RAG use cases - Master thesis presentations

Retrieval Augmented Generation with Open-Source / Offline Models
Multimodal Retrieval Augmented Generation

Ioannis Partalas
Monica Riedler

1 Retrieval Augmented Generation with Open-Source / Offline Models

Ioannis Partalas

Table of Contents

- Motivation
- Approaches
- Large Language Models
- Evaluation
  - Metrics
  - Results
- Conclusion

Motivation

Motivation – Why offline models?

Cost Effectiveness:

- free of cost to download
  - long-term use
- especially advantageous in research and development settings where continuous access to large models is needed

Data Privacy:

- allows keeping all data in-house.
  - particularly important in domains dealing with sensitive or proprietary information where data privacy and security are paramount

Transparency:

- access to the workings of LLMs, including their source code, architecture, training data, and mechanism for training and inference

Customization and Control:

- modifiable aspects of the model to better suit specific needs or to innovate on the model architecture itself (e.g. Model Fine-Tuning)

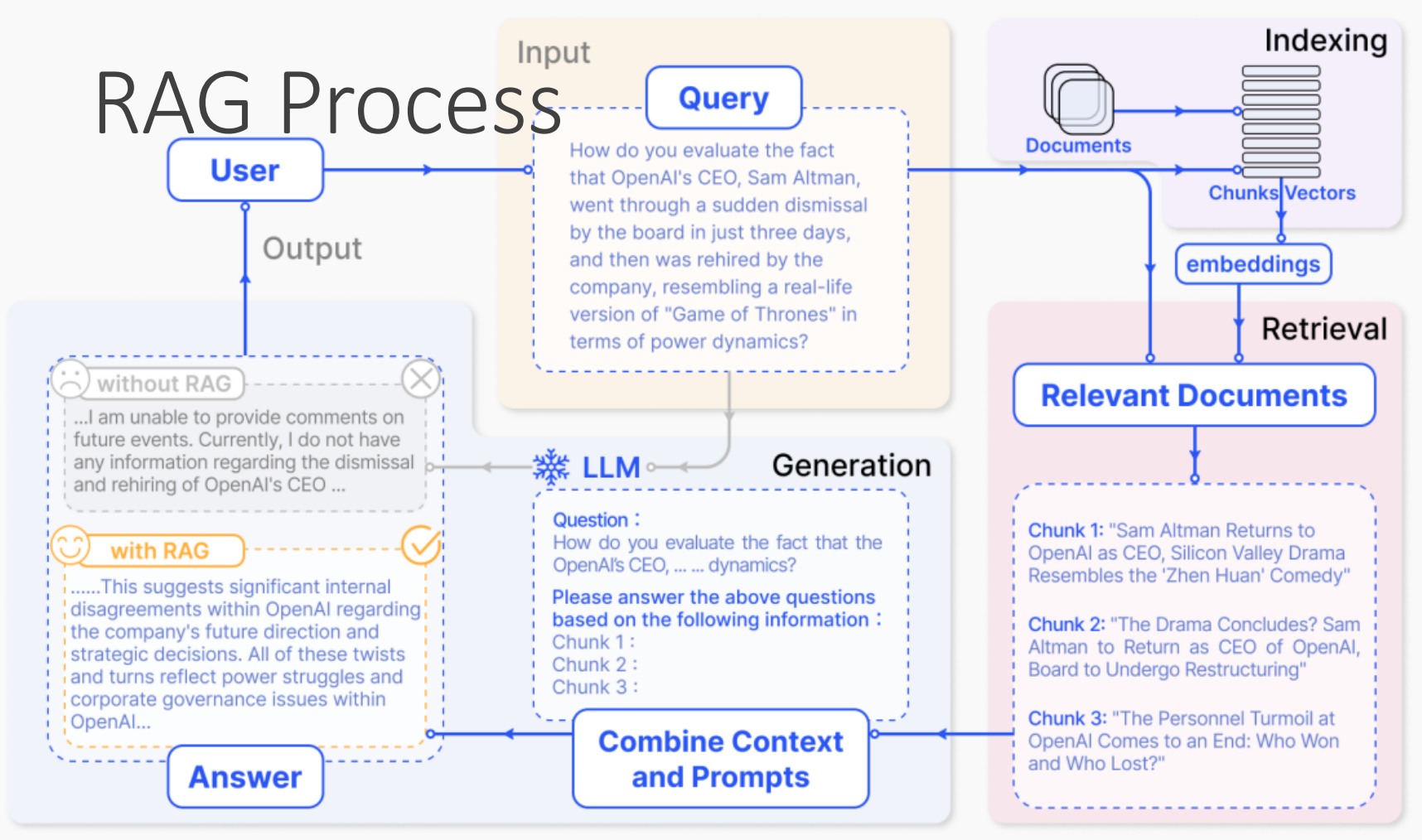
Reproducibility:

- model's behavior is not subject to changes from ongoing training or updates that can occur with online models
- can aid in reproducibility of research

Internet Independence:

- operate without the need for a constant internet connection (assuming local GPU availability)

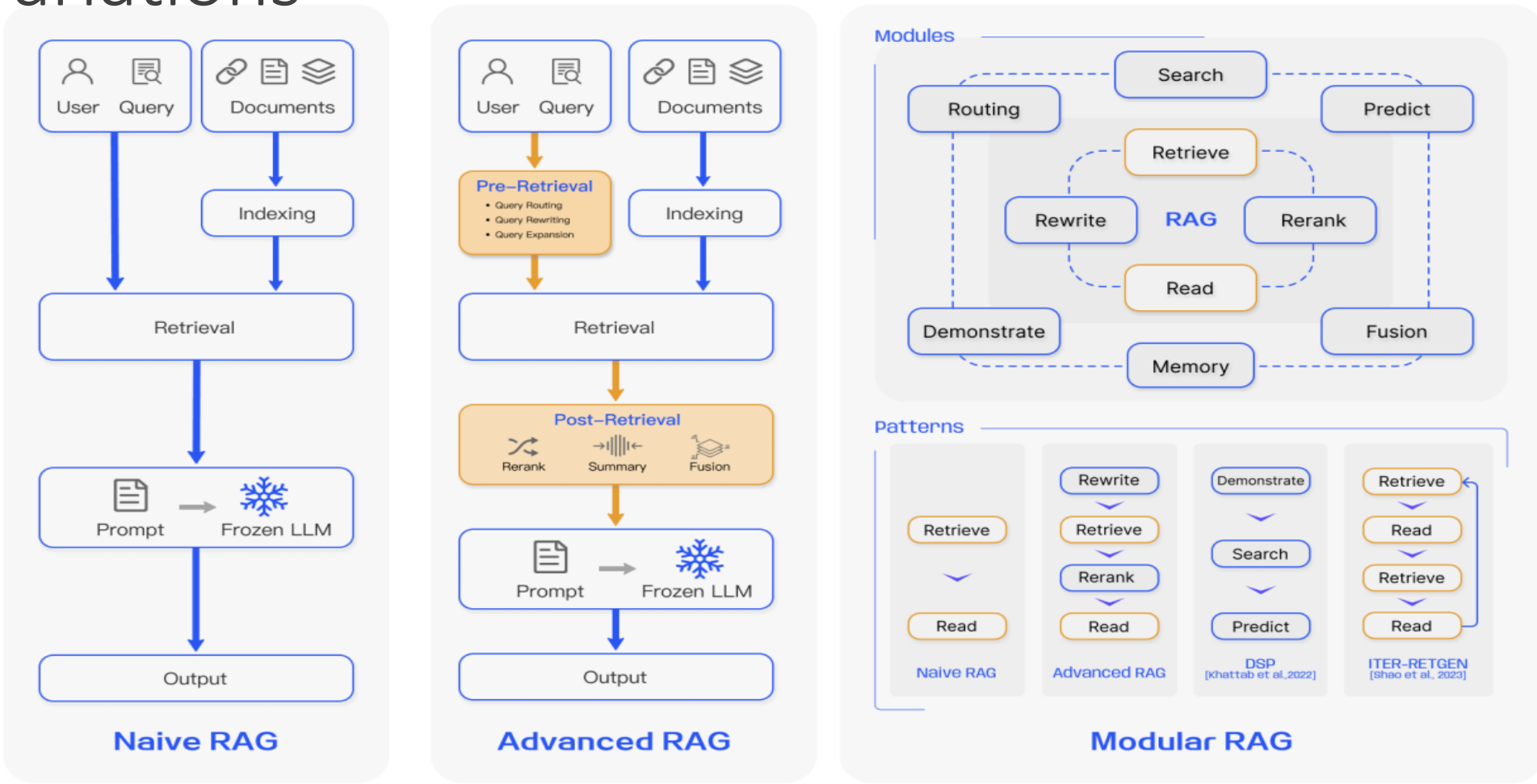
Approaches



A representative instance of the RAG process applied to question answering. It mainly consists of 3 steps. 1) Indexing. Documents are split into chunks, encoded into vectors, and stored in a vector database. 2) Retrieval. Retrieve the Top k chunks most relevant to the question based on semantic similarity. 3) Generation. Input the original question and the retrieved chunks together into LLM to generate the final answer

[2312.10997] Retrieval-Augmented Generation for Large Language Models: A Survey (arxiv.org)

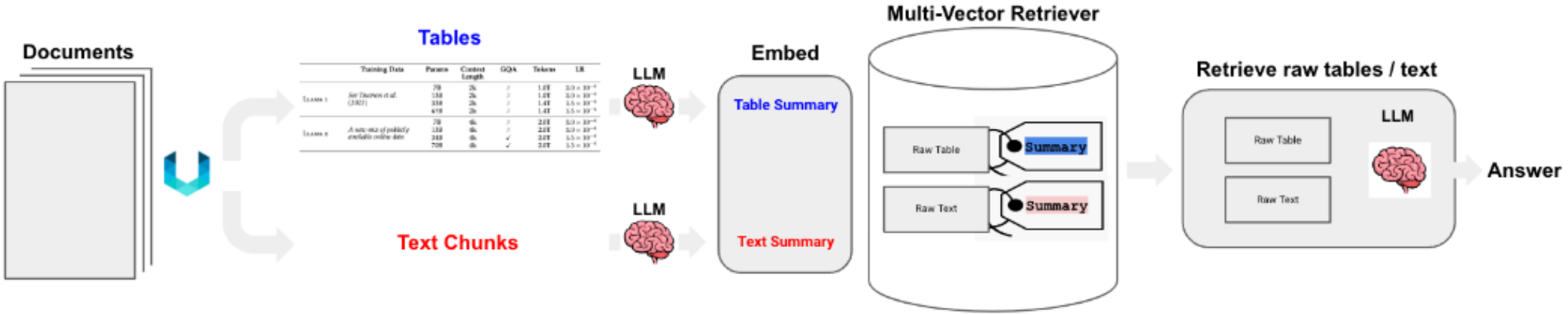
RAG Variations



- Naive RAG mainly consists of three parts: indexing, retrieval and generation.
- Advanced RAG proposes multiple optimization strategies around pre-retrieval and post-retrieval, with a process similar to the Naive RAG, still following a chain-like structure.
- Modular RAG inherits and develops from the previous paradigm, showcasing greater flexibility overall. This is evident in the introduction of multiple specific functional modules and the replacement of existing modules. The overall process is not limited to sequential retrieval and generation; it includes methods such as iterative and adaptive retrieval.

