Leveraging LLMs with RAG to recommend Points of Interest to tourists

Student Patrick Hamzaj
Supervisor Niccolò Marastoni

March 27, 2025

Università degli Studi di Verona

Table of contents

Introduction

Objective

The objective of this project is to deliver a practical implementation of an LLM-powered application to enhance **tourism experience** in Verona, Italy.

LLM = Large Language Model

Modern Large Language Models require large computational resources and data to train and inference.

Parameter Count

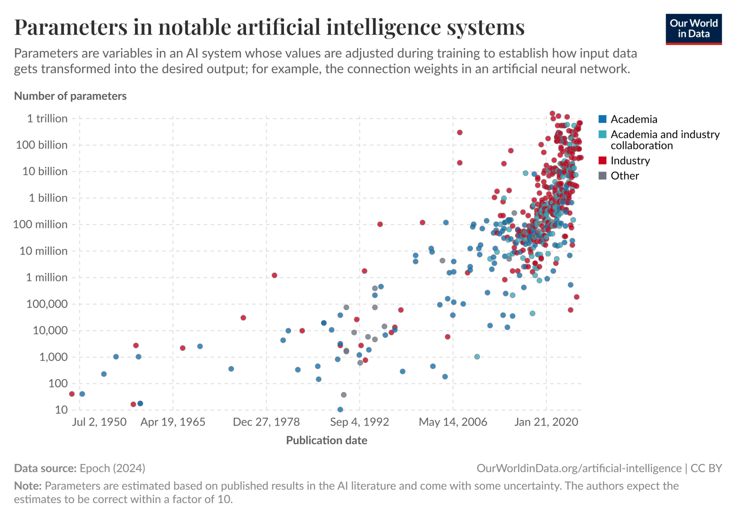

Figure 1: Parameters in LLMs, as seen in [2]

Challenges

- The number of model parameters

Challenges

- The number of model parameters
- The size of the dataset

Challenges

- The number of model parameters
- The size of the dataset
- Amount of compute resources

Scaling Law

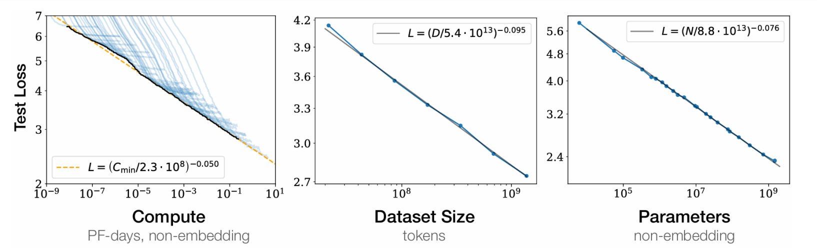

Figure 2: Scaling Law, as seen in [4]

There was a need to harness the generative power of LLMs in a limited resource environemnt and thus avoiding fine-tuning.

Theoretical Background and Glossary

Transformers

Large Language Models are based on the Transformers architecture–a type of *Neural Network*–which must undergo two learning phases:

1. **Pre-Training**
2. **Fine-Tuning**

Pre-training

The model learns general linguistic patterns, facts and knowledge from a vast corpus of text.

Fine-tuning

A further training on a smaller, task-specific dataset to improve performance for particular and domain-specific applications.

Few-Shot Capabilities

Modern open-source models come already pre-trained and fine-tuned, letting developers focus on domain-specific performance.

Alternative approaches have rised exploiting **few-shot capabilities** demonstrated by the introduction of GPT-3. [1]

In-Context Learning

The ability of a language model to perform new tasks by leveraging examples provided directly within the prompt, rather than through explicit parameter updates.

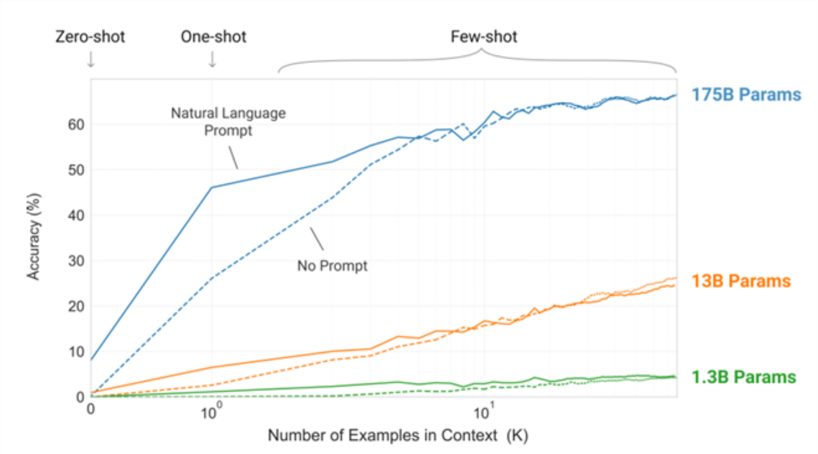

Figure 3: Few-shot capablities, as seen in [1]

Beyond Fine-Tuning

Fine-Tuning remains a common approach to tailor LLMs' behavior for specific use cases, but innovative techniques have emerged gathering already fine-tuned models for general scenarios (i.e., chat models), directing its focus on specific tasks and use cases:

- Prompt Engineering

Beyond Fine-Tuning

Fine-Tuning remains a common approach to tailor LLMs' behavior for specific use cases, but innovative techniques have emerged gathering already fine-tuned models for general scenarios (i.e., chat models), directing its focus on specific tasks and use cases:

- Prompt Engineering
- Retrieval Augmented Generation

Beyond Fine-Tuning

Fine-Tuning remains a common approach to tailor LLMs' behavior for specific use cases, but innovative techniques have emerged gathering already fine-tuned models for general scenarios (i.e., chat models), directing its focus on specific tasks and use cases:

- Prompt Engineering
- Retrieval Augmented Generation
- Agentic AI

Prompt Engineering

Prompt Engineering

Grown in importance with the rise of *instruction-tuned* models, Prompt Engineering is defined as the process of designing and structuring instructions to guide LLMs toward producing the most effetive outputs without odifying the models' internal parameters.

Role Prompting

The practice of explicitly instructing a model to adopt a specific persona or role within the prompt.

You are a tour guide assisting tourists in Verona, Italy. Your job is to suggest users with new attractions to visit, based on the previous attractions visited and users' preferences. Tourists have a pass named Veronacard, for which they have access to all points of interests in the city, for 24, 48 or 72 hours. Your tone is both professional and friendly at the same time.

Chain-of-Thought

A novel method to enhance the reasoning capabilities of LLMs by encouraging them to generate itermediate steps before arriving at a final answer.

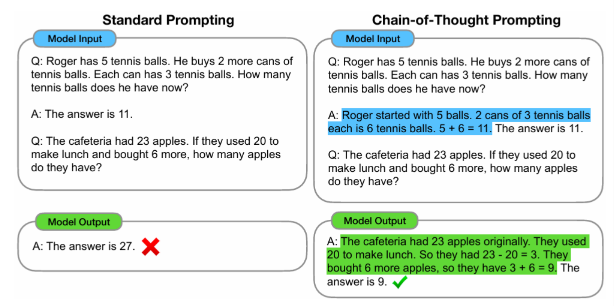

Figure 4: Chain-of-Thought, as seen in [2]

A model’s knowledge ends at the moment of its training–so it will not know about more recent events, current literature or real-time information.

Retrieval Augmented Generation

Retrieval Augmented Generation

A framework dynamically integrates specific,contextually relevant knowledge into the generation process, leading to more informed and accurate outputs

`datetime.now()`	*Today's date is Tuesday, 07 January 2025.*
Call to Openmeteo API.	*The weather throughout the day is as follows:* *At 07:00 the temperature is -1°C with overcast. The precipitation probability is 0%.*
Extract Veronacard info.	*You must suggest users on attractions to visit included in the Veronacard, which are the following:* *- Arena* *- Casa Giulietta*

Once context-awareness is achieved, a strategy must be designed to dinamically retrieve information about affluence and weather.

Agents

Agents

Systems that leverage an AI model to interact with its environment in order to achieve a user-defined objective. It combines reasoning, planning and executing actions, extending the capabilities of LLMs enabling them to act autonomously via external tools to fulfill a task.

