

Lecture 11: Policy Gradients, TRPO, and PPO

Prof. Noah Golowich

Today we will derive policy gradients, trust region policy optimization (TRPO), and proximal policy optimization (PPO).

1 Markov decision processes and policy gradients

In reinforcement learning, an agent's actions determine both its rewards and the states it visits next, so a good action must account for the rewards that follow it.

1.1 The finite-horizon model

A *finite-horizon Markov decision process* (MDP) consists of a finite state space $\mathcal{S}$, a finite action space $\mathcal{A}$, a horizon $T \geq 1$, an initial-state distribution ρ_0 , transition probabilities $P_t(s' \mid s, a)$, and bounded reward functions $r_t(s, a)$. The time index ranges over $t = 0, \dots, T-1$. A policy $\pi = (\pi_0, \dots, \pi_{T-1})$ specifies a distribution $\pi_t(\cdot \mid s)$ over actions at each state and time. A trajectory $\tau = (s_0, a_0, s_1, a_1, \dots, s_{T-1}, a_{T-1}, s_T)$ is generated by

$$s_0 \sim \rho_0, \quad a_t \sim \pi_t(\cdot \mid s_t), \quad s_{t+1} \sim P_t(\cdot \mid s_t, a_t).$$

Its return is $R(\tau) = \sum_{t=0}^{T-1} r_t(s_t, a_t)$, and our objective is to maximize

$$J(\pi) := \mathbb{E}_{\tau \sim \pi}[R(\tau)]. \quad (1)$$

If an episode ends early, we pad it with an absorbing state and zero rewards. In language generation, the state is the prompt together with the tokens generated so far, an action is the next token, and the reward is often given only when the response ends.

To evaluate action a at state s and time t , suppose we take that action and then follow π . Define

$$Q_t^\pi(s, a) := \mathbb{E} \left[\sum_{u=t}^{T-1} r_u(s_u, a_u) \mid s_t = s, a_t = a, a_u \sim \pi_u \text{ for } u > t \right], \quad (2)$$

$$V_t^\pi(s) := \sum_a \pi_t(a \mid s) Q_t^\pi(s, a), \quad V_T^\pi(s) := 0, \quad (3)$$

$$A_t^\pi(s, a) := Q_t^\pi(s, a) - V_t^\pi(s). \quad (4)$$

These quantities are defined at every state, including states that π never visits. The *advantage* $A_t^\pi(s, a)$ compares taking action a with drawing an action from π_t , following π afterward in both cases. We will repeatedly use two identities, both immediate from the definitions:

$$Q_t^\pi(s, a) = r_t(s, a) + \mathbb{E}_{s' \sim P_t(\cdot \mid s, a)}[V_{t+1}^\pi(s')], \quad (5)$$

$$\sum_a \pi_t(a \mid s) A_t^\pi(s, a) = 0. \quad (6)$$

Also, $J(\pi) = \mathbb{E}_{s_0 \sim \rho_0}[V_0^\pi(s_0)]$.

1.2 The REINFORCE estimator

Now suppose $\pi = \pi_\theta$ is parameterized by $\theta \in \mathbb{R}^d$, and write $J(\theta) := J(\pi_\theta)$. We assume that every $\pi_{\theta,t}(a | s)$ is positive and differentiable in θ , as is the case for a softmax policy. The initial distribution, rewards, and transitions do not depend on θ .

The probability of a trajectory is

$$p_\theta(\tau) = \rho_0(s_0) \prod_{t=0}^{T-1} \pi_{\theta,t}(a_t | s_t) P_t(s_{t+1} | s_t, a_t).$$

Only the policy factors depend on θ , so for any trajectory with $p_\theta(\tau) > 0$,

$$\nabla_\theta \log p_\theta(\tau) = \sum_{t=0}^{T-1} \nabla_\theta \log \pi_{\theta,t}(a_t | s_t). \quad (7)$$

Using $\nabla p = p \nabla \log p$ and summing over the finitely many trajectories, we obtain

$$\nabla_\theta J(\theta) = \sum_{\tau} R(\tau) \nabla_\theta p_\theta(\tau) = \mathbb{E}_{\tau \sim \pi_\theta} \left[R(\tau) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_{\theta,t}(a_t | s_t) \right]. \quad (8)$$

The right-hand side is an expectation over trajectories from π_θ , so we can estimate it by sampling.

