Computer Science and Application 计算机科学与应用, 2021, 11(7), 1994-2007
Published Online July 2021 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2021.117204

Hans汉斯

Q-Learning算法的改进和实现

古　彭*，李怀诚*，杨诗妍，杨　威#，刘嘉帆

西藏大学，西藏　拉萨
Email: #1296907025@qq.com

收稿日期：2021年6月23日；录用日期：2021年7月21日；发布日期：2021年7月28日

摘　要

机器学习领域开始越来越受人们关注并且也是人工智能最新的探寻方向。最近几年强化学习的研究增长部分原因是在玩一些电子游戏中可以达到人类所达不到的高水平。使用基于策略的强化学习算法可以更好地适应游戏环境，探索出一种相对稳定的路径，达到全局最优的目标。本文研究的是基于强化学习Q-learning算法的Play Flappy Bird游戏。首先研究了强化学习的理论知识，对马尔可夫决策、动态规划、值函数近似、时间差分等相关理论进行了深入研究。重点研究了建立Flappy Bird游戏中的状态、行为、奖励数学模型，为了得到最优策略，对每一个状态下的目标是使总奖励最大化。在此基础上，本文将对深度卷积神经网络模型展开训练，从而可以识别游戏状态中的图像，并对其进行分类。系统仿真成功地运用深度Q-learning模型实现Flappy Bird的自我学习，探索概率ε在550,000更新中从0.6线性下降到0，学习率一开始非常陡峭，但随后达到稳定，在比较短的时间内实现收敛效果，训练误差较低。智能体训练达到理想效果，均值得分为86分，最高得分为335分，已经超过普通人类玩家，取得了良好的成绩。

关键词

强化学习，Play Flappy Bird游戏，Q-Learning算法，深度卷积神经网络

Improvement and Implementation of Q-Learning Algorithm

Peng Gu*, Huaicheng Li*, Shiyan Yang, Wei Yang#, Jiafan Liu

Tibet University, Lhasa Tibet
Email: #1296907025@qq.com

Received: Jun. 23rd, 2021; accepted: Jul. 21st, 2021; published: Jul. 28th, 2021

*共同第一作者。
#通讯作者。

文章引用: 古彭, 李怀诚, 杨诗妍, 杨威, 刘嘉帆. Q-Learning 算法的改进和实现[J]. 计算机科学与应用, 2021, 11(7): 1994-2007. DOI: 10.12677/csa.2021.117204

Abstract

The field of machine learning has begun to attract more and more attention and is also the latest direction of artificial intelligence. Part of the reason for the growth of research in reinforcement learning in recent years is that playing some video games can reach high levels that humans cannot reach. Using strategy-based reinforcement learning algorithms can better adapt to the game environment, explore a relatively stable path, and achieve the goal of global optimization. This article studies the Play Flappy Bird game based on the Q-learning algorithm of reinforcement learning. First, the theoretical knowledge of reinforcement learning is studied, and related theories such as Markov decision-making, dynamic programming, value function approximation, time difference and other related theories are deeply studied. The focus is on the establishment of mathematical models of states, behaviors, and rewards in Flappy Bird games. In order to obtain the optimal strategy, the goal for each state is to maximize the total reward. On this basis, this article will train the deep convolutional neural network model so that images in the game state can be identified and classified. The system simulation successfully implements the self-learning of Flappy Bird using the deep Q-learning model. The exploration probability ε decreases linearly from 0.6 to 0 in the 550,000 updates. The learning rate is very steep at the beginning, but then it reaches a stable level, the convergence effect is achieved in a relatively short time, and the training error is low. The intelligent body training achieves the ideal effect. The average score is 86 points, and the highest score is 335 points, which has surpassed ordinary human players and achieved good results.

Keywords

Reinforcement Learning, Play Flappy Bird Game, Q-Learning Algorithm, Deep Convolutional Neural Network

1. 强化学习与 Q-Learning 算法

1.1. 强化学习

1.1.1. 基本原理与模型

强化学习的概念是通过优化了智能体与周围环境的交互和进行试错，使整个强化学习活动进行过程中的学生根据累计成绩得到奖励最大的一种进行强化学习的方式。实际上强化学习的基本思想其实就是通过与周围环境智能体进行交互，利用智能体对环境的反馈信息来自动实现对决策的优化。强化学习的框架和模型的结构如图 1 所示，智能体在每个相应的时间点执行动作 t 从相应的环境中选择并获得当前的状态，然后从动作的收集 A 中进行选择并继续执行一个相应的动作 a，就会直接得到环境给予智能体的一个奖励 r，而且在执行动作 a 后将导致状态转移到环境上。强化学习的原理是：如果智能体的某些动作产生了正的奖励，则智能体以后执行这些动作的概率就会大大增加，否则，智能体在学习过程中执行这些动作的概率就会大大减弱。

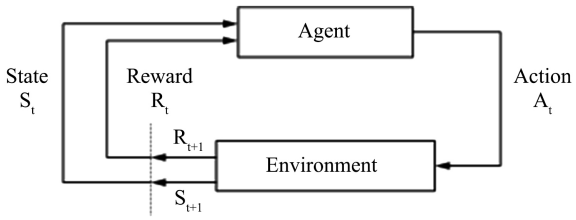


Figure 1. Reinforcement learning structure
图 1. 强化学习结构

1.1.2. 马尔可夫决策过程(Markov Decision Process, MDP)

马尔可夫决策的过程是序贯过程式决策(sequential decision)的一个数学过程和模型，用于在系统状态具有马尔可夫性质的环境中模拟智能体可实现的随机性策略与回报。

马尔可夫决策基于一组交互对象，即智能体和环境进行构建，所具有的要素包括状态、动作、策略和奖励。在马尔可夫决策的模拟中，智能体会感知系统状态，同时还需要以相应的策略作为基础，从而采取相应的实施动作，通过对起环境状态进行改变，最终获得一定的奖励，奖励随时间的积累被称为回报。MDP 的理论基础是马尔可夫链，马尔可夫性质为：

$$p\left(s_{i+1}|s_i,a_i,\cdots,s_0,a_0\right)=p\left(s_{i+1}|s_i,a_i\right) \tag{1}$$

其前一时刻的动态关系到当前时刻状态，但是和其他时刻的状态相比，具有一定的独立性。对于等式右侧而言，其对应的条件概率即为转移概率。而对于全部的马尔可夫模型而言，其都具备马尔可夫性质。但是相比而言，在 MDP 转移概率中，加入了一些智能体动作。因此，当进行强化学习的时候，可采用 MDP 的马尔科夫性质。同时，在进行强化学习的时候，其根本上需要和历史信息相符合，其关乎到整体的状态、奖励以及动作等。在此基础上，类比马尔可夫链中的样本轨道(sample path)，可定义 MDP 的轨迹(trajectory)，其公式为：

$$A_\tau=\left\{s_0,a_0,s_1,a_1,r_1\cdots,s_{\tau-1},a_{\tau-1},r_{\tau-1},s_\tau,r_\tau\right\} \tag{2}$$

环境由初始状态 s_0 按给定策略 $\pi\left(a|s\right)$ 演进至当前状态 s_t 的所有动作、状态和奖励的集合。由于 MDP (Main Display Panel)的策略和状态转移具有随机性，因此其模拟得到的轨迹是随机的，且该轨迹出现的概率有如下表示：

$$p\left(A_\tau\right)=p\left(s_0\right)\prod_{i=0}^{\tau-1}p\left(a_i|s_i\right)p\left(s_{i+1}|s_i,a_i\right) \tag{3}$$

一般地，MDP 中两个状态间的轨迹可以有多条，此时由查普曼－科莫高洛夫方程等式可知，两个状态间的 n 步转移概率是所有轨迹出现概率之和。

1.2. Q-Learning 算法

Q-learning 算法是强化学习的一种模式，它不要求智能体有环境相关的先验知识，而是基于策略学习，在这种策略学习中，智能体不需要被告知采取什么样的行为，取而代之的是，智能体通过从实验中获得最大奖励来寻找需要的动作。

1) Q-learning 算法决策是通过行为准则学习外部事物，智能体处于状态 $s1$，有两个行为 $a1$、$a2$，以智能体经验作为基础，可以得知当处于 $s1$ 状态时，此时 $a2$ 的行为潜在奖励高于 $a1$。若 $Q\left(s1,a1\right)$ 小于 $Q\left(s1,a2\right)$，则智能体选择 $a2$ 行为为下一个行为。当智能体状态更新成 $s2$ 时，重复上面的过程，比较 Q 值的大小。

2) Q-learning 算法的更新要根据 Q 表的估计，因为在 $s1$ 中，$a2$ 的值比较大，通过之前的决策方法，

在 $s1$ 采取了 $a2$，并到达 $s2$，这时我们开始更新用于决策的 Q 表，接着并没有在实际中采取任何行为，而是智能体再想象在 $s2$ 上采取了每种行为，分别看看两种行为哪一个的 Q 值大，公式如下：

$$Q(s1,a2)\text{现实：}R+\gamma^{*}\max Q(s2)$$

$$Q(s1,a2)\text{估计：}Q(s1,a2)$$

若 $Q(s1,a2)$ 高于 $Q(s2,a1)$，此时选择大的 $Q(s2,a2)$，然后和未来奖励 Υ 相乘，最终可以得到到达 $s2$ 时，其获得的即时奖励 r。该数值能够充分体现出 $Q(s1,a2)$ 的值。在得到了估计值和现实以后，可以对 $Q(s1,a2)$ 进行更新。同时还需要利用现实与估计之间的差距，乘以学习效率 α，可以对 $Q(s1,a2)$ 进行更新，从得到新值。同时，在这个过程中，将利用 $\max Q(s2)$ 对 s2 状态进行估算，更新了其行为决策以后，就需要重新计算。整体公式为：

$$Q(s,a)\leftarrow Q(s,a)+\alpha\left[r+\gamma\max_{a'}Q(s',a')-Q(s,a)\right]$$

其中 Q-learning 中的 Υ 是对未来奖励的衰减值，越远的奖励，衰减越多。Υ 在 0 到 1 之间，Υ 越大，说明模型对未来的考虑越多，越有对未来预测的作用，如下所示：

$$Q(s1)=r2+\gamma Q(s2)=r2+\gamma\left[r3+\gamma Q(s3)\right]=r2+\gamma\left[r3+\gamma\left[r4+\gamma Q(s4)\right]\right]=\ldots$$

$$Q(s1)=r2+\gamma r3+\gamma^{2}r4+\gamma^{3}r5+\gamma^{4}r6+\cdots$$

2. 神经网络与深度 Q-learning 算法

2.1. 神经网络

神经网络是根据生物学的基本原理搭建的，其会对人类大脑进行模仿，从而对复杂的事件进行处理的数学运算模型。在该模型中，其包括了多个神经节点，彼此之间相互连接。对于神经元而言，其为特定输出函数，也被称为是激活函数。不同节点间的连接意味着其不同的权重，神经网络就是如此通过一个个神经元相互传递信息最终达到模仿人类大脑记忆功能。神经网络的模型基础为神经元的数学模型。其中，在神经网络中，比较典型的模型包括 Hopfield 网络、BP (back propagation)网络。根据网络层次的不同，可以对其进行分类，共有单层感知器和多层感知器两类。其中，最简单的神经网络模型为单层感知器[1]，如下图 2 左图所示，其组成包括输入层和输出层，二者直接相连。对于多层感知器而言，其中在输出层和输入层中还加入了很多隐含层。其中，对于单隐层的神经网络而言，其本质上为多层感知器。对于整个网络而言，其组成中包含很多个神经元，每个神经元都是感知器，其中下图 2 右图表示的为盛京网络模型结构，其组成包括输出层、隐含层以及输入层。同时，输入单元、隐含单元以及输出单元各有 3 个、3 个、2 个。偏置节点将使用“+1”进行表示，也就是截距。

