Post-training II

Learning from feedback

Efficient Generative Language Models

MS&E 319
Stanford University · Autumn 2026



SFT and reinforcement learning

Supervised fine-tuning (SFT)

Learn from demonstrations of desired responses.

Reinforcement learning (RL)

Generate responses and learn from reward feedback.

Ouyang et al. (2022), InstructGPT · Rafailov et al. (2023), DPO



Today's outline

1. RLHF and reward-based learning
2. From rewards to advantages
3. Reusing samples with PPO
4. Group-relative updates: GRPO
5. Learning directly from preferences
6. Evaluation and training data



Example: Assistant responses

Explain the Moon landing to a six-year-old.

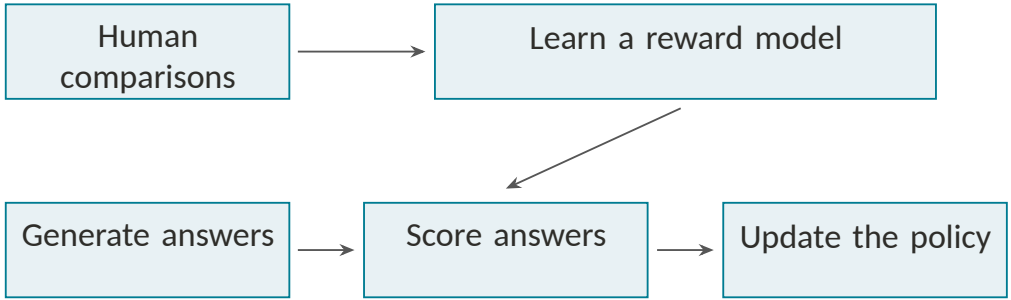
A Apollo 11 used lunar-orbit rendezvous.

B A rocket took astronauts to the Moon.
Two landed, walked around, and came home.

B is easier for a six-year-old to understand.



Recall the RLHF loop



During policy updates, the learned reward model is held fixed.

Ouyang et al. (2022), InstructGPT

What are we optimizing?

For one prompt x, improve expected reward:

$$J(\theta) = \underbrace{\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}[r_\phi(x,y)]}_{J_R(\theta)} - \beta D_{\mathsf{KL}}(\pi_\theta \| \pi_{\mathsf{ref}})$$

Policy The assistant whose parameters we update.

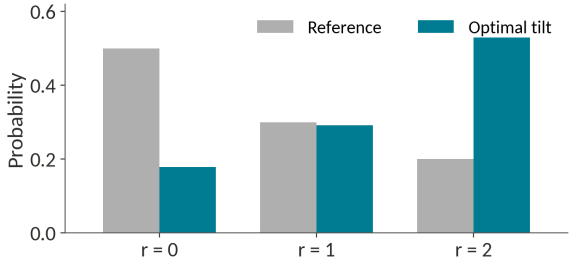
Reference A frozen checkpoint. β > 0 sets the penalty.

Ouyang et al. (2022), InstructGPT · Rafailov et al. (2023), DPO



The optimal response distribution

$$\pi^*(y \mid x) = \frac{\pi_{\text{ref}}(y|x)\exp(r_\phi(x,y)/\beta)}{Z(x)}$$



$$Z(x) = \sum_y \pi_{\text{ref}}(y \mid x)\exp(r_\phi(x,y)/\beta)$$

Higher reward increases probability relative to the reference.

Example: β = 1

Rafailov et al. (2023), DPO

Why do we still need training?

We have a formula for the optimal distribution.

$$\pi^*(y \mid x) \propto \pi_{\text{ref}}(y \mid x) e^{r(x,y)/\beta}$$

What would it take to compute Z(x)?

How would we generate the next token from this formula?

Rafailov et al. (2023), DPO



Recall the policy gradient

$$J_R(\theta) = \sum_y \pi_\theta(y \mid x)\, r(x, y)$$

$$\nabla_\theta J_R = \sum_y r(x, y)\, \nabla_\theta \pi_\theta(y \mid x)$$

$$= \sum_y \pi_\theta(y \mid x)\, r(x, y)\, \nabla_\theta \log \pi_\theta(y \mid x)$$

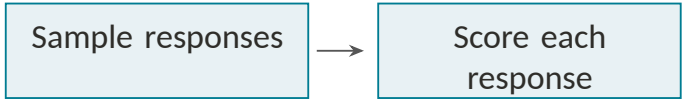
$$= \mathbb{E}_{y \sim \pi_\theta}[r(x, y)\, \nabla_\theta \log \pi_\theta(y \mid x)]$$

Estimate this expectation with sampled responses.

