Implementación de RAG con Haystack. El LLM utilizado es Mistral mistral-medium-latest. Se utiliza UnstructuredPDFLoader para cargar el archivo .pdf con información de un contexto específico. Se puede responder a la pregunta de un usuario sobre ese dominio de aplicación utilizando el poder de las respuestas generativas con dicho LLM.

```
%pip install --q unstructured langchain
%pip install --q "unstructured[all-docs]"
```

```
!pip install langchain-community
```

```
Requirement already satisfied: langchain-community in /usr/local/lib/python3.10/dist-packages (0.2.5)
Requirement already satisfied: PyYAML>=5.3 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (6.0.1)
Requirement already satisfied: SQLAlchemy<3,>=1.4 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (2.0.30)
Requirement already satisfied: aiohttp<4.0.0,>=3.8.3 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (3.9.5)
Requirement already satisfied: dataclasses-json<0.7,>=0.5.7 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.6.7)
Requirement already satisfied: langchain<0.3.0,>=0.2.5 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.2.5)
Requirement already satisfied: langchain-core<0.3.0,>=0.2.7 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.2.9)
Requirement already satisfied: langsmith<0.2.0,>=0.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (0.1.81)
Requirement already satisfied: numpy<2,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (1.25.2)
Requirement already satisfied: requests<3,>=2 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (2.31.0)
Requirement already satisfied: tenacity<9.0.0,>=8.1.0 in /usr/local/lib/python3.10/dist-packages (from langchain-community) (8.4.1)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in /usr/local/lib/python3.10/dist-packages (from aiohttp<4.0.0,>=3.8.3->langchain-community) (4.0.3)
Requirement already satisfied: marshmallow<4.0.0,>=3.18.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json<0.7,>=0.5.7->langchain-community) (3.21.3)
Requirement already satisfied: typing-inspect<1,>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from dataclasses-json<0.7,>=0.5.7->langchain-community) (0.9.0)
Requirement already satisfied: langchain-text-splitters<0.3.0,>=0.2.0 in /usr/local/lib/python3.10/dist-packages (from langchain<0.3.0,>=0.2.5->langchain-community) (0.2.1)
Requirement already satisfied: pydantic<3,>=1 in /usr/local/lib/python3.10/dist-packages (from langchain<0.3.0,>=0.2.5->langchain-community) (2.7.4)
Requirement already satisfied: jsonpatch<2.0,>=1.33 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.3.0,>=0.2.7->langchain-community) (1.33)
Requirement already satisfied: packaging<25,>=23.2 in /usr/local/lib/python3.10/dist-packages (from langchain-core<0.3.0,>=0.2.7->langchain-community) (24.1)
Requirement already satisfied: orjson<4.0.0,>=3.9.14 in /usr/local/lib/python3.10/dist-packages (from langsmith<0.2.0,>=0.1.0->langchain-community) (3.10.5)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3,>=2->langchain-community) (2024.6.2)
Requirement already satisfied: typing-extensions>=4.6.0 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain-community) (4.12.2)
Requirement already satisfied: greenlet!=0.4.17 in /usr/local/lib/python3.10/dist-packages (from SQLAlchemy<3,>=1.4->langchain-community) (3.0.3)
Requirement already satisfied: jsonpointer>=1.9 in /usr/local/lib/python3.10/dist-packages (from jsonpatch<2.0,>=1.33->langchain-core<0.3.0,>=0.2.7->langchain-community) (3.
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0.3.0,>=0.2.5->langchain-community) (0.7.0)
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1->langchain<0.3.0,>=0.2.5->langchain-community) (2.18.4)
Requirement already satisfied: mypy-extensions>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from typing-inspect<1,>=0.4.0->dataclasses-json<0.7,>=0.5.7->langchain-comm
```

```
from langchain_community.document_loaders import UnstructuredPDFLoader
from langchain_community.document_loaders import OnlinePDFLoader
```

