

Hello!

Nirhoshan Sivaroopan

Software Engineer - AI Team



nirhoshan@wso2.com



Building Trust in Generative AI: Accuracy Evaluation and Automation



Generative AI: With and Without Context

Generative AI

- Strengths: Generative AI excels at producing fluent, creative responses.
- Limitations: Without relevant context, responses can be generic or off-topic, lacking precision.

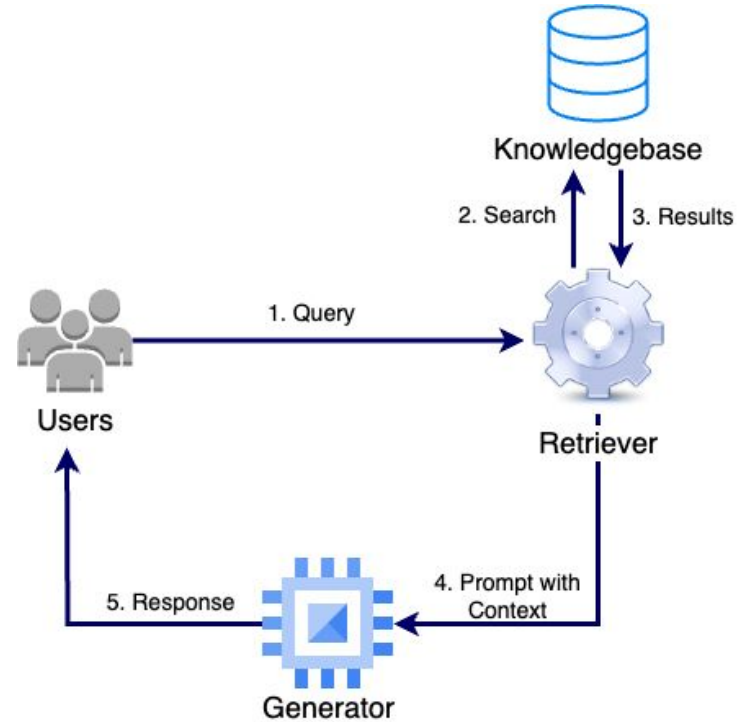
Generative AI with Context

- Enhanced Capabilities: Providing relevant context in the prompt ensures that the AI is aware of dynamic, specific information.
- Improved Accuracy: Contextual prompts guide the AI to generate more accurate and tailored responses.



Retrieval Augmented Generation (RAG)

- **RAG enhances Generative AI** by integrating external knowledge sources, improving response accuracy and relevance.
- The **retriever component** identifies relevant knowledge from a knowledge base through semantic similarity.
- The **generator component** produces content based on the context retrieved by the retriever, ensuring accurate responses.



Evaluating RAG Application Accuracy

- RAGAS library has defined multiple metrics to evaluate the accuracy of RAG application.
- Retriever can be evaluated with the following metrics
 - ⦿ Context Recall
 - ⦿ Context Precision
- Generator can be evaluated with the following metrics
 - ⦿ Faithfulness
 - ⦿ Answer Semantic Similarity

Some keywords

- Question – User's query that is being input to the RAG.
- Context – The documents that are retrieved from the knowledge base by the retriever.
- Answer – Answer generated by the RAG based on the retrieved context for the user's question.
- Ground truth (GT) – Answer annotated by human to the user's question.

Context Recall

- Measures the extent to which the retrieved context aligns with the GT.
- To achieve high context recall, all the claims in the GT answer should be available in the retrieved context.
- Incorrect context being retrieved will result in low context recall.

$$\textit{Context Recall} = \frac{|GT \text{ sentences that can be attributed to the context}|}{|Number of sentences in GT|}$$

Context Precision

- Evaluates whether all the ground-truth relevant items present in the contexts are ranked higher or not.
- Higher the rank of the most relevant document in the context, will result in higher context precision.

$$\text{Context Precision @ } K = \frac{\sum_{k=1}^K (\text{Precision@}k \times v_k)}{\text{Total number of relevant items in the top } K \text{ results}}$$

- Where K is the total number of docs in the contexts and $v_k \in \{0,1\}$ is the relevance indicator at rank k .

$$\text{Precision@}k = \frac{\text{true positives @}k}{(\text{true positives@}k + \text{false positives@}k)}$$

Faithfulness

- Measures the factual consistency of the generated answer against the given context.
- If any sentence has been hallucinated in the generated answer, then the faithfulness score will be low as that sentence is not available in the retrieved context.

