Alma Mater Studiorum · Università di Bologna

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
ARTIFICIAL INTELLIGENCE

# An effective strategy for reducing the size of LLaMA-based Language Models

**Supervisor:**
**Prof. Francesco Conti**

**Presented by:**
**Angelo Galavotti**

**Co-supervisor:**
**Luca Bompani**

**Sessione I**
**Anno Accademico 2024/2025**

*Alla migliore madre del mondo,*
*al miglior padre del mondo.*

**Abstract**

Pending

# Contents

# List of Figures

# Elenco dei frammenti di codice

# Chapter 1

# Introduction

It is no secret that in the last three years, *Large Language Models* (LLMs) have fundamentally transformed our relationship with technology. Their impact rivals the most significant innovations of the past century, such as the internet and smartphone. When people contemplate Artificial Intelligence today, they immediately think of ChatGPT or Claude, which have seamlessly integrated into our daily routines. Yet these powerful tools come with significant environmental concerns. Their development and operation consume vast amounts of energy and water resource: modern data centers supporting these models require extensive cooling systems and electricity consumption that can rival small cities.

In this opening chapter we will briefly examine the evolution of LLMs and provide a more technical description of their capabilities. Furthermore, we'll investigate their considerable environmental footprint while highlighting the growing imperative for efficient, locally-deployable models that democratize access without depleting our planet's resources. In this context, we will introduce the *FRANKEN-LLAMA* project, which aims to create a more sustainable and efficient LLM. The future of AI depends not just on what these models can do, but how sustainably they can do it.

## 1.1 A brief overview of the evolution of LLMs

Fundamentally, at the core of LLMs lies the concept of *transformers*, a neural network architecture introduced in 2017 by Vaswani et al. in their famous paper "Attention is All You Need". Initially designed for translation tasks, transformers have since been adapted for a wide range of natural language processing (NLP) tasks such as summarization and sentiment analysis. A famous example of a transformer model is *BERT* (Bidirectional Encoder Representations from Transformers), which has been widely used for various NLP tasks. BERT's architecture allows it to understand the context of words in a sentence by considering both the left and right context simultaneously, making it particularly effective for classification tasks such as entity named recognition.

However, the biggest impact of transformers has been in the realm of text generation, where they can produce consistent and contextually relevant text based on a given prompt. This is achieved through a mechanism called *self-attention*, which allows the model to weigh the importance of different words in a sentence when generating text. By using self-attention, transformers can capture long-range dependencies and relationships between words. A more technical overview of the transformer architecture is provided in Section **??**. The GPT (Generative Pre-trained Transformer) series, developed by OpenAI, is a prime example of this capability, with GPT-4 being the most recent version. These models are pre-trained on vast amounts of text data and its performance has become a new benchmark for other models in the field.

## 1.2 The darker side of LLMs and scope of this project

## 1.3 Document structure

AAAAAAAA AM I GOING CRAZY?

# Chapter 2

# Background and Related Work

Before explaining the details and implementation of the methodology used in this project, it is essential to provide an overview of the evolution of the inner workings of the Transformer architecture as well as Large Language Models. In addition, we will also discuss revelant compression techniques that have been developed in this context, and how they influenced this work. Finally, we will also shed some light on the target hardware, whose limitations have been a driving force behind the design choices made in this project.

## 2.1 The architecture of Transformers

## 2.2 The structure of Large Language Models

## 2.3 Relevant compression techniques

# Chapter 3

# Methodology

nada

# Chapter 4

# Implementation

nada

# Chapter 5

# Experimental Results and Analysis

nada

# Chapter 6

# Conclusion and Future Work

## 6.1 Future work

ok

## 6.2 Final remarks

The objective of this project was to research and implement a methodology for compressing LLaMA based LLMs,

# Bibliography


[1] WLCG, *WLCG*, https://wlcg.web.cern.ch/.

[2] CERN, *The Large Hadron Collider*, https://home.cern/science/accelerators/large-hadron-collider.

