
AWS
re:Invent
DECEMBER 2 – 6, 2024 | LAS VEGAS, NV

AIM359

Streamline RAG and model evaluation on Amazon Bedrock

Dr. Jesse Manders
(he/him)
Senior Product Manager – Technical
Amazon Web Services

Hardik Vasa
(he/him)
Senior Solutions Architect
Amazon Web Services

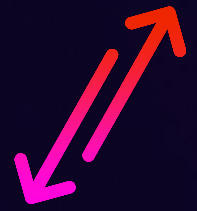
Shalendra Chhabra
(he/him)
Senior Manager, Product Management
Amazon Web Services

Agenda

- What is evaluation, why does it matter, and what are the challenges?
- New: LLM-as-a-judge for Amazon Bedrock model evaluation
- Demo
- What is RAG and how do you evaluate it?
- New: RAG evaluation with Amazon Bedrock Knowledge Bases
- Demo
- Wrap-up

What is evaluation?

QUALITY



TRADEOFF

COST

LATENCY

Why is evaluation important?

Make quality, cost, and latency tradeoffs

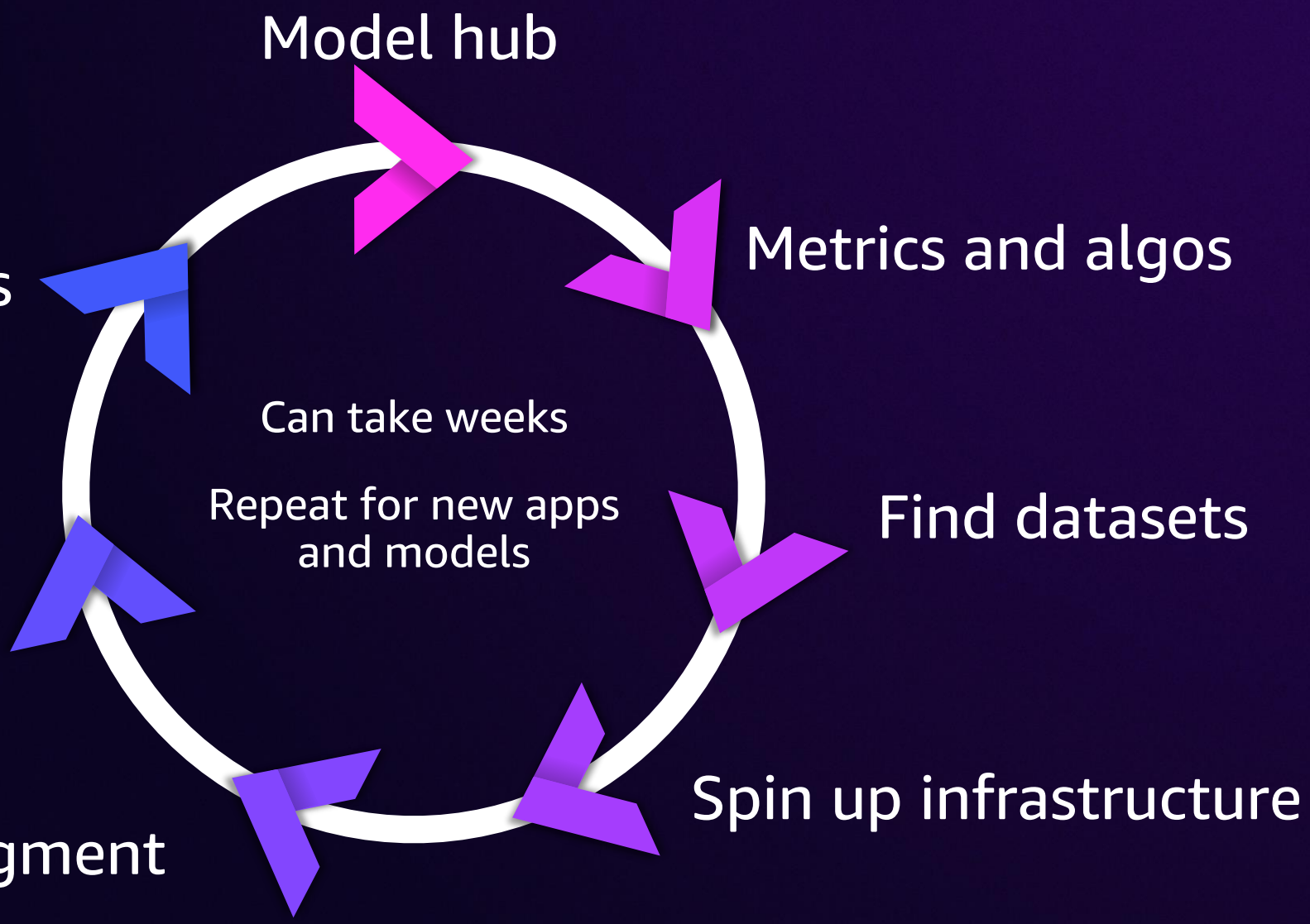
Align to your company’s style and brand voice

