**FRI Natural language processing course 2025**
**Project report**

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

# Conversational Agent with Retrieval-Augmented Generation

Matej Belšak, Gorazd Gorup, Luka Bajić

**Abstract**

Develop a conversational agent that enhances the quality and accuracy of its responses by dynamically retrieving and integrating relevant external documents from the web. Unlike traditional chatbots that rely solely on pre-trained knowledge, this system will perform real-time information retrieval, ensuring up-to-date answers. Potential applications include customer support, academic research assistance and general knowledge queries. The project will involve natural language processing, web scraping, and retrieval-augmented generation techniques to optimize answer quality.

**Keywords**

Conversational agent, Retrieval-Augmented Generation

*Advisors: Aleš Žagar*

## Introduction

While Large Language Models (LLMs) have evolved considerably and now produce convincing replies, they have inherent limitations. They rely on training data consisting of documents from the past and may not possess knowledge of current events and developments. Due to differences and properties of training datasets, they may not contain specific domain knowledge, failing to answer certain prompts or outright hallucinating. The latter could prove especially disastrous in dedicated chatbots, for example for legal guidance or health care [1].

One possibility would be to retrain the chat model in intervals to try to keep it up to date, but that would still produce periods where the information is missing from the LLM or is outdated. This approach, while completelly time and energy inefficient, would also not guard against hallucinations. To solve these issues, Retrieval Augmented Generation (RAG) is used to provide the missing knowledge to the LLM. RAG employs different techniques to retrieve information from external sources based on user's prompt and through prompt augmentation feed the LLM sufficient information to provide informative and factually correct answer. In the survey by Gao *et al.*, [2], multiple approaches to RAG are presented, highlighting three architectures: naive RAG, which analyzes the user's prompt, retrieves the required information and appends it, letting the LLM do the rest; advanced RAG, which employs pre-retrieval and post-retrieval modifications to the prompt to make it more suitable for information retrieval and subsequent interpretation by LLM; lastly, modular RAG combines multiple approaches, using iterative prompt enhancement, ranking, fusion, etc.

To better evaluate RAG performance, Lyu *et al.*, [3] describe the CRUD framework, employing metrics such as ROUGE, BLEU, precision and recall. Various operations on text (creative generation from context, usage of information to answer questions, identification and correction of false information, summarization, ...) are measured separately to give a more detailed overview of the model.

For document summarization, LLMs, statistical models, graph-based models and other approaches are used to extract the most important information from text. Zhang *et al.*, [4] present multiple solutions, noting that LLMs, while consuming more resources, tend to be more coherent and precise in their summarization if trained correctly.

In this paper, we focus on RAG methods and their use in chatbots. To that end, we design a conversational agent operating on knowledge about different art and media. Specifically, the agent is to suggest and converse about films and other related media based on the user's prompts and preferences. Our contributions are:

- Implementation of a conversational agent using two different pretrained models: DeepSeek-R1 [5] and Qwen 3 [6].
- Implementation of two RAG techniques, a primitive and advanced one, with capabilities of retrieving data from various film-related databases and web sources.

- Evaluation of conversational agents with respect to the model used and the RAG technique. Agents are evaluated against the baseline RAG-less chatbots. Human and LLM judges are used to score responses, and different metrics are analyzed.

## Methods

We designed two RAG pipelines, as presented in fig. 1. The naive RAG pipeline used Natural Language Processing (NLP) methods to extract the important feature information from prompt via lemmatization, stop-word removal, and entity detection. Prompt information was used to determine movie titles and names of people to query for in our selected databases. That information was then appended to the prompt and fed into an LLM.

A more advanced version of our RAG system utilized various components, but we ultimately chose to employ function-calling functionality that is present in some LLMs. Based on the prompt, the LLM decides on the most appropriate information retrieval function to call and informs the RAG system, which then retrieves that information. The information is then inserted into the prompt, which is fed into the LLM.