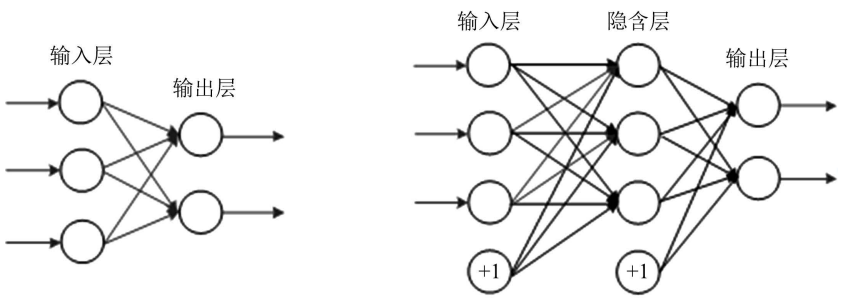


Figure 2. (a) Single-layer perceptron; (b) Multi-layer perceptron
图 2. (a) 单层感知器；(b) 多层感知器

2.1.1. 激活函数

激活函数可以在一定程度上展现其神经网络性能。因此，激活函数会使神经网络在收敛方面表现的不同，具体不同的问题需要采取不同的激活函数。主要采取两种激活函数。(选用 relu 激活函数，原因在于 relu 激活函数在梯度下降时可以在一定情况下抑制梯度消失问题)

1) 单极 S 型(Sigmoid)函数，表达式：

$$f(x)=\text{sigmoid}(x)=\frac{1}{1+e^{-x}}$$

对于 Sigmoid 函数而言，其使用的阈值函数具有单调性、连续性。同时，在神经网络中，还存在的一种激活函数。从图 3 可以看出 Sigmoid 函数的定义域是整个实数，值域介于 0 到 1 之间，因为在此之间的输出正好符合输入量可能发生的概率，适用于拟合。最重要在于它的导函数是由因变量组成，在后续的函数求导中起到了关键性作用。导函数的表达式为：

$$f'(x)=f(x)\cdot(1-f(x))$$

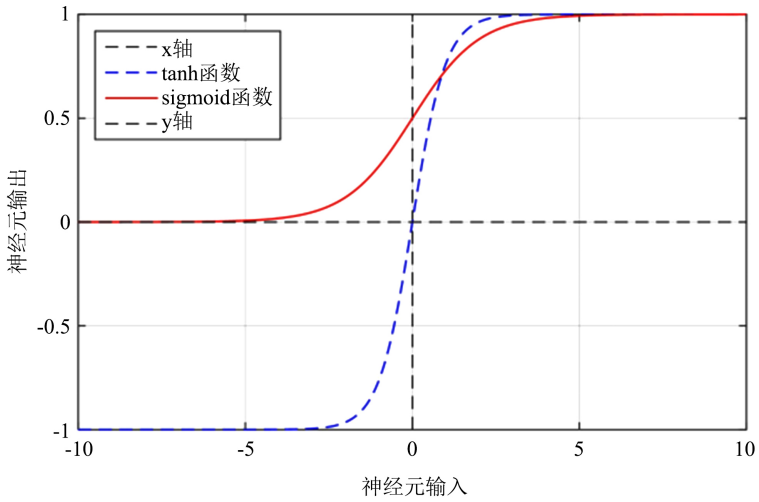


Figure 3. Sigmoid function representation
图 3. Sigmoid 函数表示

2) 双极性 S 型函数(Tanh)，如图 3 所示，其表达式为：

$$f(x)=\tanh(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$$

双极性 S 型函数类似于 sigmoid 函数曲线。S 型函数的值域是[−1, 1]，这可以刻画出一些需要反向含义的神经网络，定义域是整体实数，在一定程度上反应了输入对输出特定的需求。

3) 修正线性函数[2] (Rectified linear unit, ReLU)，如图 4 所示，其表达式为：

$$\text{ReLU}(x)=\begin{cases}x & \text{if } x>0\\ 0 & \text{if } x\le 0\end{cases}$$

ReLU 函数相比较 Sigmoid 函数与 Tanh 函数的优点在于克服了梯度下降问题并且加快了伸进网络的训练速度，但是面对输入为负数的情况下，ReLU 函数会失去训练作用。ReLU 函数输出只会是零或正数，不会出现以零为中心的函数。

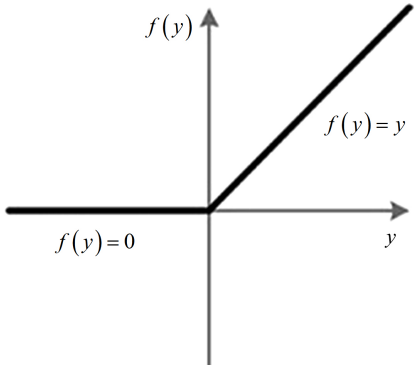


Figure 4. Relu function representation
图 4. Relu 函数表示

2.1.2. 卷积神经网络

卷积神经网络是一种对于二维图像处理的多层感知器，其采用的生物神经网络类似于共享网络结构，能够降低其复杂性，并可以将其连接权值数量降低，能够降低训练的时间。特别是多维图像是网络输入的对象，此时无需提取特征，并重构数据。其中，卷积神经网络属于多层感知器，可以识别二维形状，在对神经网络进行训练的时候，将使用梯度下降法以及 BP 算法。尤其是在网络结构上，其还加入了下采样、权值共享以及局部感受等[3]，在处理缩放和平移的时候，存在高度不变性，性能稳定。

2.1.3. 卷积运算

从本质上看，卷积属于数学运算，其主要是指在设定的范围内，两个变量先相乘然后进行求和。其运算和傅里叶变换之间具有十分精密的关系。利用该特性，可以对傅里叶分析进行简化。

在处理图像的时候，最有效的方式就是卷积。因此在确定输入图像的时候，其每个像素都是相应的像素加权平均值。而卷积核的函数定义就是权值。其中，卷积核是指普通神经网络权值。但是在处理图像卷积、图像信号时，二者存在着一定的差异。对于卷积神经网络而言，对大样本特征进行学习，从而得到小样本。例如，如果小样本大小是 10 × 10，则神经元得到的特征是(100 − 10 + 1) × (100 − 10 + 1)，而其对应的卷积特征有 100 × 89 × 89 个。