Evaluate for your specific use cases

Evaluate with your company’s data

Monitor biases, safety, and trust

Evaluation lifecycle and challenges



Amazon Bedrock model evaluation

Amazon Bedrock model evaluation

Evaluate, compare, and select the best foundation model for your use case

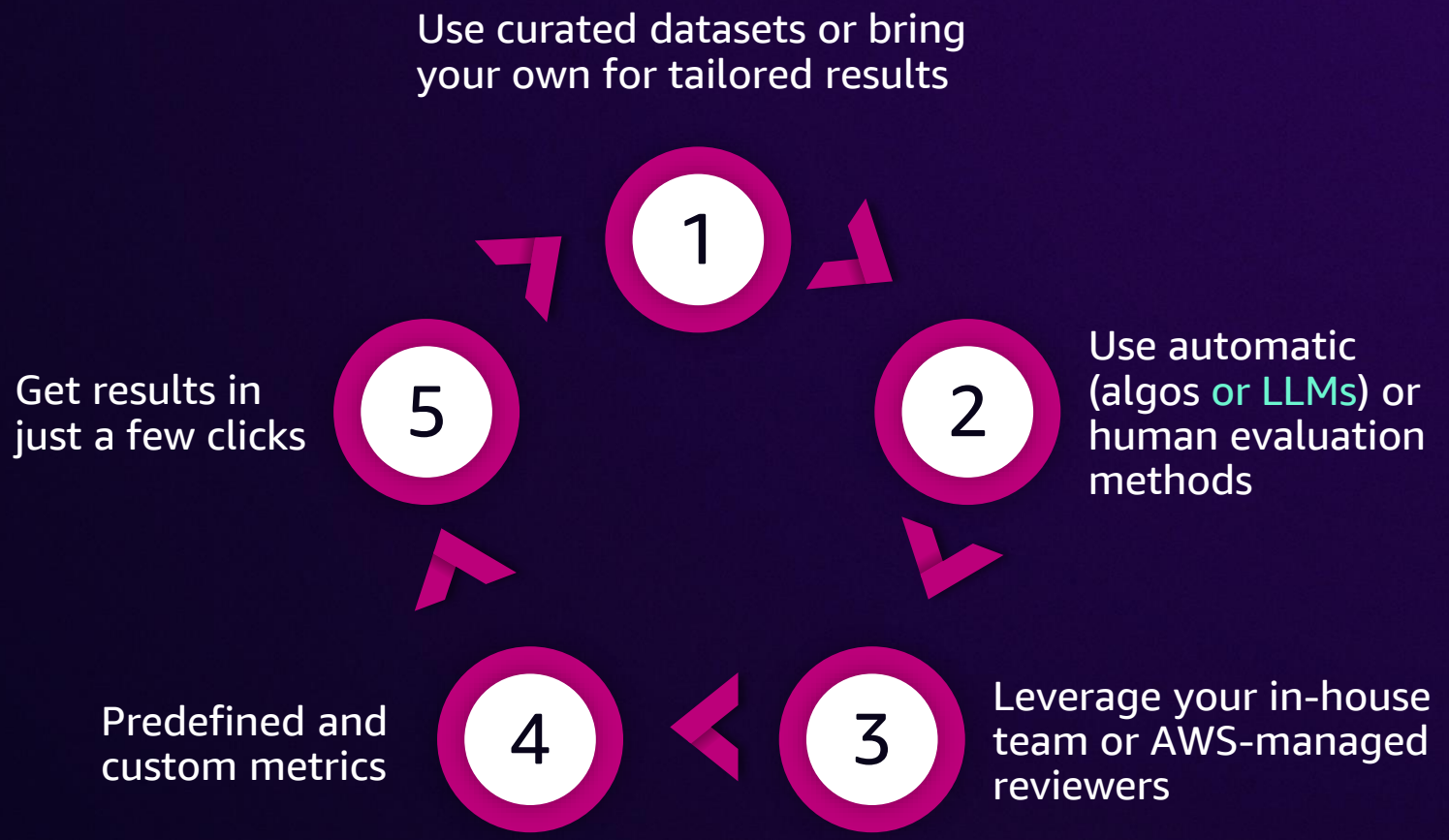
New:
Public API
Evaluate custom models
Evaluate distilled models
Evaluate imported models
Evaluate prompt routers
Use an LLM-as-a-judge (Preview)

1. Use curated datasets or bring your own for tailored results
2. Use automatic (algos or LLMs) or human evaluation methods
3. Leverage your in-house team or AWS-managed reviewers
4. Predefined and custom metrics
5. Get results in just a few clicks