Referencia:
https://api.python.langchain.com/en/latest/document_loaders/langchain_community.document_loaders.pdf.UnstructuredPDFLoader.html

```
# Instalación de PyPDF2
!pip install PyPDF2

# Importar la biblioteca necesaria
import PyPDF2

# Ruta local del archivo PDF
local_path = "/content/00f6e6_ManualMEGAMIG500Spanish.pdf"

# Función para unir todas las páginas en un archivo .txt
def unir_paginas_en_txt(file_path):
    # Verificar que se haya proporcionado un archivo PDF
    if file_path:
        # Cargar el archivo PDF usando UnstructuredPDFLoader
        loader = UnstructuredPDFLoader(file_path=file_path)
        pdf_obj = loader.load()  # Suponiendo que esto devuelve un objeto que permite acceder a cada página

        # Crear un objeto PDF de PyPDF2 para manejar el archivo PDF
        pdf_reader = PyPDF2.PdfReader(open(file_path, "rb"))

        # Ruta de salida para el archivo .txt combinado
        output_file = "/content/combined_text.txt"

        # Abrir el archivo de texto para escribir
        with open(output_file, "w", encoding="utf-8") as txt_file:
            # Recorrer todas las páginas del PDF
            for page_num in range(len(pdf_reader.pages)): # Use len(pdf_reader.pages) instead of pdf_reader.numPages
                page = pdf_reader.pages[page_num] # Access pages using index
                page_text = page.extract_text()

                # Escribir el contenido de la página en el archivo de texto
                txt_file.write(f"Página {page_num + 1}\n\n")
                txt_file.write(page_text)
                txt_file.write("\n\n---\n\n")  # Separador entre páginas

        print(f"Se ha creado el archivo combinado: {output_file}")
    else:
        print("Sube un archivo PDF")

# Llamar a la función para unir todas las páginas en un archivo .txt
unir_paginas_en_txt(local_path)
```

```
Requirement already satisfied: PyPDF2 in /usr/local/lib/python3.10/dist-packages (3.0.1)
Se ha creado el archivo combinado: /content/combined_text.txt
```

```
!pip install mistral-haystack==0.0.1
```

```
Collecting mistral-haystack==0.0.1
  Downloading mistral_haystack-0.0.1-py3-none-any.whl (11 kB)
Collecting haystack-ai>=2.0.0b6 (from mistral-haystack==0.0.1)
  Downloading haystack_ai-2.2.3-py3-none-any.whl (345 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 345.2/345.2 kB 4.8 MB/s eta 0:00:00
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (3.1.4)
Collecting lazy-imports (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
```

```
  Downloading lazy_imports-0.3.1-py3-none-any.whl (12 kB)
Requirement already satisfied: more-itertools in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (10.1.0)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (3.3)
Requirement already satisfied: numpy<2 in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (1.25.2)
Collecting openai>=1.1.0 (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
  Downloading openai-1.35.3-py3-none-any.whl (327 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 327.4/327.4 kB 9.0 MB/s eta 0:00:00
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.0.3)
Collecting posthog (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
  Downloading posthog-3.5.0-py2.py3-none-any.whl (41 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 41.3/41.3 kB 4.0 MB/s eta 0:00:00
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.8.2)
Requirement already satisfied: pyyaml in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (6.0.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.31.0)
Requirement already satisfied: tenacity!=8.4.0 in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (8.4.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (4.66.4)
Requirement already satisfied: typing-extensions>=4.7 in /usr/local/lib/python3.10/dist-packages (from haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (4.12.2)
Requirement already satisfied: anyio<5,>=3.5.0 in /usr/local/lib/python3.10/dist-packages (from openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (3.7.1)
Requirement already satisfied: distro<2,>=1.7.0 in /usr/lib/python3/dist-packages (from openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (1.7.0)
Requirement already satisfied: httpx<1,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (0.27.0)