Averaging over N independent trajectories from π_θ gives the unbiased *REINFORCE* estimator [1]

$$\hat{g}(\theta) = \frac{1}{N} \sum_{n=1}^N \sum_{t=0}^{T-1} \nabla_\theta \log \pi_{\theta,t}(a_t^{(n)} | s_t^{(n)}) R(\tau^{(n)}), \quad (9)$$

and we update $\theta \leftarrow \theta + h \hat{g}(\theta)$ with a step size $h > 0$.

Baselines and advantages. We may subtract a fixed function $b_t(s_t)$ from $R(\tau)$ in the term for time t without changing the estimator's expectation. Indeed, conditional on s_t ,

$$\mathbb{E}_{a_t \sim \pi_{\theta,t}(\cdot | s_t)} [\nabla_\theta \log \pi_{\theta,t}(a_t | s_t)] = \sum_a \nabla_\theta \pi_{\theta,t}(a | s_t) = 0. \quad (10)$$

A good choice of this *baseline* reduces the variance. The gradient also has an equivalent advantage form, which follows by differentiating the surrogate in Proposition 1 below:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_{\theta,t}(a_t | s_t) A_t^{\pi_\theta}(s_t, a_t) \right]. \quad (11)$$

In an implementation, the sampled returns and baseline values are treated as constants: we differentiate only $\log \pi_{\theta,t}$.

The on-policy data requirement. The trajectories in (9) must come from the policy π_θ at which we want the gradient; such data is called *on-policy*. After we update θ , the same trajectories are *off-policy*, and reusing them in (9) gives a biased gradient. Importance weighting can correct the mismatch, but the weight of a trajectory is a product of T action ratios and can have large variance. Is there a better way to reuse a batch for several optimization steps?

2 A surrogate objective and trust region policy optimization

Fix the policy π that collected our data, and consider a candidate policy $\tilde{\pi}$. Moving from π to $\tilde{\pi}$ changes both the actions taken at each state and the probability of reaching each state. Following [2, 3], we will separate these two effects.

2.1 The performance difference lemma

Lemma 1 (Performance difference). *For any two policies π and $\tilde{\pi}$ in the same finite-horizon MDP,*

$$J(\tilde{\pi}) - J(\pi) = \mathbb{E}_{\tau \sim \tilde{\pi}} \left[\sum_{t=0}^{T-1} A_t^\pi(s_t, a_t) \right]. \quad (12)$$

The proof is a telescoping argument, given in Appendix A. The advantages in (12) are those of π , while the trajectory is drawn from $\tilde{\pi}$.

To write this identity as a sum over states, define the *occupancy measure* on time–state pairs by

$$\rho_\pi(t, s) := \Pr_{\tau \sim \pi}(s_t = s).$$

For each t it sums to one over s , so its total mass is T . Expanding the expectation in (12) gives

$$J(\tilde{\pi}) = J(\pi) + \sum_{t=0}^{T-1} \sum_s \rho_{\tilde{\pi}}(t, s) \sum_a \tilde{\pi}_t(a | s) A_t^\pi(s, a). \quad (13)$$

So if the expected advantage $\sum_a \tilde{\pi}_t(a | s) A_t^\pi(s, a)$ is nonnegative at every time and state, then $J(\tilde{\pi}) \geq J(\pi)$.

2.2 Freezing the state distribution

The difficulty with (13) is that the occupancy measure $\rho_{\tilde{\pi}}$ depends on the candidate, whereas our data comes from ρ_π . We therefore replace $\rho_{\tilde{\pi}}$ by ρ_π and define

$$L_\pi(\tilde{\pi}) := J(\pi) + \sum_{t=0}^{T-1} \sum_s \rho_\pi(t, s) \sum_a \tilde{\pi}_t(a | s) A_t^\pi(s, a). \quad (14)$$

This *surrogate objective* evaluates the new action probabilities at the old states. If $\pi_t(a | s) > 0$ whenever $\tilde{\pi}_t(a | s) > 0$ at states visited by π , then

$$L_\pi(\tilde{\pi}) - J(\pi) = \mathbb{E}_{\tau \sim \pi} \left[\sum_{t=0}^{T-1} \frac{\tilde{\pi}_t(a_t | s_t)}{\pi_t(a_t | s_t)} A_t^\pi(s_t, a_t) \right]. \quad (15)$$

(To check this, condition on s_t and expand the expectation over a_t ; the factor $\pi_t(a | s_t)$ cancels.) We can estimate the right-hand side from trajectories and advantage estimates under π , and then vary $\tilde{\pi}$ on the same batch. Unlike the trajectory weights of Section 1, each summand involves only one action ratio, since we correct the action distribution but not the state distribution.

