深度强化学习与双Q学习

哈多·范·哈塞尔特、亚瑟·格兹和大卫·西尔弗 Google D eepMind

摘要

流行的Q学习算法在某些条件下会高估动作值。此前尚不清楚在实践中这种高估是否普遍存在，是否会影响性能，以及是否能够普遍预防。在本文中，我们对所有这些问题都给出了肯定的回答。特别是，我们首先展示了结合Q学习与深度神经网络的近期DQN算法在Atari 2600领域的某些游戏中存在显著的高估现象。接着，我们证明了在表格环境中引入的双重Q学习算法的思想可以推广到大规模函数近似中。我们提出了对DQN算法的特定调整，并展示了所得到的算法不仅如假设的那样减少了观察到的高估，而且在多个游戏上带来了更好的性能。

强化学习（Sutton 和 Barto，1998）的目标是通过优化累积的未来奖励信号，为序列决策问题学习良好的策略。Q-learning（Watkins，1989）是最流行的强化学习算法之一，但它有时会学习到不切实际的高动作值，因为它包含了对估计动作值的最大化步骤，这往往倾向于高估而非低估的值。

在以往的研究中，高估现象被归因于函数逼近不够灵活（Thrun和Schwartz，1993）以及噪声（van Hasselt，2010，2011）。本文中，我们统一了这些观点，并表明当动作值不准确时，无论逼近误差的来源如何，都可能发生高估。当然，在学习过程中，价值估计不精确是常态，这表明高估可能比之前认为的要普遍得多。

目前尚不清楚的是，如果确实出现了高估，这是否会对实际性能产生负面影响。过于乐观的价值估计本身并不一定是个问题。如果所有价值都均匀地提高，那么相对的行动偏好得以保留，我们不会预期由此产生的策略会变得更差。此外，众所周知，有时乐观是有益的：面对不确定性时的乐观态度是一种广为人知的策略。

探索技术（Kaelbling 等，1996）。然而，如果高估并不均匀，并且不集中在我们希望进一步了解的状态上，那么它们可能会对最终策略的质量产生负面影响。Thrun 和 Schwartz（1993）给出了具体的例子，说明这会导致次优策略，甚至渐近地如此。

为了测试在实践中和大规模情况下是否会出现高估现象，我们研究了最近的DQN算法（Mnih等，2015）的表现。DQN将Q学习与灵活的深度神经网络相结合，并在多种多样的确定性Atari 2600游戏上进行了测试，在许多游戏中达到了人类水平的表现。在某些方面，这种设置是Q学习的最佳情况，因为深度神经网络提供了灵活的函数逼近，具有低渐近逼近误差的潜力，而环境的确定性则防止了噪声的有害影响。或许令人惊讶的是，我们表明，即使在这种相对有利的设置下，DQN有时也会显著高估动作的价值{v*}。

我们展示了Double Q-learning算法（van Hasselt, 2010）背后的思想，该算法最初是在表格环境下提出的，可以推广到适用于任意函数逼近，包括深度神经网络。我们利用这一点构建了一个新算法，称为Double DQN。随后，我们证明该算法不仅能产生更准确的价值估计，还能在多个游戏中获得更高的分数。这表明DQN的高估确实导致了较差的策略，减少这些高估是有益的。此外，通过对DQN的改进，我们在Atari领域获得了最先进的结果。

背景

为了解决序列决策问题，我们可以学习每个动作的最优值估计，定义为采取该动作并随后遵循最优策略时未来奖励的期望总和。在给定策略π下，状态s中动作a的真实值为

$$Q_\pi(s,a) \equiv \mathbb{E}\left[R_1 + \gamma R_2 + \ldots \mid S_0 = s, A_0 = a, \pi\right],$$

其中 $\gamma \in [0, 1]$ 是一个折扣因子，用于权衡即时奖励和后续奖励的重要性。最优值则为 $Q_*(s,a) = \max_\pi Q_\pi(s,a)$。通过在每个状态下选择价值最高的动作，可以轻松地从最优值中导出最优策略。

最优动作值的估计可以通过Q学习（Watkins, 1989）来学习，这是一种时间差分学习（Sutton, 1988）的形式。大多数有趣的问题规模太大，无法分别学习所有状态下的所有动作值。相反，我们可以学习一个参数化的价值函数$Q(s, a; \boldsymbol{\theta}_t)$。在状态$S_t$中采取动作$A_t$并观察到即时奖励$R_{t+1}$和结果状态$S_{t+1}$后，参数的标准Q学习更新为

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(Y_t^{\mathrm{Q}} - Q(S_t, A_t; \boldsymbol{\theta}_t))\nabla_{\boldsymbol{\theta}_t} Q(S_t, A_t; \boldsymbol{\theta}_t)\,. \quad (1)$$

其中 α 是标量步长，目标 Y_t^{Q} 定义为

$$Y_t^{\mathrm{Q}} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t)\,. \quad (2)$$

此更新类似于随机梯度下降，将当前值 $Q(S_t, A_t; \boldsymbol{\theta}_t)$ 向目标值 Y_t^{Q} 更新。

深度Q网络

深度Q网络（DQN）是一种多层神经网络，对于给定状态s，它输出一个动作值向量$Q(s, \cdot\,; \boldsymbol{\theta})$，其中$\boldsymbol{\theta}$是网络的参数。对于一个$n$维状态空间和包含$m$个动作的动作空间，神经网络是一个从$\mathbb{R}^n$到$\mathbb{R}^m$的函数。Mnih等人（2015）提出的DQN算法的两个重要组成部分是目标网络的使用和经验回放的使用。目标网络具有参数$\boldsymbol{\theta}^-$，除了其参数每隔τ步从在线网络复制一次，以便$\boldsymbol{\theta}_t^- = \boldsymbol{\theta}_t$，并在所有其他步骤中保持固定。DQN使用的目标网络即为如此。

$$Y_t^{\mathrm{DQN}} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t^-)\,. \quad (3)$$

对于经验回放（Lin, 1992），观察到的转换会存储一段时间，并从这个记忆库中均匀采样以更新网络。目标网络和经验回放都显著提高了算法的性能（Mnih等, 2015）。

双Q学习

标准Q学习和DQN中的max操作符，在(2)和(3)中，使用相同的值来选择并评估一个动作。这使得选择高估值的可能性更大，导致过于乐观的价值估计。为了防止这种情况，我们可以将选择与评估分离开来。这就是双Q学习（van Hasselt, 2010）背后的思想。

在原始的双Q学习算法中，通过将每个经验随机分配给更新两个价值函数之一来学习两个价值函数，因此存在两组权重，{v*}和$\boldsymbol{\theta}'$。每次更新时，一组权重用于确定贪婪策略，另一组用于确定其价值。为了清晰比较，我们可以首先解耦Q学习中的选择和评估，并将其目标（2）重写为

$$Y_t^{\mathrm{Q}} = R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}_t)\,.$$

Double Q-learning 误差可以表示为

$$Y_t^{\mathrm{DoubleQ}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}'_t)\,. \quad (4)$$

注意到，在argmax中选择动作仍然依赖于在线权重$\boldsymbol{\theta}_t$。这意味着，与Q学习一样，我们仍然根据当前值$\boldsymbol{\theta}_t$来估计贪婪策略的价值。然而，我们使用第二组权重$\boldsymbol{\theta}'_t$来公平评估该策略的价值。这第二组权重可以通过交换$\boldsymbol{\theta}$和$\boldsymbol{\theta}'$的角色来对称更新。