Requirement already satisfied: pydantic<3,>=1.9.0 in /usr/local/lib/python3.10/dist-packages (from openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.7.4)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (1.3.1)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.1.5)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2024.1)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (1.16.0)
Collecting monotonic>=1.5 (from posthog->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
  Downloading monotonic-1.6-py2.py3-none-any.whl (8.2 kB)
Requirement already satisfied: backoff>=1.10.0 in /usr/local/lib/python3.10/dist-packages (from posthog->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.2.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1) (2024.6.2)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio<5,>=3.5.0->openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.23.0->openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haystack==0.0.1)
Requirement already satisfied: h11<0.15,>=0.13 in /usr/local/lib/python3.10/dist-packages (from httpcore==1.*->httpx<1,>=0.23.0->openai>=1.1.0->haystack-ai>=2.0.0b6->mistral
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1.9.0->openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-hayst
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=1.9.0->openai>=1.1.0->haystack-ai>=2.0.0b6->mistral-haysta
Installing collected packages: monotonic, lazy-imports, posthog, openai, haystack-ai, mistral-haystack
Successfully installed haystack-ai-2.2.3 lazy-imports-0.3.1 mistral-haystack-0.0.1 monotonic-1.6 openai-1.35.3 posthog-3.5.0
```

```
!pip install mistralai
```

```
Collecting mistralai
  Downloading mistralai-0.4.1-py3-none-any.whl (19 kB)
Requirement already satisfied: httpx<1,>=0.25 in /usr/local/lib/python3.10/dist-packages (from mistralai) (0.27.0)
Requirement already satisfied: orjson<3.11,>=3.9.10 in /usr/local/lib/python3.10/dist-packages (from mistralai) (3.10.5)
Requirement already satisfied: pydantic<3,>=2.5.2 in /usr/local/lib/python3.10/dist-packages (from mistralai) (2.7.4)
Requirement already satisfied: anyio in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.25->mistralai) (3.7.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.25->mistralai) (2024.6.2)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.25->mistralai) (1.0.5)
Requirement already satisfied: idna in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.25->mistralai) (3.7)
Requirement already satisfied: sniffio in /usr/local/lib/python3.10/dist-packages (from httpx<1,>=0.25->mistralai) (1.3.1)
Requirement already satisfied: h11<0.15,>=0.13 in /usr/local/lib/python3.10/dist-packages (from httpcore==1.*->httpx<1,>=0.25->mistralai) (0.14.0)
Requirement already satisfied: annotated-types>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=2.5.2->mistralai) (0.7.0)
Requirement already satisfied: pydantic-core==2.18.4 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=2.5.2->mistralai) (2.18.4)
Requirement already satisfied: typing-extensions>=4.6.1 in /usr/local/lib/python3.10/dist-packages (from pydantic<3,>=2.5.2->mistralai) (4.12.2)
Requirement already satisfied: exceptiongroup in /usr/local/lib/python3.10/dist-packages (from anyio->httpx<1,>=0.25->mistralai) (1.2.1)
```

```
Installing collected packages: mistralai
Successfully installed mistralai-0.4.1
```

```
from mistralai.client import MistralClient
from mistralai.models.chat_completion import ChatMessage
from getpass import getpass

api_key= getpass("Type your API Key")
client = MistralClient(api_key=api_key)
```

```
Type your API Key·····
```

```
from haystack import Pipeline
from haystack.document_stores.in_memory import InMemoryDocumentStore
from haystack.dataclasses import ChatMessage
from haystack.utils.auth import Secret

from haystack.components.builders import DynamicChatPromptBuilder
from haystack.components.converters import TextFileToDocument
from haystack.components.preprocessors import DocumentSplitter
from haystack.components.retrievers.in_memory import InMemoryEmbeddingRetriever
from haystack.components.writers import DocumentWriter
from haystack_integrations.components.embedders.mistral import MistralDocumentEmbedder, MistralTextEmbedder
from haystack_integrations.components.generators.mistral import MistralChatGenerator

document_store = InMemoryDocumentStore()

docs = TextFileToDocument().run(sources=["/content/combined_text.txt"])