You can call the function `retrieve_affluency` *when the user asks how crowded is a certain attraction. You can use the function* `get_weather_forecast` *when the user asks about weather forecast in the next days.*

1. Define the set of tools to be called:

1. Define the set of tools to be called:
    - *`retrieve_affluency()`*

1. Define the set of tools to be called:
    - *retrieve_affluency()*
    - *get_weather_forecast()*

1. Define the set of tools to be called:
    - *retrieve_affluency()*
    - *get_weather_forecast()*
2. Make the model aware of the set tool and provide some examples.

1. Define the set of tools to be called:
   - *`retrieve_affluency()`*
   - *`get_weather_forecast()`*
2. Make the model aware of the set tool and provide some examples.
   - ***USER:*** *I would like to vist the Arena today at around 3pm, will it be sunny?*

1. Define the set of tools to be called:
    - *`retrieve_affluency()`*
    - *`get_weather_forecast()`*
2. Make the model aware of the set tool and provide some examples.
    - ***`USER:`*** *I would like to vist the Arena today at around 3pm, will it be sunny?*
    - *`ASSISTANT: get_weather_forecast('27-01-2025 15:00:00')`*

1. Define the set of tools to be called:
    - `retrieve_affluency()`
    - `get_weather_forecast()`
2. Make the model aware of the set tool and provide some examples.
    - ***USER:*** *I would like to vist the Arena today at around 3pm, will it be sunny?*
    - *ASSISTANT:* `get_weather_forecast('27-01-2025 15:00:00')`
    - ***ASSISTANT:*** *It looks like this afternoon will be a great time to visit the Arena, as the temperature is 14 degrees with no clouds.*

Experimental Setup

Choice of architecture

Table 1: Experimental setup

Parameter	Choice
Model	LLaMa 3.1 8b
Version	Instruct
Quantization	8 bit
Source	Hugging Face
Environment	Google Colab
Hardware	T4 GPU
VRAM	16 GB DDR5

Demo

Link to the Streamlit app.

Limitations

Hallucinations refer to instances where the AI generates content that appears factual and coherent but is ungrounded or incorrect. This phenomenon occurs when the model produces information that isn't supported by its training data or external knowledge sources, leading to plausible-sounding yet inaccurate responses. [3]

Next, take a romantic gondola ride along the Adige River and pass under the famous Ponte Pietra, a beautiful Roman bridge. After that, visit the Casa di Giulietta (Juliet's House), a 14th-century house that inspired Shakespeare's famous balcony scene from Romeo and Juliet.

Time and Multi-Turn Restrictions

Figure 5 shows that there is not a correlation between the number of tokens generated and the time required to give an output, though it demonstrates that it can take up to 30 seconds to produce a response, as it can significantly influence the user experience.

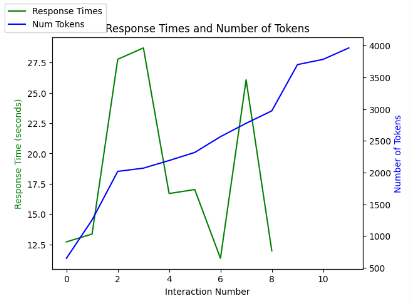

Figure 5: Number of tokens and time required.

Conclusions

The current approach reveals that high computational resources are not strictly necessary for achieving effective and natural reponses.However, the model’s reliance on foundational weights can lead to hallucinations and issue related to context length and memory remains critical.

Conclusions

Pros

- Fast to implement
- Low cost
- Easy to understand

Cons

- Limited flexibility
- Not scalable
- Requires manual updates

- T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei.
  Language models are few-shot learners, 2020.
- C. Giattino, E. Mathieu, V. Samborska, and M. Roser.
  Data page: Parameters in notable artificial intelligence systems.
  `https://ourworldindata.org/grapher/artificial-intelligence-parameter-count`, 2023.
  Part of the publication *Artificial Intelligence*. Data adapted from Epoch. [Online resource].

References ii

Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. J. Bang, A. Madotto, and P. Fung.
Survey of hallucination in natural language generation.
ACM Computing Surveys, 55(12):1–38, Mar. 2023.

J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei.
Scaling laws for neural language models, 2020.