[2312.10997] Retrieval-Augmented Generation for Large Language Models: A Survey (arxiv.org)

Semi-Structured RAG



Chunking for semi-structured formats (e.g. PDF):

- segment a PDF document by using a document image analysis model
- apply OCR for the table content

langchain/cookbook/Semi_Structured_RAG.ipynb at master · langchain-ai/langchain (github.com)

Common Pitfalls in RAG Pipelines

Retrieval Step

- the retriever is responsible for the retrieval step
- the **retrieval context** (i.e. a list of text chunks) is what the retriever retrieves
- **Does the embedding model you're using capture domain-specific nuances?** (If you're working on a medical use case, a generic embedding model offered by OpenAI might not embed optimally the retrieval context)
- **Does your reranker model rank the retrieved nodes in the "correct" order?**
- **Are you retrieving the right amount of information?** This is influenced by hyperparameters text chunk size, top-K number

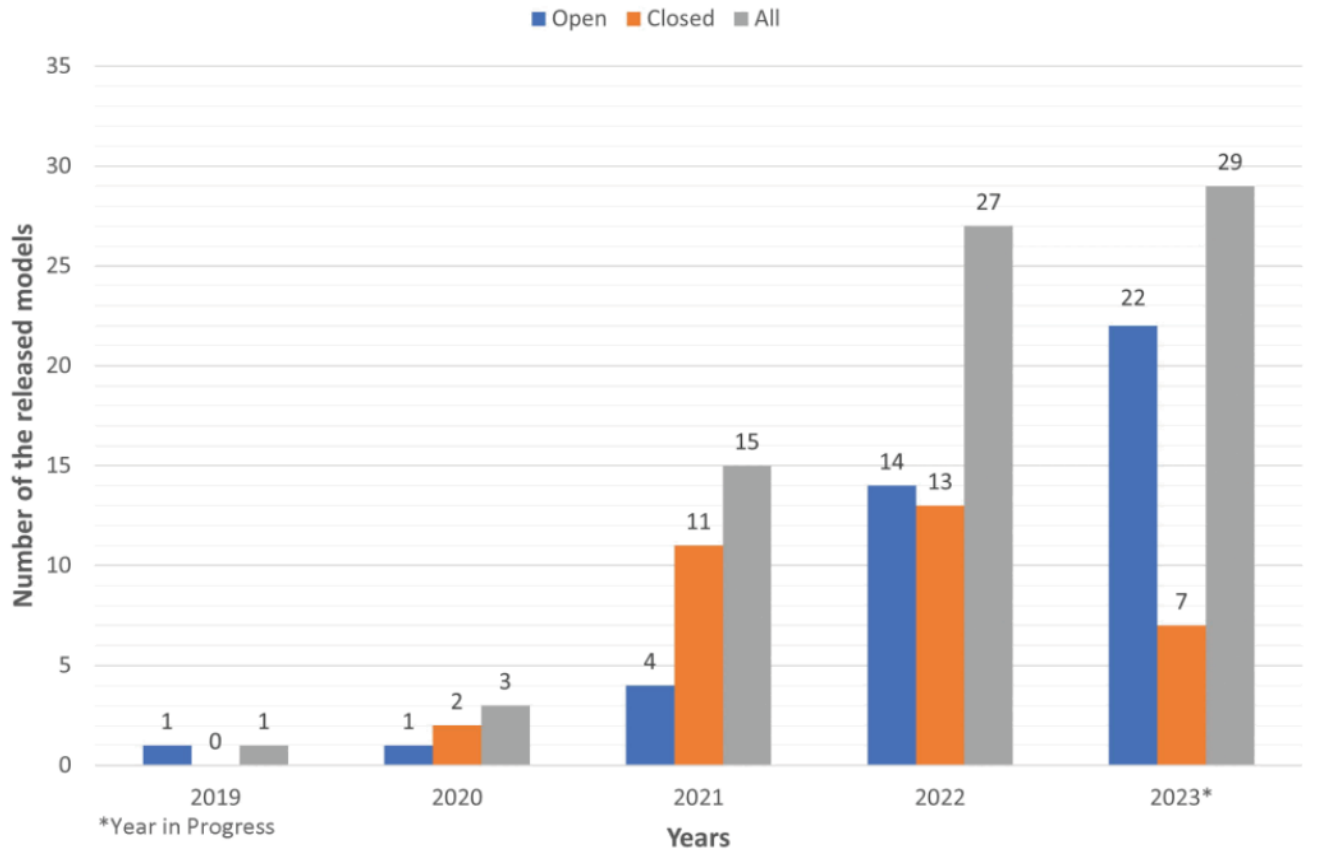
Generation Step

- the generator is responsible for the generation step
- the **LLM output** is what the generator generates
- **Can you use a smaller, faster, cheaper LLM?** This often involves exploring open-source alternatives like LLaMA-3, Mistral 7B, and fine-tuning your own versions of it
- **Would a higher temperature give better results?**
- **How does changing the prompt template affect output quality?** This is where most LLM practitioners spend most time on

RAG Evaluation | DeepEval - The Open-Source LLM Evaluation Framework (confident-ai.com)

Large Language Models

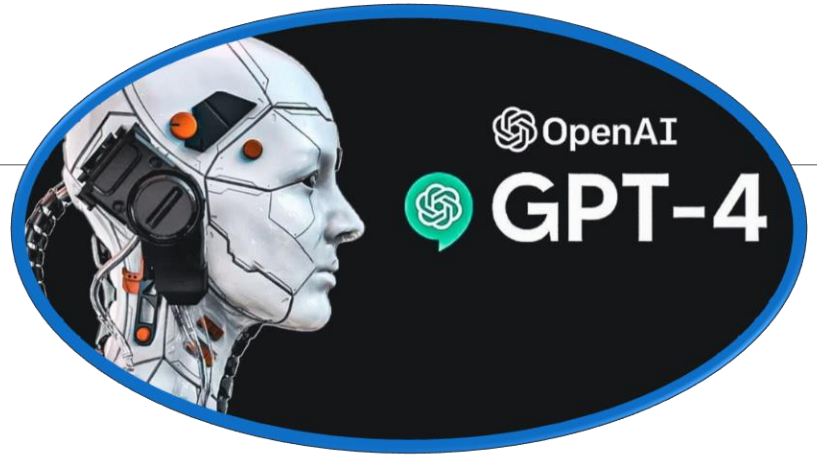
Large Language Models



The number of LLMs introduced until July 2023

[2307.06435] A Comprehensive Overview of Large Language Models (arxiv.org)

Large Language Models



Large Language Models Usage – Common Questions

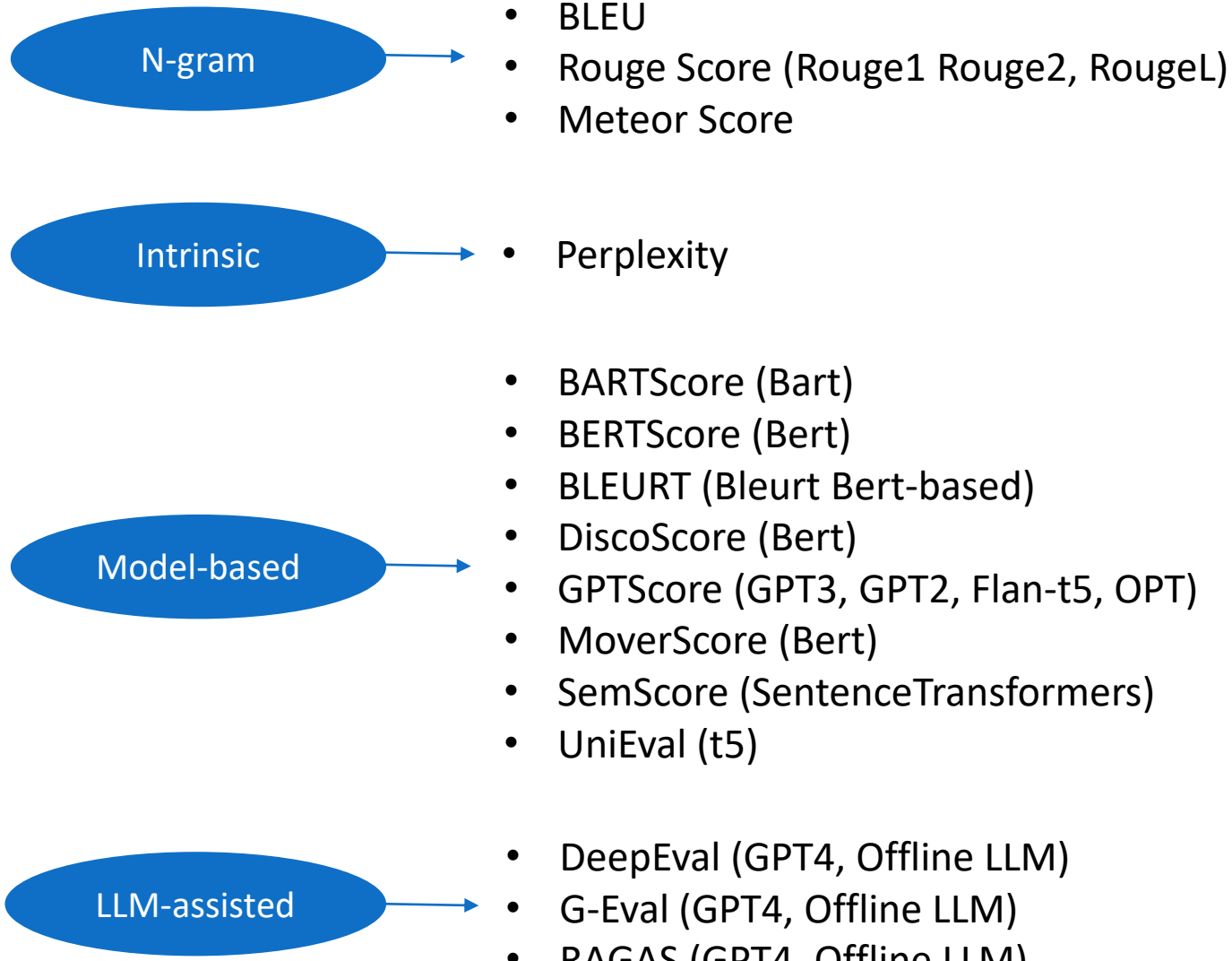
- Where to find?
  - Models - Hugging Face
  - Ollama

- How to select?
  - Open LLM Leaderboard - a Hugging Face Space by open-llm-leaderboard
  - Chat with Open Large Language Models (lmsys.org)

- What if the model size exceeds my GPU capabilities?
  - Quantization (huggingface.co)
  - AWQ, AutoGPTQ, Bitsandbytes etc.

