In [1]:
```
pip install -q datasets sentence-transformers faiss-cpu accelerate
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 542.0/542.0 kB 5.6 M
B/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 171.5/171.5 kB 7.3 M
B/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 27.0/27.0 MB 11.8 M
B/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 302.6/302.6 kB 17.1
MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 11.6
MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.1/194.1 kB 8.6 M
B/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 134.8/134.8 kB 10.2
MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 401.2/401.2 kB 8.7 M
B/s eta 0:00:00
```

In [2]:
```
pip install -q bitsandbytes
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 119.8/119.8 MB 5.9 M
B/s eta 0:00:00
```

In [3]:
```
pip install pypdf
```

```
Collecting pypdf
  Downloading pypdf-4.2.0-py3-none-any.whl (290 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 290.4/290.4 kB 5.6 M
B/s eta 0:00:00
Requirement already satisfied: typing_extensions>=4.0 in /usr/loca
l/lib/python3.10/dist-packages (from pypdf) (4.11.0)
Installing collected packages: pypdf
Successfully installed pypdf-4.2.0
```

In [4]:
```
pip install -q langchain_community
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.1/2.1 MB 11.5 MB/s
eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 973.7/973.7 kB 19.1
MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 307.9/307.9 kB 13.2
MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 121.2/121.2 kB 4.9 M
B/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 49.3/49.3 kB 4.8 MB/
s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 53.0/53.0 kB 5.9 MB/
s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 142.5/142.5 kB 12.4
MB/s eta 0:00:00
```

In [5]:
```
pip install -U langchain-text-splitters
```

...

In [6]:
```
pip install langchain
```

...

In [7]:
```
pip install nltk rouge-score
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/d
ist-packages (3.8.1)
Collecting rouge-score
  Downloading rouge_score-0.1.2.tar.gz (17 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: click in /usr/local/lib/python3.10/
dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.1
0/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/p
ython3.10/dist-packages (from nltk) (2023.12.25)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/d
ist-packages (from nltk) (4.66.4)
Requirement already satisfied: absl-py in /usr/local/lib/python3.1
0/dist-packages (from rouge-score) (1.4.0)
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/pytho
n3.10/dist-packages (from rouge-score) (1.16.0)
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-no
ne-any.whl size=24933 sha256=b5d7a603bb93950d620fac57dc43dc3652e59
d993973eebb0dceaca6ba84ffb2
  Stored in directory: /root/.cache/pip/wheels/5f/dd/89/461065a73b
e61a532ff8599a28e9beef17985c9e9c31e541b4
Successfully built rouge-score
Installing collected packages: rouge-score
Successfully installed rouge-score-0.1.2
```

In [66]:
```
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsA
import torch
from langchain_community.document_loaders import PyPDFLoader
from langchain_community.embeddings import HuggingFaceEmbeddings
from langchain_community.vectorstores import FAISS
from torch import cuda
from transformers import StoppingCriteria, StoppingCriteriaList
import transformers
from langchain_community.llms import HuggingFacePipeline
from langchain_core.prompts import ChatPromptTemplate
from langchain.chains import ConversationalRetrievalChain, LLMChain
from tqdm import tqdm
import warnings
warnings.filterwarnings('ignore')
import nltk
from rouge_score import rouge_scorer
from nltk.translate.bleu_score import sentence_bleu
from statistics import mean
```

In [60]:
```
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

Out[60]:
```
True
```

In [9]:
```
#using opensource Llama3
model_id = "nvidia/Llama3-ChatQA-1.5-8B"

# use quantization to lower GPU usage (loading only quantized versi
bnb_config = BitsAndBytesConfig(
    load_in_4bit=True, bnb_4bit_use_double_quant=True, bnb_4bit_qua
)

#getting model from hugging face
hf_auth = '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'

tokenizer = AutoTokenizer.from_pretrained(model_id,use_auth_token=h

#initializing model
model = AutoModelForCausalLM.from_pretrained(
    model_id,
    torch_dtype=torch.bfloat16,
    device_map="auto",
    quantization_config=bnb_config
)


terminators = [
    tokenizer.eos_token_id,
    tokenizer.convert_tokens_to_ids("<|eot_id|>")
]
```

```
/usr/local/lib/python3.10/dist-packages/transformers/models/auto/t
okenization_auto.py:757: FutureWarning: The `use_auth_token` argum
ent is deprecated and will be removed in v5 of Transformers. Pleas
e use `token` instead.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_downl
oad.py:1132: FutureWarning: `resume_download` is deprecated and wi
ll be removed in version 1.0.0. Downloads always resume when possi
ble. If you want to force a new download, use `force_download=True
`.
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_tok
en.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), set it as s
ecret in your Google Colab and restart your session.
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:   0%|          | 0.00/51.3k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/9.08M [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/73.0 [00:00<?, ?B/
s]

