超越 ReLU：Chebyshev-DQN 用于增强深度 Q 网络

Saman Yazdannik
Researcher
K. N. Toosi University of Technology, Tehran, Iran
Machine Learning Engineer
AFLAK Science and Technology, Tehran, Iran
s.yazdannik@email.kntu.ac.ir

Morteza Tayefi
Assistant Professor
Intelligent Control Systems Institute
K. N. Toosi University of Technology, Tehran, Iran
tayefi@kntu.ac.ir

Shamim Sanisales
Independent Researcher

2025 年 8 月 24 日

Abstract

深度 Q 网络（DQN）的性能在很大程度上取决于其底层神经网络准确逼近动作价值函数的能力。标准的功能近似器，如多层感知器，在有效表示许多强化学习问题中固有的复杂价值景观方面可能会遇到困难。本文介绍了一种新颖的架构——切比雪夫-DQN（Ch-DQN），它将切比雪夫多项式基整合到 DQN 框架中以创建更有效的特征表示。通过利用切比雪夫多项式的强大功能逼近属性，我们假设 Ch-DQN 可以更高效地学习并实现更高的性能。我们在 CartPole-v1 基准上评估了我们的模型，并将其与具有相当参数数量的标准 DQN 进行了比较。结果表明，使用适度的多项式次数（N=4）时，Ch-DQN 在渐近性能方面显著优于基线，提高了约 39%。然而，我们还发现多项式的次数是一个关键的超参数，因为高次数（N=8）可能对学习产生不利影响。这项工作验证了使用正交多项式基在深度强化学习中的潜在价值，同时也强调了模型复杂性所涉及的权衡。

1 介绍

深度强化学习（DRL）在人工智能领域标志着一个重大突破，使代理能够在一系列复杂的顺序决策任务中达到超人的表现。里程碑式的成功案例包括从原始像素数据掌握 Atari 游戏 [1, 5] 和在围棋游戏中击败世界冠军 [2]，以及在机器人技术、自动驾驶和资源管理方面取得显著进展 [6]。这一成功的基石是深度 Q 网络（DQN）算法，该算法首次证明了一个深度神经网络可以使用统一的架构来解决一系列具有挑战性的控制问题 [1]。中心机制涉及使用深度神经网络作为强大的函数逼近器来估计动作值（Q 值）函数。这种近似的准确性至关重要，因为它直接决定了代理所学策略的质量。

在标准的 DQN 框架中，函数逼近器通常是多层感知机（MLP）或卷积神经网络（CNN）。虽然这些架构已被证明非常强大，但它们的应用并非没有挑战。众所周知，非线性函数逼近器与异策略学习和自举相结合，通常被称为“致命三重奏”，会导致显著的训练不稳定性和发散问题 [3, 7]。因此，DQN 需要专门的技术，如经验回放和固定的靶网络来缓解这些问题 [1]。此外，现代的 DRL 算法常常因为样本效率低下而受到批评，通常需要数百万次环境交互才能学习到有效的策略，这与生物学习相比是一个显著的缺点 [8]。这引发了一个关

键问题：传统的神经网络架构是否是表示强化学习任务中价值函数的最有效工具?
本文认为，我们可以基于经典逼近理论的基本原则设计出更有效的 DRL 代理。我们不是第一个考虑替代功能基的人；先前的研究已经探索了使用线性模型与傅里叶基和径向基函数 [9]。然而，我们将注意力转向切比雪夫多项式，这一类正交多项式因其强大的理论基础而闻名。切比雪夫多项式的特性之一是其“极小极大”性质，它表明在 L-无穷范数下，它们提供了对连续函数的最佳多项式逼近，最小化了可能的最大误差 [4, 10]。这表明，基于切比雪夫多项式的功能基可能会更有效地表示复杂的价值函数。
在此基础上，我们引入了 Chebyshev-DQN（Ch-DQN），这是一种新颖的架构，在 DQN 框架内整合了一个切比雪夫神经网络作为主要的功能近似器。与依赖 ReLU 等标准激活函数的不同，Ch-DQN 使用切比雪夫多项式的基来转换输入状态为特征表示。据我们所知，这是首次探索将这种集成应用于深度强化学习的工作。我们的中心假设是，切比雪夫基的优越逼近特性将使 Ch-DQN 能够学习更精确和稳定的 Q 函数表示。我们预期这将在样本效率方面带来显著性能改进。为了验证这一假设，我们在经典的连续控制基准上进行了一系列实验。本工作旨在弥合逼近理论与现代深度强化学习之间的差距，提出了一条新的路径以开发更加稳健、高效且高性能的 DRL 代理。

