Continuous Control with Deep Reinforcement Learning

SL LV

2024 年 8 月 16 日

1 摘要

我们将深度Q学习的成功理念适用于连续动作领域。本文提出了一种基于确定性策略梯度的演员-评论家模型，这是一种无模型算法，能够操作连续的动作空间。通过使用相同的学习算法、网络架构和超参数，我们的算法稳定地解决了超过20种模拟物理任务，包括经典问题如卡车摆动、灵巧操纵、有腿的运动和汽车驾驶等。我们的算法找到的策略在性能上能够与有权访问完整动态信息及其导数的规划算法找到的策略相竞争。我们进一步展示了该算法能够“端到端”学习策略：直接从原始像素输入中学习。

2 引言

深度Q网络（DQN）技术在处理离散动作空间的任务中取得了显著成功，例如在Atari视频游戏中表现出接近或超过人类的水平。这一成就部分归功于其能够处理高维观测空间并利用深度神经网络近似动作价值函数。然而，DQN在处理连续动作空间的任务时存在限制，因为它需要在每一步都进行迭代优化来找出最大化动作价值函数的动作，这对于连续动作空间是不切实际的。

为了解决这一问题，我们提出了一种新的方法，该方法不依赖于模型，能够直接在连续动作空间中操作。我们采用了基于确定性策略梯度（DPG）的演员-评论家框架，并将其与最近在DQN中使用的一些创新结合起来，如经验回放和目标网络，这些技术已被证明可以在使用复杂、非线性函数逼近器时稳定学习过程。

此外，我们的方法能够使用相同的学习算法、网络架构和超参数，在多个物理模拟任务中表现出色，这些任务包括经典问题如倒立摆起摆和更具挑战性的任务如机器人行走。这些任务通常需要精细的动作控制，涉及高维的连续动作空间。我们的研究不仅展示了在这些连续动作空间任务中学习有效策略的可能性，还证明了我们的方法能够在这些问题上学习到与具有完整物理模型访问权限的规划算法相媲美的策略。

3 背景知识

在标准的强化学习框架中，考虑一个智能体与离散时间步的环境E进行交互。在每一个时间步t，智能体接收到一个观察结果z_t，根据策略π选择一个动作a_t，然后接收一个标量奖励r_t。所有环境都假定动作是连续的，动作空间是实数域$\mathbb{R}^N$。环境可能部分可观察，所以整个历史的观测-动作对$\{(z_1, a_1), \ldots, (z_{t-1}, a_{t-1})\}$可能需要用来描述状态$s_t$。我们将环境建模为一个马尔科夫决策过程，其具有状态空间S，动作空间$A = \mathbb{R}^N$，初始状态分布$p(s_1)$，转移动态$p(s_{t+1}|s_t, a_t)$，以及奖励函数$r(s_t, a_t)$。

智能体的行为由策略π定义，它将状态映射到动作的概率分布上$\pi : S \to P(A)$。从状态s出发的回报被定义为折扣未来奖励的总和$R_t = \sum_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i)$，其中$\gamma \in [0, 1]$是折扣因子。回报取决于选择的动作，因此取决于策略$\pi$，可能是随机的。强化学习的目标是学习一个策略，以最大化从初始分布出发的期望回报$J = \mathbb{E}[R_1]$。我们用ρ^π表示策略π下的折扣状态访问分布。

动作价值函数在许多强化学习算法中都有使用。它描述了在状态s_t下采取动作a_t并且此后遵循策略π后的期望回报：

$$Q^\pi(s_t, a_t) = \mathbb{E}[R_t|s_t, a_t]$$

强化学习中许多方法都使用了递归关系，称为贝尔曼方程：

$$Q^\pi(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E} \left[r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi} [Q^\pi(s_{t+1}, a_{t+1})] \right]$$

如果目标策略是确定性的，我们可以将其描述为一个函数$\mu : S \rightarrow A$，并避免内部期望：

$$Q^{\mu}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}\left[r(s_t, a_t) + \gamma Q^{\mu}(s_{t+1}, \mu(s_{t+1}))\right]$$

