Methods of Reducing Computational Requirements for Large Language Models

Oleksandr Kononov [*]

[*] *South East Technological University, Cork Road, Waterford, Ireland (e-mail: 20071032@mail.wit.ie).*

Abstract: TODO

Keywords: Artificial intelligence, Neural networks,

1. INTRODUCTION

1.1 Background

A Large Language Model (LLM) is neural network model which is capable at working with natural language tasks such as text generation, text summarization, translation, and more. Vaswani et al. (2017) proposed the novel Transformer architecture which has revolutionized the field by introducing more efficient multi-headed self-attention mechanism compared to Recurrent Neural Networks that came before. With the improvements in training times, better parallelization on GPUs and overall quality of output. Following this, OpenAI have used this novel Transformer architecture to design and develop their Generative Pre-Trained Transformer (GPT) LLMs the following years. In particular, the release of GPT-3 in 2020, has sparked a global interest in the continued development of LLMs from various companies such as Meta with LLaMa, Google with Gemma, Antropic with Claude, etc.

Touvron et al. (2023) developed a series of open-weight LLMs called LLaMa, ranging from 7B parameters to 65B parameters. Their paper demonstrates, using various benchmark tests, we can draw a correlation between the increase in LLMs parameters and the scores that it can achieve on various benchmark tests such as HellaSwag (Zellers et al. (2019)), WinoGrande (Sakaguchi et al. (2019)), ARC (Clark et al. (2018)) and OpenBookQA (Mihaylov et al. (2018)). There are challenges with regards to the computational requirements necessary for inferencing LLMs,"the compute and memory requirements of state-of-the-art language models have grown by three orders of magnitude in the last three years, and are projected to continue growing far faster than hardware capabilities" (Bommasani et al., 2022, p. 97).

Quantization and pruning are some of the strategies that can be used to help reduce computational requirement and memory footprint of LLMs. However applying these strategies often comes at the cost of increasing the LLM Perplexity (PPL), a metric for evaluation the uncertainty of a model in predicting a sequence. Quantization methods involve reducing the numerical precision of the model's weights, allowing 32-bit floating point value to be represented as an 8-bit floating point value or even lower. Popular quantization include GPT-Generated Unified Format (GGUF), Activation-Aware Weight Quantization (AWQ), Vector Post-Training Quantization (VPTQ) and others. Pruning involves removing parts of the model that have little effect on the output, this process could involve removing neurons or entire layers.

1.2 Problem Statement and Motivation

As mentioned in the previous section, the hardware requirements for medium to large sized LLMs make it difficult for consumers or small organisations to run their own local LLMs, often requiring to use third-party providers for access to modern powerful LLMs. This potentially reduces their privacy, security and accessibility, which could be otherwise achieved by running LLMs locally on their own hardware. For large businesses who might already be hosting their own models, this could be an opportunity to potentially reduce their running costs with regards to LLMs.

1.3 Research Scope and Limitations

This research will be using a select few foundational LLMs for testing and evaluation. The selected LLMs will be Gemma2 9B from Google, LLaMa 3.1 from Meta 8B, Phi3-Small 7B from Microsoft and Qwen2.5 7B from Alibaba. These models were selected due to their research permissive licenses, LLM community popularity, reputability of the companies that have trained them.

The hardware for conducting this research will be limited to a single Nvidia RTX 4090 GPU, which will be sufficient to run the selected LLMs without any quantization to establish a baseline.

1.4 Research Questions

This paper aims to answer the following Research Questions (RQ):

RQ1: What quantization methods (GGUF, AWQ, VPTQ) are the most effective for reducing LLM (LLaMa 3.1 8B) inferencing requirements while retaining most of it's output quality?

RQ2: What pruning strategies (block-wise, channel-wise, layer-wise) are the most effective for reducing LLM

(LLaMa 3.1 8B) inferencing requirements while retaining most of it's output quality?

RQ3: Deriving the best performing method (or combination of methods) from the previous questions, what is the best achievable performance of LLaMa 3.1 8B on a Raspberry Pi 4b (8GB RAM version)?

2. PRELIMINARY LITERATURE REVIEW

3. WORKING THEORY

4. RESEARCH DESIGN

4.1 Introduction

4.2 Design

5. CONCLUSION

ACKNOWLEDGEMENTS

Place acknowledgments here. (left-over from IFAC template, not sure if I will need it yet)

REFERENCES

Bommasani, R., Hudson, D.A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M.S., Bohg, J., Bosselut, A., Brunskill, E., Brynjolfsson, E., Buch, S., Card, D., Castellon, R., Chatterji, N., Chen, A., Creel, K., Davis, J.Q., Demszky, D., Donahue, C., Doumbouya, M., Durmus, E., Ermon, S., Etchemendy, J., Ethayarajh, K., Fei-Fei, L., Finn, C., Gale, T., Gillespie, L., Goel, K., Goodman, N., Grossman, S., Guha, N., Hashimoto, T., Henderson, P., Hewitt, J., Ho, D.E., Hong, J., Hsu, K., Huang, J., Icard, T., Jain, S., Jurafsky, D., Kalluri, P., Karamcheti, S., Keeling, G., Khani, F., Khattab, O., Koh, P.W., Krass, M., Krishna, R., Kuditipudi, R., Kumar, A., Ladhak, F., Lee, M., Lee, T., Leskovec, J., Levent, I., Li, X.L., Li, X., Ma, T., Malik, A., Manning, C.D., Mirchandani, S., Mitchell, E., Munyikwa, Z., Nair, S., Narayan, A., Narayanan, D., Newman, B., Nie, A., Niebles, J.C., Nilforoshan, H., Nyarko, J., Ogut, G., Orr, L., Papadimitriou, I., Park, J.S., Piech, C., Portelance, E., Potts, C., Raghunathan, A., Reich, R., Ren, H., Rong, F., Roohani, Y., Ruiz, C., Ryan, J., Ré, C., Sadigh, D., Sagawa, S., Santhanam, K., Shih, A., Srinivasan, K., Tamkin, A., Taori, R., Thomas, A.W., Tramèr, F., Wang, R.E., Wang, W., Wu, B., Wu, J., Wu, Y., Xie, S.M., Yasunaga, M., You, J., Zaharia, M., Zhang, M., Zhang, T., Zhang, X., Zhang, Y., Zheng, L., Zhou, K., and Liang, P. (2022). On the opportunities and risks of foundation models. URL https://arxiv.org/abs/2108.07258.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have solved question answering? try arc, the ai2 reasoning challenge. URL https://arxiv.org/abs/1803.05457.

Mihaylov, T., Clark, P., Khot, T., and Sabharwal, A. (2018). Can a suit of armor conduct electricity? a new dataset for open book question answering. URL https://arxiv.org/abs/1809.02789.

Sakaguchi, K., Bras, R.L., Bhagavatula, C., and Choi, Y. (2019). Winogrande: An adversarial winograd schema challenge at scale. URL https://arxiv.org/abs/1907.10641.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). Llama: Open and efficient foundation language models. URL https://arxiv.org/abs/2302.13971.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. URL https://arxiv.org/abs/1706.03762.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. (2019). Hellaswag: Can a machine really finish your sentence? URL https://arxiv.org/abs/1905.07830.

ABBREVIATIONS

AWQ Activation-Aware Weight Quantization. 1

GGUF GPT-Generated Unified Format. 1
GPT Generative Pre-Trained Transformer. 1

LLM Large Language Model. 1

PPL Perplexity. 1

VPTQ Vector Post-Training Quantization. 1

Appendix A. APPENDIX A