MPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures

Espeholt [*1] Hubert Soyer [*1] Remi Munos [*1] Karen Simonyan [1] Volodymyr Mnih [1] Tom Wa
tam Doron [1] Vlad Firoiu [1] Tim Harley [1] Iain Dunning [1] Shane Legg [1] Koray Kavukcuoglu [1]

【强化学习 44】IMPALA/V-trace

张楚珩
清华大学 交叉信息院博士在读

这篇文章主要讲了两个重要的技术：IMPALA全称是IMPortance weighted Actor-Learner Architecture，V-trace和之前本专栏讲的Retrace类似，只不过Retrace估计的是Q函数，这里估计的是V函数，因此叫V-trace。

原文传送门

Espeholt, Lasse, et al. "Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures." arXiv preprint arXiv:1802.01561 (2018).

特色

IMPLALA是一个大规模强化学习训练的框架，具有较高的性能（high throughput）、较好的扩展性（scalability）和较高的效率（data-efficiency）。在大规模计算的框架下，采样和策略更新会有一些错位（不再是完全的on-policy），在这种情况下，文章通过V-trace技术来完成地使用off-policy样本进行训练。

过程

1. IMPALA

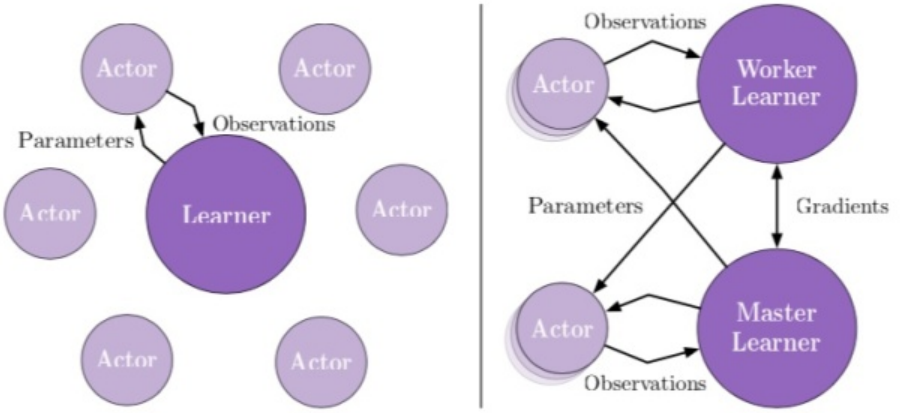

Figure 1. **Left: Single Learner.** Each *actor* generates trajectories and sends them via a queue to the *learner*. Before starting the next trajectory, *actor* retrieves the latest policy parameters from *learner*. **Right: Multiple Synchronous Learners.** Policy parameters are distributed across multiple *learners* that work synchronously.

首先，我们来看一个single learner的模式。learner的主要作用是通过获取actor得到的轨迹来做SGD来更新各个神经网络的参数，神经网络训练本身可并行的特性，learner使用的是一块GPU。actor定期从learner获取最新的神经网络参数，并且每个actor起一个模拟环境，来使用自己能获得的最新策略去采样，并且把获取到的 $\{x_t, a_t, r_t, \mu(a_t|x_t)\}$ 传回供learner去更新各个神经网络参数。由于模拟环境的运行通常不方便做并行，actor一般使用CPU。由于actor上的策略 μ 可能不是learner中最新的策略 π，因此这里使用了不同的符号来表示。

下一步，当训练规模扩大的时候，可以考虑使用多个learner（多块GPU）并且每块GPU配套多个actor（CPU）。每个learner只从自己的actor们中获取样本进行更新，learner之间定期交换gradient并且更新网络参数，actor也定期从任意learner上获取并更新神经网络参数。（这里有点没搞懂，为啥actor会去从别的learner那里拿神经网络参数？参考了[1]还是不明白）

IMPALA中actor和learner相互异步工作，极大提高了时间利用率。文章还与与batched A2C做了对比，如下图所示。a图中，正向传播和反向传播都想凑一批来做（可能是给到GPU来算），因此每一步都需要同步，而模拟环境各步所需时间方差很大，这样浪费了大量的等待时间；b图中，只把耗时较长的反向传播凑一批来做，正向传播就给各个actor自己做；而c图中的IMPALA则完全把actor和learner分开异步进行，这样actor不用去等待别的actor，可以尽可能多的做采样，相应地，所作出的牺牲就是每次更新得到的样本变为了off-policy样本。接下来本文提出了V-trace对于off-policy样本做修正。

