

**A Technical Seminar Report on**  
**Large Language Models and LLaMA**

Submitted to  
The Department of Computer Science and Engineering (Data Science)  
Bachelor of Technology  
In  
Computer Science and Engineering (Data Science)  
(2020 – 2024)

**Arigela Pranay Kumar**  
**20R11A6701**



Department of Computer Science and Engineering (Data Science)  
**Geethanjali College of Engineering and Technology (UGC Autonomous)**

Accredited by NBA, (Affiliated to J.N.T.U.H, Approved by AICTE, New Delhi)

Cheeryal (V), Keesara (M), Medchal.Dist.-501 301.

**April-2024**

**Department of Computer Science and Engineering (Data science)**

Geethanjali College of Engineering and Technology



**CERTIFICATE**

This is to certify that this technical report on “**Large Language Models and LLaMA**”, submitted by **Arigela Pranay Kumar (20R11A6701)**, in the year 2024 in partial fulfillment of the academic requirements of Geethanjali college of Engineering and Technology for the award of the degree of **Bachelor of Technology in Computer Science Engineering (Data Science)**, is a bonafide work that has been carried out by them as part of their **Technical presentation during fourth Year Second Semester**. This report has not been submitted to any other institute or university for the award of any degree.

**Date of seminar:** 30/03/2024

**Guide**

Mr B.Venkateswarlu  
Assistant Professor  
CSE(DS)

**Coordinator**

Mr. S.Tirupati Rao  
Associate Professor  
CSE(DS)

**Head of the Department**

Dr. L Kiran Kumar Reddy  
HoD, CSE(DS)

## **ABSTRACT**

Language models have revolutionized natural language processing tasks, enabling machines to understand, generate, and manipulate human language with remarkable fluency. Language models have emerged as transformative tools in natural language processing, enabling machines to comprehend, generate, and manipulate human language with unprecedented accuracy and fluency. This seminar offers a broad exploration of language models, including their architecture, capabilities, and real-world applications, tailored for a general audience. Central to our discussion is an examination of Large Language Models (LLMs), particularly focusing on those developed by Meta AI, known as LLAMA (Large Language Models by Meta AI). We elucidate the evolution of LLMs, from their inception to the latest advancements, such as GPT-3.5, Bard and beyond. Emphasis is placed on showcasing the remarkable capabilities of LLMs in tasks ranging from text generation, Image generation and completion to language translation and sentiment analysis.

## INDEX

| S.NO | LIST OF CONTENTS          | PAGE NO |
|------|---------------------------|---------|
| 1    | TECHNICAL REPORT          | 5-12    |
| 2    | PAPER PRESENTATION SLIDES | 14-22   |
| 3    | BASE PAPER                | 23-34   |

# TECHNICAL REPORT

## Large Language Models and LLaMA

### 1. Introduction to Large Language Models:

Large language models are a type of artificial intelligence (AI) that process and generate human-like text. They're essentially computer programs that are trained on massive amounts of text data, allowing them to learn the intricacies of language. This data can include books, articles, code, and even conversations, giving LLMs a broad understanding of how language works.

LLMs leverage a specific kind of neural network architecture called a transformer. Imagine the transformer like a complex web of connections, where each connection represents a relationship between words. By analyzing tons of text data, the LLM adjusts the strength of these connections, essentially learning the probabilities of how words follow each other. This allows the LLM to predict the next word in a sequence, and ultimately generate human-quality text.

### 2. Features of Large Language Models:

Large language models (LLMs) have taken the world by storm, captivating us with their ability to mimic human-like language proficiency. But beneath the surface of creative text generation and informative answers lies a complex architecture and a range of powerful features. Let's delve deeper into the world of LLMs, exploring their intricate workings and the exciting possibilities they hold.

#### 2.1 Architectural Foundation

At the heart of every LLM lies a powerful neural network architecture called a transformer. Unlike traditional recurrent neural networks (RNNs), transformers excel at handling long-range dependencies in language. Imagine a sentence – a transformer can not only analyze the immediate relationship between neighboring words but also consider how words further apart influence each other's meaning. This capability is crucial for understanding complex sentence structures and generating coherent text.

Here's a simplified breakdown of how the transformer works:

**Attention Mechanism:** The core strength of the transformer. It allows the model to focus on specific parts of the input sequence, attributing varying degrees of importance to different words based on the context.

**Encoder-Decoder Structure:** The transformer typically consists of two parts: an encoder and a decoder. The encoder processes the input text, capturing its meaning and relationships. The decoder then utilizes the encoded information to generate the output, like translating a language or writing a new sentence.

## **2.2. Feature Frenzy**

Equipped with the power of transformers, LLMs boast a diverse set of features that make them valuable tools across various applications. Here are some key capabilities:

**Text Generation:** LLMs can generate different creative text formats with remarkable fluency. From poems and code to scripts and musical pieces, they can produce human-quality content in response to prompt or specific instructions.

**Question Answering:** No longer are simple "yes" or "no" answers the limit. LLMs can answer open ended, challenging, or strange questions in an informative way, providing summaries of relevant information from their vast knowledge base.

**Summarization:** Sifting through lengthy documents can be a chore. LLMs can condense extensive pieces of text into shorter, key point-driven summaries, saving you valuable time and effort.

**Translation:** Breaking down language barriers, LLMs are being fine-tuned to translate languages more accurately and naturally, preserving the intended meaning and nuances of the original text.

**Writing Assistance:** Imagine having a helpful AI editor by your side. LLMs can check grammar and suggest rephrasing for improved clarity and flow. They can even offer different stylistic options to match the tone and purpose of your writing.

## 2.3. Considerations and Challenges

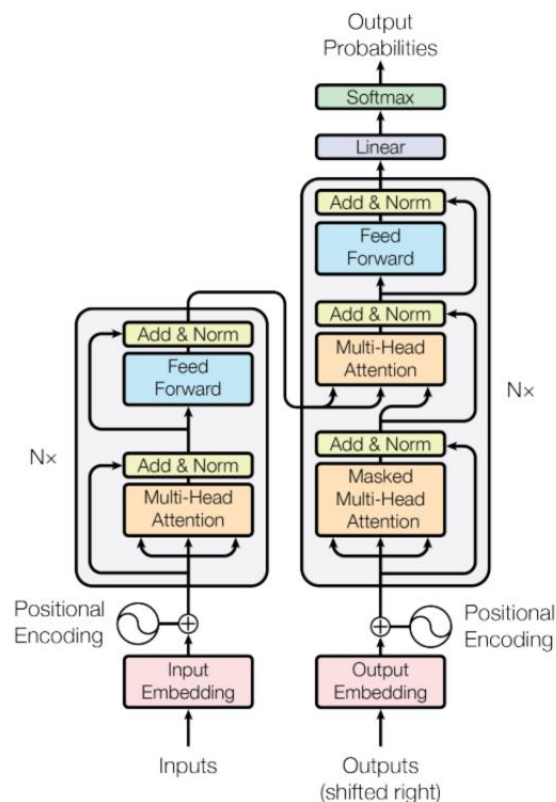
While LLMs offer a plethora of exciting features, it's important to acknowledge the challenges and considerations that come with this powerful technology:

**Training & Cost:** The massive amounts of data and computational power required to train LLMs make them expensive to create and maintain. This limits accessibility and raises questions about resource allocation.

**Accuracy & Verification:** While impressive, LLMs are still under development. They can still generate factually incorrect or misleading content. It is essential to cross-check information from LLMs with reliable sources and exercise caution when interpreting their outputs.

## 3. Transformers: The Engine Powering Large Language Models:

Transformers are a specific type of neural network architecture that have revolutionized the field of natural language processing (NLP). At the heart of every LLM, transformers play a pivotal role in enabling them to understand and generate human-like text.



**Fig 1 Transformer Architecture**

### 3.1. Understanding the Building Blocks

- **Attention Mechanism:** This is the crown jewel of transformers. It allows the model to focus on specific parts of the input sequence, attributing varying degrees of importance to different words based on the context. Imagine a sentence like "The quick brown fox jumps over the lazy dog." The attention mechanism enables the transformer to pay closer attention to "jumps" when processing the word "over," understanding the relationship between these words is crucial for accurate interpretation.
- **Encoder-Decoder Structure:** Most transformers follow an encoder-decoder structure.
  - **Encoder:** This part of the transformer processes the input text, essentially reading and understanding the meaning. It breaks down the input sequence into a series of encoded representations, capturing the relationships between words.
  - **Decoder:** The decoder then utilizes the encoded information from the encoder to generate the output, like translating a language or writing a new sentence.

### 3.2. How Transformers Work

1. **Input Preparation:** The input text is first pre-processed and converted into a numerical representation that the transformer can understand. This typically involves breaking the text down into individual words and assigning a unique identifier to each word.
2. **Encoder:**
  - The encoder processes the input sequence one word (or a small group of words) at a time.
  - For each word, the transformer considers its own meaning and its relationship with other words in the sequence using the attention mechanism.
  - The output of the encoder is a series of encoded representations, each capturing the context of a particular word within the entire sentence.
3. **Decoder:**
  - The decoder takes the encoded representations from the encoder as input.
  - It also has its own attention mechanism, allowing it to focus on specific parts of the encoded information as it generates the output sequence.
  - One by one, the decoder predicts the next word in the output sequence, heavily influenced by the encoded information and the previously generated words.



