Proximal Policy Optimization (PPO)

From PPO to GRPO and GSPO

Lin Li

Alphatec Spine

Reinforcement Learning for Language Models

November 2, 2025

Overview

Objective: Understand the evolution of policy optimization algorithms from traditional PPO to modern variants used in LLM training.

Key Topics Covered:

- **Background:** Fundamental concepts of Reinforcement Learning
  - Value functions, Q-functions, and state-value functions
  - TD learning and policy gradients
- **Proximal Policy Optimization (PPO):** Core algorithm
  - Clipped surrogate objective
  - Importance sampling and advantage estimation
- **Group Relative Policy Optimization (GRPO):** Modern variant
  - Group-based advantage estimation
  - Avoiding value model training
- **Group Sequence Policy Optimization (GSPO):** Sequence-level optimization
  - Geometric mean of token probabilities
  - Practical considerations for LLMs

Table of Contents

1. Background
2. Proximal Policy Optimization
3. Group Relative Policy Optimization
4. Group Sequence Policy Optimization
5. Practical Considerations
6. Conclusion

Reinforcement Learning Fundamentals

Goal: Maximize expected cumulative reward

$$\mathcal{J}_{naive}(\theta) = \mathbb{E}_S[V_\pi(S)] \tag{1}$$

Key Functions:

Action-Value Function (Q-function):

$$Q_\pi(s_t, a_t) = \mathbb{E}_{S_{t+1}, A_{t+1}, \cdots}[U_t | S_t = s_t, A_t = a_t] \tag{2}$$

Optimal Q-function:

$$Q_*(s_t, a_t) = \max_\pi Q_\pi(s_t, a_t) \tag{3}$$

State-Value Function:

$$V_\pi(s_t) = \mathbb{E}_{A_t \sim \pi(\cdot|s_t)}[Q_\pi(s_t, A_t)] = \sum_{a \in \mathcal{A}} \pi(a|s_t) \cdot Q_\pi(s_t, a) \tag{4}$$

Training Q-Network

Training Data: Tuples of (s_t, a_t, r_t, s_{t+1})

ϵ**-greedy Strategy:**

$$a_t = \begin{cases} \arg\max_a Q(s_t, a; w) & \text{with probability } 1 - \epsilon \\ \text{random action in } \mathcal{A} & \text{with probability } \epsilon \end{cases} \tag{5}$$

Loss Function:

$$L(w) = \frac{1}{2}[Q(s_t, a_t; w) - \hat{y}_t]^2 \tag{6}$$

Gradient:

$$\nabla_w L(w) = [Q(s_t, a_t; w) - \hat{y}_t] \cdot \nabla_w Q(s_t, a_t; w) \tag{7}$$

Training Q-Function: Forward & Backward Pass

Forward Propagation:

$$\hat{q}_j = Q(s_j, a_j; w_{now}) \tag{8}$$

$$\hat{q}_{j+1} = \max_{a \in \mathcal{A}} Q(s_{j+1}, a; w_{now}) \tag{9}$$

TD Target & Error:

$$\hat{y}_j = r_j + \gamma \cdot \hat{q}_{j+1} \tag{10}$$

$$\delta_j = \hat{q}_j - \hat{y}_j \tag{11}$$

Backpropagation:

$$g_j = \nabla_w Q(s_j, a_j; w) \tag{12}$$

Proximal Policy Optimization (PPO)[1]

PPO Objective Function:

$$\mathcal{L}_{PPO} = \mathbb{E}_t \left[\min\left(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t\right)\right] - \beta D_{KL}[\pi_\theta || \pi_{\theta_{old}}] \tag{13}$$

where:

- $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ (importance sampling ratio)
- A_t is the advantage function: $A_t = Q(s_t, a_t) - V(s_t)$
- ϵ is the clipping parameter (typically 0.1 or 0.2)
- β controls KL divergence penalty

Key Insight: Clipping prevents large policy updates, ensuring stable training.

Group Relative Policy Optimization (GRPO) Algorithm

Motivation: Simplify PPO by removing the value network while maintaining stability.

GRPO Algorithm Steps:

1. For each prompt x, generate K responses (typically $K = 4$) with different random seeds
2. Use a reward model to compute score r_k for each response
3. Compute **group-relative advantage:**

$$A_k = \frac{r_k - \text{mean}(\{r_i\}_{i=1}^{K})}{\text{std}(\{r_i\}_{i=1}^{K})} \tag{14}$$

4. Optimize the policy using token-level clipped objective

Key Benefit: No need to train a separate value network!