这意味着可以使用从不同随机行为策略β生成的转换来离策略地学习Q^{μ}。Q学习是一个常用的离策略算法，使用贪婪策略$\mu(s) = \arg\max_a Q(s, a)$。我们考虑使用由$\theta^Q$参数化的函数逼近器，并通过最小化损失来优化它：

$$L(\theta^Q) = \mathbb{E}_{s_t \sim \rho^{\beta}, a_t \sim \beta, r_t \sim E}\left[\left(Q(s_t, a_t|\theta^Q) - y_t\right)^2\right]$$

其中，

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q)$$

尽管y_t也依赖于θ^Q，这通常被忽略。

4 算法详解

本节详细解释了深度确定性策略梯度（DDPG）算法，该算法结合了演员-评论家框架与确定性策略梯度，以适应连续动作空间的挑战。

算法基础

由于在连续动作空间中，传统的Q学习算法需要对每一个动作执行优化，这使得算法在实际应用中变得不切实际。因此，采用基于确定性策略梯度的演员-评论家方法，维持一个参数化的演员函数 $\mu(s|\theta^{\mu})$ 指定当前策略，通过将状态映射到动作来确定性地执行动作。

策略梯度公式

演员的更新依赖于策略的性能梯度，即策略的期望回报关于演员参数的梯度。这可以通过以下公式近似计算：

$$\nabla_{\theta^{\mu}} J \approx \mathbb{E}_{s \sim \rho^{\beta}}\left[\nabla_a Q(s, a|\theta^Q)\big|_{a=\mu(s|\theta^{\mu})} \nabla_{\theta^{\mu}} \mu(s|\theta^{\mu})\right]$$

其中 $Q(s, a|\theta^Q)$ 是评论家网络评估的动作价值函数。

学习稳定性的改进

为了解决学习过程中的不稳定性问题，引入了目标网络和经验回放。目标网络帮助提供稳定的目标值，而经验回放则通过随机抽样破坏样本间的时间相关性，两者共同提升算法的学习效果。

批量归一化与探索策略

批量归一化

为了解决特征在不同环境和单元之间规模差异大的问题，本文采用了批量归一化技术，该技术通过对小批量数据的每个维度进行归一化处理，使其具有单位均值和方差，从而减少训练过程中的协变量偏移。这种方法在深度神经网络中尤为重要，因为它确保了每一层在接收输入时的稳定性和一致性，有助于网络更有效地学习。在低维情况下，该技术应用于状态输入和所有μ网络层以及Q网络的动作输入之前的所有层。

探索策略

在连续动作空间中学习的一个主要挑战是如何有效探索。DDPG算法通过在策略中添加采样自噪声过程$\mathcal{N}$的探索策略$\mu'(s_t)$来解决这一问题，这允许策略在执行时引入随机性，从而探索更多的状态-动作对。公式表达为：

$$\mu'(s_t) = \mu(s_t|\theta^{\mu}) + \mathcal{N}$$

其中，$\mathcal{N}$是适应环境特性的噪声过程，例如文中提到的Ornstein-Uhlenbeck过程，这是一种用于生成时间相关噪声的方法，特别适合物理控制问题中具有惯性的环境，因为它模仿了物理世界中的摩擦和阻力。

通过这些技术的结合使用，DDPG不仅能够在各种环境中稳定地学习，而且能够通过有效的探索策略探索复杂的动作空间，提高学习效率和策略性能。

5 DDPG算法详细分析

本节详细解释深度确定性策略梯度（DDPG）算法的具体步骤和执行过程，此算法适用于连续动作空间的问题。

算法初始化

首先，随机初始化评论家网络 $Q(s, a|\theta^Q)$ 和演员网络 $\mu(s|\theta^{\mu})$，包括它们的权重 θ^Q 和 θ^{μ}。此外，初始化目标网络 Q' 和 μ'，它们的权重被设置为 $\theta^{Q'} \leftarrow \theta^Q$ 和 $\theta^{\mu'} \leftarrow \theta^{\mu}$，以保持与原始网络同步更新的基准。经验回放缓冲区 R 也在此时初始化，用于存储和重用旧的状态转换。