由于估计错误导致的过度乐观

Q-learning的过高估计首先由Thrun和Schwartz（1993）进行研究，他们表明，如果动作值包含在区间$[-\epsilon, \epsilon]$内均匀分布的随机误差，则每个目标会被高估至$\gamma\epsilon\frac{m-1}{m+1}$，其中$m$是动作的数量。此外，Thrun和Schwartz给出了一个具体例子，说明这些过高估计甚至渐近地导致次优策略，并展示了在使用函数逼近时，这些过高估计在一个小型玩具问题中的表现。后来van Hasselt（2010）提出，即使在使用表格表示时，环境中的噪声也可能导致过高估计，并提出Double Q-learning作为解决方案。

在本节中，我们更普遍地证明了任何类型的估计误差都可能引起向上的偏差，无论这些误差是由于环境噪声、函数近似、非平稳性还是任何其他来源。这一点很重要，因为在实践中，任何方法在学习过程中都会产生一些不准确性，这仅仅是因为真实值最初是未知的。

Thrun和Schwartz（1993）引用的上述结果为特定设置下的高估提供了一个上限，但同样可能且可能更有趣的是推导出一个下限。

定理 1. *Consider a state s in which all the true optimal action values are equal at $Q_*(s, a) = V_*(s)$ for some $V_*(s)$. Let Q_t be arbitrary value estimates that are on the whole unbiased in the sense that $\sum_a (Q_t(s, a) - V_*(s)) = 0$, but that are not all correct, such that $\frac{1}{m}\sum_a (Q_t(s, a) - V_*(s))^2 = C$ for some $C > 0$, where $m \geq 2$ is the number of actions in s. Under these conditions, $\max_a Q_t(s, a) \geq V_*(s) + \sqrt{\frac{C}{m-1}}$.*

This lower bound is tight. Under the same conditions, the lower bound on the absolute error of the Double Q-learning estimate is zero. (Proof in appendix.)

请注意，我们无需假设不同动作的估计误差是独立的。该定理表明，即使价值估计在平均意义上是正确的，任何来源的估计误差都可能导致估计值偏离真实的最优值{v*}。

定理1中的下界随着动作数量的增加而减小。这是考虑下界的一个结果，它要求达到非常特定的值。更典型的情况是，过度乐观随着动作数量的增加而增加，如图1所示。Q学习的过度估计确实随着动作数量的增加而增加，

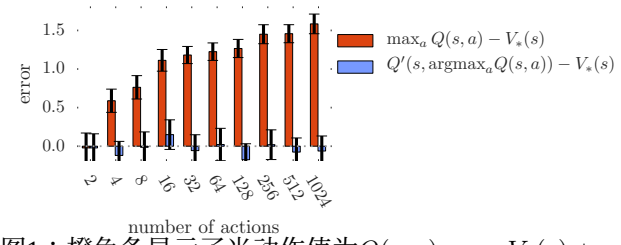

图1：橙色条显示了当动作值为$Q(s,a) = V_*(s) + \epsilon_a$且误差$\{\epsilon_a\}_{a=1}^m$为独立标准正态随机变量时，单次Q学习更新中的偏差。用于蓝色条的第二组动作值Q'是以相同且独立的方式生成的。所有条均为100次重复的平均值。

虽然双Q学习是无偏的。再举一个例子，如果对于所有动作$Q_*(s,a) = V_*(s)$和估计误差$Q_t(s,a) - V_*(s)$在[−1, 1]中均匀随机，那么过度乐观为$\frac{m-1}{m+1}$。（证明见附录。）我们现在转向函数逼近，并考虑一个实值连续状态空间，每个状态有10个离散动作。为简化起见，本例中的真实最优动作值仅依赖于状态，因此每个状态下所有动作的真实值相同。这些真实值如图2左侧图中的紫色线所示，定义为$Q_*(s,a) = \sin(s)$（上排）或$Q_*(s,a) = 2\exp(-s^2)$（中排和下排）。左侧图还显示了单个动作的逼近（绿线）作为状态的函数，以及估计所基于的样本（绿点）。估计是一个d次多项式，拟合于采样状态的真实值，其中$d = 6$（上排和中排）或$d = 9$（下排）。样本与真实函数完全匹配：没有噪声，我们假设在这些采样状态上拥有动作值的真实数据。对于上两排，即使在采样状态上，逼近也不准确，因为函数逼近不够灵活。在下排，函数足够灵活以拟合绿点，但这降低了未采样状态的准确性。注意，采样状态在左侧图的左侧间隔较远，导致更大的估计误差。在许多方面，这是一个典型的学习场景，每个时间点我们只有有限的数据。

图2中间列的图表展示了所有10个动作的估计动作价值函数（绿色线条），作为状态的函数，以及每个状态中的最大动作价值（黑色虚线）。尽管所有动作的真实价值函数相同，但由于我们提供了不同的采样状态集，近似值有所不同。[1] 最大值通常高于左侧紫色显示的真实值。这在右侧图表中得到证实，该图表以橙色显示了黑色和紫色曲线之间的差异。橙色线条几乎总是正值，表明存在向上的偏差。右侧图表还以蓝色显示了Double Q-learning的估计值[2]，这些值平均上更接近零。这表明Double Q-learning确实能够成功减少Q-learning的过度乐观。

图2中的不同行展示了同一实验的变体。顶部和中间行之间的差异在于真实值函数，表明高估并非特定真实值函数的产物。中间和底部行之间的差异在于函数逼近的灵活性。在左中图中，由于函数不够灵活，对某些采样状态的估计甚至是不正确的。左下图中的函数更为灵活，但这导致了对未见状态的更高估计误差，从而产生了更高的高估。这一点很重要，因为在强化学习中经常使用灵活的参数函数逼近器（参见，例如，Tesauro 1995；Sallans和Hinton 2004；Riedmiller 2005；Mnih等，2015）。

与van Hasselt（2010年）不同，我们并未采用统计论据来发现高估现象，获取图2的过程是完全确定性的。与Thrun和Schwartz（1993年）相比，我们没有依赖具有不可约渐近误差的僵化函数逼近；底行显示，一个足够灵活以覆盖所有样本的函数会导致高度的高估。这表明高估现象可能相当普遍地发生。

在上述示例中，即使假设我们在某些状态下拥有*true*动作值的样本，仍然会出现高估现象。如果我们基于已经过于乐观的动作值进行自举，价值估计可能会进一步恶化，因为这会导致高估在整个估计中传播。尽管*uniformly*高估价值可能不会损害最终策略，但在实践中，不同状态和动作的高估误差会有所不同。高估与自举相结合，会产生有害的影响，传播关于哪些状态比其他状态更有价值的错误相对信息，直接影响学习策略的质量。

不应将高估与面对不确定性时的乐观主义混淆（Sutton, 1990; Agrawal, 1995; Kaelbling et al., 1996; Auer et al., 2002; Brafman and Tennenholtz, 2003; Szita and Lőrincz, 2008; Strehl et al., 2009），后者是对具有不确定价值的状态或行动给予探索奖励。相反，这里讨论的高估仅在更新后发生，导致在面对明显确定性时产生过度乐观。Thrun和Schwartz（1993）已经观察到这一点，他们指出，与面对不确定性时的乐观不同，这些高估实际上可能阻碍学习最优策略。我们将在后续实验中再次证实这种对策略质量的负面影响：当我们使用双Q学习减少高估时，策略会得到改善。

[1] Each action-value function is fit with a different subset of integer states. States -6 and 6 are always included to avoid extrapolations, and for each action two adjacent integers are missing: for action a_1 states -5 and -4 are not sampled, for a_2 states -4 and -3 are not sampled, and so on. This causes the estimated values to differ.

[2] We arbitrarily used the samples of action a_{i+5} (for $i \leq 5$) or a_{i-5} (for $i > 5$) as the second set of samples for the double estimator of action a_i.

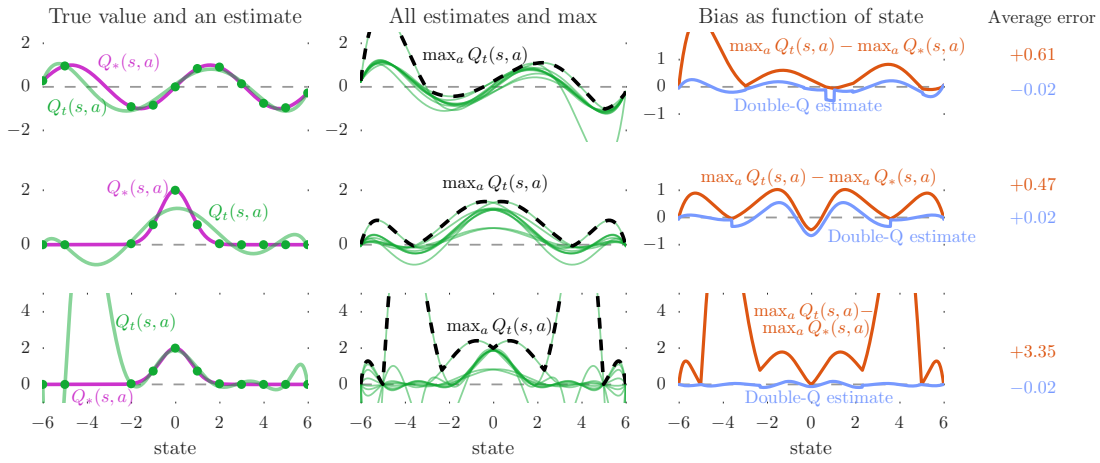

图2：学习过程中高估现象的图示。在每个状态（x轴）下，有10个动作。左列显示了真实值$V_*(s)$ (紫色线)。所有真实动作值由$Q_*(s,a) = V_*(s)$定义。绿线显示了一个动作的估计值$Q(s,a)$作为状态的函数，拟合到几个采样状态（绿点）的真实值。中间列的图表显示了所有估计值（绿色），以及这些值的最大值（黑色虚线）。最大值几乎在所有地方都高于真实值（紫色，左图）。右列图表以橙色显示了差异。右图中的蓝线是双Q学习使用的估计值，每个状态使用第二组样本。蓝线更接近零，表明偏差较小。三行对应于不同的真实函数（左，紫色）或拟合函数的容量（左，绿色）。（详见正文）

双DQN

双Q学习的思想是通过将目标中的最大操作分解为动作选择和动作评估来减少过高估计。尽管没有完全解耦，DQN架构中的目标网络为第二个价值函数提供了一个自然的候选，而无需引入额外的网络。因此，我们建议根据在线网络评估贪婪策略，但使用目标网络来估计其价值。参考双Q学习和DQN，我们将所得算法称为双DQN。其更新与DQN相同，但将目标{v*}替换为

$$Y_t^{\text{DoubleDQN}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \boldsymbol{\theta}_t), \boldsymbol{\theta}_t^-) .$$

与双Q学习（4）相比，第二个网络$\boldsymbol{\theta}_t'$的权重被替换为目标网络$\boldsymbol{\theta}_t^-$的权重，用于评估当前贪婪策略。目标网络的更新与DQN保持不变，仍然是定期复制在线网络。

这个版本的Double DQN可能是对DQN进行最小改动以实现Double Q-learning的方式。目标是获得Double Q-learning的大部分好处，同时保持DQN算法的其余部分不变以进行公平比较，并且计算开销最小。

实证结果

在本节中，我们分析了DQN的高估现象，并展示了Double DQN在值准确性和策略质量方面均优于DQN。为了进一步测试该方法的鲁棒性，我们还按照Nair等人（2015）提出的方法，使用从专家人类轨迹生成的随机起点对算法进行了评估。

我们的测试平台由Atari 2600游戏组成，使用Arcade学习环境（Bellemare等，2013）。目标是让一个算法，在固定一组超参数的情况下，能够仅通过屏幕像素作为输入，从交互中学会分别玩每一款游戏。这是一个要求极高的测试平台：不仅输入是高维的，游戏视觉效果和游戏机制在不同游戏之间差异巨大。因此，优秀的解决方案必须极大地依赖于学习算法——仅依靠调参来过度拟合领域在实际中是不可行的。

我们严格遵循Mnih等人（2015年）提出的实验设置和网络架构。简而言之，该网络架构是一个卷积神经网络（Fukushima, 1988; LeCun et al., 1998），包含3个卷积层和一个全连接的隐藏层（总计约150万个参数）。网络以最后四帧作为输入，并输出每个动作的动作值{v*}。在每款游戏上，网络在单个GPU上训练2亿帧，大约耗时1周。

关于过度乐观的结果

图3展示了DQN在六款Atari游戏中的高估示例。DQN和Double DQN均在Mnih等人（2015）所述的完全相同条件下进行训练。DQN对当前贪婪策略的价值始终且有时过于乐观，这一点可以通过比较顶部图中橙色学习曲线与代表最佳学习策略实际折现值的橙色直线看出。更准确地说，（平均）价值估计是在训练期间定期计算的，每次完整评估阶段长度为$T = 125,000$步。

$$\frac{1}{T}\sum_{t=1}^{T} \operatorname*{argmax}_a Q(S_t, a; \boldsymbol{\theta}) .$$

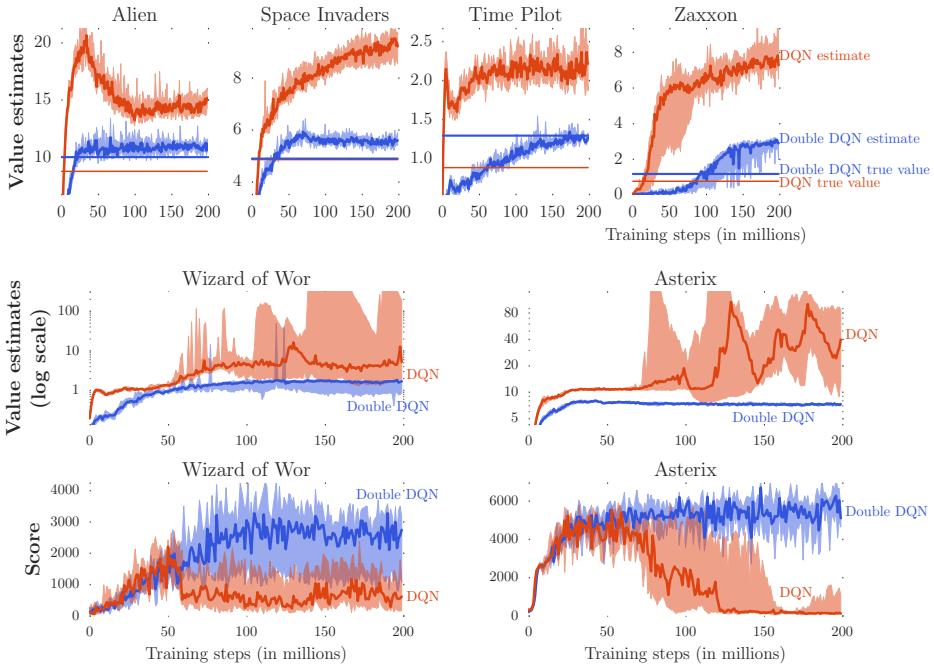

图3：顶部和中间行展示了DQN（橙色）和Double DQN（蓝色）在六款Atari游戏上的价值估计。这些结果是通过使用Mnih等人（2015年）采用的超参数，运行DQN和Double DQN并采用6种不同的随机种子获得的。较深的线显示了种子间的中位数，我们通过平均两个极值来获得阴影区域（即10%和90%分位数，并进行线性插值）。顶部行中的水平橙色（DQN）和蓝色（Double DQN）直线是通过在学习结束后运行相应的代理，并从每个访问状态获得的实际折扣回报的平均值计算得出的。如果没有偏差，这些直线将与图右侧的学习曲线相匹配。中间行展示了两款游戏中DQN过度乐观的价值估计（对数尺度）。底部行展示了这种过度乐观对代理在训练期间评估得分的有害影响：当过度估计开始时，得分下降。使用Double DQN学习则更加稳定。

通过运行最佳学习策略多个回合并计算实际累积奖励，获得了地面真实平均值。如果没有过高估计，我们期望这些量能够匹配（即曲线与每个图右侧的直线匹配）。然而，DQN的学习曲线始终远高于真实值。以蓝色显示的双DQN学习曲线更接近代表最终策略真实值的蓝色直线。请注意，蓝色直线通常高于橙色直线。这表明双DQN不仅产生了更准确的价值估计，还产生了更好的策略。

中间两幅图展示了更为极端的高估现象，其中DQN在游戏《Asterix》和《Wizard of Wor》上表现出极不稳定性。请注意y轴上的数值采用了对数刻度。底部两幅图则显示了这两款游戏的相应得分。注意到中间图中DQN价值估计的增加与底部图中得分的下降相吻合。这再次表明，高估现象损害了最终策略的质量。如果单独来看，人们可能会认为观察到的不稳定性与使用函数逼近的离策略学习固有的不稳定性问题有关（Baird, 1995; Tsitsiklis and Van Roy, 1997; Sutton et al., 2008; Maei, 2011; Sutton et al., 2015）。然而，我们发现使用Double DQN时学习过程要稳定得多，

	DQN	Double DQN
Median	93.5%	114.7%
Mean	241.1%	330.3%

表1：49款游戏在5分钟游戏时间内的归一化性能总结。DQN的结果来自Mnih等人（2015）

这表明这些不稳定性的原因实际上是Q学习的过度乐观。图3仅展示了一些例子，但在所有49个测试的Atari游戏中，DQN都观察到了高估现象，尽管程度有所不同。

学习策略的质量

过度乐观并不总是对学习到的策略质量产生负面影响。例如，尽管DQN在Pong游戏中略微高估了策略价值，但它仍能实现最优行为。然而，减少高估可以显著提高学习的稳定性；我们在图3中看到了明显的例子。现在，我们通过在所有49款DQN测试过的游戏上进行评估，更广泛地评估Double DQN在策略质量方面的帮助程度。如Mnih等人（2015）所述，每个评估回合开始时，会执行一个特殊的无操作动作，该动作对环境无影响，最多执行30次，以为代理提供不同的起点。评估期间的一些探索提供了额外的随机性。对于Double DQN，我们使用了与DQN完全相同的超参数，

	DQN	Double DQN	Double DQN (tuned)
Median	47.5%	88.4%	116.7%
Mean	122.0%	273.1%	475.2%

表2：在49款以人类开局为基础的游戏中，至多30分钟游戏时间的归一化性能总结。DQN的结果来自Nair等人（2015）。

为了进行一项专注于减少高估的受控实验。学习到的策略在模拟器时间5分钟（18,000帧）内通过ϵ-贪婪策略进行评估，其中$\epsilon = 0.05$。分数在100次回合中取平均值。Double DQN与DQN之间的唯一区别在于目标，使用$Y_t^{\text{DoubleDQN}}$而非Y^{DQN}。这一评估具有一定的对抗性，因为所使用的超参数是为DQN而非Double DQN调整的。

为了获得跨游戏的汇总统计，我们对每个游戏的分数进行如下归一化处理：

$$\text{score}_{\text{normalized}} = \frac{\text{score}_{\text{agent}} - \text{score}_{\text{random}}}{\text{score}_{\text{human}} - \text{score}_{\text{random}}}. \qquad (5)$$

“随机”和“人类”得分与Mnih等人（2015年）使用的相同，并在附录中给出。

表1在无操作情况下显示，总体上Double DQN明显优于DQN。详细比较（见附录）表明，有几款游戏中Double DQN大幅超越DQN，值得注意的例子包括《Road Runner》（从233%提升至617%）、《Asterix》（从70%提升至180%）、《Zaxxon》（从54%提升至111%）以及《Double Dunk》（从17%跃升至397%）。

Gorila算法（Nair等，2015年），作为DQN的大规模分布式版本，未包含在表格中，因为其架构和基础设施差异显著，难以进行直接比较。为完整起见，我们指出Gorila获得的中位数和平均归一化分数分别为96%和495%。

对人类起始的鲁棒性

先前评估的一个担忧是，在具有唯一起始点的确定性游戏中，学习者可能学会记住动作序列，而无需太多泛化。虽然成功，但这种解决方案并不特别稳健。通过从不同起始点测试智能体，我们可以测试所找到的解决方案是否具有良好的泛化能力，从而为学习到的策略提供一个具有挑战性的测试平台（Nair等，2015）。

我们按照Nair等人（2015年）提出的方法，从人类专家的轨迹中为每个游戏采样了100个起点。我们从这些起点中的每一个开始一个评估片段，并运行模拟器最多108,000帧（在60Hz下为30分钟，包括起点前的轨迹）。每个智能体仅在起点之后累积的奖励上进行评估。

在此评估中，我们包含了一个经过调优的双重DQN版本。进行适当的调优是必要的，因为超参数原本是为DQN调整的，而DQN是另一种算法。对于调优后的双重DQN版本，我们将目标网络每两次复制之间的帧数从10,000增加到了30,000，以进一步减少高估现象，因为在每次切换后，DQN和双重DQN会立即出现这种情况。