Amazon Bedrock model evaluation

Evaluate, compare, and select the best foundation model for your use case

New:
Public API
Evaluate custom models
Evaluate distilled models
Evaluate imported models
Evaluate prompt routers
Use an LLM-as-a-judge (Preview)



Input data format

Input dataset contains 3 things in JSONL format

1. Prompt
2. Optional prompt category
3. Optional golden ground truth

```
{"prompt":“String”,"category":“String”,"referenceResponse":“String”}
```

Choice of evaluation methods

New

Programmatic evaluation

Accuracy | Robustness | Toxicity

LLM-as-a-judge

Correctness | Completeness | Helpfulness

Relevance | Coherence | Readability

Human evaluation

Creativity | Style | Tone

Accuracy | Consistency | Brand voice

Algorithms

BERTScore | Classification accuracy
F1 | Real-world knowledge score

LLM reasoning

Multistep reasoning | Correlation with expert human evaluators

Rating methods

Thumbs up/down | 5-point Likert scales
Binary choice buttons | Ordinal ranking

NOW IN PREVIEW

LLM-as-a-judge

LLM-as-a-judge metrics

01 Correctness

02 Completeness

03 Faithfulness

04 Helpfulness

05 Coherence

06 Relevance

07 Following instructions

08 Professional style and tone

09 Readability

10 Harmfulness

11 Stereotyping

12 Answer refusal

Ground truth response is optional

How correctness works

Example input

prompt: What is the capital of Spain?

referenceResponse: Madrid

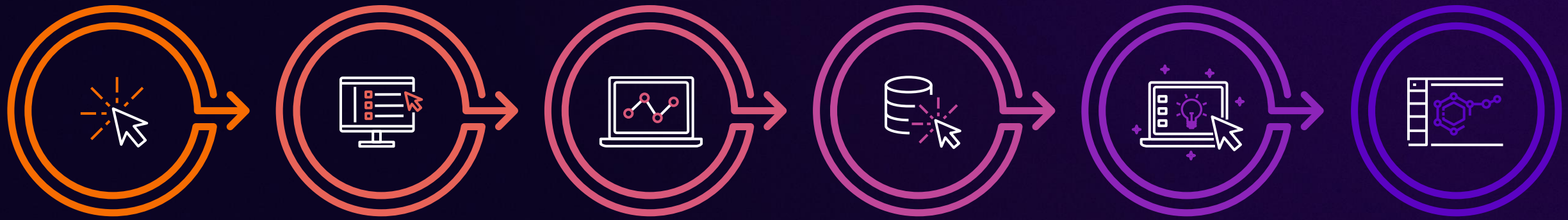
Model response: Barcelona

Judge prompt (simplified)

```
You are a helpful assistant…

You are given a question, a candidate
response from an LLM, and reference
response.

Your task is to check if the candidate
response is correct compared to the
reference response…

Here is the actual task:
Question:  {prompt}
Reference Response: {referenceResponse}
Candidate Response: {Model response}

Explain your response, followed by your
evaluation:
2) Correct
1) Partially correct
0) Incorrect
```