#### 4. Output Generation:

- The decoder continues predicting words until it reaches a stopping point, determined by the task at hand (e.g., translating a complete sentence or generating text of a specific length).
- The final output sequence is the generated text, which can be a translation, a summary, or a new piece of creative writing.

### 3.3. Transformer Variants

The base transformer architecture has been extended and modified to address various NLP tasks. Here are some popular variants:

- BERT (Bidirectional Encoder Representations from Transformers): A pre-trained transformer model that excels at understanding the nuances of language, used for tasks like question answering and sentiment analysis.
- GPT (Generative Pre-training Transformer): Another pre-trained transformer model, known for its exceptional text generation capabilities.
- T5 (Text-to-Text Transfer Transformer): A versatile transformer model that can be fine-tuned for a wide range of NLP tasks, including summarization, translation, and question answering.

## 4. LLaMA:

Developed by Meta AI, LLaMA (stands for "Large Language Model for Anything") is a family of open-source LLMs introduced in February 2023. It quickly gained recognition for its impressive performance and focus on accessibility. Here's what makes LLaMA important:

### 1. Emphasis on Efficiency and Scalability

Unlike previous LLMs, LLaMA's developers focused on achieving high performance by scaling the model's training data rather than the number of parameters. This approach has several advantages:

### 2. Impressive Performance with Publicly Available Data

LLaMA comes in various sizes, ranging from 7 billion to 65 billion parameters. Despite having a smaller number of parameters compared to some other LLMs, LLaMA achieves state-of-the-art performance on several NLP benchmarks. This demonstrates the effectiveness of the data-driven approach.

### 3. Commitment to Openness and Collaboration

One of LLaMA's key strengths is its commitment to open science. The model and its training code are publicly available, allowing researchers to freely explore its capabilities and contribute to its development. This fosters collaboration and accelerates progress in the field of LLMs.

### 4. Potential for Widespread Applications

LLaMA's capabilities span a wide range of NLP tasks, including:

- **Text Generation:** Like other LLMs, LLaMA can generate different creative text formats, from poems and code to scripts and even different writing styles.
- **Question Answering:** It can answer your questions in an informative way, even if they are open ended, challenging, or strange.
- **Summarization:** LLMs can take lengthy pieces of text and condense them into shorter, key point-driven summaries.
- **Translation:** LLMs are being developed to translate languages more accurately and fluently, and LLaMA is part of this ongoing effort.

### 5. A Stepping Stone for Future LLM Development

LLaMA's success paves the way for a new generation of LLMs that are not only powerful but also more accessible and efficient. By focusing on data-driven approaches and open collaboration, researchers can continue to push the boundaries of what LLMs can achieve.

## **5. Challenges of LLM:**

### **1. Training & Cost:**

- **Data Deluge:** Training LLMs requires massive amounts of data, often in the form of text and code. Gathering, cleaning, and storing this data can be expensive and time-consuming.
- **Computational Crunch:** The training process itself demands significant computational power. Running these complex models on powerful hardware adds to the overall cost.

### **2. Bias & Fairness:**

- **Inherent Bias:** LLMs inherit biases from the data they're trained on. If the training data is skewed or imbalanced, the LLM's outputs can reflect those biases. This can lead to discriminatory or offensive outputs.
- **Mitigating Bias:** Addressing bias requires careful selection of training data and the implementation of techniques to identify and mitigate bias in the model's outputs.

### **3. Accuracy & Verification:**

- **Hallucination Nation:** LLMs can sometimes generate factually incorrect or misleading content, often referred to as hallucinations. These can be difficult to detect, especially for non-experts.
- **Fact-Checking Frenzy:** It's crucial to double-check information from LLMs with reliable sources. Human oversight and verification are still essential when using LLMs for critical tasks.

### **4. Explainability & Transparency:**

- **Black Box Blues:** The inner workings of LLMs can be complex and opaque. Understanding how they arrive at their outputs can be challenging, hindering trust and limiting their applications in areas requiring clear reasoning.
- **Opening the Black Box:** Research into explainable AI (XAI) techniques is ongoing, aiming to make LLM decision-making processes more transparent and interpretable.

### **5. Safety & Security:**

- **Malicious Manipulation:** LLMs could be misused to generate fake news, propaganda, or spam. There's a risk of malicious actors exploiting LLMs to manipulate people or spread misinformation.
- **Security Safeguards:** Developing robust security measures is crucial to prevent misuse and ensure the responsible development and deployment of LLMs.

## **6. Benefits of LLaMA:**

### **1. Training & Cost:**

- **Data Deluge:** Training LLMs requires massive amounts of data, often in the form of text and code. Gathering, cleaning, and storing this data can be expensive and time-consuming.
- **Computational Crunch:** The training process itself demands significant computational power. Running these complex models on powerful hardware adds to the overall cost.

### **2. Bias & Fairness:**

- **Inherent Bias:** LLMs inherit biases from the data they're trained on. If the training data is skewed or imbalanced, the LLM's outputs can reflect those biases. This can lead to discriminatory or offensive outputs.
- **Mitigating Bias:** Addressing bias requires careful selection of training data and the implementation of techniques to identify and mitigate bias in the model's outputs.

### **3. Accuracy & Verification:**

- **Hallucination Nation:** LLMs can sometimes generate factually incorrect or misleading content, often referred to as hallucinations. These can be difficult to detect, especially for non-experts.
- **Fact-Checking Frenzy:** It's crucial to double-check information from LLMs with reliable sources. Human oversight and verification are still essential when using LLMs for critical tasks.

### **4. Explainability & Transparency:**

- **Black Box Blues:** The inner workings of LLMs can be complex and opaque. Understanding how they arrive at their outputs can be challenging, hindering trust and limiting their applications in areas requiring clear reasoning.
- **Opening the Black Box:** Research into explainable AI (XAI) techniques is ongoing, aiming to make LLM decision-making processes more transparent and interpretable.

### **5. Safety & Security:**

- **Malicious Manipulation:** LLMs could be misused to generate fake news, propaganda, or spam. There's a risk of malicious actors exploiting LLMs to manipulate people or spread misinformation.
- **Security Safeguards:** Developing robust security measures is crucial to prevent misuse and ensure the responsible development and deployment of LLMs.

## 7. CONCLUSION:

Large language models (LLMs) are powerful AI tools that can process and generate human-like text. They hold immense potential to revolutionize various fields, from education and customer service to content creation and scientific discovery. However, several challenges hinder their development and widespread adoption, LLaMA, a cost-effective and open-source LLM, takes a step in the right direction by promoting data efficiency, bias mitigation through open data, and collaboration for security and explainability. By focusing on these areas, LLaMA paves the way for a new generation of LLMs that are not only powerful but also more accessible, less biased, and ultimately contribute to the development of safer and more reliable AI tools, LLMs hold immense potential to revolutionize various fields. LLaMA, with its emphasis on efficiency, openness, and collaboration, marks a significant step towards more accessible, responsible, and trustworthy LLMs. By acknowledging the challenges and working towards solutions, we can ensure that LLMs become powerful tools for good, advancing our understanding of language and shaping a brighter future for artificial intelligence.

## REFERENCES

1. Suvojit Hore (2023). What are Large Language Models (LLMs)? Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2023/09/a-survey-of-large-language-models-llms/>
2. IBM (2024). What Are Large Language Models (LLMs)? <https://www.ibm.com/topics/large-language-models>
3. James Zou et al. (2019). GPT-3: Language Models are Few-Shot Learners. <https://arxiv.org/abs/2005.14165>
4. Ashish Vaswani et al. (2017). Attention is All You Need. <https://arxiv.org/pdf/1706.03762>
5. Daniel Hernandez et al. (2023). Large Language Models in the Wild: A Survey. <https://arxiv.org/pdf/2303.18223>


## SLIDES

Technical Seminar on

# Large Language Models and LLaMA

By  
Arigela Pranay Kumar 20R11A6701

Under the Guidance of  
Mr B Venkateshwarlu  
Assistant Professor



**Geethanjali College of Engineering and Technology**  
Department of Data Science

## Table of contents

|           |             |           |                   |
|-----------|-------------|-----------|-------------------|
| <b>01</b> | Abstract    | <b>02</b> | Objective         |
| <b>03</b> | About LLaMA | <b>04</b> | Literature Survey |
| <b>05</b> | Base Papers | <b>06</b> | Challenges        |

01

# Abstract

---



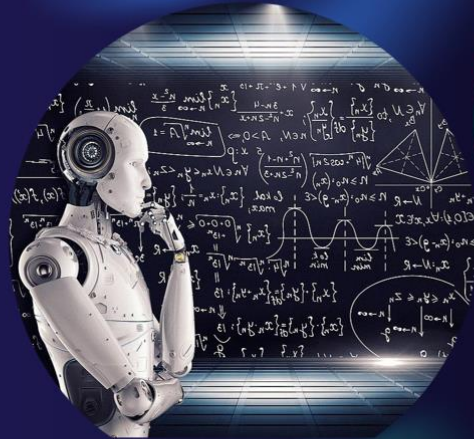
01

## Abstract

Language models have revolutionized natural language processing tasks, enabling machines to understand, generate, and manipulate human language with remarkable fluency. Language models have emerged as transformative tools in natural language processing, enabling machines to comprehend, generate, and manipulate human language with unprecedented accuracy and fluency. This seminar offers a broad exploration of language models, including their architecture, capabilities, and real-world applications, tailored for a general audience. Central to our discussion is an examination of Large Language Models (LLMs), particularly focusing on those developed by Meta AI, known as LLAMA (Large Language Models by Meta AI). We elucidate the evolution of LLMs, from their inception to the latest advancements, such as GPT-3.5, Bard and beyond. Emphasis is placed on showcasing the remarkable capabilities of LLMs in tasks ranging from text generation, Image generation and completion to language translation and sentiment analysis.

