# Lecture on Proximal Policy Optimization

**Introduction**

In 2017, OpenAI introduced a new family of policy gradient methods for reinforcement learning called **Proximal Policy Optimization (PPO)**.

This algorithm quickly became one of the most widely used RL methods due to its balance of simplicity, stability, and efficiency.

PPO optimizes a **surrogate objective** with **clipped probability ratios**, ensuring that policy updates remain stable without requiring complex second-order optimization. Unlike standard policy gradient methods, which update policies once per data sample, PPO allows multiple **minibatch updates**, improving sample efficiency. Empirical results have shown that PPO outperforms other policy gradient methods in tasks like robotic control and Atari games. It is also the foundation of widely used AI models like **ChatGPT**.

To fully understand why PPO was introduced, we must first look at its predecessor: **Trust Region Policy Optimization (TRPO).**

**TRPO – The Predecessor to PPO**

Before PPO, Trust Region Policy Optimization (TRPO) was a leading approach for improving policy gradient methods. TRPO was designed to ensure stable updates by preventing the new policy from deviating too much from the previous one.

**How it works**

* In standard policy gradient methods, large updates can cause drastic policy shifts, leading to instability.
* TRPO solves this by introducing a **trust region constraint**, which limits how much the policy can change in each update.
* It does this using **Kullback-Leibler (KL) divergence**, a measure of how different two probability distributions are.

**Mathematical Equation -** on slide 4

**Drawbacks of TRPO**

1. **Complexity** – It requires **second-order optimization** (computing Hessians), which is computationally expensive.
2. **Hard to implement** – The KL divergence constraint requires a **constrained optimization solver**, making it harder to apply.
3. **Inefficient Data Use** – TRPO processes data once per batch, meaning it doesn't fully utilize available training samples.

These limitations led to the development of **PPO**, which retains the benefits of TRPO while being simpler and more efficient.

**Transition to PPO**

PPO builds upon TRPO by **removing the need for complex constraints** and using **clipping** to stabilize updates. Instead of explicitly enforcing a trust region, PPO introduces a modified objective function that **penalizes excessive policy updates** while allowing multiple minibatch updates for better sample efficiency.

In the next section, we will dive deeper into **how PPO works** and why it became the standard for modern RL applications.

**Understanding Policy Gradients:**

Policy Gradient methods are a fundamental approach in reinforcement learning. Instead of learning a value function to determine the best action, they **directly optimize the policy** π(a∣s) by adjusting the parameters θ to maximize expected rewards.

**Value Functions:**

While policy gradients **tell us how to act**, value functions **help evaluate** those actions efficiently.

**Types of Value Functions**

**1. State Value Function V(s)**

* It measures **how good a state is on average**.

**2. Action-Value Function Q(s,a)**

* It measures **how good a specific action is from a given state**.

**Why is the Value Function Important?**

* The **value function** is included to help compute the **advantage function** A(t)A(t)A(t).
* It estimates **how good an action is on average**, providing a **baseline** for comparing different actions effectively.

**Actor-Critic Method**

* When **policy gradients and value functions** are used **together**, they form the **Actor-Critic method**, a key **optimization technique** in Reinforcement Learning (RL).
* **Actor:** The agent that takes actions based on the policy.
* **Critic:** The value function, which **evaluates how good** the agent’s actions are.
* This combination helps improve learning efficiency and stability in RL algorithms.

**Advantage Estimation & Generalized Advantage Estimation (GAE)**

From Slide 8 and 9

**Importance Sampling:**

In real-life applications, data can be unstable and limited, meaning even small variance or noise can cause significant fluctuations in learning. This issue is called variance, and giving too much importance to a particular data sample can introduce bias.

Why is Importance Sampling Needed?

* In traditional Reinforcement Learning (RL), the agent collects data using an old policy and then updates the policy based on that data.
* However, when the policy is updated, the distribution of actions shifts, meaning the new policy may take very different actions compared to the old policy.
* This creates a challenge: How can we still learn effectively from older data when our policy has changed?

How Importance Sampling Helps

Importance Sampling (IS) is a technique that allows us to correct for this shift in data distribution. It helps us reuse old data while still making accurate updates by adjusting for the differences in probability distributions.

**PPO Clipping:**

While Importance Sampling helps reuse old data efficiently, it introduces high variance and can make policy updates unstable. If the importance sampling ratio

becomes too large, it can lead to overly aggressive updates, causing the policy to change too drastically.

To prevent extreme updates while still allowing policy improvement, Proximal Policy Optimization (PPO) introduces a clipped objective function, which restricts the importance ratio within a safe range.

The minimum function ensures that if rt(θ) moves too far from 1, the update is clipped, preventing drastic changes.