1 近似分布学习：算法与数值实验

1.1 近似分布学习

1.2 算法实现细节

本次作业中，我们实现了经验重放的近似分布学习 Double DQN 算法，在 OpenAI gym 环境下对 Atari 游戏 Alien 进行学习，并与 OpenAI Baselines:DQN 算法在相同机器上进行了效果比较。

我们实现的算法有如下值得提及的细节：

- 智能体在其生命周期内，分为三个行为阶段：观察阶段、探索阶段和学习阶段 [1]。

  1. 智能体的前 N_o 个时间步为观察阶段。在这一阶段，智能体完全随机选择每一步的行动 a_t，并将观测到的转换组 $(\phi_t, a_t, r_t, \phi_{t+1})$ 存入重放缓存中。此时的智能体不进行网络的训练；
  2. 智能体在观察阶段后的 N_e 个时间步内为探索阶段。在探索阶段，智能体仍然完全随机选择每一步的行动，并存储相应的转换组作为经验。但此时的智能体开始初步学习，训练网络；
  3. 智能体在探索阶段后进入训练阶段。此时的智能体按照 ϵ-贪心方法选择动作 a_t，仍然存储经验并训练网络。值得注意的是，在前两个阶段，随机选择行动 a_t 相当于参数 $\epsilon = 1$ 的 ϵ-贪心方法。在训练阶段，我们并不给定一个固定的 ϵ 值，而是以

  $$\epsilon \leftarrow \epsilon_0 - \min\left(1, \frac{t - (N_\mathrm{o} + N_\mathrm{e})}{f_\mathrm{e} M}\right) \cdot (\epsilon_0 - \epsilon_\mathrm{min})$$

  来确定 ϵ。其中 $\epsilon_0 = 1$，ϵ_min 为 ϵ 最小值，N_o 和 N_e 分别是观察和探索步数，M 为最大行动步数，f_e 为衰减系数。可见，ϵ 以线性方式由 1 衰减至最小值 ϵ_min，而后保持不变。

- 值分布的支集 $\{z_i = V_\mathrm{min} + i\Delta z : 0 \le i < N_\mathrm{atom}\}$ 上的概率 $\{p_i(x, a)\}$ 由神经网络参数化，具体的网络结构如下：

  1. 输入层：从环境 gym 中得到的游戏画面数据，图片像素的行数、列数和信道数随游戏而改变，样本数为 batch size 和记忆大小两者的较小值；
  2. 第一隐藏层：2 维卷积层，filters=32，kernel_size=(8,8)，strides=(4,4)，activation='relu'；
  3. 第二隐藏层：2 维卷积层，filters=64，kernel_size=(4,4)，strides=(2,2)，activation='relu'；
  4. 第三隐藏层：2 维卷积层，filters=64，kernel_size=(3,3)，strides=(1,1)，activation='relu'；
  5. 第四隐藏层：Flatten 层；
  6. 第五隐藏层：全连接层，units=256；

[1] 该想法参考了 https://github.com/flyyufelix/C51-DDQN-Keras

7. 输出层：N_a 个共享隐藏层的全连接层，每个全连接层的神经元数目 units=N_{atom}，激活函数 actication='softmax'。其中 N_a 为从 gym 环境得到的当前游戏中全部可能行动的数目。

   网络损失函数的形式为 categorical_crossentropy。

- 参数设置：

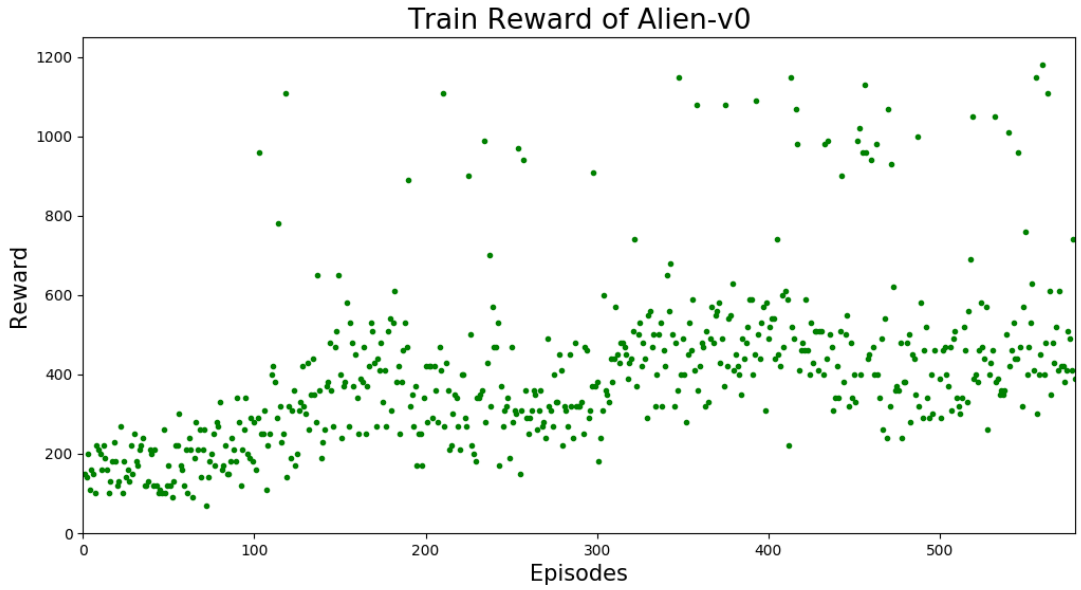
V_{min}	V_{max}	N_{atom}	N_{o}	N_{e}	M	f_{e}	ϵ_0	ϵ_{min}	γ
0	1000	51	10000	40000	500000	0.2	1	0.01	0.99