Note: Correctness can be with or without ground truth

How to set up LLM-as-a-judge evaluations



Choose generator model

Choose evaluator model

Choose metrics

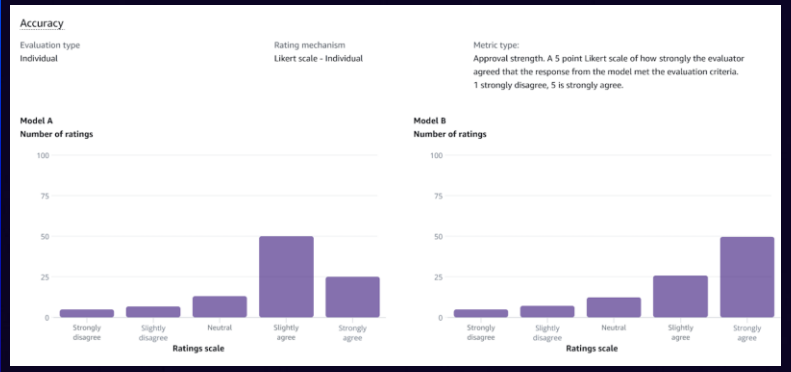
Upload datasets

Inference and scoring

View results

Get results in a few clicks

Human evaluation reports

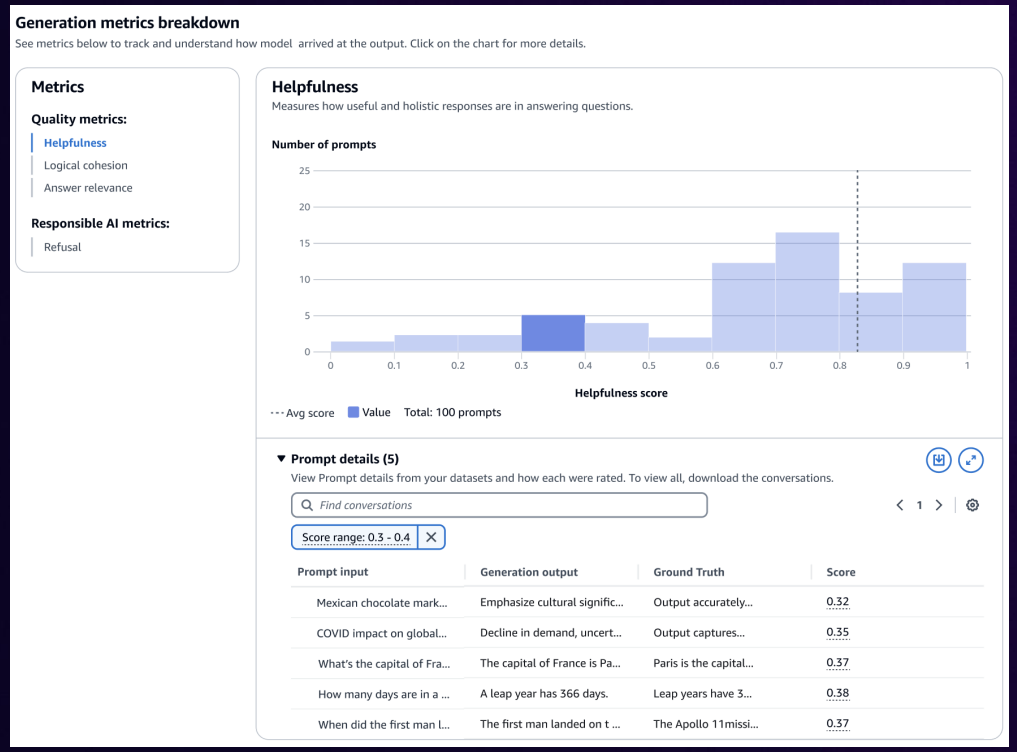


Track ratings from your team

See distributions visually

Simple metric explanations

LLM-as-a-judge reports

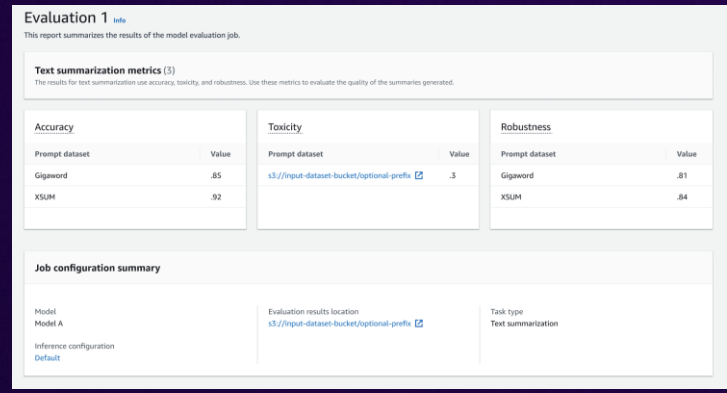


Simple to read scores

See distributions visually

See ratings explanations

Automatic evaluation reports



