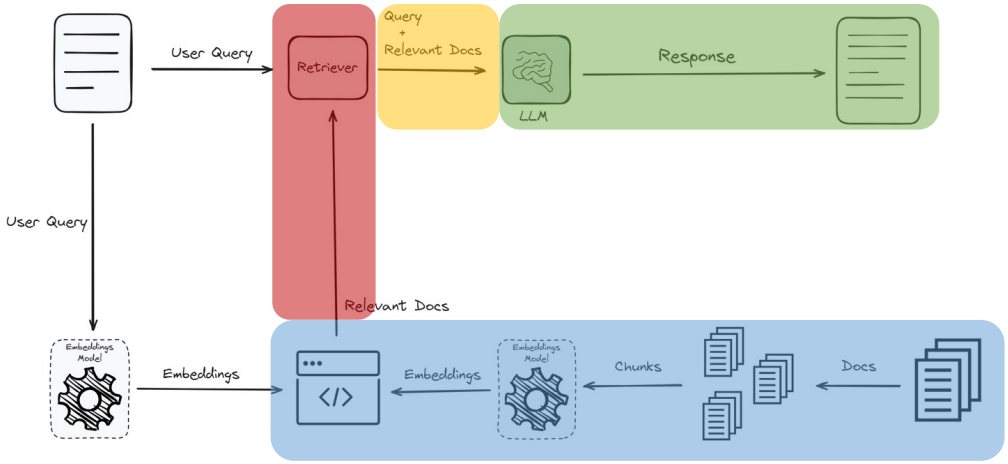
RAG system improvements & evaluation

RAG 101



- Motivation: Base LLM lacks internal knowledge
- RAG: concept to provide LLMs with additional information from an external knowledge source. This allows them to generate more accurate and contextual answers while reducing hallucinations.
- 4 steps:
  1. **Create**: documents are stored into vector database
  2. **Retrieve:** The user query is used to retrieve relevant context from an external knowledge source. For this, the user query is embedded with an embedding model into the same vector space as the additional context in the vector database. This allows to perform a similarity search, and the top k closest data objects from the vector database are returned.
  3. **Augment:** The user query and the retrieved additional context are stuffed into a prompt template.
  4. **Generate:** Finally, the retrieval-augmented prompt is fed to the LLM.

RAG Evaluation Framework

- TRIAD framework:
  - RAGAS package: https://docs.ragas.io/en/stable/
  - DeepEval package: https://docs.confident-ai.com/docs/getting-started
  - TruEval package: https://github.com/truera/trulens
- Using LLM as a judge
  - https://huggingface.co/learn/cookbook/en/rag_evaluation
- RAG system evaluation is 3-folds:
  - Chunking Evaluation
  - Retrieval Evaluation
  - Overall Response Evaluation

https://www.pinecone.io/learn/series/vector-databases-in-production-for-busy-engineers/rag-evaluation/#Determining-which-RAG-evaluation-framework-is-right-for-you

RAG evaluation framework (1/2)

Use Case	Recommended Framework	Metrics Used	Reasoning
Initial RAG evaluations	RAGAS	Average Precision (AP), Faithfulness	RAGAS is ideal for initial evaluations, especially in environments where reference data is scarce. It focuses on precision and how faithfully the response matches the provided context.
Dynamic, continuous RAG deployments	ARES	MRR, NDCG	ARES uses synthetic data and LLM judges, which are suitable for environments needing continuous updates and training and focusing on response ranking and relevance.
Full system traces including LLMs and Vector storage	TraceLoop	Information Gain, Factual Consistency, Citation Accuracy	TraceLoop is best suited for applications where tracing the flow and provenance of information used in the generated output is critical, such as academic research or journalism.
Real-time RAG monitoring	Arize	Precision, Recall, F1	Arize excels in real-time performance monitoring, making it perfect for deployments where immediate feedback on RAG performance is essential
Enterprise-level RAG applications	Galileo	Custom metrics, Context Adherence	Galileo provides advanced insights and metrics integration for complex applications, ensuring RAG's adherence to context.
Optimizing RAG for specific domains	TruLens	Domain-specific accuracy, Precision	TruLens is designed to optimize RAG systems within specific domains, by enhancing the accuracy and precision of domain-relevant responses

https://www.pinecone.io/learn/series/vector-databases-in-production-for-busy-engineers/rag-evaluation/#Determining-which-RAG-evaluation-framework-is-right-for-you