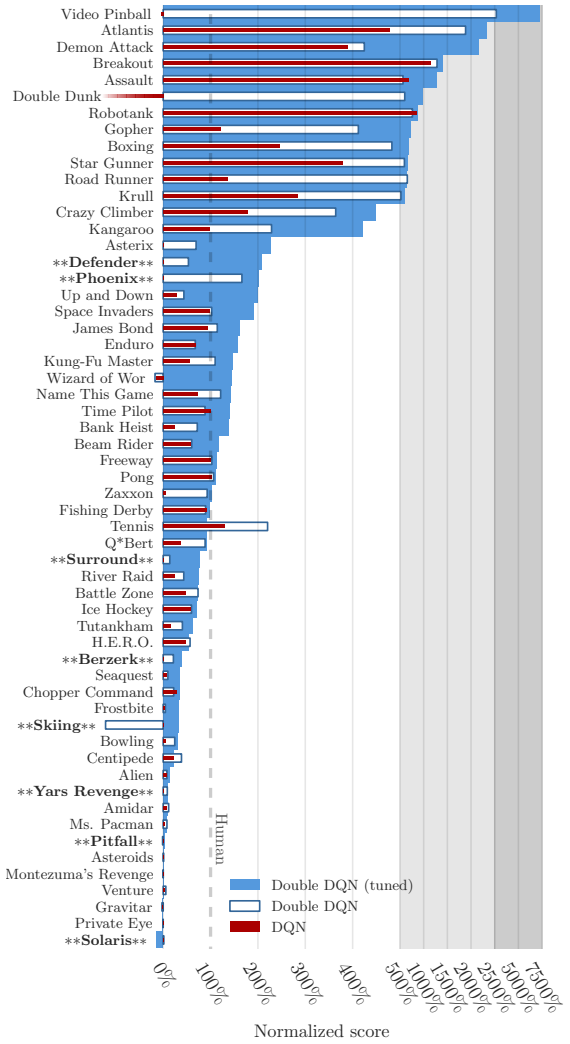

图4：在57款Atari游戏上的归一化得分，每款游戏测试100次，采用人类起始点。与Mnih等人（2015年）相比，额外测试了八款游戏。这些游戏用星号和粗体字标示。

两者都回归到Q学习。此外，我们将学习期间的探索从$\epsilon = 0.1$减少到$\epsilon = 0.01$，然后在评估期间使用$\epsilon = 0.001$。最终，调整后的版本在网络顶层为所有动作值使用单一共享偏差。这些变化中的每一个都提高了性能，共同作用带来了明显更好的结果。[3]

表2报告了针对Mnih等人（2015年）49款游戏的评估汇总统计。Double DQN明显获得了更高的中位数和平均分数。再次说明，Gorila DQN（Nair等人，2015年）未包含在表中，但为完整起见，请注意其中位数为78%，平均数为259%。详细结果，以及另外8款游戏的结果，可在图4和附录中查阅。在几款游戏中，从DQN到Double DQN的改进显著，在某些情况下使分数更接近{v*}。

[3] Except for Tennis, where the lower ϵ during training seemed to hurt rather than help.

人类，甚至超越这些。

Double DQN 在这种更具挑战性的评估中表现出更强的鲁棒性，这表明发生了适当的泛化，并且所找到的解决方案并未利用环境的确定性。这一点颇具吸引力，因为它表明在寻找通用解决方案方面取得了进展，而不是依赖于确定性步骤序列，后者往往不够鲁棒。

讨论

本文有五项贡献。首先，我们展示了为什么在大规模问题中，即使这些问题具有确定性，由于学习固有的估计误差，Q学习可能会过于乐观。其次，通过分析Atari游戏中的价值估计，我们表明这些高估在实践中比以往认识到的更为普遍和严重。第三，我们证明了大规模使用双Q学习可以成功减少这种过度乐观，从而实现更稳定和可靠的学习。第四，我们提出了一种名为双DQN的具体实现，它利用了DQN算法的现有架构和深度神经网络，无需额外的网络或参数。最后，我们展示了双DQN能够找到更好的策略，在Atari 2600领域取得了新的最先进成果。

致谢

我们要感谢Tom Schaul、Volodymyr Mnih、Marc Bellemare、Thomas Degris、Georg Ostrovski和Richard Sutton的有益评论，以及Google DeepMind的每个人为我们提供了建设性的研究环境。

参考文献

R. Agrawal. 基于样本均值的指数策略，具有O(log n)遗憾的多臂赌博机问题。*Advances in Applied Probability*, 页码1054 – 1078, 1995.

P. Auer, N. Cesa-Bianchi, 和 P. Fischer. 多臂赌博机问题的有限时间分析。*Machine learning*, 47(2-3):235 – 256, 2002.

L. Baird. 残差算法：基于函数逼近的强化学习。在 *Machine Learning: Proceedings of the Twelfth International Conference*, 页码30 – 37, 1995.

M. G. Bellemare, Y. Naddaf, J. Veness, 和 M. Bowling. Arcade学习环境：通用代理的评估平台。*J. Artif. Intell. Res. (JAIR)*, 47:253 – 279, 2013.

R. I. Brafman 和 M. Tennenholtz. R-max：一种接近最优强化学习的通用多项式时间算法。*The Journal of Machine Learning Research*, 3:213 – 231, 2003.

K. Fukushima. 新认知机：一种能够进行视觉模式识别的分层神经网络。*Neural networks*, 1(2):119 – 130, 1988.

L. P. Kaelbling, M. L. Littman, 和 A. W. Moore. 强化学习：综述。*Journal of Artificial Intelligence Research*, 4:237 – 285, 1996.

Y. LeCun, L. Bottou, Y. Bengio, 和 P. Haffner. 基于梯度的学习应用于文档识别。*Proceedings of the IEEE*, 86(11):2278 – 2324, 1998.

L. Lin. 基于强化学习、规划和教学的自我改进反应代理。*Machine learning*, 8(3):293 – 321, 1992.

H. R. Maei. *Gradient temporal-difference learning algorithms*。博士论文，阿尔伯塔大学，2011年。

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, 和 D. Hassabis。通过深度强化学习实现人类水平的控制。*Nature*, 518 (7540):529 – 533, 2015年。

A. Nair, P. Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A. D. Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, S. Legg, V. Mnih, K. Kavukcuoglu, 和 D. Silver。深度强化学习的大规模并行方法。在 *Deep Learning Workshop, ICML*, 2015年。

M. Riedmiller。神经拟合Q迭代——一种数据高效的神经强化学习方法的初步经验。在 J. Gama, R. Camacho, P. Brazdil, A. Jorge, 和 L. Torgo 编辑的 *Proceedings of the 16th European Conference on Machine Learning (ECML'05)*, 第317 – 328页。Springer, 2005年。

B. Sallans 和 G. E. Hinton。基于因子化状态和动作的强化学习。*The Journal of Machine Learning Research*, 5:1063 – 1088, 2004年。

A. L. Strehl, L. Li, 和 M. L. Littman。有限MDP中的强化学习：PAC分析。*The Journal of Machine Learning Research*, 10:2413 – 2444, 2009年。

R. S. Sutton。通过时间差分方法进行预测学习。*Machine learning*, 3(1):9 – 44, 1988年。

R. S. Sutton。基于近似动态规划的学习、规划和反应的集成架构。在 *Proceedings of the seventh international conference on machine learning*，第216 – 224页，1990年。

R. S. Sutton 和 A. G. Barto。*Introduction to reinforcement learn- ing*。MIT Press, 1998年。

R. S. Sutton, C. Szepesvári, 和 H. R. Maei。一种收敛的O(n)算法，用于具有线性函数近似的时间差分学习。*Advances in Neural Information Processing Systems 21 (NIPS-08)*, 21:1609 – 1616, 2008年。

R. S. Sutton, A. R. Mahmood, 和 M. White。一种强调方法解决时间差分学习中的离策略问题。*arXiv preprint arXiv:1503.04269*, 2015年。

I. Szita 和 A. Lőrincz。乐观的多种面貌：一种统一的方法。在 *Proceedings of the 25th international conference on Machine learning*，第1048 – 1055页。ACM, 2008年。

G. Tesauro。时间差分学习与td-gammon。*Communications of the ACM*, 38(3):58 – 68, 1995年。

S. Thrun 和 A. Schwartz。在强化学习中使用函数逼近的问题。在 M. Mozer, P. Smolensky, D. Touretzky, J. Elman, 和 A. Weigend 编辑的 *Proceedings of the 1993 Connectionist Models Summer School*，Hillsdale, NJ, 1993年。Lawrence Erlbaum。

J. N. Tsitsiklis 和 B. Van Roy。时间差分学习与函数逼近的分析。*IEEE Transactions on Automatic Control*, 42(5):674 – 690, 1997年。

H. van Hasselt。双Q学习。*Advances in Neural Information Processing Systems*, 23:2613 – 2621, 2010年。

H. van Hasselt。*Insights in Reinforcement Learning*。博士论文，乌得勒支大学，2011年。

C. J. C. H. Watkins。*Learning from delayed rewards*。博士论文，剑桥大学，1989年。

附录

定理 1. *Consider a state s in which all the true optimal action values are equal at $Q_*(s,a) = V_*(s)$ for some $V_*(s)$. Let Q_t be arbitrary value estimates that are on the whole unbiased in the sense that $\sum_a (Q_t(s,a) - V_*(s)) = 0$, but that are not all zero, such that $\frac{1}{m}\sum_a (Q_t(s,a) - V_*(s))^2 = C$ for some $C > 0$, where $m \geq 2$ is the number of actions in s. Under these conditions, $\max_a Q_t(s,a) \geq V_*(s) + \sqrt{\frac{C}{m-1}}$. This lower bound is tight. Under the same conditions, the lower bound on the absolute error of the Double Q-learning estimate is zero.*

Proof of Theorem 1. 定义每个动作a的误差为$\epsilon_a = Q_t(s,a) - V_*(s)$。假设存在$\{\epsilon_a\}$的某种设置，使得$\max_a \epsilon_a < \sqrt{\frac{C}{m-1}}$。令$\{\epsilon_i^+\}$为大小为$n$的正$\epsilon$集合，$\{\epsilon_j^-\}$为大小为$m-n$ (的严格负ϵ集合，使得$\{\epsilon\} = \{\epsilon_i^+\} \cup \{\epsilon_j^-\}$。如果$n = m$，则$\sum_a \epsilon_a = 0 \Longrightarrow \epsilon_a = 0\, \forall a$，这与$\sum_a \epsilon_a^2 = mC$矛盾。因此，必有$n \leq m-1$。于是，$\sum_{i=1}^n \epsilon_i^+ \leq n \max_i \epsilon_i^+ < n\sqrt{\frac{C}{m-1}}$，因此（利用约束$\sum_a \epsilon_a = 0$）我们也有$\sum_{j=1}^{m-n} |\epsilon_j^-| < n\sqrt{\frac{C}{m-1}}$。这意味着$\max_j |\epsilon_j^-| < n\sqrt{\frac{C}{m-1}}$。根据Hölder不等式，则

$$\begin{aligned}\sum_{j=1}^{m-n} (\epsilon_j^-)^2 &\leq \sum_{j=1}^{m-n} |\epsilon_j^-| \cdot \max_j |\epsilon_j^-| \\ &< n\sqrt{\frac{C}{m-1}} n\sqrt{\frac{C}{m-1}}\,.\end{aligned}$$

我们现在可以结合这些关系来计算所有{v*}的平方和的上界：

$$\begin{aligned}\sum_{a=1}^{m} (\epsilon_a)^2 &= \sum_{i=1}^{n} (\epsilon_i^+)^2 + \sum_{j=1}^{m-n} (\epsilon_j^-)^2 \\ &< n\frac{C}{m-1} + n\sqrt{\frac{C}{m-1}} n\sqrt{\frac{C}{m-1}} \\ &= C\frac{n(n+1)}{m-1} \\ &\leq mC.\end{aligned}$$

这与假设$\sum_{a=1}^m \epsilon_a^2 < mC$相矛盾，因此对于满足约束条件的所有$\epsilon$设置，$\max_a \epsilon_a \geq \sqrt{\frac{C}{m-1}}$成立。我们可以通过设置$\epsilon_a = \sqrt{\frac{C}{m-1}}$为$a = 1, \ldots, m-1$和$\epsilon_m = -\sqrt{(m-1)C}$来验证下界是紧的。这验证了$\sum_a \epsilon 2_a = mC$和$\sum_a \epsilon_a = 0$.

Double Q-学习$|Q'_t(s, \text{argmax}_a\, Q_t(s,a)) - V_*(s)|$的绝对误差的唯一紧下界为零。这可以通过以下原因看出：我们可以有

$$Q_t(s,a_1) = V_*(s) + \sqrt{C\frac{m-1}{m}}\,,$$

和

$$Q_t(s,a_i) = V_*(s) - \sqrt{C\frac{1}{m(m-1)}}\,, \text{for } i > 1.$$

那么定理的条件成立。如果进一步我们有$Q'_t(s,a_1) = V_*(s)$，那么误差为零。对于$i > 1$，其余的动作值$Q'_t(s,a_i)$是任意的。□

定理2. *Consider a state s in which all the true optimal action values are equal at $Q_*(s,a) = V_*(s)$. Suppose that the estimation errors $Q_t(s,a) - Q_*(s,a)$ are independently distributed uniformly randomly in $[-1, 1]$. Then,*

$$\mathbb{E}\left[\max_a Q_t(s,a) - V_*(s)\right] = \frac{m-1}{m+1}$$

Proof. 定义 $\epsilon_a = Q_t(s,a) - Q_*(s,a)$；这是一个在 $[-1,1]$ 上的均匀随机变量。对于某些 x，$\max_a Q_t(s,a) \leq x$ 的概率等于所有 a 同时满足 $\epsilon_a \leq x$ 的概率。由于估计误差是独立的，我们可以推导出

$$\begin{aligned}P(\max_a \epsilon_a \leq x) &= P(X_1 \leq x \wedge X_2 \leq x \wedge \ldots \wedge X_m \leq x) \\ &= \prod_{a=1}^{m} P(\epsilon_a \leq x)\,.\end{aligned}$$

函数 $P(\epsilon_a \leq x)$ 是 ϵ_a 的累积分布函数（CDF），在此简单定义为

$$P(\epsilon_a \leq x) = \begin{cases} 0 & \text{if } x \leq -1 \\ \frac{1+x}{2} & \text{if } x \in (-1,1) \\ 1 & \text{if } x \geq 1 \end{cases}$$

这意味着

$$\begin{aligned}P(\max_a \epsilon_a \leq x) &= \prod_{a=1}^{m} P(\epsilon_a \leq x) \\ &= \begin{cases} 0 & \text{if } x \leq -1 \\ \left(\frac{1+x}{2}\right)^m & \text{if } x \in (-1,1) \\ 1 & \text{if } x \geq 1 \end{cases}\end{aligned}$$

这给出了随机变量 $\max_a \epsilon_a$ 的累积分布函数（CDF）。其期望可以写成一个积分

$$\mathbb{E}\left[\max_a \epsilon_a\right] = \int_{-1}^{1} x f_{\max}(x)\, \mathrm{d}x\,,$$

其中 $f_{\max}$ 是该变量的概率密度函数，定义为累积分布函数的导数：$f_{\max}(x) = \frac{\mathrm{d}}{\mathrm{d}x} P(\max_a \epsilon_a \leq x)$，因此对于 $x \in [-1,1]$，我们有 $f_{\max}(x) = \frac{m}{2}\left(\frac{1+x}{2}\right)^{m-1}$。积分求值得出

$$\begin{aligned}\mathbb{E}\left[\max_a \epsilon_a\right] &= \int_{-1}^{1} x f_{\max}(x)\, \mathrm{d}x \\ &= \left[\left(\frac{x+1}{2}\right)^m \frac{mx-1}{m+1}\right]_{-1}^{1} \\ &= \frac{m-1}{m+1}\,.\end{aligned}$$

□

Atari 2600领域的实验细节

我们选择了49款游戏以匹配Mnih等人（2015年）使用的列表，完整列表见下表。每个智能体步骤由四帧组成（在这些帧中重复最后选择的动作），并且奖励值（从Arcade Learning Environment（Bellemare等人，2013年）获得）被限制在-1和1之间。

网络架构

实验中使用的卷积网络正是Mnih等人（2015）提出的网络，我们在此仅为了完整性提供细节。网络的输入是一个84x84x4的张量，包含最后四帧的缩放和灰度版本。第一层卷积层使用32个大小为8（步幅4）的滤波器对输入进行卷积，第二层有64个大小为4（步幅2）的滤波器，最后的卷积层有64个大小为3（步幅1）的滤波器。随后是一个包含512个单元的全连接隐藏层。所有这些层之间都由整流线性单元（ReLU）分隔。最后，一个全连接的线性层将输出投影到网络的输出，即Q值。用于训练网络的优化方法是RMSProp（动量参数为0.95）。

超参数

在所有实验中，折扣率设置为$\gamma = 0.99$，学习率设置为$\alpha = 0.00025$。目标网络更新之间的步数为$\tau = 10,000$。训练进行了5000万步（即2亿帧）。每100万步对智能体进行一次评估，并在这些评估中保留最佳策略作为学习过程的输出。经验回放记忆的大小为100万个元组。每4步从记忆中采样一次，使用大小为32的小批量更新网络。使用的简单探索策略是ϵ-贪婪策略，ϵ在100万步内从1线性减少到0.1。

Atari 2600 领域的补充结果

下表提供了我们在Atari领域实验的进一步详细结果。

Game	Random	Human	DQN	Double DQN
Alien	227.80	6875.40	3069.33	2907.30
Amidar	5.80	1675.80	739.50	702.10
Assault	222.40	1496.40	3358.63	5022.90
Asterix	210.00	8503.30	6011.67	15150.00
Asteroids	719.10	13156.70	1629.33	930.60
Atlantis	12850.00	29028.10	85950.00	64758.00
Bank Heist	14.20	734.40	429.67	728.30
Battle Zone	2360.00	37800.00	26300.00	25730.00
Beam Rider	363.90	5774.70	6845.93	7654.00
Bowling	23.10	154.80	42.40	70.50
Boxing	0.10	4.30	71.83	81.70
Breakout	1.70	31.80	401.20	375.00
Centipede	2090.90	11963.20	8309.40	4139.40
Chopper Command	811.00	9881.80	6686.67	4653.00
Crazy Climber	10780.50	35410.50	114103.33	101874.00
Demon Attack	152.10	3401.30	9711.17	9711.90
Double Dunk	-18.60	-15.50	-18.07	-6.30
Enduro	0.00	309.60	301.77	319.50
Fishing Derby	-91.70	5.50	-0.80	20.30
Freeway	0.00	29.60	30.30	31.80
Frostbite	65.20	4334.70	328.33	241.50
Gopher	257.60	2321.00	8520.00	8215.40
Gravitar	173.00	2672.00	306.67	170.50
H.E.R.O.	1027.00	25762.50	19950.33	20357.00
Ice Hockey	-11.20	0.90	-1.60	-2.40
James Bond	29.00	406.70	576.67	438.00
Kangaroo	52.00	3035.00	6740.00	13651.00
Krull	1598.00	2394.60	3804.67	4396.70
Kung-Fu Master	258.50	22736.20	23270.00	29486.00
Montezuma's Revenge	0.00	4366.70	0.00	0.00
Ms. Pacman	307.30	15693.40	2311.00	3210.00
Name This Game	2292.30	4076.20	7256.67	6997.10
Pong	-20.70	9.30	18.90	21.00
Private Eye	24.90	69571.30	1787.57	670.10
Q*Bert	163.90	13455.00	10595.83	14875.00
River Raid	1338.50	13513.30	8315.67	12015.30
Road Runner	11.50	7845.00	18256.67	48377.00
Robotank	2.20	11.90	51.57	46.70
Seaquest	68.40	20181.80	5286.00	7995.00
Space Invaders	148.00	1652.30	1975.50	3154.60
Star Gunner	664.00	10250.00	57996.67	65188.00
Tennis	-23.80	-8.90	-2.47	1.70
Time Pilot	3568.00	5925.00	5946.67	7964.00
Tutankham	11.40	167.60	186.70	190.60
Up and Down	533.40	9082.00	8456.33	16769.90
Venture	0.00	1187.50	380.00	93.00
Video Pinball	16256.90	17297.60	42684.07	70009.00
Wizard of Wor	563.50	4756.50	3393.33	5204.00
Zaxxon	32.50	9173.30	4976.67	10182.00

表3：无操作评估条件下的原始分数（5分钟模拟器时间）。DQN由Mnih等人（2015）给出。

Game	DQN	Double DQN
Alien	42.75 %	40.31 %
Amidar	43.93 %	41.69 %
Assault	246.17 %	376.81 %
Asterix	69.96 %	180.15 %
Asteroids	7.32 %	1.70 %
Atlantis	451.85 %	320.85 %
Bank Heist	57.69 %	99.15 %
Battle Zone	67.55 %	65.94 %
Beam Rider	119.80 %	134.73 %
Bowling	14.65 %	35.99 %
Boxing	1707.86 %	1942.86 %
Breakout	1327.24 %	1240.20 %
Centipede	62.99 %	20.75 %
Chopper Command	64.78 %	42.36 %
Crazy Climber	419.50 %	369.85 %
Demon Attack	294.20 %	294.22 %
Double Dunk	17.10 %	396.77 %
Enduro	97.47 %	103.20 %
Fishing Derby	93.52 %	115.23 %
Freeway	102.36 %	107.43 %
Frostbite	6.16 %	4.13 %
Gopher	400.43 %	385.66 %
Gravitar	5.35 %	-0.10 %
H.E.R.O.	76.50 %	78.15 %
Ice Hockey	79.34 %	72.73 %
James Bond	145.00 %	108.29 %
Kangaroo	224.20 %	455.88 %
Krull	277.01 %	351.33 %
Kung-Fu Master	102.37 %	130.03 %
Montezuma's Revenge	0.00 %	0.00 %
Ms. Pacman	13.02 %	18.87 %
Name This Game	278.29 %	263.74 %
Pong	132.00 %	139.00 %
Private Eye	2.53 %	0.93 %
Q*Bert	78.49 %	110.68 %
River Raid	57.31 %	87.70 %
Road Runner	232.91 %	617.42 %
Robotank	508.97 %	458.76 %
Seaquest	25.94 %	39.41 %
Space Invaders	121.49 %	199.87 %
Star Gunner	598.09 %	673.11 %
Tennis	143.15 %	171.14 %
Time Pilot	100.92 %	186.51 %
Tutankham	112.23 %	114.72 %
Up and Down	92.68 %	189.93 %
Venture	32.00 %	7.83 %
Video Pinball	2539.36 %	5164.99 %
Wizard of Wor	67.49 %	110.67 %
Zaxxon	54.09 %	111.04 %

