Motivations

1. Consider the λ-return:

$$g_\eta(\tau_t) = R_{t+1} + \gamma(1-\lambda)v_\theta(S_{t+1}) + \gamma\lambda g_\eta(\tau_{t+1})$$

where $\eta = \{\gamma, \lambda\}$. The value of the hyper-parameter γ, τ plays a significant role in the performance of the trained policies.

2. The paper proposes to learn the value of γ and λ during training by treating g as a function of η.

3. η is referred to as meta-parameters.

Meta-gradient RL algorithm

1. At the core of RL algorithm is an update function

$$\theta' = \theta + f(\tau, \theta, \eta)$$

where τ is a sequence of experience.

2. Define $\overline{J}(\tau', \theta', \overline{\eta})$, as the meta-objective. $\overline{\eta} = \{\overline{\lambda}, \overline{\gamma}\}$ should indicate the value of the objective we ultimately care about, not the one we optimize for. For example, setting $\overline{\lambda}, \overline{\gamma}$ both to one will set the meta-objective to the sum of un-discounted reward. The value of $\overline{\eta}$ is chosen before training and never change.

3. During the training process, we would like to update η to increase $\overline{J}$. This can be done by taking the gradient:

$$\frac{\partial \overline{J}(\tau', \theta', \overline{\eta})}{\partial \eta} = \frac{\partial \overline{J}(\tau', \theta', \overline{\eta})}{\partial \theta'} \frac{\mathrm{d}\theta'}{\mathrm{d}\eta}$$

4. $\frac{\partial \overline{J}(\tau', \theta', \overline{\eta})}{\partial \theta'}$ is usually not difficult to obtain, but it is much tricker to obtain $\frac{\mathrm{d}\theta'}{\mathrm{d}\eta}$.

5. They note that

$$\frac{\mathrm{d}\theta'}{\mathrm{d}\eta} = \frac{\mathrm{d}\theta}{\mathrm{d}\eta} + \frac{\partial f(\tau, \theta, \eta)}{\partial \eta} + \frac{\partial f(\tau, \theta, \eta)}{\partial \theta}\frac{\mathrm{d}\theta}{\mathrm{d}\eta} = \left(\mathrm{I} + \frac{\partial f(\tau, \theta, \eta)}{\partial \theta}\right)\frac{\mathrm{d}\theta}{\mathrm{d}\eta} + \frac{\partial f(\tau, \theta, \eta)}{\partial \eta}$$

6. $\frac{\partial f(\tau, \theta, \eta)}{\partial \theta}$ is a $n \times n$ matrix, where n is the size of θ. It is thus expensive to obtain. They thus make the approximation.

$$\frac{\mathrm{d}\theta'}{\mathrm{d}\eta} \approx \frac{\partial f(\tau, \theta, \eta)}{\partial \eta}$$

7. Then, the meta-parameter is updated by:

$$\Delta\eta = -\beta \frac{\partial \overline{J}(\tau', \theta', \overline{\eta})}{\partial \theta'} \frac{\partial f(\tau, \theta, \eta)}{\partial \eta}$$

Online Cross Validation

1. The key thing in this paper is that they do not need to collect extra data to update η.

2. The data used to update η at this iteration can be used by the underlying RL algorithm to update the policy parameters at the next iteration.

3. For example, at the beginning of the current iteration, we have a policy parameters value θ that was used to collect trajectory τ, τ is then used to update θ into θ'. θ' is then used to collect another trajectory τ', which is used to update η. Now, note that τ' was collected by θ', so it can be used to update θ' as well.

Meta-gradient for training the value function (prediction task)

1. Training objective is $J(\tau, \theta, \eta) = (g_\eta(\tau) - v_\theta(S))^2$.
$\frac{\partial J(\tau,\theta,\eta)}{\partial \theta} = -2\left(g_\eta(\tau) - v_\theta(S)\right)\frac{\partial v_\theta(S)}{\partial \theta}$ where τ is a trajectory startin with state S.
$f(\tau, \theta, \eta) = -\frac{\alpha}{2}\frac{\partial J(\tau,\theta,\eta)}{\partial \theta} = \alpha\left(g_\eta(\tau) - v_\theta(S)\right)\frac{\partial v_\theta(S)}{\partial \theta}$

$$\frac{\partial f(\tau,\theta,\eta)}{\partial \eta} = -\frac{\alpha}{2}\frac{\partial^2 J(\tau,\theta,\eta)}{\partial\theta\partial\eta} = \alpha\frac{\partial g_\eta(\tau)}{\partial \eta}\frac{\partial v_\theta(S)}{\partial \theta}$$

2. The meta-objective is the same as the training objective in this case.

Meta-gradient for training the policy

1. The semi-gradient of the A2C objective is defined as followed:

$$-\frac{\partial J(\tau,\theta,\eta)}{\partial \theta} = \left(g_\eta(\tau) - v_\theta(S)\right)\frac{\partial \log \pi_\theta(A|S)}{\partial \theta} + b\left(g_\eta(\tau) - v_\theta(S)\right)\frac{\partial v_\theta(S)}{\partial \theta} + c\frac{\partial H\left(\pi_\theta(\cdot|S)\right)}{\partial \theta}$$

2.

$$f(\tau,\theta,\eta) = -\alpha\frac{\partial J(\tau,\theta,\eta)}{\partial \theta} \quad \frac{\partial f(\tau,\theta,\eta)}{\partial \eta} = \alpha\frac{\partial g_\eta(\tau)}{\partial \eta}\left[\frac{\partial \log \pi_\theta(A|S)}{\partial \theta} + b\frac{\partial v_\theta(S)}{\partial \theta}\right]$$

3. The goal is to identify the return function that maximizes the return of the policy. This is measured by a meta-objective that focuses exclusively on maximizing returns, i.e. the policy gradient objective and we have:

$$\frac{\partial \overline{J}\left(\tau', \theta', \overline{\eta}\right)}{\partial \theta'} = \left(g_{\overline{\eta}}\left(\tau'\right) - v_{\theta'}\left(S'\right)\right)\frac{\partial \log \pi_{\theta'}\left(A'|S'\right)}{\partial \theta'}$$

Meta-parameter conditioned policy and value function

1. One complication with this approach is that the return function is non-stationary and its parameters η are being learned alongside the parameters of the value and policy function.

2. Thus, as η changes, there is a danger that the value and policy function becomes inaccurate wrt the return functions.

3. To deal with non-stationarity in the value function and policy, they provide the meta-parameter as an additional input to the value function and policy:

$$v_\theta^\eta(S) = v_\theta\left([S; \mathbf{e}_\eta]\right), \quad \pi_\theta^\eta(S) = \pi_\theta\left([S; \mathbf{e}_\eta]\right)$$

where e_η is the embedding of η and is updated by backpropagation during training but the gradient is not flowing through η.

Comment

1. They picked the cheapest possible approximation for $\frac{\mathrm{d}\theta'}{\mathrm{d}\eta}$. How is the effect of the approximation on the final performance. Is it possible to have computationally efficient and less erroneous approximation?