FRI Natural language processing course 2025
Project report

University *of Ljubljana*
Faculty *of Computer and Information Science*

Conversational Agent with Retrieval-Augmented Generation

Blaž Grilj, Ana Poklukar, Kristjan Sever

Abstract

We propose the development of a conversational agent that utilizes Retrieval-Augmented Generation (RAG) to improve the accuracy and relevance of responses by grounding them in real-time external knowledge. Unlike traditional large language models (LLMs) that rely solely on pre-trained data, our envisioned system will retrieve information—primarily from Wikipedia—to enhance its ability to answer general knowledge and current events queries. The project will explore different approaches to model architecture, including the use of open-source LLMs and fine-tuning smaller pre-trained models. Key components will include a Wikipedia-focused retrieval system, a query processing pipeline, and context integration for response generation. To evaluate our approach, we plan to benchmark the RAG-enhanced system against a non-augmented baseline using metrics such as factual accuracy, hallucination rate, and response relevance. This proposal outlines the initial design and objectives of a system aimed at demonstrating the potential of RAG in conversational AI.

Keywords
NLP, LLM, RAG

Advisors: Aleš Žagar

Introduction

The development of large language models (LLMs) has significantly advanced the capabilities of conversational agents. However, traditional LLM-based chatbots rely solely on pre-trained knowledge, which limits their ability to provide accurate and up-to-date information, especially in dynamic or domain-specific contexts. Retrieval-Augmented Generation (RAG) offers a promising solution by combining real-time information retrieval with generative language models, allowing agents to ground their responses in relevant external data.

In this project, we're building a conversational agent that uses Retrieval-Augmented Generation (RAG) to pull in information from the web in real time. The focus is on answering general knowledge questions—anything from historical facts to current events—with responses that are accurate, coherent, and relevant. Instead of relying only on what it learned during training, the agent will actively search for up-to-date information and use it to generate better answers.

Related Work

Retrieval-Augmented Generation has emerged as a promising approach to enhance the performance of large language models by addressing key limitations such as factual hallucinations, outdated knowledge, and the lack of domain-specific expertise. While LLMs like ChatGPT (OpenAI, 2022) have demonstrated impressive general capabilities (Bang et al. [1]), they are still prone to producing incorrect or outdated information (Cao et al. [2]). These shortcomings are particularly problematic in scenarios requiring factual accuracy and current information.

To address these issues, RAG integrates external information sources—often retrieved via search engines or document databases—into the response generation process. By grounding responses in real-time data, RAG significantly improves the factual consistency and relevance of generated outputs (Guu et al. [3]). This method is particularly effective when paired with robust retrieval systems, which can provide up-to-date context from the vast and constantly evolving content available online.

The field of RAG is evolving rapidly. Gao et al. [4] present a comprehensive survey that categorizes over 100 RAG-related studies into three core research paradigms and outlines the technical challenges across the stages of retrieval, generation, and augmentation. Their work offers a detailed roadmap of the current landscape and future directions of RAG in the context of LLMs.

Complementing this, Chen et al. [5] propose the Retrieval-Augmented Generation Benchmark (RGB), the first benchmark designed to systematically evaluate four core capabilities of RAG-enabled LLMs in both English and Chinese. Their analysis highlights several current limitations of LLMs in RAG settings, including difficulties in accurate retrieval, context integration, and factually consistent generation. The study also suggests concrete directions for future improvement, making it a key contribution to the evaluation of RAG systems.

Initial Idea

Our plan is to create a conversational agent using Naive RAG implementation [4] for a general knowledge LLM with Wikipedia and/or some news outlet as the primary external information source. Since most of the pretrained models like LLama and DeepSeek have been trained on Wikipedia already, the best way to implement RAG for it would be to try to supplement the queries with data after the cut-off date, before which the model was trained.

Model Foundation

Our work will utilize a Large Language Model (LLM) as the foundation for the RAG system. We have two potential approaches:

- Leverage an existing open-source LLM such as LLAMA or similar models, which would reduce computational requirements but may limit customization
- Train a smaller custom model with faculty's high performance compute.

Given resource constraints, we anticipate using a pre-trained model like LLAMA that can be efficiently fine-tuned or adapted to our specific use case without extensive retraining.