Simple-to-read scores

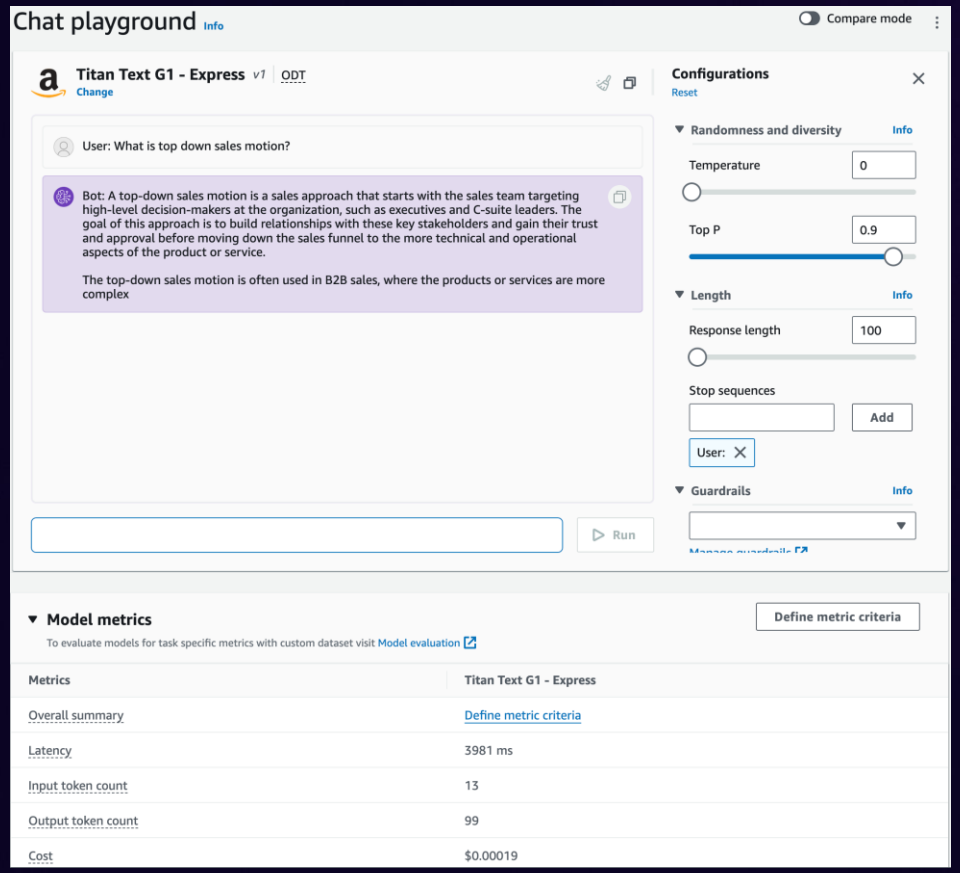
Curated per task type

Popular algorithms utilized

Quality/Cost/Latency tradeoffs

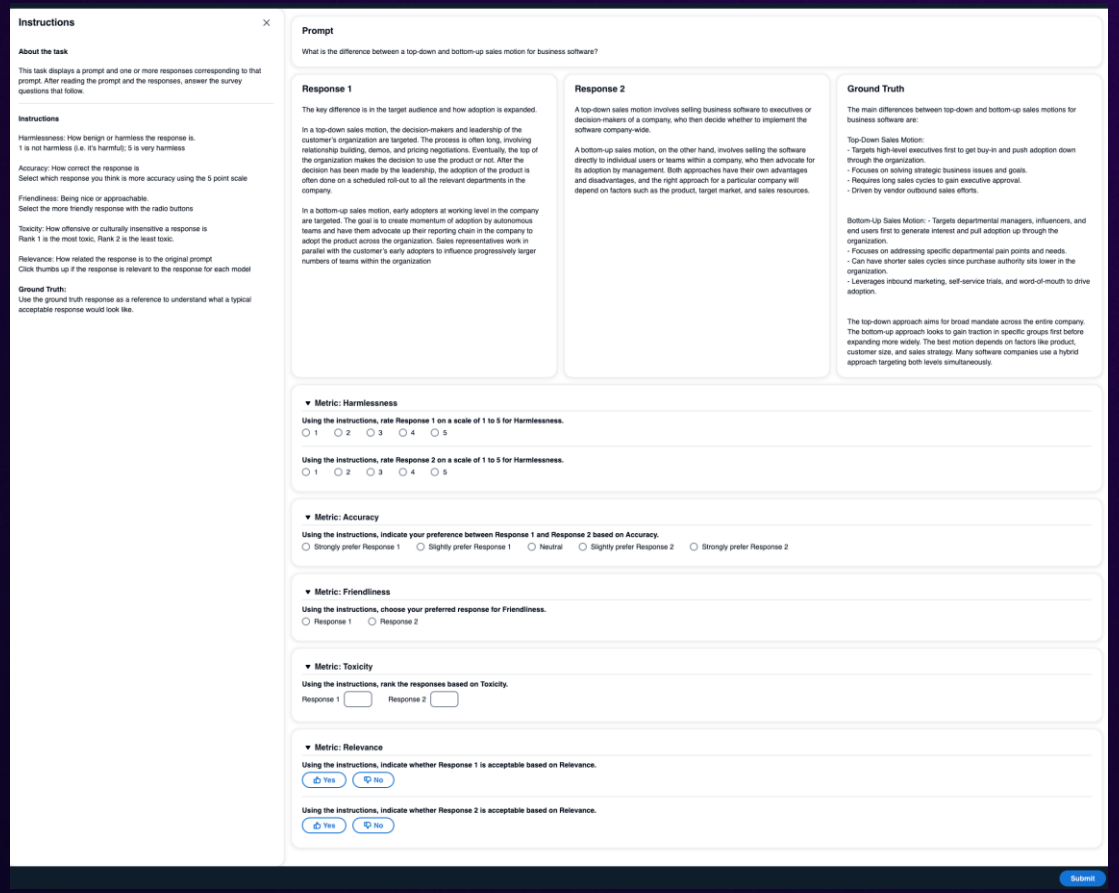
Cost and latency

Amazon Bedrock playgrounds



Quality metrics

Detailed evaluations
LLMaaJ, human, programmatic



Demo

LLM-as-a-judge

Model eval recap

Curated and custom tasks

Curated datasets or bring your own data

Easy-to-read reports

Native to Amazon Bedrock

Curated metrics or define your own

Curated algorithms

Automatic (programmatic)
Automatic (LLM-as-a-judge)
Human – Your team
Human – AWS-managed

