Real-time State Synchronization Solutions for Decentralized AI Agents Over Slow Networks

-Progress Update – Baseline, Early Results & Next Steps

Team: Brandon Bedoya & Haoliang Zhang

Project Goal

Goal:
Make AI training work smoothly when devices have slow or inconsistent network connections.

Why:
Large models send huge amounts of data (states, activations, updates).

Slow network = **Slow training**

Our idea:
Reduce how much data gets sent each step by compressing or simplifying the model updates.

Baseline (What We Have Working Now)

Our **baseline** is simple:

- Use GPT-2 small
- Load a text dataset
- Run a normal fine-tuning step
- Measure:
    - **Loss**
    - **Training time per step**

We are creating our own baseline because:

1. The NeurIPS papers focus on massive multi-GPU clusters we don't have access to.
2. We need a version that is small, reproducible, and runnable on Colab so we can measure our improvements.

-Having a clean baseline lets us measure communication savings later

Our Contribution

We take a small AI model (like GPT-2 small) and train it normally

Then we change ONE thing:

Instead of sending the full activations every step, we only send the difference from last time (called an **activation delta**).

This makes communication WAY smaller, which means:

- faster training
- less bandwidth wasted
- could work on laptops, edge devices, or slow networks

- It's like instead of texting someone your entire essay every time you make a small change…
you just send them what you changed.

Expected Result: Lower communication cost, similar accuracy.

Why It Matters

Not everyone has 8 GPUs, fiber internet, and a data center under their bed.

This project helps:

- universities
- small research labs
- companies with weak infrastructure
- edge devices (drones, phones, robots)

train AI models without needing big infrastructure.

Evaluation Plan

We will compare **Baseline vs. Compressed Training** using:

- **Training time**
- **Estimated communication volume (bytes sent)**
- **Loss / accuracy**
- **Behavior under simulated slow network**

Success = same accuracy, much lower communication.

Challenges

- Simulating distributed training without multi-GPU hardware
- Ensuring compression doesn’t hurt accuracy (Activation deltas may distort information too much)
- Measuring communication cost inside Colab runtime (Colab doesn't show network usage, so we calculate it ourselves.)

How we’re fixing it:

Start small → test piece-by-piece → scale up.

-First get a clean baseline, then test compression on tiny pieces, make sure accuracy holds up, and gradually scale the method.

CODE DEMO