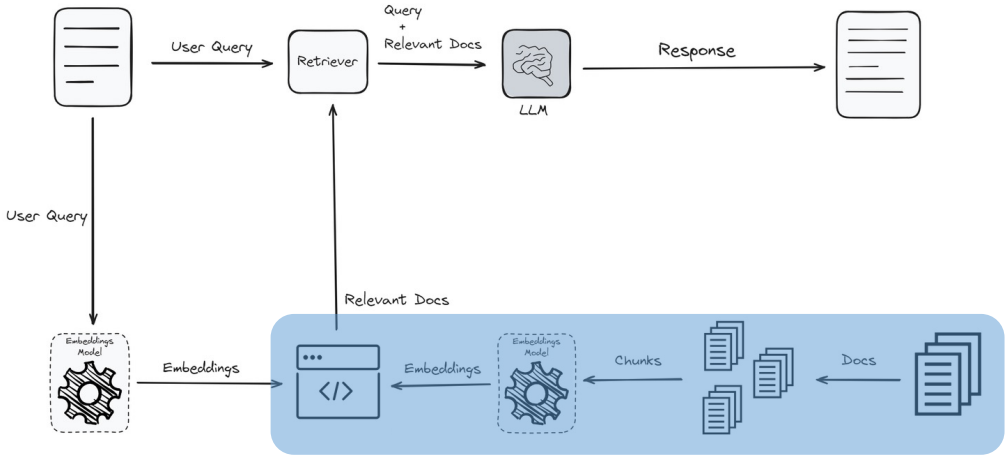
RAG evaluation framework (2/2)

- **Arize:** Arize acts as a model monitoring platform and adapts well to evaluating RAG systems by focusing on **Precision**, **Recall**, and **F1 Score**. It is beneficial in scenarios requiring ongoing performance tracking, ensuring RAG systems consistently meet accuracy thresholds in real-time applications. Arize is a proprietary paid offering providing robust support and continuous updates for enterprise deployments.
- **ARES:** ARES leverages synthetic data and LLM judges, emphasizing **Mean Reciprocal Rank (MRR)** and **Normalized Discounted Cumulative Gain (NDCG)**. It is ideal for dynamic environments where continuous training and updates are necessary to maintain system relevance and accuracy. ARES is an open-source framework that provides data sets to facilitate getting started.
- **RAGAS:** RAGAS offers streamlined, reference-free evaluation focusing on **Average Precision (AP)** and custom metrics like **Faithfulness**. It assesses how well the content generated aligns with provided contexts and is suitable for initial assessments or when reference data is scarce. RAGAS is an open-source tool, allowing for flexible adaptation and integration into diverse RAG systems without the financial overhead of licensed software.
- **TraceLoop:** TraceLoop is an open-source RAG evaluation framework that focuses on tracing the origins and flow of information throughout the retrieval and generation process.
- **TruLens:** TruLens specializes in domain-specific optimizations for RAG systems, emphasizing accuracy and precision tailored to specific fields. It offers detailed metrics to assess retrieval components' domain relevance. TruLens is a proprietary tool for enterprises seeking specialized, high-performance RAG solutions with robust customer support and regular updates to align with evolving domain-specific needs.
- **Galileo:** Galileo's RAG tool integrates advanced insights and metrics into users' workflows, focusing on enhancing the performance and transparency of RAG systems. It facilitates easy access to evaluation metrics and simplifies the management of large-scale RAG deployments. Galileo offers its solutions as a proprietary service, which is ideal for businesses seeking comprehensive, scalable AI tools emphasizing usability and commercial application integration.