Special tokens have been added in the vocabulary, make sure the as
sociated word embeddings are fine-tuned or trained.

config.json:   0%|          | 0.00/653 [00:00<?, ?B/s]

/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_downl
oad.py:1132: FutureWarning: `resume_download` is deprecated and wi
ll be removed in version 1.0.0. Downloads always resume when possi
ble. If you want to force a new download, use `force_download=True
`.
  warnings.warn(

model.safetensors.index.json:   0%|          | 0.00/28.1k [00:00
<?, ?B/s]

Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]

model-00001-of-00002.safetensors:   0%|          | 0.00/9.98G [00:
00<?, ?B/s]

model-00002-of-00002.safetensors:   0%|          | 0.00/6.08G [00:
00<?, ?B/s]

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

generation_config.json:   0%|          | 0.00/136 [00:00<?, ?B/s]
```

In [10]:
```
# seeting to evaluation mode
model.eval()
```

Out[10]:
```
LlamaForCausalLM(
  (model): LlamaModel(
    (embed_tokens): Embedding(128256, 4096)
    (layers): ModuleList(
      (0-31): 32 x LlamaDecoderLayer(
        (self_attn): LlamaSdpaAttention(
          (q_proj): Linear4bit(in_features=4096, out_features=409
6, bias=False)
          (k_proj): Linear4bit(in_features=4096, out_features=102
4, bias=False)
          (v_proj): Linear4bit(in_features=4096, out_features=102
4, bias=False)
          (o_proj): Linear4bit(in_features=4096, out_features=409
6, bias=False)
          (rotary_emb): LlamaRotaryEmbedding()
        )
        (mlp): LlamaMLP(
          (gate_proj): Linear4bit(in_features=4096, out_features=1
4336, bias=False)
          (up_proj): Linear4bit(in_features=4096, out_features=143
36, bias=False)
          (down_proj): Linear4bit(in_features=14336, out_features=
4096, bias=False)
          (act_fn): SiLU()
        )
        (input_layernorm): LlamaRMSNorm()
        (post_attention_layernorm): LlamaRMSNorm()
      )
    )
    (norm): LlamaRMSNorm()
  )
  (lm_head): Linear(in_features=4096, out_features=128256, bias=Fa
lse)
)
```

In [11]:
```
# reading the PDF
loader = PyPDFLoader("/content/Data_pdf.pdf")
pages = loader.load_and_split()
```

```
WARNING:pypdf._reader:Ignoring wrong pointing object 6 0 (offset
0)
WARNING:pypdf._reader:Ignoring wrong pointing object 8 0 (offset
0)
WARNING:pypdf._reader:Ignoring wrong pointing object 14 0 (offset
0)
```

In [12]:
```
#to understand the infromation of PDF
model_name = "sentence-transformers/all-mpnet-base-v2"
model_kwargs = {"device": "cuda"}

# loading from huggingface
embeddings = HuggingFaceEmbeddings(model_name=model_name, model_kwa

#to store vectors formed from the pdf word embeddings
vectorstore = FAISS.from_documents(pages, embeddings)
```

```
modules.json:   0%|          | 0.00/349 [00:00<?, ?B/s]

config_sentence_transformers.json:   0%|          | 0.00/116 [00:0
0<?, ?B/s]

README.md:   0%|          | 0.00/10.6k [00:00<?, ?B/s]

sentence_bert_config.json:   0%|          | 0.00/53.0 [00:00<?, ?
B/s]

/usr/local/lib/python3.10/dist-packages/huggingface_hub/file_downl
oad.py:1132: FutureWarning: `resume_download` is deprecated and wi
ll be removed in version 1.0.0. Downloads always resume when possi
ble. If you want to force a new download, use `force_download=True
`.
  warnings.warn(

config.json:   0%|          | 0.00/571 [00:00<?, ?B/s]

model.safetensors:   0%|          | 0.00/438M [00:00<?, ?B/s]

tokenizer_config.json:   0%|          | 0.00/363 [00:00<?, ?B/s]

vocab.txt:   0%|          | 0.00/232k [00:00<?, ?B/s]

tokenizer.json:   0%|          | 0.00/466k [00:00<?, ?B/s]

special_tokens_map.json:   0%|          | 0.00/239 [00:00<?, ?B/s]

