# Reinforcement Learning

Trong

November 1, 2018—January 21, 2019

## Contents



## 1 DOOM: Health Gathering in Acid Lake

Agent seems to converge to a specific behavior, e.g. if in the beginning it tends to turn left, it will gradually turn tighter and tighter corners until it's spinning in one place. I suspect this is a property of the loss function

$$L = -\frac{1}{n}\sum_i \langle A, \log s(\pi)\rangle \cdot G$$

$$G = \frac{[G_0 \ \cdots \ G_n] - \mu}{\sigma}$$

and not the network architecture. This is because the loss function's main goal via the $\langle A, \log s(\pi)\rangle$ factor is to create actions that conform with previous behavior. Adding randomness in the loss function, e.g. in the way

we generate the discounted rewards, seems to help, but that only delays the pathological behavior.

# 2 Actor Critic

Kinda similar to GAN's. The Critic network measures the value of the actions, while the Actor network outputs actions. The Actor Critic Model updates parameters at each step instead of waiting for the entire episode to finish.

Recall the update rule

$$\Delta\theta = \alpha G_t \nabla_\theta \ln \pi(a_t|s_t),$$

where $G$ is the reward function. In Actor Critic, we want the Critic to learn $G$ instead of soft-coding it. The loss function then becomes:

$$\Delta\theta = \alpha q(s_t, a_t) \nabla_\theta \ln \pi(a_t|s_t),$$

where $q$ is the value of taking action $a_t$ at $s_t$.

In addition to updating $\theta$ for $\pi$ during training, we must also independently train and update

$$\Delta w = \beta(q_{\text{target}} - q(s_t, a_t)) \nabla_w q(s_t, a_t),$$

where

$$q_{\text{target}} = R(s_t, a_t) + \gamma q(s_{t+1}, a_{t+1})$$

is the target value we've seen before. The difference

$$q_{\text{target}} - q(s_t, a_t)$$

is also called the Temporal Difference Error.

# 3 Advantage Function

**Definition 1.** *Define the Advantage Function to be*

$$A(s, a) = Q(s, a) - V(s),$$

*where $V(s)$ is the average value of state $s$, i.e. the average value of all actions at $s$. In other words, $A(s, a)$ measures how well action $a$ performs compared to all other actions at $s$.*

Turns out that the Advantage Function can be approximated by the Temporal Difference Error we just saw:

$$\begin{aligned} A(s,a) &= Q(s,a) - V(s) \\ &\approx R(s_t,a_t) + \gamma q(s_{t+1},a_{t+1}) - q(s_t,a_t). \end{aligned}$$

The interpretation is that in a minibatch, on average

$$V(s) \approx q(s_t,a_t),$$

and

$$Q(s,a) \approx R(s_t,a_t) + \gamma q(s_{t+1},a_{t+1}).$$

IOW the value of taking action $a$ at $s$ is the sum of the immediate reward plus the discounted expected future reward.

# 4 A2C and A3C

In A3C / Asynchronous Advantage Actor Critic, multiple agents execute in parallel, each updating the global model asynchronously. In contrast, in A2C / Advantage Actor Critic, we wait until all the agents have finished calculating their gradients before averaging them and updating the network all at once.

# 5 General Advantage Estimation

**Definition 2.** *Define the* $1,\dots,k$*-step advantages starting from step* $t$ *to be*

$$\begin{aligned} \hat{A}_t^{(1)} &:= \delta_t^V &&:= -V(s_t) + r_t + \gamma V(s_{t+1}) \\ \hat{A}_t^{(2)} &:= \delta_t^V + \gamma\delta_{t+1}^V &&= -V(s_t) + r_t + +\gamma r_{t+1} + \gamma^2 V(s_{t+2}) \\ &\cdots \\ \hat{A}_t^{(k)} &:= \sum_{l=0}^{k-1} \gamma^l \delta_{t+l}^V &&= -V\left(s_t\right) + \sum_{l=0}^{k-1} \gamma^l r_{t+l} + \gamma^k V\left(s_{t+k}\right). \end{aligned}$$

*Next define the generalized advantage estimator to be the exponentially weighted sum*

$$\begin{aligned} \hat{A}_t^{\mathrm{GAE}(\gamma,\lambda)} &= (1-\lambda)(\hat{A}_t^{(1)} + \lambda\hat{A}_t^{(2)} + \lambda^2\hat{A}_t^{(3)} + \cdots) \\ &= \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}^V. \end{aligned}$$

In other words, the $k$-step advantage $\hat{A}_t^{(k)}$ measures the advantage taking into account $k$ steps into the future (discounted by $\gamma$) and the GAE adds another level of averaging (discounted by $\lambda$) on those advantages.

# 6 Proximal Policy Optimization