# Spring 2025: ECE 759 Final Project Proposal

**Name**: Sumukha Madodi

**Email**: madodi@wisc.edu

**Home department**: Electrical and Computer Engineering

**Status**: MS Student

**Name of your teammate (everyone needs to submit the same proposal):**
Shashank M Bharadwaj
Shraddha Singh
Sahana Diwakar

**Project Title**: Overlapping Gradient Computation and All-Reduce Communication for Scalable Deep Learning Training

https://github.com/Sumukha-M/proj_759

**Problem statement:** This project aims to accelerate distributed deep learning training by reducing communication overhead. Specifically, we will implement a system that overlaps gradient computation with communication using CUDA and OpenMP. Gradients for each layer will be computed and communicated asynchronously to improve GPU utilization and training throughput.

**Motivation/Rationale**: In modern deep learning workloads, communication of gradients (via AllReduce) during distributed training can become a bottleneck, especially at scale. By overlapping computation and communication, we aim to improve efficiency and reduce idle GPU time. This approach mimics optimizations seen in frameworks like DeepSpeed and Megatron-LM but is built from scratch for educational and analytical purposes.

**Explain how you contemplate going about it:**

- Computing Model: Combination of GPU (CUDA) and CPU (OpenMP) parallel computing.

- Libraries/Tools: CUDA for matrix computation and gradient computation, OpenMP for multithreaded coordination, MPI or NCCL for inter-GPU AllReduce communication.

- Approach:
  1: CUDA-only baseline forward/backward pass.
  2: CPU-side OpenMP threading to simulate overlapping.
  3: Integrate MPI with OpenMP and CUDA to enable gradient aggregation.
  4: Optimize with CUDA streams and NCCL-based true GPU-to-GPU AllReduce.

**ECE 759 aspects the proposed work draws on:**

- CUDA kernel programming and memory management
- OpenMP-based CPU-side multithreading
- Distributed memory programming using MPI.
- Performance benchmarking and profiling (e.g., Nsight Systems, timers)
- Scalability and parallel efficiency analysis

**Deliverables**:

- A working multi-phase codebase demonstrating overlapping training logic
- Benchmarks comparing training with and without overlap
- Presentation with results, profiling screenshots, and graphs
- Final report summarizing methodology, results, and insights

**How you will demonstrate what you accomplished:**

- Live demo of training loop with/without communication overlap
- Runtime plots showing reduced idle times and improved throughput
- Timing results and breakdowns from profiling tools
- GitHub commit history and documentation for reproducibility
- Potential continuation: integrating into a real DL framework (e.g., PyTorch C++ frontend)

**Other remarks:** We plan to start with a minimal example for better control and analysis, then scale to larger models and more devices. We aim to focus equally on performance and clean system design. All development will be version-controlled and structured for future extensibility.