表4：无操作评估条件下的归一化结果（5分钟模拟器时间）。

Game	Random	Human	DQN	Double DQN	Double DQN (tuned)
Alien	128.30	6371.30	570.2	621.6	1033.4
Amidar	11.80	1540.40	133.4	188.2	169.1
Assault	166.90	628.90	3332.3	2774.3	6060.8
Asterix	164.50	7536.00	124.5	5285.0	16837.0
Asteroids	871.30	36517.30	697.1	1219.0	1193.2
Atlantis	13463.00	26575.00	76108.0	260556.0	319688.0
Bank Heist	21.70	644.50	176.3	469.8	886.0
Battle Zone	3560.00	33030.00	17560.0	25240.0	24740.0
Beam Rider	254.60	14961.00	8672.4	9107.9	17417.2
Berzerk	196.10	2237.50		635.8	1011.1
Bowling	35.20	146.50	41.2	62.3	69.6
Boxing	-1.50	9.60	25.8	52.1	73.5
Breakout	1.60	27.90	303.9	338.7	368.9
Centipede	1925.50	10321.90	3773.1	5166.6	3853.5
Chopper Command	644.00	8930.00	3046.0	2483.0	3495.0
Crazy Climber	9337.00	32667.00	50992.0	94315.0	113782.0
Defender	1965.50	14296.00		8531.0	27510.0
Demon Attack	208.30	3442.80	12835.2	13943.5	69803.4
Double Dunk	-16.00	-14.40	-21.6	-6.4	-0.3
Enduro	-81.80	740.20	475.6	475.9	1216.6
Fishing Derby	-77.10	5.10	-2.3	-3.4	3.2
Freeway	0.10	25.60	25.8	26.3	28.8
Frostbite	66.40	4202.80	157.4	258.3	1448.1
Gopher	250.00	2311.00	2731.8	8742.8	15253.0
Gravitar	245.50	3116.00	216.5	170.0	200.5
H.E.R.O.	1580.30	25839.40	12952.5	15341.4	14892.5
Ice Hockey	-9.70	0.50	-3.8	-3.6	-2.5
James Bond	33.50	368.50	348.5	416.0	573.0
Kangaroo	100.00	2739.00	2696.0	6138.0	11204.0
Krull	1151.90	2109.10	3864.0	6130.4	6796.1
Kung-Fu Master	304.00	20786.80	11875.0	22771.0	30207.0
Montezuma's Revenge	25.00	4182.00	50.0	30.0	42.0
Ms. Pacman	197.80	15375.00	763.5	1401.8	1241.3
Name This Game	1747.80	6796.00	5439.9	7871.5	8960.3
Phoenix	1134.40	6686.20		10364.0	12366.5
Pit Fall	-348.80	5998.90		-432.9	-186.7
Pong	-18.00	15.50	16.2	17.7	19.1
Private Eye	662.80	64169.10	298.2	346.3	-575.5
Q*Bert	183.00	12085.00	4589.8	10713.3	11020.8
River Raid	588.30	14382.20	4065.3	6579.0	10838.4
Road Runner	200.00	6878.00	9264.0	43884.0	43156.0
Robotank	2.40	8.90	58.5	52.0	59.1
Seaquest	215.50	40425.80	2793.9	4199.4	14498.0
Skiing	-15287.40	-3686.60		-29404.3	-11490.4
Solaris	2047.20	11032.60		2166.8	810.0
Space Invaders	182.60	1464.90	1449.7	1495.7	2628.7
Star Gunner	697.00	9528.00	34081.0	53052.0	58365.0
Surround	-9.70	5.40		-7.6	1.9
Tennis	-21.40	-6.70	-2.3	11.0	-7.8
Time Pilot	3273.00	5650.00	5640.0	5375.0	6608.0
Tutankham	12.70	138.30	32.4	63.6	92.2
Up and Down	707.20	9896.10	3311.3	4721.1	19086.9
Venture	18.00	1039.00	54.0	75.0	21.0
Video Pinball	20452.0	15641.10	20228.1	148883.6	367823.7
Wizard of Wor	804.00	4556.00	246.0	155.0	6201.0
Yars Revenge	1476.90	47135.20		5439.5	6270.6
Zaxxon	475.00	8443.00	831.0	7874.0	8593.0

表5：人类起始条件下的原始分数（30分钟模拟器时间）。DQN由Nair等人（2015）给出。

Game	DQN	Double DQN	Double DQN (tuned)
Alien	7.08%	7.90%	14.50%
Amidar	7.95%	11.54%	10.29%
Assault	685.15%	564.37%	1275.74%
Asterix	-0.54%	69.46%	226.18%
Asteroids	-0.49%	0.98%	0.90%
Atlantis	477.77%	1884.48%	2335.46%
Bank Heist	24.82%	71.95%	138.78%
Battle Zone	47.51%	73.57%	71.87%
Beam Rider	57.24%	60.20%	116.70%
Berzerk		21.54%	39.92%
Bowling	5.39%	24.35%	30.91%
Boxing	245.95%	482.88%	675.68%
Breakout	1149.43%	1281.75%	1396.58%
Centipede	22.00%	38.60%	22.96%
Chopper Command	28.99%	22.19%	34.41%
Crazy Climber	178.55%	364.24%	447.69%
Defender		53.25%	207.17%
Demon Attack	390.38%	424.65%	2151.65%
Double Dunk	-350.00%	600.00%	981.25%
Enduro	67.81%	67.85%	157.96%
Fishing Derby	91.00%	89.66%	97.69%
Freeway	100.78%	102.75%	112.55%
Frostbite	2.20%	4.64%	33.40%
Gopher	120.42%	412.07%	727.95%
Gravitar	-1.01%	-2.63%	-1.57%
H.E.R.O.	46.88%	56.73%	54.88%
Ice Hockey	57.84%	59.80%	70.59%
James Bond	94.03%	114.18%	161.04%
Kangaroo	98.37%	228.80%	420.77%
Krull	283.34%	520.11%	589.66%
Kung-Fu Master	56.49%	109.69%	145.99%
Montezuma’s Revenge	0.60%	0.12%	0.41%
Ms. Pacman	3.73%	7.93%	6.88%
Name This Game	73.14%	121.30%	142.87%
Phoenix		166.25%	202.31%
Pit Fall		-1.32%	2.55%
Pong	102.09%	106.57%	110.75%
Private Eye	-0.57%	-0.50%	-1.95%
Q*Bert	37.03%	88.48%	91.06%
River Raid	25.21%	43.43%	74.31%
Road Runner	135.73%	654.15%	643.25%
Robotank	863.08%	763.08%	872.31%
Seaquest	6.41%	9.91%	35.52%
Skiing		-121.69%	32.73%
Solaris		1.33%	-13.77%
Space Invaders	98.81%	102.40%	190.76%
Star Gunner	378.03%	592.85%	653.02%
Surround		13.91%	76.82%
Tennis	129.93%	220.41%	92.52%
Time Pilot	99.58%	88.43%	140.30%
Tutankham	15.68%	40.53%	63.30%
Up and Down	28.34%	43.68%	200.02%
Venture	3.53%	5.58%	0.29%
Video Pinball	-4.65%	2669.60%	7220.51%
Wizard of Wor	-14.87%	-17.30%	143.84%
Yars Revenge		8.68%	10.50%
Zaxxon	4.47%	92.86%	101.88%

表6：人类起始条件的归一化分数（30分钟模拟器时间）。