Williams (1992), REINFORCE



REINFORCE update



REINFORCE

$$y^{(1)}, \dots, y^{(G)} \sim \pi_\theta(\cdot \mid x), \qquad r_i = r(x, y^{(i)})$$

$$\hat{g} = \frac{1}{G} \sum_{i=1}^{G} r_i \, \nabla_\theta \log \pi_\theta(y^{(i)} \mid x)$$

$$\theta \leftarrow \theta + \eta \hat{g}, \qquad \eta > 0$$

Token-level policy gradient

Fix response i from the previous slide.

$$y^{(i)} = (y_1^{(i)}, \dots, y_{T_i}^{(i)}), \qquad r_i = r(x, y^{(i)})$$

$$g_t^{(i)} = \nabla_\theta \log \pi_\theta(y_t^{(i)} \mid x, y_{<t}^{(i)})$$

$$r_i \nabla_\theta \log \pi_\theta(y^{(i)} \mid x) = r_i \sum_{t=1}^{T_i} g_t^{(i)}$$

Each token's gradient is weighted by the same reward.



SFT and outcome-based feedback

SFT

A demonstrated response y*

$$\sum_t \nabla_\theta \log \pi_\theta(y_t^* \mid x, y_{<t}^*)$$

Outcome-based RL

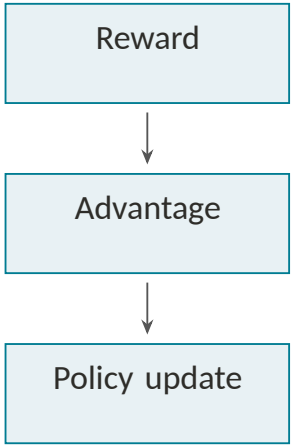
A sampled response $y^{(i)}$

$$r_i \sum_t \nabla_\theta \log \pi_\theta(y_t^{(i)} \mid x, y_{<t}^{(i)})$$

Shao et al. (2024), DeepSeekMath



Reward, advantage, policy update



A reward model or verifier scores a response.

Compare the reward with a baseline:
a critic or a group of responses.

Weight log-probability gradients.
PPO controls reuse of sampled responses.

Where can the score come from?

Labeled feedback

Learned reward model

Checkable outcomes

Tests / answer checker



A score for the sampled answer

Ouyang et al. (2022), InstructGPT · Uesato et al. (2022), math feedback



A task we can check

Write valid(s): are the brackets correctly nested?

Input	Required output
([])	True
([)]	False
)(	False

Reward = 1 if the program passes the test suite; otherwise 0.



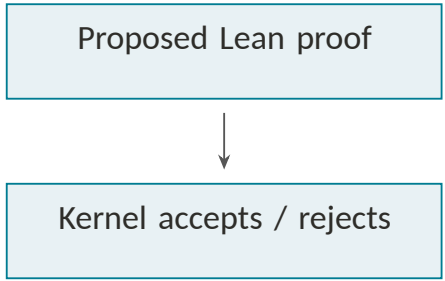
Mathematics and formal proofs

Check the final answer

Find the remainder of 2^{100} / 7.

$$2^3 \equiv 1 \pmod 7 \implies 2^{100} \equiv 2 \pmod 7$$

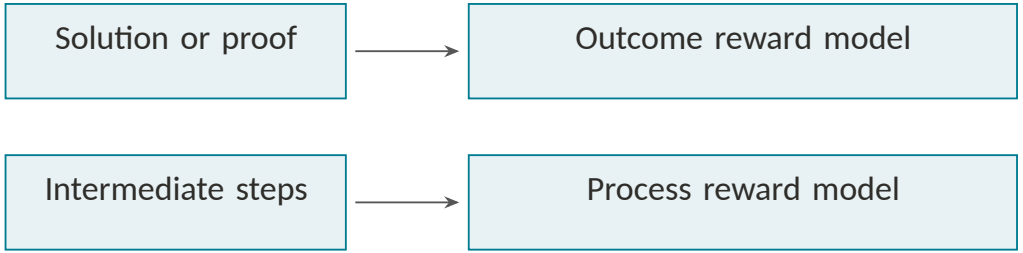
Check a formal proof



DeepSeek-AI (2025), DeepSeek-R1 · Lean reference manual, proof checking

Learned rewards for mathematics

Learned models can score solutions and proof attempts.

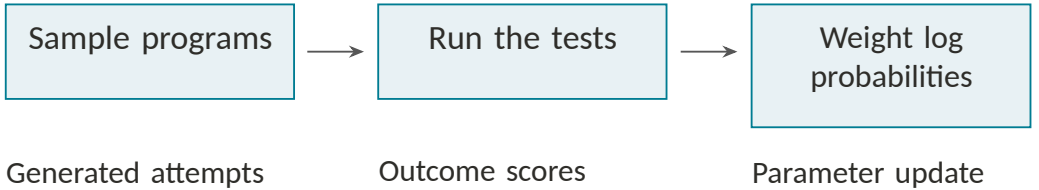


Train on correctness labels or feedback on individual steps.

Uesato et al. (2022), math feedback · Lightman et al. (2023), process supervision



RL with a verifier



How should a score become an update weight?

Where we are

1 RLHF and reward-based learning

2 From rewards to advantages

3 Reusing samples with PPO

4 Group-relative updates: GRPO

5 Learning directly from preferences

6 Evaluation and training data



A two-outcome policy

Let p be the probability of passing.