Choose your metric type

Compare across jobs

NOW IN PREVIEW

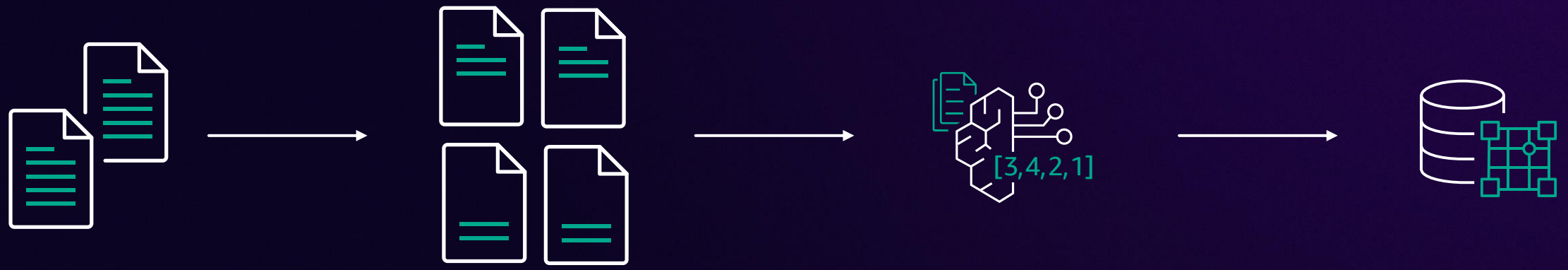
RAG evaluation with Amazon Bedrock Knowledge Bases

What is retrieval-augmented generation (RAG)?

- “LLM-powered search/answer generation”
- Augment prompt with retrieved information
- Useful for proprietary data
- Also useful for information after LLM training cutoff date
- Reduce hallucinations by grounding the prompt

RAG data preparation with Amazon Bedrock Knowledge Bases

The (very) simple version



User inputs data (documents, etc.)

Chunking

Convert to vector embeddings

Store in vector database

RAG during runtime

The (very) simple version

Behind the scenes with Amazon Bedrock Knowledge Bases

User

Submit prompt to application

Vectorize prompt

Search vector DB to **retrieve** relevant content

Augment prompt with retrieved content

Generate response with LLM

Return response to user

Special challenges with RAG evaluation

Use relevant data from your knowledge base

Retrieve the right context from documents

Generate a correct, complete, and grounded answer minimizing hallucinations

Iteratively improve your RAG system and compare across changes

Evaluate biases, safety, and trust

Public Preview

RAG evaluation on Amazon Bedrock Knowledge Bases

Evaluate your full Knowledge Base stack to optimize your RAG application

1. Bring your own datasets for tailored results
2. Evaluate retrieval alone or retrieval + generation with a choice of LLM-as-a-judge
3. Built-in metrics for quality and responsible AI, compatible with Amazon Bedrock Guardrails
4. Compare across multiple evaluation jobs
5. Get results in just a few clicks

aws

RAG evaluation input data format

Input dataset contains 2 things in JSONL format

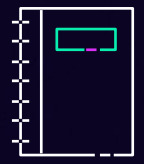
1. Prompt
2. Optional golden ground truth

```
{
    "conversationTurns": [{
        "referenceResponses": [{
            "content": [{
                "text": "This is a reference response"
            }]
        }],
        "prompt": {
            "content": [{
                "text": "This is a prompt"
            }]
        }
    }]
}
```