The surrogate agrees with J to first order at the old policy.

Proposition 1 (First-order agreement). *For a differentiable policy family and any parameter θ_0 ,*

$$L_{\pi_{\theta_0}}(\pi_{\theta_0}) = J(\theta_0), \quad (16)$$

$$\nabla_\theta L_{\pi_{\theta_0}}(\pi_\theta) \Big|_{\theta=\theta_0} = \nabla_\theta J(\theta) \Big|_{\theta=\theta_0}. \quad (17)$$

Proof. Fix $\pi = \pi_{\theta_0}$ and write $f_t(\theta, s) = \sum_a \pi_{\theta,t}(a | s) A_t^\pi(s, a)$. By (6), $f_t(\theta_0, s) = 0$ for every t, s , which proves (16). Also, (13) gives

$$J(\theta) = J(\pi) + \sum_{t,s} \rho_{\pi_\theta}(t, s) f_t(\theta, s).$$

Each $\rho_{\pi_\theta}(t, s)$ is a finite sum of products of policy probabilities, hence differentiable in θ . By the product rule at θ_0 ,

$$\begin{aligned} \nabla_\theta J(\theta)|_{\theta_0} &= \sum_{t,s} \left[\nabla_\theta \rho_{\pi_\theta}(t, s)|_{\theta_0} \underbrace{f_t(\theta_0, s)}_{=0} + \rho_\pi(t, s) \nabla_\theta f_t(\theta, s)|_{\theta_0} \right] \\ &= \nabla_\theta L_\pi(\pi_\theta)|_{\theta_0}. \end{aligned}$$

The term involving the change in state visitation vanishes because the old policy has zero expected advantage at every state. $\square$

Hence a small enough step in any ascent direction for L_π at θ_0 also increases J . How large can the step be? To answer this, we need a bound on $J(\tilde{\pi}) - L_\pi(\tilde{\pi})$.

2.3 An idealized policy improvement algorithm

We will control the surrogate error by penalizing changes to the policy. For distributions p, q on $\mathcal{A}$, define the total variation distance and the Kullback–Leibler (KL) divergence by

$$D_{\text{TV}}(p, q) := \frac{1}{2} \sum_a |p(a) - q(a)|, \quad D_{\text{KL}}(p||q) := \sum_a p(a) \log \frac{p(a)}{q(a)},$$

where logarithms are natural, terms with $p(a) = 0$ are zero, and the divergence is infinite if $p(a) > 0 = q(a)$ for some a . For policies, we take the worst case over times and states:

$$\begin{aligned} D_{\text{TV}}^{\max}(\pi, \tilde{\pi}) &:= \max_{0 \leq t < T, s \in \mathcal{S}} D_{\text{TV}}(\pi_t(\cdot | s), \tilde{\pi}_t(\cdot | s)), \\ D_{\text{KL}}^{\max}(\pi, \tilde{\pi}) &:= \max_{0 \leq t < T, s \in \mathcal{S}} D_{\text{KL}}(\pi_t(\cdot | s) || \tilde{\pi}_t(\cdot | s)). \end{aligned}$$

The maxima range over all states, including those that the old policy never visits.

Algorithm 1, from [3], is an idealized procedure that assumes we can compute the advantages and occupancy measures exactly. Theorem 1 below justifies its penalty coefficient.

Algorithm 1 Policy iteration with a guaranteed improvement bound

- 1: Initialize a Markov policy π_0 .
 - 2: **for** $i = 0, 1, 2, \dots$ **do**
 - 3: Compute $A_t^{\pi_i}(s, a)$ and $\rho_{\pi_i}(t, s)$ for all t, s, a .
 - 4: Set $\varepsilon_i = \max_{t,s,a} |A_t^{\pi_i}(s, a)|$ and $C_i = \varepsilon_i T(T-1)$.
 - 5: Define $M_i(\pi) = L_{\pi_i}(\pi) - C_i D_{\text{KL}}^{\max}(\pi_i, \pi)$.
 - 6: Choose $\pi_{i+1} \in \arg \max_\pi M_i(\pi)$ over all Markov policies.
 - 7: **end for**
-

We remark that the analysis below does not need exact maximization in Line 6, only $M_i(\pi_{i+1}) \geq M_i(\pi_i)$.