探索和执行动作

对于每个训练周期或称为*episode*，初始化一个随机过程 $\mathcal{N}$ 用于行动探索。在每个时间步 t，根据当前策略和探索噪声，选择动作 $a_t = \mu(s_t|\theta^{\mu}) + \mathcal{N}_t$，执行该动作并观察得到奖励 r_t 和新的状态 s_{t+1}。

存储和采样

将转换 (s_t, a_t, r_t, s_{t+1}) 存储在回放缓冲区 R 中。随机采样 R 中的 N 个转换构成一个小批量，用于训练网络。

目标值和损失更新

对于每个采样的转换，计算目标值 $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$，其中 γ 是折扣因子。然后更新评论家网络，最小化损失 $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$。

演员网络和目标网络更新

更新演员网络，使用采样的策略梯度：

$$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)\big|_{s=s_i, a=\mu(s_i|\theta^\mu)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)\big|_{s_i}$$

同时，目标网络的权重通过软更新方式进行更新，$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$ 和 $\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$，其中 τ 是目标网络更新速率。

高维任务和环境仿真

算法适用于各种高维任务，如机械手臂操作、接触任务（如击打曲棍球）和奔跑任务（如猎豹）。所有任务除猎豹外均采用MuJoCo环境进行仿真，猎豹任务使用的是MuJoCo中的骨骼关节动力学。

以上步骤完整描述了DDPG算法的实现细节和操作过程，使其能够在连续动作空间中有效学习策略。

6 DDPG算法的Python实现详解

本节详细解读了深度确定性策略梯度（DDPG）算法的Python实现，结合论文中的算法原理和代码实现的对应关系。

6.1 初始化阶段

首先初始化网络和经验回放缓冲区。初始化评论家网络 $Q(s, a|\theta^Q)$ 和演员网络 $\mu(s|\theta^\mu)$ 以及对应的目标网络。

```
memory = Memory(limit=int(1e6), action_shape=
    env.action_space.shape, observation_shape
    =env.observation_space.shape)
critic = Critic(network=network, **
    network_kwargs)
actor = Actor(nb_actions, network=network, **
    network_kwargs)
```

6.2 探索噪声设置

配置不同类型的噪声来探索动作空间，根据噪声类型设置自适应参数噪声或正态动作噪声等。

```
if 'adaptive-param' in current_noise_type:
    param_noise = AdaptiveParamNoiseSpec(
    initial_stddev=float(stddev),
    desired_action_stddev=float(stddev))
elif 'normal' in current_noise_type:
    action_noise = NormalActionNoise(mu=np.
    zeros(nb_actions), sigma=float(stddev) *
    np.ones(nb_actions))
elif 'ou' in current_noise_type:
    action_noise =
    OrnsteinUhlenbeckActionNoise(mu=np.zeros(
    nb_actions), sigma=float(stddev) * np.
    ones(nb_actions))
```

6.3 训练与执行

描述算法的执行阶段，其中包括选择动作、执行动作、存储转换，并从经验回放中学习。

```
action, q, _, _ = agent.step(obs, apply_noise
    =True, compute_Q=True)
new_obs, r, done, info = env.step(max_action
    * action)
agent.store_transition(obs, action, r,
    new_obs, done)
```

6.4 目标网络更新

通过软更新方法逐渐逼近主网络的权重，增强学习的稳定性。

```
theta_Q' <- tau * theta_Q + (1 - tau) *
    theta_Q'
theta_mu' <- tau * theta_mu + (1 - tau) *
    theta_mu'
```

6.5 性能评估与日志记录

每个周期结束时，记录并分析训练性能，包括奖励、Q值和损失等。

```
logger.record_tabular(...)
logger.dump_tabular()
```

通过这些详细的代码段和解释，DDPG算法的核心实现被清晰展示，表明了理论与实践的紧密结合。这种方法确保了内容在单栏排版中的适应性和阅读的易懂性。