Outcome	Probability	Reward
Pass	p	1
Fail	1 – p	0

$$J_R(p) = p \cdot 1 + (1-p) \cdot 0 = p, \qquad \frac{dJ_R}{dp} = 1$$



Example: Log-probability gradients

$$\pi_p(\text{pass}) = p, \quad \pi_p(\text{fail}) = 1 - p, \quad p = \tfrac{1}{2}$$

Pass

$$\frac{d}{dp} \log p = \frac{1}{p} \cdot \frac{dp}{dp} = \frac{1}{p}$$

$$p = \tfrac{1}{2} : \qquad \frac{1}{1/2} = +2$$

Fail

$$\frac{d}{dp} \log(1 - p) = \frac{1}{1-p} \cdot (-1)$$

$$p = \tfrac{1}{2} : \qquad -\frac{1}{1/2} = -2$$



Calculating the gradient estimate

$$\hat{g} = r(y)\frac{d}{dp}\log \pi_p(y), \qquad p = \tfrac{1}{2}$$

Sample	Reward	Log derivative	Product
Pass	1	+2	$1 \times 2 = 2$
Fail	0	−2	$0 \times (-2) = 0$

Each outcome occurs with probability 1/2.

$$\mathbb{E}[\hat{g}] = \tfrac{1}{2} \cdot 2 + \tfrac{1}{2} \cdot 0 = 1 = J_R'(p)$$



Centering before the update

$$b(x) = 0.5, \qquad \hat{g}_b = (r(y) - b(x)) \tfrac{d}{dp} \log \pi_p(y)$$

Pass $(1 - 0.5) \times (+2) = 0.5 \times 2 = 1$

Fail $(0 - 0.5) \times (-2) = (-0.5) \times (-2) = 1$

The estimate is always 1, so its variance is zero.



Advantages

$$\widehat{A}_i = r(x, y^{(i)}) - b(x)$$

Reward – baseline

$$\hat{g} = \frac{1}{G} \sum_{i=1}^{G} \widehat{A}_i \nabla_\theta \log \pi_\theta(y^{(i)} \mid x)$$

Positive: better than the baseline. Negative: worse.

Williams (1992), REINFORCE



Expected log-probability gradient

$$g(y) = \nabla_\theta \log \pi_\theta(y \mid x)$$

$$\mathbb{E}_{y\sim\pi_\theta}[g(y)] = \sum_y \pi_\theta(y \mid x)\, \nabla_\theta \log \pi_\theta(y \mid x)$$

$$= \sum_y \pi_\theta(y \mid x)\, \frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)}$$

The sampling probability cancels the log derivative's denominator.



Probabilities sum to one

$$\mathbb{E}[g(y)] = \sum_y \nabla_\theta \pi_\theta(y \mid x)$$

$$= \nabla_\theta \sum_y \pi_\theta(y \mid x) = \nabla_\theta 1 = 0$$

$$\mathbb{E}[(r - b(x))g] = \mathbb{E}[rg] - b(x)\mathbb{E}[g]$$

$$= \mathbb{E}[rg]$$

b(x) is independent of the sampled response and fixed during this update.

Williams (1992), REINFORCE



Variance of the gradient estimate

Fix the prompt and the policy. Write:

$$g(y) = \nabla_\theta \log \pi_\theta(y \mid x), \qquad \hat{g}_b = (r - b(x))g(y)$$

$$\mu = \mathbb{E}[\hat{g}_b] = \nabla_\theta J_R \quad \text{independent of } b(x)$$

$$\begin{aligned} V(b) &= \mathbb{E}\big[\|\hat{g}_b - \mu\|^2\big] \\ &= \mathbb{E}\big[(r - b(x))^2 \|g(y)\|^2\big] - \|\mu\|^2 \end{aligned}$$

Greensmith et al. (2004), variance reduction



The variance-minimizing baseline

$$w(y) = \|g(y)\|^2, \qquad F(b) = \mathbb{E}[w(y)(r-b)^2]$$

$$F'(b) = 2(b\,\mathbb{E}[w] - \mathbb{E}[wr])$$

$$b^*(x) = \frac{\mathbb{E}[r\,\|g(y)\|^2]}{\mathbb{E}[\|g(y)\|^2]}$$

Use the squared log-probability gradient as the weight.

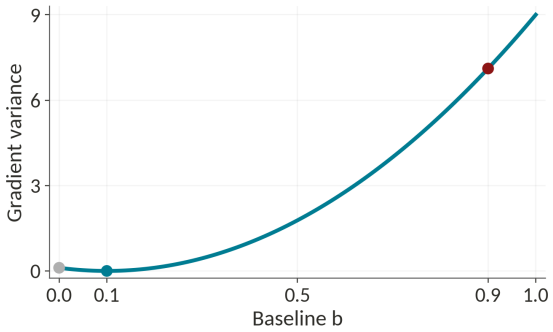
If all $\|g(y)\|$ are equal, $b^*(x) = \mathbb{E}[r]$.

