Sample Efficient Actor-Critic with Experience Replay

Ziyu Wang
DeepMind
ziyu@google.com

Victor Bapst
DeepMind
vbapst@google.com

Nicolas Heess
DeepMind
heess@google.com

Volodymyr Mnih
DeepMind
vmnih@google.com

Remi Munos
DeepMind
Munos@google.com

Koray Kavukcuoglu
DeepMind
korayk@google.com

Nando de Freitas
DeepMind, CIFAR, Oxford University
nandodefreitas@google.com

【强化学习算法 6】ACER

张楚珩
清华大学 交叉信息院博士在读

从文章的标题可以看出ACER指的actor-critic with experience replay。

原文传送门：

Mnih, Volodymyr, et al. "Asynchronous methods for deep reinforcement learning." International conference on machine learning. 2016.（前序工作）

Munos, Rémi, et al. "Safe and efficient off-policy reinforcement learning." Advances in Neural Information Processing Systems. 2016.（前序工作）

Degris, Thomas, Martha White, and Richard S. Sutton. "Off-policy actor-critic." arXiv preprint arXiv:1205.4839 (2012).（前序工作）

Wang, Ziyu, et al. "Sample efficient actor-critic with experience replay." arXiv preprint arXiv:1611.01224 (2016).

特色： 强化学习里面和环境的交互成本是比较高的，我们希望一个算法能够尽可能少地与环境交互，这个特性称之为sample efficient。提高sample efficiency的一个好办法是使用experience replay，这样就需要使用off-policy的方法来做了。这里就展示了一种off-policy actor-critic方法。

分类： Model-free、***Policy-based***、***Off-policy***、Continuous State Space、Continuous Action Space（also discrete）、Support High-dim Input

理论依据：

对于off-policy的策略来说，拿到的轨迹和on-policy的轨迹不一样，需要使用importance ratio来对其权重进行调整之后才好使用。因此policy gradient可以被写作

$$\hat{g} = (\prod_{t=0}^{k}\rho_t)\sum_{t=0}^{k}(\sum_{i=0}^{k-t}\gamma^i r_{t+i})\nabla_\theta \log\pi_\theta(a_t|x_t)$$

importance ratio的连乘是对于整个轨迹来做的，因子 $\prod_{t=0}^{k}\rho_t$ 就很容易过大或者为零。Off-policy Actor-critic这篇工作说off-policy policy gradient大致上可以单步拆成这样

$$g = \mathbb{E}_\beta[\rho_t \nabla_\theta \log\pi_\theta(a_t|x_t)Q^\pi(x_t,a_t)] \quad (1)$$

过程：

这篇工作下面就是围绕(1)式展开来做的。

1. 由于 $\rho_t = \frac{\pi(a_t|x_t)}{\mu(a_t|x_t)}$ ，这个数值很容易过大，使算法不稳定，因此使用了一个叫做importance weight truncation with bias correction的技术，做如下变换 $\mathbb{E}_\mu[\rho_t \cdots] = \mathbb{E}_\mu[\bar{\rho}_t \cdots] + \mathbb{E}_{a\sim\pi}\left[[\frac{\rho_t(a)-c}{\rho_t(a)}]_+ \cdots\right]$ ，其中 $\bar{\rho}_t = \min(c, \rho_t)$ 。这样的变换既不会产生额外的bias，而且产生的两项单独都是bounded的，前一个小于c，后一个小于1。

2. $Q^\pi(x_t, a_t)$ 的估计使用了Retrace这种技术

$$Q^{\text{ret}}(x_t, a_t) = r_t + \gamma\bar{\rho}_{t+1}[Q^{\text{ret}}(x_{t+1}, a_{t+1}) - Q(x_{t+1}, a_{t+1})] + \gamma V(x_{t+1}),$$

3. 上述的Q和V的估计使用了dueling network的结构，对于连续的行动空间，文章提出了stochastic dueling network。主要是因为行动空间连续的时候之前dueling network的 $\sum_a A_{\theta_v}(x,a)$ 没法算了，因此这里用采样的方法来计算。

$$\widetilde{Q}_{\theta_v}(x_t, a_t) \sim V_{\theta_v}(x_t) + A_{\theta_v}(x_t, a_t) - \frac{1}{n}\sum_{i=1}^{n} A_{\theta_v}(x_t, u_i), \quad \text{and} \quad u_i \sim \pi_\theta(\cdot|x_t)$$