1.3 数值实验

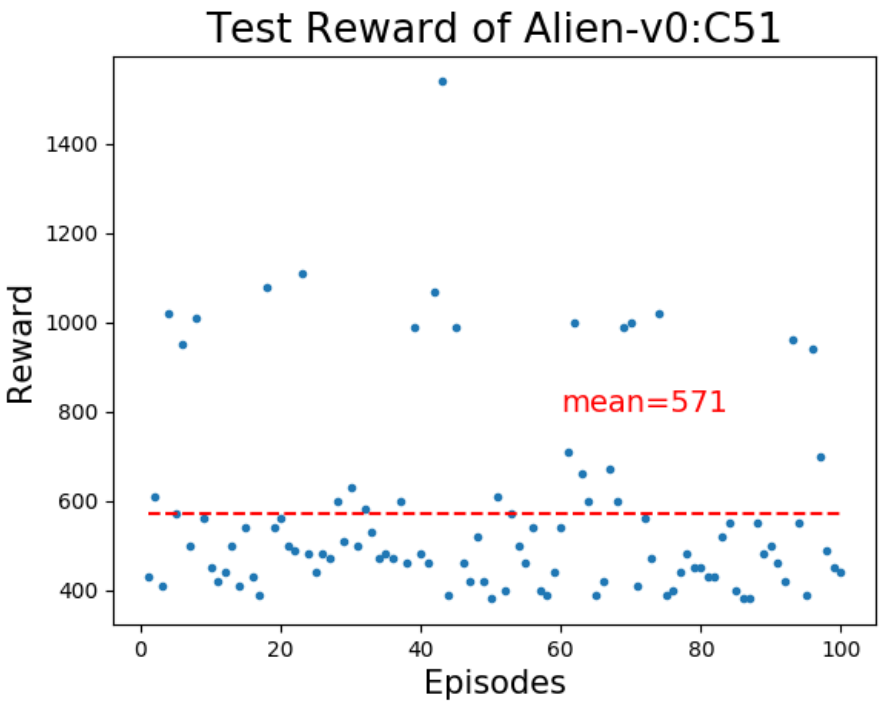
我们用来学习的硬件信息为：

- Intel(R) Core(TM) i7-4790 CPU@3.60GHz, 1 物理处理器,4 核心,8 线程；
- RAM:16GB
- 无独立显卡

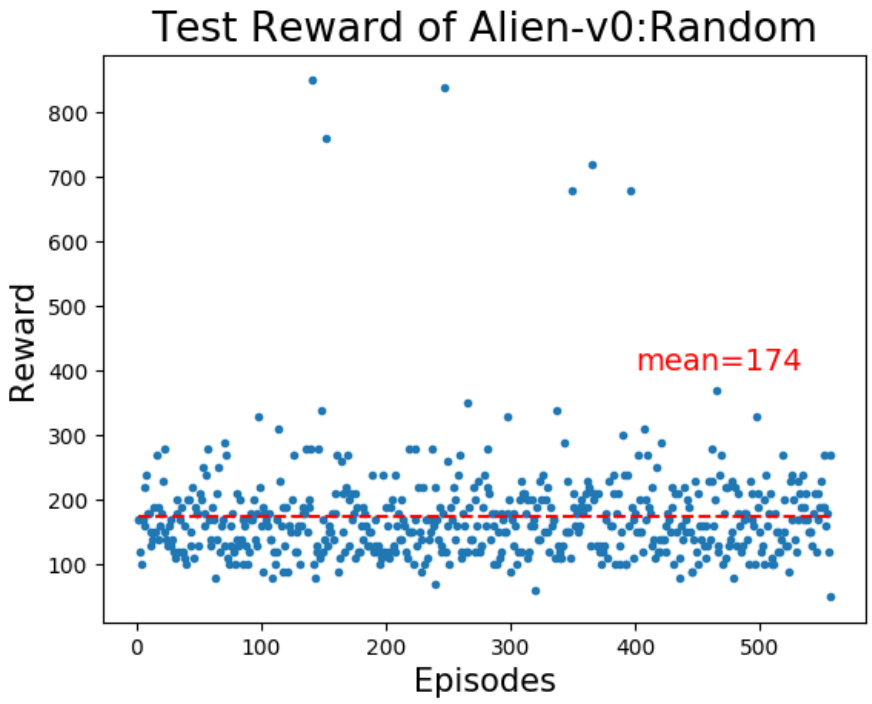
经过 578 个 episodes 的学习，近似分布学习方法的训练得分如下：



智能体在 100 次测试中得分表现为：



与随机行动时的得分相比，可见智能体的表现有着显著的提高：



从测试表现中可以看出，训练后的智能体得分分别集中在 500 分附近和 1000 分附近。通过

观察游戏画面，我们发现，两个得分范围的差别主要在于游戏主角是否杀死过敌人。如果有更好的硬件设备进行更长时间的训练，我们有信心将测试平均分提升到 1000 分左右。