Methodology

1. Create test dataset
2. Choose model parameters:
    1. chunking methods
    2. Embedding model
3. Choose metrics
4. Evaluate metrics

Initial step: internal documents embeddings

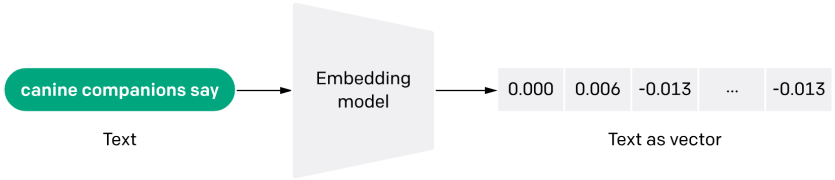


1. **Chunking Method** (from naïve to more advanced)
    1. Fixed Sized Chunking
        1. CharacterTextSplitting
        2. SentenceTextSplitting
    2. Recursive Chunking
    3. Document Based Chunking
    4. Token Based Chunking
    5. Semantic Chunking
    6. Agentic Chunking
2. **Embedding Model**

Chunking methods

1. **Fixed Length**: this is the most crude and simplest method of segmenting the text. It breaks down the text into chunks of a specified number of characters, regardless of their content or structure.
2. **Recursive Chunking**: advanced version that divides the text into chunks until a certain condition is met, such as reaching a minimum chunk size. This method ensures that the chunking process aligns with the text's structure, preserving more meaning. Its adaptability makes Recursive Character Chunking great for texts with varied structures.
3. **Document splitting**: strategy that respects the document's structure. Rather than using a set number of characters or a recursive process, it creates chunks that align with the logical sections of the document, like paragraphs or subsections. This approach maintains the original author's organization of content and helps keep the text coherent. It makes the retrieved information more relevant and useful, particularly for structured documents with clearly defined sections.
4. **Tokens base**: Language models used in the rest of your possible RAG pipeline have a token limit, which should not be exceeded.
5. **Semantic based**: method aims to extract semantic meaning from embeddings and then assess the semantic relationship between these chunks. The core idea is to keep together chunks that are semantic similar.
6. **Agentic Chunking**: uses Large Language Models to dynamically optimize text chunking based on context.

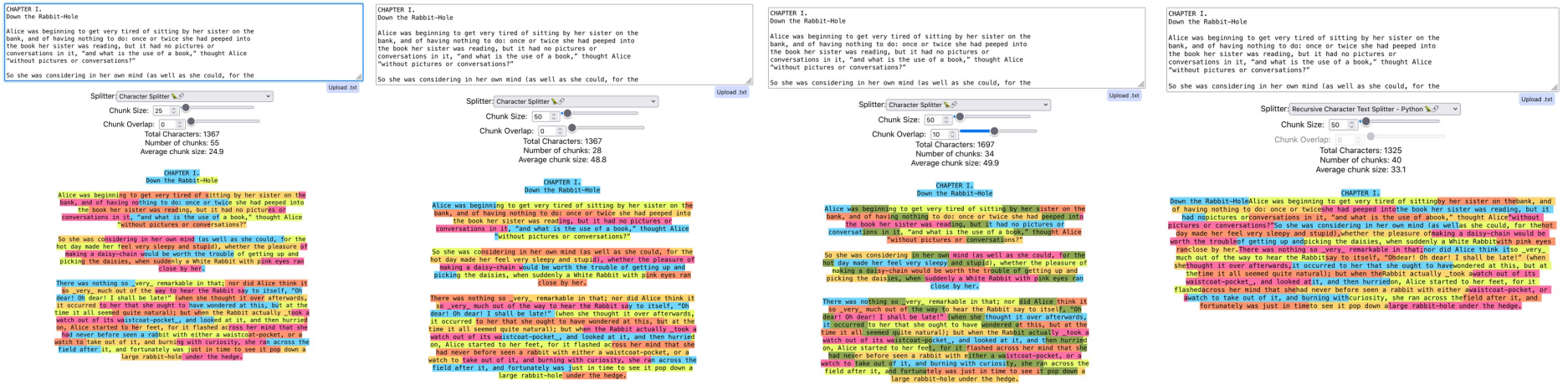
Embedding model



- Embedding = string encoding
- Motivation: allow the efficient retrieval of relevant information.
- When selecting an embedding model, consider the vector dimension, average retrieval performance, and model size.
- Other consideration:
  - Cost
  - Search Latency
  - Language Support