Greensmith et al. (2004), variance reduction



A baseline can increase variance

$$p = 0.9, \qquad r(\text{pass}) = 1, \quad r(\text{fail}) = 0$$



No baseline

$$b = 0: \quad V = \frac{1}{9}$$

Optimal baseline

$$b = 0.1: \quad V = 0$$

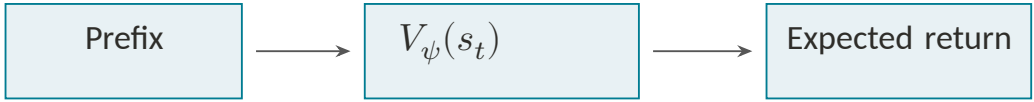
Mean reward

$$b = 0.9: \quad V = \frac{64}{9}$$

Greensmith et al. (2004), variance reduction

The critic predicts future reward

$s_t = (x, y_{<t})$ prompt and tokens already generated



$$V_\psi(s_t) \approx \mathbb{E}[R \mid s_t]$$

Fit the critic to the returns observed in rollouts.

Sutton et al. (1999), policy gradient methods

Token-level advantages

Suppose the completed response receives reward R = 1.

$$\widehat{A}_t = R - V_\psi(s_t)$$

Example critic predictions before three token choices:

Token position	Predicted return	Observed − predicted
1	0.2	1 − 0.2 = 0.8
2	0.6	1 − 0.6 = 0.4
3	0.9	1 − 0.9 = 0.1

The reward stays fixed. The baseline depends on the prefix.



A learned critic adds training cost

Evaluate values at token prefixes

A causal transformer can compute them together for a stored response.

Fit and maintain the value predictor

Additional computation, activations, parameters and optimizer state.

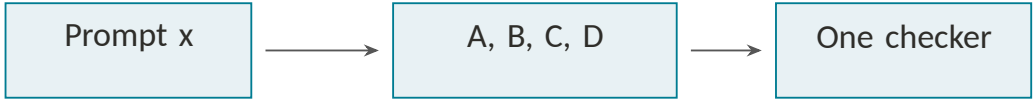
Can sibling responses supply a baseline without a critic?

Shao et al. (2024), DeepSeekMath



A group can estimate the baseline

Sample several responses to the same prompt.



$$(r_1, r_2, r_3, r_4) = (1, 0, 1, 0), \qquad \overline{r} = 0.5$$

Use the group average instead of fitting a value predictor.

Shao et al. (2024), DeepSeekMath

Example: Program rewards

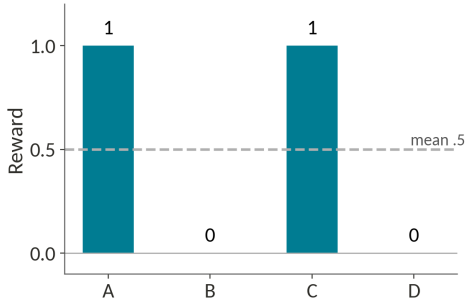
Reward rule: 1 if all six tests pass, otherwise 0.

Candidate	Tests passed	Reward
A	6 / 6	1
B	4 / 6	0
C	6 / 6	1
D	4 / 6	0

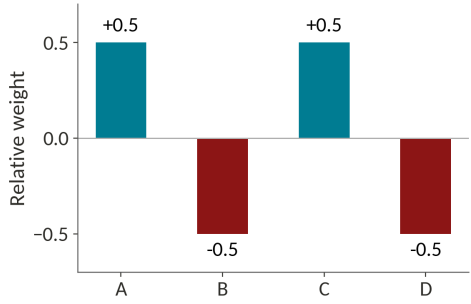


Subtracting the group mean

Rewards



Subtract the mean: 0.5



$$r_i - \overline{r}: \quad (+0.5, -0.5, +0.5, -0.5)$$

Shao et al. (2024), DeepSeekMath



GRPO: group-relative advantages

$$\widehat{A}_i = \frac{r_i - \overline{r}}{s_r + \delta}$$

$\overline{r}$: group mean (0.5 here)

s_r: group standard deviation (0.5)

A small $\delta > 0$ prevents division by zero.

Candidate	A	B	C	D
Reward	1	0	1	0
Centered reward	+0.5	−0.5	+0.5	−0.5
Advantage (δ ≈ 0)	+1	−1	+1	−1

GRPO = Group Relative Policy Optimization.

Shao et al. (2024), DeepSeekMath



What if all four attempts fail?

Attempt	A	B	C	D
Reward	0	0	0	0

$\theta \leftarrow \theta + \eta \hat{g}$ $(\eta > 0$ is the learning rate$)$

What are the group advantages?

Would increasing η create a reward signal?



Sparse rewards limit learning

The rewards within a group must vary to give nonzero advantages.

Example: Proving P ≠ NP

Suppose every attempted proof fails verification.

$$r_1 = \cdots = r_G = 0 \quad \Longrightarrow \quad \widehat{A}_1 = \cdots = \widehat{A}_G = 0$$

All advantages are zero, so increasing the learning rate cannot help this reward update.