网络输出的是 Q_{θ_v} 和 A_{θ_v} ，然后通过这个方法拼起来得到Q。

4. 通过综合了以上技术，就得到了ACER的off-policy policy gradient

$$\begin{aligned}\widehat{g}_t^{\text{acer}} &= \bar{\rho}_t \nabla_{\phi_\theta(x_t)} \log f(a_t|\phi_\theta(x))[Q^{\text{ret}}(x_t, a_t) - V_{\theta_v}(x_t)] \\ &\quad + \mathop{\mathbb{E}}_{a\sim\pi}\left(\left[\frac{\rho_t(a) - c}{\rho_t(a)}\right]_+ \nabla_{\phi_\theta(x_t)} \log f(a_t|\phi_\theta(x))[Q_{\theta_v}(x_t, a) - V_{\theta_v}(x_t)]\right).\end{aligned}$$

下面需要确定每步更新的步长。希望步长产生的变化在策略空间不要太大，因此我们希望限制KL散度的变化，为了避免像TRPO那样对于KL关于 θ 求Hessian，这里1）使用了一个average policy network，其参数更新方式为 $\theta_a \leftarrow \alpha\theta_a + (1-\alpha)\theta$ ，这样KL散度的变化主要是被一个一阶项主导，避免了求Hessian的问题；2）都对于策略分布的参数 ϕ_θ 求导，所有的优化都在对 ϕ_θ 的梯度上算清楚了，再一次BP传递到 θ 上，避免了多次BP经过神经网络的复杂运算。

通过写出trust region optimization problem

$$\begin{aligned}&\underset{z}{\text{minimize}} && \frac{1}{2}\|\hat{g}_t^{\text{acer}} - z\|_2^2 \\ &\text{subject to} && \nabla_{\phi_\theta(x_t)} D_{KL}\left[f(\cdot|\phi_{\theta_a}(x_t))\|f(\cdot|\phi_\theta(x_t))\right]^T z \le \delta\end{aligned}$$

直接解析求得最优解

$$z^* = \hat{g}_t^{\text{acer}} - \max\left\{0, \frac{k^T\hat{g}_t^{\text{acer}} - \delta}{\|k\|_2^2}\right\} k$$

可以得到参数更新公式

$\theta \leftarrow \theta + \frac{\partial\phi_\theta(x)}{\partial\theta} z^*$

算法：

Algorithm 1 ACER for discrete actions (master algorithm)

// *Assume global shared parameter vectors θ and θ_v.*
// *Assume ratio of replay r.*
repeat
  Call ACER on-policy, Algorithm 2.
  $n \leftarrow \text{Possion}(r)$
  for $i \in \{1, \cdots, n\}$ **do**
    Call ACER off-policy, Algorithm 2.
  end for
until Max iteration or time reached.

Algorithm 2 ACER for discrete actions

Reset gradients $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
Initialize parameters $\theta' \leftarrow \theta$ and $\theta'_v \leftarrow \theta_v$.
if not On-Policy **then**
  Sample the trajectory $\{x_0, a_0, r_0, \mu(\cdot|x_0), \cdots, x_k, a_k, r_k, \mu(\cdot|x_k)\}$ from the replay memory.
else
  Get state x_0
end if
for $i \in \{0, \cdots, k\}$ **do**
  Compute $f(\cdot|\phi_{\theta'}(x_i))$, $Q_{\theta'_v}(x_i, \cdot)$ and $f(\cdot|\phi_{\theta_a}(x_i))$.
  if On-Policy **then**
    Perform a_i according to $f(\cdot|\phi_{\theta'}(x_i))$
    Receive reward r_i and new state x_{i+1}
    $\mu(\cdot|x_i) \leftarrow f(\cdot|\phi_{\theta'}(x_i))$
  end if
  $\bar{\rho}_i \leftarrow \min\left\{1, \frac{f(a_i|\phi_{\theta'}(x_i))}{\mu(a_i|x_i)}\right\}$.
end for

$$Q^{ret} \leftarrow \begin{cases} 0 & \text{for terminal } x_k \\ \sum_a Q_{\theta'_v}(x_k, a) f(a|\phi_{\theta'}(x_k)) & \text{otherwise} \end{cases}$$

for $i \in \{k-1, \cdots, 0\}$ **do**
  $Q^{ret} \leftarrow r_i + \gamma Q^{ret}$
  $V_i \leftarrow \sum_a Q_{\theta'_v}(x_i, a) f(a|\phi_{\theta'}(x_i))$
  Computing quantities needed for trust region updating:

$$\begin{aligned} g &\leftarrow \min\{c, \rho_i(a_i)\} \nabla_{\phi_{\theta'}(x_i)} \log f(a_i|\phi_{\theta'}(x_i))(Q^{ret} - V_i) \\ &\quad + \sum_a \left[1 - \frac{c}{\rho_i(a)}\right]_+ f(a|\phi_{\theta'}(x_i)) \nabla_{\phi_{\theta'}(x_i)} \log f(a|\phi_{\theta'}(x_i))(Q_{\theta'_v}(x_i, a_i) - V_i) \\ k &\leftarrow \nabla_{\phi_{\theta'}(x_i)} D_{KL}\left[f(\cdot|\phi_{\theta_a}(x_i)) \| f(\cdot|\phi_{\theta'}(x_i))\right] \end{aligned}$$