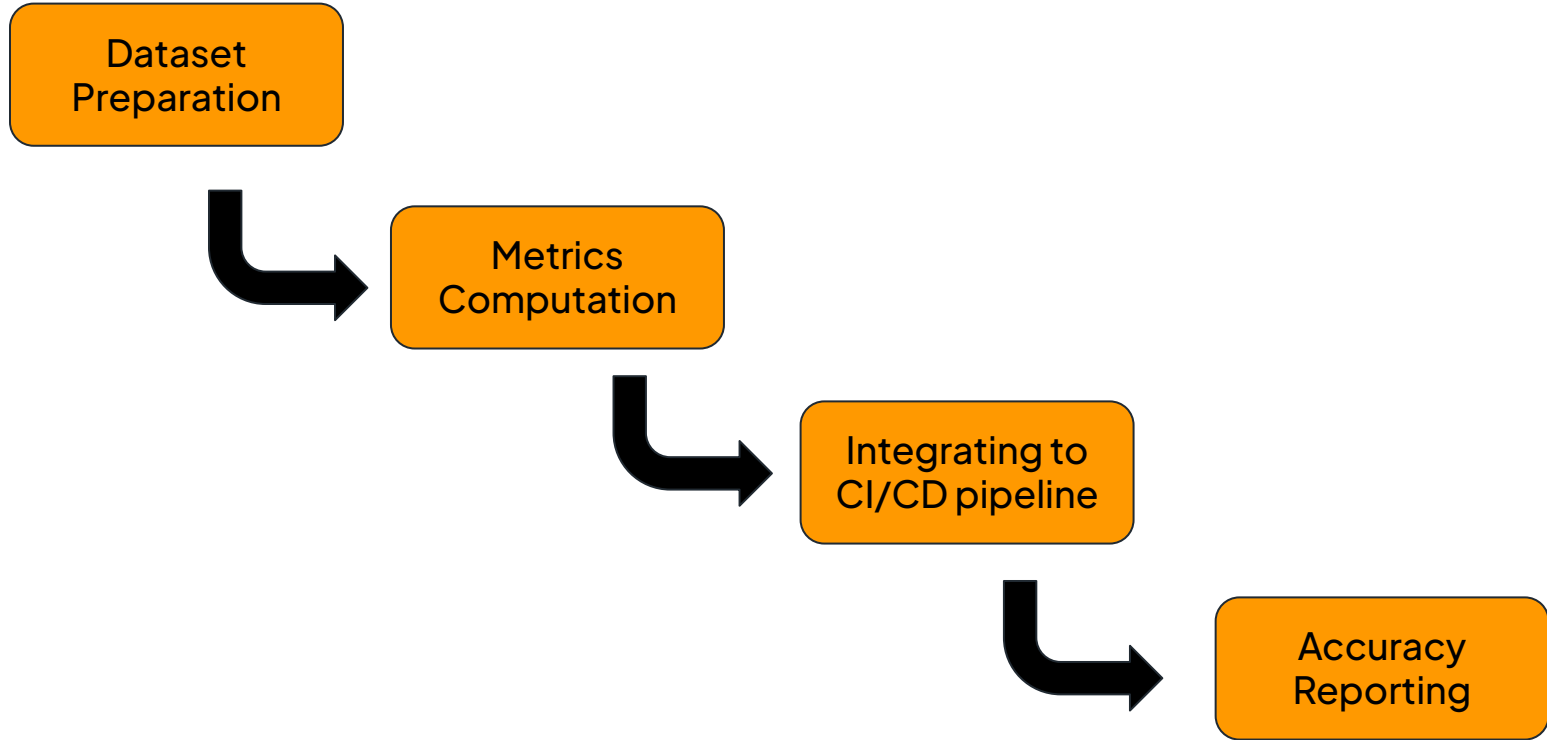
$$\textit{Faithfulness score} = \frac{|\textit{Number of claims in the generated answer that can be inferred from given context}|}{|\textit{Total number of claims in the generated answer}|}$$



Answer Semantic Similarity

- Compares the generated answer with the GT answer, assessing how well the output aligns in semantic meaning with the expected response.
- Both the GT and generated answer will be vectorized through the embeddings model and the cosine similarity will be calculated.

Key stages of accuracy evaluation



Dataset Preparation

1	Questions	Ground truths
2	How to deploy a webapp?	[To deploy a web application in Choreo, follow the steps below:\n\n1. Create a project: On the organization home page, click + Create Project. Enter a display name, unique name, and description for the project. Click Create. This creates the project and opens the project home page. \n\n2. Create a Web Application component: On the project home page, click Web Application under Create a Single Component. Enter a unique name and a description for the web application. \n\n3. Connect to GitHub: Go to the GitHub tab and click Authorize with GitHub. If you have not already connected your GitHub repository to Choreo, enter your GitHub credentials and select the repository. You can also paste the repository URL in the Provide Repository URL field to connect to it without requiring authorization. \n\n4. Enter the required information: Enter the required information such as your GitHub account in the Organization field, repository name in the Repository field, and the branch name in the Branch field. \n\n5. Select Buildpack: Select NodeJS as the Buildpack and provide the necessary details such as NodeJS Project Directory, Language Version, and Port. \n\n6. Create the component: Click Create. Choreo initializes the component with the sample implementation and opens the Overview page of the component.\n\n7. Deploy the web application: In the left navigation menu, click Deploy. In the Set Up card, click Configure and Deploy. Click Next to move to the Authentication Settings pane. Keep the default settings and click Deploy. \n\n8. Verify the deployment: To verify that you have successfully hosted the web application, click the Web App URL in the Development card. This takes you to the web application.\n\nFor more information, you can refer to the official Choreo documentation [here](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/#step-2-create-a-web-application-component) and [here](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/#step-4-deploy-and-access-your-web-application).]

Metrics Computation

```
Python
from ragas import evaluate
from ragas.metrics import context_recall, context_precision, faithfulness,
answer_similarity

def test_metrics_thresholds(self):
    questions, ground_truths = load_data('test_data/validation_dataset.csv')
    questions, answers, contexts = asyncio.run(process_questions(questions))
    data = {
        "question": questions,
        "answer": answers,
        "contexts": contexts,
        "ground_truths": ground_truths
    }
    dataset = Dataset.from_dict(data)
    results = evaluate(
        dataset=dataset,
        metrics=[context_precision, context_recall, faithfulness,
            answer_similarity],
    ).to_pandas()
    results.to_csv('test_data/accuracy_results.csv', index=False)
    metric_columns = ['context_precision', 'context_recall', 'faithfulness',
        'answer_similarity']
    metric_scores = results[metric_columns]
    mean_scores = metric_scores.mean()
    thresholds = {'context_precision': 0.95, 'context_recall': 0.90,
        'faithfulness': 0.90, 'answer_similarity': 0.90}
    for metric, threshold in thresholds.items():
        with self.subTest(metric=metric):
            self.assertGreaterEqual(mean_scores[metric], threshold,
                f"{metric} average of {mean_scores[metric]:.2f} is below the threshold"
                    f"of {threshold}.")
```

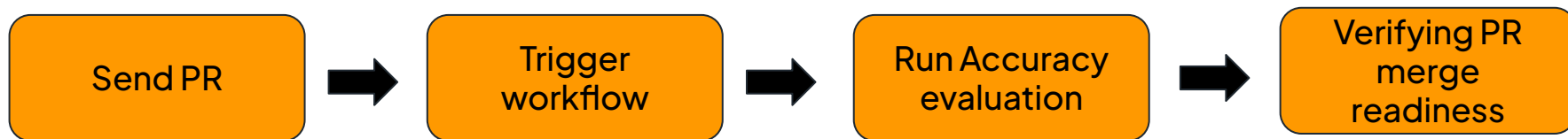
Integrating to CI/CD pipeline

```
Unset
name: Accuracy Check CI
on:
  push:
    branches:
      - main
jobs:
  accuracy-check:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout code
        uses: actions/checkout@v3
      - name: Set up Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.11'
      - name: Install dependencies
        run: |
          sudo apt update
          python3.11 -m pip install --upgrade pip setuptools wheel pytest
          python3.11 -m pip install --no-cache-dir -r requirements.txt
      - name: Run Accuracy Test
        run: |
          export PYTHONPATH="$(pwd)/docs-assistant-bot:$PYTHONPATH"
          set -eo pipefail
          python3.11 -m pytest tests/test_accuracy.py --log-cli-level=INFO
    env:
      OPENAI_API_KEY: ${ secrets.OPENAI_API_KEY }
      ZILLIZ_CLOUD_API_KEY: ${ secrets.ZILLIZ_CLOUD_API_KEY }
      ZILLIZ_CLOUD_URI: ${ vars.ZILLIZ_CLOUD_URI }
      - name: Publish Accuracy Test Results
        uses: actions/upload-artifact@v3
        with:
          name: AccuracyTestResults
          path: 'test_data/accuracy_results.csv'
```

Accuracy Reporting

1	Questions	Ground truths	Answer	Context	Context precision	Context Recall	Faithfulness	Answer Similarity
2	How to deploy a webapp?	[To deploy a web application in Choreo, follow the steps below:\n\n1. Create a project: On the organization home page, click ++ Create Project. Enter a display name, unique name, and description for the project. Click Create. This creates the project and opens the project home page. \n\n2. Create a Web Application component: On the project home page, click Web Application under Create a Single Component. Enter a unique name and a description for the web application. \n\n3. Connect to GitHub: Go to the GitHub tab and click Authorize with GitHub. \n\n4. Enter the required information: Enter the required information such as your GitHub account in the Organization field, repository name in the Repository field, and the branch name in the Branch field. \n\n5. Select Buildpack: Select NodeJS as the Buildpack and provide the necessary details such as NodeJS Project Directory, Language Version, and Port. \n\n6. Create the component: Click Create. Choreo initializes the component with the sample implementation and opens the Overview page of the component. \n\nFor more information, you can refer to the official Choreo documentation [here](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/#step-2-create-a-web-application-component) and [here](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/#step-4-deploy-and-access-your-web-application)].	# To deploy a web application using Choreo, follow these steps:\n\n### Step 1: Create a Project\n1. Navigate to the Choreo Console. \n2. Create a Project: \n - Click on ++ Create Project. \n - Enter a display name, a unique name, and a description for the project. \n - Click Create to initiate the project. \n\n### Step 2: Create a Web Application Component\n1. On the project home page, select Web Application under Create a Component. \n2. Provide a unique name and description for your web application. \n3. Connect to GitHub: \n - Go to the GitHub tab and click Authorize with GitHub. \n4. Fill in the necessary fields: \n - Organization: Your GitHub account \n - Repository: The name of your repository \n - Branch: The branch you want to use (e.g., <code>main</code>) \n5. Select NodeJS as the Buildpack. \n6. Enter additional configuration: \n - NodeJS Project Directory: <code>/</code> \n - Language Version: e.g., <code>20.x.x</code> \n - Port: <code>8080</code> \n7. Click Create to initialize the component. \n\nFor more detailed guidance, you can refer to the following documentation: \n- [Build and Deploy a Single-Page Web Application](https://wso2.com/choreo/docs/develop-op-components/develop-web-applications/build-and-deploy-a-single-page-web-application/) \n- [Deploy Your First Static Web Application](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/) \n- [Deploy a Web Application that Consumes a Backend Service](https://wso2.com/choreo/docs/quick-start-guides/deploy-a-web-application-that-consume-s-a-backend-service/)	"[doc: #Deploy Your First Static Web Application\n\n##Step 2: Create a Web Application component\nTo create a Web Application component, follow the steps given below: \n1. On the project home page, click Web Application under Create a Component. \n2. Enter a unique name and a description for the web application. \n3. Go to the GitHub tab. \n4. To allow Choreo to connect to your GitHub account, click Authorize with GitHub. \n5. Enter the required information: Enter the required information such as your GitHub account in the Organization field, repository name in the Repository field, and the branch name in the Branch field. \n\n6. Select Buildpack: Select NodeJS as the Buildpack and provide the necessary details such as NodeJS Project Directory, Language Version, and Port. \n7. Click Create. Choreo initializes the component with the sample implementation and opens the Overview page of the component. \n\nNow let's build and deploy the web application.. You can refer to this [document](https://wso2.com/choreo/docs/quick-start-guides/deploy-your-first-static-web-application/#step-2-create-a-web-application-component) for more details.]"	0.998	0.98	1	0.9501297805

Automation pipeline



Challenges

- Evaluation results are depended on the accuracy of the LLM used underneath.
- Dataset should cover questions of different complexities
- Creating high-quality ground truth datasets is time-consuming and expensive
- LLM API costs can add up for large-scale evaluations

Thank you !

