Evaluating the Speeds of Local LLMs by Isaac Restrick

Abstract

The goal of this project was to test the inference speeds of a few open source large language models that were small enough to run on a MacBook Pro. Specifically, I set up a basic client server architecture that I then used to evaluate the time-to-first-token and tokens per second of various quantizations of the Llama-2 7B model. Using a python wrapper of the llama.cpp library, I tested the speeds of models quantized through 2-bit, 3-bit, 4-bit, and 5-bit quantization. The results of this experiment were as expected - that as model size increased, time-to-first-token increased, and tokens per second decreased.

Introduction

The larger problem this project addresses is that while the most powerful LLMs (GPT-3.5, GPT-4, Claude 2, etc) are extremely useful, they are both too large to run locally and as they are not open source they can't be run independently at all. It is important that models that can be run locally exist so that the public may better study and understand the frontiers of AI. In this paper, I set up quantizations of the LLama-2 7B model locally in order to have the general experience of playing around with an LLM on my own computer, and to measure inference speed as well to have a better feel for the effects of quantization and model size.

Background

Roughly a year ago, ChatGPT was released and gave many including myself the first experience of talking to a seemingly intelligent model. Large language models such as ChatGPT (by OpenAI) and LLama 2 (by Meta) are powered by decoder only transformer architectures, where they are trained to predict the next token on internet-scale data, before further tuning is performed to align them. The current state of the art is the GPT-4-Turbo model which has a 128k token context window and also accepts image inputs.

Methodology / Approach

This project involved two key steps: 1) setting up an environment to run LLama-2 locally including a client, server, and downloading quantized models and 2) using this environment to quantitatively evaluate inference speeds of models of varied sizes. To be more specific, I evaluated the time-to-first-token (time from receiving prompt to outputting the first token) and tokens per second (completion tokens / total time taken) across four models (linked in references) quantized by "TheBloke" on huggingface.co: llama-2-7b-chat.Q2_K.gguf, llama-2-7b-chat.Q3_K_M.gguf, llama-2-7b-chat.Q4_K_M.gguf, llama-2-7b-chat.Q5_K_M.gguf. My prompt consisted of the message "hi" wrapped in the prompting format used by these models (includes adding user:, assistant: to messages) and setting a maximum of 32 tokens for the output. I re-ran this 5 times per model and averaged results.

Results

The results are attached at the end of the paper (Figure 1 and Figure 2). The results demonstrate that time to first token increases with model size, and tokens per second decrease with model size. These results are sensible and the project worked - the larger models are more computationally expensive and take longer.

Discussion

This project helps the field in that it is a data point for how quantizing models can lower inference speeds of large language models on a Macbook Pro. An important tradeoff to remember is that while increasing inference speed, quantization can impact model performance. With more time and resources, I would evaluate how model performance evolves / degrades as

the model is quantized with less and less precision. This project is not any better than other methods for evaluating inference speed but provides a data point for how these methods run in the field.

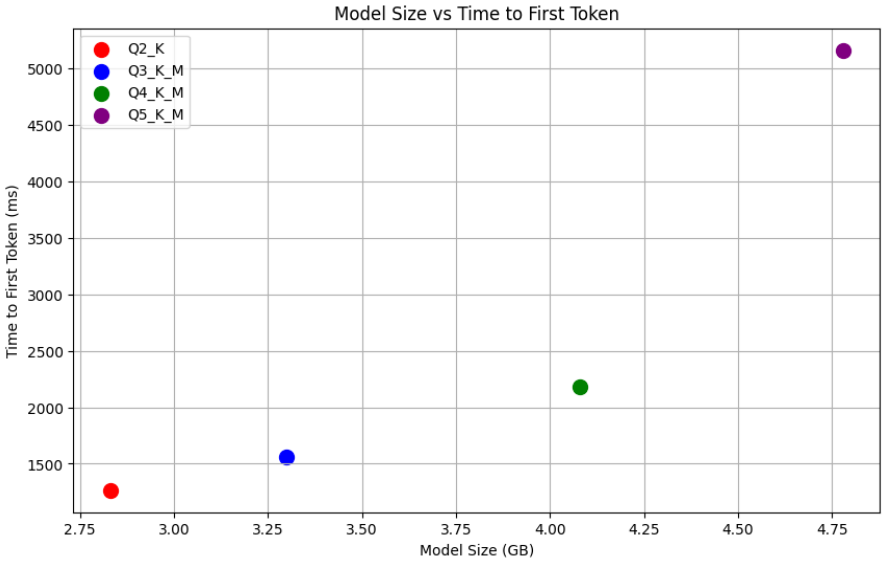

Figure 1: Model size vs time to first token

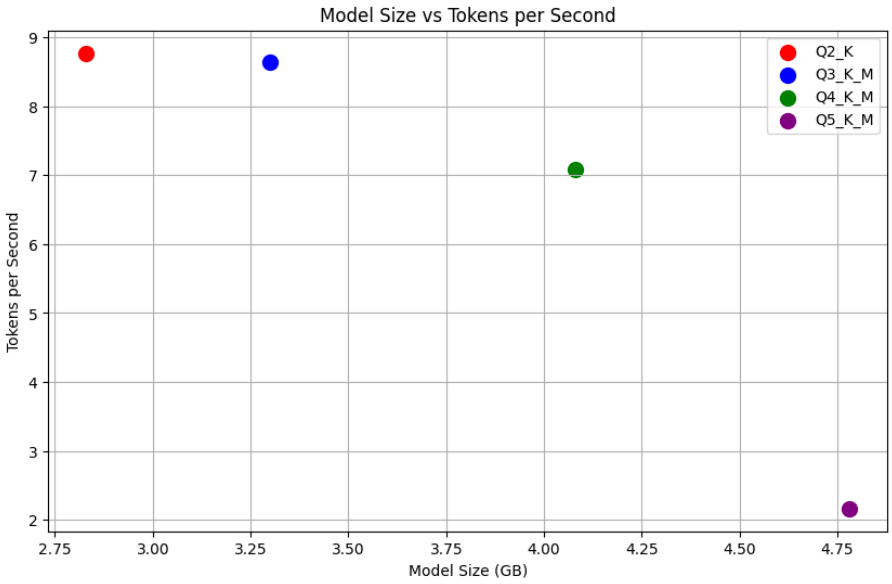

Figure 2: Model size vs Tokens per second

References:

Meta AI. “Introducing Llama 2.” Meta AI, https://ai.meta.com/llama/

Betlen, Abet. “llama-cpp-python.” Github, https://github.com/abetlen/llama-cpp-python

TheBloke. “Llama-2-7B-Chat-GGUF.” Hugging Face, https://huggingface.co/TheBloke/Llama-2-7B-Chat-GGUF

OpenAI. "New Models and Developer Products Announced at DevDay." OpenAI Blog, 6 November 2023, https://openai.com/blog/new-models-and-developer-products-announced-at-devday