Retrieve jobs have `referenceContexts`

Choice of evaluation metrics

Retrieval



Context Coverage

Context Relevance

Retrieval + Generation

Correctness

Completeness

Helpfulness

Logical coherence

Faithfulness

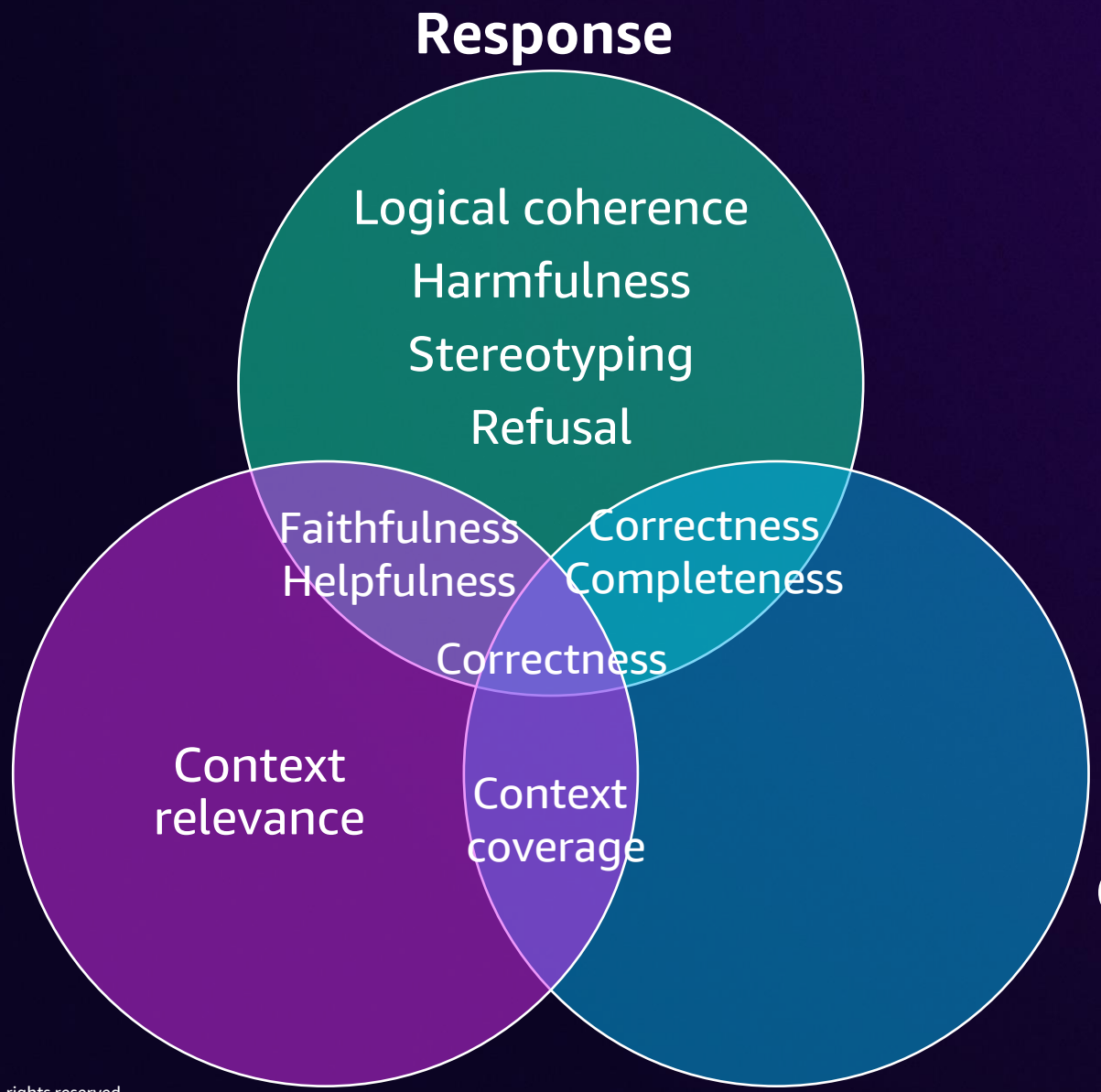
Responsible AI

Harmfulness

Stereotyping

Refusal

RAG evaluation metrics



*Prompt feeds into almost all

*Ground truth optional for correctness and completeness

How correctness works

Example input

prompt: What is the capital of Spain?

referenceResponse: Madrid

Model response: Barcelona

Judge prompt (simplified)

```
You are a helpful assistant…

You are given a question, a candidate
response from an LLM, and reference response.

Your task is to check if the candidate
response is correct compared to the reference
response…

Here is the actual task:
Question:  {prompt}
Reference Response: {referenceResponse}
Candidate Response: {Model response}

Explain your response, followed by your
evaluation:
2) Correct
1) Partially correct
0) Incorrect
```