Theorem 1 (Finite-horizon policy improvement bound). *Fix any old policy π and candidate policy $\tilde{\pi}$. Let*

$$\alpha = D_{\text{TV}}^{\max}(\pi, \tilde{\pi}), \quad \varepsilon_A = \max_{t,s,a} |A_t^\pi(s, a)|.$$

Then

$$|J(\tilde{\pi}) - L_\pi(\tilde{\pi})| \leq 2\varepsilon_A T(T-1)\alpha^2. \quad (18)$$

In particular, $J(\tilde{\pi}) \geq L_\pi(\tilde{\pi}) - 2\varepsilon_A T(T-1)\alpha^2$.

Proof. We couple trajectories of the two policies, as in the analysis of conservative policy iteration [2, 3]. One factor of α comes from the expected advantage at a fixed state, and the other from the change in the state distribution.

First, let $f_t(s) := \sum_a \tilde{\pi}_t(a | s) A_t^\pi(s, a)$. Subtracting (6) gives

$$|f_t(s)| = \left| \sum_a (\tilde{\pi}_t(a | s) - \pi_t(a | s)) A_t^\pi(s, a) \right| \leq 2\varepsilon_A \alpha. \quad (19)$$

Next, generate a trajectory under π and one under $\tilde{\pi}$ jointly, with states s_t and $\tilde{s}_t$, as follows. Start both from the same initial state. As long as the two histories agree, couple the two action distributions so that the actions differ with probability at most α , and if the actions agree, use the same next state for both. (Such a coupling of distributions p, q exists: put mass $\min\{p(a), q(a)\}$ on each pair (a, a) , for a total of $1 - D_{\text{TV}}(p, q)$ on the diagonal, and pair the remaining mass arbitrarily.) After the first disagreement, continue the two trajectories in any way with the correct marginals, e.g., independently.

At each step, the probability of a first disagreement is at most α , so a union bound over the actions $a_0, \dots, a_{t-1}$ gives

$$\Pr(s_t \neq \tilde{s}_t) \leq t\alpha. \quad (20)$$

Since s_t has distribution $\rho_\pi(t, \cdot)$ and $\tilde{s}_t$ has distribution $\rho_{\tilde{\pi}}(t, \cdot)$, subtracting (14) from (13) gives

$$\begin{aligned} |J(\tilde{\pi}) - L_\pi(\tilde{\pi})| &\leq \sum_{t=0}^{T-1} \mathbb{E}[|f_t(\tilde{s}_t) - f_t(s_t)|] \\ &\leq \sum_{t=0}^{T-1} 4\varepsilon_A \alpha \Pr(s_t \neq \tilde{s}_t) \\ &\leq 4\varepsilon_A \alpha^2 \sum_{t=0}^{T-1} t = 2\varepsilon_A T(T-1)\alpha^2. \end{aligned}$$

The second inequality holds because $f_t(\tilde{s}_t) - f_t(s_t)$ vanishes when the states agree, and is at most $4\varepsilon_A \alpha$ in absolute value otherwise by (19). $\square$

From total variation to KL. Pinsker's inequality states that $D_{\text{TV}}(p, q)^2 \leq \frac{1}{2} D_{\text{KL}}(p \| q)$. Applying it at each time and state in (18) gives

$$J(\tilde{\pi}) \geq L_\pi(\tilde{\pi}) - C_\pi D_{\text{KL}}^{\max}(\pi, \tilde{\pi}), \quad C_\pi := \varepsilon_A T(T-1). \quad (21)$$

The factor $T(T-1)$ arises because the trajectories have t chances to diverge before time t . As a sanity check, when $T = 1$ the bound says $L_\pi(\tilde{\pi}) = J(\tilde{\pi})$, which holds because all policies share the initial-state distribution.

Why the idealized update is monotone. By (21), the function M_i in Algorithm 1 is a lower bound on J . The two agree at π_i , since $L_{\pi_i}(\pi_i) = J(\pi_i)$ and the KL term vanishes there. Hence

$$J(\pi_{i+1}) - J(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i) \geq 0. \quad (22)$$

The return never decreases, and any gain in M_i is a guaranteed gain in J .

2.4 The challenge of exploration

The returns $J(\pi_i)$ are nondecreasing and bounded, so they converge. But the limit need not be optimal: the frozen state distribution hides improvements that require changing actions at unvisited states.

Consider an MDP with $T = 2$. At the initial state s_0 , action *safe* gives reward $1/2$ and moves to an absorbing state with zero subsequent reward. Action *try* gives reward zero and moves to a state s_* , where *good* gives reward 1 and *bad* gives reward zero. Suppose π always chooses *safe* at s_0 and *bad* at s_* . Then $J(\pi) = 1/2$, whereas choosing *try* and then *good* achieves return 1.

Because π plays *bad* at s_* , however, *try* has advantage $-1/2$ under π . Moreover, $\rho_\pi(1, s_*) = 0$. Hence for every candidate policy $\tilde{\pi}$,

$$L_\pi(\tilde{\pi}) = \frac{1}{2} - \frac{1}{2}\tilde{\pi}_0(\text{try} \mid s_0) \leq \frac{1}{2}.$$

The candidate's action at s_* does not appear in the surrogate at all. Since the KL penalty is nonnegative, π itself maximizes the objective in Algorithm 1, and the algorithm stalls at return $1/2$ even though it knows all advantages exactly.

With sampled data the situation is worse, since *safe* trajectories tell us nothing about the rewards at s_* . Even if π chooses *try* with a small probability $p > 0$, all N episodes miss s_* with probability $(1 - p)^N$. So giving every action positive probability does not by itself guarantee efficient exploration.

2.5 Practical TRPO: hard constraints and soft penalties

We now return to parameterized policies and let $\pi_{\text{old}} = \pi_{\theta_{\text{old}}}$ be the policy that collected the data. Following Algorithm 1 literally would mean maximizing

$$L_{\pi_{\text{old}}}(\pi_\theta) - C_{\pi_{\text{old}}} D_{\text{KL}}^{\max}(\pi_{\text{old}}, \pi_\theta)$$

over θ . The coefficient $C_{\pi_{\text{old}}}$ comes from a worst-case bound, and in practice it forces very small steps. Instead, we can choose a *trust-region* radius $\delta > 0$ and solve

$$\max_{\theta} L_{\pi_{\text{old}}}(\pi_\theta) \quad \text{subject to} \quad D_{\text{KL}}^{\max}(\pi_{\text{old}}, \pi_\theta) \leq \delta. \quad (23)$$

Checking this constraint still requires every time and state, so TRPO [3] replaces the maximum by an average over the states visited by the old policy.

Precisely, let $\mathbb{E}_{\text{old}}$ denote expectation over (t, s, a) , where (t, s) is drawn with probability $\rho_{\pi_{\text{old}}}(t, s)/T$ and then $a \sim \pi_{\text{old},t}(\cdot \mid s)$. Define

$$F(\theta) := \mathbb{E}_{\text{old}} \left[\frac{\pi_{\theta,t}(a \mid s)}{\pi_{\text{old},t}(a \mid s)} A_t^{\pi_{\text{old}}}(s, a) \right] = \frac{L_{\pi_{\text{old}}}(\pi_\theta) - J(\pi_{\text{old}})}{T}, \quad (24)$$

$$\bar{D}_{\text{KL}}(\theta_{\text{old}}, \theta) := \mathbb{E}_{\text{old}} [D_{\text{KL}}(\pi_{\text{old},t}(\cdot \mid s) \parallel \pi_{\theta,t}(\cdot \mid s))]. \quad (25)$$

The second expression for F follows from (15), where we again assume that π_{old} covers the candidate's action support at sampled states. The factor $1/T$ does not change the maximizer.

There are two natural updates, one with a hard constraint and one with a soft penalty:

$$\max_{\theta} F(\theta) \quad \text{subject to} \quad \overline{D}_{\text{KL}}(\theta_{\text{old}}, \theta) \leq \delta, \quad (26)$$

$$\max_{\theta} \left\{ F(\theta) - \beta \overline{D}_{\text{KL}}(\theta_{\text{old}}, \theta) \right\}, \quad \beta > 0. \quad (27)$$

The hard version is the usual TRPO formulation. In the soft version, we can adjust β between batches based on the observed KL.