Information Retrieval Component

The system will incorporate a web scraping module focused primarily on Wikipedia as an information source due to its breadth of knowledge and structural consistency. This will involve:

- Developing a robust web scraper capable of efficiently retrieving Wikipedia articles
- Implementing rate limiting and caching strategies to respect website policies and reduce redundant requests
- Creating parsers to extract relevant content while preserving semantic structure

Query Processing System

A critical challenge will be developing an effective query processing system that can:

- Extract key information needs from potentially ambiguous user prompts
- Transform these needs into specific search queries optimized for Wikipedia's search system
- Handle disambiguation when multiple possible interpretations exist
- Dynamically refine queries based on initial search results
- Provide context from retrieved documents in a LLM frendly way, so not including too much information and ranking it based on relevance.

Evaluation Framework

To measure the effectiveness of our RAG implementation, we will establish a benchmark comparing:

- Performance of the base LLM without retrieval augmentation
- The same model enhanced with our RAG system
- Metrics will include factual accuracy, hallucination reduction rate, response relevance, and response completeness
- Use external benchmarking and ideas from [5]

This comparative analysis will provide quantitative evidence of how external knowledge retrieval impacts model performance across different query types and knowledge domains.

Datasets, data and corpus

To support our project, we will use a variety of datasets and corpora. Our primary corpus will be Wikipedia Dump, which provides a broad spectrum of general knowledge. In addition, we will utilize Wikidata to complement Wikipedia dump, offering structured knowledge and detailed entity relationships for enhanced context. To ensure our data remains up-to-date, we will continuously scrape Wikipedia pages for the latest updates. For evaluation purposes, we plan to use datasets like Natural Questions[1] to benchmark retrieval and generation performance, along with TriviaQA[2] to assess factual accuracy.

Initial Implementation

Description

Firstly, we implemented a web scraper to collect all relevant Wikipedia page links from the main Video games category. The purpose of this initial step was to generate a comprehensive list of pages that could later be used for further content scraping and analysis. The scraper works by recursively visiting category pages, following pagination to gather all pages listed under each category, and then navigating into subcategories to repeat the process. It filters for valid Wikipedia page links, avoids revisiting categories, and stores progress to disk to prevent data loss during long runs or interruptions. Using this approach, we successfully compiled a dataset of approximately 50,000 unique Wikipedia page URLs. However,

[1] ai.google.com/research/NaturalQuestions
[2] nlp.cs.washington.edu/triviaqa

for initial testing and development, we worked with a smaller subset of these links to ensure correctness and performance before scaling up.

After collecting the list of Wikipedia URLs, we developed a second script to extract and vectorize the textual content of each page. This scraper loads the URLs from a text file and visits each one individually, retrieving and cleaning the main body text using BeautifulSoup [6]. It removes citation markers and other noise to produce a clean description for each Wikipedia page. The cleaned text is then encoded into a dense vector representation using the ‘all-MiniLM-L6-v2‘ model from the SentenceTransformers [7] library, enabling semantic search or clustering in later stages. These vectors are added to a FAISS [8] ndex for efficient similarity search, and a mapping file is maintained to link each vector back to the original Wikipedia page’s title, URL, and text.

To make the collected and vectorized page data searchable, we implemented a semantic search system using FAISS. This script loads the previously saved FAISS index and JSON mapping file, along with the same SentenceTransformer model used during embedding. When a user inputs a natural language query, the model encodes it into a vector, which is then compared to all stored page vectors in the FAISS index to find the most semantically similar matches. The top results are retrieved along with their titles, URLs, and a brief text snippet, along with the computed similarity distance. This allows us to perform efficient and meaningful searches over the Wikipedia page dataset based on the content rather than just keywords.

To complete the system with a fully functional Retrieval-Augmented Generation (RAG) pipeline, we implemented a class-based module that combines document indexing with generative question answering. This component loads a quantized DeepSeek [9] large language model (’deepseek-llm-7b-chat’) for text generation and a SentenceTransformer model (’all-mpnet-base-v2’) for embedding document chunks. It first retrieves the most relevant text passages from the FAISS index based on the query, then tokenizes and segments those passages into manageable chunks, encodes them into embeddings, and uses them to generate partial answers using LLM. These partial answers are then synthesized into a final, coherent response via a second generative prompt. This design allows the model to ground its answers in specific retrieved contexts, making it more accurate and explainable than pure generative approaches.

Evaluation and Analysis

To test our initial implementation, we created a controlled evaluation setup by selecting a subset of 72 Wikipedia links corresponding to upcoming video games. This choice was intentional, as these titles are generally less likely to be part of a language model’s pretraining data, meaning that any correct answers would rely on retrieving and understanding the context rather than relying on prior knowledge. We then designed a set of 10 targeted questions that required specific details from the pages—such as settings, release platforms, or involved actors. This made for a robust and fair test of our retrieval and answer generation pipeline, as it assessed whether the model could accurately retrieve relevant passages and generate context-grounded responses.