2.1.4. 局部感受野和权值共享

从本质上看，局部感受就是隐含神经元的局部连接，其可以对人的眼睛进行模仿。其中，当眼球保持不动的时候，其如果在观看东西，也无法将全目光集中自爱全部物品上，只能关注一个局部。在这种情况下，神经元并不会感知到全部的原始图像信息，出现这种情况下的主要原因是在网络结构中，其设计了 pooling 层、卷积层，二者之间为局部连接关系。对于神经元而言，其能够充分感受到野的值，此时会提高其原始图像的范围，这也表明其语义层的特征值更高。反之，则说明其更加重视细节和局部。通过对野的值进行感受，从而确定其所处的抽象层次。而在 Conv1 中，对于其每个单元而言，其原始图像大小为 3 × 3。而对于 Conv2 而言，其组成成分为 2 × 2 的 Conv1。通过这种方式，可以得到原始图像。其中，对于 Conv1 和 Conv2 的感受野分别是 3、5。对于输入图像而言，其感受野定义是 1，具体如图 5。

从本质上看，权值共享属于节省训练连接权值。其将对感受野的权值进行共享，其类似于神经系统。尤其是在功能和结构等方面，具有一定的可替代性。而在卷积神经网络中，其各个神经元的权值都一样。

在多层感知器中，对于隐含层神经元而言，在其图像中，像素之间为全连接的方式。尤其是对于卷积神经网络中，局部范围中像素点进行连接，通过该方式能够降低训练权值参数，并且优化训练效果，提升速率，从整体上看，提高系统运行的效率。

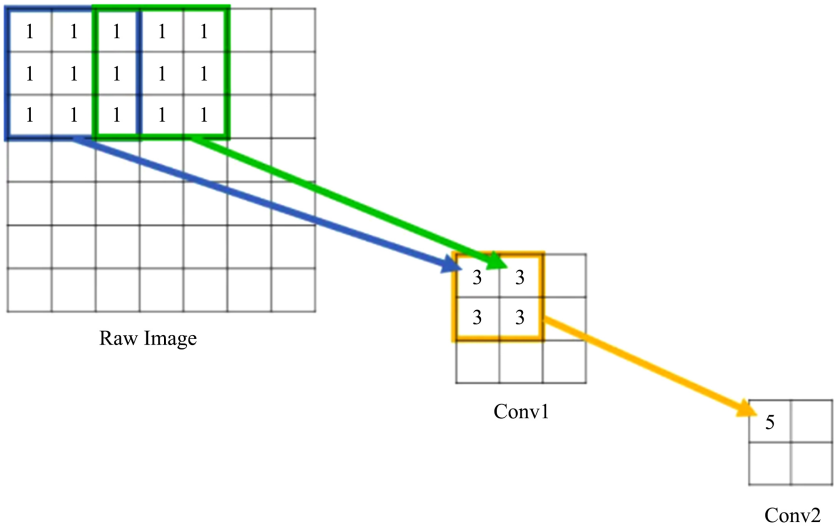


Figure 5. Shows the convolution process
图 5. 为卷积过程

2.1.5. 卷积神经网络的结构

在卷积神经网络中，其为多层感知器，具有一定的特殊性，可以对二维图形进行识别。在该网络结构中，其每层的组成都有多个二维平面，其中又有大量的神经图。具体结构如图 6。

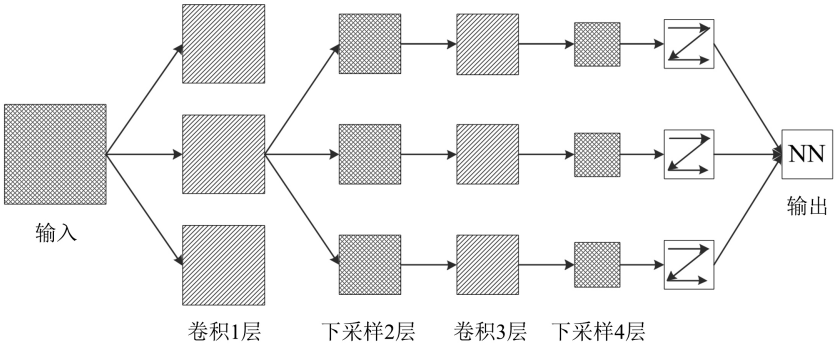


Figure 6. General structure of convolutional neural network
图 6. 卷积神经网络的一般结构

对于一般的卷积神经网络而言，其含有的隐含层共有两种，分别是下采样层以及卷积层。其中，卷积层数据获得的方式是卷积运算，而对于下采样层而言，其会利用均匀卷积核的方式，对局部数据进行平均，然后删除重叠的部分展开二次提取。对于输入层而言，当图像输入以后，利用卷积运算的方式，可以得到三个特征映射图。然后对其展开分组和求和，并加入权值和偏置。最后，可以利用 Sigmoid 函数最终得到采样 2 层对应的映射图。以此进行操作，可以分别得到卷积 2 层以及下采样 4 层对应的特征映射图。之间，将这些图像进行光栅化处理，最后输出相应的值。

同时，在同一个映射面中，其可以将神经元的权值进行共享，同时在网络环境下，其自由参数也出现了大幅度的降低，此时网络复杂度也随之降低。对于卷积神经网络而言，其会以下采样层位基础，卷积层会进行局部平均，然后开展二次提取，但是该方式比较特殊，利用该特点，网络不仅可以识别样本，同时还能够容忍畸变。

2.1.6. 预处理

在 Flappy Bird 游戏中，其最终出现的图片像素为 284 × 512。同时，为了降低存储量，将对图像进行压缩，将其缩小到 84 × 84。在这种情况下，图像色阶设定为 0~255。为了提升精度，将使用纯黑色背景替代原有的图片背景，如图 7。

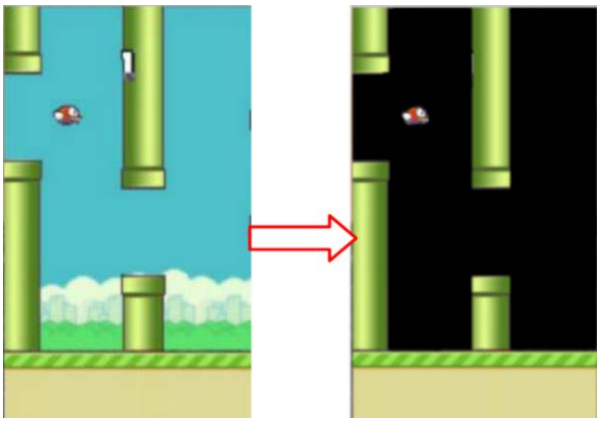


Figure 7. Remove the game background
图 7. 去除游戏背景

按照预定的顺序，对游戏图像进行处理，其具体的步骤包括缩放、灰度化、亮度处理等。对于当前帧而言，其在进出下个状态前，需要对多个图像进行处理，通过叠加组合的方式获得多维图像数据。其中，当两个不同帧重叠以后，此时其灰度就会随之下降。

其中，蓝色部分、绿色部分、红色部分分别表示的全连接层、最大池化层以及卷积层。如图 8。

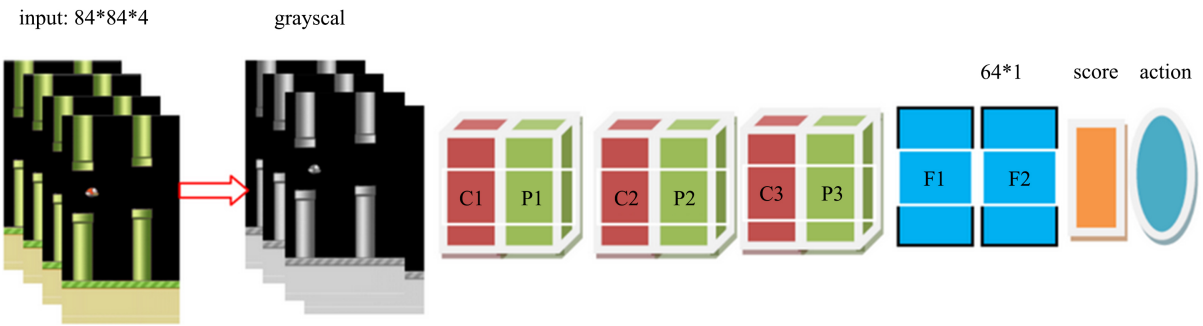


Figure 8. Image processing flowchart
图 8. 图像处理流程图

2.2. 深度 Q-Learning 算法

2.2.1. 深度 Q-Learning 结构

如下图 9 所示，在目前的模型结构中，其隐藏层的数量为 3 个。卷积层和完全连接层各有 2 个，其最终输出的信息为两个动作得分，其主要是通过损失函数计算而得到的。在对损失函数进行优化以后，可以设置 Q 学习参数。同时还以空间批量作为规范，针对每个卷积层，将加入最大池化层以及 ReLu。同时，在首个仿射层中，将利用 ReLu 激活函数，并对标准层进行批量出来。对于卷积层而言，其中设有滤波器。在每个时刻中，其输出操作有两种可能性，最终可以得到相应的得分值，从而确定最佳的动作。

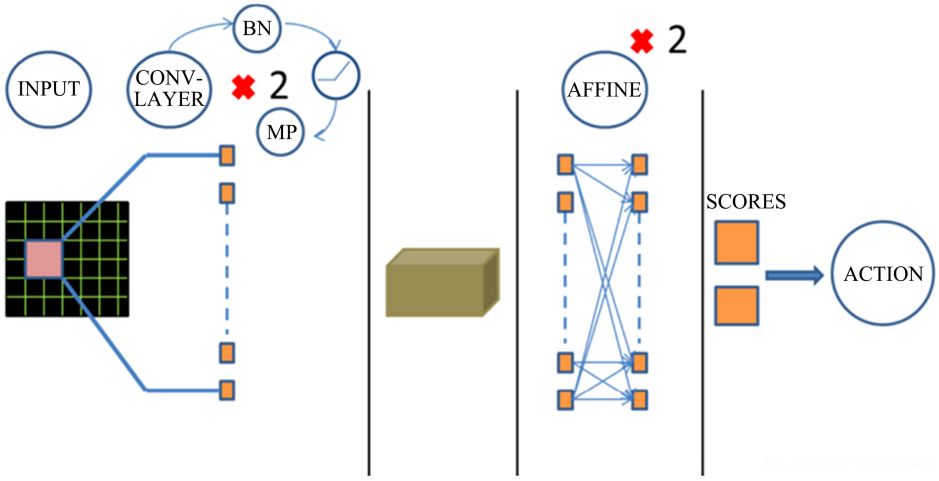


Figure 9. Convolutional neural network structure diagram
图 9. 卷积神经网络结构图

以模型处理精度作为基础，可以进一步提升模型复杂度。在本次设计中，将改进卷积层的数量，设立为 3 个。而全连接层以及池化层分别是 2 个和 3 个。在进行叠加时，需要对连续 4 帧预处理，并输入相应的图像值。在表 1 中，确定了各个步骤的数据和参数。

Table 1. Changes in parameters of each layer
表 1. 各层参数变化

Layer	Output	Activation	Num filters	Stride	Filter size	Input
conv1	20 × 20 × 32	ReLU	32	4	8 × 8	84 × 84 × 4
pooling1						20 × 20 × 32
conv2	8 × 8 × 64	ReLU	64	1	3 × 3	10 × 10 × 32
pooling2						8 × 8 × 64
conv3	2 × 2 × 64	ReLU	64	1	3 × 3	4 × 4 × 64
pooling3						2 × 2 × 64
Fc1						1 × 1 × 4
Fc2						1 × 64