In practice we have N sampled episodes from π_{old} . Write

$$\widehat{\mathbb{E}}[f_t(s_t, a_t)] := \frac{1}{NT} \sum_{n=1}^N \sum_{t=0}^{T-1} f_t(s_t^{(n)}, a_t^{(n)}),$$

and replace $\mathbb{E}_{\text{old}}$ by $\widehat{\mathbb{E}}$ and $A_t^{\pi_{\text{old}}}$ by estimates $\widehat{A}_t$ in (26) and (27). The samples, the old action probabilities, and the advantage estimates stay fixed during the optimization.

This answers the question from the end of Section 1. The batch becomes off-policy as soon as θ moves, but the surrogate lets us keep using it for several optimization steps while π_{θ} stays close to π_{old} . We then collect fresh trajectories from the updated policy and repeat.

3 Proximal policy optimization

PPO [4] replaces TRPO's constrained optimization with minibatch gradient ascent on a modified surrogate. We keep the sampling distribution and notation of Section 2.

3.1 Clipping the surrogate

For a sampled triple (t, s_t, a_t) , let

$$w_t(\theta) := \frac{\pi_{\theta,t}(a_t | s_t)}{\pi_{\text{old},t}(a_t | s_t)}, \quad w_t(\theta_{\text{old}}) = 1, \quad (28)$$

and let $\widehat{A}_t$ be an estimate of the old policy's advantage $A_t^{\pi_{\text{old}}}(s_t, a_t)$. For example, from a complete rollout we can take

$$\widehat{A}_t = \sum_{u=t}^{T-1} r_u(s_u, a_u) - \widehat{V}_t(s_t),$$

where $\widehat{V}_t$ approximates $V_t^{\pi_{\text{old}}}$. If $\widehat{V}_t$ is exact, the conditional expectation of this estimate given (s_t, a_t) is $A_t^{\pi_{\text{old}}}(s_t, a_t)$. We discuss estimates based on shorter trajectory segments below. In all cases, the estimates stay fixed while we optimize θ .

The empirical version of the surrogate (24) is $\widehat{F}(\theta) = \widehat{\mathbb{E}}[w_t(\theta)\widehat{A}_t]$. Maximizing it with no constraint or penalty can move the policy so far that the surrogate is no longer accurate. PPO instead removes the incentive to move far. Choose a clipping parameter $0 < \epsilon < 1$ and define

$$\text{clip}(w, 1 - \epsilon, 1 + \epsilon) := \min\{1 + \epsilon, \max\{1 - \epsilon, w\}\}.$$

The clipped objective is

$$\begin{aligned} \ell_t^{\text{CLIP}}(\theta) &:= \min\{w_t(\theta)\widehat{A}_t, \text{clip}(w_t(\theta), 1 - \epsilon, 1 + \epsilon)\widehat{A}_t\}, \\ \widehat{L}^{\text{CLIP}}(\theta) &:= \widehat{\mathbb{E}}[\ell_t^{\text{CLIP}}(\theta)]. \end{aligned} \quad (29)$$

To see why we take the minimum, write one sample's contribution as

$$\ell_t^{\text{CLIP}}(\theta) = \begin{cases} \hat{A}_t \min\{w_t(\theta), 1 + \epsilon\}, & \hat{A}_t \geq 0, \\ \hat{A}_t \max\{w_t(\theta), 1 - \epsilon\}, & \hat{A}_t < 0. \end{cases} \quad (30)$$

Suppose $\hat{A}_t > 0$, so that we want to increase the probability of the sampled action. Once the ratio reaches $1 + \epsilon$, further increases do not improve this sample's objective. Decreasing the probability, on the other hand, always lowers the objective, even when the ratio is below $1 - \epsilon$. The case $\hat{A}_t < 0$ is symmetric, with threshold $1 - \epsilon$; see Figure 1. Had we only clipped the ratio, the objective would also be flat after a large move in the wrong direction. The minimum keeps penalizing such moves.