2 背景及相关工作

2.1 深度 Q 网络（DQN）

Q-学习是一种无模型的强化学习算法，旨在学习最优的动作价值函数，$Q^*(s,a)$ [12]。该函数定义为从给定的状态-动作对 (s,a) 所能获得的最大期望累计奖励。最优的 Q 函数遵循贝尔曼最优性方程 [13]:

$$Q^*(s,a) = \mathbb{E}\left[r + \gamma \max_{a'} Q^*(s',a') \mid s,a\right] \tag{1}$$

其中，r 是即时奖励，γ 是折扣因子，而 s' 是下一状态。对于具有小规模离散状态空间的问题，Q 值可以存储在一个查找表（即 Q-表）中。然而，在大规模或连续状态空间问题中，这种方法变得不可行。

DQN 算法由 Mnih 等人引入，[5, 1] 解决了可扩展性问题，通过用一个深度神经网络 $Q(s,a;\theta)$ 替换 Q 表，该网络由权重 θ 参数化。为了确保稳定的训练，DQN 引入了两个关键创新：

- **经验回放：** 代理将其经验 (s,a,r,s') 存储在回放缓冲区中。在训练过程中，从该缓冲区随机抽取经验的小批量来更新网络权重。这打破了连续样本之间的时间相关性，并平滑了训练分布。
- **目标网络：** DQN 使用第二个独立的“目标网络”，$Q(s,a;\theta^-)$，来生成贝尔曼更新的目标值。这个目标网络的权重在一定步数内保持固定，并且只定期用主“策略网络”的权重进行更新。这可以防止使用快速变化的网络同时估计当前 Q 值和目标值时所产生的不稳定。

该网络通过最小化目标 Q 值与策略网络预测的值之间的均方误差（MSE）损失来进行训练。

2.2 函数逼近在强化学习中的应用及其相关问题

函数逼近器的使用对于将强化学习应用于实际问题至关重要。目标是找到一个参数化函数，可以从有限的训练样本集中推广到整个状态-动作空间 [3]。这一研究领域早于深度学习，早期工作主要集中在线性函数逼近器上，其中 Q 函数表示为一组基函数或特征 [9] 的线性组合。

尽管计算效率高，线性方法可能缺乏表达复杂非线性价值函数的能力。深度学习的出现为非线性函数逼近提供了一个强大的工具包，神经网络成为常见的选择。然而，非线性逼近器、离策略学习和引导（“致命三联体”）的结合可能导致不稳定性和发散 [7, 3]，这促使了 DQN 架构上的创新。我们的工作位于这一研究领域内，特别关注于改进用于逼近的基础函数。

2.3 切比雪夫多项式和神经网络

第一类切比雪夫多项式，记为 $T_n(x)$，是一系列正交多项式，具有使其非常适合函数逼近的多个性质 [10]。正交性有助于避免使用标准单项式基（例如 $1, x, x^2, \dots$）时可能出现的数值不稳定性和多重共线性问题。此外，它们在最小最大意义上是最佳的，在最大范数下提供最优的多项式逼近 [4]。

最近，这些特性已经在神经网络的背景下得到了应用。例如，**切比雪夫特征神经网络（CFNN）**的概念被提出作为一种强大的监督学习任务架构 [14]。CFNN 通常由一个初始隐藏层组成，该层使用切比雪夫函数的基来变换输入状态 s。然后将这个切比雪夫特征层的输出馈送到一个或多个标准全连接层中。这种结构利用了切比雪夫基的优越逼近能力，同时保留了深度网络的强大表示学习能力。这类网络在函数逼近任务中实现高精度的能力使其成为增强 DQN 代理核心的有吸引力的选择。

3 切比雪夫-DQN 架构

在本节中，我们详细介绍了所提出的 Chebyshev-DQN（Ch-DQN）的架构和机制。Ch-DQN 的基本创新之处在于用一个显式利用 Chebyshev 多项式基进行特征表示的网络替换了标准的多层感知器（MLP）函数逼近器。这种设计选择旨在为强化学习代理提供更强大和高效的函数空间来近似 Q 值函数。

3.1 概念框架

Ch-DQN 背后的核心概念是将特征提取和价值估计这两个任务分开。在一个基于标准 MLP 的 DQN 中，这两个任务隐含地纠缠在网络的隐藏层中。相比之下，Ch-DQN 首先将输入状态投影到由切比雪夫多项式基定义的高维特征空间中。这种投影作为一种稳健且以数学为基础的特征工程形式。随后的神经网络则学习将这些经过工程处理的特征映射到相应的 Q 值。