2.2.2. 损失函数

强化学习的目标是使总回报最大化。而在进行 Q 学习的时候，其属于非策略。因此在进行迭代更新的时候，将利用贝尔曼方程，其计算公式为：

$$Q_{i+1}(s,a)=r+\gamma\max_{a'}Q_i(s',a')$$

在上述公式中，s'为下一帧状态，a'为下一帧动作，γ 是折扣因子，r 是奖励。(s,a)矩阵在第 i 次迭代的 Q 值使用 $Q_i(s,a)$ 表示。利用更新迭代的方式能够达到收敛的效果，从而得到最优的 Q-learning。同时，为了避免出现学习僵化的情况，可以使用函数的方式来表达动作值函数，可以对不可预见的状态进行概括。而对于学习算法而言，其输入点的构成是 s_t，a_t，r_t，s_{t+1}。利用函数，通过输入点可以构建相应的模型对动作进行预测。方程如下：

$$L=\sum_{s,a,r,s'}\left(Q(s,a;\theta)-\left(r+\gamma\max_{a'}Q(s',a';\theta^{-})\right)\right)^{2}$$

$$\Delta_{\theta}L=\sum_{s,a,r,s'}-2\left(Q(s,a;\theta)-\left(r+\gamma\max_{a'}Q(s',a';\theta^{-})\right)\right)\Delta_{\theta}Q(s,a;\theta)$$

其中，θ 是训练后的 DQN 参数，θ^{-}是 Q 值函数的非更新参数。通过该方式，可以对上述损失函数展开计算，并利用反向传播算法、随机梯度下降法，对神经网络权重进行更新。对于算法 1 而言，其设计的初衷是进行训练，以 ε 贪婪算法作为基础，并进行了强化。而在进行训练的时候，由于存在 ε 的概率，此时就可以得到最优的工作，其对应的函数关系为 $a_{opt}=\text{argmax}_{a'}Q(s,a';\theta)$。随着更新次数的提高，$\varepsilon$ 学会随之下降，最终变为 0。同时，为了提高损失函数的稳定性，可以利用方式 2，最终得到 DQN 模型。

2.2.3. 经验回放

在 Q-learning 中，而通过连续方式进行记录，其和经验数据之间具有密切的关系。在对 DQN 参数进行处理时，其使用的顺序相同，同时也会干扰整个训练的过程。针对这种情况，需要设置经验回放存储器，可以保存每帧游戏的有关数据(s_t,a_t,r_t,s_{t+1})，从而得到最大存储容量。当经验数据存量达到一定量以后，可以在 DQN 参数进行处理时将使用随机抽取小批经验。在对 DQN 参数进行处理，需要对其进行定期更新。考虑到在使用小批量选择结果时，其具有随机性，因此在对 DQN 参数进行更新时，需要确保其去相关。

3. 实验设计与仿真

3.1. 研究手段与途径

操作系统使用 Windows10 系统，安装 Anaconda3 (64 bit)和 PyCharm2019.3.4 (64 bit)集成编译环境，使用 python3.6.8 版本编译。需要的 package：pygame、sys、os、argparse、cycle、random 等。

3.2. 游戏设计

Flappy Bird 游戏要求通过控制一只小鸟，飞行间跨越由各种随机出现的不同高度的水管障碍，每次点击一下小鸟就会向上飞行，否则就会下落，游戏进行中必须控制节奏，拿捏好点击的时间点，让小鸟能在下落的瞬间跳起并恰好能够通过狭窄的水管缝隙。当小鸟顺利通过水管时，游戏加一分。当小鸟碰到水管或者是接触地面的时候，小鸟死亡，游戏立即宣布结束。Flappy Bird 游戏环境运行流程如图 10：

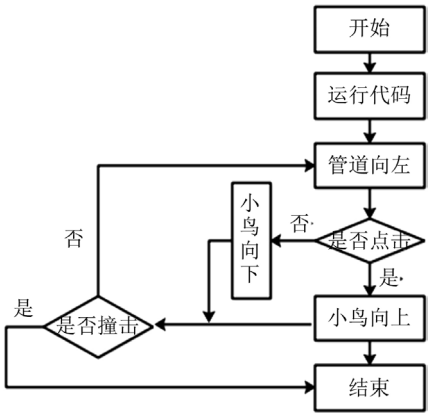


Figure 10. Flappy Bird flow chart
图 10. Flappy Bird 流程图

3.3. 游戏模型的建立

强化学习的主要过程主要是让小鸟学会如何躲避障碍物，其中三要素为状态(state)、动作(action)、奖励(reward)。小鸟需要根据当前状态来采取相应的动作，获得相应的奖励之后，再去改进这些动作，使得下次再遇到相同的状态时，小鸟能做出更优的动作。

小鸟只有两种动作的选择：触发空格键向上飞动或不操作向下自由降落。而背景图向左运动，使小鸟相对向右运动。

状态的选择是取整个游戏画面做图像处理，通过根据小鸟的高度和管子的距离，即取小鸟到下一根下侧管子的水平距离和垂直距离差作为小鸟的状态，由于两个水管之间距离是一定的，使用相对坐标，把坐标原点设置在水管的外面，水管宽度固定，只需要考虑小鸟的坐标。如图 11 所示：

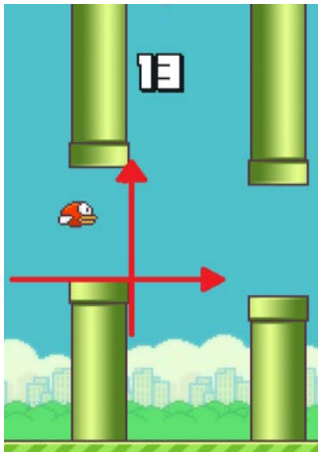


Figure 11. Location description
图 11. 位置描述

在当前的状态中，其构成包括经预处理以后的当前帧原始图像$\left(X\left(t\right)\right)$以及先前帧图像$\left(X\left(t-1\right),X\left(t-2\right),\cdots\right)$。通过这种方式，可以确保每个状态下，小鸟运动的位置及轨迹都是唯一对应的，可以及时提供信息给模型，在此时先前存储的帧数为超参数。如果运行方式理想，从 $t=1$ 起，其对应的所有帧函数利用 $S\left(t\right)$ 函数，但是为了降低状态空间的大小，使用帧的数量是有限的。通过上述分析可以得出，如果鸟撞到了屏幕边缘或者管道的时候，则会得到负面奖励。但是如果其顺利通过间隙，可以获得正面奖励。其和人类玩家类似，需要将可能降低死亡出现的频率，提高得分。而在本游戏中，奖励类型共有两种，包括 rewardPass 和 rewardDie。而 γ 则为折扣系统，设定为 0.9。