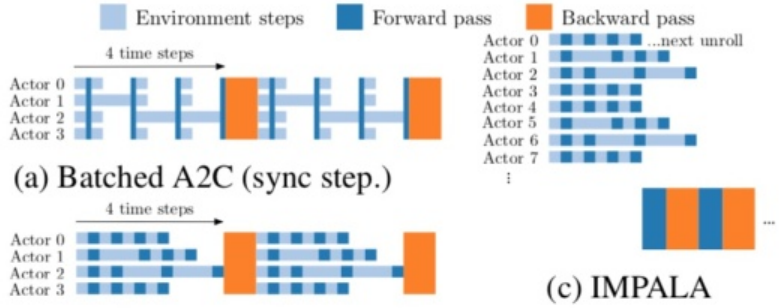

Figure 2. Timeline for one unroll with 4 steps using different architectures. Strategies shown in **(a)** and **(b)** can lead to low GPU utilisation due to rendering time variance within a batch. In **(a)**, the actors are synchronised after every step. In **(b)** after every n steps. IMPALA **(c)** decouples acting from learning.

2. V-trace

算法中需要根据采样到的样本来维护一个状态价值函数 。V-trace的目的是根据采样到的 $\{x_t, a_t, r_t, \mu(a_t|x_t)\}$ 和当前状态价值函数网络来给出当前状态价值函数的一个更好的估计 v_s（ s 下标表示它是其中的一个样本），这样价值神经网络就可以把它作为一个更新的目标来更新权重。

我们直接写出 v_s 的表达形式。

$$v_s \overset{\text{def}}{=} V(x_s) + \sum_{t=s}^{s+n-1} \gamma^{t-s} \left(\prod_{i=s}^{t-1} c_i \right) \delta_t V$$

$$\delta_t V \overset{\text{def}}{=} \rho_t \big(r_t + \gamma V(x_{t+1}) - V(x_t) \big)$$

$$\rho_t \overset{\text{def}}{=} \min \left(\bar{\rho}, \frac{\pi(a_t|x_t)}{\mu(a_t|x_t)} \right) \quad \text{such that } \bar{\rho} \geq \bar{c}$$

$$c_i \overset{\text{def}}{=} \min \left(\bar{c}, \frac{\pi(a_i|x_i)}{\mu(a_i|x_i)} \right)$$

它具有如下性质：

- 状态价值函数 V 每次往 v_s 上更新，最后能够收敛；最后面我们会证明如果有V-trace算子 $\mathcal{R}: V(x_s) \to v_s$ ，那么该算子是contraction。
- 状态价值函数 V 每次往 v_s 上更新，收敛到的状态价值函数是介于 V^π 和 V^μ 之间的某个价值函数，我们记该价值函数为 $V^{\pi_{\bar{\rho}}}$ ，该价值函数对应的策略如下所示；最后面我们通过计算V-trace算子的不动点可以得到这个结论。

$$\pi_{\bar{\rho}}(a|x) \stackrel{\text{def}}{=} \frac{\min\left(\bar{\rho}\mu(a|x), \pi(a|x)\right)}{\sum_{b \in A} \min\left(\bar{\rho}\mu(b|x), \pi(b|x)\right)},$$

- 为了避免importance weight发散，我们需要加上相应的上界来避免；参数 $\bar{\rho}$ 决定了收敛到的不动点位置； $\bar{c}$ 和 $\bar{\rho}$ 决定了收敛的速率。
- 在on-policy的情况下，如果 $\bar{\rho} \geq \bar{c} \geq 1$ ，那么 v_s 就退化为on-policy n-steps Bellman target。

3. Actor-critic

IMPALA中需要维护两个神经网络，一个是策略神经网络（用作actor），一个是状态价值函数网络（用作critic）。前面提到，V-trace技术就是根据采样到的 $\{x_t, a_t, r_t, \mu(a_t|x_t)\}$ 和当前状态价值函数网络来给出当前状态价值函数的一个更好的估计 v_s 。