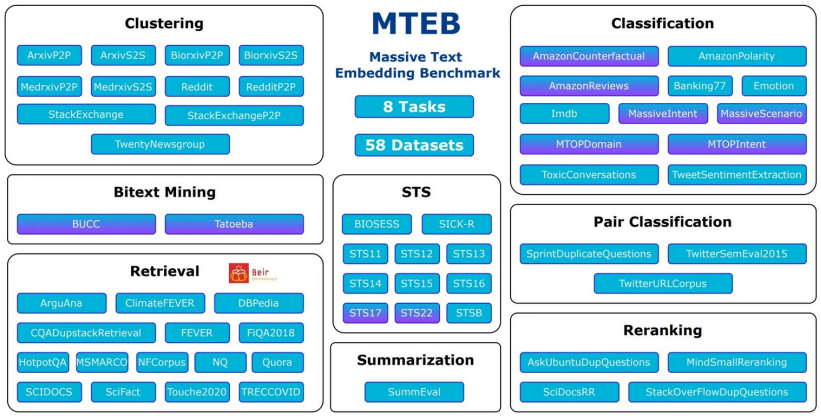
1. **Retrieval Average**: Represents average Normalized Discounted Cumulative Gain (NDCG) @ 10 across several datasets. NDCG is a common metric to measure the performance of retrieval systems. A higher NDCG indicates a model that is better at ranking relevant items higher in the list of retrieved results.
2. **Model Size**: Size of the model (in GB). It gives an idea of the computational resources required to run the model. While retrieval performance scales with model size, it is important to note that model size also has a direct impact on latency. The latency-performance trade-off becomes especially important in a production setup.
3. **Max Tokens**: Number of tokens that can be compressed into a single embedding. You typically don't want to put more than a single paragraph of text (~100 tokens) into a single embedding. So even models with max tokens of 512 should be more than enough.
4. **Embedding Dimensions**: Length of the embedding vector. Smaller embeddings offer faster inference and are more storage-efficient, while more dimensions can capture nuanced details and relationships in the data. Ultimately, we want a good trade-off between capturing the complexity of data and operational efficiency.

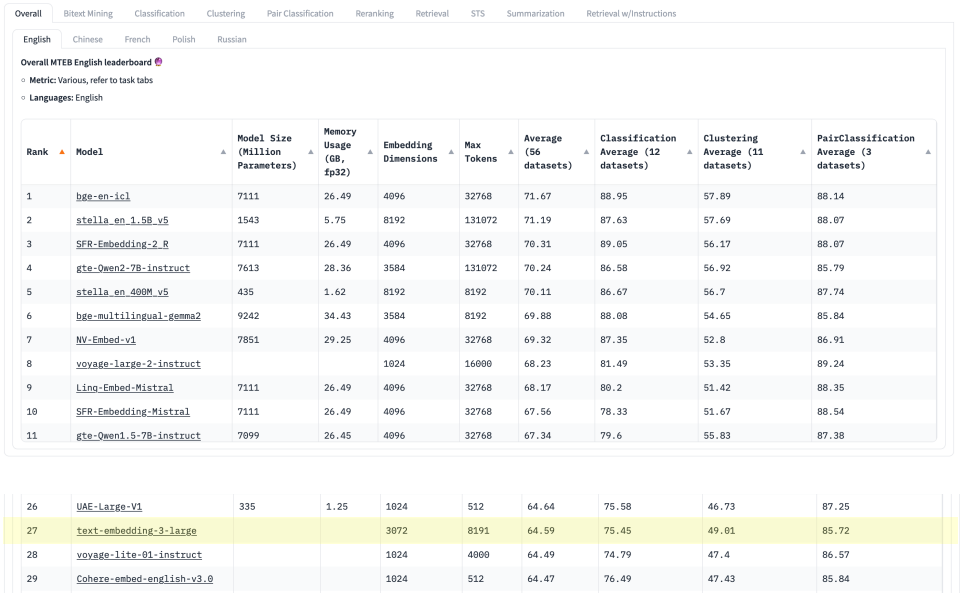
Examples



https://chunkviz.up.railway.app/

Embedding models Performances





Overall | Bitext Mining | Classification | Clustering | Pair Classification | Reranking | Retrieval | STS | Summarization | Retrieval w/Instructions

English | Chinese | French | Polish | Russian

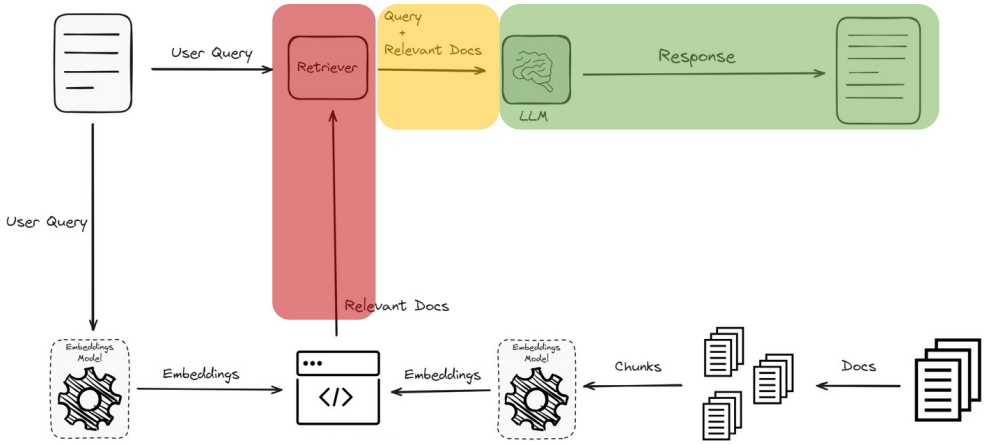
Overall MTEB English leaderboard

- Metric: Various, refer to task tabs
- Languages: English

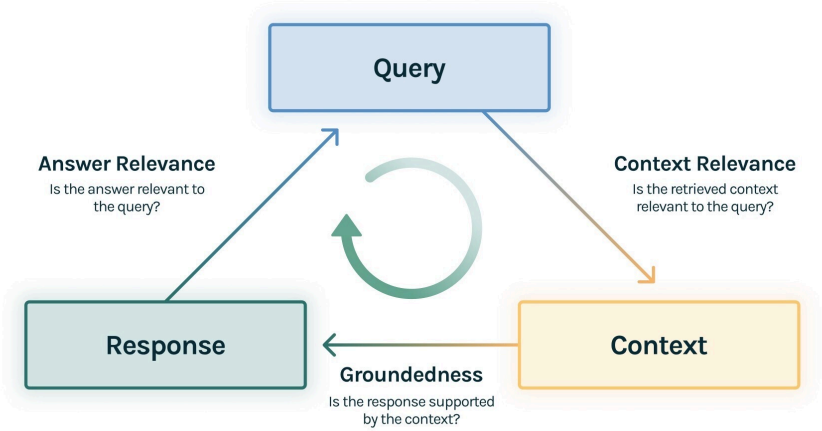
Rank	Model	Model Size (Million Parameters)	Memory Usage (GB, fp32)	Embedding Dimensions	Max Tokens	Average (56 datasets)	Classification Average (12 datasets)	Clustering Average (11 datasets)	PairClassification Average (3 datasets)
1	bge-en-icl	7111	26.49	4096	32768	71.67	88.95	57.89	88.14
2	stella_en_1.5B_v5	1543	5.75	8192	131072	71.19	87.63	57.69	88.07
3	SFR-Embedding-2_R	7111	26.49	4096	32768	70.31	89.05	56.17	88.07
4	gte-Qwen2-7B-instruct	7613	28.36	3584	131072	70.24	86.58	56.92	85.79
5	stella_en_400M_v5	435	1.62	8192	8192	70.11	86.67	56.7	87.74
6	bge-multilingual-gemma2	9242	34.43	3584	8192	69.88	88.08	54.65	85.84
7	NV-Embed-v1	7851	29.25	4096	32768	69.32	87.35	52.8	86.91
8	voyage-large-2-instruct			1024	16000	68.23	81.49	53.35	89.24
9	Linq-Embed-Mistral	7111	26.49	4096	32768	68.17	80.2	51.42	88.35
10	SFR-Embedding-Mistral	7111	26.49	4096	32768	67.56	78.33	51.67	88.54
11	gte-Qwen1.5-7B-instruct	7099	26.45	4096	32768	67.34	79.6	55.83	87.38
26	UAE-Large-V1	335	1.25	1024	512	64.64	75.58	46.73	87.25
27	text-embedding-3-large			3072	8191	64.59	75.45	49.01	85.72
28	voyage-lite-01-instruct			1024	4000	64.49	74.79	47.4	86.57
29	Cohere-embed-english-v3.0			1024	512	64.47	76.49	47.43	85.84

Source: HugginFace
https://huggingface.co/spaces/mteb/leaderboard

Retrieval and Overall Response evaluation



Metrics: TRIAD framework



https://www.trulens.org/trulens_eval/getting_started/core_concepts/rag_triad/

- **Context Relevance**: This component evaluates the **retrieval part** of the RAG system. It evaluates how accurately the documents were retrieved from the large dataset. Metrics like precision, recall, MRR, and MAP are used here.
- **Faithfulness (Groundedness)**: This component falls under the **Response evaluation**. It checks if the generated response is factually accurate and grounded in the retrieved documents. Methods such as human evaluation, automated fact-checking tools, and consistency checks are used to assess faithfulness.
- **Answer Relevance**: This is also part of the **Response Evaluation**. It measures how well the generated response addresses the user's query and provides useful information. Metrics like BLEU, ROUGE, METEOR, and embedding-based evaluations are used.

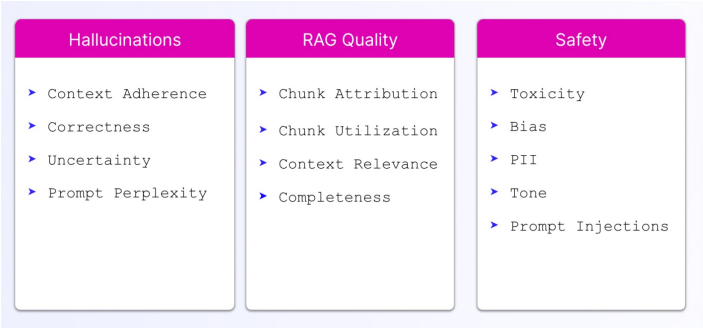
Metrics for RAG

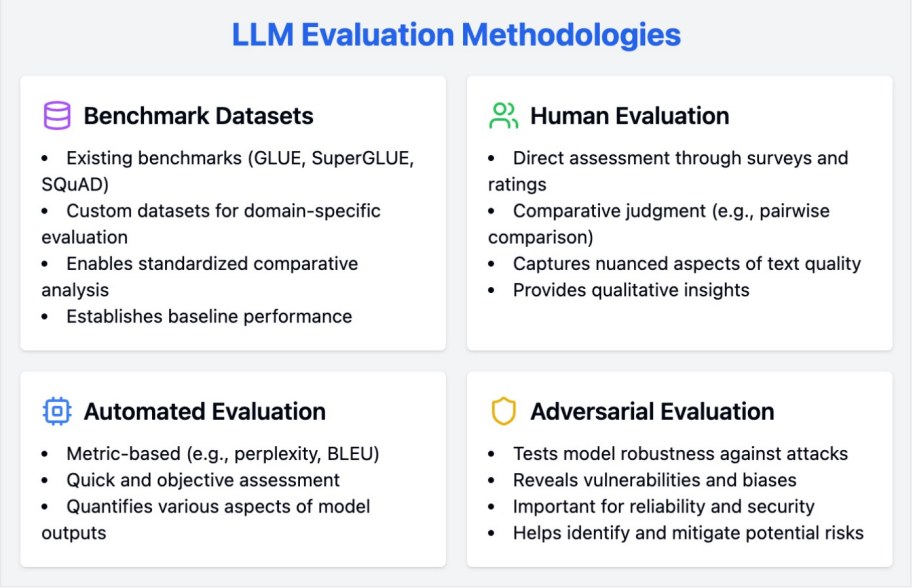
Component	metric	Metric definition
RETRIEVAL	**Context precision**	Evaluates the ability of the retriever to rank retrieved items in order of relevance to the ground truth answer
	Context recall	Measures the extent to which the retrieved context aligns with the ground truth answer
GENERATION	**Faithfulness**	Measures the factual consistency of the generated answer against the retrieved context
	Answer relevance	Measures how relevant the generated answer is to the given prompt (question + retrieved context)
OVERALL	**Answer semantic similarity**	Measures the semantic similarity between the generated answer and the ground truth
	Answer correctness	Measures the accuracy of the generated answer compared to the ground truth

LLM evaluations

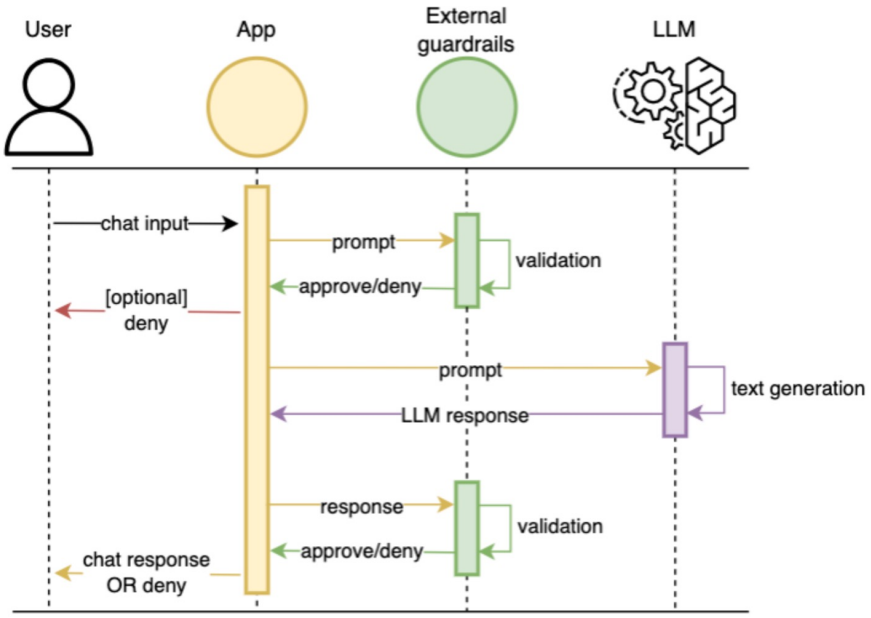
LLM Evaluations

- RAG vs. Fine Tuning
- Guardrails
- Prompt injection
- Pii removal when sending to LLM
- RAG: multimodal
- Opensource vs. proprietary
- Regulatory compliance

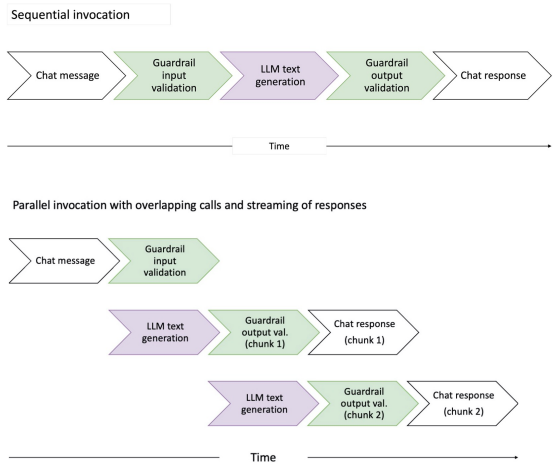




LLM Evaluations: guardrails



- Both input and output need to be filtered by Guardrails
- To improve latency, evaluation of individual output chunks can be performed



Vector Database Comparison (2023)

