Advantage Actor-Critic (A2C) Tutorial

September 21, 2025

1 Introduction

Advantage Actor-Critic (A2C) is a synchronous variant of the actor-critic framework where a policy (actor) is updated using gradients weighted by advantage estimates supplied by a value function (critic). By updating both components jointly, A2C stabilizes policy gradient learning and supports synchronous batching across environments.

2 Theory and Formulas

2.1 Actor-Critic Objective

With policy $\pi_\theta(a \mid s)$ and value function $V_w(s)$, the policy gradient uses the advantage function $A^\pi(s,a) = Q^\pi(s,a) - V^\pi(s)$:

$$\nabla_\theta J(\theta) = \mathbb{E}_{s,a}\big[\nabla_\theta \log \pi_\theta(a \mid s)\, A^\pi(s,a)\big]. \tag{1}$$

Temporal-difference (TD) error provides a low-variance estimate of the advantage.

2.2 Critic Update

The critic minimizes the squared TD error

$$\delta_t = r_{t+1} + \gamma V_w(s_{t+1}) - V_w(s_t), \qquad w \leftarrow w + \beta \delta_t \nabla_w V_w(s_t). \tag{2}$$

In tabular settings V_w is simply updated with $V(s_t) \leftarrow V(s_t) + \beta\delta_t$.

2.3 Actor Update

The actor performs gradient ascent using the same TD error as an advantage estimate:

$$\theta \leftarrow \theta + \alpha \delta_t \nabla_\theta \log \pi_\theta(a_t \mid s_t). \tag{3}$$

Entropy regularization $H[\pi_\theta(\cdot \mid s)]$ is often added to encourage exploration. A2C typically executes multiple environments in parallel, synchronizes gradients, and updates parameters in a single batch.

3 Applications and Tips

- **Discrete control**: grid-world navigation, Atari benchmarks with synchronous roll-outs.
- **Multi-environment training**: leverage vectorized simulators to reduce variance.
- **Low-latency robotics**: when on-policy updates are feasible and stability is needed.
- **Best practices**: normalize advantages, tune entropy coefficients, monitor actor and critic losses separately, and ensure value targets remain in range via gradient clipping.

4 Python Practice

The script `gen_a2c_figures.py` trains a tabular A2C agent on a grid-world with terminal rewards. It visualizes the learning curve and the critic's value estimates after training.

Listing 1: Excerpt from gen$_a 2c_f igures.py$

```
prob = softmax(theta[state])
action = rng.choice(n_actions, p=prob)
next_state, reward, done = env.step(state, action)
td_error = reward + gamma * V[next_state] * (1.0 - float(done)) - V[state]
V[state] += critic_lr * td_error
theta[state] += actor_lr * td_error * (one_hot(action, n_actions) - prob)
```

5 Result

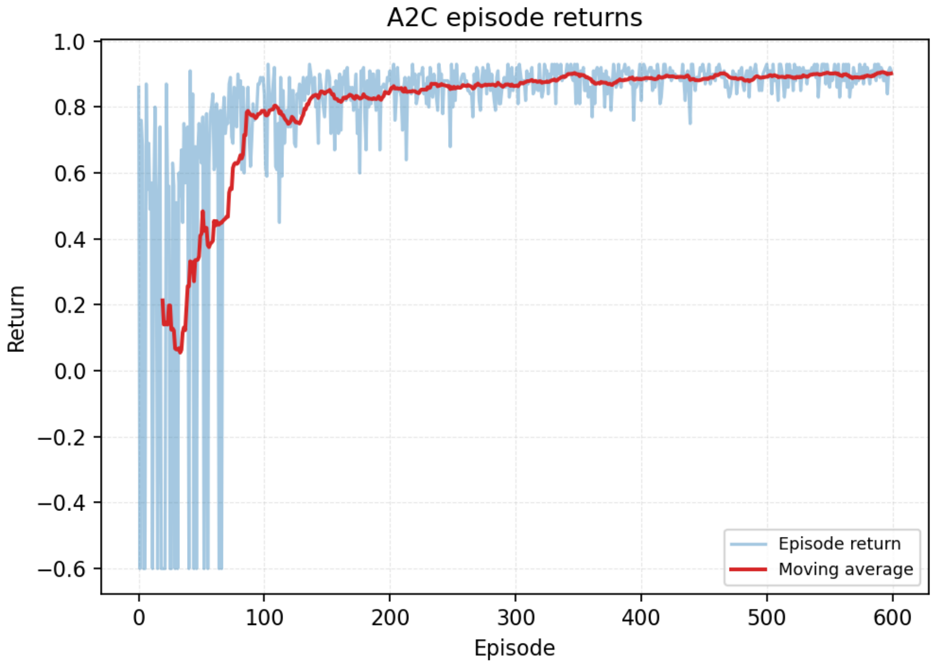

Figure 1: Episode returns during A2C training with moving average smoothing

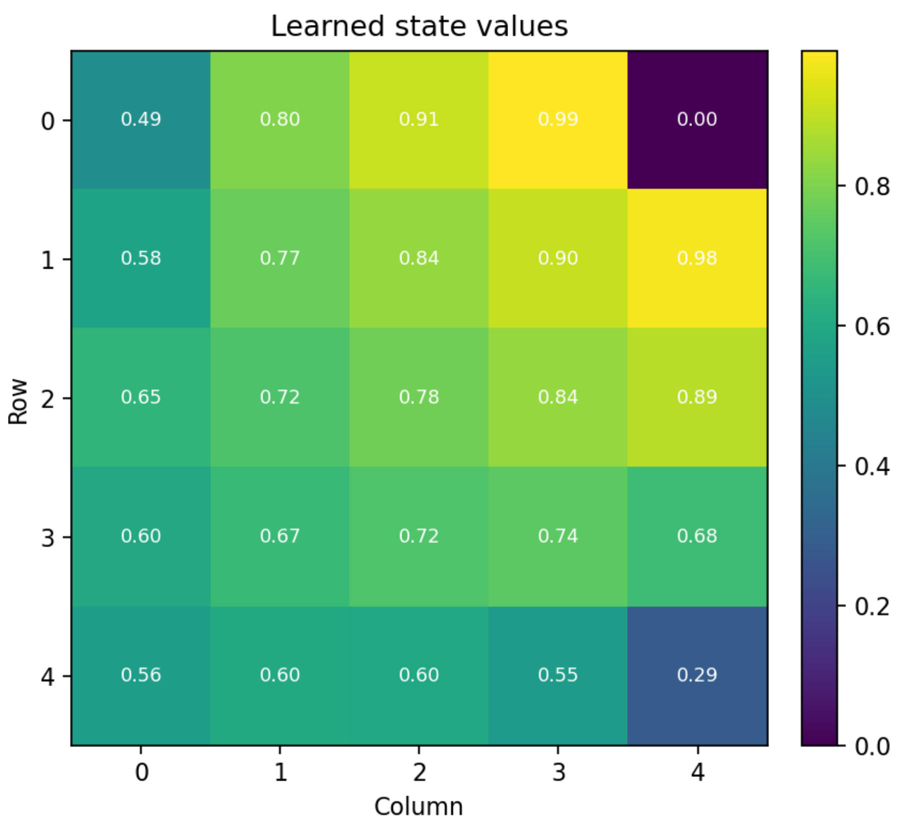

Figure 2: Value function heatmap learned by the critic, highlighting preferred routes

6 Summary

A2C synchronously updates actor and critic to reduce variance and stabilize policy gradients. Proper batching, advantage normalization, and entropy control ensure robust performance. The grid-world example demonstrates steadily improving returns and interpretable value estimates that capture the shortest-path structure.