The City College
of New York

CSC 59866-E: Senior Project I
AI Agents for Decision Making in the Real World

By Saptarashmi Bandyopadhyay
Email: sbandyopadhyay@ccny.cuny.edu, sbandyopadhyay@gc.cuny.edu
Assistant Professor of Computer Science
City College of New York and Graduate Center at the City University of New York

Systems: Latency and Bandwidth balancing

Today’s Agenda

Recap: The Role of Reinforcement Learning in NLP

- How John Schulman tricked NLP researchers into being RL researchers

The Physical Limits of Agents

- Latency, Bandwidth, Throughput

Architecting for System Constraints

- Cloud vs. Edge Architectures

Engineering Solutions

- Compression, Quantization, Offloading, Async. Execution

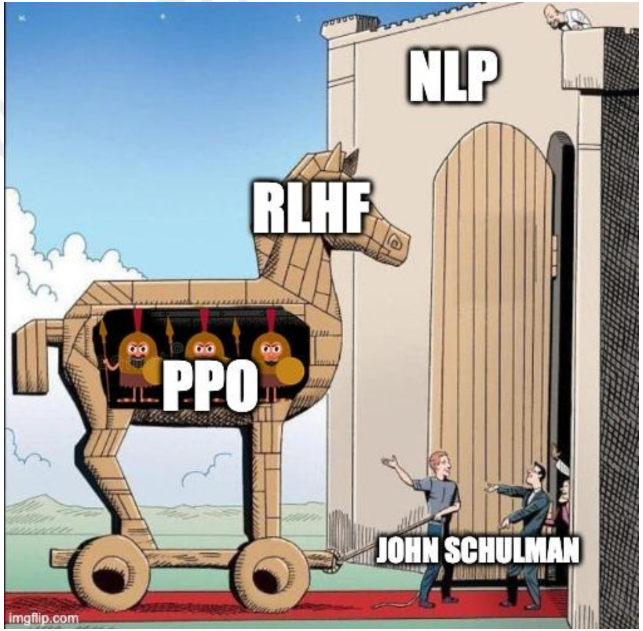
Recap: The Role of Reinforcement Learning in NLP

PPO in NLP

For years, NLP was purely supervised learning. You just trained a model to predict the next word in a massive dataset.

To make LLMs act like helpful *agents* (like ChatGPT), researchers introduced **Reinforcement Learning from Human Feedback (RLHF)** to align them with human preferences.

To actually optimize that human reward, they needed a robust RL algorithm. They smuggled in **Proximal Policy Optimization (PPO)**!

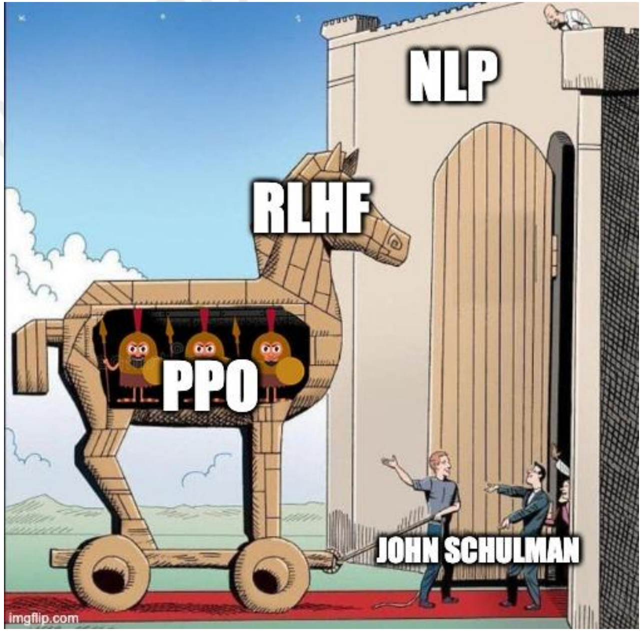


PPO in NLP

If your group is working on **LLM Agents**, you are not just doing text prediction. You are running an RL loop.

The LLM generates text (Action), the human or system provides feedback (Reward), and the model updates its weights using PPO to maximize that reward.

Takeaway: The line between a "Language Model" and an "RL Agent" no longer exists!



The Physical Limits of Agents

Definitions: Latency & Bandwidth

Latency: The time it takes for a single piece of data to travel from Point A to Point B.

- *Analogy:* The speed limit of the vehicles on the highway.
- *Unit:* Milliseconds (ms).

Bandwidth: The maximum rate of data transfer across a given path.

- *Analogy:* The number of lanes on the highway.
- *Unit:* Megabits per second (Mbps) or Gigabytes per second (GB/s).

Throughput: The *actual* rate of successful data delivery over the network (often lower than bandwidth due to packet loss and overhead).

Why Should we Care About Latency & Bandwidth?

A reinforcement learning agent playing chess can afford to think for 5 seconds. An agent driving a Connected Autonomous Vehicle (CAV) cannot.

