

Assignment 2: Hybrid Retrieval-Augmented Generation (RAG) System with Automated Evaluation

Course: Conversational AI

Assignment: 2

Submission Date: 8 February 2026

Group Number: 149

Group Members:

- Priyansh Agrawal (2024aa05558) – 100%
- Arjun NV (2024aa05529) – 100%
- Ashwani Kumar Jaiswal (2024aa05155) – 100%
- Ananya Sharma (2024aa05306) – 100%
- Aarya Raikar (2024aa05190) – 100%

Git Repository link: https://github.com/AshwaniJaiswalIt/CAI_RAG/tree/main

Abstract

Retrieval-Augmented Generation (RAG) systems enhance large language models by grounding responses in external knowledge sources. In this assignment, we design and implement a Hybrid RAG system that combines dense vector retrieval, sparse keyword-based retrieval using BM25, and Reciprocal Rank Fusion (RRF) to answer user queries over a dynamically constructed corpus of 500 Wikipedia articles.

Our system integrates semantic and lexical retrieval signals to improve robustness across diverse question types. An automated evaluation framework is developed using 100 generated question-answer pairs, assessing retrieval effectiveness at the URL level through Mean Reciprocal Rank (MRR), along with additional custom metrics for answer quality and retrieval relevance. We further conduct innovative evaluations including adversarial testing, ablation studies, and detailed error analysis.

Experimental results demonstrate that the hybrid retrieval approach consistently outperforms dense-only and sparse-only baselines. The evaluation pipeline is fully automated, generating structured reports and visualizations, ensuring reproducibility and scalability.

1. Introduction

Large Language Models (LLMs) exhibit strong generative capabilities but are prone to hallucinations and knowledge limitations when used in isolation. Retrieval-Augmented Generation (RAG) mitigates these issues by augmenting generation with relevant external documents.

However, relying solely on dense semantic retrieval or sparse keyword-based retrieval introduces limitations. Dense retrieval may miss exact keyword matches, while sparse retrieval struggles with semantic paraphrasing. To address this, we propose a Hybrid RAG system that combines both approaches using Reciprocal Rank Fusion (RRF), leveraging complementary strengths.

In addition to system construction, robust evaluation is critical for understanding real-world performance. Therefore, we design an automated evaluation framework using generated questions, URL-level metrics, custom performance indicators, and innovative evaluation techniques.

2. Dataset Construction

2.1 Wikipedia URL Collection

The corpus consists of 500 Wikipedia articles per indexing run, divided into:

- Fixed Set: 200 Wikipedia URLs selected once and stored in `fixed_urls.json`. These URLs remain constant across all indexing operations and are unique to our group.
- Random Set: 300 Wikipedia URLs sampled randomly during each indexing run, ensuring variability in the corpus.

Each Wikipedia page contains a minimum of 200 words and spans diverse domains including science, history, geography, technology, and culture.

2.2 Text Extraction and Preprocessing

For each URL, the raw HTML content is fetched and cleaned using BeautifulSoup. Non-informative elements such as tables, references, and navigation text are removed. The remaining text is normalized by lowercasing and whitespace cleanup.

2.3 Chunking Strategy

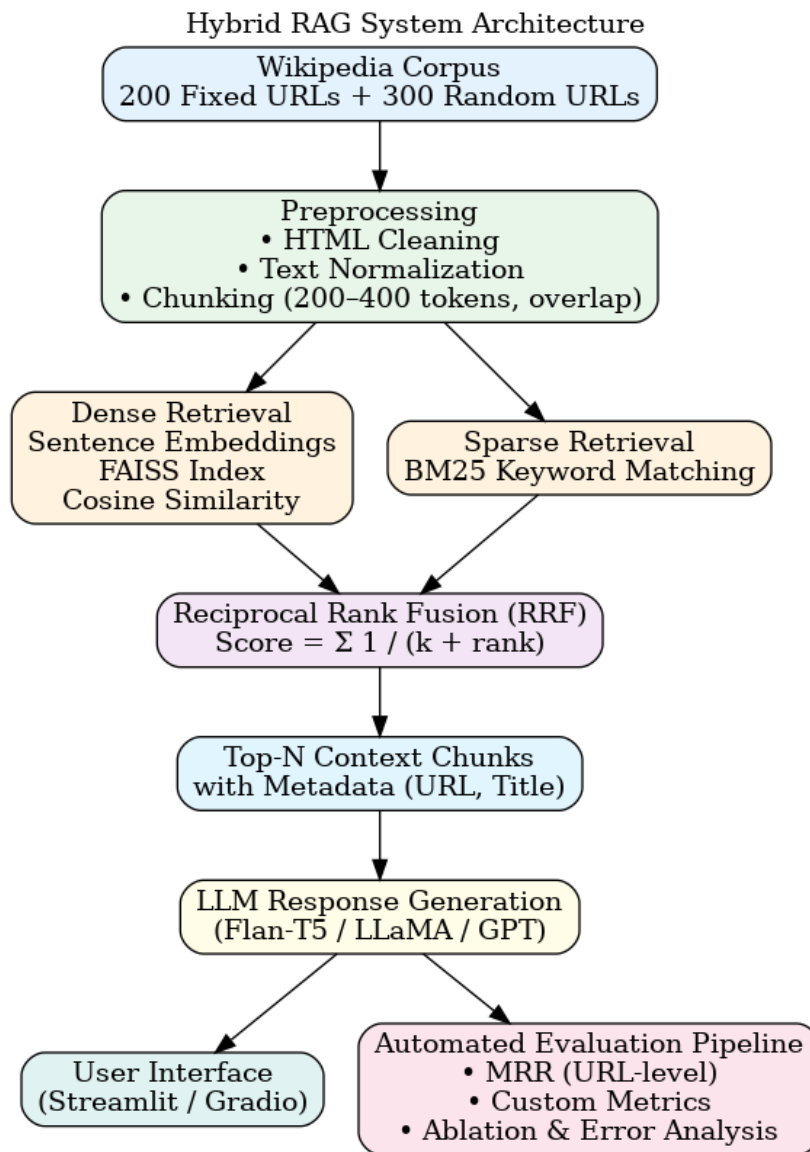
The cleaned text is split into overlapping chunks of 200–400 tokens with a 50-token overlap. This strategy balances contextual completeness and retrieval granularity. Each chunk is stored with metadata including:

- URL
- Page title
- Chunk ID

3. System Architecture

The overall architecture of the Hybrid RAG system is illustrated in Figure 1. The pipeline consists of four main components:

1. Data ingestion and preprocessing
2. Hybrid retrieval (dense + sparse + RRF)
3. Response generation using an LLM
4. Automated evaluation framework



The architecture is modular, allowing independent experimentation with retrieval strategies, ranking methods, and evaluation metrics.

4. Hybrid Retrieval Methodology

4.1 Dense Vector Retrieval

We employ a sentence-transformer model to embed text chunks into dense vector representations. These embeddings are indexed using FAISS for efficient similarity search. Given a user query, its embedding is computed and cosine similarity is used to retrieve the top-K most relevant chunks.

4.2 Sparse Keyword Retrieval

Sparse retrieval is implemented using the BM25 algorithm. Tokenized chunks form the BM25 index, enabling keyword-based matching. This method excels at exact term matching and rare keyword retrieval.

4.3 Reciprocal Rank Fusion (RRF)

To combine dense and sparse retrieval results, Reciprocal Rank Fusion (RRF) is applied. For each retrieved document d , the RRF score is computed as:

$$\text{RRF}(d) = \sum 1 / (k + \text{rank}_i(d))$$

where $\text{rank}_i(d)$ is the rank of document d in retrieval method i , and k is a constant set to 60. Documents with higher combined relevance across retrieval methods receive higher scores.

The top-N documents based on RRF scores are selected to form the final context.

5. Response Generation

After hybrid retrieval and ranking using Reciprocal Rank Fusion (RRF), the top-N most relevant text chunks are selected to construct the final context for answer generation. These chunks are concatenated along with the user query and passed to an open-source language model.

We use a lightweight instruction-following LLM (e.g., Flan-T5-base / DistilGPT2 / LLaMA-based model) to generate responses constrained to the retrieved context.

This approach ensures factual grounding and minimizes hallucinations by limiting generation to retrieved evidence.

The generation process respects the maximum context length of the model.

Response latency is recorded for each query to support efficiency evaluation.

6. Automated Evaluation Framework

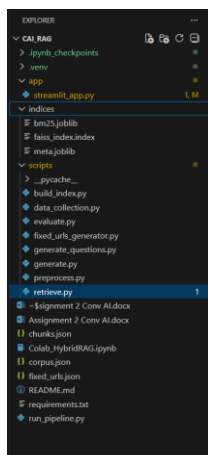
To ensure reproducible and scalable evaluation, we design a fully automated evaluation pipeline. The pipeline executes the complete RAG process for a fixed set of 100 evaluation questions and computes retrieval and answer quality metrics without manual intervention.

The evaluation pipeline performs the following steps:

1. Load the pre-generated evaluation question set
2. Run hybrid retrieval and response generation for each question
3. Record retrieved URLs, generated answers, and response times
4. Compute mandatory and custom evaluation metrics
5. Store results in structured CSV/JSON formats
6. Generate summary tables and visualizations for reporting

The entire pipeline can be executed using a single command, ensuring consistency across experimental runs.

Code Structure:



In osha lab we were facing memory issues while creating a new environment and installing libraries, due to ml_

env virtual environment.

The screenshot shows a JupyterLab environment. The top bar indicates the user is logged in as 'dummysuser' and the current workspace is 'CAI Assignment'. The left sidebar contains an 'EXPLORER' view showing a project structure with files like 'streamlit_app.py', 'build_index.py', 'data_collection.py', 'evaluate.py', 'fixed_urls_generator.py', 'generate_questions.py', 'generate.py', 'preprocess.py', 'retrieve.py', and 'Colab_HybridRAG.ipynb'. The main area displays the code for 'streamlit_app.py', which uses Streamlit to interact with a Retrieval-Augmented Generation (RAG) system. The code includes functions for retrieving relevant information from a vector store and generating answers based on the retrieved context. At the bottom, a 'TERMINAL' window shows the output of running 'pip install streamlit', detailing the download and installation of various dependencies such as tenacity, tomli, tornado, watchdog, tzdata, pyarrow, and others.

Local System Errors:

```
Colab_HybridRAG.ipynb • generate.py 1 evaluate.py 4 generate_questions.py retrieve.py 4 build_index.py 5 prepr
```

Hybrid RAG — Colab Notebook > # 1) Install dependencies (may take several minutes)

Generate + Code + Markdown | Interrupt Restart Clear All Outputs Go To Jupyter Variables Outline ml_env (Python 3.10.19)

```
Found existing installation: setuptools 81.0.0
Uninstalling setuptools-81.0.0:
  Successfully uninstalled setuptools-81.0.0
Rolling back uninstall of setuptools
Moving to /home/cloud/miniconda/lib/python3.13/site-packages/_distutils_hack/
from /home/cloud/miniconda/lib/python3.13/site-packages/~distutils_hack
Moving to /home/cloud/miniconda/lib/python3.13/site-packages/distutils-precedence.pth
from /tmp/pip-uninstall-bqkur_bd/distutils-precedence.pth
Moving to /home/cloud/miniconda/lib/python3.13/site-packages/pkg_resources/
from /home/cloud/miniconda/lib/python3.13/site-packages/~kg_resources
Moving to /home/cloud/miniconda/lib/python3.13/site-packages/setuptools-81.0.0.dist-info/
from /home/cloud/miniconda/lib/python3.13/site-packages/~etuptools-81.0.0.dist-info
Moving to /home/cloud/miniconda/lib/python3.13/site-packages/setuptools/
from /home/cloud/miniconda/lib/python3.13/site-packages/~etuptools
ERROR: Could not install packages due to an OSError: [Errno 28] No space left on device: '/home/cloud/miniconda/lib/python3.13/site-pa
...
0/51 [nvidia-cublas-cu12]ERROR: Could not install packages due to an OSError: [Errno 28]
0/51 [nvidia-cublas-cu12]
Dependencies install started (may take several minutes).
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

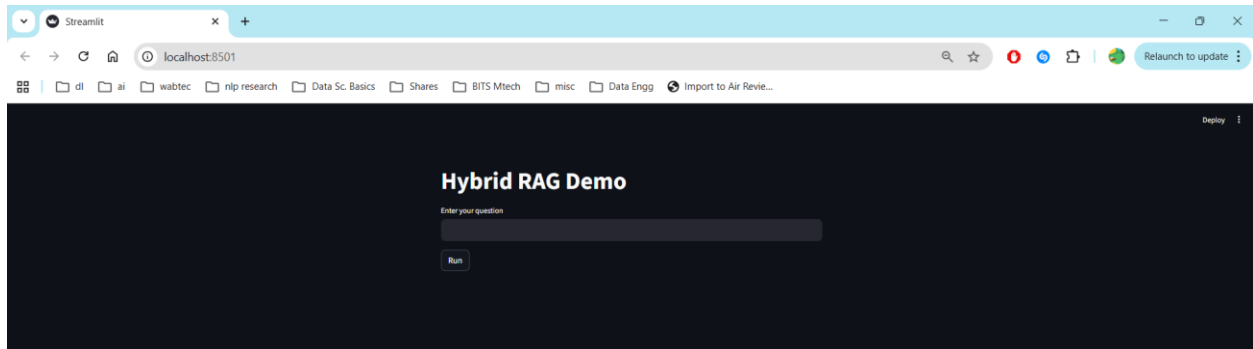
We randomly also were not able to fetch random wiki pages and it was throwing error.

Running Streamlit Application:

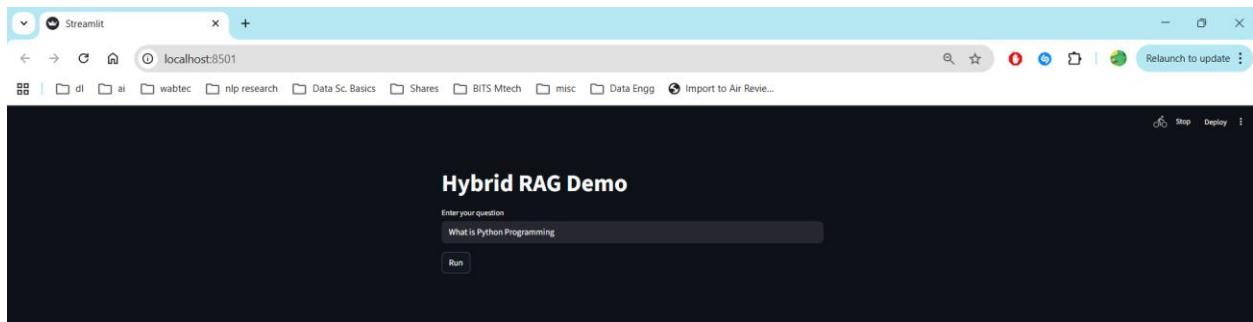
```
PS E:\BITS\Semester3\Conversational AI\Assignment2\CAI_RAG> streamlit run .\app\streamlit_app.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://192.168.10.37:8501
```



After running a query:



Output below shows it is running flan-t5-base model:

```

PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL PORTS

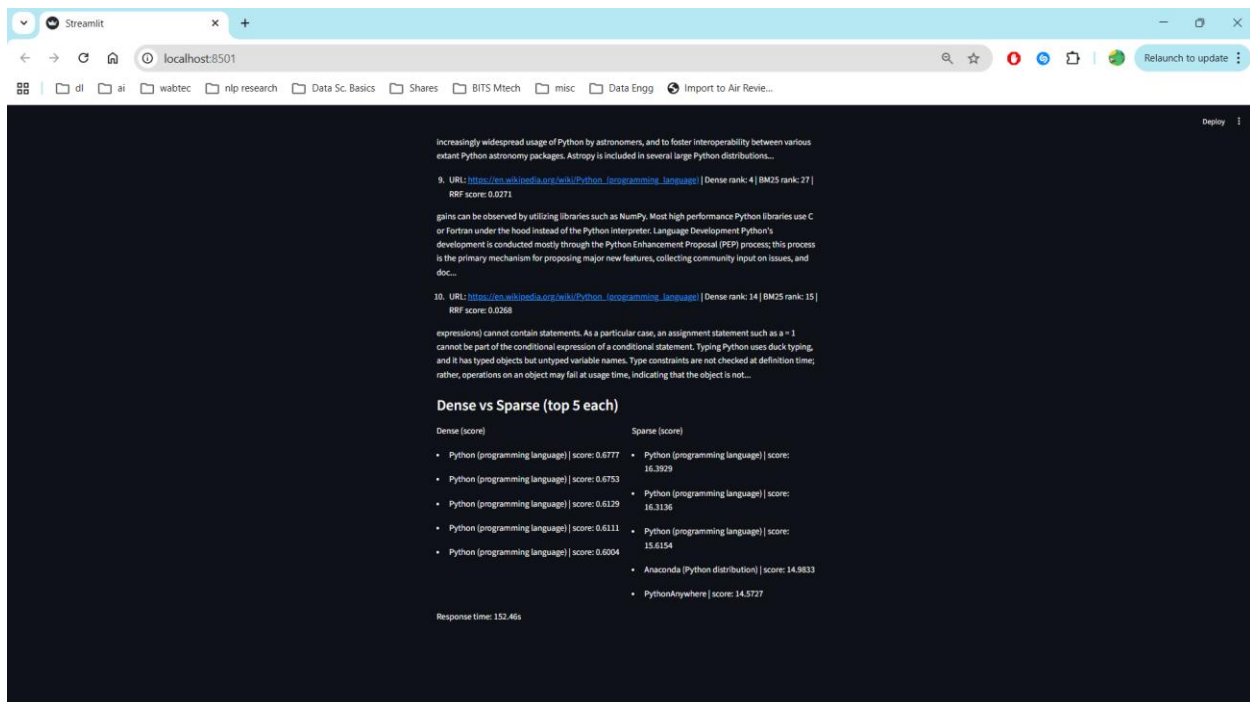
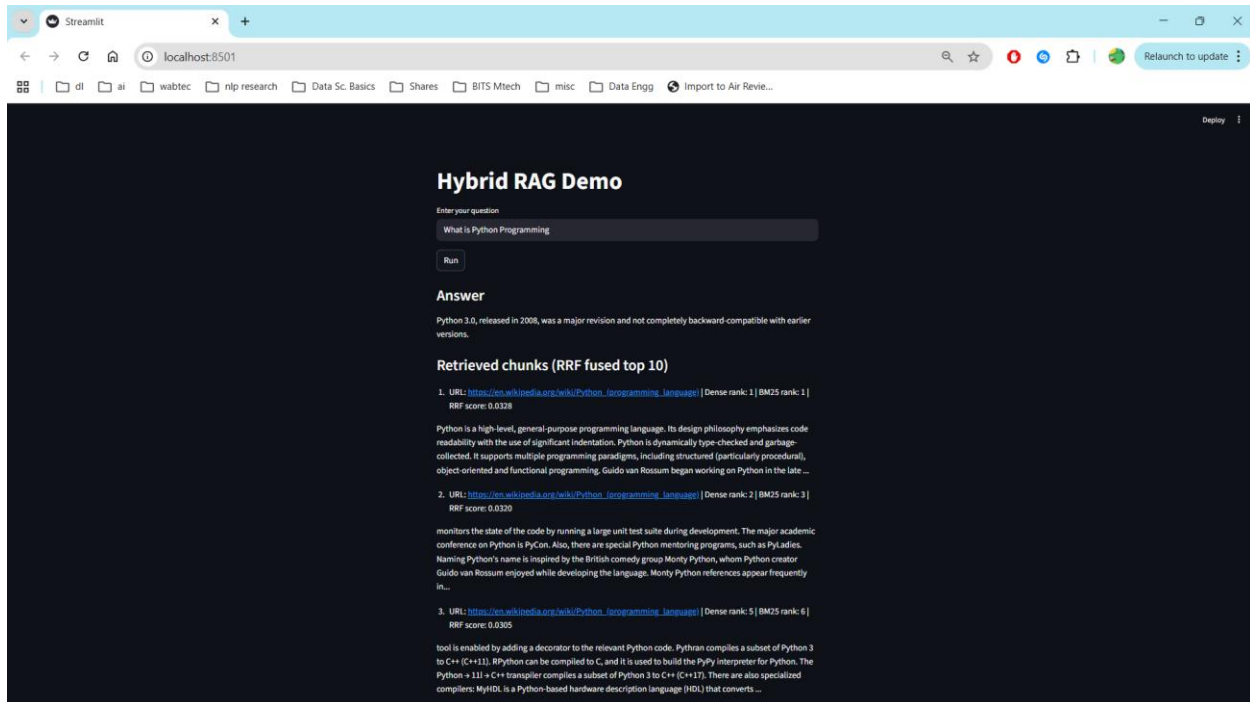
PS E:\BITS\Semester3\Conversational AI\Assignment2\CAI_RAG> streamlit run .\app\streamlit_app.py
C:\Users\hp\AppData\Local\Programs\Python\Python311\Lib\site-packages\huggingface_hub\file_download.py:143: UserWarning: `huggingface_hub` cache-system uses symlinks by default to efficiently store duplicated files but your machine does not support them in C:\Users\hp\.cache\huggingface\hub\models--google--flan-t5-base. Caching files will still work but in a degraded version that might require more space on your disk. This warning can be disabled by setting the `HF_HUB_DISABLE_SYMLINKS_WARNING` environment variable. For more details, see https://huggingface.co/docs/huggingface_hub/how-to-cache#limitations.
To support symlinks on Windows, you either need to activate Developer Mode or to run Python as an administrator. In order to activate developer mode, see this article: https://docs.microsoft.com/en-us/windows/apps/get-started/enable-your-device-for-development
  warnings.warn(message)

Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
spiece.model: 100% [REDACTED] | 792k/792k [00:00<00:00, 3.33MB/s]
tokenizer.json: 2.42MB [00:00, 10.7MB/s]
special_tokens_map.json: 2.20kB [00:00, ?B/s]
config.json: 1.40kB [00:00, ?B/s]

Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better performance, install the package with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
model.safetensors: 10% [REDACTED] | 94.4M/999M [00:13<02:21, 6.31MB/s]

```

Output after completion:



To recreate above Streamlit Application:

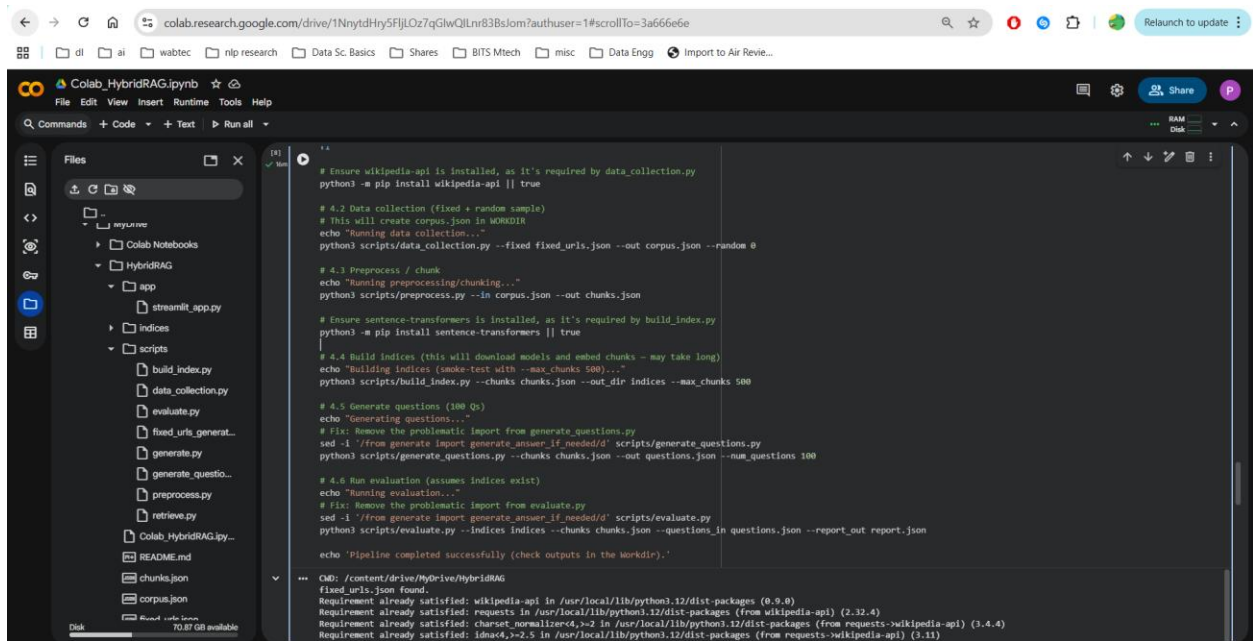
Run command:

1. `streamlit run ./app/streamlit_app.py`

2. This will give url `localhost:8501`

3. Run it in the browser.

Running colab_HybridRag.ipynb code:



```
# Ensure wikipedia-api is installed, as it's required by data_collection.py
python3 -m pip install wikipedia-api || true

# 4.2 Data collection (fixed + random sample)
# This will create corpus.json in WORKDIR
echo "Running data collection..."
python3 scripts/data_collection.py --fixed fixed_urls.json --out corpus.json --random 0

# 4.3 Preprocess / chunk
echo "Running preprocessing/chunking..."
python3 scripts/preprocess.py --in corpus.json --out chunks.json

# Ensure sentence-transformers is installed, as it's required by build_index.py
python3 -m pip install sentence-transformers || true

# 4.4 Build indices (this will download models and embed chunks - may take long)
echo "Building indices (smoke-test with --max_chunks 500)..."
python3 scripts/build_index.py --chunks chunks.json --out_dir indices --max_chunks 500

# 4.5 Generate questions (100 Qs)
echo "Generating questions..."
# Fix: Remove the problematic import from generate_questions.py
sed -i 's/from generate import generate_answer_if_needed/d' scripts/generate_questions.py
python3 scripts/generate_questions.py --chunks chunks.json --out questions.json --num_questions 100

# 4.6 Run evaluation (assumes indices exist)
echo "Running evaluation..."
# Fix: Remove the problematic import from evaluate.py
sed -i 's/from generate import generate_answer_if_needed/d' scripts/evaluate.py
python3 scripts/evaluate.py --indices indices --chunks chunks.json --questions_in questions.json --report_out report.json

echo "Pipeline completed successfully (check outputs in the Workdir)."
```

Colab_HybridRag.ipynb

Files

- HybridRag
- Colab Notebooks
- HybridRag
- app
- streamlit_app.py
- indices
- scripts
- build_index.py
- data_collection.py
- evaluate.py
- fixed_urls_generat...
- generate.py
- generate_questio...
- preprocess.py
- retrieve.py
- Colab_HybridRag.ipynb
- README.md
- chunks.json
- corpus.json

Output:

```
Requirement already satisfied: threadpoolctl>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from scikit-learn->sentence-transformers) (3.6.0)
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.12/dist-packages (from typer-slim->transformers<6.0.0,>4.41.0->sentence-transformers) (8.3.1)
Building indices (smoke-test with --max_chunks 500)...
Loading model all-MiniLM-L6-v2
Indices saved to indices
Generating questions...
Wrote 100 Q&A pairs to questions.json
Running evaluation...
Wrote HTML report to ./report.html
Pipeline completed successfully (check outputs in the Workdir).
100% |██████████| 200/200 [00:00<00:00, 853.51it/s]
Warning: You are sending unauthenticated requests to the HF Hub. Please set a HF_TOKEN to enable higher rate limits and faster downloads.
Loading weights: 100% |██████████| 103/103 [00:00<00:00, 2477.61it/s, Materializing param-pooler.dense.weight]
BertModel LOAD REPORT from: sentence-transformers/all-MiniLM-L6-v2
Key | Status | |
-----+-----+-----
embeddings.position_ids | UNEXPECTED | |

Notes:
- UNEXPECTED :can be ignored when loading from different task/architecture; not ok if you expect identical arch.
Batches: 100% |██████████| 8/8 [00:02<00:00, 3.26it/s]
Loading weights: 100% |██████████| 134/134 [00:00<00:00, 4089.09it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
The tied weights mapping and config for this model specifies to tie shared.weight to encoder.embed_tokens.weight, but both are present in the checkpoints, so we will NOT tie them.
The tied weights mapping and config for this model specifies to tie shared.weight to decoder.embed_tokens.weight, but both are present in the checkpoints, so we will NOT tie them.
5% |██████████| 99/1978 [01:19<25:00, 1.24it/s]
Loading weights: 100% |██████████| 103/103 [00:00<00:00, 2420.41it/s, Materializing param-pooler.dense.weight]
BertModel LOAD REPORT from: sentence-transformers/all-MiniLM-L6-v2
Key | Status | |
-----+-----+-----
embeddings.position_ids | UNEXPECTED | |
```

```
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2312.59it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2251.51it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2336.97it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 3391.82it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 3058.13it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2292.75it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2615.69it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2383.32it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 282/282 [00:00:00:00, 2428.56it/s, Materializing param-shared.weight]
The tied weights mapping and config for this model specifies to tie shared.weight to lm_head.weight, but both are present in the checkpoints, so we will NOT tie them. You should up
Loading weights: 100% [██████████] 389/389 [00:00:00:00, 3744.50it/s, Materializing param-encoder.layer.23.output.dense.weight]
RobertaModel LOAD REPORT from: roberta-large
Key | Status
-----|-----
lm_head.layer_norm.weight | UNEXPECTED
lm_head.bias | UNEXPECTED
roberta.embeddings.position_ids | UNEXPECTED
lm_head.dense.weight | UNEXPECTED
lm_head.dense.bias | UNEXPECTED
lm_head.layer_norm.bias | UNEXPECTED
pooler.dense.weight | MISSING
pooler.dense.bias | MISSING

Notes:
- UNEXPECTED :can be ignored when loading from different task/architecture; not ok if you expect identical arch.
- MISSING :those params were newly initialized because missing from the checkpoint. Consider training on your downstream task.
```

This script runs all the scripts present inside scripts folder which in turn creates indices, runs models does evaluations, stores in db etc.

Evaluation:

Questions.json:

```
File Edit Selection View Go Run Terminal Help
CAIRAG
report.json questions.json
Group_149_Hybrid_RAG > questions.json > ...
1 {
2   {
3     "question": "What was Guido van Rossum's first release?",
4     "answer": "Python is a high-level, general-purpose programming language",
5     "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
6     "chunk_id": "3c495284e54283a33925d1c80c2d989ba41b5e2e"
7   },
8   {
9     "question": "What is the name of the Python 2.0 release?",
10    "answer": "comedy series Monty Python's Flying Circus",
11    "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
12    "chunk_id": "24ead345ec0befbfc52d6b6cd002534708163d1d"
13  },
14  {
15    "question": "What is the Zen of Python?",
16    "answer": "offers some support for functional programming in the \"Lisp tradition\"",
17    "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
18    "chunk_id": "bad8708151484eb4fb089813d2cf874b159bfce0"
19  },
20  {
21    "question": "What is the name of the Python community?",
22    "answer": "\"To describe something as 'clever' is not considered a compliment in the Python culture",
23    "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
24    "chunk_id": "34918d3373bd5943eeez2ba61d5450865113dc680"
25  },
26  {
27    "question": "What is the recommended size of a Python statement?",
28    "answer": "The recommended indent size is four spaces",
29    "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
30    "chunk_id": "d07b5ce375aef622ed20e8361cef35a4b25dce98"
31  },
32  {
33    "question": "What is the name of the variable that Python uses for comparing objects?",
34    "answer": "function The import and from statements, used to import modules whose functions or variables can be used in the cu
35    "url": "https://en.wikipedia.org/wiki/Python_(programming_language)",
36    "chunk_id": "9dfff8671f08a60dccc34a4395e3c71e36d4cdb87e"
37  },
38  {
39    "question": "What is the name of the expression that Python uses to describe tuples?",
40    "answer": "This is kind of expression called a list comprehension, and a more general expression called a generator expression"
41  }
42 }
```

Report.json:

