Project Proposal: Optimizing LLM Inference on Consumer Hardware

Group Member: - Yash Patel (yp2378)

Project Summary: This project focuses on developing an optimized large language model (LLM) inference engine from scratch, designed for single-batch inference on consumer-grade CPU and GPU hardware. The objective is to create a lightweight, high-performance solution in C++ and CUDA, targeting the Mistral-7B-Instruct-v0.2 model for prompt completion tasks on modern hardware. By implementing techniques such as multithreading, weight quantization, and GPU acceleration without relying on external libraries, the project aims to achieve competitive token throughput—starting with a baseline of 4-5 tok/s on CPU, improving to 8-9 tok/s with optimizations, and exceeding 60 tok/s on GPU. Current progress can be tracked at https://github.com/yashp5/inference-engine. This effort will enhance my understanding of LLM architectures, memory bottlenecks, and hardware-specific optimizations while delivering a practical tool for local inference.

Project Plan: The project will unfold over eight weeks. Week 1 will involve researching transformer architectures, inference mechanics, and Mistral-7B specifics, followed by Week 2 for setting up the development environment and acquiring model weights. Weeks 3-4 will focus on building a baseline CPU inference engine with multithreading and FP32 weights, aiming for 4-5 tok/s. In Weeks 5-6, I will enhance the CPU version with FP16 quantization and SIMD vectorization to reach 8-9 tok/s, while initiating a basic GPU implementation in CUDA. Weeks 7-8 will refine the GPU version with optimized matrix multiplications and prefetching, targeting over 60 tok/s, concluding with testing and documentation. Weekly milestones will guide progress, with adjustments as needed based on optimization challenges.

Individual Responsibilities: As the sole team member, Yash Patel (yp2378) will handle all aspects of the project. I will develop the CPU inference engine, implementing multithreading with OpenMP and SIMD vectorization using AVX2 and F16C extensions, leveraging my skills in parallel programming. I will also design and optimize the GPU inference engine in CUDA, focusing on kernel development for matrix multiplications and prefetching, building on my experience with GPU programming. Additionally, I will manage environment setup, model weight conversion (e.g., to safetensors), benchmarking against existing tools like llama.cpp, and documentation, ensuring a cohesive and well-tested final product. Progress will be updated regularly at https://github.com/yashp5/inference-engine.