3.1 Critic的更新

Critic的更新方式为最小化拟合的价值函数 $V_\theta(x_s)$ 相对于目标价值函数 v_s 的均方误差，即朝如下方向更新

$$\left(v_s - V_\theta(x_s)\right)\nabla_\theta V_\theta(x_s),$$

3.2 Actor的更新

Actor朝着off-policy policy gradient给出的梯度方向更新，即 $\mathbb{E}_\mu[\nabla \log \mu(a_s|x_s) Q^\mu(x_s, a_s)]$ 。我们更新的目标是策略 π ，而不是策略 μ ，因此要做代换 $\mu \to (\frac{\mu}{\pi})\pi$ ，把括号中的当做系数，后面的 π 才是变量，即

$$\mathbb{E}_\mu[\nabla \log \mu(a_s|x_s) Q^\mu(x_s, a_s)] = \mathbb{E}_\mu[\frac{1}{\frac{\mu}{\pi}\pi} \nabla \pi Q^\pi(x_s, a_s)] = \mathbb{E}_\mu[\frac{\pi(a_s|x_s)}{\mu(a_s|x_s)} \nabla \log \pi Q^\pi(x_s, a_s)]$$

接下来，

- $\frac{\pi(a_s|x_s)}{\mu(a_s|x_s)}$ 容易算发散，我们把它换成 $\frac{\pi_{\bar{\rho}}(a_s|x_s)}{\mu(a_s|x_s)} \propto \min(\bar{\rho}, \frac{\pi(a_s|x_s)}{\mu(a_s|x_s)}) = \rho_s$ 。
- Q^π 没法估计到，我们只能估计到 $Q^{\pi_{\bar{\rho}}}$ ，即使用 $r_s + \gamma v_{s+1}$ 作为估计。
- 使用baseline来减小误差，即减去 $V_\theta(x_s)$ 。

最后，actor的更新方向可以写为

$$\rho_s \nabla_\omega \log \pi_\omega(a_s|x_s)\big(r_s + \gamma v_{s+1} - V_\theta(x_s)\big).$$

3.3 熵

前两项再加上对于熵的激励，就可以得到最后的算法更新公式。

$$-\nabla_\omega \sum_a \pi_\omega(a|x_s) \log \pi_\omega(a|x_s).$$

实验结果

实验主要说明了以下几点：

- 相比于A3C和batched A2C，具有更好的高性能计算性能；
- 单任务训练上相比于分布式A3C、单机A3C和batched A2C有更好的性能，并且对于超参数更稳定；
- 本文中使用的V-trace相比于no correction、 ϵ-correction、1-step importance sampling都有更好的效果（ablation study）。其中no correction指的是认为样本都是on-policy样本； ϵ-correction指的是仅仅在计算 $\log\pi$ 的时候加上一个很小的数值防止不稳定（不太懂）；1-step importance sampling，V-trace其实是做了多步，这里只做一步。
- 训练单一智能体去完成多个任务。

其他技术

在这种大规模训练中，训练一次耗资巨大，为了避免训练的这一波陷入局部极小值点，采用了**population based training（PBT）**方法。每次训练若干个智能体，每隔一段时间剔除表现不好的，并且对于表现较好的智能体进行mutation（通常是扰动一下超参数组合）。通过这种方法，保证长达几天的训练结束后能得到好的结果。

有意思是，通过这种方法，学习率会随着学习进度自然慢慢减小，这和很多算法里面linear scheduled learning rate的trick不谋而合。

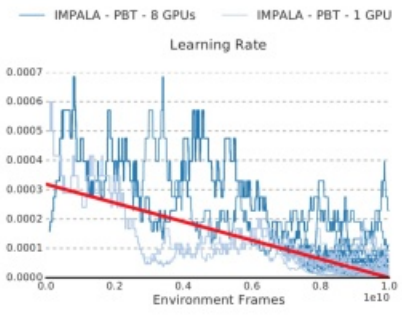

Figure F.1. Learning rate schedule that is discovered by the PBT Jaderberg et al. (2017) method compared against the linear anneal schedule of the best run from the parameter sweep (red line).

V-trace定理证明

首先定义V-trace的算子

$$\mathcal{R}V(x) \stackrel{\text{def}}{=} V(x) + \mathbb{E}_\mu\Big[\sum_{t\geq 0}\gamma^t(c_0\dots c_{t-1})\rho_t\big(r_t+\gamma V(x_{t+1})-V(x_t)\big)\big|x_0=x,\mu\Big], \tag{5}$$

不考虑神经网络 $V_\theta(x_s)$ 的拟合误差，可以认为每轮更新中价值函数都在做 $V(x)\to\mathcal{R}V(x),\forall x$ 。

定理1

关于V-trace算子不动点和收敛速度的定理可以表述为

Theorem 1. *Let* $\rho_t=\min\big(\bar\rho,\frac{\pi(a_t|x_t)}{\mu(a_t|x_t)}\big)$ *and* $c_t=\min\big(\bar c,\frac{\pi(a_t|x_t)}{\mu(a_t|x_t)}\big)$ *be truncated importance sampling weights, with* $\bar\rho\geq\bar c$. *Assume that there exists* $\beta\in(0,1]$ *such that* $\mathbb{E}_\mu\rho_0\geq\beta$. *Then the operator* $\mathcal{R}$ *defined by (5) has a unique fixed point* $V^{\pi_{\bar\rho}}$, *which is the value function of the policy* $\pi_{\bar\rho}$ *defined by*

$$\pi_{\bar\rho}(a|x)\stackrel{\text{def}}{=}\frac{\min\big(\bar\rho\mu(a|x),\pi(a|x)\big)}{\sum_{b\in A}\min\big(\bar\rho\mu(b|x),\pi(b|x)\big)}, \tag{6}$$

Furthermore, $\mathcal{R}$ *is a* η*-contraction mapping in sup-norm, with*

$$\eta\stackrel{\text{def}}{=}\gamma^{-1}-(\gamma^{-1}-1)\mathbb{E}_\mu[\sum_{t\geq 0}\gamma^t(\prod_{i=0}^{t-2}c_i)\rho_{t-1}]\leq 1-(1-\gamma)\beta<1.$$

证明

首先把算子重写，前面一个求和里面有前后两个状态的 $V(x_t),V(x_{t+1})$ ，这里把它们重排一下，

$$\mathcal{R}V(x)=(1-\mathbb{E}_\mu\rho_0)V(x)+\mathbb{E}_\mu\left[\sum_{t\geq 0}\gamma^t\Big(\prod_{s=0}^{t-1}c_s\Big)\Big(\rho_t r_t+\gamma[\rho_t-c_t\rho_{t+1}]V(x_{t+1})\Big)\right].$$

证明不动点和收敛速率最关键的是证明contraction，我们复习一下contraction的形式

$$\|\mathcal{R}V_1(x)-\mathcal{R}V_2(x)\|\leq\eta\|V_1-V_2\|_\infty$$

即每作用一次该算子，任意两个状态价值函数 V_1 离 V_2 的距离都会减小，这样作用无穷多次之后，不管最开始状态价值函数是什么，都会收敛到同一个状态价值函数上。因此，我们自然需要把上式左手边写出来看看。

$$\begin{aligned}\mathcal{R}V_1(x) - \mathcal{R}V_2(x) &= (1 - \mathbb{E}_\mu \rho_0)\big[V_1(x) - V_2(x)\big] + \mathbb{E}_\mu \left[\sum_{t\geq 0} \gamma^{t+1} \Big(\prod_{s=0}^{t-1} c_s\Big)[\rho_t - c_t\rho_{t+1}]\big[V_1(x_{t+1}) - V_2(x_{t+1})\big]\right], \\ &= \mathbb{E}_\mu \left[\sum_{t\geq 0} \gamma^t \Big(\prod_{s=0}^{t-2} c_s\Big) \underbrace{[\rho_{t-1} - c_{t-1}\rho_t]}_{\alpha_t} \big[V_1(x_t) - V_2(x_t)\big]\right],\end{aligned}$$

化简的要点就是凑出右手边 $V_1 - V_2$ 这种形式。定义系数 α_t ，并且考虑到 ρ_t 的定义，有

$$\mathbb{E}_\mu \alpha_t = \mathbb{E}\big[\rho_{t-1} - c_{t-1}\rho_t\big] \geq \mathbb{E}_\mu\big[c_{t-1}(1 - \rho_t)\big] \geq 0,$$

可以看出 $\mathcal{R}V_1(x) - \mathcal{R}V_2(x)$ 被凑成了其他 $V_1(x_t) - V_2(x_t)$ 的线性组合，如果我们能够证明组合中的各项系数和小于1，那么就容易证明 $\|\mathcal{R}^n V_1(x) - \mathcal{R}^n V_2(x)\|_\infty \to 0$ 。

$$\begin{aligned}&\sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-2} c_s\Big)[\rho_{t-1} - c_{t-1}\rho_t]\right] \\ =\ &\sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-2} c_s\Big)\rho_{t-1}\right] - \sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-1} c_s\Big)\rho_t\right] \\ =\ &\sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-2} c_s\Big)\rho_{t-1}\right] - \gamma^{-1}\left(\sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-2} c_s\Big)\rho_{t-1}\right] - 1\right) \\ =\ &\gamma^{-1} - (\gamma^{-1} - 1)\underbrace{\sum_{t\geq 0} \gamma^t \mathbb{E}_\mu \left[\Big(\prod_{s=0}^{t-2} c_s\Big)\rho_{t-1}\right]}_{\geq 1 + \gamma \mathbb{E}_\mu \rho_0} \\ \leq\ &1 - (1 - \gamma)\mathbb{E}_\mu \rho_0 \\ \leq\ &1 - (1 - \gamma)\beta \\ <\ &1.\end{aligned}$$

