

# Page Number [Metadata retrieval] In query result.

The result contains the file name and Page number Successfully

The screenshot displays the AI Applications interface with a sidebar on the left containing 'App overview', 'Data', 'Preview', 'Configurations', 'Integration', and 'Analytics'. The main content area shows search results for two queries. The first query, 'What are some effective querying methods for Graph-Based RAG?', returns results from 'paper3.pdf' (Page 44) and '[N/A]'. The second query, 'How are trap words used in the adversarial dataset?', returns results from 'paper1' (Page 5) and '[N/A]'. Both results include a disclaimer: 'Generative AI may display inaccurate information, including about people, so double-check its responses.'

**Query 1: What are some effective querying methods for Graph-Based RAG?**

Generative AI may display inaccurate information, including about people, so double-check its responses.

[File: paper3.pdf] [Page: 44] [N/A]

Graph-Based RAG systems employ several strategies for generating context-relevant answers using LLMs. These strategies include the From Local to Global Graph RAG approach, which uses a two-stage summarization pipeline. This approach locates "communities" of entities and relationships that cohere around a topic, generating partial answers specific to each subgraph, which are then merged in a subsequent global step. Another approach, LightRAG, uses a dual-level retrieval paradigm aligned with different user query types. For highly specific queries, it retrieves low-level nodes and edges, providing precise snippets for the LLM. For abstract or conceptual queries, LightRAG assembles larger subgraphs that capture broader themes or relationships.

Techniques such as offline community detection, incremental graph updates, subgraph summarization, and parallel querying are pivotal for keeping costs and latency within acceptable bounds. Hierarchical summarization distills large subgraphs or communities into concise

**Query 2: How are trap words used in the adversarial dataset?**

Generative AI may display inaccurate information, including about people, so double-check its responses.

[File: paper1] [Page: 5] [N/A]

Trap words are used in the adversarial dataset by identifying word pairs where the suffix (or whole) of the first word can be concatenated with the prefix (or whole) of the second word to form a token existing in the vocabulary, while the concatenation of the two original words is not a token. This "trap word" is then used to create instances that challenge the tokenization of LLMs.

There are three scenarios related to how the trap word is used:

1. If Word 1 is included by Trap Word, this corresponds to inserting a span after the Trap Word.
2. If Word 2 is included by Trap Word, this corresponds to inserting a span before the Trap Word.
3. If neither Word 1 nor Word 2 is included by Trap Word, this corresponds to inserting a span both before and after the Trap Word.

Google Cloud

rag-agent-project

Search (/) for resources, docs, products, and more

Search

7

A

AI Applications

Apps > rag-agent-app > Configurations

Feedback on AI Applications

App overview

Data

Preview

Configurations

Integration

Analytics

Autocomplete

UI

Tuning

Serving

Advanced

Control

Gemini 1.5 Pro

The Vertex AI 1M token context window model. [Learn more about Gemini 1.5 Pro](#)

Customize answer

Tailor the tone, style, and length of generated answers.

Instructions

For every retrieved chunk, show metadata before the answer:

```
{ [File: {{source_file | default("N/A")}}] [Page: {{page | default("N/A")}}] [Paragraph: {{paragraph_index | default("N/A")}}] [Section: {{section_heading | default("N/A")}}]
```

Replace with a template Reset

Summary language

Preview

English

Choose the language of the summary

Preview

What are some prompt engineering techniques used in RAG?

Generative AI may display inaccurate information, including about people, so double-check its responses.

[File: paper3.pdf] [Page: 1] [Paragraph: 1] [Section: N/A]  
Prompt engineering techniques used in Retrieval-Augmented Generation (RAG) include role and format instructions, zero, one and few-shot prompting, chain-of-thoughts, self-consistency, and tree-of-thought prompting. Prompt engineering is also used to prevent prompt-based attacks. [paper3](#)

[File: paper3.pdf] [Page: 4] [Paragraph: 1] [Section: N/A]  
RAG incorporates prompt engineering as a key area. [paper3](#)

[File: paper3.pdf] [Page: 2] [Paragraph: 1] [Section: N/A]  
Prompt engineering involves crafting and organizing input text to guide and optimize the model's output, influencing style, content, and reasoning. Techniques like few-shot and zero-shot prompting enhance flexibility and reduce adaptation costs, while chain-of-thought prompts improve transparency

Save and publish

Widget Configurations Experience

# Chunked Meta-data successfully stored in Data Storage

Google Cloud

rag-agent-project

Search (/) for resources, docs, products, and more

Search

7

A

AI Applications

Apps > rag-agent-app > Data

Feedback on AI Applications

Learn

App overview

Data

Preview

Configurations

Integration

Analytics

An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk1	N/A	Indexed: 7/8/2025, 5:02:20 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk10	N/A	Indexed: 7/8/2025, 3:19:10 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk2	N/A	Indexed: 7/8/2025, 5:02:22 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk3	N/A	Indexed: 7/8/2025, 5:02:21 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk4	N/A	Indexed: 7/8/2025, 5:02:21 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk5	N/A	Indexed: 7/8/2025, 3:19:07 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk6	N/A	Indexed: 7/8/2025, 3:19:09 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk7	N/A	Indexed: 7/8/2025, 3:19:07 PM, America/New_York	View Document
An_Effective_Retrieval_Method_to_Improve_RAG_Performance_chunk8	N/A	Indexed: 7/8/2025, 3:19:13 PM, America/New_York	View Document

If Page number not present in document: It does not take page number

AI Applications

App overview

Data

Preview

Configurations

Integration

Analytics Preview

Apps > rag-agent-app > Search Preview

What is retrieval-augmented generation described in these PDFs?

Retrieval-augmented generation (RAG) is a general purpose machine learning approach that combines pre-trained, parametric-memory generation models with a non-parametric memory.

paper3

NeurIPS-202...

[File: An\_Effective\_Retrieval\_Method\_to\_Improve\_RAG\_Performance.pdf]

[N/A]

[An Effective Retrieval Method to Improve RAG Performance]

Retrieval-Augmented Generation (RAG) combines retrieval and generation processes to enhance the relevance, accuracy, and diversity of LLM responses.

paper3

An\_Effective...

📄

:

Suggestions

How does RAG mitigate hallucinations in LLMs?

What are some examples of RAG applications?

What are the key components of the RAG workflow?