StrongBird：基于 DQN 的 Flappy bird AI

蔡引江

2019 年 4 月 17 日

摘要

Deep Q-learning network 将 Q-learning 算法和神经网络结合并应用于玩 Atari 游戏上，最终模型可以达到人类水平。本文将在应用 DQN 于 Flappy bird 游戏的过程中逐步讲解模型各部分的功能及背后的原理。即便对只了解机器学习一些基本概念（如监督学习等）的读者也可以理解整篇文章，并可以以此为基础去阅读原文献。

1 监督学习

我们不妨先将文献中的模型设为硬性条件：只给定游戏帧和游戏得分，怎样训练出一个 AI 学会玩这个游戏？

自然我们会想到一个十分通用的方案，使用监督学习！如果能训练出一个神经网络，它的输入是游戏帧，输出是当前应该采取的行动，那么问题就解决了。再改进以下方案，输入还是游戏帧，输出为采取各个行动的能获得的长期累积奖励，只要选择奖励最大的行动即可。

我们先将某时刻的游戏帧定义为状态 s_i，采取的行动定义为 a_i，得到的奖励为 r_i。我们可以将游戏过程抽象为

$$((s_1, a_1, r_1) \to (s_2, a_2, r_2) \to \cdots \to (s_t, a_t, r_t)) \quad (1)$$

对时刻 i，神经网络选择会获得最大长期累积奖励的行动 a_i。

$$a_i = \max_a \ R(s_i, a; \theta) \quad (2)$$

$$R(s_i, a) = r_{i+1} + r_{i+2} + \cdots + r_t \quad (3)$$

最大长期累积奖励是一种预估的奖励，神经网络认为自己采取行动 a_i 后依据自己的策略能够得到的所有预期奖励和。

现在核心问题已变成监督学习中的回归问题了，只要能以某种方法找到**最大长期累积奖励**的真实值，我们就可以对神经网络使用**均方误差**损失函数进行回归的训练了。

2 Q-learning

那么，最大长期累积奖励的真实值怎么得到呢？我们依据 Q-learning 算法和游戏得分，通过不断迭代其实就可以获得一个比较不错的最大长期奖励的**近似值**。公式 4体现了 Q-learning 的核心思想

$$Q^*(s, a) = \mathbb{E}_{s'}\left[r + \gamma \max_{a'} Q(s', a') | s, a\right] \quad (4)$$

在当前状态 s，采取行动 a 得到的最大预期奖励和应该包括：环境对行动 a 反馈的奖励 r、折扣因子影响下的新状态 s' 能预期得到的最大奖励和。添加折扣因子的原因是预期奖励不如已获得的奖励实在，需要打个折扣。

依据 Q-learning 我们给出近似真实的最大长期累积奖励：

$$R^* = \begin{cases} r & \text{，游戏在 s 处结束} \\ r + \gamma \max_{a'} Q(s', a') & \text{，否则} \end{cases} \quad (5)$$

使用均方误差进行训练：

$$loss = (R - R^*)^2 \quad (6)$$

我们给出一个例子来展示 Q-learning 是如何通过迭代起作用的。假设有游戏序列如下：

$$(s_0, a_0, 0) \to (s_1, a_1, 0) \to (s_2, a_2, -1)$$

游戏在 s_2 处结束，获得奖励-1 以惩罚所导致结束的行动 a_2。在以 $(s_2, a_2, -1)$ 为样本进行训练后，神经网络会将以 s_2 为输入时的输出 $R[\ a_2\ ]$ 逼近向-1，表明当前不应采取该行动。对于非终点游戏帧，如 s_1，依据 Q-learning 公式 4得到 $R^* = \gamma \max_{a'} Q(s_2, a')$。如果 a_2 仍为最大预期奖励和的话，则预期奖励和为 $\gamma * -1$。如果 a_2 不是最大预期奖励和的行动，也表明终点的-1 影响已经回馈到 s_1 状态下了（不将其考虑进最大预期奖励和）。总之，根据迭代更新，只要我们充分与环境交互并学习，最终预期奖励和能够近似代表真实环境反馈的奖励。