Metrics

Metrics


N-gram
BLEU
Rouge Score (Rouge1 Rouge2, RougeL)
Meteor Score
Intrinsic
Perplexity
Model-based
BARTScore (Bart)
BERTScore (Bert)
BLEURT (Bleurt Bert-based)
DiscoScore (Bert)
GPTScore (GPT3, GPT2, Flan-t5, OPT)
MoverScore (Bert)
SemScore (SentenceTransformers)
UniEval (t5)
LLM-assisted
DeepEval (GPT4, Offline LLM)
G-Eval (GPT4, Offline LLM)
RAGAS (GPT4, Offline LLM)

Metrics: LLM-assisted

Concept:

- exploring the use of “LLMs as a judge” for automated evaluation by using powerful LLMs such as GPT-4 to perform evaluation for the LLM outputs.

Challenges:

- **Alignment with Human Grading**: how well does an LLM judge’s grading reflect the actual human preference in terms of correctness, readability and comprehensiveness of the answers?
- **Appropriate Grade Scales**: What grading scale is recommended because different grading scales are used by different frameworks (e.g., AzureML uses 0 to 100 whereas langchain uses binary scales)?
- **Accuracy through Examples**: What’s the effectiveness of providing a few grading examples to the LLM judge and how much does it increase the reliability and reusability of the LLM judge on different metrics?
- **Applicability Across Use Cases**: With the same evaluation metric (e.g. correctness), to what extent can the evaluation metric be reused across different use cases (e.g. casual chat, content summarization, retrieval-augmented generation)?

Best Practices for LLM Evaluation of RAG Applications | Databricks Blog

Metrics: LLM-assisted

Evaluating Retrieval:

- Contextual Precision Metric: evaluates whether the **reranker** in your retriever ranks more relevant nodes in your retrieval context higher than irrelevant ones.
- Contextual Recall Metric: evaluates whether the **embedding model** in your retriever is able to accurately capture and retrieve relevant information based on the context of the input.
- Contextual Relevancy Metric: evaluates whether the **text chunk size** and **top-K** of your retriever is able to retrieve information without much irrelevancies.

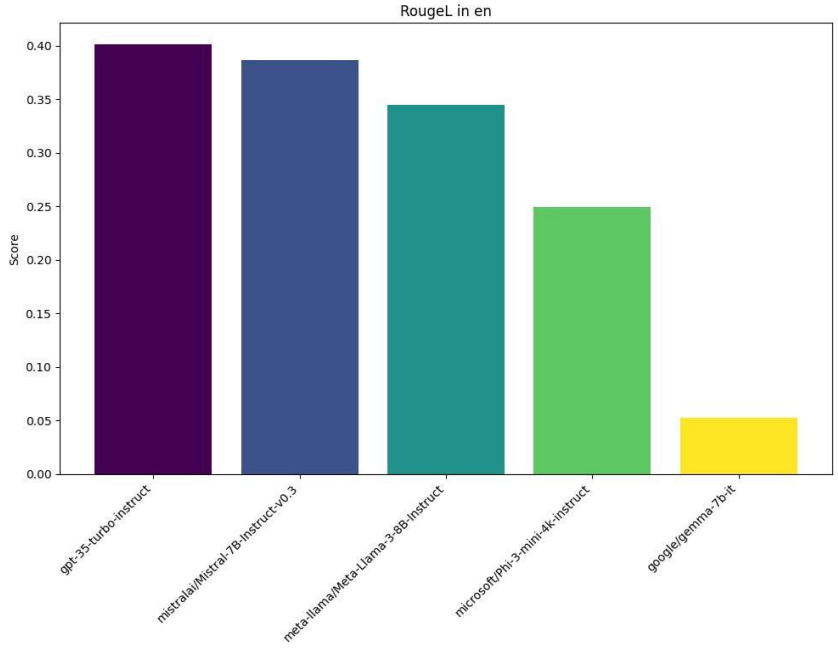
Evaluating Generation:

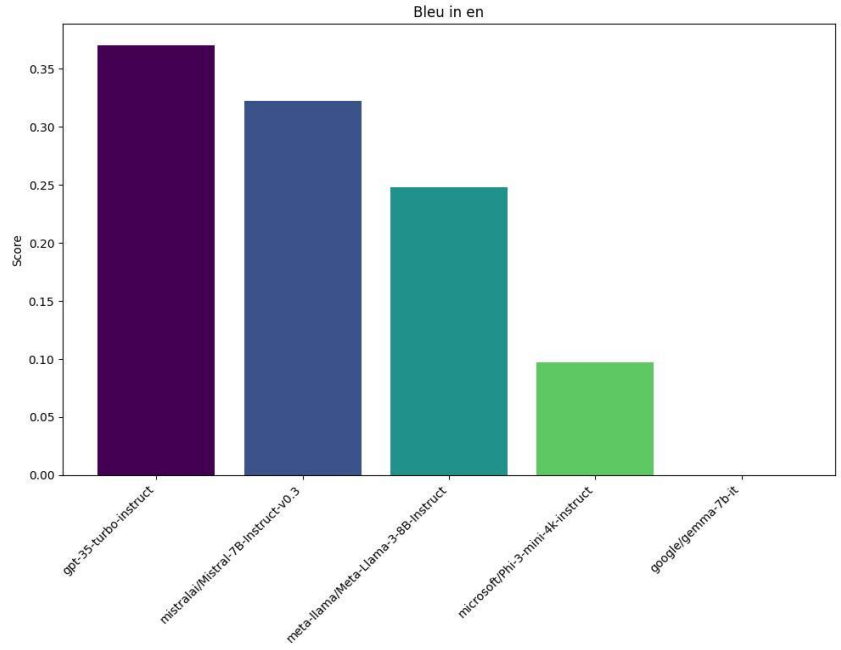
- Answer Relevancy Metric: evaluates whether the **prompt template** in your generator is able to instruct your LLM to output relevant and helpful outputs based on the retrieval_context.
- Faithfulness Metric: evaluates whether the **LLM** used in your generator can output information that does not hallucinate **AND** contradict any factual information presented in the retrieval_context.
- The GEval metric can be used to evaluate generation on customized criteria (e.g. coherence).

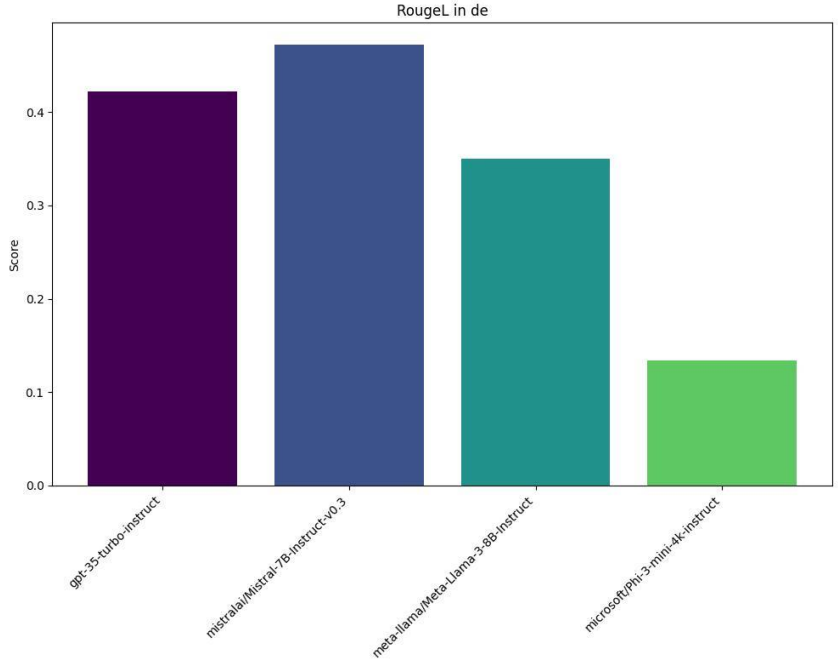
RAG Evaluation | DeepEval - The Open-Source LLM Evaluation Framework (confident-ai.com)

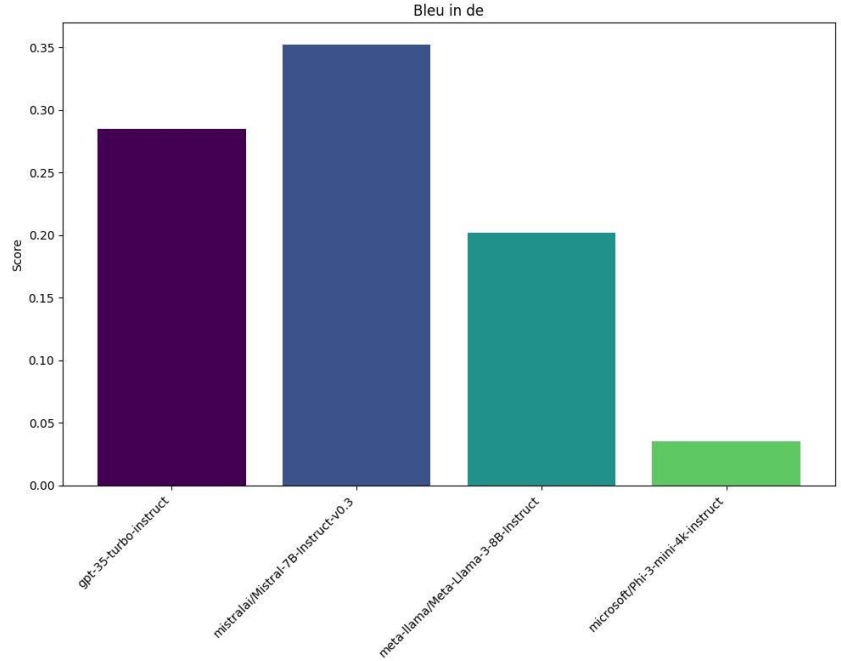
Results

Metrics: N-Gram

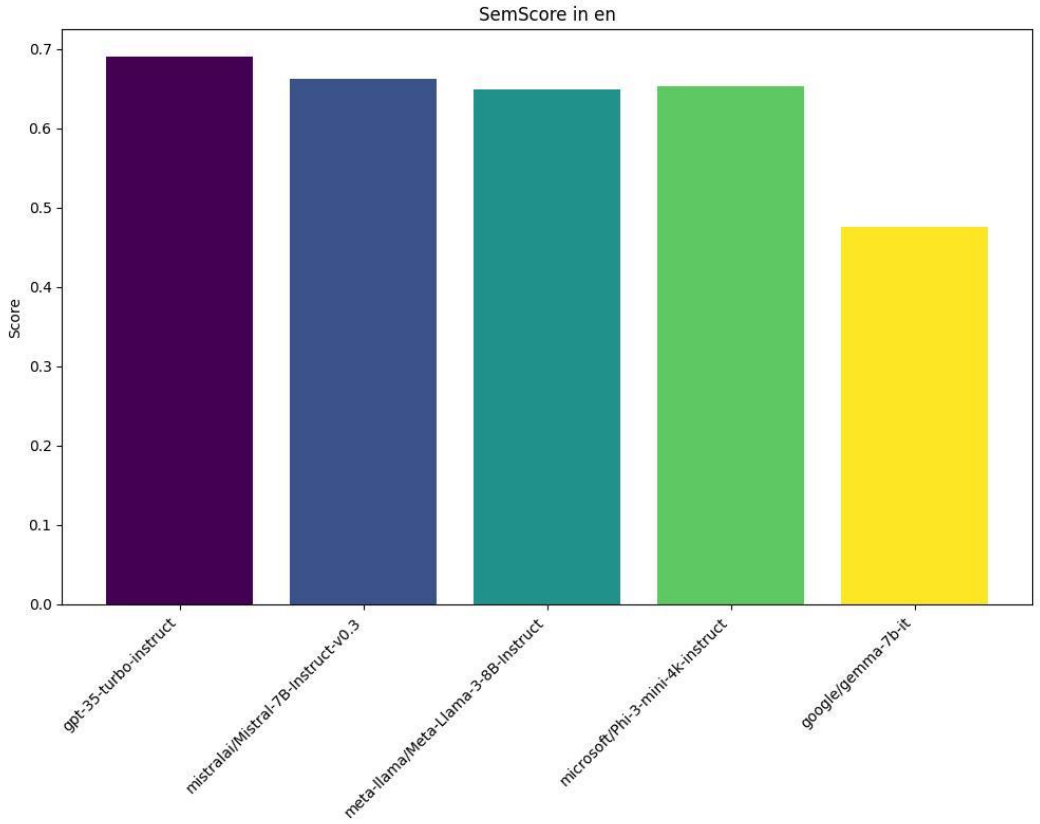


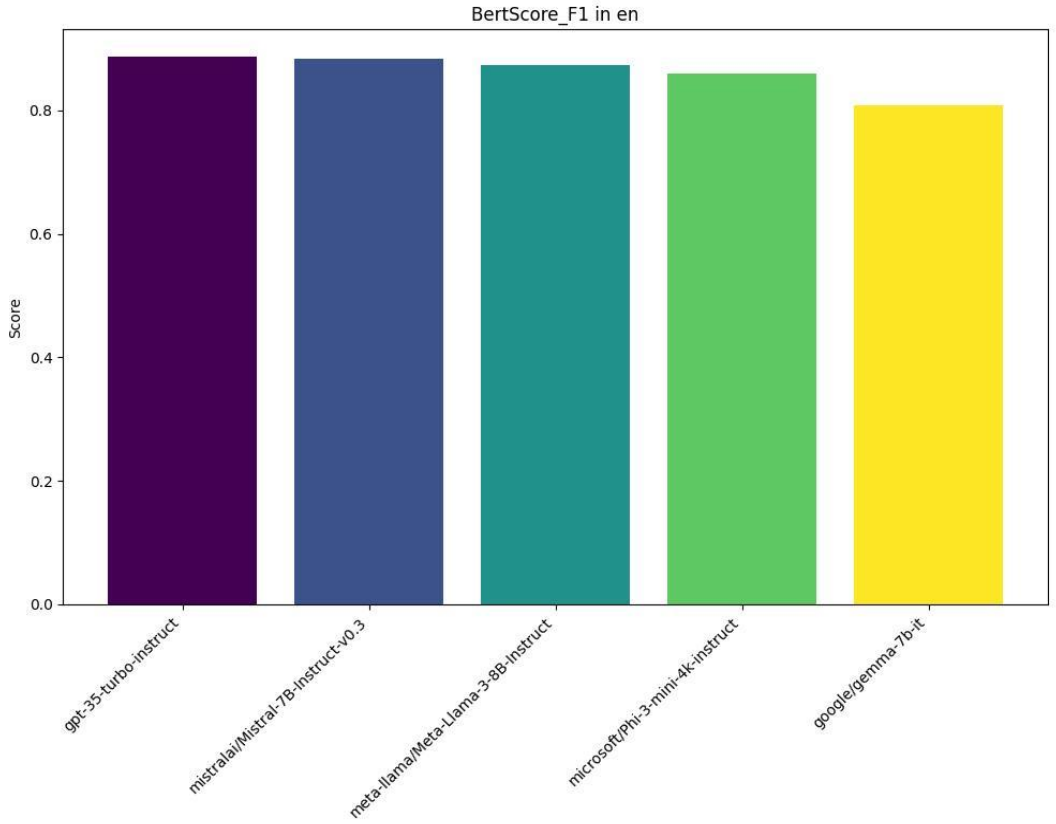




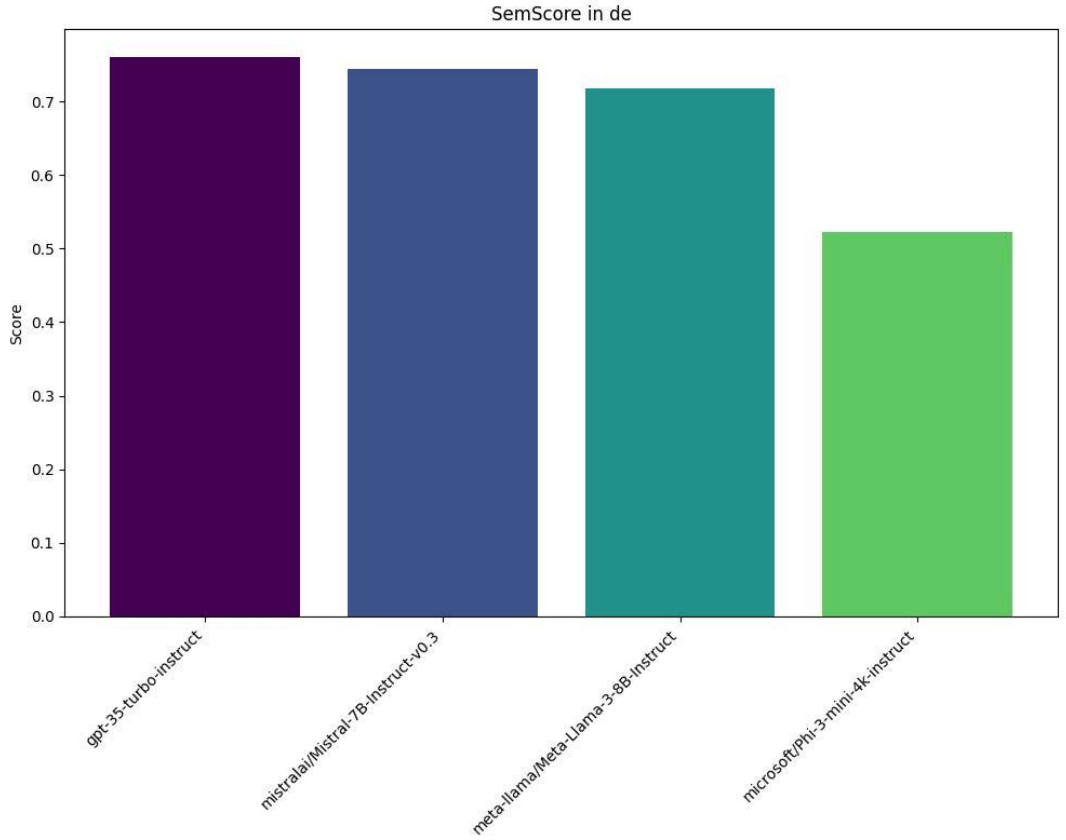


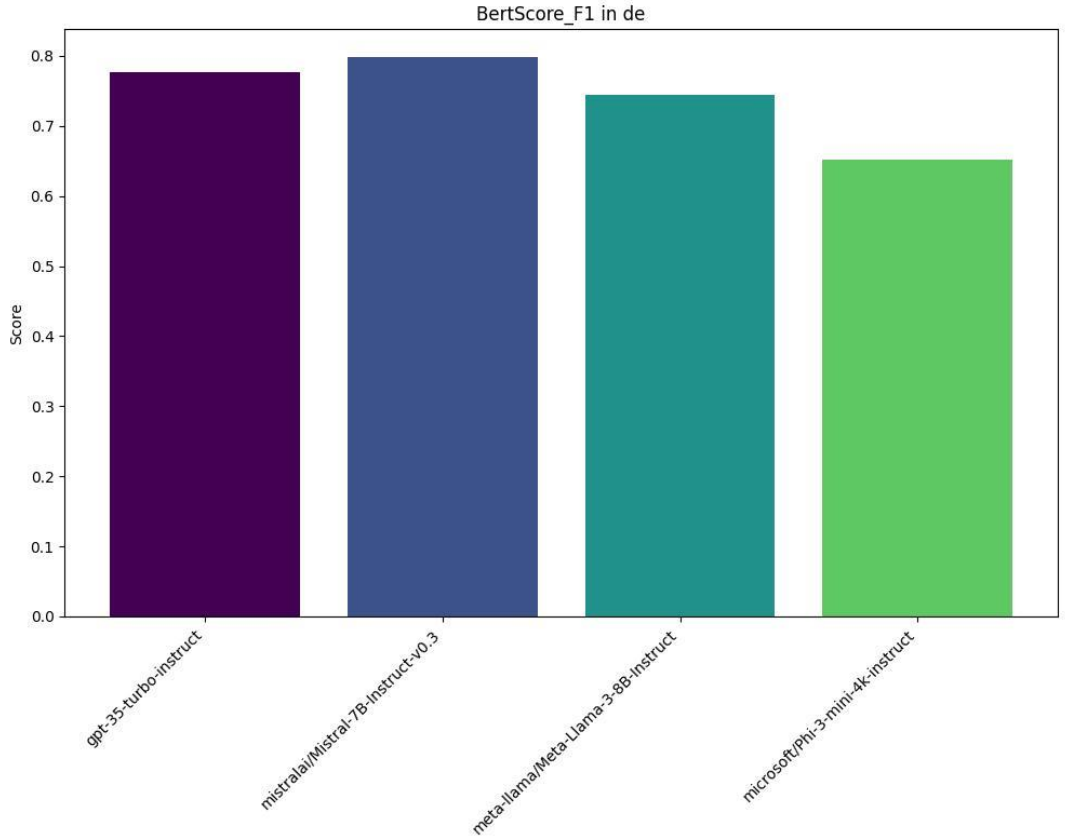
Metrics: Semantic-Similarity based



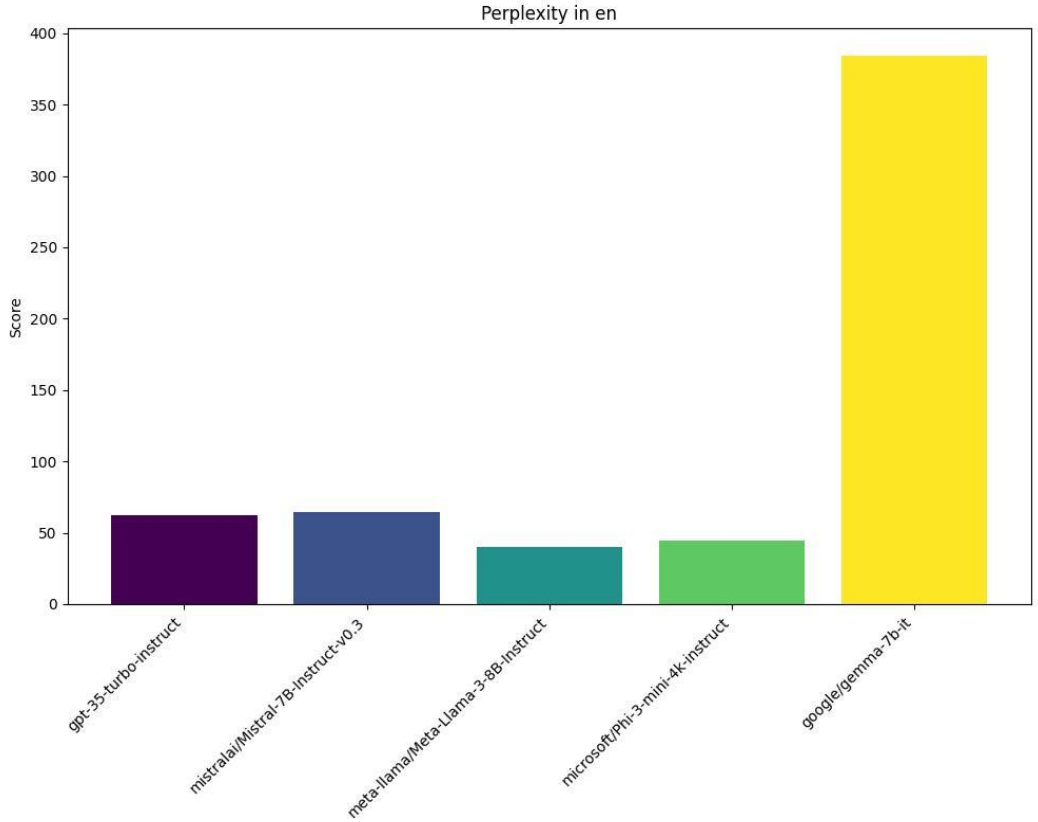


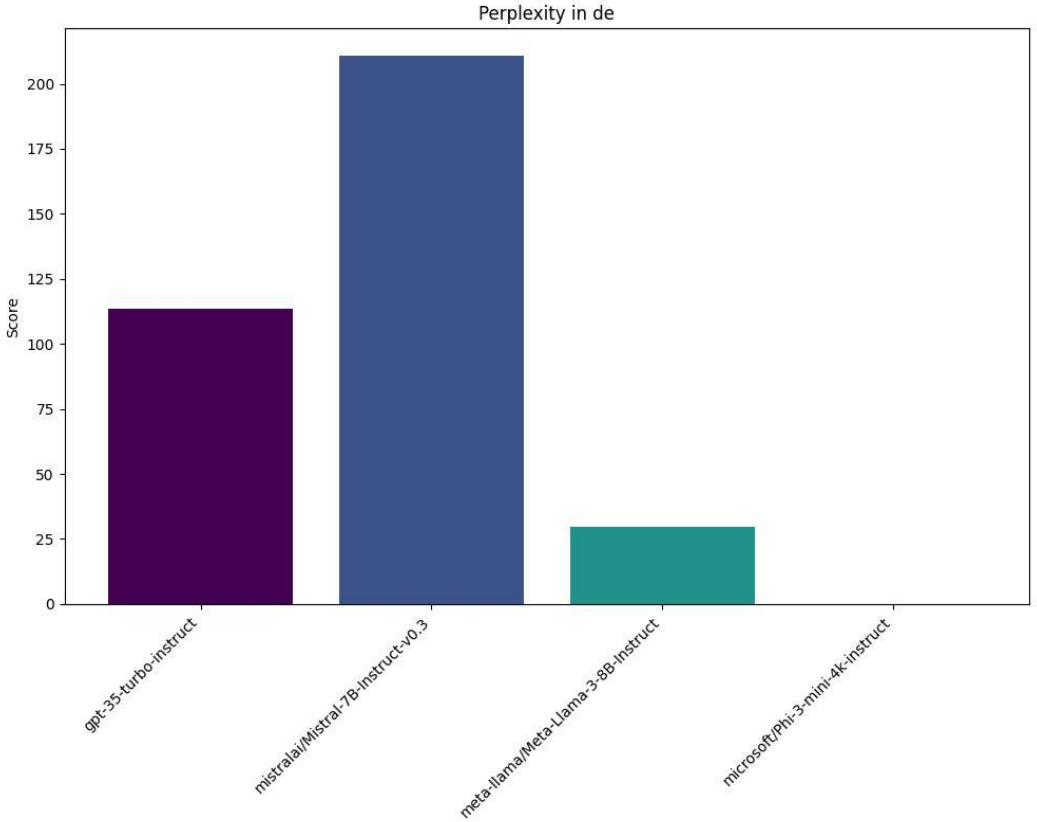
Metrics: Semantic-Similarity based





Metrics: Intrinsic





Conclusion & TODOs

Conclusion

- No final conlusion can be made yet, due to ongoing experiments.
- Mistral seems to be OpenAI's best opponent until now.
- Yet to try:
  - Translation of all data to English to compare model performance in German.
  - Application of GPT-4.
  - Run all models across the whole evaluation dataset for all metrics.
  - Quantized models of larger LLMs.

Best Practices for LLM Evaluation of RAG Applications | Databricks Blog

2 Multimodal Retrieval Augmented Generation

Monica Riedler

Table of Contents

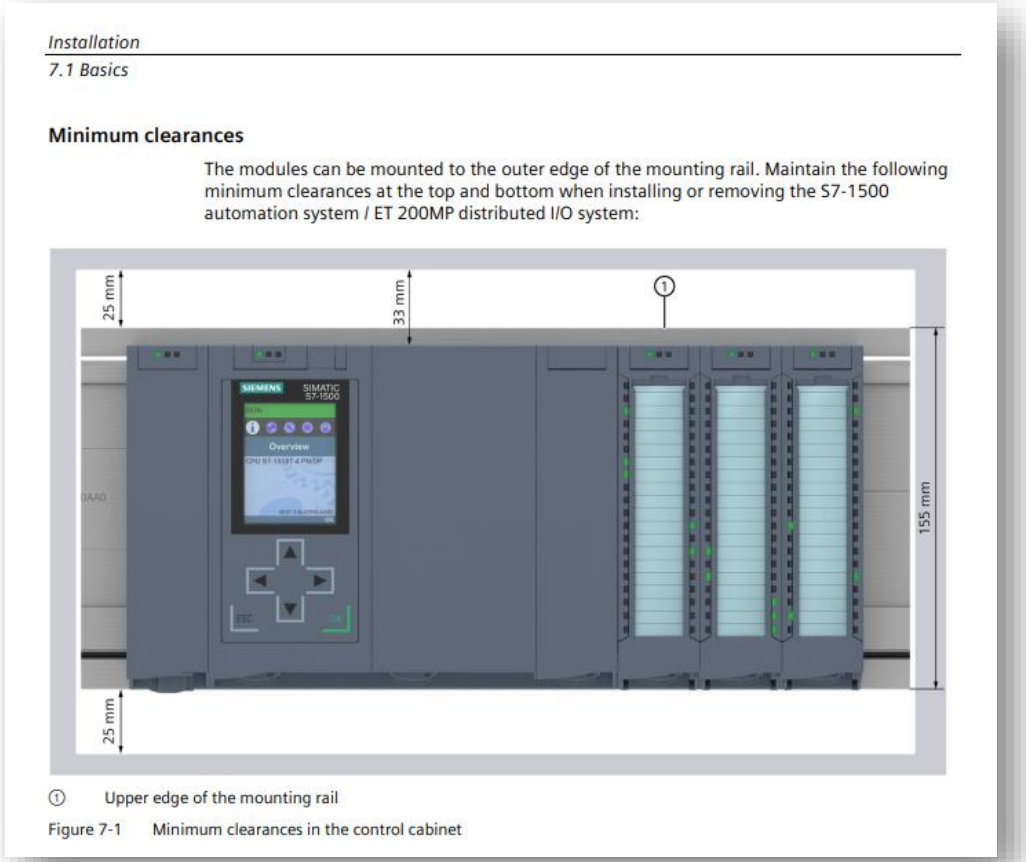
- Motivation
- Approaches:
  - Multimodal Embeddings (CLIP)
  - Text-Embeddings
  - Text-Embeddings + Original Images
- Evaluation
  - Metrics
  - Results
- Conclusion

Motivation

Motivation – Why Multimodal RAG?

- Sometimes text might not be enough
- Sometimes a piece of information is contained only in an image
- Example question:
  “What are the minimum clearances at the top and bottom when installing the S7-1500 automation system?”
- Answer:
  “The minimum clearances are 25 mm at the bottom, 25 mm at the top left and 33 mm at the top center”

Example document excerpt:



Installation

7.1 Basics

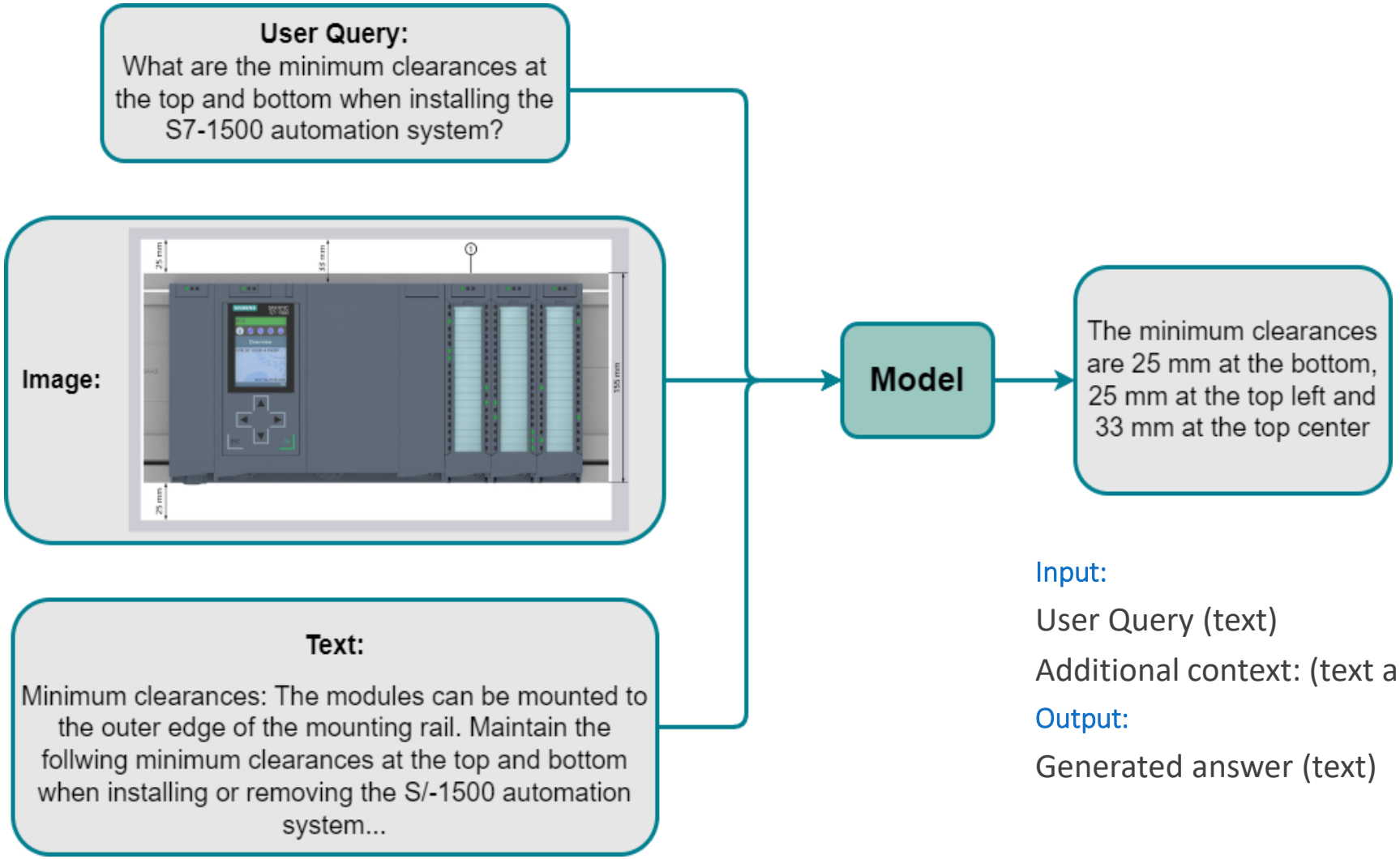
Minimum clearances

The modules can be mounted to the outer edge of the mounting rail. Maintain the following minimum clearances at the top and bottom when installing or removing the S7-1500 automation system / ET 200MP distributed I/O system:

① Upper edge of the mounting rail

Figure 7-1 Minimum clearances in the control cabinet

Motivation – Why Multimodal RAG?



Input:

User Query (text)

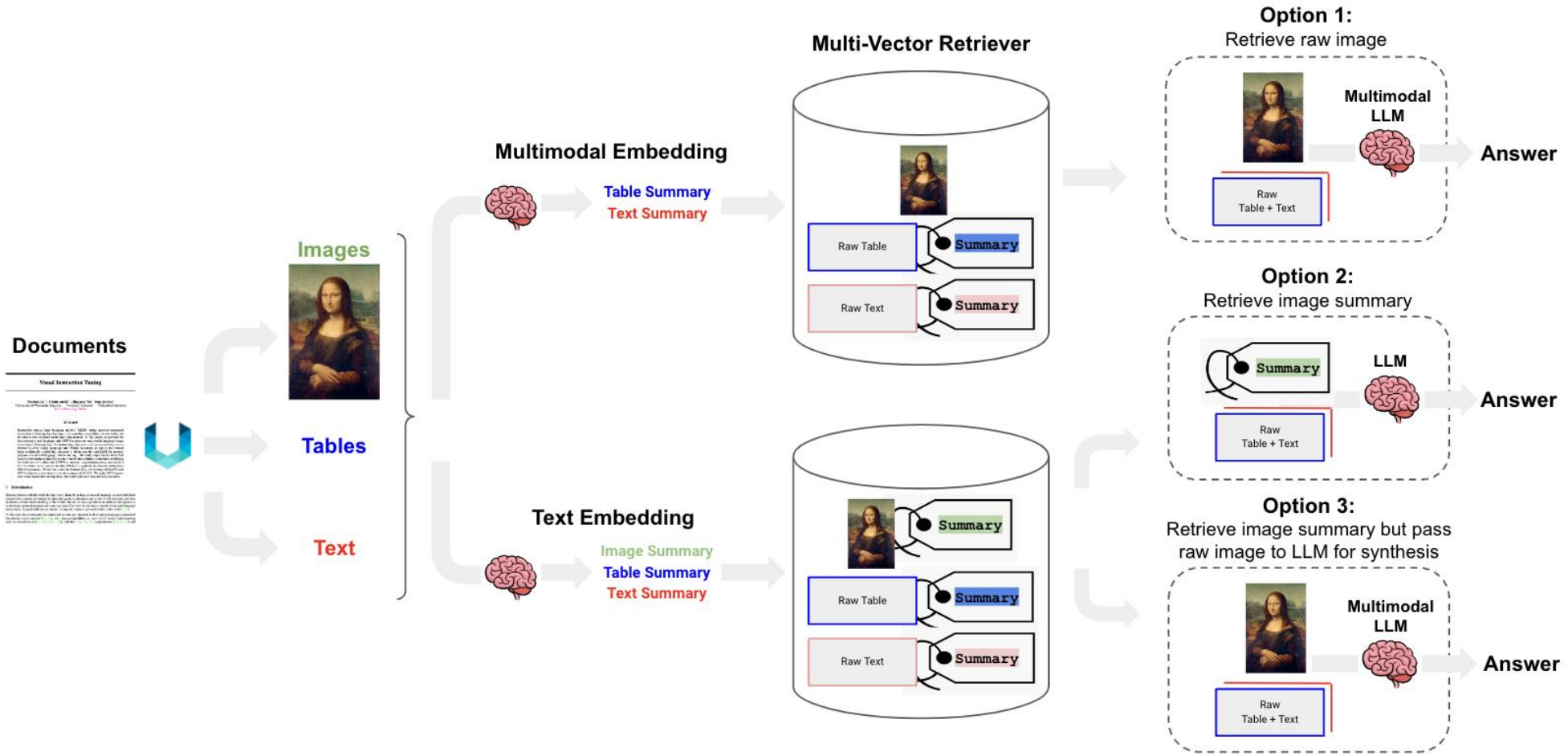
Additional context: (text and/or image)

Output:

Generated answer (text)

Approaches

- Multimodal Embeddings (CLIP)
- Text Embeddings
- Text Embeddings + Original Image



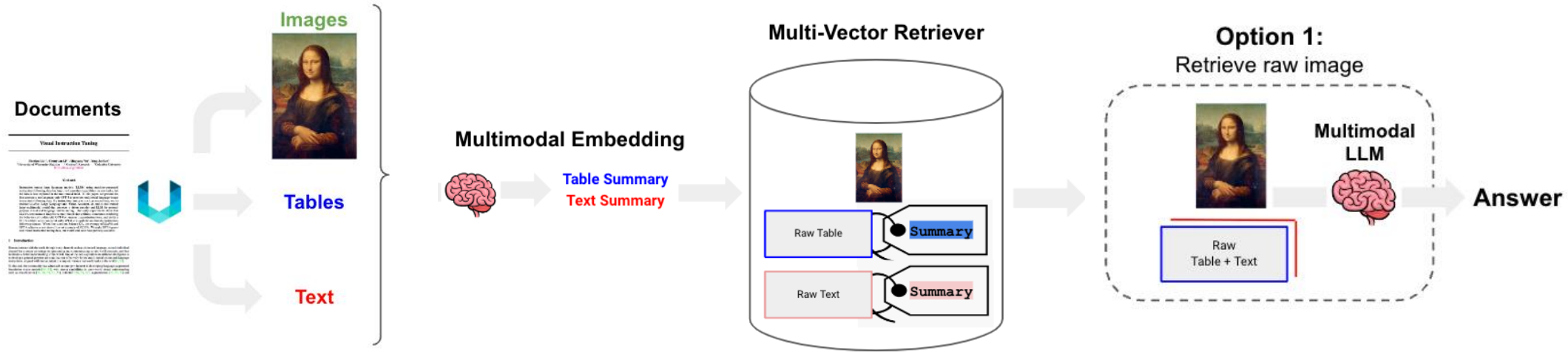
https://github.com/langchain-ai/langchain/blob/master/cookbook/Multi_modal_RAG.ipynb