A comparison of leading vector databases

	Pinecone	Weaviate	Milvus	Qdrant	Chroma	Elasticsearch	PGvector
Is open source	❌	✅	✅	✅	✅	❌	✅
Self-host	❌	✅	✅	✅	✅	✅	✅
Cloud management	✅	✅	✅	✅	❌	✅	(✓)
Purpose-built for Vectors	✅	✅	✅	✅	✅	❌	❌
Developer experience	👍👍👍	👍👍	👍👍	👍👍	👍👍	👍	👍
Community	Community page & events	8k☆ github, 4k slack	23k☆ github, 4k slack	13k☆ github, 3k discord	9k☆ github, 6k discord	23k slack	6k☆ github
Queries per second (using text nytimes-256-angular)	150 *for p2, but more pods can be added	791	2406	326	?	700-100 *from various reports	141
Latency, ms (Recall/Percentile 95 (millis), nytimes-256-angular)	1 *batched search, 0.99 recall, 200k SBERT	2	1	4	?	?	8
Supported index types	?	HNSW	Multiple (11 total)	HNSW	HNSW	HNSW	HNSW/IVFFlat
Hybrid Search (i.e. scalar filtering)	✅	✅	✅	✅	✅	✅	✅
Disk index support	✅	✅	✅	✅	✅	❌	✅
Role-based access control	✅	❌	✅	❌	❌	✅	❌
Dynamic segment placement vs. static data sharding	?	Static sharding	Dynamic segment placement	Static sharding	Dynamic segment placement	Static sharding	-
Free hosted tier	✅	✅	✅	(free self-hosted)	(free self-hosted)	(free self-hosted)	(varies)
Pricing (50k vectors @1536)	$70	fr. $25	fr. $65	est. $9	Varies	$95	Varies
Pricing (20M vectors, 20M req. @768)	$227 ($2074 for high performance)	$1536	fr. $309 ($2291 for high performance)	fr. $281 ($820 for high performance)	Varies	est. $1225	Varies

Vector Database Comparison (2023)

- **Is open source**: Indicates if the software's source code is freely available to the public, allowing developers to review, modify, and distribute the software.
- **Self-host**: Specifies if the database can be hosted on a user's own infrastructure rather than being dependent on a third-party cloud service.
- **Cloud management**: Offers an interface for database cloud management
- **Purpose-built for Vectors**: This means the database was specifically designed with vector storage and retrieval in mind, rather than being a general database with added vector capabilities.
- **Developer experience**: Evaluates how user-friendly and intuitive it is for developers to work with the database, considering aspects like documentation, SDKs, and API design.
- **Community**: Assesses the size and activity of the developer community around the database. A strong community often indicates good support, contributions, and the potential for continued development.
- **Queries per second**: How many queries the database can handle per second using a specific dataset for benchmarking (in this case, the nytimes-256-angular dataset)
- **Latency**: the delay (in milliseconds) between initiating a request and receiving a response. 95% of query latencies fall under the specified time for the nytimes-256-angular dataset.
- **Supported index types**: Refers to the various indexing techniques the database supports, which can influence search speed and accuracy. Some vector databases may support multiple indexing types like HNSW, IVF, and more.
- **Hybrid Search**: Determines if the database allows for combining traditional (scalar) queries with vector queries. This can be crucial for applications that need to filter results based on non-vector criteria.
- **Disk index support**: Indicates if the database supports storing indexes on disk. This is essential for handling large datasets that cannot fit into memory.
- **Role-based access control**: Checks if the database has security mechanisms that allow permissions to be granted to specific roles or users, enhancing data security.
- **Dynamic segment placement vs. static data sharding**: Refers to how the database manages data distribution and scaling. Dynamic segment placement allows for more flexible data distribution based on real-time needs, while static data sharding divides data into predetermined segments.
- **Free hosted tier**: Specifies if the database provider offers a free cloud-hosted version, allowing users to test or use the database without initial investment.
- **Pricing (50k vectors @1536)** and **Pricing (20M vectors, 20M req. @768)**: Provides information on the cost associated with storing and querying specific amounts of data, giving an insight into the database's cost-effectiveness for both small and large-scale use cases.