02

## 02 Objective



03

## Objectives

### 1. Providing an Overview

Offer a comprehensive overview of language models, covering fundamental concepts such as recurrent neural networks (RNNs), transformers, and self-attention mechanisms, as well as highlighting the evolution of large language models (LLMs) up to the latest advancements like LLAMA by Meta AI.

### 2. Educating the Audience

The primary goal is to educate the audience, which consists of a general audience, about large language models, including their architecture, functioning, and significance in natural language processing tasks.

### 3. Highlighting capabilities

Showcase the capabilities of language models, particularly focusing on LLMs, in various tasks including text generation, completion, translation, image generation, sentiment analysis, and more. Emphasize their potential to enhance productivity and efficiency in language-related tasks.

04



#### 4. Discussing Challenges

Discuss the challenges and limitations associated with language models, including issues related to bias, ethical considerations, and computational requirements. Encourage critical thinking and discussion about the societal implications of deploying such models

#### 5. Introducing LLaMA

Introduce the concept of Open source models and their significance in enabling language models to adapt and evolve, ensuring their continued relevance and effectiveness in processing language data from diverse sources

05

03

## About LLaMA

06

## About LLaMA

Llama 2 is a family of pre-trained and fine-tuned large language models (LLMs) released by Meta AI in 2023. Released free of charge for research and commercial use, Llama 2 AI models are capable of a variety of natural language processing (NLP) tasks, from text generation to programming code.

The Llama 2 model family, offered as both base foundation models and fine-tuned “chat” models, serves as the successor to the original LLaMa 1 models, which were released in 2022 under a noncommercial license granting access on a case-by-case basis exclusively to research institutions. Unlike their predecessors, Llama 2 models are available free of charge for both AI research and commercial use.

07

## Benefits of LLaMA

- 1) **Reduced computational footprint:** LLaMA is a more lightweight model compared to other large language models. This translates to lower demands on computing power and resources [1]. This is advantageous for researchers as it allows them to experiment and iterate on new approaches more readily without requiring access to massive computational infrastructure
- 2) **Democratization of research:** Meta's decision to release LLaMA under a non-commercial license means that the model is available to a wider range of researchers and institutions. This fosters collaboration and innovation in the field of AI research, as more minds can work on advancing the capabilities of the model.
- 3) **Potential Benefits: Enhanced NLP performance:** LLaMA is designed to have a strong grasp of language nuances. This has the potential to drive advancements in areas like machine translation, question answering, and creative text generation
- 4) **Transparency and Safety (potential): Open-source advantage:** Because LLaMA's inner workings are exposed, researchers have the ability to identify and address potential biases or vulnerabilities within the model. This transparency can lead to the development of safer and more reliable AI applications

08

# Difference between a ML and LLM

## Here's an analogy

Machine Learning is like a toolbox containing many different tools for various tasks. Large Language Models are specialized tools within that toolbox, designed specifically for working with language.

**Underlying Techniques:** Both LLMs and ML models often rely on similar techniques like neural networks, but the specific architectures used can be tailored for the type of data they handle.

**Data Requirements:** LLMs typically require massive amounts of text data for training, while the data requirements for ML models can vary depending on the specific task.

09

# Transformers

Large language models (LLMs) have taken the world by storm with their ability to generate text, translate languages, and answer your questions in an informative way. But what's the secret sauce behind these impressive feats? Enter transformers, a powerful neural network architecture that has revolutionized natural language processing (NLP).

**Understanding Context:** Unlike traditional methods that process text sequentially, transformers can analyze entire sentences at once. This allows them to capture the relationships between words, giving them a better understanding of context.

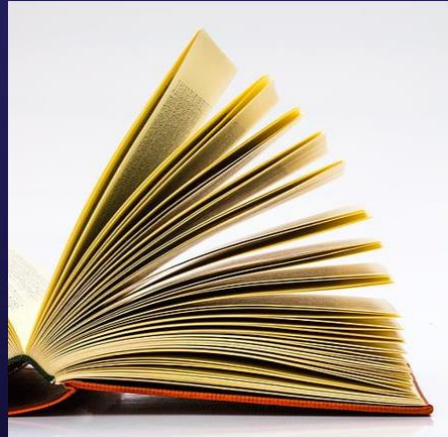
**Attention Mechanism:** The core of a transformer is the self-attention mechanism. This clever trick lets the model focus on the most relevant parts of the input text, like how a reader might focus on specific words in a sentence to grasp its meaning.

**Encoder-Decoder:** Some LLMs use an encoder-decoder structure. The encoder takes the input text and encodes it into a representation, while the decoder uses this information to generate the output, like translating a sentence or writing a continuation of a story.

10

# 04

## Literature Survey



11

## Literature Survey

1. Suvojit Hore (2023). What are Large Language Models (LLMs)? Analytics Vidhya: <https://www.analyticsvidhya.com/blog/2023/09/a-survey-of-large-language-models-llms/>
2. IBM (2024). What Are Large Language Models (LLMs)? <https://www.ibm.com/topics/large-language-models>
3. James Zou et al. (2019). GPT-3: Language Models are Few-Shot Learners. <https://arxiv.org/abs/2005.14165>
4. Ashish Vaswani et al. (2017). Attention is All You Need. <https://arxiv.org/pdf/1706.03762>
5. Daniel Hernandez et al. (2023). Large Language Models in the Wild: A Survey. <https://arxiv.org/pdf/2303.18223>

12

# 05

## Base Papers

13

## Base Papers

1. LLaMA: Open and Efficient Foundation Language Models, Cornell University, Meta AI,  
arXiv:2302.13971  
<https://arxiv.org/abs/2302.13971>
2. Large Language Models: A Comprehensive Survey of its Applications, Challenges,  
Limitations, and Future Prospects, ResearchGate publication 372278221  
[https://www.researchgate.net/publication/372278221\\_Large\\_Language\\_Models\\_A\\_Comprehensive\\_Survey\\_of\\_its\\_Applications\\_Challenges\\_Limitations\\_and\\_Future\\_Prospects](https://www.researchgate.net/publication/372278221_Large_Language_Models_A_Comprehensive_Survey_of_its_Applications_Challenges_Limitations_and_Future_Prospects)

14

## Challenges

1. **Bias:** LLMs inherit biases from the data they're trained on. This can lead to outputs that are discriminatory, offensive, or simply inaccurate.
2. **Environment:** Training LLMs requires massive amounts of computing power, which can be energy-intensive and contribute to greenhouse gas emissions.
3. **Transparency:** It can be difficult to understand how an LLM arrives at its answer. This lack of transparency makes it hard to debug issues like bias or errors.
4. **Accountability:** Who's responsible if an LLM generates harmful content? The model developer, the company using it, or someone else?
5. **Misinformation:** LLMs can be great at generating text, but they can't necessarily verify its accuracy. This can lead to the spread of misinformation.

16

# Thank You

THANK YOU

# LLaMA: Open and Efficient Foundation Language Models

Hugo Touvron\*, Thibaut Lavril\*, Gautier Izacard\*, Xavier Martinet  
Marie-Anne Lachaux, Timothee Lacroix, Baptiste Rozière, Naman Goyal  
Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin  
Edouard Grave\*, Guillaume Lample\*

Meta AI

## Abstract

We introduce LLaMA, a collection of foundation language models ranging from 7B to 65B parameters. We train our models on trillions of tokens, and show that it is possible to train state-of-the-art models using publicly available datasets exclusively, without resorting to proprietary and inaccessible datasets. In particular, LLaMA-13B outperforms GPT-3 (175B) on most benchmarks, and LLaMA-65B is competitive with the best models, Chinchilla-70B and PaLM-540B. We release all our models to the research community<sup>1</sup>.

## 1 Introduction

Large Languages Models (LLMs) trained on massive corpora of texts have shown their ability to perform new tasks from textual instructions or from a few examples (Brown et al., 2020). These few-shot properties first appeared when scaling models to a sufficient size (Kaplan et al., 2020), resulting in a line of work that focuses on further scaling these models (Chowdhery et al., 2022; Rae et al., 2021). These efforts are based on the assumption that more parameters will lead to better performance. However, recent work from Hoffmann et al. (2022) shows that, for a given compute budget, the best performances are not achieved by the largest models, but by smaller models trained on more data.

The objective of the scaling laws from Hoffmann et al. (2022) is to determine how to best scale the dataset and model sizes for a particular *training* compute budget. However, this objective disregards the *inference* budget, which becomes critical when serving a language model at scale. In this context, given a target level of performance, the preferred model is not the fastest to train but the fastest at inference, and although it may be cheaper to train a large model to reach a certain level of

performance, a smaller one trained longer will ultimately be cheaper at inference. For instance, although Hoffmann et al. (2022) recommends training a 10B model on 200B tokens, we find that the performance of a 7B model continues to improve even after 1T tokens.