[3] INFN CNAF, *Calcolo - INFN-CNAF*, https://www.cnaf.infn.it/calcolo/.

[4] OpenPolicyAgent, *OpenPolicyAgent*, https://www.openpolicyagent.org/.

[5] OpenPolicyAgent, *Policy Language*, https://www.openpolicyagent.org/docs/latest/policy-language/.

[6] OpenPolicyAgent, *OPA REST API* https://www.openpolicyagent.org/docs/latest/rest-api/.

[7] Nginx, Inc., *NGINX*, https://www.nginx.com/.

[8] Nginx, Inc., *NGINX Documentation*, https://nginx.org/en/docs/.

[9] NGINX, *njs scripting language*, https://nginx.org/en/docs/njs/.

[10] Docker, *Docker*, https://www.docker.com/.

[11] Docker, *Compose*, hhttps://docs.docker.com/compose/.

[12] IETF, *RFC 3820*, https://www.rfc-editor.org/rfc/rfc3820.html.

[13] Daniel Stenberg, *cURL*, https://curl.se/.

[14] IETF, *RFC 7519*, https://www.rfc-editor.org/rfc/rfc7519.

[15] IETF, *RFC 6749*, https://www.rfc-editor.org/rfc/rfc6749.

[16] OpenID, *OpenID Connect*, https://openid.net/connect/.

[17] auth0, *Single Sign On*, https://auth0.com/docs/authenticate/single-sign-on.

[18] Indigo-DC, *oidc-agent*, https://github.com/indigo-dc/oidc-agent.

[19] Mine Altunay, Brian Bockelman, Andrea Ceccanti, et al., *WLCG Common JWT Profiles*, 2019, https://doi.org/10.5281/zenodo.3460258.

[20] CERN, *A quick introduction to VOMS*, https://twiki.cern.ch/twiki/pub/LCG/LhcbPage/A_quick_introduction_to_VOMS.pdf.

[21] INFN CNAF SD, *VOMS Client Guide*, https://italiangrid.github.io/voms/documentation/voms-clients-guide/3.0.3/.

[22] INFN CNAF Software Development, *NGINX HTTP VOMS Module*, https://baltig.infn.it/storm2/ngx_http_voms_module/.

[23] Riccardo Zappi, *Understanding StoRM*, https://agenda.cnaf.infn.it/getFile.py/access?contribId=2&resId=1&materialId=slides&confId=305.

[24] WLCG, *WLCG Tape API*, hhttps://indico.cern.ch/event/1026385/contributions/4309594/attachments/2301962/3915769/WLCG%20Tape%20REST%20API%20reference%20document.pdf.

# Acknowledgements

Ringrazio il professor Ozalp Babaoglu per la sua disponibilità come relatore, nonostante fosse prossimo alla pensione. Sono molto grato di essere uno dei suoi ultimi tesisti della sua carriera. I suoi contributi nell'ambito dei sistemi operativi sono inestimabili.

Ringrazio il professor Francesco Giacomini, per avermi dato l'accesso a una delle esperienze formative più importanti della mia vita. Lo ringrazio inoltre per avermi assistito attentamente nello sviluppo del progetto e nella scrittura di questa tesi. Grazie anche per la tua pazienza nei miei confronti.

Ringrazio tutto il personale dell'INFN CNAF per aver reso l'esperienza ancora più gradevole e per avermi fatto visitare il centro di calcolo, trattandomi sempre come se fossi un loro collega.

Un ringraziamento speciale va alla mia famiglia, per avermi sempre dato la spinta di andare avanti e per credere in me, senza negarmi mai nulla.

Ringrazio tutti gli amici che ho conosciuto nel corso durante questi tre anni: Leon, Drif, Giaco, Baldo, Adriano, Donnoh, Vir, Matteo, Pino, Denis, Samuele, Alice e tanti altri. Non avrei potuto chiedere dei compagni migliori.

Infine, ringrazio Leti, per essermi stata accanto nei momenti più bui e avermi dato tutto l'affetto che esiste in questo mondo, anche quando non me lo meritavo.