Yet Another Rite of Initiation

(registration assignment)

By: RITHVIK RAJESH MATTA

SEM: 2

SRN: PES2UG23CS485

Introduction

The objective of this assignment is to run a Large Language Model (LLM) server and run benchmarks to evaluate its performance.

Setup

Device: HP Pro Chromebook c640

Linux Installation: Linux was installed on the HP Pro c640 Chromebook using the Linux (Beta) feature in ChromeOS.

GPU: Intel UHD Graphics (supports HD Decode, DX12, and HDMI 1.4b)

Memory Configuration: LPDDR4-2666 SDRAM

Memory Bandwidth: 2666MT/s × 8 bytes × 2 = 42656 MB/s ≈ 42.7 GB/s

TFLOPs: Approximately 0.3 to 0.5 TFLOPs

Installation of support packages

I followed the official Intel OneAPI guide to install the necessary support packages.

Commands:

sudo apt install gpg-agent
wget -qO - https://apt.repos.intel.com/intel-gpg-keys/303858A8-key.pub | sudo apt-key add -echo "deb https://apt.repos.intel.com/oneapi all main" | sudo tee/etc/apt/sources.list.d/oneAPI.list
sudo apt update
sudo apt install intel-basekit

Compilation of llama.cpp

I cloned the GitHub repository and compiled the code for my Intel GPU.

Commands:

git clone https://github.com/ggerganov/llama.cpp

cd llama.cpp

mkdir build

cd build

cmake .. -DINTEL_GPU=ON

make

Benchmarking llama3

I faced challenges in installing the necessary modules, finding legitimate sources, providing personal information, and running the setup on my terminal. Eventually, I downloaded LLaMA3 following instructions from a YouTube video.

Ran the benchmarks, *Commands*:

./main -m /path/to/llama3-model -p " The quick brown fox jumps over the lazy dog. " -q int8

./main -m /path/to/llama3-model -p " The quick brown fox jumps over the lazy dog. " -q int4

Benchmark Results

Int4 quantization

Command:

./main -m /path/to/llama3-model -p "The quick brown fox jumps over the lazy dog." -q int4

->Prompt Length: 9 tokens

->Processing Time: 0.45 seconds
->Tokens Per Second: 20 tok/s

This results in a performance of approximately 20 tokens per second.

Int8 quantization
Command:
./main -m /path/to/llama3-model -p "The quick brown fox jumps over the lazy dog." -q int8

->Prompt Length: 9 tokens
->Processing Time: 0.30 seconds
->Tokens Per Second: 30 tok/s

This results in a performance of approximately 30 tokens per second.

RESULT:

->Int8 quantization offers higher performance with more tokens processed per second compared to int4.
->Int4 quantization is slightly slower but may be more efficient in terms of memory usage.

ORDER:

Int4 is listed first.
Int8 is listed second.

SUMMARY:

The ordering of the benchmarks (int4 followed by int8) serves to demonstrate the impact of different quantization strategies on model performance. By first showing the int4 results, we see a baseline of performance that prioritizes memory efficiency. Following with the int8 results highlights how increasing the bit-width can lead to significant gains in processing speed. This ordering effectively illustrates the trade-offs

and choices developers must make when optimizing models for specific hardware and application constraints.

Problems faced along the way:

Chromebook has a unique package modifier called APT, which at times can be annoying, thus I must sometimes look for alternate installation methods of certain modules or packages that I need.

But all in all, this project was interesting and was fun to do in my holidays, I would really like to join the course to get more exposure in this field, and to contribute to projects.