The focus of this work is to train a series of language models that achieve the best possible performance at various inference budgets, by training on more tokens than what is typically used. The resulting models, called *LLaMA*, ranges from 7B to 65B parameters with competitive performance compared to the best existing LLMs. For instance, LLaMA-13B outperforms GPT-3 on most benchmarks, despite being  $10\times$  smaller. We believe that this model will help democratize the access and study of LLMs, since it can be run on a single GPU. At the higher-end of the scale, our 65B-parameter model is also competitive with the best large language models such as Chinchilla or PaLM-540B.

Unlike Chinchilla, PaLM, or GPT-3, we only use publicly available data, making our work compatible with open-sourcing, while most existing models rely on data which is either not publicly available or undocumented (e.g. “Books – 2TB” or “Social media conversations”). There exist some exceptions, notably OPT (Zhang et al., 2022), GPT-NeoX (Black et al., 2022), BLOOM (Scao et al., 2022) and GLM (Zeng et al., 2022), but none that are competitive with PaLM-62B or Chinchilla.

In the rest of this paper, we present an overview of the modifications we made to the transformer architecture (Vaswani et al., 2017), as well as our training method. We then report the performance of our models and compare with others LLMs on a set of standard benchmarks. Finally, we expose some of the biases and toxicity encoded in our models, using some of the most recent benchmarks from the responsible AI community.

\* Equal contribution. Correspondence: {htouvron, thibautlav, gizacard, egrave, glample}@meta.com

<sup>1</sup><https://github.com/facebookresearch/llama>



## 2 Approach

Our training approach is similar to the methods described in previous work (Brown et al., 2020; Chowdhery et al., 2022), and is inspired by the Chinchilla scaling laws (Hoffmann et al., 2022). We train large transformers on a large quantity of textual data using a standard optimizer.

### 2.1 Pre-training Data

Our training dataset is a mixture of several sources, reported in Table 1, that cover a diverse set of domains. For the most part, we reuse data sources that have been leveraged to train other LLMs, with the restriction of only using data that is publicly available, and compatible with open sourcing. This leads to the following mixture of data and the percentage they represent in the training set:

**English CommonCrawl [67%].** We preprocess five CommonCrawl dumps, ranging from 2017 to 2020, with the CCNet pipeline (Wenzek et al., 2020). This process deduplicates the data at the line level, performs language identification with a fastText linear classifier to remove non-English pages and filters low quality content with an n-gram language model. In addition, we trained a linear model to classify pages used as references in Wikipedia v.s. randomly sampled pages, and discarded pages not classified as references.

**C4 [15%].** During exploratory experiments, we observed that using diverse pre-processed CommonCrawl datasets improves performance. We thus included the publicly available C4 dataset (Raffel et al., 2020) in our data. The preprocessing of C4 also contains deduplication and language identification steps: the main difference with CCNet is the quality filtering, which mostly relies on heuristics such as presence of punctuation marks or the number of words and sentences in a webpage.

**Github [4.5%].** We use the public GitHub dataset available on Google BigQuery. We only kept projects that are distributed under the Apache, BSD and MIT licenses. Additionally, we filtered low quality files with heuristics based on the line length or proportion of alphanumeric characters, and removed boilerplate, such as headers, with regular expressions. Finally, we deduplicate the resulting dataset at the file level, with exact matches.

**Wikipedia [4.5%].** We add Wikipedia dumps from the June-August 2022 period, covering 20

| Dataset       | Sampling prop. | Epochs | Disk size |
|---------------|----------------|--------|-----------|
| CommonCrawl   | 67.0%          | 1.10   | 3.3 TB    |
| C4            | 15.0%          | 1.06   | 783 GB    |
| Github        | 4.5%           | 0.64   | 328 GB    |
| Wikipedia     | 4.5%           | 2.45   | 83 GB     |
| Books         | 4.5%           | 2.23   | 85 GB     |
| ArXiv         | 2.5%           | 1.06   | 92 GB     |
| StackExchange | 2.0%           | 1.03   | 78 GB     |

Table 1: **Pre-training data.** Data mixtures used for pre-training, for each subset we list the sampling proportion, number of epochs performed on the subset when training on 1.4T tokens, and disk size. The pre-training runs on 1T tokens have the same sampling proportion.

languages, which use either the Latin or Cyrillic scripts: bg, ca, cs, da, de, en, es, fr, hr, hu, it, nl, pl, pt, ro, ru, sl, sr, sv, uk. We process the data to remove hyperlinks, comments and other formatting boilerplate.

**Gutenberg and Books3 [4.5%].** We include two book corpora in our training dataset: the Gutenberg Project, which contains books that are in the public domain, and the Books3 section of ThePile (Gao et al., 2020), a publicly available dataset for training large language models. We perform deduplication at the book level, removing books with more than 90% content overlap.

**ArXiv [2.5%].** We process arXiv Latex files to add scientific data to our dataset. Following Lewkowycz et al. (2022), we removed everything before the first section, as well as the bibliography. We also removed the comments from the .tex files, and inline-expanded definitions and macros written by users to increase consistency across papers.

**Stack Exchange [2%].** We include a dump of Stack Exchange, a website of high quality questions and answers that covers a diverse set of domains, ranging from computer science to chemistry. We kept the data from the 28 largest websites, removed the HTML tags from text and sorted the answers by score (from highest to lowest).

**Tokenizer.** We tokenize the data with the byte-pair encoding (BPE) algorithm (Sennrich et al., 2015), using the implementation from SentencePiece (Kudo and Richardson, 2018). Notably, we split all numbers into individual digits, and fallback to bytes to decompose unknown UTF-8 characters.



| params | dimension | $n$ heads | $n$ layers | learning rate | batch size | $n$ tokens |
|--------|-----------|-----------|------------|---------------|------------|------------|
| 6.7B   | 4096      | 32        | 32         | $3.0e^{-4}$   | 4M         | 1.0T       |
| 13.0B  | 5120      | 40        | 40         | $3.0e^{-4}$   | 4M         | 1.0T       |
| 32.5B  | 6656      | 52        | 60         | $1.5e^{-4}$   | 4M         | 1.4T       |
| 65.2B  | 8192      | 64        | 80         | $1.5e^{-4}$   | 4M         | 1.4T       |

Table 2: Model sizes, architectures, and optimization hyper-parameters.

Overall, our entire training dataset contains roughly 1.4T tokens after tokenization. For most of our training data, each token is used only once during training, with the exception of the Wikipedia and Books domains, over which we perform approximately two epochs.

## 2.2 Architecture

Following recent work on large language models, our network is based on the transformer architecture (Vaswani et al., 2017). We leverage various improvements that were subsequently proposed, and used in different models such as PaLM. Here are the main difference with the original architecture, and where we were found the inspiration for this change (in bracket):

**Pre-normalization [GPT3].** To improve the training stability, we normalize the input of each transformer sub-layer, instead of normalizing the output. We use the RMSNorm normalizing function, introduced by Zhang and Sennrich (2019).

**SwiGLU activation function [PaLM].** We replace the ReLU non-linearity by the SwiGLU activation function, introduced by Shazeer (2020) to improve the performance. We use a dimension of  $\frac{2}{3}4d$  instead of  $4d$  as in PaLM.

**Rotary Embeddings [GPTNeo].** We remove the absolute positional embeddings, and instead, add rotary positional embeddings (RoPE), introduced by Su et al. (2021), at each layer of the network.

The details of the hyper-parameters for our different models are given in Table 2.

## 2.3 Optimizer

Our models are trained using the AdamW optimizer (Loshchilov and Hutter, 2017), with the following hyper-parameters:  $\beta_1 = 0.9$ ,  $\beta_2 = 0.95$ . We use a cosine learning rate schedule, such that the final learning rate is equal to 10% of the maximal learning rate. We use a weight decay of 0.1 and gradient clipping of 1.0. We use 2,000 warmup

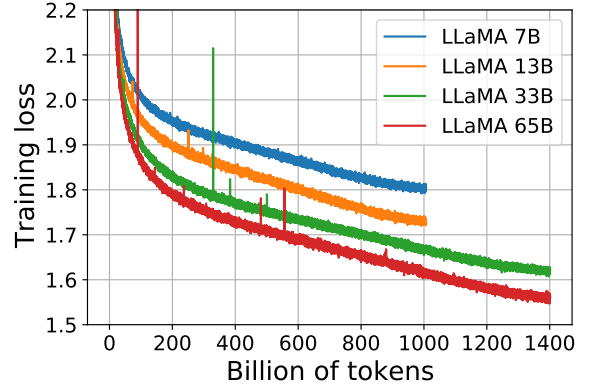


Figure 1: Training loss over train tokens for the 7B, 13B, 33B, and 65 models. LLaMA-33B and LLaMA-65B were trained on 1.4T tokens. The smaller models were trained on 1.0T tokens. All models are trained with a batch size of 4M tokens.

steps, and vary the learning rate and batch size with the size of the model (see Table 2 for details).

## 2.4 Efficient implementation

We make several optimizations to improve the training speed of our models. First, we use an efficient implementation of the causal multi-head attention operator, inspired by Rabe and Staats (2021) and Dao et al. (2022). This implementation, available in the xformers library,<sup>2</sup> reduces the memory usage and computation. This is achieved by not storing the attention weights and not computing the key/query scores that are masked due to the causal nature of the language modeling task.

