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

Conversational Agent with Retrieval-Augmented Generation

Blaž Špacapan, Matevž Jecl, Tilen Ožbot

Abstract

This project, developed for the NLP course, presents a Python-based chatbot that maintains up-to-date sports news knowledge using a real-time Retrieval-Augmented Generation (RAG) pipeline. We implement automated web-scraping of rtvslo.si to fetch full articles and metadata, perform HTML parsing, text normalization and duplicate removal. Upon a user query, the system extracts keywords via a spaCy+LLM hybrid approach, scrapes and indexes new articles on the fly, retrieves and re-ranks the top semantically similar chunks, and synthesizes a coherent answer using a large language model—all within sub-second latency. We evaluate two RAG configurations, GAMS and Gemini, on our sports QA test suite, obtaining up to 57.22 BLEU and 0.753 ROUGE-L for the best model, and complement these with human judgments of coherence, informativeness, and factual consistency. A pilot study with classmates confirms the chatbot's ability to deliver accurate, contextually relevant responses in dynamic sports scenarios, demonstrating that a lightweight real-time RAG system can be built effectively at the undergraduate level and extended to other news domains.

Keywords
RAG, web-scraping, sports news

Advisors: Aleš Žagar

Introduction

Various fields such as customer service, education, and even general knowledge have shifted greatly with the introduction of sophisticated language models as conversational agents. Even with all the advancements made in the industry, conventional chatbots depend purely on pre-trained static knowledge, which can quickly become outdated, making it impossible to retrieve accurate information. The goal of this project is to address that gap by creating a chatbot that automatically improves its accuracy and responsiveness by web-scraping related news articles from the web, in real-time. The proposed conversational agent utilizes Retrieval-Augmented Generation (RAG) techniques to dynamically access current information, ensuring the responses provided are accurate and up to date.

Master's thesis from Chalmers University of Technology exploring enhancements to conversational agents with Retrieval-Augmented Generation by integrating an autoregressive Large Language Model that generates real-time dense retrieval vectors, significantly improving multi-hop reasoning through synthetic data generation and attention-based relabeling to reduce hallucinations [1].

Research from ESP Journal of Engineering & Technology Advancements introduces a conversational Retrieval-Augmented Generation framework designed for real-time crisis management, integrating structured and unstructured data sources to deliver validated, contextually relevant insights that enhance emergency decision-making and response effectiveness [2].

In this article, the authors propose Retrieval-Augmented Generation (RAG) models that integrate parametric seq2seq generation with non-parametric retrieval mechanisms, demonstrating improved performance in conversational agents through enhanced factual accuracy and response specificity in knowledge-intensive interactions [3].

The paper proposes a misinformation detection framework, FCRV (Full-Context Retrieval and Verification), that integrates claim extraction via Large Language Models (LLMs) with Retrieval-Augmented Generation (RAG) to construct a comprehensive context for news verification, significantly improving detection accuracy, robustness, and scalability against both human and AI-generated fake news [4].

The paper demonstrates how Retrieval-Augmented Generation (RAG) with Large Language Models (LLMs) effectively addresses challenges in extracting precise business event information from diverse and evolving data sources, significantly enhancing adaptability and accuracy in dynamic business environments [5].

Methods

Data

News content will be collected exclusively by web-scraping major outlets. Each script retrieves full article text and metadata (title, author, publication date, URL) and stores the raw content for processing. A preprocessing pipeline then performs HTML parsing, text normalization, duplicate removal, and formatting. Cleaned articles are converted into vector representations and indexed in a vector database to enable efficient semantic retrieval. This approach produces a continuously updated, semantically organized corpus of both current and historical news articles, supporting rapid access to relevant documents for the conversational agent.

Database

The system relies on a PostgreSQL database enhanced with the `pgvector` extension. While earlier versions used this database both to compute and store vector embeddings for each text fragment, embedding generation has since been moved upstream. Today, the database's sole responsibility is to retrieve content of already fetched articles. The database schema consists of a single table, `content_chunks`, which holds:

- `id` (integer): a primary key for each chunk, generated by a sequence.
- `chunk_text` (text): the raw text of the content fragment.
- `embedding` (vector(384)): the precomputed 384-dimensional embedding for similarity search.
- `source_identifier` (text): a unique identifier such as a URL or filename for the original source.
- `created_at` (`timestamptz`): a timestamp set at insertion.
- `metadata` (JSONB): arbitrary metadata (e.g., page numbers, tags) associated with the chunk.

Pipeline

We selected `rtvslo.si` for initial testing because it offers robust filtering by category and date. For this phase, we focus exclusively on sports news. We use the following parameters when querying the `rtvslo.si` API:

- `q`: the search query string
- `s`: section filter (e.g., `3` for sports, `null` for all)
- `sort`: sort order (`1` = newest first, `2` = most popular)
- `a`: time range (`1` = all time, `2` = last 24 h, `3` = last week, `4` = last month, `5` = last year)
- `per_page`: number of results to return per page
- `group`: content type group (`1` = news, `15` = video, `16` = audio)

1. **Receive user query**: The system accepts a natural-language question or request from the user.
2. **Extract keywords from the query**: Initially we use a spaCy-based named entity recognition (NER) method to identify key entities within the question. If no entities are found, the function assesses the ambiguity of the input using an LLM (`gemma-2-2b-it` or `gemma-3-4b-it`). If the question is deemed clear enough, keywords are extracted using an LLM-based approach that captures more nuanced information. However, if the text is found to be overly ambiguous, the function generates an explanatory output with a suggestion for a less ambiguous question and terminates further processing. This layered strategy enables efficient handling of clearly defined questions while providing robust feedback for ambiguous queries. **Example:**

   ¿ Koliko točk je dobil na zadnji tekmi?

   # To vprašanje je slabo zastavljeno, saj ne navaja konteksta (katero tekmo). #
3. **Perform web-scraping to discover relevant articles**: Using the extracted keywords, the scraper queries selected news pages to retrieve potentially relevant articles. Since `robots.txt` is not provided by the site, we enforce a 1s timeout between requests to avoid overwhelming the server. For enough context 50 pages were fetched.
4. **Parse and collect article metadata and content**: Relevant content from newly found articles is processed, converted into vector representations (embeddings), and stored in a database. The system retrieves a predefined number of the most semantically similar information chunks from the entire database (including newly added ones) based on the user's query embedding.
5. **Chunk Reranking**: We re-rank all candidate chunks with the cross-encoder *cross-encoder/ms-marco-MiniLM-L-6-v2*; chunks whose similarity score is ≥ 1.0 are discarded.
6. **LLM-based Answer Synthesis**: Finally, a large language model (*GaMS-9B-Instruct-Q4_k* or *gemini-1.5-flash-latest*) receives the refined, relevant information chunks and the original user query (potentially along with conversation history). It synthesizes this information to generate a comprehensive and contextually appropriate answer.

Results

We evaluate both RAG configurations on our sports QA test suite using automatic metrics (BLEU, ROUGE) and human judgments (coherence, informativeness, factual consistency) across 11 sampled QA pairs.

Automatic Metrics Table 1 shows BLEU and ROUGE scores, computed on answers generated *with the identical retrieval + reranking pipeline* described above

Table 1. Automatic evaluation results

Model	BLEU (%)	ROUGE-1	ROUGE-2	ROUGE-L
GAMS	30.37	0.455	0.307	0.418
Gemini	57.22	0.779	0.652	0.753

Human Evaluation

- **Coherence**: logical flow and structure
- **Informativeness**: amount of relevant information
- **Factual consistency**: alignment with ground truth

Overall, Gemini outperformed GAMS on every dimension. Table 2 summarizes average ratings.

Table 2. Average human evaluation scores (1–5)

Model	Coherence	Informativeness	Factual Consistency
GAMS	4.2	4.0	3.1
Gemini	4.8	4.5	4.7

Error Analysis

- **Wrong answers by GAMS (despite context):**
  - Q10 (*Benjamin Šeško*): predicted "Eintracht Frankfurt" (correct: Stuttgart); factual score 1/5.
  - Q8 (L. Dončić's penultimate game): predicted 28 points (correct: 38); factual score 1/5.
  - Q5 (J. Oblak's last Atlético match): predicted 8 saves (correct: 6); factual score 1/5.
- **Absence detection:**
  - Q3 (winner of final Tour de France stage): both models correctly abstained ("not in context"); factual score 5/5.
- **Consistently correct responses:**
  - Q1 (L. Dončić's last game): 28 points.
  - Q11 (A. Kopitar's last opponent): Edmonton.

Discussion

Our results demonstrate a clear advantage of the Gemini configuration over GAMS. The automatic metrics reveal a BLEU gain of 26.85 points and a ROUGE-L improvement of 0.335, indicating much stronger alignment with reference answers. Human evaluations corroborate these findings: Gemini achieves higher coherence (+0.6), informativeness (+0.5), and factual consistency (+1.6) on average.

GAMS's recurrent factual errors, even when relevant context was retrieved, suggest limitations in its reranking or answer synthesis stages, leading to hallucinations. Although Gemini's improved answer generation strategies substantially reduce misinformation compared to GAMS, it too occasionally generates minor inaccuracies even with the correct context, indicating that enhanced grounding and verification remain necessary.

Overall, these results validate that a lightweight, real-time RAG pipeline can be implemented effectively.

References

[1] Malte Landgren and Oskar Giljegård. Real-time Relevance: RAG with Dynamic Context for Improved Natural Language Responses. Master's thesis, Chalmers University of Technology, 2024.

[2] Abhinav Balasubramanian. Esp-jeta. *ESP Journal of Engineering & Technology Advancements*, 4(4):118–126, 11 2024.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc., 2020.

[4] Yangxiao Bai and Kaiqun Fu. A large language model-based fake news detection framework with rag fact-checking. In *2024 IEEE International Conference on Big Data (BigData)*, pages 8617–8619, 2024.

[5] Muhammad Arslan, Saba Munawar, and Christophe Cruz. Exploring business events using multi-source rag. *Procedia Computer Science*, 246:4534–4540, 2024. 28th International Conference on Knowledge Based and Intelligent information and Engineering Systems (KES 2024).