Easy tasks can also give no contrast

Example: “What is 12×12? Answer briefly.”

Response A 144.

Response B 12×10 is 120 and 12×2 is 24. Adding them gives 144.

Both are correct: binary accuracy gives both reward 1.

If brevity matters, the feedback must also capture it.



Partial-credit rewards

Score each program by the fraction of tests it passes.

Tests passed	Binary reward	Fraction passed
0 / 6	0	0
2 / 6	0	1 / 3
4 / 6	0	2 / 3

Use partial credit when passing more tests indicates progress toward a correct program.



REINFORCE leave-one-out

Baseline: mean reward of the other responses.

$$b_{-i}(x) = \frac{1}{G-1}\sum_{j\neq i} r_j, \qquad \widehat{A}_i^{\mathrm{LOO}} = r_i - b_{-i}(x)$$

Response	Own reward	Other rewards	Advantage
A	1	0, 1, 0	1 − 1/3 = 2/3
B	0	1, 1, 0	0 − 2/3 = −2/3

For independent attempts, the baseline excludes the response it weights.
Ahmadian et al. (2024), REINFORCE leave-one-out



RLOO versus GRPO advantages

RLOO

Subtract the mean of the other responses.

$$\widehat{A}_i^{\text{LOO}} = r_i - b_{-i}(x)$$

Outcome-based GRPO

Subtract the group mean and divide by the standard deviation.

$$\widehat{A}_i^{\text{GRPO}} = \frac{r_i - \overline{r}}{s_r + \delta}$$

Both avoid a learned critic.

Ahmadian et al. (2024), REINFORCE leave-one-out · Shao et al. (2024), DeepSeekMath



Choosing a group baseline

RLOO

- No learned critic.
- Basic on-policy estimator preserves the reward gradient in expectation.

Raw reward scale still affects the size of updates.

GRPO normalization

- No learned critic.
- Reduces sensitivity to differences in reward scale across groups.

Small group spread can magnify noise; normalization changes task weighting.

Ahmadian et al. (2024), REINFORCE leave-one-out · Shao et al. (2024), DeepSeekMath



Where we are

1 RLHF and reward-based learning

2 From rewards to advantages

3 Reusing samples with PPO

4 Group-relative updates: GRPO

5 Learning directly from preferences

6 Evaluation and training data



Rollout cost

Generate tokens	Run tests / score
Sequential decoding	**Environment or scorer calls**

Can we learn more from a batch we have already generated?



Collect a rollout batch



Example rollout batch	Stored per response
8 prompt–response pairs	Tokens and old log probabilities
Score each response	Reward and estimated advantages

Fresh rollout batch: N = 8 responses.

Schulman et al. (2017), PPO



Steps and epochs within one batch

8 stored responses; minibatches of 2.

Step One parameter update from one minibatch.

Epoch One pass through all 8 stored responses.

Epoch 1	1, 2	3, 4	5, 6	7, 8
Epoch 2	3, 8	2, 5	1, 7	4, 6

4 steps per epoch $\times$ 2 epochs = 8 updates; no new responses.

Schulman et al. (2017), PPO



Old and current policies

At generation

π_{old}

Generated this batch.
Frozen while we reuse it.

During updates

π_θ

Starts at the old policy.
Changes after each step.

Compare their probabilities on the same sampled tokens.



The probability ratio

One token of response i, with its sampled prefix held fixed:

Old probability

0.20

Current probability

0.24

$$\rho_{i,t}(\theta) = \frac{\pi_\theta(y_t^{(i)}|x, y_{<t}^{(i)})}{\pi_{\text{old}}(y_t^{(i)}|x, y_{<t}^{(i)})} = \frac{0.24}{0.20} = 1.2$$

Schulman et al. (2017), PPO



Advantage versus probability ratio

Advantage

$\widehat{A}$

Signed feedback for this sample.
Fixed during batch reuse.

Probability ratio

$\rho(\theta)$

Current / old probability.
Changes as θ changes.

For the next few slides, we suppress the response and token indices.

Schulman et al. (2017), PPO · Shao et al. (2024), DeepSeekMath



Reusing responses: an exact identity

For fixed x, let q(y) be the policy that generated the batch.

$$J_R(\theta) = \sum_y \pi_\theta(y \mid x) r(y)$$

$$= \sum_y q(y) \frac{\pi_\theta(y|x)}{q(y)} r(y)$$

$$= \mathbb{E}_{y \sim q}\left[\rho_{\text{seq}}(y;\theta) r(y)\right], \quad \rho_{\text{seq}} = \frac{\pi_\theta(y|x)}{q(y)}$$

A full-response ratio corrects for the distribution that generated the data.
Schulman et al. (2017), PPO



Local surrogate objective

$$\rho_{\text{seq}} = \prod_t \rho_t, \qquad \rho_t = \frac{\pi_\theta(y_t|s_t)}{\pi_{\text{old}}(y_t|s_t)}$$

PPO uses token ratios on stored prefixes:

$$L_{\text{local}}(\theta) = \mathbb{E}_{\text{old}}\left[\sum_t \rho_t(\theta)\widehat{A}_t\right]$$

$$\nabla_\theta L_{\text{local}}|_{\theta=\theta_{\text{old}}} = \mathbb{E}_{\text{old}}\left[\sum_t \widehat{A}_t \nabla_\theta \log \pi_\theta(y_t \mid s_t)\right]_{\theta=\theta_{\text{old}}}$$

At the old policy, recover the policy-gradient direction. Then keep updates local.
Schulman et al. (2017), PPO



Why clip the objective?