4. 实验结果分析

4.1. 训练参数

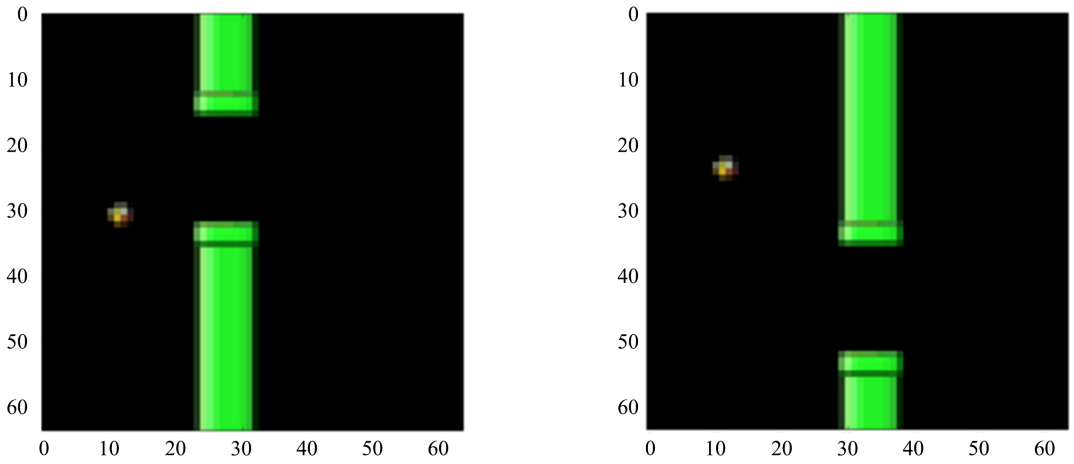
模型的参数设定：在 Flappy Bird 游戏中，其每秒播放 10 帧，而需要对最近的 4 帧进行处理，然后组合生产新状态。此时，$\gamma=0.9$；对于奖励设置而言，rewardDie = −1.0，rewardPass = +1.0。

DQN 参数：对于探索概率 ε 而言，当其更新速度为 600,000，此时会从 0.6 降低为 0，降低变化为线性。对于回放存储器而言，其设置的大小值是 20,000。其中，每当经验累积次数达到 500 次的时候，需要通过小批量采样的方式处理存储器中的数据。而当步长更新为 100，此时目标模型 θ^{-} 的参数也会进行相应的更新。

训练参数：将利用梯度下降更新法对 DQN 参数进行更新，此时学习率为 1e−6，$\beta 1 = 0.9$，$\beta 2 = 0.95$。并且以试错作为基础，确定相应的参数。并对损失值的收敛性进行观察。而在卷积进行处理时，其权重初始化是 0，其方差是 1e−2，呈现出正态分布的趋势。

4.2. 结果分析

在结束了训练以后，将利用模型的方式对游戏状态进行测试，从而判断是否可以得到结果。而在图 12 中，表示的是一部分游戏场景和预测结果。根据显示模型的情况，进行分类。

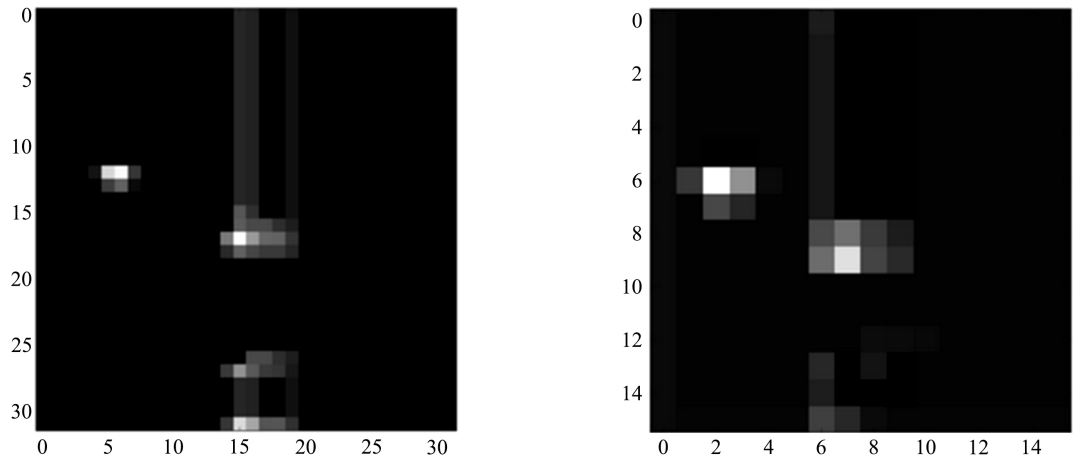


(a) Score: UP = **1.870**, DOWN = -1.830

(b) Score: UP = **-1.999**, DOWN = 1.983

Figure 12. Example game scene and its corresponding score
图 12. 示例游戏场景及其相应的分数

分析此次设计的工作原理，输出图像为下图 13，其中左图表示的是第一卷积层处理以后的图像，发现能够实现可视化。因此可以得知，当经过第二层卷积以后，就可以显示出清晰的斑块和空隙图 13 右图。通过数据分析可以得出：神经网络学习寻找空隙和小鸟的位置。