### Models

To better analyze the impact and viability of RAG on different LLMs, we decided to use two models: one with reasoning capabilities and one without.

Since we were limited by hardware capabilities during our research, we focused on models with quantizied or distilled variants. We ultimatelly decided on a Qwen3 [1] model as our non-reasoning LLM, and a distilled Llama variant of DeepSeek R1 [2] as our reasoning LLM. Both models were 8-billion-parameters versions. We chose DeepSeek as it is a novel set of models with positive benchmarking results [5].

Both models were loaded from HuggingFace and were run through the `transformers` API.

#### 0.0.1 Function Calling

In some cases, we leveraged the function calling capabilities of our models. While DeepSeek hints at possibility of using this feature [7], their proposed prompt templates do not support it. We instead turned to Qwen's function calling capabilities, which proved to be sufficient.

In function calling, the model is presented with the user prompt and structured information about programmatic functions it may call to fullfil the user's request. The LLM is tought to return a similarly structured set of instructions on which functions to call, and with what parameters.

### Information Retrieval

To provide a sufficient level of information related to movies and similar media, we selected the following sources:

- **The Movie Database (TMDB).** This is an open database of movies, TV-shows, and people associated with them. They provide a free-to-use API for various types of requests – release dates, cast, crew, similar films ...
- **Letterboxd.** This is a movie-themed social network to log watched films, provide reviews, and engage in movie-related discussions. We chose to include it to give the model a better understanding of how people percieve certain media.
- **JustWatch.** This website tracks the availability of media on digital and streaming platforms.
- **Wikipedia.** We used it to retrieve information such as film plots, summaries, etc.

In our naive RAG implementation, we retrieved data from all four sources. This would amount to over 250,000 characters of context.

For our advanced version, we prepared nine custom scraping functions that would equip the model with essential information about movie titles and film-related people. These functions had to be specifically annotated and various versions of descriptions were used to elicit a proper function-calling response from the model.

### RAG

For the naive RAG system, we first analyzed the user prompt and gathered all relevant entities, consisting of film titles and names of people. We used a Roberta-like `spacy` model [3]. We performed unfiltered knowledge injection, gathering all relevant documents based on prompt entities, structured them in JSON format, and passed them allong with user query to the LLM.

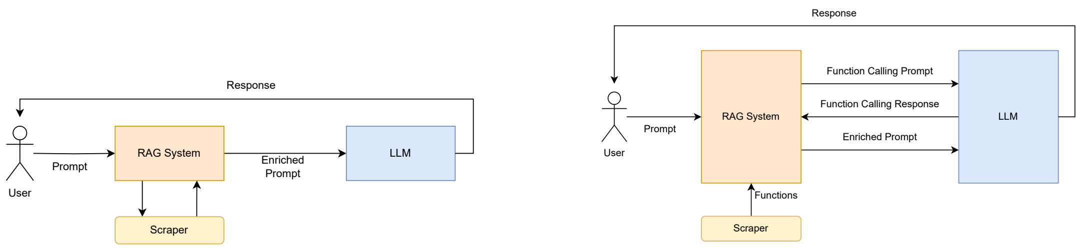
For advanced RAG system, we tried several versions before basing it on function calling. Our earlier attempts performed the entity extraction from prompt, retrieval of documents from all sources based on those entities, and content curation based on statistical methods and user prompt. For example, we treated all sentences in the retrieved data as documents, performing TF-IDF sorting and using only top $N$ most relevant sentences. Unfortunatelly, results of these extractions accounted only for literal appearance of words from the prompt in documents, instead of accounting for larger context, synonyms, etc.

To address these shortcomings, we turned to function calling, letting the LLM decide what information to retrieve. In the system prompt, we also appended additional information about current date for better context formulation.

### Prompt formulation

We structured our prompts to consist of three roles: system, user, and assistant. We put system prompt first, followed by the user prompt, and finally by assistant tag to signal the LLM to generate the answer, not to continue the user prompt.