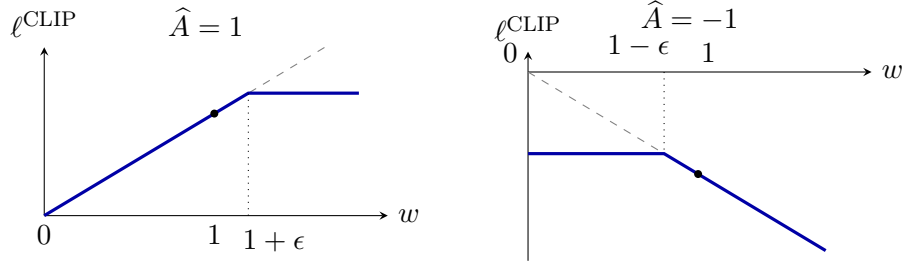


Figure 1: One sample's clipped objective for $\epsilon = 0.2$. Dashed lines show the unclipped term $w\hat{A}$; dots mark the old policy.

3.2 The combined policy, value, and entropy objective

The advantage estimates use a value predictor, which we usually train along with the policy. Suppose θ also parameterizes a value head $V_{\theta,t}(s)$, possibly sharing parameters with the policy. (This is a learned predictor, not the exact value function $V_t^{\pi_\theta}$.) Given a fixed return target V_t^{targ} for each sample, specified below, we fit the predictor with the squared loss

$$\ell_t^{\text{VF}}(\theta) := (V_{\theta,t}(s_t) - V_t^{\text{targ}})^2.$$

To encourage stochastic actions, we also reward the entropy of the policy,

$$\mathcal{H}(\pi_{\theta,t}(\cdot | s)) := - \sum_a \pi_{\theta,t}(a | s) \log \pi_{\theta,t}(a | s).$$

For coefficients $c_1, c_2 \geq 0$, the combined objective to *maximize* is

$$\hat{L}^{\text{PPO}}(\theta) := \hat{\mathbb{E}} \left[\ell_t^{\text{CLIP}}(\theta) - c_1 \ell_t^{\text{VF}}(\theta) + c_2 \mathcal{H}(\pi_{\theta,t}(\cdot | s_t)) \right]. \quad (31)$$

The value term gives us a baseline for later batches. The entropy bonus discourages the policy from concentrating too early on a few actions, though it does not guarantee efficient exploration.

Bootstrapped advantage estimates. When episodes are long, we may want to update the policy before observing the full return. Suppose our segment ends at time b , where $t < b \leq T$, and let $\hat{V}_u$ denote the value predictor at the time of data collection, with $\hat{V}_T := 0$. Define

$$V_t^{\text{targ}} := \sum_{u=t}^{b-1} r_u(s_u, a_u) + \hat{V}_b(s_b), \quad \hat{A}_t := V_t^{\text{targ}} - \hat{V}_t(s_t). \quad (32)$$

Here $\widehat{V}_b(s_b)$ stands in for the unobserved return after time b . If $\widehat{V}_t(s_t) = V_t^{\pi_{\text{old}}}(s_t)$ and $\widehat{V}_b(s_b) = V_b^{\pi_{\text{old}}}(s_b)$, then $\widehat{A}_t$ has conditional expectation $A_t^{\pi_{\text{old}}}(s_t, a_t)$. Taking $b = T$ recovers the rollout estimate above.

More generally, *generalized advantage estimation* (GAE) [5] combines estimates with different lookahead lengths. For $\lambda \in [0, 1]$, it uses

$$\widehat{A}_t^{\text{GAE}} = \sum_{u=t}^{b-1} \lambda^{u-t} \delta_u, \quad \delta_u := r_u(s_u, a_u) + \widehat{V}_{u+1}(s_{u+1}) - \widehat{V}_u(s_u). \quad (33)$$

When $\lambda = 1$, the value terms telescope and we recover (32). Smaller λ puts more weight on short lookaheads, which typically lowers the variance but relies more heavily on the value predictor.

The training loop. Each outer iteration fixes θ_{old} , collects trajectories under π_{old} , and computes the advantage estimates and value targets. Starting from $\theta = \theta_{\text{old}}$, we then take several epochs of minibatch gradient steps on (31), holding the advantage estimates, the value targets, and the denominators in (28) fixed. The result becomes the next θ_{old} .

3.3 How the loss components are used in later work

Which terms of (31) are used depends on the task and on how advantages are estimated. Even the PPO paper [4] varied these choices: its continuous-control experiments trained separate policy and value networks with no entropy bonus, while its Atari experiments used an entropy coefficient of 0.01.

PPO for language-model alignment. InstructGPT [6] uses PPO with a learned value function, and adds a per-token KL penalty toward a fixed reference policy π_{ref} , the supervised fine-tuned model. This penalty plays a different role from the clipping around π_{old} . The reference is fixed throughout training and changes which policies the objective prefers, whereas π_{old} changes after every batch and only limits the size of each update. The penalty also differs from an entropy bonus: at a fixed state,

$$-D_{\text{KL}}(\pi \parallel \pi_{\text{ref}}) = \mathcal{H}(\pi) + \mathbb{E}_{a \sim \pi}[\log \pi_{\text{ref}}(a)],$$

so it rewards entropy but also favors actions that the reference considers likely.

Removing the learned value function. Group relative policy optimization (GRPO), introduced with DeepSeekMath [7], keeps the clipped policy term but has no learned critic, and hence no value loss. It estimates advantages by sampling several responses to the same prompt and comparing their rewards. Its objective includes a KL penalty toward a reference policy and no separate entropy bonus. These group-relative estimates are not, in general, unbiased for the token-level advantages $A_t^{\pi_{\text{old}}}$ defined above.

Further changes to regularization. DAPO [8] drops the reference KL penalty and uses different upper and lower clipping thresholds. The larger upper threshold lets actions with positive estimated advantage gain more probability, which the authors use to counter entropy collapse. Across these methods the clipped policy term stays, while the critic and the regularization terms vary.

A Proof of the performance difference lemma

Proof of Lemma 1. By (5), along a trajectory drawn from $\tilde{\pi}$ we have

$$A_t^\pi(s_t, a_t) = r_t(s_t, a_t) + \mathbb{E}[V_{t+1}^\pi(s_{t+1}) \mid s_t, a_t] - V_t^\pi(s_t),$$

since the transition law given (s_t, a_t) does not depend on the policy. Taking expectations under $\tilde{\pi}$ and summing over t gives

$$\begin{aligned}\mathbb{E}_{\tau \sim \tilde{\pi}} \left[\sum_{t=0}^{T-1} A_t^{\pi}(s_t, a_t) \right] &= \mathbb{E}_{\tau \sim \tilde{\pi}} \left[\sum_{t=0}^{T-1} (r_t(s_t, a_t) + V_{t+1}^{\pi}(s_{t+1}) - V_t^{\pi}(s_t)) \right] \\ &= J(\tilde{\pi}) + \mathbb{E}_{\tau \sim \tilde{\pi}} [V_T^{\pi}(s_T) - V_0^{\pi}(s_0)] \\ &= J(\tilde{\pi}) - J(\pi).\end{aligned}$$

In the second line the value terms telescope. The last line uses $V_T^{\pi} = 0$ and $\mathbb{E}_{s_0 \sim \rho_0} [V_0^{\pi}(s_0)] = J(\pi)$, since both policies start from ρ_0 . $\square$

References

- [1] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992. <https://doi.org/10.1007/BF00992696>.
- [2] Sham Kakade and John Langford. Approximately Optimal Approximate Reinforcement Learning. *ICML*, 2002. <https://people.eecs.berkeley.edu/~pabbeel/cs287-fa09/readings/KakadeLangford-icml2002.pdf>.
- [3] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust Region Policy Optimization. *ICML*, 2015. <https://arxiv.org/abs/1502.05477>.
- [4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. 2017. <https://arxiv.org/abs/1707.06347>.
- [5] John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-Dimensional Continuous Control Using Generalized Advantage Estimation. *ICLR*, 2016. <https://arxiv.org/abs/1506.02438>.
- [6] Long Ouyang et al. Training language models to follow instructions with human feedback. *NeurIPS*, 2022. <https://arxiv.org/abs/2203.02155>.
- [7] Zhihong Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. 2024. <https://arxiv.org/abs/2402.03300>.
- [8] Qiyang Yu et al. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. 2025. <https://arxiv.org/abs/2503.14476>.

AI Use. I prepared these lecture notes with the aid of AI assistance; I take full responsibility for all content in the notes.