这种方法基于假设，Chebyshev 基为表示价值函数提供了一组比标准 MLP 隐式学习的特征更合适的特征集。通过提供一组丰富的正交基函数，我们减轻了网络的负担，使其能够专注于学习这些特征的正确线性组合，而不是必须从头开始发现价值景观的基础功能形式。

3.2 模型架构

Ch-DQN 架构由三个主要组件组成：输入归一化步骤、Chebyshev 特征层和全连接输出网络。架构的可视化表示如图 1 所示。

1. **输入规范化：** 切比雪夫多项式在区间 $[-1, 1]$ 上形式定义。因此，我们模型的一个先决条件是输入状态向量 $\boldsymbol{s}$ 被标准化为落在这个范围内。对于具有已知状态空间边界的环境（例如 MountainCar 中的位置和速度），可以通过简单的线性缩放来实现这一点。
2. **切比雪夫特征层：** 这是我们架构的新组件。对于给定的归一化输入状态 $\boldsymbol{s} \in \mathbb{R}^D$，这一层计算每个状态向量分量 s_i 的前 N 个第一类 Chebyshev 多项式 $T_n(x)$。多项式由递推关系定义：

$$
\begin{aligned}
T_0(x) &= 1 \\
T_1(x) &= x \\
T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x) \quad \text{for } n \geq 1.
\end{aligned}
$$

这一层的输出是一个特征向量，$\boldsymbol{\Phi}(\boldsymbol{s})$，它是每个状态维度上的多项式求值的连接：

$$
\boldsymbol{\Phi}(\boldsymbol{s}) = [T_0(s_1), \dots, T_N(s_1), T_0(s_2), \dots, T_N(s_2), \dots, T_0(s_D), \dots, T_N(s_D)] \tag{2}
$$

多项式基的次数 N 是模型的关键超参数，控制着特征表示的丰富性。得到的特征向量 $\boldsymbol{\Phi}(\boldsymbol{s})$ 具有 $D \times (N+1)$ 维。

3. **全连接网络：**特征向量 $\boldsymbol{\Phi}(\boldsymbol{s})$ 然后被传入一个标准的前馈神经网络。该网络可以是一个简单的线性层，直接将切比雪夫特征映射到 Q 值上，或者它也可以是一个小型的多层感知器（例如，带有一个或两个具有 ReLU 激活函数的隐藏层）以捕捉特征之间的非线性关系。
4. **输出层：**最后一层是一个具有 A 个输出节点的线性层，其中 A 是代理可用的离散动作的数量。每个输出节点对应一个估计的 Q 值，$Q(s, a)$。

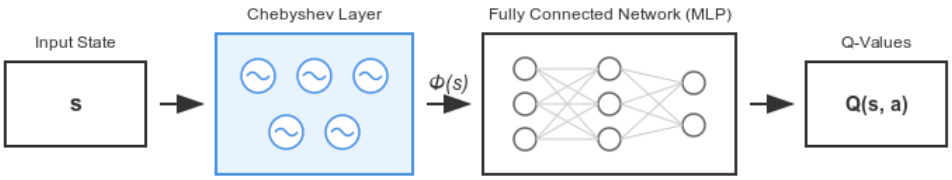

图 1: 提出的 Chebyshev-DQN（Ch-DQN）架构。输入状态 $\boldsymbol{s}$ 首先被归一化，然后通过 Chebyshev 层转换为特征向量 $\boldsymbol{\Phi}(\boldsymbol{s})$。该特征向量随后由全连接网络处理以生成每个动作的最终 Q 值。

3.3 数学公式表述

形式上，由 Chebyshev-DQN 近似的 Q 值函数表示为：

$$Q(\boldsymbol{s}, a; \boldsymbol{\theta}) = f_a(\boldsymbol{\Phi}(\boldsymbol{s}); \boldsymbol{\theta}) \tag{3}$$

其中：

- $\boldsymbol{s}$ 是归一化的输入状态。
- $\boldsymbol{\Phi}(\boldsymbol{s})$ 是由切比雪夫层生成的特征向量。
- f 是由权重 $\boldsymbol{\theta}$ 参数化的全连接网络。
- a 表示与某个动作对应的网络的特定输出节点。

