ACTOR-CRITIC METHODS

Class of algorithms which combine:

→ Actor: Policy

→ Critic: Value function

Issues with REINFORCE:

- High variance
- Poor Sample efficiency
- Performance collapse

Next steps:

- Bootstrapping with Temporal Difference

  → reduces variance

- Trust regions

  → addresses performance collapse

The best baseline: advantage

$$A^{\pi_\theta}(s_t, a_t) = Q^{\pi_\theta}(s_t, a_t) - V^{\pi_\theta}(s_t)$$

Yields:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t \geq 0} \gamma^t A(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t | s_t)\right]$$

$$\mathcal{L}(\theta) = -\mathbb{E}_{\tau \sim \pi_\theta}\left[\sum_{t \geq 0} \gamma^t A^\theta(s_t, a_t) \log \pi_\theta(a_t | s_t)\right]$$

→ even further reduces the variance, BUT:

→ we have a new problem:

computing the advantage function

GENERALISED ADVANTAGE ESTIMATION

We assume an estimate of the value function

Goal: strike a balance between **bias** and **variance** for estimating the advantage

Two extremes:

- **Monte Carlo estimates:**

$$A(s_t, a_t) \hat{=} G_t - V(s_t)$$

$$\left(\text{Reminder: } G_t = \sum_{t' \geq t} \gamma^{t'-t} r_{t'}\right)$$

Low-bias: even unbiased

High variance: depends on full ($\rightarrow$ noisy) trajectories

- One-step Temporal Difference (TD)

$$A(s_t, a_t) \simeq \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

High bias : relies heavily on V

Low variance : depends on a few variables

GAE FORMULA

$$A(s_t, a_t) \simeq \sum_{t' \geq t} (\underbrace{\gamma}_{\text{discount factor}} \underbrace{\lambda}_{\substack{\text{hyperparameter} \\ \lambda \simeq 0.95}})^{t'-t} \delta_{t'}$$

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$\lambda = 0$: TD δ_t

$\lambda = 1$: MC

At this point we have the

A2C algorithm: Advantage Actor-Critic

Remark: A3C is

Asynchronous Advantage Actor-Critic,

not discussed in this course

Next step:

A2C updates can lead to catastrophic

performance drops.

REINFORCE is sample inefficient because after each update the collected data must be thrown away: it concerns an outdated strategy

Solution: IMPORTANCE SAMPLING

Goal: estimate the loss using samples collected from π_{old}

The policy loss was:

$$\mathcal{L}(\theta) = - \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t \geq 0} \gamma^t A^\theta(s_t, a_t) \log \pi_\theta(a_t | s_t) \right]$$

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[R(\tau) \right] = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t \geq 0} \gamma^t A^\theta(s_t, a_t) \right]$$

$$J^{IS}(\theta) = \mathbb{E}_{\tau \sim \pi_{old}}\left[\sum_{t \geq 0} \gamma^t \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} A^\theta(s_t, a_t)\right]$$

it is equal to $J(\theta)$ but it lets us update θ using old data

$$\mathcal{L}(\theta) = -\mathbb{E}_{\tau \sim \pi_{old}}\left[\sum_{t \geq 0} \gamma^t \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} A^\theta(s_t, a_t) \log \pi_\theta(a_t|s_t)\right]$$

Now : We can update many times with the same data.

We don't want to stray too far!

CLIPPED SURROGATE OBJECTIVE

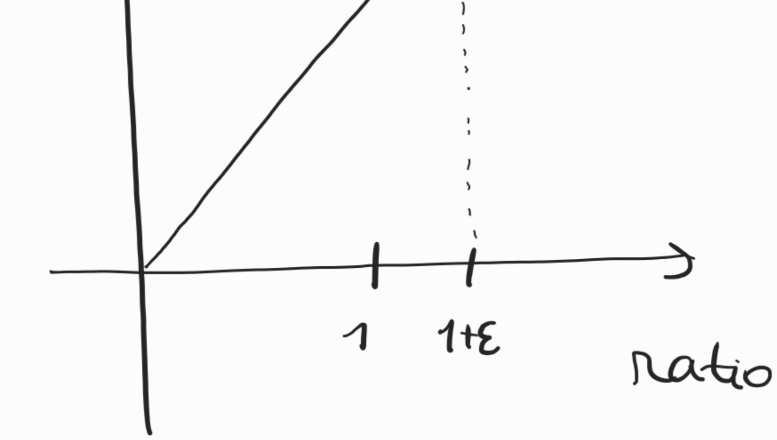
let's introduce:

$$g(r, \varepsilon, A) = \begin{cases} \min(r, 1+\varepsilon) \cdot A & A \geq 0 \\ \min(r, 1-\varepsilon) \cdot A & A < 0 \end{cases}$$

$$J^{clip}(\theta) = \mathbb{E}_{\tau \sim \sigma_{old}} \left[\sum_{t \geq 0} \gamma^t \, g\left(\frac{\sigma_\theta(a_t|s_t)}{\sigma_{old}(a_t|s_t)}, \varepsilon, A^\theta(s_t, a_t) \right) \right]$$

$$\mathcal{L}^{clip}(\theta) = - \mathbb{E}_{\tau \sim \sigma_\theta} \left[\sum_{t \geq 0} \gamma^t \, g\left(\frac{\sigma_\theta(a_t|s_t)}{\sigma_{old}(a_t|s_t)}, \varepsilon, A^\theta(s_t, a_t) \right) \log \sigma_\theta(a_t|s_t) \right]$$

case $A \geq 0$



PPO PSEUDOCODE

Data collection:

Generate a batch of steps

typical batch size: 2048

Policy optimization:

Multiple epochs over the same batch

typical: 4-10 epochs

One epoch:

- full batch is shuffled and partitioned into mini-batches

  typical: size of minibatch 32-256

- for each mini batch:
  - compute loss
  - update parameters

BELLS AND WHISTLES

- entropy loss on policy network
  - → encourages exploration
- advantage normalisation
  - → stability
- learning rate scheduling
- clipped value function
- observation normalisation
- early stopping : stop an epoch if $KL(\sigma_{old}, \sigma_{\theta}) > 0.015$
- vectorised environments