Approaches: Pros and Cons

Option 1:

Multimodal Embeddings (CLIP):

- Texts and images are converted into embeddings via multimodal model (CLIP)
- Retrieval of texts and images via similarity search to the query
- Pass raw images and text chunks to a multimodal LLM for answer generation



https://github.com/langchain-ai/langchain/blob/master/cookbook/Multi_modal_RAG.ipynb

Approaches: Pros and Cons

Option 1:

Multimodal Embeddings (CLIP):

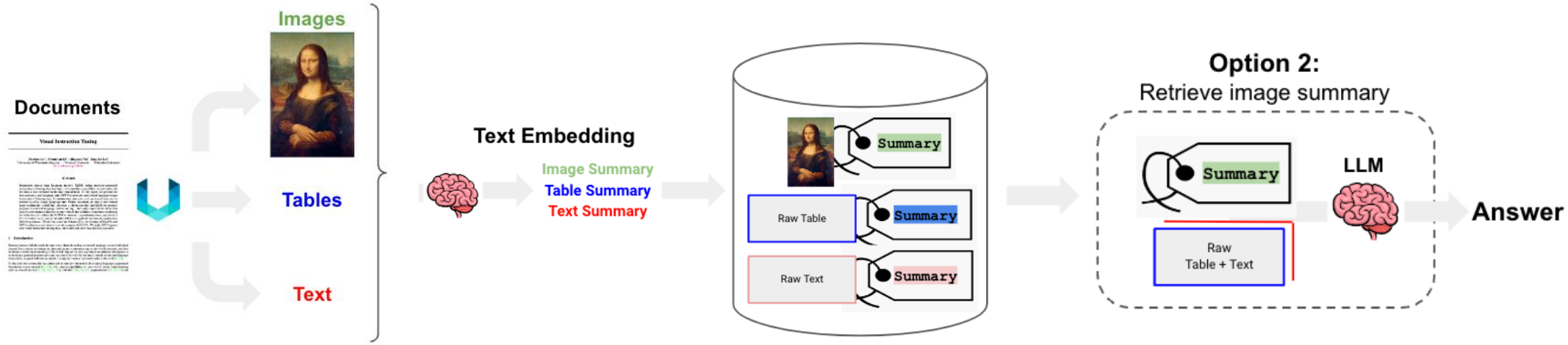
- Pros:
  - Straightforward implementation
  - Structure of the RAG pipeline very similar to text-based pipeline
  - No additional processing step for images necessary, both images and texts can be embedded directly
- Cons:
  - Less flexible than approaches that implement retrieval via text embeddings (fewer embedding models available)
  - Images of tables or charts that appear visually similar will have a similar representation in the embedding space

Approaches: Pros and Cons

Option 2:

Text Embeddings of Image Summaries

- Multimodal LLM creates text summary of an image
- Retrieval is done via summaries
- Summaries are passed to LLM to generate the response
- Multimodal component of the pipeline ends after image summaries are created



https://github.com/langchain-ai/langchain/blob/master/cookbook/Multi_modal_RAG.ipynb

Approaches: Pros and Cons

Option 2:

Text Embeddings of Image Summaries:

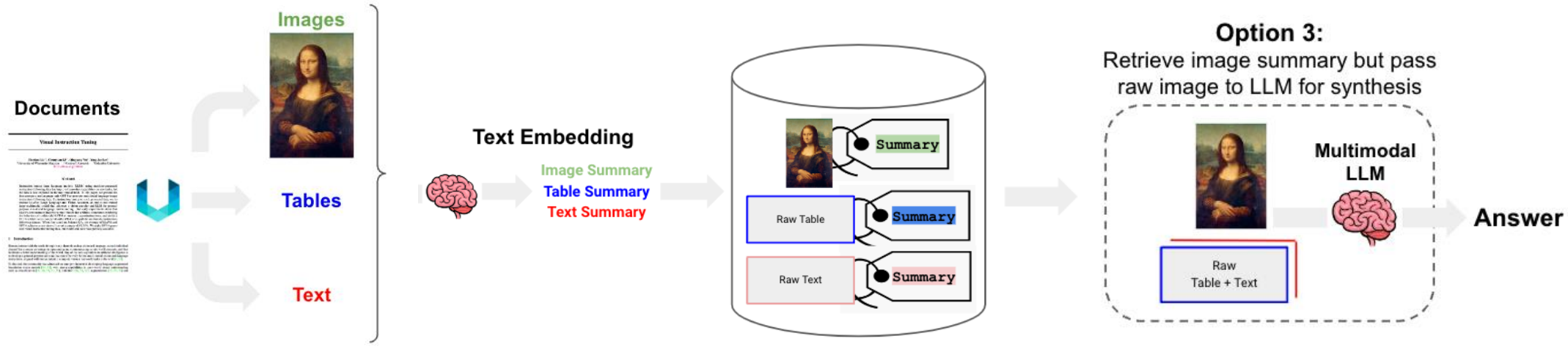
- Pros:
  - Reduced cost (multimodal model needed only once for summary generation)
  - Appropriate when a multimodal LLM cannot be used for answer synthesis
  - Pipeline almost the same as text-based RAG, except for image summary generation step
- Cons:
  - Loss of information through image summarization
  - The capability of the model to answer a question related to an image entirely relies on the quality of the image summary

Approaches: Pros and Cons

Option 3:

Text Embeddings of Image Summaries + Original Image

- Multimodal LLM creates text summary of an image
- Retrieval is done via summaries
- Original images are passed to LLM to generate the response



https://github.com/langchain-ai/langchain/blob/master/cookbook/Multi_modal_RAG.ipynb

Approaches: Pros and Cons

Option 3:

Text Embeddings of Image Summaries + Original Image

- Pros:
  - Powerful text embedding models can be used for image summaries
  - Multimodal models can already extract keywords and technical terms that may appear in the user query when generating the image summary
- Cons:
  - More complex structure
  - Additional step required for image preprocessing
  - Higher costs

Evaluation

- Metrics
- Results

Metrics

LLM-based metrics:

- LLMs are used as judges to evaluate the quality of the output of an LLM (LLM-as-a-judge)
- Set of binary metrics averaged over the dataset to obtain a score
- The model is prompted to answer only with YES or NO when evaluating a metric
- Additionally, the model is asked to provide a reason for its evaluation (Chain-of-Thought)

Models used as evaluators:

- GPT4-vision
- LLaVA

Metrics

Metrics employed:

- **Answer Correctness:**
  You are given a question, the correct reference answer, and the student's answer.
  You are asked to grade the student's answer as either correct or incorrect, based on the reference answer.
  Ignore differences in punctuation and phrasing between the student answer and true answer.
  It is OK if the student answer contains more information than the true answer, as long as it does not contain any conflicting statements.
  USER QUERY: …
  REFERENCE ANSWER: …
  STUDENT ANSWER: …
  Is the student's answer correct? (YES or NO)

Metrics

Metrics employed:

- **Answer Relevancy:**
  Is the generated answer relevant to the user query? (YES or NO)
- **Text Faithfulness:**
  Is the answer faithful to the context provided by the text, i.e. does it factually align with the context? (YES or NO)
- **Text Context Relevancy:**
  Is the context provided by the text relevant to the user query? (YES or NO)
- **Image Faithfulness:**
  Is the answer faithful to the context provided by the image, i.e. does it factually align with the context? (YES or NO)
- **Image Context Relevancy:**
  Is the context provided by the image relevant to the user query? (YES or NO)

Results

- **Evaluation Setups:**
  - Text-Only
  - Image-Only
  - Text-and-Image

Generation Models:

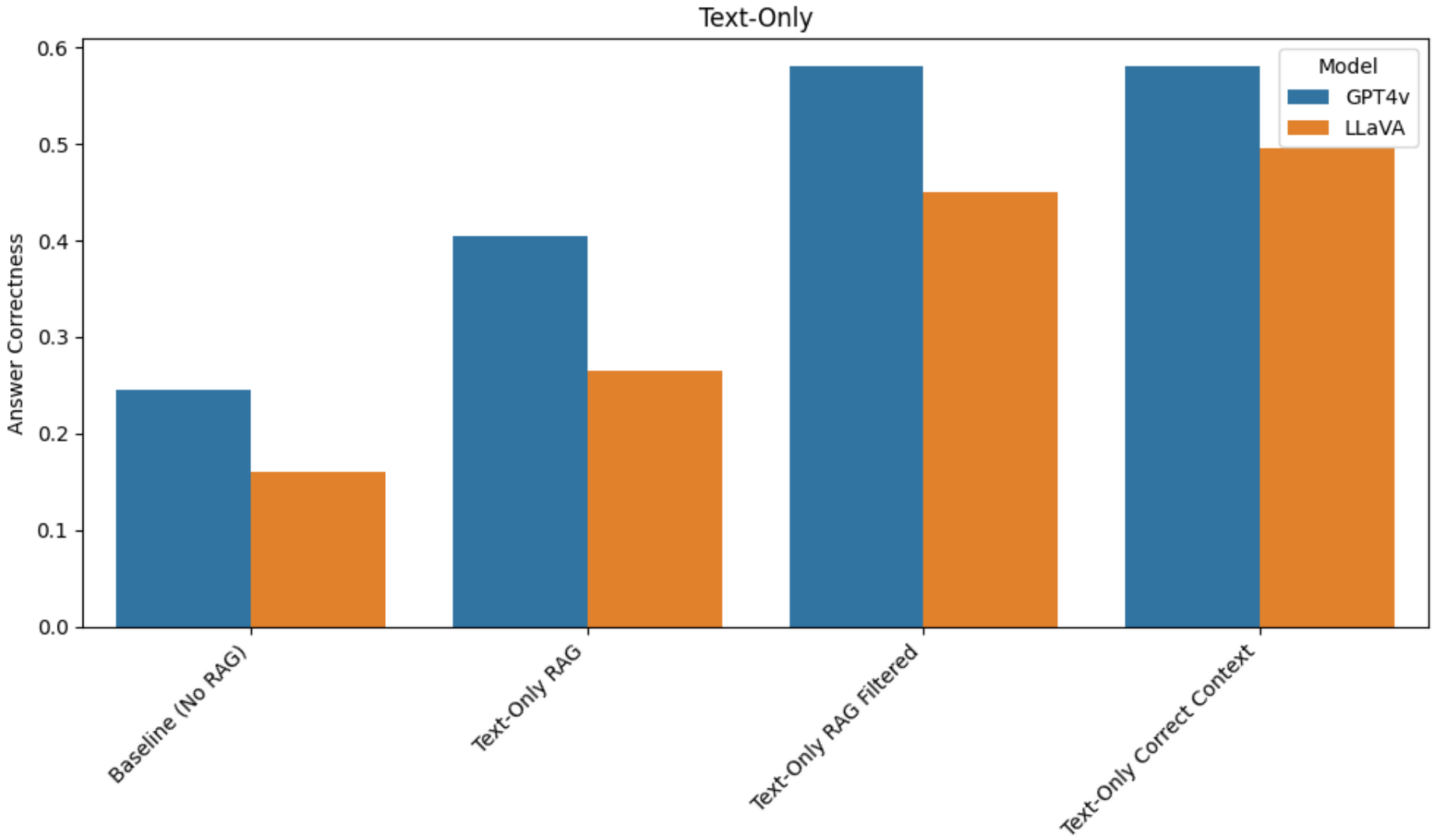
GPT4-vision

LLaVA

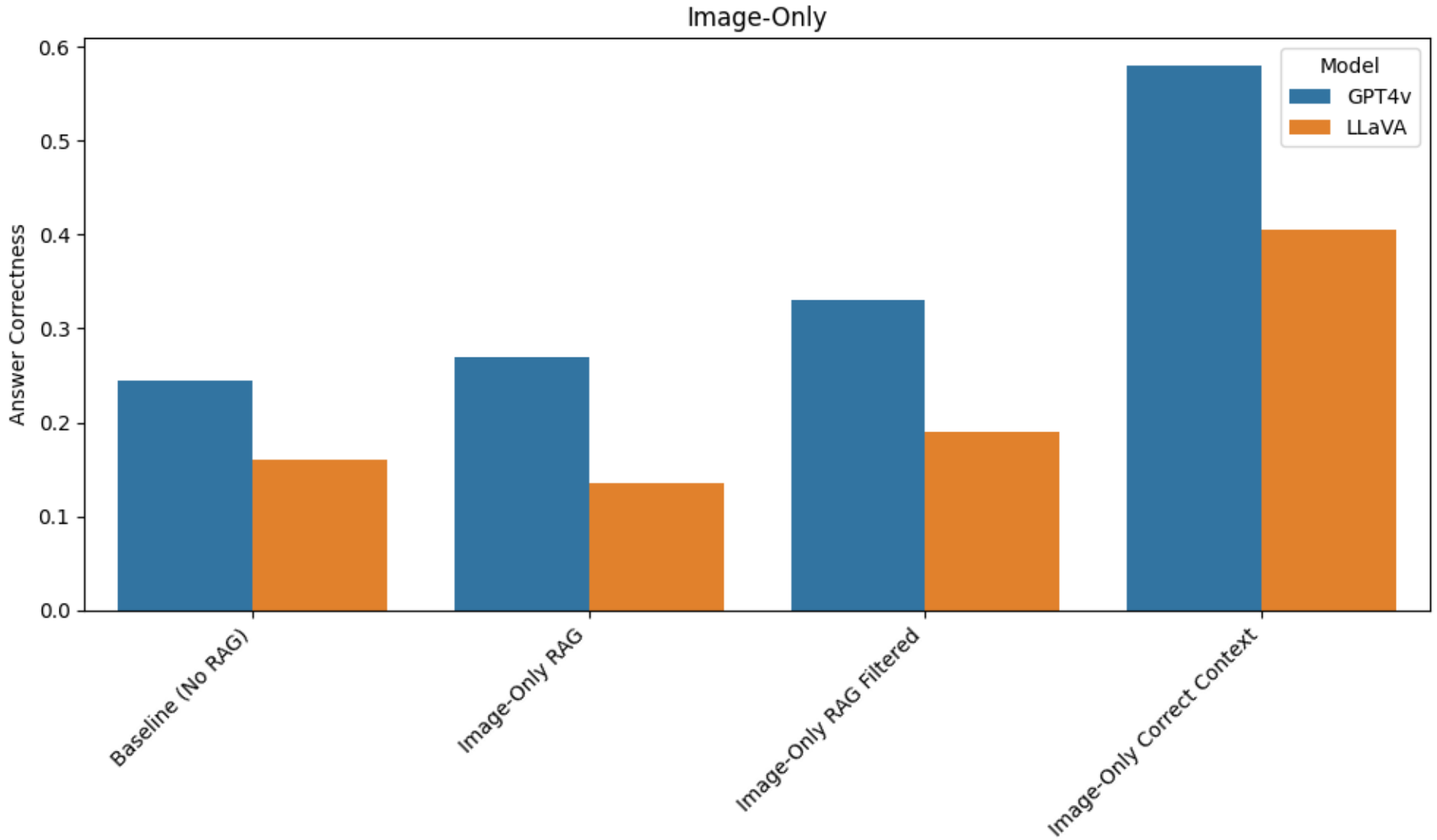
Each of these setups includes four evaluations:

- **Baseline:** Same for all setups, no RAG, just prompt the model directly
- **Standard RAG:** RAG using either both or a single modality (text-only or image-only)
- **Filtered RAG:** Prefiltering of the document collection. Retrieval is performed only on one document instead of using the entire document collection
- **Correct Context:** The retriever is not used, instead the correct texts/images that should be retrieved, are directly provided to the model to evaluate generation performance of the RAG pipeline (upper bound)

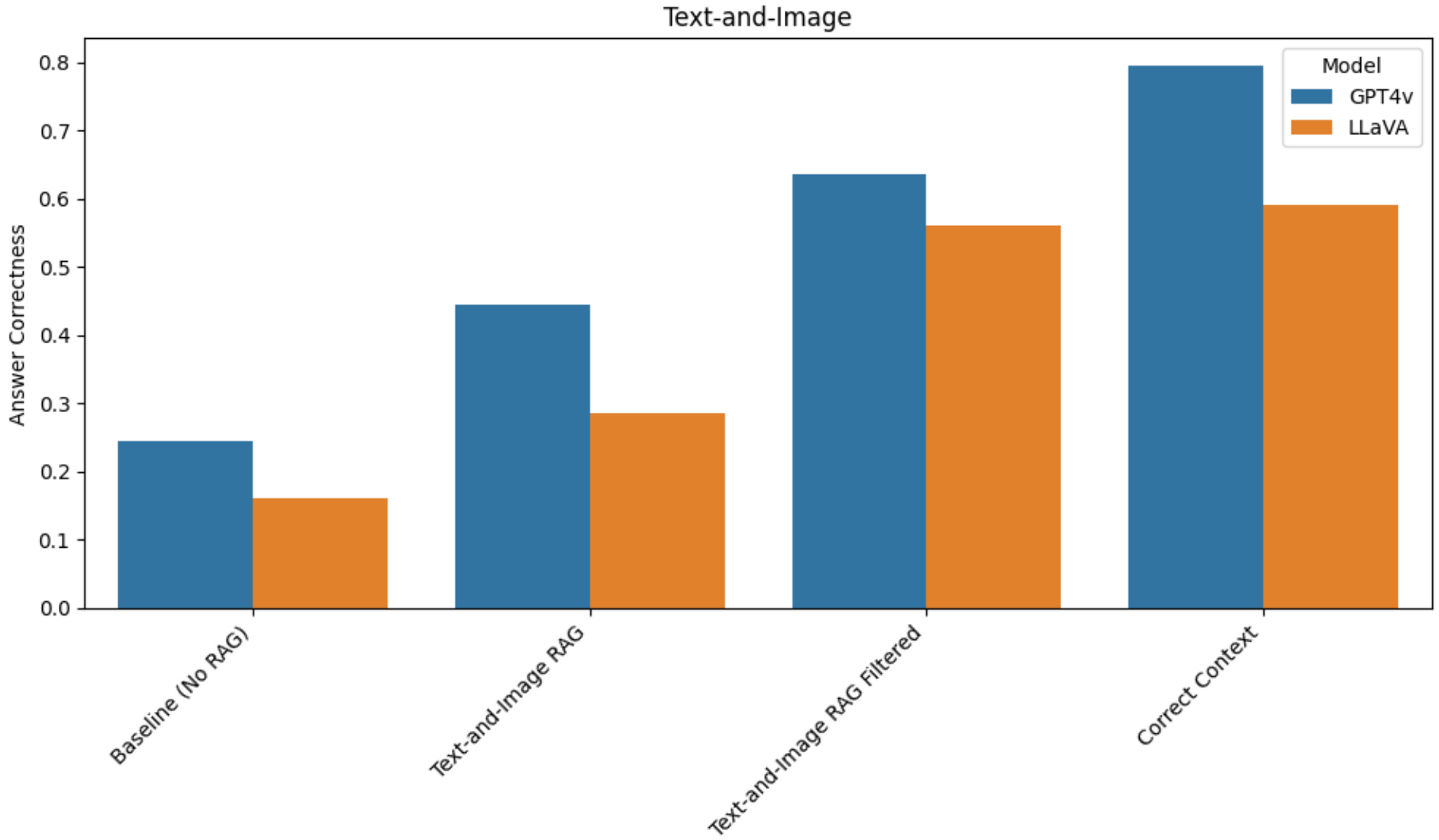
Results – Text-Only



Results – Image-Only



Results – Text-and-Image



Conclusion

Conclusion

- Text-retrieval already performs quite reasonably
- Image retrieval seems more challenging than text retrieval

- Still many ideas to improve results for image retrieval:
  1. Optimize the quality of the image summaries:
     - Prompt engineering: create better prompts to obtain better image summaries
     - Use OCR to extract relevant keywords
  2. Optimize the retrieval:
     - Use an ensemble retriever (neural-based + bm25)
     - Use a reranker on top of the retriever
     - Use advanced retrieval methods such as corrective RAG

- RAG with CLIP embeddings still to be evaluated…