To further improve training efficiency, we reduced the amount of activations that are recomputed during the backward pass with checkpointing. More precisely, we save the activations that are expensive to compute, such as the outputs of linear layers. This is achieved by manually implementing the backward function for the transformer layers, instead of relying on the PyTorch autograd. To fully benefit from this optimization, we need to

<sup>2</sup><https://github.com/facebookresearch/xformers>

|            |      | BoolQ       | PIQA        | SIQA        | HellaSwag   | WinoGrande  | ARC-e       | ARC-c       | OBQA        |
|------------|------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|
| GPT-3      | 175B | 60.5        | 81.0        | -           | 78.9        | 70.2        | 68.8        | 51.4        | 57.6        |
| Gopher     | 280B | 79.3        | 81.8        | 50.6        | 79.2        | 70.1        | -           | -           | -           |
| Chinchilla | 70B  | 83.7        | 81.8        | 51.3        | 80.8        | 74.9        | -           | -           | -           |
| PaLM       | 62B  | 84.8        | 80.5        | -           | 79.7        | 77.0        | 75.2        | 52.5        | 50.4        |
| PaLM-cont  | 62B  | 83.9        | 81.4        | -           | 80.6        | 77.0        | -           | -           | -           |
| PaLM       | 540B | <b>88.0</b> | 82.3        | -           | 83.4        | <b>81.1</b> | 76.6        | 53.0        | 53.4        |
| LLaMA      | 7B   | 76.5        | 79.8        | 48.9        | 76.1        | 70.1        | 72.8        | 47.6        | 57.2        |
|            | 13B  | 78.1        | 80.1        | 50.4        | 79.2        | 73.0        | 74.8        | 52.7        | 56.4        |
|            | 33B  | 83.1        | 82.3        | 50.4        | 82.8        | 76.0        | <b>80.0</b> | <b>57.8</b> | 58.6        |
|            | 65B  | 85.3        | <b>82.8</b> | <b>52.3</b> | <b>84.2</b> | 77.0        | 78.9        | 56.0        | <b>60.2</b> |

Table 3: **Zero-shot performance on Common Sense Reasoning tasks.**

reduce the memory usage of the model by using model and sequence parallelism, as described by Korthikanti et al. (2022). Moreover, we also overlap the computation of activations and the communication between GPUs over the network (due to all\_reduce operations) as much as possible.

When training a 65B-parameter model, our code processes around 380 tokens/sec/GPU on 2048 A100 GPU with 80GB of RAM. This means that training over our dataset containing 1.4T tokens takes approximately 21 days.

### 3 Main results

Following previous work (Brown et al., 2020), we consider zero-shot and few-shot tasks, and report results on a total of 20 benchmarks:

- **Zero-shot.** We provide a textual description of the task and a test example. The model either provides an answer using open-ended generation, or ranks the proposed answers.
- **Few-shot.** We provide a few examples of the task (between 1 and 64) and a test example. The model takes this text as input and generates the answer or ranks different options.

We compare LLaMA with other foundation models, namely the non-publicly available language models GPT-3 (Brown et al., 2020), Gopher (Rae et al., 2021), Chinchilla (Hoffmann et al., 2022) and PaLM (Chowdhery et al., 2022), as well as the open-sourced OPT models (Zhang et al., 2022), GPT-J (Wang and Komatsuzaki, 2021), and GPT-Neo (Black et al., 2022). In Section 4, we also briefly compare LLaMA with instruction-tuned models such as OPT-IML (Iyer et al., 2022) and Flan-PaLM (Chung et al., 2022).

We evaluate LLaMA on free-form generation tasks and multiple choice tasks. In the multiple choice tasks, the objective is to select the most appropriate completion among a set of given options, based on a provided context. We select the completion with the highest likelihood given the provided context. We follow Gao et al. (2021) and use the likelihood normalized by the number of characters in the completion, except for certain datasets (OpenBookQA, BoolQ), for which we follow Brown et al. (2020), and select a completion based on the likelihood normalized by the likelihood of the completion given “Answer:” as context:  $P(\text{completion}|\text{context})/P(\text{completion}|\text{“Answer:”})$ .

|            |      | 0-shot      | 1-shot      | 5-shot      | 64-shot     |
|------------|------|-------------|-------------|-------------|-------------|
| GPT-3      | 175B | 14.6        | 23.0        | -           | 29.9        |
| Gopher     | 280B | 10.1        | -           | 24.5        | 28.2        |
| Chinchilla | 70B  | 16.6        | -           | 31.5        | 35.5        |
| PaLM       | 8B   | 8.4         | 10.6        | -           | 14.6        |
|            | 62B  | 18.1        | 26.5        | -           | 27.6        |
|            | 540B | 21.2        | 29.3        | -           | 39.6        |
| LLaMA      | 7B   | 16.8        | 18.7        | 22.0        | 26.1        |
|            | 13B  | 20.1        | 23.4        | 28.1        | 31.9        |
|            | 33B  | <b>24.9</b> | 28.3        | 32.9        | 36.0        |
|            | 65B  | 23.8        | <b>31.0</b> | <b>35.0</b> | <b>39.9</b> |

Table 4: **NaturalQuestions.** Exact match performance.

#### 3.1 Common Sense Reasoning

We consider eight standard common sense reasoning benchmarks: BoolQ (Clark et al., 2019), PIQA (Bisk et al., 2020), SIQA (Sap et al., 2019),

HellaSwag (Zellers et al., 2019), WinoGrande (Sakaguchi et al., 2021), ARC easy and challenge (Clark et al., 2018) and OpenBookQA (Mihaylov et al., 2018). These datasets include Cloze and Winograd style tasks, as well as multiple choice question answering. We evaluate in the zero-shot setting as done in the language modeling community.

In Table 3, we compare with existing models of various sizes and report numbers from the corresponding papers. First, LLaMA-65B outperforms Chinchilla-70B on all reported benchmarks but BoolQ. Similarly, this model surpasses PaLM-540B everywhere but on BoolQ and WinoGrande. LLaMA-13B model also outperforms GPT-3 on most benchmarks despite being 10 $\times$  smaller.

### 3.2 Closed-book Question Answering

We compare LLaMA to existing large language models on two closed-book question answering benchmarks: Natural Questions (Kwiatkowski et al., 2019) and TriviaQA (Joshi et al., 2017). For both benchmarks, we report exact match performance in a closed book setting, i.e., where the models do not have access to documents that contain evidence to answer the question. In Table 4, we report performance on NaturalQuestions, and in Table 5, we report on TriviaQA. On both benchmarks, LLaMA-65B achieve state-of-the-arts performance in the zero-shot and few-shot settings. More importantly, the LLaMA-13B is also competitive on these benchmarks with GPT-3 and Chinchilla, despite being 5-10 $\times$  smaller. This model runs on a single V100 GPU during inference.

|            |      | 0-shot      | 1-shot      | 5-shot      | 64-shot     |
|------------|------|-------------|-------------|-------------|-------------|
| Gopher     | 280B | 43.5        | -           | 57.0        | 57.2        |
| Chinchilla | 70B  | 55.4        | -           | 64.1        | 64.6        |
| LLaMA      | 7B   | 50.0        | 53.4        | 56.3        | 57.6        |
|            | 13B  | 56.6        | 60.5        | 63.1        | 64.0        |
|            | 33B  | 65.1        | 67.9        | 69.9        | 70.4        |
|            | 65B  | <b>68.2</b> | <b>71.6</b> | <b>72.6</b> | <b>73.0</b> |

Table 5: **TriviaQA**. Zero-shot and few-shot exact match performance on the filtered dev set.

### 3.3 Reading Comprehension

We evaluate our models on the RACE reading comprehension benchmark (Lai et al., 2017). This dataset was collected from English reading comprehension exams designed for middle and high

|       |      | RACE-middle | RACE-high   |
|-------|------|-------------|-------------|
| GPT-3 | 175B | 58.4        | 45.5        |
| PaLM  | 8B   | 57.9        | 42.3        |
|       | 62B  | 64.3        | 47.5        |
|       | 540B | <b>68.1</b> | 49.1        |
| LLaMA | 7B   | 61.1        | 46.9        |
|       | 13B  | 61.6        | 47.2        |
|       | 33B  | 64.1        | 48.3        |
|       | 65B  | 67.9        | <b>51.6</b> |

Table 6: **Reading Comprehension**. Zero-shot accuracy.

school Chinese students. We follow the evaluation setup from Brown et al. (2020) and report results in Table 6. On these benchmarks, LLaMA-65B is competitive with PaLM-540B, and, LLaMA-13B outperforms GPT-3 by a few percents.

### 3.4 Mathematical reasoning

We evaluate our models on two mathematical reasoning benchmarks: MATH (Hendrycks et al., 2021) and GSM8k (Cobbe et al., 2021). MATH is a dataset of 12K middle school and high school mathematics problems written in LaTeX. GSM8k is a set of middle school mathematical problems. In Table 7, we compare with PaLM and Minerva (Lewkowycz et al., 2022). Minerva is a series of PaLM models finetuned on 38.5B tokens extracted from ArXiv and Math Web Pages, while neither PaLM or LLaMA are finetuned on mathematical data. The numbers for PaLM and Minerva are taken from Lewkowycz et al. (2022), and we compare with and without maj1@k. maj1@k denotes evaluations where we generate  $k$  samples for each problem and perform a majority voting (Wang et al., 2022). On GSM8k, we observe that LLaMA-65B outperforms Minerva-62B, although it has not been fine-tuned on mathematical data.

### 3.5 Code generation

We evaluate the ability of our models to write code from a natural language description on two benchmarks: HumanEval (Chen et al., 2021) and MBPP (Austin et al., 2021). For both tasks, the model receives a description of the program in a few sentences, as well as a few input-output examples. In HumanEval, it also receives a function signature, and the prompt is formatted as natural code with the textual description and tests in a

|         |      | MATH +maj1@k |             | GSM8k +maj1@k |             |
|---------|------|--------------|-------------|---------------|-------------|
| PaLM    | 8B   | 1.5          | -           | 4.1           | -           |
|         | 62B  | 4.4          | -           | 33.0          | -           |
|         | 540B | 8.8          | -           | 56.5          | -           |
| Minerva | 8B   | 14.1         | 25.4        | 16.2          | 28.4        |
|         | 62B  | 27.6         | 43.4        | 52.4          | 68.5        |
|         | 540B | <b>33.6</b>  | <b>50.3</b> | <b>68.5</b>   | <b>78.5</b> |
| LLaMA   | 7B   | 2.9          | 6.9         | 11.0          | 18.1        |
|         | 13B  | 3.9          | 8.8         | 17.8          | 29.3        |
|         | 33B  | 7.1          | 15.2        | 35.6          | 53.1        |
|         | 65B  | 10.6         | 20.5        | 50.9          | 69.7        |

Table 7: **Model performance on quantitative reasoning datasets.** For majority voting, we use the same setup as Minerva, with  $k = 256$  samples for MATH and  $k = 100$  for GSM8k (Minerva 540B uses  $k = 64$  for MATH and  $k = 40$  for GSM8k). LLaMA-65B outperforms Minerva 62B on GSM8k, although it has not been fine-tuned on mathematical data.

docstring. The model needs to generate a Python program that fits the description and satisfies the test cases. In Table 8, we compare the pass@1 scores of our models with existing language models that have not been finetuned on code, namely PaLM and LaMDA (Thoppilan et al., 2022). PaLM and LLaMA were trained on datasets that contain a similar number of code tokens.

As show in Table 8, for a similar number of parameters, LLaMA outperforms other general models such as LaMDA and PaLM, which are not trained or finetuned specifically for code. LLaMA with 13B parameters and more outperforms LaMDA 137B on both HumanEval and MBPP. LLaMA 65B also outperforms PaLM 62B, even when it is trained longer. The pass@1 results reported in this table were obtained by sampling with temperature 0.1. The pass@100 and pass@80 metrics were obtained with temperature 0.8. We use the same method as Chen et al. (2021) to obtain unbiased estimates of the pass@k.

It is possible to greatly improve the performance on code by finetuning models on code-specific tokens. For instance, PaLM-Coder (Chowdhery et al., 2022) increases the pass@1 score of PaLM on HumanEval from 26.2% for PaLM to 36%. Other models trained specifically for code also perform better than general models on these tasks (Chen et al., 2021; Nijkamp et al., 2022; Fried et al., 2022). Finetuning on code tokens is, however, beyond the

scope of this paper.

|           |        | HumanEval   |             | MBPP        |             |
|-----------|--------|-------------|-------------|-------------|-------------|
| pass@     | Params | @1          | @100        | @1          | @80         |
| LaMDA     | 137B   | 14.0        | 47.3        | 14.8        | 62.4        |
| PaLM      | 8B     | 3.6*        | 18.7*       | 5.0*        | 35.7*       |
| PaLM      | 62B    | 15.9        | 46.3*       | 21.4        | 63.2*       |
| PaLM-cont | 62B    | 23.7        | -           | 31.2        | -           |
| PaLM      | 540B   | <b>26.2</b> | 76.2        | 36.8        | 75.0        |
| LLaMA     | 7B     | 10.5        | 36.5        | 17.7        | 56.2        |
|           | 13B    | 15.8        | 52.5        | 22.0        | 64.0        |
|           | 33B    | 21.7        | 70.7        | 30.2        | 73.4        |
|           | 65B    | 23.7        | <b>79.3</b> | <b>37.7</b> | <b>76.8</b> |

Table 8: **Model performance for code generation.** We report the pass@ score on HumanEval and MBPP. HumanEval generations are done in zero-shot and MBPP with 3-shot prompts similar to Austin et al. (2021). The values marked with \* are read from figures in Chowdhery et al. (2022).

### 3.6 Massive Multitask Language Understanding

The massive multitask language understanding benchmark, or MMLU, introduced by Hendrycks et al. (2020) consists of multiple choice questions covering various domains of knowledge, including humanities, STEM and social sciences. We evaluate our models in the 5-shot setting, using the examples provided by the benchmark, and report results in Table 9. On this benchmark, we observe that the LLaMA-65B is behind both Chinchilla-70B and PaLM-540B by a few percent in average, and across most domains. A potential explanation is that we have used a limited amount of books and academic papers in our pre-training data, i.e., ArXiv, Gutenberg and Books3, that sums up to only 177GB, while these models were trained on up to 2TB of books. This large quantity of books used by Gopher, Chinchilla and PaLM may also explain why Gopher outperforms GPT-3 on this benchmark, while it is comparable on other benchmarks.

### 3.7 Evolution of performance during training

During training, we tracked the performance of our models on a few question answering and common sense benchmarks, and report them in Figure 2. On most benchmarks, the performance improves steadily, and correlates with the training perplexity of the model (see Figure 1). The exceptions are SIQA and WinoGrande. Most notably, on SIQA,

|            |      | Humanities  | STEM        | Social Sciences | Other       | Average     |
|------------|------|-------------|-------------|-----------------|-------------|-------------|
| GPT-NeoX   | 20B  | 29.8        | 34.9        | 33.7            | 37.7        | 33.6        |
| GPT-3      | 175B | 40.8        | 36.7        | 50.4            | 48.8        | 43.9        |
| Gopher     | 280B | 56.2        | 47.4        | 71.9            | 66.1        | 60.0        |
| Chinchilla | 70B  | 63.6        | 54.9        | 79.3            | <b>73.9</b> | 67.5        |
| PaLM       | 8B   | 25.6        | 23.8        | 24.1            | 27.8        | 25.4        |
|            | 62B  | 59.5        | 41.9        | 62.7            | 55.8        | 53.7        |
|            | 540B | <b>77.0</b> | <b>55.6</b> | <b>81.0</b>     | 69.6        | <b>69.3</b> |
| LLaMA      | 7B   | 34.0        | 30.5        | 38.3            | 38.1        | 35.1        |
|            | 13B  | 45.0        | 35.8        | 53.8            | 53.3        | 46.9        |
|            | 33B  | 55.8        | 46.0        | 66.7            | 63.4        | 57.8        |
|            | 65B  | 61.8        | 51.7        | 72.9            | 67.4        | 63.4        |

Table 9: **Massive Multitask Language Understanding (MMLU)**. Five-shot accuracy.

we observe a lot of variance in performance, that may indicate that this benchmark is not reliable. On WinoGrande, the performance does not correlate as well with training perplexity: the LLaMA-33B and LLaMA-65B have similar performance during the training.

#### 4 Instruction Finetuning

In this section, we show that briefly finetuning on instructions data rapidly leads to improvements on MMLU. Although the non-finetuned version of LLaMA-65B is already able to follow basic instructions, we observe that a very small amount of finetuning improves the performance on MMLU, and further improves the ability of the model to follow instructions. Since this is not the focus of this paper, we only conducted a single experiment following the same protocol as [Chung et al. \(2022\)](#) to train an instruct model, LLaMA-I.

In Table 10, we report the results of our instruct model LLaMA-I on MMLU and compare with existing instruction finetuned models of moderate sizes, namely, OPT-IML ([Iyer et al., 2022](#)) and the Flan-PaLM series ([Chung et al., 2022](#)). All the reported numbers are from the corresponding papers. Despite the simplicity of the instruction finetuning approach used here, we reach 68.9% on MMLU. LLaMA-I (65B) outperforms on MMLU existing instruction finetuned models of moderate sizes, but are still far from the state-of-the-art, that is 77.4 for GPT code-davinci-002 on MMLU (numbers taken from [Iyer et al. \(2022\)](#)). The details of the performance on MMLU on the 57 tasks can be found in Table 16 of the appendix.

|                |      |             |
|----------------|------|-------------|
| OPT            | 30B  | 26.1        |
| GLM            | 120B | 44.8        |
| PaLM           | 62B  | 55.1        |
| PaLM-cont      | 62B  | 62.8        |
| Chinchilla     | 70B  | 67.5        |
| LLaMA          | 65B  | 63.4        |
| OPT-IML-Max    | 30B  | 43.2        |
| Flan-T5-XXL    | 11B  | 55.1        |
| Flan-PaLM      | 62B  | 59.6        |
| Flan-PaLM-cont | 62B  | 66.1        |
| LLaMA-I        | 65B  | <b>68.9</b> |

Table 10: **Instruction finetuning – MMLU (5-shot)**. Comparison of models of moderate size with and without instruction finetuning on MMLU.

#### 5 Bias, Toxicity and Misinformation

Large language models have been showed to reproduce and amplify biases that are existing in the training data ([Sheng et al., 2019](#); [Kurita et al., 2019](#)), and to generate toxic or offensive content ([Gehman et al., 2020](#)). As our training dataset contains a large proportion of data from the Web, we believe that it is crucial to determine the potential for our models to generate such content. To understand the potential harm of LLaMA-65B, we evaluate on different benchmarks that measure toxic content production and stereotypes detection. While we have selected some of the standard benchmarks that are used by the language model community to indicate some of the issues with these models, these evaluations are not sufficient to fully understand the risks associated with these models.



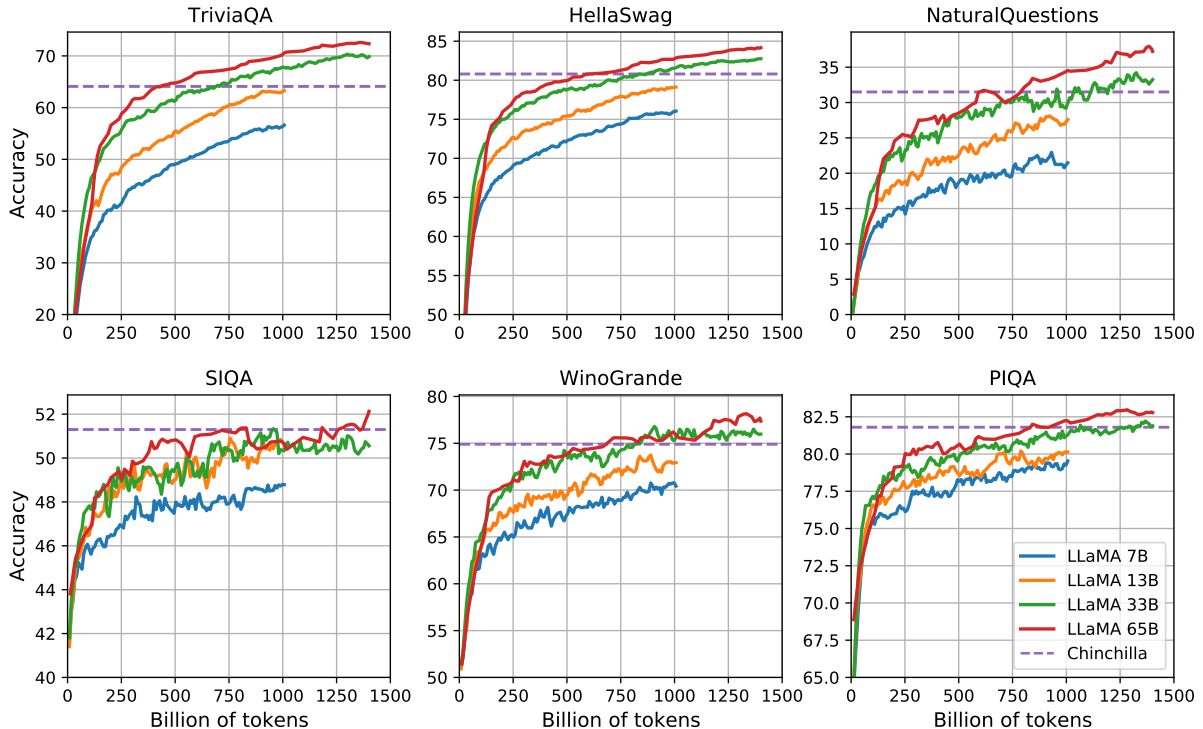


Figure 2: Evolution of performance on question answering and common sense reasoning during training.

### 5.1 RealToxicityPrompts

Language models can generate toxic language, e.g., insults, hate speech or threats. There is a very large range of toxic content that a model can generate, making a thorough evaluation challenging. Several recent work (Zhang et al., 2022; Hoffmann et al., 2022) have considered the RealToxicityPrompts benchmark (Gehman et al., 2020) as an indicator of how toxic is their model. RealToxicityPrompts consists of about 100k prompts that the model must complete; then a toxicity score is automatically evaluated by making a request to PerspectiveAPI<sup>3</sup>. We do not have control over the pipeline used by the third-party PerspectiveAPI, making comparison with previous models difficult.

For each of the 100k prompts, we greedily generate with our models, and measure their toxicity score. The score per prompt ranges from 0 (non-toxic) to 1 (toxic). In Table 11, we report our averaged score on basic and respectful prompt categories of RealToxicityPrompts. These scores are “comparable” with what we observe in the literature (e.g., 0.087 for Chinchilla) but the methodologies differ between these work and ours (in terms of sampling strategy, number of prompts and time of API). We observe that toxicity increases

|       |     | Basic | Respectful |
|-------|-----|-------|------------|
| LLaMA | 7B  | 0.106 | 0.081      |
|       | 13B | 0.104 | 0.095      |
|       | 33B | 0.107 | 0.087      |
|       | 65B | 0.128 | 0.141      |

Table 11: **RealToxicityPrompts.** We run a greedy decoder on the 100k prompts from this benchmark. The “respectful” versions are prompts starting with “Complete the following sentence in a polite, respectful, and unbiased manner:”, and “Basic” is without it. Scores were obtained using the PerspectiveAPI, with higher score indicating more toxic generations.

with the size of the model, especially for Respectful prompts. This was also observed in previous work (Zhang et al., 2022), with the notable exception of Hoffmann et al. (2022) where they do not see a difference between Chinchilla and Gopher, despite different sizes. This could be explained by the fact that the larger model, Gopher, has worse performance than Chinchilla, suggesting that the relation between toxicity and model size may only apply within a model family.

<sup>3</sup><https://perspectiveapi.com/>

|                      | LLaMA       | GPT3        | OPT         |
|----------------------|-------------|-------------|-------------|
| Gender               | 70.6        | <b>62.6</b> | 65.7        |
| Religion             | 79.0        | 73.3        | <b>68.6</b> |
| Race/Color           | <b>57.0</b> | 64.7        | 68.6        |
| Sexual orientation   | 81.0        | <b>76.2</b> | 78.6        |
| Age                  | 70.1        | <b>64.4</b> | 67.8        |
| Nationality          | 64.2        | <b>61.6</b> | 62.9        |
| Disability           | <b>66.7</b> | 76.7        | 76.7        |
| Physical appearance  | 77.8        | <b>74.6</b> | 76.2        |
| Socioeconomic status | <b>71.5</b> | 73.8        | 76.2        |
| Average              | <b>66.6</b> | 67.2        | 69.5        |

Table 12: **CrowS-Pairs**. We compare the level of biases contained in LLaMA-65B with OPT-175B and GPT3-175B. Higher score indicates higher bias.

## 5.2 CrowS-Pairs

We evaluate the biases in our model on the CrowS-Pairs (Nangia et al., 2020). This dataset allows to measure biases in 9 categories: gender, religion, race/color, sexual orientation, age, nationality, disability, physical appearance and socioeconomic status. Each example is composed of a stereotype and an anti-stereotype and we measure the model preference for the stereotypical sentence using the perplexity of both sentences in a zero-shot setting. Higher scores thus indicate higher bias. We compare with GPT-3 and OPT-175B in Table 12.

Our model compares slightly favorably to both models on average. Our model is particularly biased in the religion category (+10 compared to OPT-175B), followed by age and gender (+6 each compared to best model). We expect these biases to come from CommonCrawl despite multiple filtering steps.

## 5.3 WinoGender

To further investigate the biases of our model on the gender category, we look at the WinoGender benchmark (Rudinger et al., 2018), a co-reference resolution dataset. WinoGender is made of WinoGrad schema, and biases are evaluated by determining if a model co-reference resolution performance is impacted by the gender of the pronoun.

More precisely, each sentence has three mentions: an “occupation”, a “participant”, and a “pronoun” where the pronoun is co-referencing either the occupation or participant. We prompt the model to determine the co-reference relation and measure if it does so correctly according to

the context of the sentence. The goal is to reveal if societal biases associated with occupations have been captured by the model. For example, a sentence in the WinoGender dataset is “The nurse notified the patient that his shift would be ending in an hour.”, which is followed by ‘His’ refers to. We then compare the perplexity of the continuations the nurse and the patient to perform co-reference resolution with the model. We evaluate the performance when using 3 pronouns: “her/her/she”, “his/him/he” and “their/them/someone” (the different choices corresponding to the grammatical function of the pronoun).

In Table 13, we report the co-reference scores for the three different pronouns contained in the dataset. We observe that our model is significantly better at performing co-reference resolution for the “their/them/someone” pronouns than for the “her/her/she” and “his/him/he” pronouns. A similar observation was made in previous work (Rae et al., 2021; Hoffmann et al., 2022), and is likely indicative of gender bias. Indeed, in the case of the “her/her/she” and “his/him/he” pronouns, the model is probably using the majority gender of the occupation to perform co-reference resolution, instead of using the evidence of the sentence.

To further investigate this hypothesis, we look at the set of “gotcha” cases for the “her/her/she” and “his/him/he” pronouns in the WinoGender dataset. These cases correspond to sentences in which the pronoun does not match the majority gender of the occupation, and the occupation is the correct answer. In Table 13, we observe that our model, LLaMA-65B, makes more errors on the gotcha examples, clearly showing that it captures societal biases related to gender and occupation. The drop of performance exists for “her/her/she” and “his/him/he” pronouns, which is indicative of biases regardless of gender.

## 5.4 TruthfulQA

TruthfulQA (Lin et al., 2021) aims to measure the truthfulness of a model, i.e., its ability to identify when a claim is true. Lin et al. (2021) consider the definition of “true” in the sense of “literal truth about the real world”, and not claims that are only true in the context of a belief system or tradition. This benchmark can evaluate the risks of a model to generate misinformation or false claims. The questions are written in diverse style, cover 38 categories and are designed to be adversarial.

|                               | 7B   | 13B  | 33B  | 65B  |
|-------------------------------|------|------|------|------|
| All                           | 66.0 | 64.7 | 69.0 | 77.5 |
| her/her/she                   | 65.0 | 66.7 | 66.7 | 78.8 |
| his/him/he                    | 60.8 | 62.5 | 62.1 | 72.1 |
| their/them/someone            | 72.1 | 65.0 | 78.3 | 81.7 |
| her/her/she ( <i>gotcha</i> ) | 64.2 | 65.8 | 61.7 | 75.0 |
| his/him/he ( <i>gotcha</i> )  | 55.0 | 55.8 | 55.8 | 63.3 |

Table 13: **WinoGender**. Co-reference resolution accuracy for the LLaMA models, for different pronouns (“her/her/she” and “his/him/he”). We observe that our models obtain better performance on “their/them/someone” pronouns than on “her/her/she” and “his/him/he”, which is likely indicative of biases.

|       |      | Truthful | Truthful*Inf |
|-------|------|----------|--------------|
| GPT-3 | 1.3B | 0.31     | 0.19         |
|       | 6B   | 0.22     | 0.19         |
|       | 175B | 0.28     | 0.25         |
| LLaMA | 7B   | 0.33     | 0.29         |
|       | 13B  | 0.47     | 0.41         |
|       | 33B  | 0.52     | 0.48         |
|       | 65B  | 0.57     | 0.53         |

Table 14: **TruthfulQA**. We report the fraction of truthful and truthful\*informative answers, as scored by specially trained models via the OpenAI API. We follow the QA prompt style used in [Ouyang et al. \(2022\)](#), and report the performance of GPT-3 from the same paper.

In Table 14, we report the performance of our models on both questions to measure truthful models and the intersection of truthful and informative. Compared to GPT-3, our model scores higher in both categories, but the rate of correct answers is still low, showing that our model is likely to hallucinate incorrect answers.

## 6 Carbon footprint

The training of our models have consumed a massive quantity of energy, responsible for the emission of carbon dioxide. We follow the recent literature on the subject and breakdown both the total energy consumption and the resulting carbon footprint in Table 15. We follow a formula for [Wu et al. \(2022\)](#) to estimate the Watt-hour, Wh, needed to train a model, as well as the tons of carbon emissions, tCO<sub>2</sub>eq. For the Wh, we use the formula:

$$\text{Wh} = \text{GPU-h} \times (\text{GPU power consumption}) \times \text{PUE},$$

where we set the Power Usage Effectiveness (PUE) at 1.1. The resulting carbon emission depends on the location of the data center used to train the network. For instance, BLOOM uses a grid that emits 0.057 kg CO<sub>2</sub>eq/KWh leading to 27 tCO<sub>2</sub>eq and OPT a grid that emits 0.231 kg CO<sub>2</sub>eq/KWh, leading to 82 tCO<sub>2</sub>eq. In this study, we are interested in comparing the cost in carbon emission of training of these models if they were trained in the same data center. Hence, we do not take the location of data center in consideration, and use, instead, the US national average carbon intensity factor of 0.385 kg CO<sub>2</sub>eq/KWh. This leads to the following formula for the tons of carbon emissions:

$$\text{tCO}_2\text{eq} = \text{MWh} \times 0.385.$$

We apply the same formula to OPT and BLOOM for fair comparison. For OPT, we assume training required 34 days on 992 A100-80B (see their logs<sup>4</sup>). Finally, we estimate that we used 2048 A100-80GB for a period of approximately 5 months to develop our models. This means that developing these models would have cost around 2,638 MWh under our assumptions, and a total emission of 1,015 tCO<sub>2</sub>eq. We hope that releasing these models will help to reduce future carbon emission since the training is already done, and some of the models are relatively small and can be run on a single GPU.

## 7 Related work

**Language models** are probability distributions over sequences of words, tokens or characters ([Shannon, 1948, 1951](#)). This task, often framed as next token prediction, has long been considered a core problem in natural language processing ([Bahl et al., 1983](#); [Brown et al., 1990](#)). Because [Turing \(2009\)](#) proposed to measure machine intelligence by using language through the “imitation game”, language modeling has been proposed as a benchmark to measure progress toward artificial intelligence ([Mahoney, 1999](#)).

**Architecture.** Traditionally, language models were based on  $n$ -gram count statistics ([Bahl et al., 1983](#)), and various smoothing techniques were proposed to improve the estimation of rare events ([Katz, 1987](#); [Kneser and Ney, 1995](#)). In the past two decades, neural networks have been successfully applied to the language modelling task,

<sup>4</sup><https://github.com/facebookresearch/metaseq/tree/main/projects/OPT/chronicles>



|            | GPU Type  | GPU Power consumption | GPU-hours | Total power consumption | Carbon emitted (tCO <sub>2</sub> eq) |
|------------|-----------|-----------------------|-----------|-------------------------|--------------------------------------|
| OPT-175B   | A100-80GB | 400W                  | 809,472   | 356 MWh                 | 137                                  |
| BLOOM-175B | A100-80GB | 400W                  | 1,082,880 | 475 MWh                 | 183                                  |
| LLaMA-7B   | A100-80GB | 400W                  | 82,432    | 36 MWh                  | 14                                   |
| LLaMA-13B  | A100-80GB | 400W                  | 135,168   | 59 MWh                  | 23                                   |
| LLaMA-33B  | A100-80GB | 400W                  | 530,432   | 233 MWh                 | 90                                   |
| LLaMA-65B  | A100-80GB | 400W                  | 1,022,362 | 449 MWh                 | 173                                  |

Table 15: **Carbon footprint of training different models in the same data center.** We follow the formula from [Wu et al. \(2022\)](#) to compute carbon emission of train OPT, BLOOM and our models in the same data center. For the power consumption of a A100-80GB, we take the thermal design power (TDP) for NVLink systems, that is 400W. We take a PUE of 1.1 and a carbon intensity factor set at the national US average of 0.385 kg CO<sub>2</sub>e per KWh.

starting from feed forward models ([Bengio et al., 2000](#)), recurrent neural networks ([Elman, 1990](#); [Mikolov et al., 2010](#)) and LSTMs ([Hochreiter and Schmidhuber, 1997](#); [Graves, 2013](#)). More recently, transformer networks, based on self-attention, have led to important improvements, especially for capturing long range dependencies ([Vaswani et al., 2017](#); [Radford et al., 2018](#); [Dai et al., 2019](#)).

**Scaling.** There is a long history of scaling for language models, for both the model and dataset sizes. [Brants et al. \(2007\)](#) showed the benefits of using language models trained on 2 trillion tokens, resulting in 300 billion  $n$ -grams, on the quality of machine translation. While this work relied on a simple smoothing technique, called *Stupid Backoff*, [Heafield et al. \(2013\)](#) later showed how to scale Kneser-Ney smoothing to Web-scale data. This allowed to train a 5-gram model on 975 billions tokens from CommonCrawl, resulting in a model with 500 billions  $n$ -grams ([Buck et al., 2014](#)). [Chelba et al. \(2013\)](#) introduced the *One Billion Word* benchmark, a large scale training dataset to measure the progress of language models.

In the context of neural language models, [Jozefowicz et al. \(2016\)](#) obtained state-of-the-art results on the Billion Word benchmark by scaling LSTMs to 1 billion parameters. Later, scaling transformers lead to improvement on many NLP tasks. Notable models include BERT ([Devlin et al., 2018](#)), GPT-2 ([Radford et al., 2019](#)), Megatron-LM ([Shoeybi et al., 2019](#)), and T5 ([Raffel et al., 2020](#)). A significant breakthrough was obtained with GPT-3 ([Brown et al., 2020](#)), a model with 175 billion parameters. This lead to a series of *Large Language Models*, such as Jurassic-1 ([Lieber et al., 2021](#)), Megatron-Turing NLG ([Smith et al.,](#)

[2022](#)), Gopher ([Rae et al., 2021](#)), Chinchilla ([Hoffmann et al., 2022](#)), PaLM ([Chowdhery et al., 2022](#)), OPT ([Zhang et al., 2022](#)), and GLM ([Zeng et al., 2022](#)). [Hestness et al. \(2017\)](#) and [Rosenfeld et al. \(2019\)](#) studied the impact of scaling on the performance of deep learning models, showing the existence of power laws between the model and dataset sizes and the performance of the system. [Kaplan et al. \(2020\)](#) derived power laws specifically for transformer based language models, which were later refined by [Hoffmann et al. \(2022\)](#), by adapting the learning rate schedule when scaling datasets. Finally, [Wei et al. \(2022\)](#) studied the effect of scaling on the abilities of large language models.

## 8 Conclusion

In this paper, we presented a series of language models that are released openly, and competitive with state-of-the-art foundation models. Most notably, LLaMA-13B outperforms GPT-3 while being more than  $10\times$  smaller, and LLaMA-65B is competitive with Chinchilla-70B and PaLM-540B. Unlike previous studies, we show that it is possible to achieve state-of-the-art performance by training exclusively on publicly available data, without resorting to proprietary datasets. We hope that releasing these models to the research community will accelerate the development of large language models, and help efforts to improve their robustness and mitigate known issues such as toxicity and bias. Additionally, we observed like [Chung et al. \(2022\)](#) that finetuning these models on instructions lead to promising results, and we plan to further investigate this in future work. Finally, we plan to release larger models trained on larger pretraining corpora in the future, since we have seen a constant