状态的所有可能动作进行扩展。

我们来看一下整个强化学习过程的计算复杂度。假设需要重复游戏轮数为Kp，每次游戏的最大行动回合为Km，每回合需要模拟游戏选择->扩展及预估->反推最大次数为Ks，模拟的最大深度为Kd，每个状态的最大可能动作数为Ka那么整个强化学习过程的时间复杂度为O(KpKmKsKdKa)，其中神经网需要训练的次数为O(KpKm)。在掷骰子游戏的参考代码实现中，考虑到游戏的实际规则和训练时间等因素，控制Km<=100, Ks=100, Kd<=100, Ka<=200。

四. 实验结果

引用：

1. AlphaGo Zero, Google Deepmind, <https://deepmind.com/blog/alphago-zero-learning-scratch/>, 2017

2. AlphaGo, Google Deepmind, <https://deepmind.com/research/alphago/>, 2016

3. DiceGame, Liang Feng, <https://github.com/liangfengsid/dicegame>, 2018