(a) After 1st conv layer

(b) After 2nd conv layer

Figure 13. Convolutional layer activation
图 13. 卷积层激活

通常情况下，利用 L2 损失函数来对雅达利游戏进行实验。在这种情况下，需要利用 L1 损失函数对其进行正则化处理。此时，得到的图像如下图 14，其在最开始的时候图像十分的陡峭。通过分析这两种情况可知，平均得分的稳定性提高，这表明模型在不断学习。通过对表 2 进行分析可以得知，当 DQN 模型训练结束以后，其成绩超过了人类水平。

$$L=\sum_{s,a,r,s'}\left|Q(s,a;\theta)-\left(r+\gamma\max_{a'}Q(s',a';\theta^{-})\right)\right|$$

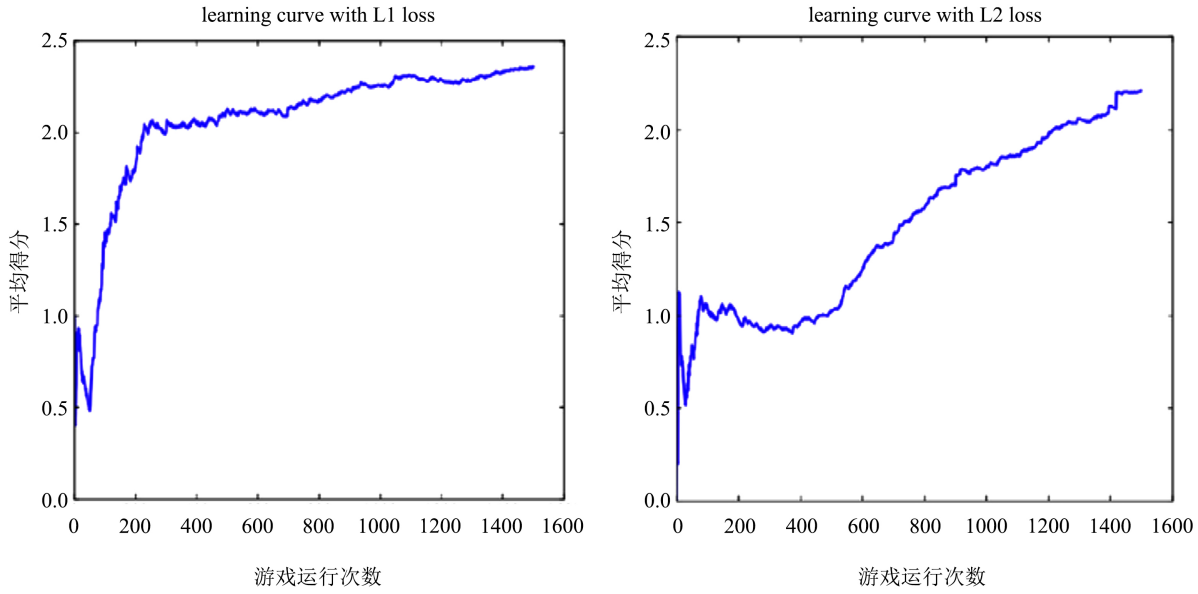


Figure 14. The learning curve of two different loss functions
图 14. 两种不同损失函数的学习曲线

通常情况下，利用 L2 损失函数来对雅达利游戏进行实验。在这种情况下，需要利用 L1 损失函数对其进行正则化处理。此时，得到的图像如下图 14，其在最开始的时候图像十分的陡峭。通过分析这两种情况可知，平均得分的稳定性提高，这表明模型在不断学习。通过对表 2 进行分析可以得知，当 DQN 模型训练结束以后，其成绩超过了人类水平。

$$L2=\sum_{s,a,r,s'}\left|Q(s,a;\theta)-\left(r+\gamma\max_{a'}Q(s',a';\theta^{-})\right)\right|$$

$$L1=\sum_{s,a,r,s'}\left(Q(s,a;\theta)-\left(r+\gamma\max_{a'}Q(s',a';\theta^{-})\right)\right)^{2}$$

Table 2. Player scores and DQN algorithm scores
表 2. 玩家得分与 DQN 算法得分

	玩家	DQN with L1	DQN with L2
平均得分	4.32	75	86
最高得分	25	205	335

5. 总结与展望

本文研究的是基于深度 Q-learning 算法实现 Flappy Bird 的自主学习游戏。通过应用卷积神经网络对游戏图像上的信息进行有效处理，而并非简单的图片分类处理。通过深度 Q-learning 算法可以在 Flappy Bird 游戏环境下选择最佳路径躲避障碍物，达到回报率最大。实现智能体对画面的每一帧分析状态，利用模型的方式对游戏处于各种状态进行测试，而不是常规的列表法分析状态。深度 Q-learning 算法实现了将策略游戏转化为高维图像分析识别，采用这种算法有效地解决了神经网络化繁为简的问题。由于 Q-learning 算法属于离线学习的一种，本文通过构建一个记忆库的方法，利用以往智能体(Flappy Bird)以往实验的经历，每一次更新都从记忆库中来更新神经网络的参数，不仅提高了实验效率，而且打破了经历之间的相关性[4]。

但由于深度 Q-learning 算法无法解决高难度的 Atari 游戏，当游戏维度升高到一定程度时，深度 Q-learning 算法无法在短时间得到有效的结果。并且卷积神经网络的结构有一定的局限性，会产生局部最优解而非全局最优解，导致智能体(Flappy Bird)不能达到非常高的分数，这些都是亟需解决的问题。

参考文献

[1] 刘炎锴. 基于深度学习的人脸身份认证方法研究[D]: [硕士学位论文]. 西安: 西安理工大学, 2017.

[2] 郭树旭, 马树志, 李晶, 张惠茅, 孙长建, 金兰依, 刘晓鸣, 刘奇楠, 李雪妍. 基于全卷积神经网络的肝脏 CT 影像分割研究[J]. 计算机工程与应用, 2017, 53(18): 126-131.

[3] 陈小平. 基于深度模型学习的跨模态检索[D]: [硕士学位论文]. 北京: 北京邮电大学, 2018.

[4] 晋帅, 李煊鹏, 何嘉颖, 李纾昶, 周敬淞. 基于强化学习的两轮模型车控制仿真分析[J]. 测控技术, 2019, 38(12): 115-121.