Ways to Assess RAG

Your **Retrieval-Augmented Generation (RAG) pipeline** can be assessed using three broad categories:

1. **Traditional NLP Metrics** (Text Overlap)
2. **LLM-Based Metrics** (Coarse-Grained & Rubric-Based)
3. **RAG-Specific Metrics** (Retrieval & Generation Quality)

Each category helps in understanding different aspects of RAG performance.

1️⃣ Traditional NLP Metrics (Text Overlap)

These metrics compare the **generated answer** with the **reference answer** using token overlap.

✓ ROUGE Score

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/traditional/#rouge-scorete

- **What it does:** Measures n-gram overlap between the **generated response** and the **reference answer**.
- **Why it matters:** Higher ROUGE scores mean the generated answer has similar words as the expected response.
- **Limitations:** Doesn't measure **semantic correctness** or **factually grounded responses**.

✅ **How you used it:**

```
from ragas.metrics import RougeScore

scorer = RougeScore()
score = await scorer.single_turn_ascore(sample)
```

2️⃣ LLM-Based Metrics

LLM-based evaluations provide a more **context-aware assessment** of the response.

✓ Simple Criteria Score

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/general_purpose/#simple-criteria-scoring

- **What it does:** Provides a **0-10 rating** on how similar the response is to the reference.
- **Why it matters:** Quick way to get a **coarse-grained** assessment.
- **Limitations:** Doesn't explain **why** the response was rated a certain way.

✅ **How you used it:**

```
simple_scorer = SimpleCriteriaScore(
    name="course_grained_score",
    definition="Score 0 to 10 by similarity",
    llm=evaluator_llm
)
score = await simple_scorer.single_turn_ascore(sample)
```

✓ Rubrics Score

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/general_purpose/#rubrics-based-criteria-scoring

- **What it does:** Evaluates responses based on predefined **grading rubrics**.
- **Why it matters:** Helps measure **answer quality** using structured guidelines.
- **Limitations:** Requires **well-defined rubrics**.

✅ **How you used it:**

```
rubrics = {
    "score1_description": "Completely incorrect response",
    "score5_description": "Perfectly accurate and complete response",
}

rubrics_scorer = RubricsScore(rubrics=rubrics, llm=evaluator_llm)
score = await rubrics_scorer.single_turn_ascore(sample)
```

3️⃣ RAG-Specific Metrics

These metrics **directly evaluate** both retrieval and generation.

✓ Factual Correctness

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/factual_correctness/

- **What it does:** Measures how **factually accurate** the response is **compared to the reference answer**.
- **Why it matters:** Ensures the model is **not hallucinating**.
- **Limitations:** Doesn't check if the facts came from the **retrieved context**.

✅ **How you used it:**

```
from ragas.metrics._factual_correctness import FactualCorrectness

scorer = FactualCorrectness(llm=evaluator_llm)
score = await scorer.single_turn_ascore(sample)
```

✓ Semantic Similarity

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/semantic_similarity/

- **What it does:** Measures **how semantically similar** the generated response is to the reference.
- **Why it matters:** Ensures that answers are **contextually accurate**, even if they use different wording.
- **Limitations:** Doesn't consider factual correctness.

✅ **How you used it:**

```
from ragas.metrics import SemanticSimilarity

scorer = SemanticSimilarity(embeddings=evaluator_embeddings)
score = await scorer.single_turn_ascore(sample)
```

✓ Context Precision (LLMContextPrecisionWithReference)

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_precision/

- **What it does:** Measures **how much of the retrieved chunks were actually relevant** to the answer.
- **Why it matters:** Ensures **retrieved contexts are useful**.
- **Limitations:** Doesn't check if all relevant documents were retrieved.

✅ **How you used it:**

```
from ragas.metrics import LLMContextPrecisionWithReference

scorer = LLMContextPrecisionWithReference(llm=evaluator_llm)
score = await scorer.single_turn_ascore(sample)
```

✓ Context Recall (LLMContextRecall)

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/context_recall/

- **What it does:** Measures **if all necessary information** was present in the retrieved context.
- **Why it matters:** Ensures **important evidence is not missing**.
- **Limitations:** Doesn't verify **if the answer is actually correct**.

✅ **How you used it:**

```
from ragas.metrics import LLMContextRecall

scorer = LLMContextRecall(llm=evaluator_llm)
score = await scorer.single_turn_ascore(sample)
```

✓ Response Relevancy

https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/answer_relevance/

- **What it does:** Measures how **relevant the generated response** is to the original user input.
- **Why it matters:** Ensures responses are **on-topic** and not **irrelevant**.
- **Limitations:** Doesn't check for **factual correctness**.

✅ **How you used it:**

```
from ragas.metrics import ResponseRelevancy

scorer = ResponseRelevancy(llm=evaluator_llm, embeddings=evaluator_embeddings)
score = await scorer.single_turn_ascore(sample)
```

Summary: How to Assess RAG?

Category	Metric	What It Measures	Key Strength	Limitation
Traditional NLP	ROUGE Score	Token overlap	Fast & simple	Ignores meaning
LLM-Based	Simple Criteria Score	0-10 similarity score	Coarse but effective	No detailed feedback
	Rubrics Score	Accuracy based on predefined grading scale	Structured evaluation	Needs well-designed rubrics
RAG-Specific	Factual Correctness	Checks factual consistency	Avoids hallucinations	Doesn't check source
	Semantic Similarity	Measures meaning similarity	Catches paraphrased responses	Ignores facts
	Context Precision	Measures retrieval relevance	Ensures useful chunks	Doesn't check completeness
	Context Recall	Measures if retrieval is complete	Ensures key facts are present	Doesn't check answer correctness
	Response Relevancy	Measures alignment with query	Prevents off-topic responses	Doesn't check facts

Which Metrics Should You Prioritize?

- **If Retrieval is the Focus → Context Precision + Context Recall**
- **If Answer Correctness is Critical → Factual Correctness + Response Relevancy**
- **For Overall Performance → Rubrics Score + Semantic Similarity**