1_Pooling/config.json:   0%|          | 0.00/190 [00:00<?, ?B/s]
```

In [13]:
```
# examples for end of prompt
stop_list = ['\nHuman:', '\n```\n']
stop_token_ids = [tokenizer(x)['input_ids'] for x in stop_list]
stop_token_ids
```

Out[13]:
```
[[198, 35075, 25], [198, 14196, 4077]]
```

In [14]:
```
# to use on gpus
device = f'cuda:{cuda.current_device()}' if cuda.is_available() els
stop_token_ids = [torch.LongTensor(x).to(device) for x in stop_toke
stop_token_ids
```

Out[14]:
```
[tensor([  198, 35075,    25], device='cuda:0'),
 tensor([  198, 14196,  4077], device='cuda:0')]
```

In [15]:
```
class StopOnTokens(StoppingCriteria):
    def __call__(self, input_ids: torch.LongTensor, scores: torch.F
        for stop_ids in stop_token_ids:
            if torch.eq(input_ids[0][-len(stop_ids):], stop_ids).al
                return True
        return False

# give stop tokens to generatetext
stopping_criteria = StoppingCriteriaList([StopOnTokens()])
```

In [72]:
```
generate_text = transformers.pipeline(
    model=model,
    tokenizer=tokenizer,
    return_full_text=True,  # langchain expects the full text
    task='text-generation',
    # we pass model parameters here too
    stopping_criteria=stopping_criteria,  # without this model ramb
    temperature=0.4,  # 'randomness' of outputs, 0.0 is the min and
    max_new_tokens=250,  # max number of tokens to generate in the
    repetition_penalty=1.1  # without this output begins repeating
)
```

In [73]:
```
# create an instance of a pipeline using the Hugging Face Transform
llm = HuggingFacePipeline(pipeline=generate_text)
```

In [74]:
```
# creating prompt template
template = """
<<SYS>>
You are an assistant for answering questions.
You are given the extracted parts of a long document and a question
If you don't know the answer, just say "I do not know." Don't make
<</SYS>>
[INST]
Question: {question}
[/INST]
"""
prompt = ChatPromptTemplate.from_template(template)

question_generator_chain = LLMChain(llm=llm, prompt=prompt)
```

In [77]:
```
chain = ConversationalRetrievalChain.from_llm(llm, vectorstore.as_r
```

In [78]:
```
# for answer extraction
def extract_helpful_answer(response):
    marker = "Helpful Answer:"
    marker_index = response.find(marker)
    if marker_index != -1:
        answer_start = marker_index + len(marker)
        answer = response[answer_start:].strip()
        return answer
    else:
        return "I do not know."
```

In [79]:
```
# generating response from LLM
def answer_rag(query):
  result = chain({"question": query, "chat_history": chat_history})
  llm_response = result['answer']
  extracted_answer = extract_helpful_answer(llm_response)
  return extracted_answer
```

In [80]:
```
questions = []
with open('questions.txt', 'r') as file:
    for line in tqdm(file):
        questions.append(line.strip())
```

```
20it [00:00, 26605.16it/s]
```

In [81]:
```
questions
```

Out[81]:
```
['What is the future outlook for online learning in higher educati
on?',
 'Are there specific accreditation bodies for online education?',
 'What documentation is required to enroll in online courses?',
 'How does online learning democratize education?',
 'What are the cost differences between online and traditional edu
cation?',
 'How do online courses address the issue of isolation?',
 'What challenges do students face with online learning?',
 'Can students transfer credits from online courses to traditional
programs?',
 'How do institutions ensure online courses are engaging?',
 'Are there financial aid options available for online educatio
n?',
 'How do online learning platforms support student interaction?',
 'What skills do students develop through online learning?',
 'Can online learning accommodate students with busy schedules?',
 'How do online courses handle student assessments?',
 'What types of courses are available through online learning plat
forms?',
 'Are online degrees as respected as traditional on-campus degree
s?',
 'How do online learning platforms ensure the quality of educatio
n?',
 'What technologies are essential for online learning?',
 'How has the COVID-19 pandemic impacted online learning?',
 'What are the main advantages of online learning in higher educat
ion?']
```

In [ ]:
```
answers = []
for question in tqdm(questions):
  answers.append(answer_rag(question))
```

In [84]:
```
gold_answers = []
with open('gold_answers.txt', 'r') as file:
    for line in file:
        gold_answers.append(line.strip())
```

In [85]:
```
# claculation rouge scores
rouge = rouge_scorer.RougeScorer(['rouge1', 'rouge2', 'rougeL'], us
```

In [86]:
```
rouge_scores = []
for ref, gen in zip(gold_answers, answers):
    scores = rouge.score(ref, gen)
    rouge_scores.append(scores)
```

In [87]:
```
# calculating bleu scores
bleu_scores = []
for ref, gen in zip(gold_answers, answers):
    ref_tokens = nltk.word_tokenize(ref)
    gen_tokens = nltk.word_tokenize(gen)
    score = sentence_bleu([ref_tokens], gen_tokens)
    bleu_scores.append(score)
```

In [88]:
```
# rouge and bleu scores for individual question
rouge1_precisions, rouge1_recalls, rouge1_fmeasures = [], [], []
rouge2_precisions, rouge2_recalls, rouge2_fmeasures = [], [], []
rougeL_precisions, rougeL_recalls, rougeL_fmeasures = [], [], []