其中倒数第三个不等式是利用了求和符号内各项的非负性，只保留了 $t=0$ 和 $t=1$ 项，扔掉了其他项。其中倒数第三项就是该contraction的收敛速率 η ，可以看出，该速率与 β 和 ε 有关。contraction的存在，也说明了唯一不动点的存在。

下面验证 V^{π_θ} 为不动点，这只需要说明 $\mathcal{R}V^{\pi_\theta} - V^{\pi_\theta} = 0$ 即可。

$$
\begin{aligned}
& \mathbb{E}_\mu\left[\rho_t\left(r_t+\gamma V^{\pi_\rho}(x_{t+1})-V^{\pi_\rho}(x_t)\right)\middle|x_t\right] \\
= & \sum_a \mu(a|x_t)\min\left(\bar{\rho},\frac{\pi(a|x_t)}{\mu(a|x_t)}\right)\left[r(x_t,a)+\gamma\sum_y p(y|x_t,a)V^{\pi_\rho}(y)-V^{\pi_\rho}(x_t)\right] \\
= & \underbrace{\sum_a \pi_{\bar{\rho}}(a|x_t)\left[r(x_t,a)+\gamma\sum_y p(y|x_t,a)V^{\pi_\rho}(y)-V^{\pi_\rho}(x_t)\right]}_{=0}\sum_b \min\left(\bar{\rho}\mu(b|x_t),\pi(b|x_t)\right) \\
= & \ 0,
\end{aligned}
$$

其中第一等式是按照 ρ_t 的定义写出，第二个等式可以由前面 $\pi_{\bar{\rho}}$ 的定义得到，最后一个等式是因为 $V^{\pi_{\bar{\rho}}}(x_t)=\sum_a \pi_{\bar{\rho}}(a|x_t)[r(x_t,a)+\gamma\sum_{x_{t+1}} p(x_{t+1}|x_t,a)V^{\pi_{\bar{\rho}}}(x_{t+1})]$ 。

定理2

定理1说明了如果在每轮，每个状态都访问并且更新一遍，那么能收敛到一个确定的不动点。但实际中，每个状态并不能在每一轮中都是均匀和遍历地访问的，而是走一个轨迹，走到哪个状态就更新哪个状态。这种情况下（online learning），是否还能收敛呢？下面这个定理说明，它也能收敛。

Theorem 2. *Assume a tabular representation, i.e. the state and action spaces are finite. Consider a set of trajectories, with the k^{th} trajectory $x_0, a_0, r_0, x_1, a_1, r_1, \ldots$ generated by following μ: $a_t \sim \mu(\cdot|x_t)$. For each state x_s along this trajectory, update*

$$V_{k+1}(x_s) = V_k(x_s) + \alpha_k(x_s)\sum_{t\geq s}\gamma^{t-s}(c_s\ldots c_{t-1})\rho_t\left(r_t+\gamma V_k(x_{t+1})-V_k(x_t)\right), \tag{7}$$

with $c_i = \min\left(\bar{c}, \frac{\pi(a_i|x_i)}{\mu(a_i|x_i)}\right)$, $\rho_i = \min\left(\bar{\rho}, \frac{\pi(a_i|x_i)}{\mu(a_i|x_i)}\right)$, $\bar{\rho}\geq\bar{c}$. Assume that (1) all states are visited infinitely often, and (2) the stepsizes obey the usual Robbins-Munro conditions: for each state x, $\sum_k \alpha_k(x)=\infty$, $\sum_k \alpha_k^2(x)<\infty$. Then $V_k \to V^{\pi_\rho}$ almost surely.

The proof is a straightforward application of the convergence result for stochastic approximation algorithms to the fixed point of a contraction operator, see e.g. Dayan & Sejnowski (1994); Bertsekas & Tsitsiklis (1996); Kushner & Yin (2003).

参考文献

[1] Chen, Jianmin, et al. "Revisiting distributed synchronous SGD."*arXiv preprint arXiv:1604.00981*(2016).

编辑于 2019-03-06

强化学习 (Reinforcement Learning)