GRPO Loss Function

GRPO Token-Level Objective:

$$\mathcal{L}_{GRPO} = \sum_{i=1}^{G} \frac{1}{|y_i|} \sum_{t=1}^{|y_i|} \min\left(w_{i,t}(\theta)A_i, \text{clip}(w_{i,t}(\theta), 1-\epsilon, 1+\epsilon)A_i\right) + \beta D_{KL} \tag{15}$$

where:

- $w_{i,t}(\theta) = \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{old}}(y_{i,t}|x, y_{i,<t})}$ (token-level importance ratio)
- $|y_i|$ is the number of tokens in response y_i
- G is the group size (number of responses per prompt)

Advantage Normalization:

$$A_i = \frac{r(x, y_i) - \text{mean}(\{r(x, y_j)\}_{j=1}^{G})}{\text{std}(\{r(x, y_j)\}_{j=1}^{G})} \tag{16}$$

Group Sequence Policy Optimization (GSPO)

Motivation: Treat the entire response as a single sequence rather than individual tokens.

GSPO Objective:

$$\mathcal{L}_{GSPO} = \sum_{i=1}^{G} \min\left(r_i(\theta)A_i, \text{clip}(r_i(\theta), 1-\epsilon, 1+\epsilon)A_i\right) + \beta D_{KL}[\pi_\theta || \pi_{\theta_{old}}] \tag{17}$$

where the sequence-level ratio is:

$$r_i(\theta) = \left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{old}}(y_i|x)}\right)^{\frac{1}{|y_i|}} \tag{18}$$

Key Difference: Uses geometric mean of token probabilities instead of arithmetic mean.

GSPO: Geometric Mean Ratio

Expanding the Sequence Ratio:

$$r_i(\theta) = \left(\frac{\pi_\theta(y_i|x)}{\pi_{\theta_{old}}(y_i|x)}\right)^{\frac{1}{|y_i|}} = \left(\prod_{t=1}^{|y_i|} \frac{\pi_\theta(y_{i,t}|x, y_{i,<t})}{\pi_{\theta_{old}}(y_{i,t}|x, y_{i,<t})}\right)^{\frac{1}{|y_i|}} \tag{19}$$

Interpretation:

- This is the **geometric mean** of token-level probability ratios
- More stable than arithmetic mean for long sequences
- Prevents one bad token from dominating the gradient
- Better length normalization

Getting Started with RLHF (Cost-Effectively)

Bootstrap Your RLHF Pipeline:

1. **Synthetic Data Generation**
   - Use GPT-4-turbo to generate 100K synthetic preferences
   - Cost: \$500–\$1,000
2. **Human Annotation** (for quality)
   - Hire annotators for 10K high-ambiguity pairs
   - Cost: \$5,000–\$20,000
3. **Train Reward Model**
   - Train a 1B parameter RM with LoRA
   - Cost: <\$1,000 on cloud GPUs
4. **Active Learning Loop**
   - Expand dataset only where RM uncertainty is high
   - Iteratively improve without massive annotation costs

GRPO vs PPO Comparison

PPO	GRPO
Requires value network	No value network
Actor-Critic architecture	Policy-only architecture
Individual advantage estimation	Group-based advantage
More stable (value baseline)	Simpler (fewer components)
Higher computational cost	Lower computational cost

Advantages of GRPO:

- **Unsupervised learning** (no manual labeling during training)
- **No value model** (saves training cost and complexity)
- **Simpler architecture** (easier to implement and debug)
- **Competitive performance** with PPO on many tasks

PPO Mathematical Formulation

Standard PPO Objective:

$$\max_{\theta} \mathbb{E}_{q\sim P(Q)} \left[\frac{\pi_\theta(o|q)}{\pi_{\theta_{old}}(o|q)} A_{\theta_{old}}(q,o) \right] - \beta \mathbb{E}_{q\sim P(Q)}[D_{KL}[\pi_\theta(\cdot|q)||\pi_{\theta_{old}}(\cdot|q)]] \tag{20}$$

where:

- q is the query (input prompt)
- o is the output (model response)
- $A_{\theta_{old}}(q,o)$ is the advantage estimated by the value network
- β controls the KL divergence penalty

Challenge: Need to train and maintain a separate value network $V_\theta(q)$ to estimate advantages.

Summary

Evolution of Policy Optimization:

1. **PPO (2017):** Gold standard for stable RL training
    - Clipped objective prevents destructive updates
    - Requires value network for advantage estimation
2. **GRPO (Recent):** Simplified variant for LLMs
    - Group-based advantage removes need for value network
    - Token-level optimization with length normalization
3. **GSPO:** Sequence-level alternative
    - Geometric mean of token ratios
    - Better for long-sequence generation

Key Takeaway: Modern LLM training increasingly favors simpler algorithms (like GRPO) that reduce computational overhead while maintaining performance.

References I

[1] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: 1707.06347 [cs.LG]. URL: https://arxiv.org/abs/1707.06347.