The Latency Budget:

- Human reaction time to visual stimulus: ~250ms.
- Autonomous Emergency Braking budget: < 100ms.
- AR/VR motion-to-photon limit: < 20ms (to prevent motion sickness).

The Bandwidth Wall: A single 4K camera on a robotic agent generates ~4 Gbps of raw data. You cannot stream this to a centralized server over standard wireless networks.

Architecting for System Constraints

Cloud Computing vs. Edge Computing

- **The Cloud (Centralized):**
  - *Pros:* Infinite compute, massive GPUs, holds the biggest foundation models (e.g., 70B+ parameter LLMs).
  - *Cons:* Requires high bandwidth to send inputs (video/audio). Highly susceptible to network latency and connection drops.
- **The Edge (Decentralized):**
  - *Pros:* Zero network latency. Works offline. Data remains private.
  - *Cons:* Severely constrained compute and battery life. Can only run small models.

The Communication Bottleneck in Distributed Training

Recall: Distributed agents must synchronize their gradients or model weights.

The Problem: Modern AI models are massive. Syncing a 100GB model across 100 GPUs every few seconds requires astronomical bandwidth (NVLink/InfiniBand).

The Result: If bandwidth is too low, the GPUs spend 90% of their time sitting idle waiting for data to arrive over the network, wasting time and energy.

Example: Autonomous Braking

The Scenario: A Connected Autonomous Vehicle (CAV) is traveling at **30 meters/second** (approx. 67 mph). An obstacle suddenly appears. The car's camera captures an **8 Megabyte (MB)** image of the obstacle. The agent must process this image to decide whether to brake.

You have two architectural choices. Which one stops the car sooner?

Option A: The Cloud Agent

- **Network Bandwidth:** 200 Mbps
- **Network Routing Latency (Round Trip):** 40 ms
- **Cloud Inference Time (Large Model):** 10 ms.

Option B: The Edge Agent (Local)

- **Network Transfer:** 0 ms
- **Edge GPU Inference Time (Quantized Small Model):** 250 ms.

Example: Autonomous Braking (Cloud Option)

Step 0: Convert Units

- Image Size: 8 Megabytes x 8 = **64 Megabits (Mb)**.

Step 1: Calculate Option A (The Cloud)

- **Transmission Time:** $Size/Bandwidth = 64\ \text{Mb}/200\ \text{Mbps} = 0.32\ \text{seconds} = \mathbf{320}\ \text{ms}$
- **Total Cloud Latency:** Transmission (320) + Routing Latency (40) + Inference (10) = $\mathbf{370}\ \text{ms}$
- **Distance Traveled:** $30\ \text{m/s} \times 0.370\ \text{s} = \mathbf{11.1}\ \text{meters}$

Example: Autonomous Braking (Edge Option)

Step 0: Convert Units

- Image Size: 8 Megabytes x 8 = **64 Megabits (Mb).**

Step 2: Calculate Option B (The Edge)

- **Total Edge Latency:** Inference only = $\mathbf{250}\ \mathrm{ms}$
- **Distance Traveled:** $30\ \mathrm{m/s} \times 0.250\ \mathrm{s} = \mathbf{7.5}\ \mathrm{meters}$

Example: Autonomous Braking (Results)

Despite having a massively superior GPU (10ms vs 250ms), the **Cloud Agent** causes the car to travel **3.6 meters further** before reacting, simply because the bandwidth bottleneck (uploading 64Mb) dominated the timeline.

Conclusion: For latency-critical physical systems, raw compute power cannot beat local data processing without hyper-fast communication!

Engineering Solutions

Strategies for the Real World

- **Model Compression & Quantization:**
  - Converting 32-bit floats to 8-bit or 4-bit integers (INT8/INT4).
  - *Effect:* Shrinks the model size by 8x. Reduces bandwidth needed to transmit weights and allows the model to run on edge devices, slashing latency.
- **Activation Compression / Feature Offloading:**
  - Instead of sending raw 4K video to the cloud, the edge device runs the *first few layers* of a vision model to extract compressed features (embeddings).
  - It only sends the lightweight embeddings to the cloud for the final complex decision.
- **Asynchronous Execution:**
  - As discussed last week, letting agents act on "stale" information rather than waiting for a synchronous network round-trip.

Example Research Problems

Adaptive Resource Allocation: How do we dynamically route LLM tokens between the cloud and edge depending on current network conditions?

State Synchronization Over Slow Networks: How do we deal with extreme bandwidth limitations and how do decentralized agents maintain a shared worldview?

Energy-Efficient Protocols: Processing locally saves network bandwidth but burns battery. How do we optimize this trade-off?

Summary & Next Steps

Key Takeaway: The "best" AI model is useless if it takes too long to make a decision or requires more data throughput than the physical environment allows.

Next Class (Feb 23): Decentralized Execution and Centralized Training (CTDE) in Depth.

Questions?