# Extract and print individual scores
for i, (rouge_score, bleu_score) in enumerate(zip(rouge_scores, ble
    rouge1_precisions.append(rouge_score['rouge1'].precision)
    rouge1_recalls.append(rouge_score['rouge1'].recall)
    rouge1_fmeasures.append(rouge_score['rouge1'].fmeasure)

    rouge2_precisions.append(rouge_score['rouge2'].precision)
    rouge2_recalls.append(rouge_score['rouge2'].recall)
    rouge2_fmeasures.append(rouge_score['rouge2'].fmeasure)

    rougeL_precisions.append(rouge_score['rougeL'].precision)
    rougeL_recalls.append(rouge_score['rougeL'].recall)
    rougeL_fmeasures.append(rouge_score['rougeL'].fmeasure)

    print(f"FAQ {i + 1}:")
    print(f"  ROUGE-1: P={rouge_score['rouge1'].precision:.4f}, R={
    print(f"  ROUGE-2: P={rouge_score['rouge2'].precision:.4f}, R={
    print(f"  ROUGE-L: P={rouge_score['rougeL'].precision:.4f}, R={
    print(f"  BLEU: {bleu_score:.4f}")
```

```
FAQ 1:
  ROUGE-1: P=0.5000, R=0.3571, F=0.4167
  ROUGE-2: P=0.2222, R=0.1538, F=0.1818
  ROUGE-L: P=0.4000, R=0.2857, F=0.3333
  BLEU: 0.0000
FAQ 2:
  ROUGE-1: P=0.3043, R=0.5385, F=0.3889
  ROUGE-2: P=0.0909, R=0.1667, F=0.1176
  ROUGE-L: P=0.2609, R=0.4615, F=0.3333
  BLEU: 0.0000
FAQ 3:
  ROUGE-1: P=0.0465, R=0.1667, F=0.0727
  ROUGE-2: P=0.0000, R=0.0000, F=0.0000
  ROUGE-L: P=0.0465, R=0.1667, F=0.0727
  BLEU: 0.0000
FAQ 4:
  ROUGE-1: P=0.7222, R=0.8125, F=0.7647
  ROUGE-2: P=0.5882, R=0.6667, F=0.6250
  ROUGE-L: P=0.7222, R=0.8125, F=0.7647
  BLEU: 0.2915
```