  Accumulate gradients wrt θ': $d\theta' \leftarrow d\theta' + \frac{\partial \phi_{\theta'}(x_i)}{\partial \theta'}\left(g - \max\left\{0, \frac{k^T g - \delta}{\|k\|_2^2}\right\} k\right)$
  Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + \nabla_{\theta'_v}(Q^{ret} - Q_{\theta'_v}(x_i, a))^2$
  Update Retrace target: $Q^{ret} \leftarrow \bar{\rho}_i\left(Q^{ret} - Q_{\theta'_v}(x_i, a_i)\right) + V_i$
end for
Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
Updating the average policy network: $\theta_a \leftarrow \alpha\theta_a + (1 - \alpha)\theta$

Algorithm 3 ACER for Continuous Actions

Reset gradients $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
Initialize parameters $\theta' \leftarrow \theta$ and $\theta'_v \leftarrow \theta_v$.
Sample the trajectory $\{x_0, a_0, r_0, \mu(\cdot|x_0), \cdots, x_k, a_k, r_k, \mu(\cdot|x_k)\}$ from the replay memory.
for $i \in \{0, \cdots, k\}$ **do**
  Compute $f(\cdot|\phi_{\theta'}(x_i))$, $V_{\theta'_v}(x_i)$, $\widetilde{Q}_{\theta'_v}(x_i, a_i)$, and $f(\cdot|\phi_{\theta_a}(x_i))$.
  Sample $a'_i \sim f(\cdot|\phi_{\theta'}(x_i))$
  $\rho_i \leftarrow \frac{f(a_i|\phi_{\theta'}(x_i))}{\mu(a_i|x_i)}$ and $\rho'_i \leftarrow \frac{f(a'_i|\phi_{\theta'}(x_i))}{\mu(a'_i|x_i)}$
  $c_i \leftarrow \min\left\{1, (\rho_i)^{\frac{1}{d}}\right\}$.
end for

$$Q^{ret} \leftarrow \begin{cases} 0 & \text{for terminal } x_k \\ V_{\theta'_v}(x_k) & \text{otherwise} \end{cases}$$

$Q^{opc} \leftarrow Q^{ret}$
for $i \in \{k-1, \cdots, 0\}$ **do**
  $Q^{ret} \leftarrow r_i + \gamma Q^{ret}$
  $Q^{opc} \leftarrow r_i + \gamma Q^{opc}$
  Computing quantities needed for trust region updating:

$$\begin{aligned} g &\leftarrow \min\{c, \rho_i\} \nabla_{\phi_{\theta'}(x_i)} \log f(a_i|\phi_{\theta'}(x_i)) \left(Q^{opc}(x_i, a_i) - V_{\theta'_v}(x_i)\right) \\ &+ \left[1 - \frac{c}{\rho'_i}\right]_+ (\widetilde{Q}_{\theta'_v}(x_i, a'_i) - V_{\theta'_v}(x_i)) \nabla_{\phi_{\theta'}(x_i)} \log f(a'_i|\phi_{\theta'}(x_i)) \\ k &\leftarrow \nabla_{\phi_{\theta'}(x_i)} D_{KL}\left[f(\cdot|\phi_{\theta_a}(x_i)) \| f(\cdot|\phi_{\theta'}(x_i))\right] \end{aligned}$$

  Accumulate gradients wrt θ: $d\theta \leftarrow d\theta + \frac{\partial \phi_{\theta'}(x_i)}{\partial \theta'} \left(g - \max\left\{0, \frac{k^T g - \delta}{\|k\|_2^2}\right\} k\right)$
  Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + (Q^{ret} - \widetilde{Q}_{\theta'_v}(x_i, a_i)) \nabla_{\theta'_v} \widetilde{Q}_{\theta'_v}(x_i, a_i)$
  $d\theta_v \leftarrow d\theta_v + \min\{1, \rho_i\} \left(Q^{ret}(x_t, a_i) - \widetilde{Q}_{\theta'_v}(x_t, a_i)\right) \nabla_{\theta'_v} V_{\theta'_v}(x_i)$
  Update Retrace target: $Q^{ret} \leftarrow c_i \left(Q^{ret} - \widetilde{Q}_{\theta'_v}(x_i, a_i)\right) + V_{\theta'_v}(x_i)$
  Update Retrace target: $Q^{opc} \leftarrow \left(Q^{opc} - \widetilde{Q}_{\theta'_v}(x_i, a_i)\right) + V_{\theta'_v}(x_i)$
end for
Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
Updating the average policy network: $\theta_a \leftarrow \alpha\theta_a + (1-\alpha)\theta$

编辑于 2018-10-01

文章被以下专栏收录

强化学习前沿

读呀读paper

进入专栏