**Industrial Sensor Tag Comparison using RAG with LLM**

# **Project Overview**

This project involves building a Retrieval Augmented Generation (RAG) model to compare industrial sensor tags from two data sources—PG\_TAGS (original datasheet specifications) and VQ\_TAGS (vendor quotation document). The goal is to identify **matching tags**, **discrepancies**, and **concerns** such as missing tags or anomalies between these two datasets.

By leveraging **Large Language Models (LLMs)** and a **retrieval system**, this solution uses the combined power of retrieval and generation to perform and explain detailed comparisons across structured data sources.

# **Approach and Methodology**

# **Data Preparation**

# **Objective**:

To structure both PDF documents into DataFrames that can be easily queried and analyzed.

# **Procedure**:

1. **Extract Data from PDF Documents:**
   * The PG\_TAGS.pdf and VQ\_TAGS.pdf documents contain tabular data. We used pdfplumber to extract text from these documents and processed the data into structured tables.
2. **Convert to DataFrames**:
   * Each extracted table was converted into a DataFrame, with columns such as Tag Number, Calibration Range, and Operating Range to represent the tag data from each document.

# **Outcome:**

The data from each document is now stored in two DataFrames (df\_pg\_tags and df\_vq\_tags) with a uniform structure.

# **Comparison Logic**

# **Objective:**

To identify matching tags, discrepancies, and concerns between PG\_TAGS and VQ\_TAGS.

# **Procedure:**

1. **Define the Comparison Function (compare\_documents):**
   * The compare\_documents function takes df\_pg and df\_vq as inputs, iterating over each tag in df\_pg to search for corresponding tags in df\_vq.
   * This function:
   * Identifies **Matching Tags** where calibration and operating ranges are identical.
   * Flags **Discrepancies** where tags have different calibration or operating range values.
   * Lists **Concerns** for tags missing in either PG\_TAGS or VQ\_TAGS.
2. **Categorize and Record Results:**
   1. **Matching Tags**: For each tag where the values in both documents are identical, a record is added to matching\_tags.
   2. **Discrepancies**: For tags found in both documents but with differing values, both the PG and VQ values are stored in the discrepancies list.
   3. **Concerns**: For tags found in one document but missing from the other, the tag is added to concerns with an appropriate issue note.

# **Outcome:**

The compare\_documents function provides three structured lists: matching\_tags, discrepancies, and concerns, each representing the outcome of the tag comparison process.

\*\*Also added **visualization** to represent the comparison results graphically (e.g., with matplotlib or plotly) to improve user understanding of discrepancies and matching\_tags.\*\*

# **LLM-based Summarization and Explanation**

# **Objective:**

To use an LLM to generate a descriptive summary of the comparison results, including detailed explanations of discrepancies and concerns.

# **Procedure:**

1. **Prompt Template Design**:
   * We created a **prompt template** to guide the LLM (Large Language Model) in summarizing the comparison results. This template asks the LLM to:
     + Describe **matching tags** in detail.
     + Explain **discrepancies**, highlighting fields with mismatched values.
     + List any **concerns**, noting missing tags and any other detected issues.
2. **Format the Comparison Results for LLM Input**:
   * The matching\_tags, discrepancies, and concerns lists were converted into formatted strings that the LLM could interpret easily. This involved creating readable lines for each tag’s information.
3. **Generate Summary with LangChain**:
   * Using **LangChain’s LLMChain** with **OpenAI’s LLM** model, we passed the comparison results into the LLM with the prompt template.
   * The LLM generated a natural language summary, detailing:
     + Which tags matched perfectly.
     + The specific discrepancies, with explanations on calibration and operating range mismatches.
     + Concerns, including any missing tags and anomalies found.

# **Outcome:**

The LLM produced a detailed, readable summary of the comparison, outlining the matches, discrepancies, and concerns with clarity and context.

# **Conclusion**

By implementing a RAG model with an LLM, we created a powerful system capable of comparing structured data from two documents and generating human-readable summaries of the results. This approach is effective for industrial applications where precise tag comparisons and anomaly detection are crucial.

The final solution offers:

* **Efficient retrieval and comparison** of document data.
* **Descriptive, LLM-generated summaries** that explain findings in detail.
* A foundation for further enhancement with visualization, historical tracking, and fine-tuned LLM models.

This RAG model demonstrates the value of combining retrieval-based systems with LLMs for structured data analysis, making it highly adaptable for various document comparison tasks.