split_docs = DocumentSplitter(split_by="passage", split_length=2).run(documents=docs["documents"])
embeddings = MistralDocumentEmbedder(api_key=Secret.from_token(api_key)).run(documents=split_docs["documents"])
DocumentWriter(document_store=document_store).run(documents=embeddings["documents"])


text_embedder = MistralTextEmbedder(api_key=Secret.from_token(api_key))
retriever = InMemoryEmbeddingRetriever(document_store=document_store)
prompt_builder = DynamicChatPromptBuilder(runtime_variables=["documents"])
llm = MistralChatGenerator(api_key=Secret.from_token(api_key),
                           model='mistral-large-latest')

chat_template = """Responde la siguiente pregunta basada en el documento proporcionado.\n
                Question: {{query}}\n
                Documents:
                {% for document in documents %}
                    {{document.content}}
                {% endfor%}
                """
messages = [ChatMessage.from_user(chat_template)]

rag_pipeline = Pipeline()
rag_pipeline.add_component("text_embedder", text_embedder)
rag_pipeline.add_component("retriever", retriever)
rag_pipeline.add_component("prompt_builder", prompt_builder)
rag_pipeline.add_component("llm", llm)


rag_pipeline.connect("text_embedder.embedding", "retriever.query_embedding")
rag_pipeline.connect("retriever.documents", "prompt_builder.documents")
rag_pipeline.connect("prompt_builder.prompt", "llm.messages")

question = "¿Qué se debe hacer antes de soldar?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
        "llm": {"generation_kwargs": {"max_tokens": 1000}},
    }
```

```
)

# La respuesta generada por el modelo se almacena en result["llm"]["replies"][0].content
response= result["llm"]["replies"][0].content
# Imprimir la respuesta generada
print(response)
```

```
Calculating embeddings: 100%|██████████| 1/1 [00:01<00:00,  1.20s/it]
Antes de soldar, se deben tomar las siguientes medidas de seguridad y realizar los siguientes chequeos:

1. Verificar que no exista riesgo potencial de caída de objetos extraños, tanto para el operador como para la máquina.
2. Asegurarse de que el polvo, ácido o material corrosivo en el ambiente del lugar de trabajo y medio ambiente no sobrepasen los parámetros exigidos (exceptuando los provoca
3. Instalar el equipo de soldar protegido del sol, la lluvia, humedad excesiva o temperaturas por debajo de los -10° C o por encima de los 40° C.
4. Asegurarse de que haya al menos 50 cms. libres alrededor del equipo de soldar para asegurar una adecuada ventilación.
5. No introducir piezas extrañas al equipo de soldar.
6. Evitar vibraciones excesivas en el área alrededor del lugar de trabajo de la máquina.
7. Elegir un área sin interferencias electromagnéticas.
8. Instalar un interruptor automático de protección en la instalación eléctrica de acuerdo a las especificaciones indicadas por el manual de servicio del equipo.
9. Instalar el equipo de soldar horizontalmente, en una superficie plana. Si la inclinación fuera superior a 15°, se le debe adicionar elementos anti vuelco para evitar la i
10. No se recomienda conectar el equipo de soldar a un grupo generador.

Chequeo de Seguridad:

1. Verificar que la conexión de tierra de la toma eléctrica del equipo de soldar esté conectada a tierra en forma correcta.
2. Verificar que los cables de salida no estén en corto circuito.
3. Verificar que los cables de salida y alimentación estén en buenas condiciones, sin daños o alteraciones y/o fuera de estándar.
4. Cortar el suministro de corriente eléctrica antes de abrir la carcaza.
5. Ante cualquier problema técnico que presente el equipo de soldar, recurrir al servicio técnico autorizado más cercano.

Se deben realizar chequeos regulares cada seis meses por parte de personal calificado luego de la instalación del equipo de soldar. Estos chequeos deben incluir:

1. Limpieza de rutina para asegurar que no existan condiciones subnormales como conectores sueltos.
2. Inspección de los accesorios externos.
3. Chequeo del cable de soldar, para verificar que se encuentre en perfectas condiciones.
4. Reemplazo del cable alimentador si se determina que está dañado o roto.

Además, el operador debe utilizar ropa y herramientas apropiadas para evitar dañar la vista y la piel, y usar la máscara de soldar cubriendo toda la cabeza, realizando la ob
```

Medición de las métricas de las respuestas obtenidas comparadas con las respuestas esperadas

```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
!pip install rouge-score
from rouge_score import rouge_scorer
```

```
!pip install rouge-score # Install the missing module in the current notebook
from rouge_score import rouge_scorer
```

```
Collecting rouge-score
  Downloading rouge_score-0.1.2.tar.gz (17 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: absl-py in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.4.0)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from rouge-score) (3.8.1)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.25.2)
Requirement already satisfied: six>=1.14.0 in /usr/local/lib/python3.10/dist-packages (from rouge-score) (1.16.0)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (2024.5.15)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk->rouge-score) (4.66.4)
Building wheels for collected packages: rouge-score
  Building wheel for rouge-score (setup.py) ... done
  Created wheel for rouge-score: filename=rouge_score-0.1.2-py3-none-any.whl size=24933 sha256=0e2aa2e6bd67a55bff184fa33f6da220be0f592b3fbae9a28faff6df0cd136fa
  Stored in directory: /root/.cache/pip/wheels/5f/dd/89/461065a73be61a532ff8599a28e9beef17985c9e9c31e541b4
Successfully built rouge-score
Installing collected packages: rouge-score
Successfully installed rouge-score-0.1.2
```

```
!pip install bert-score # Install the missing module
from bert_score import score
```

```
Collecting bert-score
  Downloading bert_score-0.3.13-py3-none-any.whl (61 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 61.1/61.1 kB 1.9 MB/s eta 0:00:00
Requirement already satisfied: torch>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.3.0+cu121)
Requirement already satisfied: pandas>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.0.3)
Requirement already satisfied: transformers>=3.0.0 in /usr/local/lib/python3.10/dist-packages (from bert-score) (4.41.2)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from bert-score) (1.25.2)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from bert-score) (2.31.0)
Requirement already satisfied: tqdm>=4.31.1 in /usr/local/lib/python3.10/dist-packages (from bert-score) (4.66.4)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (from bert-score) (3.7.1)
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.10/dist-packages (from bert-score) (24.1)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas>=1.0.1->bert-score) (2024.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.15.1)
Requirement already satisfied: typing-extensions>=4.8.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (4.12.2)
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (1.12.1)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.3)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (3.1.4)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2023.6.0)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: nvidia-cudnn-cu12==8.9.2.26 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (8.9.2.26)
Requirement already satisfied: nvidia-cublas-cu12==12.1.3.1 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.3.1)
Requirement already satisfied: nvidia-cufft-cu12==11.0.2.54 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (11.0.2.54)
Requirement already satisfied: nvidia-curand-cu12==10.3.2.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (10.3.2.106)
Requirement already satisfied: nvidia-cusolver-cu12==11.4.5.107 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (11.4.5.107)
Requirement already satisfied: nvidia-cusparse-cu12==12.1.0.106 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.0.106)
Requirement already satisfied: nvidia-nccl-cu12==2.20.5 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2.20.5)
Requirement already satisfied: nvidia-nvtx-cu12==12.1.105 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (12.1.105)
Requirement already satisfied: triton==2.3.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.0.0->bert-score) (2.3.0)
Requirement already satisfied: nvidia-nvjitlink-cu12 in /usr/local/lib/python3.10/dist-packages (from nvidia-cusolver-cu12==11.4.5.107->torch>=1.0.0->bert-score) (12.5.40)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.0 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.23.4)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (2024.5.15)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.19.1)
```

```
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers>=3.0.0->bert-score) (0.4.3)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (1.2.1)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (4.53.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (1.4.5)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (10.3.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib->bert-score) (3.1.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->bert-score) (2024.6.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas>=1.0.1->bert-score) (1.16.0)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.0.0->bert-score) (2.1.5)
Requirement already satisfied: mpmath<1.4.0,>=1.1.0 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.0.0->bert-score) (1.3.0)
Installing collected packages: bert-score
Successfully installed bert-score-0.3.13
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Antes de soldar verifique el estado de las conexiones.
Revise que la conexión de tierra del enchufe esté correcta.
Utilice ropa y herramientas apropiadas para evitar dañar la vista y la piel.
Cuando se este soldando se debe usar la máscara de soldar cubriendo toda la cabeza, sólo se puede realizar la observación visual del arco eléctrico a través del visor de la másc
Evite la sobrecarga de su equipo revisando previamente el ciclo de trabajo de este. Tome la precaución de que la manguera de gas nunca se encuentre presionada o doblada. Siempre
Asegúrese de que no exista riesgo potencial, tanto para el operador como para la máquina, de caída de cualquier objeto extraño.
El polvo, ácido o material corrosivo en el ambiente del lugar de trabajo y medio ambiente corrosivo no deben sobrepasar los parámetros exigidos (exceptuándose los provocados a c
El equipo de soldar debe instalarse protegido del sol, la lluvia, humedad excesiva o temperaturas por bajo los -10° C o por sobre los 40° C.
Deben existir al menos 50 cms. libres alrededor del equipo de soldar para asegurar una adecuada ventilación.
No debe introducir piezas extrañas al equipo de soldar.
No deben existir vibraciones excesivas en el área alrededor del lugar de trabajo de la máquina.
Elija un área sin interferencias electromagnéticas.
Asegúrese de instalar un interruptor automático de protección en la instalación eléctrica de acuerdo a las especificaciones indicadas por el manual de servicio del equipo.
El equipo de soldar debe instalarse horizontalmente, en una superficie plana.
Si la inclinación fuera superior a 15° se le debe adicionar elementos anti vuelco para evitar la inclinación del equipo.
No se recomienda conectar su equipo de soldar a un grupo generador.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:89: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens)
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
tokenizer_config.json: 100%  49.0/49.0 [00:00<00:00, 582B/s]
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_download.py:1132: FutureWarning: `resume_download` is depr
  warnings.warn(
config.json: 100%  625/625 [00:00<00:00, 6.31kB/s]
vocab.txt: 100%  996k/996k [00:00<00:00, 4.11MB/s]
tokenizer.json: 100%  1.96M/1.96M [00:00<00:00, 10.7MB/s]
model.safetensors: 100%  714M/714M [00:12<00:00, 65.2MB/s]
calculating scores...
computing bert embedding.
100%  1/1 [00:13<00:00, 13.40s/it]
computing greedy matching.
100%  1/1 [00:00<00:00, 7.50it/s]
done in 13.56 seconds, 0.07 sentences/sec
ROUGE-1 score: Score(precision=0.5682281059063137, recall=0.8611111111111112, fmeasure=0.6846625766871166)
ROUGE-2 score: Score(precision=0.4448979591836735, recall=0.6749226006191951, fmeasure=0.5362853628536286)
ROUGE-L score: Score(precision=0.3788187372708758, recall=0.5740740740740741, fmeasure=0.4564417177914111)
BERTScore - Precision: 0.802464485168457, Recall: 0.8259664177894592, F1: 0.8140459060668945
```

```
question = "¿Qué ocurre si sucede Shock Eléctrico?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
        "llm": {"generation_kwargs": {"max_tokens": 1000}},
    }
)

# La respuesta generada por el modelo se almacena en result["llm"]["replies"][0].content
response= result["llm"]["replies"][0].content
# Imprimir la respuesta generada
print(response)
```

```
Si sucede un shock eléctrico, se deben tomar las siguientes medidas:

1. No tocar a la persona accidentada si ha quedado en contacto con algún cuerpo posiblemente energizado.
2. Cortar el suministro eléctrico que alimenta el equipo.
3. Recurrir a los cuidados de primeros auxilios.
4. Para alejar los cables y/o partes energizadas de la víctima, se recomienda utilizar trozos de madera bien seca, como una escoba o cualquier otro material aislante.

Además, el lugar de trabajo debe contar con un botiquín de primeros auxilios equipado para socorrer en forma inmediata a posibles víctimas de un shock eléctrico y tra[illegible]os
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Si la persona accidentada esta inconsciente y se sospecha un shock eléctrico, tenga la precaución de
no tocarla si ha quedado en contacto con algún cuerpo posiblemente energizado.
Corte el suministro eléctrico que alimenta el equipo y recurra a los cuidados de primeros auxilios.
Para alejar los cables y/o partes energizadas de la víctima, se recomienda utilizar trozos de madera
bien seca, como una escoba o cualquier otro material aislante.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1  [00:01<00:00,  1.91s/it]
computing greedy matching.
100%                                                  1/1  [00:00<00:00, 19.80it/s]
done in 1.98 seconds, 0.51 sentences/sec
ROUGE-1 score: Score(precision=0.5811965811965812, recall=0.8947368421052632, fmeasure=0.7046632124352332)
ROUGE-2 score: Score(precision=0.47413793103448276, recall=0.7333333333333333, fmeasure=0.5759162303664922)
ROUGE-L score: Score(precision=0.5042735042735043, recall=0.7763157894736842, fmeasure=0.6113989637305699)
BERTScore - Precision: 0.8151604533195496, Recall: 0.8794638514518738, F1: 0.846092164516449
```

```
question = "¿Cuáles son los requerimientos para la fuente de poder?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
        "llm": {"generation_kwargs": {"max_tokens": 1000}},
    }
)

# La respuesta generada por el modelo se almacena en result["llm"]["replies"][0].content
response= result["llm"]["replies"][0].content
# Imprimir la respuesta generada
print(response)
```

```
Los requerimientos para la fuente de poder según el documento proporcionado son:

a) El oscilograma de voltaje debe mostrar una onda de seno, la oscilación de la frecuencia no debe exceder ± 1% del valor indicado.

b) La oscilación del voltaje no debe superar ± 10 % del valor indicado.
```

```
    c) El desbalance de la alimentación trifásica no debe exceder el 5%.
```

+ Código + Texto

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """El oscilograma de voltaje debe mostrar una onda de seno, la oscilación de la frecuencia no
debe exceder ± 1% del valor indicado. La oscilación del voltaje no debe superar ± 10 % del valor indicado.
El desbalance de la alimentación trifásica no debe exceder el 5%.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1 [00:00<00:00,  1.01it/s]
computing greedy matching.
100%                                                  1/1 [00:00<00:00, 13.27it/s]
done in 1.09 seconds, 0.91 sentences/sec
ROUGE-1 score: Score(precision=0.75, recall=1.0, fmeasure=0.8571428571428571)
ROUGE-2 score: Score(precision=0.7142857142857143, recall=0.9574468085106383, fmeasure=0.8181818181818182)
ROUGE-L score: Score(precision=0.75, recall=1.0, fmeasure=0.8571428571428571)
BERTScore - Precision: 0.8842079639434814, Recall: 0.961182713508606, F1: 0.9210899472236633
```

```
question = "¿Cuáles son las principales tareas de mantenimiento?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
        "llm": {"generation_kwargs": {"max_tokens": 1000}},
    }
)

# La respuesta generada por el modelo se almacena en result["llm"]["replies"][0].content
response= result["llm"]["replies"][0].content
# Imprimir la respuesta generada
print(response)
```

```
Las principales tareas de mantenimiento para el equipo de soldar son:

1. Mantener el cable de poder en buen estado: Chequear periódicamente el cable de alimentación del equipo de soldar y cada vez que la máquina sea utilizada de forma pe  cil
```

```
2. Revisar los cables de soldar: El operador debe revisar que los cables de soldar no estén desconectados, no presenten cortes en el aislamiento, etc. En caso de problemas,
3. Revisar el circuito de gas: Revisar regularmente que el circuito de gas no presente fugas y se mantenga completamente sellado. Verificar también que el ventilador y el mo
4. Limpiar la boquilla de escoria y suciedad: Limpiar periódicamente la boquilla de escoria y suciedad.
5. Utilizar alambre Mig de calidad: Utilizar alambre Mig de calidad, que cuente con certificaciones.
6. Reemplazar los rodillos si se encuentran gastados o deteriorados: Apretar los rodillos contra el alambre sin sobrepresionar para asegurar un buen desplazamiento de éste.

Además de estas tareas, es importante tomar precauciones de seguridad al realizar el mantenimiento, como no destapar el equipo si la tensión de salida se eleva a un nivel an
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = """Remoción de Polvo .Debe asegurarse que la máquina no esté conectada a una fuente de
poder antes de proceder a la remoción de polvo.
Asegurece que al dejar su equipo por un largo periodo de inactividad, lo haga en un ambiente seco, limpio y con buena ventilación.
Limpieza periódica. A través de personal especializado, se debe realizar limpieza por medio de aire comprimido seco para limpiar el interior del equipo de soldar. También se
debe revisar si existen componentes o cables sueltos, si los hubiera, proceda a la reparación o reemplazo correspondiente. Generalmente cuando no existe polvo en demasía la limp
se debe realizar periódicamente, y cuando existe mucha acumulación de polvo la limpieza se debe realizar con mayor frecuencia.Mantenga el cable de poder en buen estado Es necesa
de alimentación del equipo de soldar, sin embargo, es aconsejable revisar cada vez que la máquina sea utilizada de forma portátil.
Revise Cables de soldar. Al momento de utilizar el equipo de soldar, el operador debe revisar que los cables de soldar no estén desconectados, no presenten cortes en el
aislamiento, etc. En caso de que se presente alguno de estos problemas se debe proceder al cambio o reparación correspondiente.
Revise regularmente que el circuito de gas no presente fugas y se mantenga completamente sellado. Verifique ademas que el ventilador y el motor alimentador estén funcionando
correctamente sin sonidos anormales ni cables sueltos. Limpie periódicamente la boquilla de escoria y suciedad.
Utilice alambre Mig de calidad, que cuente con certificaciones. Reemplace los rodillos si se encuentran gastados o deteriorados. Apriete los rodillos contra
el alambre sin sobrepresionar para asegurar un buen desplazamiento de éste.""" # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1  [00:03<00:00,  3.35s/it]
computing greedy matching.
100%                                                  1/1  [00:00<00:00, 24.00it/s]
done in 3.40 seconds, 0.29 sentences/sec
ROUGE-1 score: Score(precision=0.7666666666666667, recall=0.6195286195286195, fmeasure=0.6852886405959031)
ROUGE-2 score: Score(precision=0.5481171548117155, recall=0.44256756756756754, fmeasure=0.4897196261682243)
ROUGE-L score: Score(precision=0.5833333333333334, recall=0.4713804713804714, fmeasure=0.521415270018622)
BERTScore - Precision: 0.7573366165161133, Recall: 0.7411555051803589, F1: 0.7491586804389954
```

```
question = "¿Antes de armar o desarmar la pistola que se debe hacer?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
        "llm": {"generation_kwargs": {"max_tokens": 1000}},
    }
)

# La respuesta generada por el modelo se almacena en result["llm"]["replies"][0].content
response= result["llm"]["replies"][0].content
# Imprimir la respuesta generada
print(response)
```

```
Antes de armar o desarmar la pistola, se debe desconectar de la energía eléctrica.
```

```
# para comparar
generated_response = str(response)  # La respuesta generada por el modelo
# La respuesta textual del manual técnico
reference_response = "Antes de armar o desarmar la pistola se la debe desconectar de la energía eléctrica." # La respuesta esperada

# Calcular ROUGE score
scorer = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], use_stemmer=True)
rouge_scores = scorer.score(reference_response, generated_response)

# Calcular BERTScore
P, R, F1 = score([generated_response], [reference_response], lang="es", verbose=True)

# Imprime resultados
print(f"ROUGE-1 score: {rouge_scores['rouge1']}")
print(f"ROUGE-2 score: {rouge_scores['rouge2']}")
print(f"ROUGE-L score: {rouge_scores['rougeL']}")
print(f"BERTScore - Precision: {P.mean().item()}, Recall: {R.mean().item()}, F1: {F1.mean().item()}")
```

```
calculating scores...
computing bert embedding.
100%                                                  1/1  [00:00<00:00,   2.20it/s]
computing greedy matching.
100%                                                  1/1  [00:00<00:00,  11.95it/s]
done in 0.56 seconds, 1.78 sentences/sec
ROUGE-1 score: Score(precision=1.0, recall=0.9411764705882353, fmeasure=0.9696969696969697)
ROUGE-2 score: Score(precision=0.9333333333333333, recall=0.875, fmeasure=0.9032258064516129)
ROUGE-L score: Score(precision=1.0, recall=0.9411764705882353, fmeasure=0.9696969696969697)
BERTScore - Precision: 0.9663508534431458, Recall: 0.9632884860038757, F1: 0.964817225933075
```

```
question = "¿Cuándo pueden resultar peligrosas las operaciones de soldadura?"

result = rag_pipeline.run(
    {
        "text_embedder": {"text": question},
        "prompt_builder": {"template_variables": {"query": question}, "prompt_source": messages},
```