Our initial RAG implementation showed encouraging results, answering 6 out of 10 questions completely correctly. Two questions were answered entirely incorrectly due to failures in the retrieval stage—specifically, the wrong sections of the Wikipedia pages were retrieved, likely because of ambiguous wording or insufficient context overlap. The remaining two answers were partially incorrect: while the correct text was successfully retrieved, the language model generated responses that included hallucinated or outdated information, likely influenced by prior knowledge rather than staying grounded in the provided context.

Future Directions and Ideas

While our initial results are promising, there are several areas in need of improvement. One major bottleneck is the embedding process, which is currently too slow to scale efficiently—especially given the size of our full dataset (50,000 links). To address this, we plan to implement parallelized embedding generation and experiment with more advanced embedding models that may offer better retrieval accuracy. Additionally, our current retrieval method relies solely on vector similarity via FAISS. While effective, it can miss relevant content when embeddings are noisy or insufficiently distinct. To improve this, we aim to integrate hybrid retrieval techniques, such as combining keyword-based search with vector similarity, reranking retrieved candidates, and filtering out results when the distance score is too high (indicating poor match).

We also see room for optimization in the RAG pipeline itself. One concrete improvement would be introducing an embedding cache: when a user query is received, we would first check if the embedding already exists in the cache. If it does, we reuse it for the FAISS search; if not, we compute it and update the cache. This would reduce redundant computation and speed up response times in repeated usage scenarios.

That said, the biggest challenge we encountered so far has been prompt engineering. Although the idea behind our RAG model is solid, the actual generation often fails to deliver concise and focused outputs. Despite having the correct context, the LLM sometimes produces vague, overly verbose, or entirely off-topic answers. Much of our recent work has been centered around iteratively refining prompts to force the model into giving short, grounded responses—but this remains an ongoing struggle, and the most frustrating part of the pipeline.

Going forward, our next step is to implement the above improvements—particularly around speed, retrieval precision, and prompt reliability—and then scale the system to run on the full dataset of approximately 50,000 video game Wikipedia pages.

References

[1] Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity, 2023.

[2] Meng Cao, Yue Dong, Jiapeng Wu, and Jackie Chi Kit Cheung. Factual error correction for abstractive summarization models, 2021.

[3] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training, 2020.

[4] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. Retrieval-augmented generation for large language models: A survey, 2024.

[5] Jiawei Chen, Hongyu Lin, Xianpei Han, and Le Sun. Benchmarking large language models in retrieval-augmented generation, 2023.

[6] Leonard Richardson. Beautiful soup documentation. *April*, 2007.

[7] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[8] Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library, 2025.

[9] DeepSeek-AI, :, Xiao Bi, Deli Chen, Guanting Chen, Shanhuang Chen, Damai Dai, Chengqi Deng, Honghui Ding, Kai Dong, Qiushi Du, Zhe Fu, Huazuo Gao, Kaige Gao, Wenjun Gao, Ruiqi Ge, Kang Guan, Daya Guo, Jianzhong Guo, Guangbo Hao, Zhewen Hao, Ying He, Wenjie Hu, Panpan Huang, Erhang Li, Guowei Li, Jiashi Li, Yao Li, Y. K. Li, Wenfeng Liang, Fangyun Lin, A. X. Liu, Bo Liu, Wen Liu, Xiaodong Liu, Xin Liu, Yiyuan Liu, Haoyu Lu, Shanghao Lu, Fuli Luo, Shirong Ma, Xiaotao Nie, Tian Pei, Yishi Piao, Junjie Qiu, Hui Qu, Tongzheng Ren, Zehui Ren, Chong Ruan, Zhangli Sha, Zhihong Shao, Junxiao Song, Xuecheng Su, Jingxiang Sun, Yaofeng Sun, Minghui Tang, Bingxuan Wang, Peiyi Wang, Shiyu Wang, Yaohui Wang, Yongji Wang, Tong Wu, Y. Wu, Xin Xie, Zhenda Xie, Ziwei Xie, Yiliang Xiong, Hanwei Xu, R. X. Xu, Yanhong Xu, Dejian Yang, Yuxiang You, Shuiping Yu, Xingkai Yu, B. Zhang, Haowei Zhang, Lecong Zhang, Liyue Zhang, Mingchuan Zhang, Minghua Zhang, Wentao Zhang, Yichao Zhang, Chenggang Zhao, Yao Zhao, Shangyan Zhou, Shunfeng Zhou, Qihao Zhu, and Yuheng Zou. Deepseek llm: Scaling open-source language models with longtermism, 2024.