$\widehat{A} = +1, \qquad \pi_{\text{old}}(\text{token} \mid \text{prefix}) = 0.20$

Current probability	Ratio ρ	Objective value
0.20	1.0	1.0
0.24	1.2	1.2
0.30	1.5	1.5

For positive advantage, the objective keeps increasing with the sampled token's probability.

Schulman et al. (2017), PPO



Clipping a ratio

$$\varepsilon = 0.2, \qquad c(\rho) = \mathrm{clip}(\rho, 0.8, 1.2)$$

Ratio ρ	Clipped value c(ρ)
0.5	0.8
1.0	1.0
1.5	1.2

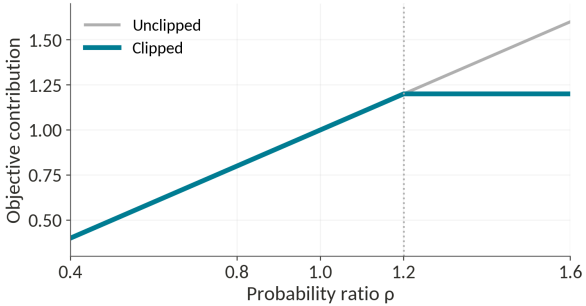
This clamps a number used in the objective. It does not overwrite probabilities.

Schulman et al. (2017), PPO



Positive advantage

$$\widehat{A} = +1, \qquad \ell(\rho) = \min(\rho, 1.2)$$



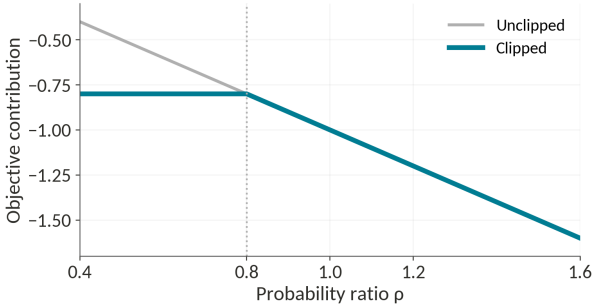
Increasing probability helps until ρ = 1.2. Then this term is flat.

Maximize the objective.

Schulman et al. (2017), PPO

Negative advantage

$$\widehat{A} = -1, \qquad \ell(\rho) = -\max(\rho, 0.8)$$



Decreasing probability helps until ρ = 0.8. Then this term is flat.

Beyond the clipping threshold, this term is flat.

Schulman et al. (2017), PPO

The PPO clipped objective

Simplified contribution from one sampled token:

$$\ell(\theta) = \min\Big\{\rho(\theta)\widehat{A},\ \mathrm{clip}(\rho(\theta), 1-\varepsilon, 1+\varepsilon)\widehat{A}\Big\}$$

After an update, one token has ρ = 1.3 with ε = 0.2.

Does that necessarily mean PPO was implemented incorrectly?

No. Clipping changes the objective, not the set of allowed policies.

Schulman et al. (2017), PPO



Reference and old policies

Reference policy

SFT checkpoint

Fixed across the training phase

Old policy

Refreshed each batch

Batch 1 | Batch 2 | Batch 3

KL uses the reference. Clipping compares to the policy that generated the batch.

Schulman et al. (2017), PPO · Ouyang et al. (2022), InstructGPT



The PPO batch lifecycle

1 · Collect	Generate a fresh batch under the old policy.
2 · Evaluate	Compute rewards, advantages and value targets.
3 · Optimize	Reuse this batch for a small number of epochs. Update the policy and critic.
4 · Replace	Discard this batch and sample a new one with the updated policy.

Starting point: 1 epoch. Compare 2–4 if held-out quality and policy drift allow.

Ouyang et al. (2022), InstructGPT · Schulman et al. (2017), PPO



Where we are

1 RLHF and reward-based learning

2 From rewards to advantages

3 Reusing samples with PPO

4 Group-relative updates: GRPO

5 Learning directly from preferences

6 Evaluation and training data



PPO and outcome-based GRPO

	Conventional PPO	Outcome-based GRPO
Baseline	Critic at each prefix	Reward mean of G responses
Advantage	Prefix-dependent estimate	One normalized value per response
Critic training	Yes	No
Policy updates	Clipped token objective	Clipped token objective

Critic prediction: $V_\psi(s_t)$

Both reuse rollout batches and can use a verifier or a reward model.