[1] https://huggingface.co/Qwen/Qwen3-8B
[2] https://huggingface.co/deepseek-ai/DeepSeek-R1
[3] https://huggingface.co/spacy/en_core_web_sm




**(a)** A naive RAG system. **(b)** An advanced RAG system.

**Figure 1.** Our proposed RAG pipelines for evaluation. The naive RAG variant preprocesses the prompt, extracts meaningful information from it, retrieves the data based on that, and enhances the prompt with that raw data. The advanced variant employs LLM function-calling capabilities to decide what data to scrape and enhances the prompt only with necessary data.

Our system prompt was:

> You are an AI chatbot, assisting user with anything related to movies. [Today's date is `{date}`.] You may only use information provided to you inside the `<data>` tags.

**Figure 2.** Out prompt for RAG usage. The text in square brackets was only used in advanced RAG, and the text in braces represents variables.

Our user prompt first presented the user's initial query, followed by the retrieved information enclosed in `<data>` tags.

### Memory

Because we designed our agent as a chatbot, we also implemented a simple answer caching mechanism to provide the chatbot with immediate history of the conversation. We stored last $n$ user questions and chatbot replies with the First-In-First-Out (FIFO) method, and fed them to the LLM as part of the prompt. The $n$ in our observations had to be relatively small due to limitations on the amount of input tokens.

## Results

In this section we compare the performance of a reasoning DeepSeek-R1-Distill-Llama-8B [5] model and a non-reasoning Qwen3-8B [6] model. For each of these we run experiments on three different sets of parameters: without RAG, with naive RAG - data is scraped and appended to the context, and advanced RAG - data is additionally processed and summarized before being inputed into the model. In total, we are evaluating the following models:

- DeepSeek-R1-Distill-Llama-8B (baseline)
- DeepSeek-R1-Distill-Llama-8B with naive RAG
- DeepSeek-R1-Distill-Llama-8B with advanced RAG
- Qwen3-8B (baseline)
- Qwen3-8B with naive RAG
- Qwen3-8B with advanced RAG

Since we are working with relatively open-ended questions, there is a lack of objective ground truth to evaluate against. Therefore, we construct a set of 50 domain-relevant questions and use manually checked answers from the commercial ChatGPT model as ground truth, so we can apply standard metrics such as ROUGE and BLEU [4]. To avoid overfitting a model to a specific set of questions, we included multiple types of questions into our test set, including yes/no questions, fact checking, list retrieval and summarization. Results are shown in Table 1.

| Model | ROUGE | BLEU |
|---|---|---|
| deepseek-baseline | 0.171003 | 0.049453 |
| deepseek-naive | 0.165164 | 0.043081 |
| deepseek-advanced | 0.16714 | 0.01866 |
| qwen-baseline | 0.192379 | 0.031395 |
| qwen-naive | 0.194950 | 0.034025 |
| qwen-advanced | 0.17380 | 0.02584 |

**Table 1.** Performance comparison against ChatGPT-generated ground truth.

### LLM-based evaluation

Metrics such as BLEU and ROUGE are ill-adjusted to our test set, therefore we employ the DeepEval framework [8] in order to obtain more relevant metrics based on context, query and response. Results are shown in Table 2. Metrics related to context are not available for baseline models, because they do not receive any context.

### Conversational chatbot

Extending a working model into a conversational chatbot is straightforward: the data retrieval pipeline remains the same, we only need to implement a mechanism for memorization of

[4] https://huggingface.co/docs/evaluate/index

| model | correctness | clarity | answer relevancy | faithfulness | contextual precision | contextual recall | contextual relevancy |
|---|---|---|---|---|---|---|---|
| deepseek-baseline | 0.19 | 0.7725 | 0.76846 | **0.98615** | / | / | / |
| deepseek-naive | 0.152 | 0.82800 | 0.78138 | 0.93233 | 0.24000 | 0.61771 | **0.40468** |
| deepseek-advanced | | | | | | | |
| qwen-baseline | **0.30962** | **0.86888** | 0.74805 | 0.94456 | / | / | / |
| qwen-naive | 0.23111 | 0.65888 | **0.84670** | 0.88345 | **0.88345** | **0.76333** | 0.37029 |
| qwen-advanced | | | | | | | |

**Table 2.** Performance comparison as evaluated by a 14B parameter Qwen model with GEval framework.

| model | correctness | clarity | answer relevancy | faithfulness | contextual precision | contextual recall | contextual relevancy |
|---|---|---|---|---|---|---|---|
| deepseek-baseline | 0.31845 | 0.74923 | 0.61720 | / | / | / | / |
| deepseek-naive | 0.33410 | 0.81443 | 0.720714 | 0.75611 | 0.34 | 0.32014 | 0.43623 |
| deepseek-advanced | 0.33412 | 0.78414 | 0.80973 | 0.93095 | **1.0** | 0.60937 | 0.56813 |
| qwen-baseline | 0.45205 | 0.81023 | **0.87944** | / | / | / | / |
| qwen-naive | 0.45116 | 0.77992 | 0.80013 | 0.83234 | 0.34 | 0.264401 | 0.51643 |
| qwen-advanced | **0.52336** | **0.83382** | 0.68042 | **0.97935** | **1.0** | **0.61096** | **0.68355** |

**Table 3.** Performance comparison as evaluated by GPT-mini-4.1 model with GEval framework.

previous questions and responses. Examples of Qwen chatbot results are shown in Appendix 3.

## Discussion

We conclude that it is possible to improve the responses of 8B parameter large language models on queries from a specific domain by employing on-the-fly information retrieval via web scraping.

## References


[1] Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. Large language models in medicine. *Nature medicine*, 29(8):1930–1940, 2023.

[2] 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.

[3] Yuanjie Lyu, Zhiyu Li, Simin Niu, Feiyu Xiong, Bo Tang, Wenjin Wang, Hao Wu, Huanyong Liu, Tong Xu, and Enhong Chen. CRUD-RAG: A comprehensive Chinese benchmark for retrieval-augmented generation of large language models, 2024.

[4] Yang Zhang, Hanlei Jin, Dan Meng, Jun Wang, and Jinghua Tan. A comprehensive survey on process-oriented automatic text summarization with exploration of LLM-based methods, 2025.

[5] DeepSeek-AI et al. DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning, 2025.

[6] Qwen Team. Qwen3, April 2025.

[7] DeepSeek. Function Calling — DeepSeek API Docs — api-docs.deepseek.com. https://api-docs.deepseek.com/guides/function_calling/. [Accessed 29-05-2025].

[8] Jeffrey Ip and Kritin Vongthongsri. deepeval, May 2025.


User: How many films did Chuck Jones make?
Qwen: Based on the data provided:
**Total Number of Films:** Chuck Jones directed **over 100 films**, spanning multiple genres and formats.
### Breakdown: - **Feature-Length Films**: - *The Bugs Bunny/Road Runner Movie* (1979) - *The Phantom Tollbooth* (1970) - *The Grinch Stole Christmas!* (1966)
- **Shorts and Documentaries**: - Approximately **100+ short cartoons** - Many of these are Looney Tunes/Merry Melodies cartoons. - Includes iconic titles such as *What's Opera, Doc?*, *Duck Amuck*, etc.
- **Documentaries**: Around **dozens**, e.g., - *The Magical World of Chuck Jones* - *Chuck Jones: Memories of Childhood*
### Summary: Chuck Jones made approximately **more than 100 films**, primarily consisting of **animated short cartoons**, a handful of **feature-length compilations**, and several **documentaries**.

**Figure 3.** Conversation with Qwen

## Appendix

### Qwen conversation example