Note: Correctness can be with or without ground truth

How RAG evaluation works with Knowledge Bases

Choose evaluator model

Choose your Knowledge Base

Choose to evaluate retrieval only or retrieve + generate

Choose your generator model

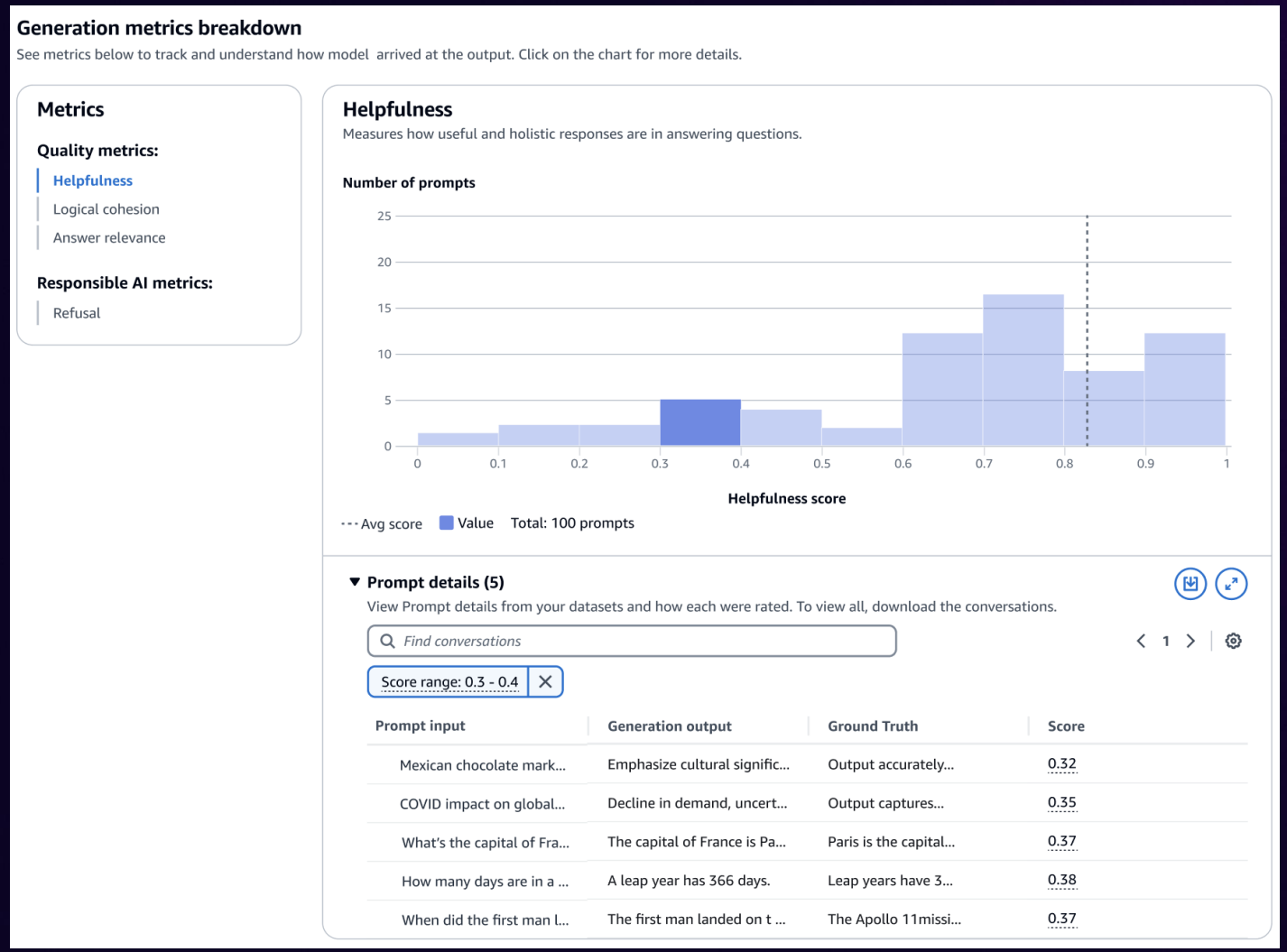
Choose your metrics

Upload your prompt dataset

Inference and evaluation

View results

Get results in a few clicks



- Simple to read scores
- See distributions visually
- See ratings explanations

Demo

RAG evaluation with Amazon Bedrock Knowledge Bases

Wrap-up

RAG evaluation recap

LLM-as-a-judge technology

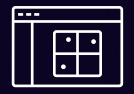
Bring your own data

Easy-to-read reports

Native to Amazon Bedrock

Curated metrics

Compare across jobs



Retrieval
Retrieve and Generate
Integration with Amazon Bedrock Guardrails

Responsible AI built in

Natural language explanations

Session recap

Model evaluation

LLM-as-a-judge, Human, Programmatic

Built-in datasets or bring your own

Human-like evaluation results from LLM-as-a-judge

Natively built on Amazon Bedrock

Compare across multiple jobs

Responsible AI metrics built in

Natural language explanations

RAG evaluation

Integrated with Amazon Bedrock Knowledge Bases

LLM-as-a-judge technology

Retrieval separately or end-to-end RAG together

Integration with Amazon Bedrock Guardrails

Bring your own dataset for relevant results

Compare across multiple jobs

Responsible AI metrics built in

Natural language explanations

Get started today

Scan for Amazon Bedrock website

Other opportunities to get started

- Schedule an Amazon Bedrock immersion day
- Reach out to your account manager
- Reach out to a solutions architect

Thank you!

Please complete the session survey in the mobile app

Jesse Manders
jrmander@amazon.com
linkedin.com/in/jessemanders

Shalendra Chhabra
shalenca@amazon.com
linkedin.com/in/shalendra

Hardik Vasa
harniva@amazon.com
linkedin.com/in/vasahardik/