Rainbow

Abstract：

目前已经有几项针对DQN的改进措施，但不知道这些改进能不能有效地结合起来。我们尝试了其中的六种，并用实验去研究它们结合使用时的效果。实验显示结合使用在Atari2600上效果很好。我们也提供消融实验的结果，这展示了各部分对于总成绩的贡献。

Introduce：

1 首先介绍了DQN，然后介绍几种改进。

2 几种改进

DoubleDQN：通过分离对bootstrap action选择和评估来解决Q-learning的过度估计偏差

（在统计学中，Bootstrap法具体是指用原样本自身的数据抽样得出新的样本及统计量。它是一类非参数Monte Carlo方法,其实质是对观测信息进行再抽样，进而对总体的分布特性进行统计推断。-[机器学习中的bootstrap到底是什么？ - 知乎 (zhihu.com)](https://zhuanlan.zhihu.com/p/261387233)）

Prioritized experience replay：通过replay更常见的transitions来提升数据效率

\*Dueling network architecture：通过分别represent 状态值（state value）和 action advantage来概况across action

Learning from multi-step bootstrap targets:转变bias-variance平衡并使新观察到的reward传播得比之前观察到的state要快

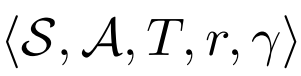
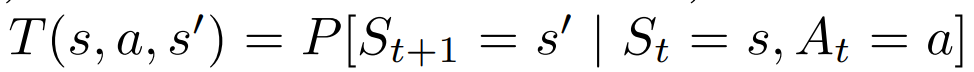
Distributional Q-learning：学习 return（总reward？）的真实分布而不是去估计

Noisy DQN：用一层随机网络去观察，以增加噪声提高模型的稳固性（robust）

下边将会把这些方法一起用

Background：

RL：用agent在环境中进行一系列动作，并使reward最大化。

Agents and environments：每个离散时间，environment给一个状态St，agent给一个动作At，然后env给下一个状态St+1和reward Rt+1和衰减因子y t+1。最终形成MDP，S、A是St，At的集合，

T是（随机）转移函数（从s转移至下一个状态s‘）

r(s,a)是reward函数，最后那个是衰减因子，实验中衰减因子为一常量。

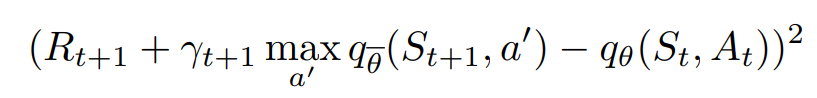
Agent用策略pi来选择动作。



Agent的目标是采用一个策略pi去最大化Gt。

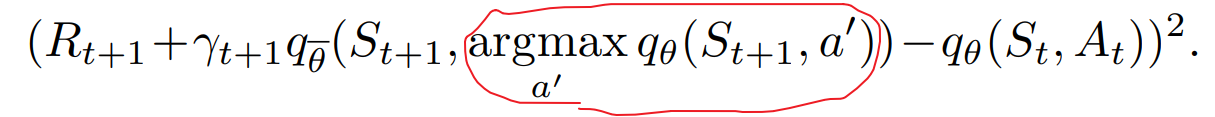
策略pi可以被直接训练，可以应用ε-greedy进行更新。

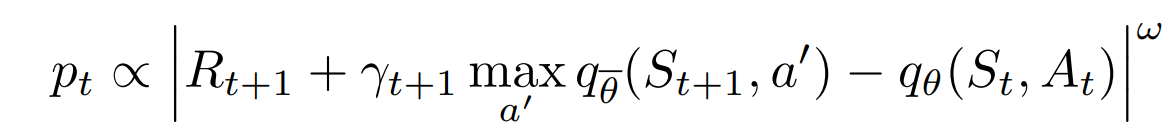
DRL和DQN：在DRL中，我们用Deep nn训练例如pi（s,a）q(s,a)等部分.