```
FAQ 5:
  ROUGE-1: P=0.7778, R=1.0000, F=0.8750
  ROUGE-2: P=0.7647, R=1.0000, F=0.8667
  ROUGE-L: P=0.7778, R=1.0000, F=0.8750
  BLEU: 0.6676
FAQ 6:
  ROUGE-1: P=0.2381, R=0.5556, F=0.3333
  ROUGE-2: P=0.0500, R=0.1250, F=0.0714
  ROUGE-L: P=0.1905, R=0.4444, F=0.2667
  BLEU: 0.0000
FAQ 7:
  ROUGE-1: P=0.4545, R=0.5882, F=0.5128
  ROUGE-2: P=0.3333, R=0.4375, F=0.3784
  ROUGE-L: P=0.4091, R=0.5294, F=0.4615
  BLEU: 0.1665
FAQ 8:
  ROUGE-1: P=0.1449, R=0.7143, F=0.2410
  ROUGE-2: P=0.0588, R=0.3077, F=0.0988
  ROUGE-L: P=0.1159, R=0.5714, F=0.1928
  BLEU: 0.0000
FAQ 9:
  ROUGE-1: P=0.1905, R=0.3333, F=0.2424
  ROUGE-2: P=0.1000, R=0.1818, F=0.1290
  ROUGE-L: P=0.1905, R=0.3333, F=0.2424
  BLEU: 0.0000
FAQ 10:
  ROUGE-1: P=0.2250, R=1.0000, F=0.3673
  ROUGE-2: P=0.0769, R=0.3750, F=0.1277
  ROUGE-L: P=0.1750, R=0.7778, F=0.2857
  BLEU: 0.1192
FAQ 11:
  ROUGE-1: P=0.5556, R=0.9091, F=0.6897
  ROUGE-2: P=0.2941, R=0.5000, F=0.3704
  ROUGE-L: P=0.4444, R=0.7273, F=0.5517
  BLEU: 0.3327
FAQ 12:
  ROUGE-1: P=0.2581, R=1.0000, F=0.4103
  ROUGE-2: P=0.2333, R=1.0000, F=0.3784
  ROUGE-L: P=0.2581, R=1.0000, F=0.4103
  BLEU: 0.2215
FAQ 13:
  ROUGE-1: P=0.5909, R=0.7647, F=0.6667
  ROUGE-2: P=0.3333, R=0.4375, F=0.3784
  ROUGE-L: P=0.5455, R=0.7059, F=0.6154
  BLEU: 0.1625
FAQ 14:
  ROUGE-1: P=0.3333, R=0.9167, F=0.4889
  ROUGE-2: P=0.2500, R=0.7273, F=0.3721
  ROUGE-L: P=0.2424, R=0.6667, F=0.3556
  BLEU: 0.2603
FAQ 15:
  ROUGE-1: P=0.1053, R=0.5714, F=0.1778
  ROUGE-2: P=0.0133, R=0.0769, F=0.0227
  ROUGE-L: P=0.0921, R=0.5000, F=0.1556
```

```
  BLEU: 0.0000
FAQ 16:
  ROUGE-1: P=0.0909, R=0.2000, F=0.1250
  ROUGE-2: P=0.0000, R=0.0000, F=0.0000
  ROUGE-L: P=0.0455, R=0.1000, F=0.0625
  BLEU: 0.0000
FAQ 17:
  ROUGE-1: P=0.0385, R=0.0833, F=0.0526
  ROUGE-2: P=0.0000, R=0.0000, F=0.0000
  ROUGE-L: P=0.0385, R=0.0833, F=0.0526
  BLEU: 0.0000
FAQ 18:
  ROUGE-1: P=0.0985, R=0.7647, F=0.1745
  ROUGE-2: P=0.0305, R=0.2500, F=0.0544
  ROUGE-L: P=0.0833, R=0.6471, F=0.1477
  BLEU: 0.0000
FAQ 19:
  ROUGE-1: P=0.4500, R=0.5625, F=0.5000
  ROUGE-2: P=0.3158, R=0.4000, F=0.3529
  ROUGE-L: P=0.4500, R=0.5625, F=0.5000
  BLEU: 0.3060
FAQ 20:
  ROUGE-1: P=0.1750, R=0.3684, F=0.2373
  ROUGE-2: P=0.0769, R=0.1667, F=0.1053

  ROUGE-L: P=0.1000, R=0.2105, F=0.1356
  BLEU: 0.0605
```

In [89]:
```
# Print overall scores by calculating the averages
avg_rouge1_precision = mean(rouge1_precisions)
avg_rouge1_recall = mean(rouge1_recalls)
avg_rouge1_fmeasure = mean(rouge1_fmeasures)

avg_rouge2_precision = mean(rouge2_precisions)
avg_rouge2_recall = mean(rouge2_recalls)
avg_rouge2_fmeasure = mean(rouge2_fmeasures)

avg_rougeL_precision = mean(rougeL_precisions)
avg_rougeL_recall = mean(rougeL_recalls)
avg_rougeL_fmeasure = mean(rougeL_fmeasures)

avg_bleu = mean(bleu_scores)
print("\nOverall Scores:")
print(f"  ROUGE-1: Precision={avg_rouge1_precision:.4f}, Recall={av
print(f"  ROUGE-2: Precision={avg_rouge2_precision:.4f}, Recall={av
print(f"  ROUGE-L: Precision={avg_rougeL_precision:.4f}, Recall={av
print(f"  BLEU: {avg_bleu:.4f}")
```

```
Overall Scores:
  ROUGE-1: Precision=0.3150, Recall=0.6104, F1-Score=0.3869
  ROUGE-2: Precision=0.1916, Recall=0.3486, F1-Score=0.2315
  ROUGE-L: Precision=0.2794, Recall=0.5293, F1-Score=0.3408
  BLEU: 0.1294
```

In [ ]: