From Policy Gradient to Actor-Critic methods

Advantage Actor Critic

Olivier Sigaud

Sorbonne Université
http://people.isir.upmc.fr/sigaud

Advantage Actor Critic

- A2C is a basic Policy Gradient method with a few simple modifications
- It computes the advantage function from the value function using the current trajectory
- It adds entropy regularization to favor exploration in the gradient calculation step
- It uses n-step return, along the forward view
- The paper defines A3C, an asynchronous version where several agents generate data without a replay buffer
- A2C can be seen as a simplified version with a single agent

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. (2016) Asynchronous methods for deep reinforcement learning. *arXiv preprint arXiv:1602.01783*

Advantage function calculation

- The policy gradient update uses: $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s}_t,\mathbf{a}_t \sim \pi_{\boldsymbol{\theta}}(.)}[\nabla_{\boldsymbol{\theta}} \text{log} \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(i)}|\mathbf{s}_t^{(i)})]\hat{A}_{\boldsymbol{\phi}}(\mathbf{s}_t, \mathbf{a}_t)$
- Where $\hat{A}_{\boldsymbol{\phi}}(\mathbf{s}_t, \mathbf{a}_t)$ is computed using the value function, but no action-value function
- We note $R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V_{\boldsymbol{\phi}}(\mathbf{s}_{t+k})$
- R_t can be seen as an approximate of $Q(\mathbf{s}_t, \mathbf{a}_t)$ computed along one trajectory
- $\hat{A}_{\boldsymbol{\phi}}(\mathbf{s}_t, \mathbf{a}_t) = R_t - V_{\boldsymbol{\phi}}(\mathbf{s}_t)$

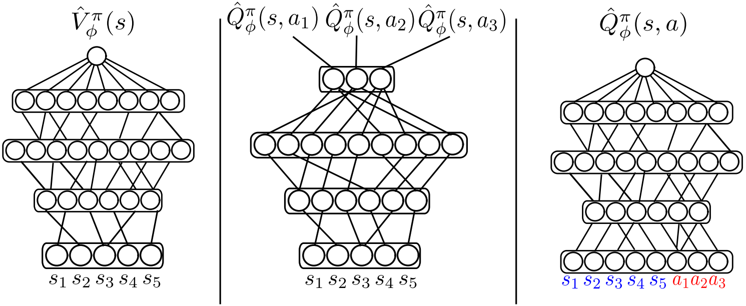
Policy Gradient updates

- Given the above calculation, the standard update would be:
- $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \pi_{\boldsymbol{\theta}}(.)}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)})](R_t - V_{\boldsymbol{\phi}}(\mathbf{s}_t))$
- But to favor exploration, A2C adds an entropy term to the gradient calculation
- $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s}_t, \mathbf{a}_t \sim \pi_{\boldsymbol{\theta}}(.)}[\nabla_{\boldsymbol{\theta}}[\log \pi_{\boldsymbol{\theta}}(\mathbf{a}_t^{(i)} | \mathbf{s}_t^{(i)})(R_t - V_{\boldsymbol{\phi}}(\mathbf{s}_t)) - \beta \mathcal{H}(\pi_{\boldsymbol{\theta}}(\mathbf{s}_t))]]$
- where $\mathcal{H}(\pi_{\boldsymbol{\theta}}(\mathbf{s}_t))$ is the entropy of policy $\pi_{\boldsymbol{\theta}}$ at state $\mathbf{s}_t$.
- Note that A2C adds entropy in the update of the actor, but not in the critic, whereas SAC adds it in the critic target, which has a deeper impact.

N-step returns: forward view

- N-step return propagates values backward to the last n visited states
- In A2C, value updates and gradient steps take place after the agent performs t_{max} steps or the episode stops.
- At each update, the agent has a collection of up to t_{max} states and rewards
- It can update the last state with the last reward, the second last step with two rewards
- And so on up to the first state with t_{max} rewards if it was before the episode stops.

Practical implementation of neural critics



- $\hat{V}^{\pi_\phi}_{\boldsymbol{\theta}}$ is smaller, but not necessarily easier to estimate
- Given the implicit max in $\hat{V}^{\pi}_{\boldsymbol{\theta}}(s)$, approx. may be less stable than $\hat{Q}^{\pi_\phi}_{\boldsymbol{\theta}}(s)$ (?)
- Note: a critic network provides a value even in unseen states