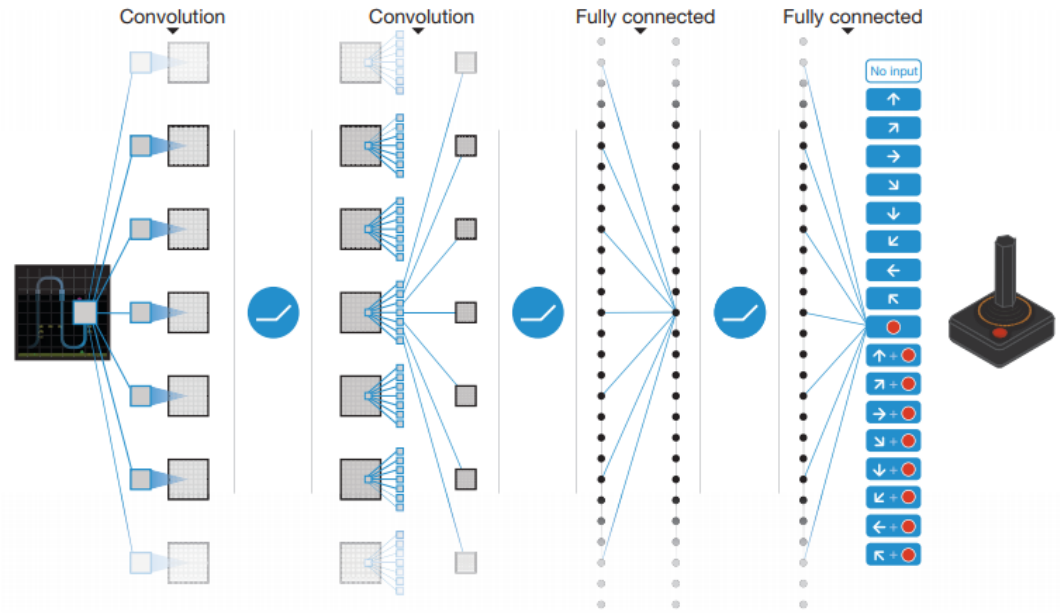

图 1: 神经网络结构

3 训练细节

3.1 target 网络

上述 Q-learning 公式 4在实际应用中会有一个问题，我们要训练的是神经网络，但产生标签（近似真实的预期奖励和）的也是神经网络。自己训练自己，就好像自己打比赛同时自己做裁判，这会导致训练过程的不稳定。文献中给出的方法是保留一个神经网络的副本—target 网络。目标神经网络的参数是不断学习更新的，而 target 网络的参数是“过时”的目标神经网络的参数，每过一段时间会把目标神经网络的参数更新到 target 网络。这时标签产生工作就从目标神经网络中剥离开了。

$$R^* = r + \gamma \max_{a'} Q(s', a'; \hat{\theta})$$

$$R = Q(s, a; \theta)$$

3.2 记忆存储

按上述介绍的模型，神经网络的学习方式是：交互取得样本、训练、再交互，不断重复。这样会带来一个问题就是样本的利用率很低，一个样本仅学习一次就被舍弃，且一次仅学习一个样本，收敛速度也很慢。文献中提出了“记忆存储”的方法，将交互得到的样本存储起来，训练时从“记忆存储”中随机选取一批样本。同时，随着模型与环境的不断交互，“记忆存储”也在不断地更新。“记忆存储”中的单个样本如下：

$$(s_t, a_t, r_t, s_{t+1})$$

Algorithm 1: deep Q-learning with experience replay.
Initialize replay memory D to capacity N
Initialize action-value function Q with random weights θ
Initialize target action-value function $\hat{Q}$ with weights $\theta^- = \theta$
For episode = 1, M **do**
  Initialize sequence $s_1 = \{x_1\}$ and preprocessed sequence $\phi_1 = \phi(s_1)$
  For t = 1,T **do**
    With probability ε select a random action a_t
    otherwise select $a_t = \text{argmax}_a Q(\phi(s_t), a; \theta)$
    Execute action a_t in emulator and observe reward r_t and image x_{t+1}
    Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    Set $y_j = \begin{cases} r_j & \text{if episode terminates at step j+1} \\ r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-) & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $\left(y_j - Q(\phi_j, a_j; \theta)\right)^2$ with respect to the network parameters θ
    Every C steps reset $\hat{Q} = Q$
  End For
End For

图 2: 模型训练算法

3.3 ϵ 贪婪

强化学习中的一个重要概念是**探索与利用**。探索是为了找到更好的策略，利用是为了模型能够更好地探索。在训练初期，模型完完全全的新手，只能通过随机采取行动来摸索规则。随着训练的进行，模型逐渐学到游戏的规则，并开始放弃随机行动而基于自身判断选择最优行动。就 Flappy bird 而言，选择最优行动有助于 AI 进入游戏的后面部分，使 AI 有机会完整地学习游戏。

具体地，我们设置参数 ϵ 表示采取随机行动的概率。训练初期 ϵ 置为 1.0，训练过程中逐渐下降到 0.1 并保持。

超参数名	超参数值
minibatch size	32
replay memory size	50000
agent history length	4
target network update frequency	5000
γ	0.99
learning rate	1e-6
optimizer	Adamoptimizer
initial ϵ	1
final ϵ	0.1

表 1: 超参数列表及取值

3.4 奖励设置

模型中非常重要的一个部分就是奖励的设置，环境奖励的设置好坏对模型的最终效果有非常大的影响。我们依据文献中的方法对各种状态设置奖励：采取行动并得分 $r = 1$、采取行动并游戏结束 $r = -1$、其他 $r = 0$。

3.5 输入样本

在大多数游戏包括 Flappy bird 中，仅靠单游戏帧能获得的信息很少。为了能捕获更多动态信息，我们将样本设置为连续的 k 帧游戏画面，模型中 $k = 4$。举例说，通过多帧，神经网络可以学习到样本中小鸟的运动方向和趋势等单帧没有的信息。

参考文献

[1] Mnih, V. et al. Human-level control through deep reinforcement learning.Nature 518, 529–533 (2015).