Schulman et al. (2017), PPO · Shao et al. (2024), DeepSeekMath



One GRPO iteration

Collect groups	For each prompt, sample G responses with the old policy.
Score and compare	Apply the same reward rule to every response. Compute each prompt's group advantages.
Reuse	Make clipped minibatch updates with fixed advantages.
Replace	Discard the batch and sample fresh groups.

Shao et al. (2024), DeepSeekMath



GRPO uses the same clipped term

$$\widehat{A}_i = \frac{r_i - \bar{r}}{s_r + \delta}, \qquad \rho_{i,t} = \frac{\pi_\theta(y_t^{(i)} | x, y_{<t}^{(i)})}{\pi_{\text{old}}(y_t^{(i)} | x, y_{<t}^{(i)})}$$

For one prompt's G responses, maximize:

$$\frac{1}{G} \sum_{i=1}^{G} \frac{1}{T_i} \sum_{t=1}^{T_i} \min\Big\{ \rho_{i,t} \widehat{A}_i,\ \text{clip}(\rho_{i,t}, 1-\varepsilon, 1+\varepsilon) \widehat{A}_i \Big\}$$

T_i is response length. Average over prompts and add reference KL if used.

Shao et al. (2024), DeepSeekMath



Example: DeepSeek-R1-Zero



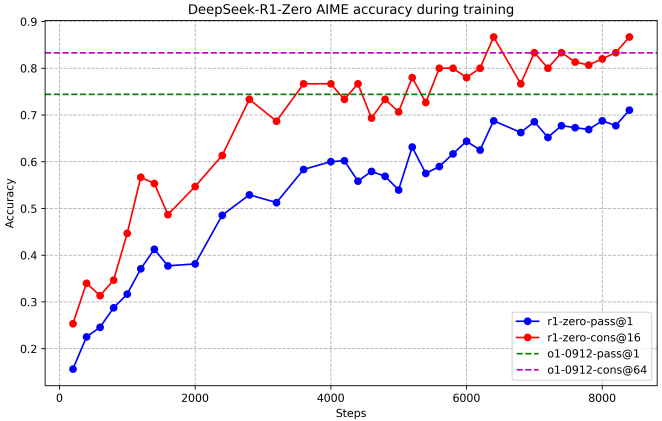
Rule-based feedback

Answer accuracy + required response format

R1-Zero uses no preliminary SFT. The full DeepSeek-R1 recipe includes SFT.

DeepSeek-AI (2025), DeepSeek-R1

DeepSeek-R1-Zero: Results



AIME 2024

pass@1
15.6% → 71.0%

r1-zero-pass@1
r1-zero-cons@16
o1-0912-pass@1
o1-0912-cons@64

Consensus uses 16 answers at inference: a different evaluation budget.

DeepSeek-AI (2025), DeepSeek-R1



Where we are

1 RLHF and reward-based learning

2 From rewards to advantages

3 Reusing samples with PPO

4 Group-relative updates: GRPO

5 Learning directly from preferences

6 Evaluation and training data



Learning from preference pairs

Explain the Moon landing to a six-year-old.

A Apollo 11 used lunar-orbit rendezvous.

B A rocket took astronauts to the Moon.
Two landed, walked around, and came home.

Can this pair train the assistant directly?

Rafailov et al. (2023), DPO



Regularized objective

For a fixed prompt x, optimize over response distributions $\pi(\cdot \mid x)$.

$$\max_\pi \sum_y \pi(y \mid x) \left[r(x, y) - \beta \log \frac{\pi(y|x)}{\pi_{\text{ref}}(y|x)} \right]$$

$$\sum_y \pi(y \mid x) = 1, \qquad \beta > 0$$

What distribution would be optimal if the reward were known?

Rafailov et al. (2023), DPO



Optimal policy

$$\pi^*(y \mid x) = \frac{\pi_{\text{ref}}(y|x)\exp(r(x,y)/\beta)}{Z(x)}$$

We can express reward differences in terms of policy probabilities.

Rafailov et al. (2023), DPO



Reward from the policy

$$\pi^*(y \mid x) = \pi_{\text{ref}}(y \mid x) \exp(r(x, y)/\beta)/Z(x)$$

$$\log \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} = \frac{r(x,y)}{\beta} - \log Z(x)$$

$$r(x, y) = \beta \log \frac{\pi^*(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x)$$

Rafailov et al. (2023), DPO



Comparisons cancel the normalizer

Preferred answer w and less-preferred answer l share prompt x.

$$r_w = \beta \log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)} \qquad \cancel{+\beta \log Z(x)}$$

$$r_l = \beta \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \qquad \cancel{+\beta \log Z(x)}$$

$$r_w - r_l = \beta \left[\log \frac{\pi^*(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log \frac{\pi^*(y_l|x)}{\pi_{\text{ref}}(y_l|x)} \right]$$

Rafailov et al. (2023), DPO

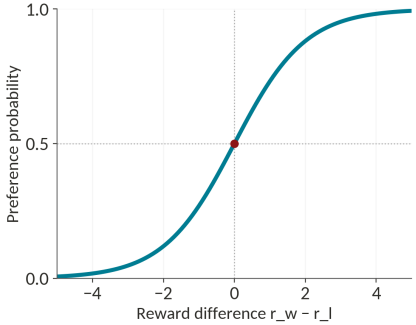


Predicting a preference

Bradley–Terry model:

$$P(y_w \succ y_l \mid x) = \sigma(r_w - r_l)$$

$$\sigma(z) = \frac{1}{1+e^{-z}}$$



Model the reward difference with the trainable policy:

$$r_{\theta,w} - r_{\theta,l} = \beta\left[\log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\right]$$

Rafailov et al. (2023), DPO



Train directly on the pair

Fit the observed preference pairs.

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\beta\left[\log\frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \log\frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right]\right)\right].$$

Preferred answer

Less-preferred answer

Rafailov et al. (2023), DPO



Compare two candidate policies

B is preferred to A.

Policy	Preferred B	Other A	All others
Reference	0.10	0.20	0.70
Candidate 1	0.20	0.30	0.50
Candidate 2	0.20	0.50	0.30

Which candidate has the lower DPO loss on this pair?



DPO compares relative odds

Policy	Preferred / other	Relative to reference
Reference	0.10 / 0.20 = 1/2	1
Candidate 1	0.20 / 0.30 = 2/3	4/3
Candidate 2	0.20 / 0.50 = 2/5	4/5

Candidate 1 has the larger margin and the lower loss.

Rafailov et al. (2023), DPO



DPO gradient

$$-\nabla_\theta \ell_{\text{DPO}} = c_\theta \left[\nabla_\theta \log \pi_\theta(y_w \mid x) - \nabla_\theta \log \pi_\theta(y_l \mid x)\right]$$

$$c_\theta = \beta\, \sigma\big(-(r_{\theta,w} - r_{\theta,l})\big) > 0$$

Preferred answer

Positive coefficient

Less-preferred answer

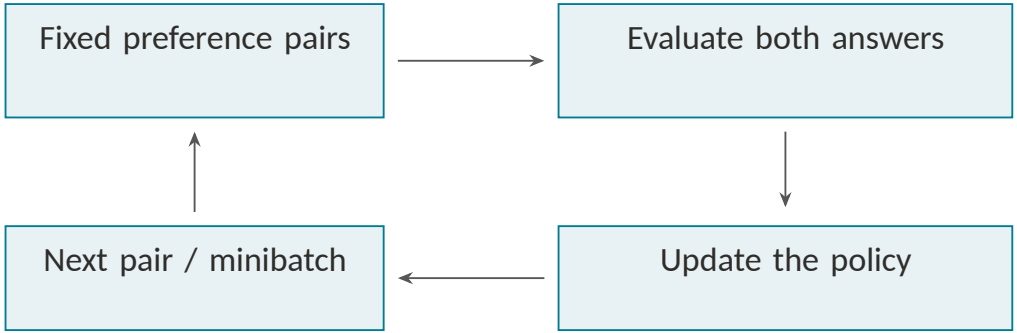
Negative coefficient

The DPO loss weights log-probability gradients computed on stored preference pairs.

Rafailov et al. (2023), DPO · Barak (2026), CS 2881, Lecture 3



The offline DPO loop



The reference stays frozen. New response generation is outside this training loop.

Rafailov et al. (2023), DPO



GRPO and offline DPO

	GRPO-style RL	Offline DPO
Responses	Fresh groups from policy	Stored preference pairs
Feedback	Score each attempt	Preferred / less preferred
Update weights	Group-relative scores	Pairwise policy margin
Generation in loop	Yes	No

Shao et al. (2024), DeepSeekMath · Rafailov et al. (2023), DPO



Where we are

1 RLHF and reward-based learning

2 From rewards to advantages

3 Reusing samples with PPO

4 Group-relative updates: GRPO

5 Learning directly from preferences

6 Evaluation and training data



Evaluation

Held-out tasks

Does the model solve new problems?

Matched inference budget

Match sample counts and token budgets.

Track held-out performance and errors as well as training reward or loss.



Choosing the training data

Prompt distribution	Cover the tasks and difficulty you care about.
Reliable feedback	RL: test the checker. DPO: check the rubric and preference labels.
Separate evaluation	Hold out problem families; check for duplicates and answer leakage.

Rafailov et al. (2023), DPO · DeepSeek-AI (2025), DeepSeek-R1



Adapting the task difficulty

All fail

0 0 0 0

No group contrast

Mixed outcomes

1 0 1 0

Relative signal

All pass

1 1 1 1

No group contrast

Adjust task difficulty as success rates change.

Monitor held-out quality, rollout tokens, truncation and output diversity.

Yu et al. (2025), DAPO



Summary

1. Use sampled responses or stored preference pairs.
2. Use feedback to construct relative weights.
3. Update the model through log probabilities.