3.4 训练算法

Ch-DQN 的训练过程紧密遵循原始 DQN 算法 [1]。架构变化完全包含在 Q 网络的结构中；学习规则和与环境的交互保持不变。我们仍然使用经验回放缓冲区和固定的目标网络以确保训练稳定性。

网络参数 $\boldsymbol{\theta}$ 通过最小化目标 Q 值与预测 Q 值之间的均方误差（MSE）进行优化。目标 y_j 是通过使用由 $\boldsymbol{\theta}^-$ 参数化的靶网络计算得出的，该值来自从回放缓冲区 $\mathcal{D}$ 中采样的经验元组 (s_j, a_j, r_j, s_{j+1})：

$$y_j = r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \boldsymbol{\theta}^-) \tag{4}$$

损失函数是对采样转换的期望：

$$L(\boldsymbol{\theta}) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\left(y - Q(s, a; \boldsymbol{\theta}) \right)^2 \right] \tag{5}$$

靶网络权重 $\boldsymbol{\theta}^-$ 定期使用策略网络权重 $\boldsymbol{\theta}$ 进行更新。完整的算法概述于算法 1 中。

Algorithm 1 带有经验回放的 Chebyshev-DQN

初始化回放内存 $\mathcal{D}$ 至容量 M
使用随机权重初始化策略网络 Ch-DQN$\boldsymbol{\theta}$
使用权重 $\boldsymbol{\theta}^- = \boldsymbol{\theta}$ 初始化目标网络 Ch-DQN
for episode = 1 to E **do**
    初始化状态 s_1
    for $t = 1$ to T **do**
        以概率 ϵ 随机选择一个动作 a_t
        否则选择 $a_t = \arg\max_a Q(s_t, a; \boldsymbol{\theta})$ ▷ Use preprocessed features $\Phi(s_t)$ if needed
        执行动作 a_t 并观察奖励 r_t 和下一状态 s_{t+1}
        将过渡 (s_t, a_t, r_t, s_{t+1}) 存储在 $\mathcal{D}$ 中
        从 $\mathcal{D}$ 随机抽样一个小型批量的过渡 (s_j, a_j, r_j, s_{j+1})
        if episode terminates at step $j + 1$ **then**
            $y_j \leftarrow r_j$
        else
            $y_j \leftarrow r_j + \gamma \max_{a'} Q(s_{j+1}, a'; \boldsymbol{\theta}^-)$
        end if
        在 $(y_j - Q(s_j, a_j; \boldsymbol{\theta}))^2$ 上执行梯度下降步骤
        每 C 步重置 $\boldsymbol{\theta}^- \leftarrow \boldsymbol{\theta}$
    end for
end for

4 实验

为了全面评估我们提出的 Chebyshev-DQN（Ch-DQN），我们在三个经典控制基准测试中进行了实验，这些基准测试的复杂度逐渐增加。目标是评估该架构相对于标准 DQN 基线的渐近性能和样本效率，并分析 Chebyshev 多项式次数 N 的最优选择如何随任务难度变化。

4.1 环境

我们从 Gymnasium 库中选择三个基准环境 [15] 来测试我们的模型在各种控制挑战中的表现：

- **CartPole-v1:** 一个复杂度低的任务，具有密集的奖励信号和 4 维状态空间。它作为稳定性和基本控制的基准。
- **MountainCar-v0: 山地车-v0:** 一项中等复杂度的任务，以其稀疏的奖励函数和二维状态空间而著称。解决它需要有效的探索。
- **摆杆机器人-v1:** 一个具有混沌动力学、稀疏奖励和 6 维状态空间的高复杂度任务。它代表了一个显著更具挑战性的函数逼近和控制问题。

4.2 模型和基线

我们比较了我们 Ch-DQN 的三种变体与使用多层感知器 (MLP) 的标准 DQN。为了确保公平的比较，我们对更复杂的 Acrobot 任务中所有模型的网络大小进行了增加。

- **Ch-DQN（我们的）:** 架构如第 3 节所述，多项式次数为 $N \in \{4, 6, 8\}$。随后的 MLP 有两个隐藏层，CartPole/MountainCar 每个隐藏层有 64 个神经元，Acrobot 每个隐藏层有 128 个神经元。

- **基线 DQN（多层感知器）:** 具有两个隐藏层的标准前馈网络，每层有 64 个神经元（CartPole/MountainCar）或 128 个神经元（Acrobot），均使用 ReLU 激活函数。

