Mixed-Precision Inference Optimization: By utilizing Tensor Cores on a GPU

Amir Mahdi Mirtajadini

Project Overview

Neural network inference is a compute-intensive task traditionally performed using single-precision floating-point arithmetic. However, modern GPU architectures, such as the NVIDIA Ada Lovelace, introduce specialized hardware units known as Tensor Cores designed to accelerate matrix operations using lower precision formats (FP16, BF16) while maintaining sufficient accuracy via mixed-precision accumulation. This project aims to implement and analyze the performance trade-offs of mixed-precision inference. The primary objective is to quantify the speedup and efficiency gains provided by Tensor Cores compared to general-purpose CUDA cores on the NVIDIA RTX 4060 GPU.

Workload and Dataset

To satisfy the project requirement for a simple inference workload while ensuring performance differences are clearly measurable, this project will implement a two-layered MLP applied to image dataset (STL-10) provided by Stanford University.

2.1 Input Data: STL-10 Dataset

Unlike standard benchmarks e.g. CIFAR-10 which operate on low-resolution thumbnails, this project will utilize the STL-10 dataset.

- **Input Resolution:** 96×96 pixels (Colored).
- **Feature Vector:** Each image is flattened into a vector of size **27,648** ($96 \times 96 \times 3$).
- **Benefit:** The high dimensionality ensures that the arithmetic intensity of the first layer is sufficiently high to saturate the GPU, making the speedup from Tensor Cores distinct and measurable.

2.2 Target Model: Two-Layer MLP

The inference workload consists of a Two-Layer MLP with a specific wide configuration to stress-test the hardware.

1. **Layer 1:**
   - **Dimensions:** Input $(27,648) \times$ Hidden (8192).
   - **Characteristics:** This layer involves a massive GEMM operation ($\approx$ 226 million parameters). It is designed to be **compute-bound**, where Tensor Cores are expected to provide maximum throughput gains.
2. **Activation:** ReLU non-linearity applied to the hidden state.

3. **Layer 2:**
    - **Dimensions:** Hidden (8192) $\times$ Output (10).
    - **Characteristics:** This layer reduces the 8192 features down to 10 class logits. It is designed to be **memory-bandwidth bound**, providing a contrast point for analysis where Tensor Core utilization may yield diminishing returns.

Execution Strategy

To ensure a fair comparison between hardware units, the implementation is divided into two distinct phases, with each phase handling the massive input dimensions defined in the workload.

3.1 Phase 1: CUDA Core Implementation

I will develop a custom CUDA kernel that explicitly runs on the Streaming Multiprocessors' FMA units.

- **Memory Coalescing:** Ensuring global memory accesses are aligned to 128-byte transaction lines, which is critical given the large footprint of the STL-10 feature vectors.
- **Shared Memory Tiling:** Implementing block-tiling to maximize data reuse. Special attention will be paid to the large K dimension (27,648) to ensure the kernel maintains high occupancy.
- **Precision:** Both FP32 and FP16 versions will be written to isolate the speedup gained purely from data type reduction versus hardware acceleration.

3.2 Phase 2: Tensor Core Implementation

I will utilize **cuBLAS** to offload the computation to the RTX 4060's 4th Gen Tensor Cores.

- **Mixed Precision Strategy:** Inputs will be cast to **Bfloat16** (BF16) or FP16, while accumulation occurs in FP32.
- **Layer-Specific Tuning:** I will utilize the `cublasGemmEx` API to execute the specific shapes of Layer 1 and Layer 2, allowing for a direct comparison of how the library optimizes for compute-bound vs. memory-bound shapes.

Experimental Setup

All experiments will be conducted on the following system:

Component	Specification
CPU	Intel Core i7 13620H
GPU	NVIDIA GeForce RTX 4060 (Ada Lovelace)
RAM	32GB DDR4
OS	Windows 11 Home
Profilers	NVIDIA Nsight Compute, Nsight Systems

Evaluation Methodology and Profiling

The core of this project is not just implementation, but detailed profiling using **NVIDIA Nsight Compute** and **Nsight Systems**.

5.1 Key Performance Indicators

- **SOL Analysis:** I will measure the achieved compute throughput against the theoretical peak of the RTX 4060.
  - *Compute SOL %:* How close are we to the peak FP32/FP16 TFLOPS?
  - *Memory SOL %:* How close are we to the peak DRAM bandwidth?
- **Latency Distribution:** Breaking down the total inference time into:
  - Host-to-Device (H2D) transfer time.
  - Kernel execution time (Compute).
  - Device-to-Host (D2H) transfer time.
- **Layer-wise Efficiency Contrast:** I will quantify the speedup factor separately for the **Compute-Heavy Layer 1** and the **Memory-Heavy Layer 2**. This is crucial to demonstrate that Tensor Core benefits are non-uniform and depend heavily on the arithmetic intensity of the layer.
- **Numerical Accuracy:** A standard Mean Squared Error (MSE) comparison will be conducted:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_{fp32} - Y_{mixed})^2$$

  This will ensure that the performance gains from reduced precision do not come at the cost of unacceptable inference degradation.

Setup Suitability

The **RTX 4060** is an ideal candidate for this study as it supports the latest **Bfloat16** format, allowing for a comparative study between FP16 (standard half precision) and BF16 (high dynamic range half precision), a feature not available on older Pascal or Turing architectures.

Expected outcomes

This proposal sets a clear path to understanding the cost vs. benefit of mixed-precision inference. By utilizing the huge STL-10 dataset and a focused Two-Layer MLP, this project goes beyond a simple speedup number. It will provide a definitive guide on when to utilize Tensor Cores and when their benefits are limited by memory bandwidth, offering a comprehensive view of ML inference on consumer-grade hardware.

It is expected that the Tensor Core implementation will show significant speedups over the FP32 baseline, provided the matrix dimensions are large enough to saturate the GPU. The report will document the crossover point where the overhead of data casting and library calls is outweighed by the raw compute throughput of the Tensor Cores.