在DQN中，使用CNN去评估动作的值，用SGD（RMSprop，SGD的变种）作为优化器。以为Loss函数，其中t为replay memory中任取的值。

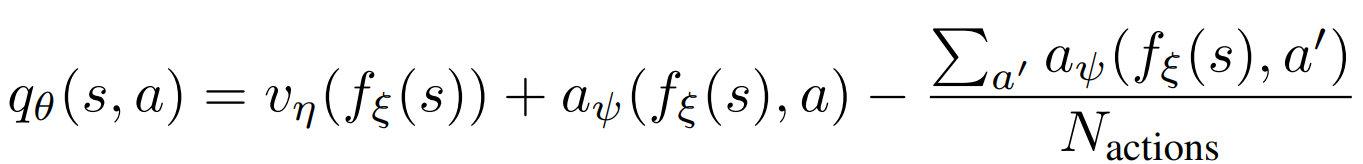
DQN的拓展方法：

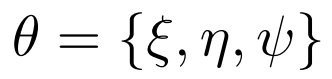
Double Q-learning：改变Loss函数，防止过高的估计

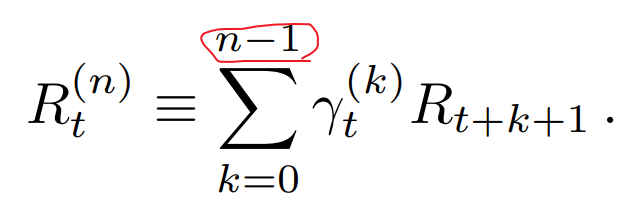


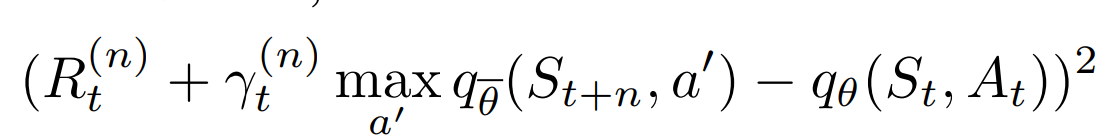
Prioritized replay：使用一个概率pt来选择需要replay的transition，其中pt定义为：

其中w是个超参数。

Dueling networks：value和advantage分别计算，它们共用一个convolution encoder，并用一个特殊的汇集器融和。

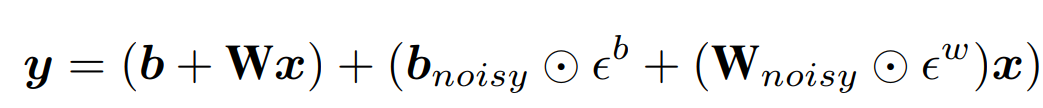
其中三个参数分别代表共用部分的参数、value的参数和advantage的参数。

Multi-step learning：采用一个截短的n步reward而不是原先的∞步。

相应的，loss函数也做了改动

通过调节n的值，可以实现快速训练。

\*Distributional RL：没看懂

Noisy Nets：引进有噪音的线性层来替代普通的线性层。其中两个ε是随机变量，代表 element-wise product（类似于点乘）。

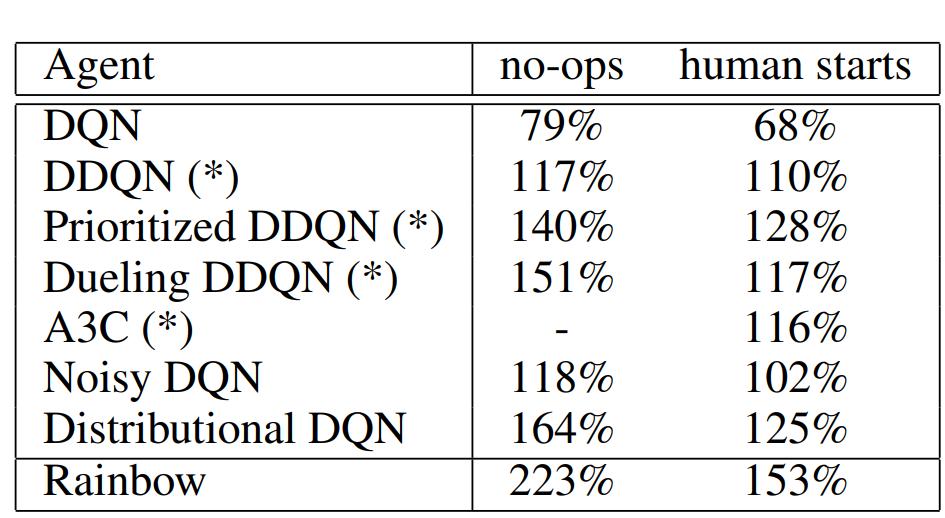
The Integrated Agent：

融合以上方法的agent，被命名为rainbow。

Experimental Methods：

使用以前文章中的环境和评估方法，将agent训练1M步然后暂停学习，评估500k frames。每个episode被截断为108k frames（或者30分钟的仿真游戏）。Agent的得分被归一化。

作者还做了消融实验，结果见下



Discussion：

结果证明，融合亿堆方法的rainbow很成功，并且也得到了每项改进的效果，但还有很多改进DQN的措施没有被融进rainbow，然后作者介绍了其他措施。