所有模型均使用 PyTorch 实现。我们为学习率和探索策略使用了特定于环境的超参数，以确保每个智能体都能很好地适应其各自的任务。

4.3 实现细节

所有模型均使用 PyTorch 实现。[16]。

Acrobot-v1，以其高维状态空间和稀疏奖励，提出了一个显著的探索挑战。为了创建一个可处理的基准以比较函数逼近的质量，我们对所有该环境中的代理进行了两项相同的修改。首先，我们将所有模型的网络规模增加到两个各含 128 个神经元的隐藏层。其次，我们引入了一个基于潜力的奖励整形项，在每一步中根据摆尖的高度添加一个小额奖励 [11]。这个密集的奖励信号引导代理，并允许我们将比较的重点放在最终策略的质量上，而不是困难的探索问题上。

对于每个实验，我们用不同的随机种子训练了 3 个独立的运行。核心超参数详见表 1。

表 1: 核心超参数用于训练等等

超参数	值
Optimizer	Adam
Discount Factor (γ)	0.99
Replay Buffer Size	50,000
Batch Size	64
Target Network Update Frequency (C)	500 steps
Chebyshev Polynomial Degree (N)	{4, 6, 8}

4.4 评估指标

评估的主要指标是每轮的累积奖励。我们绘制学习曲线以显示随时间的变化表现，比较最终平均奖励来评估渐近性能，并测量达到性能阈值所需的轮次数量来评价样本效率。

5 结果

在本节中，我们展示了实验的实证结果，比较了 Ch-DQN 架构与标准 DQN 基线在三种复杂性递增的环境中的性能。对于每个环境，我们都报告了经过 3 次独立运行平均后的最终性能。

5.1 卡特波尔-v1 的结果

在低复杂度的 CartPole 任务中，具有较低多项式次数（N=4）的 Ch-DQN 表现最佳，最终得分为 347.9，显著优于基准的 250.5。然而，将多项式次数增加到 N=8 则证明是有害的，得分仅为 144.0。这表明对于更简单的价值函数而言，过于复杂的特征基可能会阻碍学习，类似于过拟合。完整结果如图 2 所示。

5.2 MountainCar-v0 的结果

Ch-DQN 架构在稀疏奖励的 MountainCar 环境中的优势变得更加明显。如图 3 所示，所有 Ch-DQN 变体都显著优于基线。标准 DQN 最终得分为-132.3 $\pm$ 18.5，而所有 Ch-DQN 模型收敛到一个接近最优且高度稳定的得

分约为-112。最显著的结果是样本效率的提高；Ch-DQN 模型始终在不到 600 个 episode 内解决了任务，这比需要超过 1600 个 episode 的基线有了近 3 倍的改进。

5.3 Acrobot-v1 的结果

高复杂度的 Acrobot 环境作为最具挑战性的测试。与一个得分-86.0 ± 1.0 的强大且调整得很好的基线相比，具有最高多项式次数（N=8）的 Ch-DQN 找到了更优策略，最终平均奖励达到-85.5 ± 1.5。虽然绝对性能提升微乎其微，但 Ch-DQN 模型展示了更为一致的样本效率，可靠地比变化较大的基线更快解决任务（图 4）。较低次数的 Ch-DQN 模型（N=4, N=6）未能达到强大基线的表现，强调了在更复杂的任务中获得优势需要一个更加复杂的功能基础。

5.4 网络复杂度

为了确认性能提升不仅仅是因为模型容量增加的结果，我们在表 2 中报告了所有架构的参数数量。虽然 Ch-DQN 模型稍大一些，但尺寸的适度增长不足以解释观察到的重大性能差异，特别是在 MountainCar 上。这表明切比雪夫基底的结构优势是改进的主要驱动力。

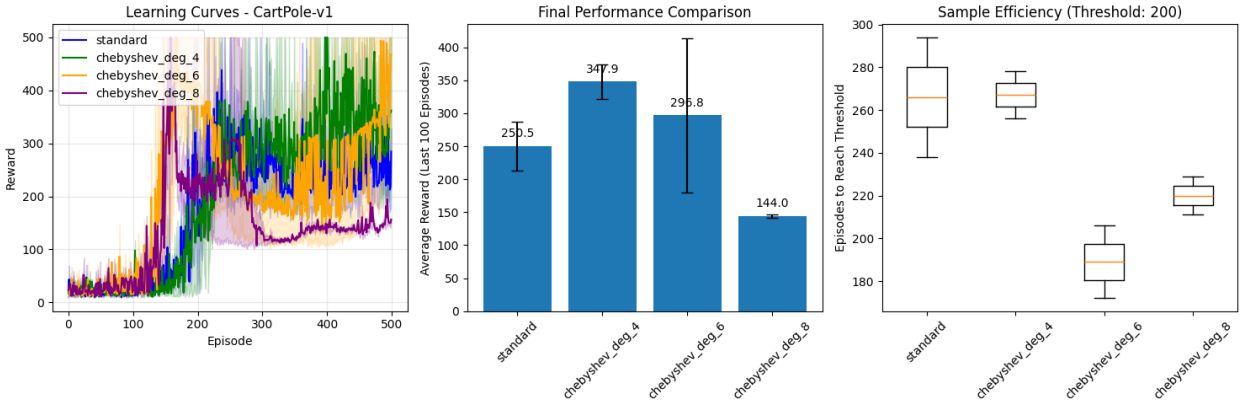

图 2: CartPole-v1 的实验结果。一个低阶的 Ch-DQN（N=4）表现出色，而高阶（N=8）表现不佳。

6 讨论

我们的实证结果表明，Ch-DQN 架构在复杂的控制任务中具有明显且一致的优势。我们可以将这一成功归因于切比雪夫多项式的数学性质与深度 Q 学习核心挑战之间的相互作用，从而为我们的发现提供理论解释。

6.1 固有逼近误差和最小最大性质

任何代理性能的基本限制是固有的近似误差 $\mathbf{r}$，ϵ_{approx}，代表在选定的函数类 $\mathcal{F}$ [7] 中对真实 Q-函数 Q^* 的最佳拟合。此误差定义为：

$$\epsilon_{\text{approx}} = \min_{f \in \mathcal{F}} \|Q^* - f\|_\mu \tag{6}$$

切比雪夫多项式的极小极大性质保证了截断的切比雪夫级数在 L_∞ 范数下是一个近最优的多项式逼近器 [4]。通过将状态投影到切比雪夫基上，Ch-DQN 向其网络提供了一组特征集，这组特征集可以在给定数量的特征（多项式次数 N）下以更低的内在近似误差表示 Q^*，相比于结构不那么严谨的基础。

