Utilizing Hadoop Ecosystem to Implement RAG for Chain of Thought Reasoning in Language Models

Aviral Pandey (21CS10089)[a], Harsh Sharma (21CS30023)[a], Ishan Raj (21CS10032)[a], Prasanna Paithankar (21CS30065)[a], Utsav Dhanuka (21CS10087)[a]

[a]Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur

Abstract*—*This project proposes integrating* ***Retrieval Augmented Generation (RAG) *into the* ***Hadoop*** *ecosystem to enhance coherence and context retention in large language models' sequential reasoning. Leveraging Hadoop's scalable, distributed computing capabilities, the approach aims to enhance the efficiency and accuracy of natural language understanding and reasoning systems in queries that involve* ***Chain of Thought Reasoning***.

keywords*—*Chain of Thought, RAG, LLMs, Hadoop ecosystem

Introduction

Retrieval augmented generation, or **RAG**, is gaining widespread attention in the Machine Learning community due to its efficiency and efficacy in interacting with language models on custom data. Utilizing RAG has proven to be much less expensive and more accurate in multiple instances compared to fine-tuning. Moreover, language models are documented not to output satisfactorily on prompts that require chained reasoning and knowledge base. To combat this, the industry explores the domain of **Chain of Though Reasoning** in language models. This consists of querying a database or a knowledge graph to retrieve the chain of information, which would be fed as a prompt into the model to output appropriately. On a different line, **Hadoop** ecosystem has provided the industry with robust solutions to manage the data. The sub-components in the ecosystem, namely a few **Solr, MapReduce, Lucene, Zookeeper**, are used to make scalable applications. We in the following sections, highlight our proposed project trajectory in which we integrate all the listed technologies and frameworks to achieve an improved and efficient Chain of Thought Reasoning on any language model without expensive fine-tuning.

Proposal

We propose a system architecture as shown in Fig. 1.

Distributed Database: MapReduce & Indexing using Solr

We'll use Hadoop's MapReduce component to retrieve responses against queries from our distributed database. Efficient indexing and searching is provided by the Apache Solr component. Furthermore, we utilize Zookeeper to manage the deployment overall.

The Language Model

For development purposes, we aim to use the **Llama 2** open-source model. This model has the necessary support to integrate and benchmark prompts and is also maintained by an active trusted community.

Testing & Benchmarking

We lay out two domains to benchmark our system architecture: Query per second (QPS) and Language model accuracy. In the former, we propose to use **Siege, Apache HttpClient** and Python requests module to benchmark the QPS using **ThreadPool/ConnectionPool** multithreading. We manually annotate the comparison between the unaugmented and augmented language model output for the latter.

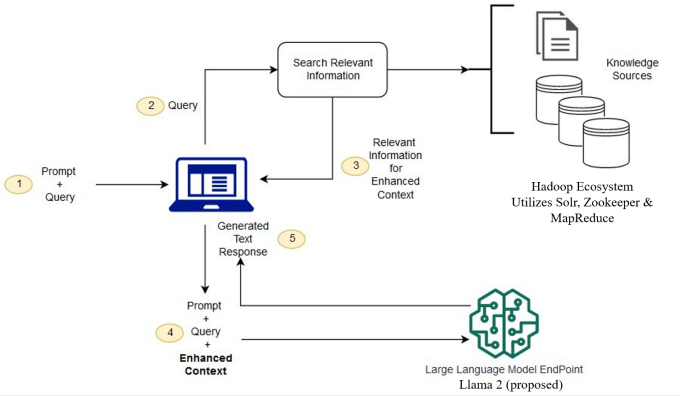

Figure 1. *System design & architecture*

Timeline & Deliverables

We present the following work timeline for the proposed project

- 4 days: Setup and familiarization with Hadoop, Solr, Lucene, Zookeeper, and HBase
- 6 days: Implementation of the LLM API, frontend and integration with distributed database
- 6 days: Scraping custom data from the web and indexing it into the system. Creating Chain of Thought tuples
- 4 days: Buffer time for any unforeseen issues
- 4 days: Testing and Benchmarking/Profiling of the system
- 2 days: Writing the documentation
- 1 day : Final presentation

References

[1] Jason Wei and Xuezhi Wang and Dale Schuurmans and Maarten Bosma and Brian Ichter and Fei Xia and Ed Chi and Quoc Le and Denny Zhou, "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models". (2023) https://arxiv.org/abs/2201.11903

[2] What is RAG? https://aws.amazon.com/what-is/retrieval-augmented-generation/

[3] Augmenting LLM Apps with Vector Databases https://medium.com/@bijit211987/augmenting-llm-apps-with-vector-databases-fe44b774334a

[4] Apache Software Foundation, 2010. Hadoop, Available at: https://hadoop.apache.org.

[5] Touvron, Hugo, et al. "Llama 2: Open foundation and fine-tuned chat models." arXiv preprint arXiv:2307.09288 (2023).