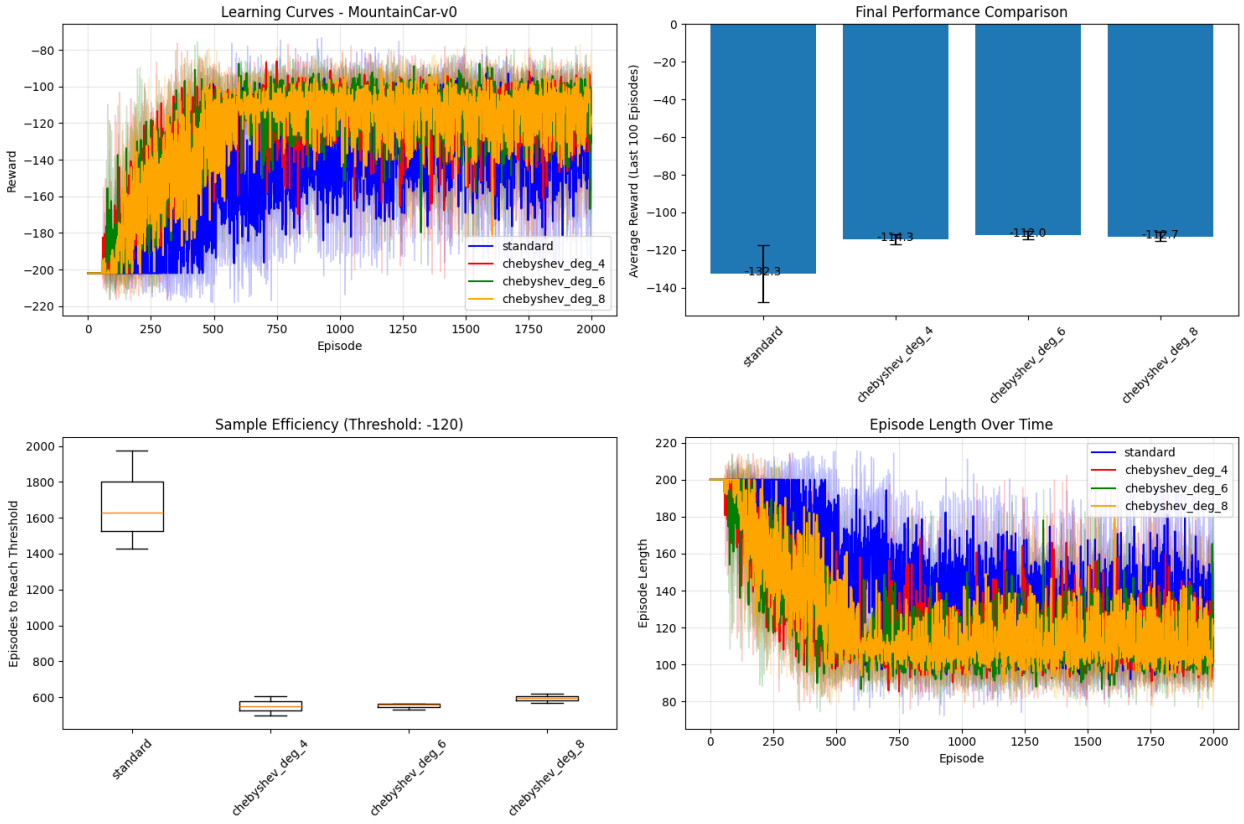

图 3: MountainCar-v0 上的实验结果。所有 Ch-DQN 模型都表现出优越的最终性能，并且在样本效率方面提高了约 3 倍。

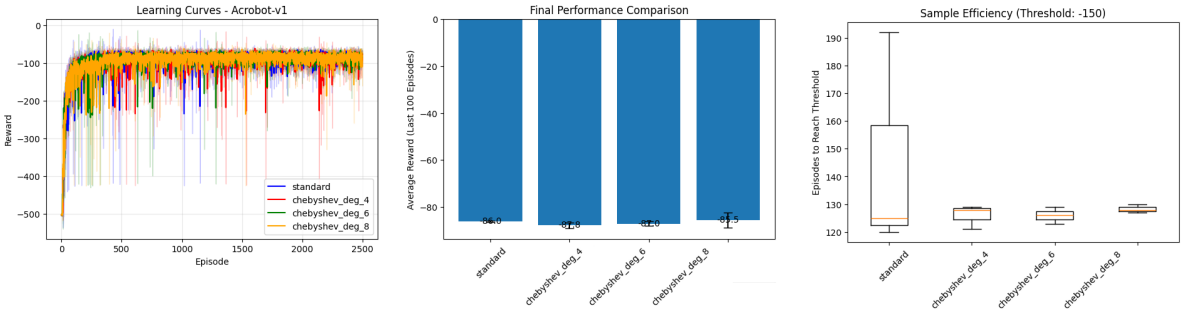

图 4: Acrobot-v1 上的实验结果。高阶 Ch-DQN（N=8）相对于强大的基线实现了最佳最终性能。

表 2: 所有环境中的可训练参数对比。

环境	模型	总参数数量
CartPole-v1	Standard DQN	4,610
(State Dim=4)	Ch-DQN (N=4)	5,634
	Ch-DQN (N=6)	6,146
	Ch-DQN (N=8)	6,658
MountainCar-v0	Standard DQN	4,483
(State Dim=2)	Ch-DQN (N=4)	5,027
	Ch-DQN (N=6)	5,355
	Ch-DQN (N=8)	5,683
Acrobot-v1	Standard DQN	17,411
(State Dim=6)	Ch-DQN (N=4)	19,715
	Ch-DQN (N=6)	21,251
	Ch-DQN (N=8)	22,787

这一理论优势直接得到了我们结果的支持。在更复杂的 MountainCar 和 Acrobot 任务上，Ch-DQN 模型始终收敛到比标准 DQN 更为最优的最终策略。这表明基线的功能类具有较高的固有误差，无法像 Ch-DQN 的功能类那样准确地表示真正的 Q^*。

6.2 学习动态与正交性

DQN 中使用的半梯度更新的稳定性因其破坏性干扰而臭名昭著地脆弱。切比雪夫基底的正交性可以通过改善学习问题的条件来减轻这一问题。多项式 $T_n(x)$ 关于权重函数 $w(x) = (1-x^2)^{-1/2}$ 是正交的：

$$\int_{-1}^{1} T_n(x)T_m(x)\frac{dx}{\sqrt{1-x^2}} = \begin{cases} 0 & n \neq m \\ \pi & n = m = 0 \\ \pi/2 & n = m \neq 0 \end{cases} \tag{7}$$

虽然我们的完整模型是非线性的，但初始投影到切比雪夫基底提供了隐式正则化的一种形式。它将原始状态转换为去相关的特征空间，为后续的 MLP 提供更好的条件输入。这可以通过减轻构成“致命三角”的破坏性干扰来稳定训练。

T 其稳定效果在我们的 MountainCar 实验中最为明显。如图 3 所示，Ch-DQN 模型的最终性能（$\sigma \approx 3.0$）方差显著低于高度不稳定的基线（$\sigma \approx 18.5$），证实了正交特征投影导致了一个更可靠和稳定的学习过程。

6.3 表达能力与过拟合：一个谱偏置视角

我们的结果表明，对于多项式次数 N 存在一个明显的“最佳点”，这可以通过**光谱偏差** [17] 的视角来解释。次数 N 明确地设计了模型特征空间的频率内容，因为 $T_n(x) = \cos(n \arccos(x))$。

TD 目标是由代理自身生成的噪声、非平稳估计。一个高度表达性的函数类，富含高频分量（如高阶 Ch-DQN），可能会过度拟合这些目标中的高频噪声，从而导致误差放大的恶性循环。这为我们在观察中所发现的权衡提供了原理上的解释：

- **关于 CartPole（低复杂度）：** 价值函数相对简单（低频）。Ch-DQN（N=4）提供了足够的基础并且表现出色。Ch-DQN（N=8）不必要的高频基函数过度拟合了目标噪声，导致性能崩溃。

- **关于 Acrobot（高复杂度）:** 价值函数非常复杂。这里，低阶 Ch-DQN（N=4）缺乏足够的表达能力来超越强大的基线模型。由 Ch-DQN（N=8）提供的高频分量成为捕捉价值景观细微特征的必要条件，使其能够找到更优的最终策略。

这证实了多项式次数 N 必须足够高以捕捉真实价值函数的相关频率，但也不能过高以至于引入不稳定模式从而过度拟合学习过程中的噪声。

7 结论

在本文中，我们介绍了 Chebyshev-DQN（Ch-DQN），这是一种将 Chebyshev 多项式基整合到 DQN 框架中的新架构。我们的实验跨越了三个不同复杂度的环境，结果显示 Ch-DQN 始终能够与经过良好调优的 DQN 基准线匹敌或超越它。

优势在 MountainCar 等具有挑战性的任务上最为明显，其中 Ch-DQN 在样本效率方面实现了近 3 倍的提升，并收敛到了更加优化和稳定的最终策略。我们的理论分析将这一成功归因于切比雪夫基函数的性质，这种基函数减少了固有的逼近误差并改善了学习问题的条件性。此外，我们通过频谱偏差的角度为观察到的多项式阶数之间的权衡提供了原理性的解释：最优阶数 N 似乎与任务的价值函数复杂度相关。

本工作验证了正交多项式基作为深度强化学习强大工具的使用。未来的工作可以探索自适应方法来调整多项式的次数，或将 Ch-DQN 架构与更先进的探索策略相结合以解决纯稀疏奖励问题。

参考文献

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

[2] Silver, D., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*.

[3] Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

[4] Trefethen, L. N. (2013). *Approximation theory and approximation practice*. SIAM.

[5] Mnih, V., et al. (2013). Playing atari with deep reinforcement learning. *NIPS deep learning workshop*.

[6] Arulkumaran, K., et al. (2017). A brief survey of deep reinforcement learning. *IEEE Signal Processing Magazine*.

[7] Tsitsiklis, J. N., & Van Roy, B. (1997). An analysis of temporal-difference learning with function approximation. *IEEE transactions on automatic control*.

[8] Botvinick, M., et al. (2019). Reinforcement learning, fast and slow. *Trends in cognitive sciences*.

[9] Konidaris, G., et al. (2011). Value function approximation in reinforcement learning using the Fourier basis. *AAAI*.

[10] Mason, J. C., & Handscomb, D. C. (2002). *Chebyshev polynomials*. CRC press.

[11] Ng, A. Y., Harada, D., & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML* (Vol. 99, pp. 278-287).

[12] Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine learning*, *8*(3-4), 279-292.

[13] Bellman, R. (1957). *Dynamic Programming*. Princeton University Press.

[14] Lim, L. B. L., et al. (2022). Chebyshev feature neural network (CFNN): A new activation function for machine accuracy approximation. *Neurocomputing*, *470*, 401-416.

[15] Brockman, G., et al. (2016). OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

[16] Paszke, A., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32* (pp. 8024-8035). Curran Associates, Inc.

[17] Rahaman, N., et al. (2019). On the Spectral Bias of Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning* (pp. 5301-5310).