FYP Research

November 20, 2024

1 Pre-processing of Data

1. Extracting the raw questions, options, answers from the .js file
2. Splitting the data to use for research
3. Creation of the test cases:
   - Case 1: Wrong answer, wrong explanation
   - Case 2: Wrong answer, right explanation
   - Case 3: Right answer, wrong explanation
   - Case 4: Right answer, right explanation

1.1 1. Extracting the raw questions, options, answers

[7]:

```
import json
import pandas as pd

def clean_js_content(content):
    # Remove 'const' declarations and combine arrays
    content = content.replace('const', '')

    # Split the content into individual array declarations
    arrays = ['easy_arrays', 'medium_arrays', 'hard_arrays',
              'easy_stacksandqueues', 'medium_stacksandqueues',␣
 ↪'hard_stacksandqueues',
              'easy_linkedlist', 'medium_linkedlist', 'hard_linkedlists',
              'easy_recursion', 'medium_recursion', 'hard_recursion',
              'easy_trees', 'medium_trees', 'hard_trees',
              'easy_hashing', 'medium_hashing', 'hard_hashing',
              'easy_heaps', 'medium_heaps', 'hard_heaps',
              'easy_graphs', 'medium_graphs', 'hard_graphs']

    all_questions = []

    for array_name in arrays:
        try:
            # Find the start of the array
            start_idx = content.find(array_name + ' = [')
            if start_idx == -1:
                continue
            # Find the end of the array
            end_idx = content.find('\n]', start_idx)
            if end_idx == -1:
                continue

            # Extract the array content
            array_content = content[start_idx + len(array_name + ' = ['):end_idx + 1]

            # Convert the content to valid JSON format
            array_content = '[' + array_content.strip()
            array_content = array_content.replace('\n', '')
            array_content = array_content.replace('    ', '')

            # Add missing commas between objects
            array_content = array_content.replace('}{', '},{')

            # Add missing brackets if needed
            if not array_content.startswith('['):
                array_content = '[' + array_content
            if not array_content.endswith(']'):
                array_content = array_content + ']'

            # Fix missing commas between objects
            array_content = array_content.replace('"}{"', '"},{"')

            try:
                questions = json.loads(array_content)
                all_questions.extend(questions)
            except json.JSONDecodeError as e:
                print(f"Error parsing {array_name}: {e}")
                continue

        except Exception as e:
            print(f"Error processing {array_name}: {e}")
            continue
    return all_questions

# Read the JS file
with open('./data/questions.js', 'r') as file:
    content = file.read()

# Combine all questions into one list
combined_questions = clean_js_content(content)

# Now create the DataFrame
```

```python
df = pd.DataFrame({
    'Title': [q['title'] for q in combined_questions],
    'Question': [q['question'] for q in combined_questions],
    'Options': [q['options'] for q in combined_questions],
    'Answer': [q['options'][q['ans']] for q in combined_questions]
})

print(f"Total number of questions: {len(df)}")
print(df.head())
```

```
Total number of questions: 240
         Title                                           Question  \
0  Easy Arrays                                  What is an array?
1  Easy Arrays                How is memory allocated for arrays?
2  Easy Arrays       What does the Insert operation do in arrays?
3  Easy Arrays        How are Python lists different from arrays?
4  Easy Arrays  What is the purpose of initializing the size o…

                                             Options  \
0  [A collection of similar elements, A collectio…
1  [Memory is allocated randomly, Memory is alloc…
2  [Deletes an element, Searches for an element, …
3  [Python lists do not store values, Python list…
4  [To allocate memory, To delete elements, To se…

                                        Answer
0             A collection of similar elements
1         Memory is allocated at the beginning
2                        Inserts a new element
3  Python lists can store different data types
4                           To allocate memory
```

[45]:
```python
df.to_excel("questions.xlsx", index=False)
```

[8]:
```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 240 entries, 0 to 239
Data columns (total 4 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Title     240 non-null    object
 1   Question  240 non-null    object
 2   Options   240 non-null    object
 3   Answer    240 non-null    object
dtypes: object(4)
memory usage: 7.6+ KB
```

1.2 2. Splitting the data to use for research

[9]:

```
# Extracting first 5 rows of each Title for testing

first_per_title = df.groupby('Title').head(1)

# Sort by Title to keep it organized
first_per_title = first_per_title.sort_values('Title')

# Reset the index
first_per_title = first_per_title.reset_index(drop=True)

first_per_title['Title'].value_counts()
#
```

[9]:

```
Title
Easy Arrays                 1
Easy Graphs                 1
Medium Stacks and Queues    1
Medium Recursion            1
Medium Linked Lists         1
Medium Heaps                1
Medium Hashing              1
Medium Graphs               1
Medium Arrays               1
Hard Trees                  1
Hard Stacks and Queues      1
Hard Recursion              1
Hard Linked Lists           1
Hard Heaps                  1
Hard Hashing                1
Hard Graphs                 1
Hard Arrays                 1
Easy Trees                  1
Easy Stacks and Queues      1
Easy Recursion              1
Easy Linked Lists           1
Easy Heaps                  1
Easy Hashing                1
Medium Trees                1
Name: count, dtype: int64
```

[10]:

```
first_per_title.head()
```

[10]:

```
               Title                                          Question  \
0        Easy Arrays                                  What is an array?
1        Easy Graphs                    What is a minimum spanning tree?
2       Easy Hashing  What is the primary purpose of using hashing i…
```

```
3         Easy Heaps  What data structure are heaps almost always im…
4  Easy Linked Lists  What is the main advantage of using a linked l…

                                             Options  \
0  [A collection of similar elements, A collectio…
1  [A tree with the minimum number of edges from …
2  [To sort data efficiently, To store data in a …
3        [Linked lists, Arrays, Hash tables, Stacks]
4  [Constant time access to elements, Contiguous …

                                              Answer
0                   A collection of similar elements
1  A tree that minimizes the total weight of edge…
2            To quickly retrieve data based on a key
3                                             Arrays
4                                       Dynamic size
```

1.3 3. Creation of the test cases

[11]:

```python
from tqdm.notebook import tqdm
from langchain_core.prompts import ChatPromptTemplate
from langchain_core.output_parsers import StrOutputParser
from langchain_core.runnables import RunnablePassthrough

def generate_explanations(df, retriever, model):
    """
    Generate explanations for a DataFrame of questions.

    Parameters:
    -----------
    df : pandas.DataFrame
        DataFrame with columns 'Question', 'Options', 'Answer'
    retriever : Retriever
        LangChain retriever for context
    model : ChatModel
        Language model for generating explanations
    prompt_template : str
        Prompt template for explanation generation

    Returns:
    --------
    pandas.DataFrame
        DataFrame with added 'Explanation' column
    """
    template = """
    Answer the question based only on the following context:
```

```python
    {context}

    Question: {question}
    """
    prompt = ChatPromptTemplate.from_template(template)

    def format_docs(docs):
        return "\n\n".join([d.page_content for d in docs])

    # Create the chain
    # chain = (
    #     {"context": retriever | format_docs, "question": RunnablePassthrough()}
    #     | prompt
    #     | model
    #     | StrOutputParser()
    # )

    chain = (
        RunnablePassthrough()
        | model
        | StrOutputParser()
    )

    # Function to generate explanation for a single row
    def generate_explanation(row):
        text = f"""
        Given the question and the options:
        {row['Question']}
        {row['Options']}

        The correct answer is {row['Answer']}.

        Please do the following:
        1. Give an accurate explanation for the correct answer
        2. Choose one of the wrong answers
        3. Pretend you are a misguided student, create a plausible wrong explanation for the chosen wrong answer

        ## Sample output format in JSON respectively:
        "Correct explanation": "XXX", "Wrong chosen answer": "YYY", "Wrong explanation": "ZZZ"
        """

        try:
            explanation = chain.invoke(text)
            return explanation
        except Exception as e:
            print(f"Error generating explanation: {e}")
            return None

    # Apply the explanation generation to each row
    tqdm.pandas()
    df['Explanation'] = df.apply(generate_explanation, axis=1)

    return df
```

[13]:

```
import os
from dotenv import load_dotenv
from langchain_community.vectorstores.faiss import FAISS
from langchain_openai import AzureChatOpenAI
from langchain_openai.embeddings import AzureOpenAIEmbeddings

load_dotenv()

model = AzureChatOpenAI(
    azure_endpoint=os.environ['AZURE_OPENAI_ENDPOINT'],
    api_key=os.environ['AZURE_OPENAI_API_KEY'],
    deployment_name=os.environ['AZURE_OPENAI_DEPLOYMENT_NAME'],
    model_name=os.environ['AZURE_OPENAI_MODEL_NAME'],
    api_version=os.environ['AZURE_OPENAI_API_VERSION'],
    temperature=0
)

embedding_model = AzureOpenAIEmbeddings(azure_endpoint=os.
 ↪environ['AZURE_OPENAI_ENDPOINT'],
                                   api_key=os.environ['AZURE_OPENAI_API_KEY'],
                                   model=os.
 ↪environ['TEXT_EMBEDDING_MODEL_NAME'],
                                   azure_deployment=os.
 ↪environ['TEXT_EMBEDDING_DEPLOYMENT_NAME'])
docsearch = FAISS.load_local(folder_path="./embed", embeddings=embedding_model,␣
 ↪allow_dangerous_deserialization=True)
retriever = docsearch.as_retriever()

generate_explanations(first_per_title, retriever, model)
```

[13]:

```
                       Title  \
0                Easy Arrays
1                Easy Graphs
2               Easy Hashing
3                 Easy Heaps
4          Easy Linked Lists
5             Easy Recursion
```

```
6     Easy Stacks and Queues
7                 Easy Trees
8                Hard Arrays
9                Hard Graphs
10              Hard Hashing
11                Hard Heaps
12         Hard Linked Lists
13            Hard Recursion
14    Hard Stacks and Queues
15                Hard Trees
16             Medium Arrays
17             Medium Graphs
18            Medium Hashing
19              Medium Heaps
20       Medium Linked Lists
21          Medium Recursion
22  Medium Stacks and Queues
23              Medium Trees

                                             Question  \
0                                   What is an array?
1                    What is a minimum spanning tree?
2   What is the primary purpose of using hashing i…
3   What data structure are heaps almost always im…
4   What is the main advantage of using a linked l…
5     What is a characteristic of recursive routines?
6   What data structure follows the Last In First …
7   In a binary tree, what is the maximum number o…
8   Explain the difference between an unordered ar…
9   Explain the difference between a strong compon…
10  Explain the concept of a perfect hash function…
11  What is the time complexity of inserting N ite…
12  How does the time complexity of searching for …
13  What is the significance of memoization in rec…
14  In the context of a disaster scenario with lim…
15  Explain the concept of trinode restructuring i…
16  What is the time complexity of inserting an el…
17  In a directed graph, what is the term used to …
18  What is the difference between linear probing …
19  What is the time complexity of finding the K h…
20  What is the difference between a singly linked…
21  In recursion, what is the significance of the …
22  When folding rags to be used in cleaning, whic…
23  What is the time complexity for searching in a…

                                              Options  \
0   [A collection of similar elements, A collectio…
```

```
1   [A tree with the minimum number of edges from …
2   [To sort data efficiently, To store data in a …
3         [Linked lists, Arrays, Hash tables, Stacks]
4   [Constant time access to elements, Contiguous …
5   [They call themselves., Each call performs its…
6                  [Queue, Stack, Linked List, Array]
7                                        [0, 1, 2, 3]
8   [Unordered arrays have faster search operation…
9   [A strong component has all vertices connected…
10  [A perfect hash function maps all keys to uniq…
11               [O(N), O(log N), O(N log N), O(N^2)]
12  [Arrays have O(1) complexity, while linked lis…
13  [It ensures that the recursion depth is limite…
14  [Allows for random access of patients, Enables…
15  [Trinode restructuring involves restructuring …
16                     [O(0), O(log N), O(N), O(N^2)]
17                  [Cycle, Loop, Circuit, Traversal]
18  [Linear probing uses a fixed step size for pro…
19  [O(N + K^2), O(N × K), O(N + K × log N), O(N l…
20  [Singly linked lists allow traversal in one di…
21  [It stores local variables for each recursive …
22        [Queue, Stack, Linked List, Priority Queue]
23                     [O(1), O(log N), O(N), O(N^2)]

                                               Answer  \
0                    A collection of similar elements
1   A tree that minimizes the total weight of edge…
2             To quickly retrieve data based on a key
3                                              Arrays
4                                        Dynamic size
5                               They call themselves.
6                                               Stack
7                                                   2
8   Ordered arrays store elements in ascending or …
9   A strong component can be reached from any oth…
10  A perfect hash function maps all keys to uniqu…
11                                         O(N log N)
12  Arrays have O(1) complexity, while linked list…
13  It stores intermediate results to avoid redund…
14             Prioritizes patients based on severity
15  Trinode restructuring involves restructuring t…
16                                               O(0)
17                                              Cycle
18  Linear probing uses a fixed step size for prob…
19                                   O(N + K × log N)
20  Singly linked lists allow traversal in one dir…
21             It manages the order of function calls
```

```
22                                                Stack
23                                              O(log N)

                                             Explanation
0   {\n  "Correct explanation": "An array is a dat…
1   {\n  "Correct explanation": "A minimum spannin…
2   {\n  "Correct explanation": "Hashing is used i…
3   {\n  "Correct explanation": "Heaps are almost …
4   {\n  "Correct explanation": "The main advantag…
5   {\n  "Correct explanation": "A characteristic …
6   {\n  "Correct explanation": "A Stack is a data…
7   {\n  "Correct explanation": "In a binary tree,…
8   {\n  "Correct explanation": "Ordered arrays st…
9   {\n  "Correct explanation": "In a directed gra…
10  {\n  "Correct explanation": "A perfect hash fu…
11  {\n  "Correct explanation": "The time complexi…
12  {\n  "Correct explanation": "Arrays have O(1) …
13  {\n  "Correct explanation": "Memoization is a …
14  {\n  "Correct explanation": "In a disaster sce…
15  {\n  "Correct explanation": "Trinode restructu…
16  {\n  "Correct explanation": "The correct answe…
17  {\n  "Correct explanation": "In a directed gra…
18  {\n  "Correct explanation": "In hashing, linea…
19  {\n  "Correct explanation": "To find the K hig…
20  {\n  "Correct explanation": "A singly linked l…
21  {\n  "Correct explanation": "In recursion, the…
22  {\n  "Correct explanation": "A stack is a data…
23  {\n  "Correct explanation": "In a balanced bin…
```

[15]:

```
first_per_title.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 5 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Title        24 non-null     object
 1   Question     24 non-null     object
 2   Options      24 non-null     object
 3   Answer       24 non-null     object
 4   Explanation  24 non-null     object
dtypes: object(5)
memory usage: 1.1+ KB
```

[16]:

```
import re
def parse_explanation(explanation_str):
    try:
        # Extract Wrong explanation
        correct_explanation_match = re.search(r'"Correct explanation":\s*"(.*?
 ↪)"', explanation_str, re.DOTALL)
        correct_explanation = correct_explanation_match.group(1) if␣
 ↪correct_explanation_match else ''

        # Extract Right explanation
        wrong_answer_match = re.search(r'"Wrong chosen answer":\s*"(.*?)"',␣
 ↪explanation_str, re.DOTALL)
        wrong_answer = wrong_answer_match.group(1) if wrong_answer_match else ''

        wrong_explanation_match = re.search(r'"Wrong explanation":\s*"(.*?)"',␣
 ↪explanation_str, re.DOTALL)
        wrong_explanation = wrong_explanation_match.group(1) if␣
 ↪wrong_explanation_match else ''

        return pd.Series({
            'Correct Explanation': correct_explanation,
            'Wrong Answer': wrong_answer,
            'Wrong Explanation': wrong_explanation
        })
    except Exception as e:
        print(f"Error parsing explanation: {e}")
        return pd.Series({
            'Correct Explanation': '',
            'Wrong Answer': '',
            'Wrong Explanation': ''
        })

# Apply the parsing to your DataFrame
first_per_title[['Correct Explanation', 'Wrong Answer', 'Wrong Explanation']] =␣
 ↪first_per_title['Explanation'].apply(parse_explanation)
first_per_title.head()
```

```
[16]:               Title                                          Question  \
      0        Easy Arrays                                  What is an array?
      1        Easy Graphs                    What is a minimum spanning tree?
      2       Easy Hashing  What is the primary purpose of using hashing i…
      3         Easy Heaps  What data structure are heaps almost always im…
      4  Easy Linked Lists  What is the main advantage of using a linked l…

                                                   Options  \
      0  [A collection of similar elements, A collectio…
      1  [A tree with the minimum number of edges from …
      2  [To sort data efficiently, To store data in a …
      3        [Linked lists, Arrays, Hash tables, Stacks]
      4  [Constant time access to elements, Contiguous …
```

```
                                              Answer  \
0                  A collection of similar elements
1  A tree that minimizes the total weight of edge…
2            To quickly retrieve data based on a key
3                                             Arrays
4                                       Dynamic size

                                         Explanation  \
0  {\n  "Correct explanation": "An array is a dat…
1  {\n  "Correct explanation": "A minimum spannin…
2  {\n  "Correct explanation": "Hashing is used i…
3  {\n  "Correct explanation": "Heaps are almost …
4  {\n  "Correct explanation": "The main advantag…

                                 Correct Explanation  \
0  An array is a data structure that can hold mul…
1  A minimum spanning tree (MST) is a subset of t…
2  Hashing is used in data structures to quickly …
3  Heaps are almost always implemented as arrays …
4  The main advantage of using a linked list over…

                                        Wrong Answer  \
0                 A collection of different elements
1  A tree that spans all the vertices using the l…
2                           To sort data efficiently
3                                       Linked lists
4                   Constant time access to elements

                                   Wrong Explanation
0  An array is a versatile data structure that ca…
1  A minimum spanning tree is a tree that spans a…
2  Hashing is used to sort data efficiently becau…
3  Heaps are implemented as linked lists because …
4  Linked lists provide constant time access to e…
```

Creating test cases where each question can be split into 4 different scenarios - Case 1: Wrong answer, wrong explanation - Case 2: Wrong answer, right explanation - Case 3: Right answer, wrong explanation - Case 4: Right answer, right explanation

[17]:

```
def split_row(row):
    return [
        # Wrong answer, wrong explanation
        {'Title': row['Title'],
         'Question': f"""
        Given the question and the options:
        {row['Question']}
        {row['Options']}
        The correct answer is {row['Answer']}

        I chose the wrong answer {row['Wrong Answer']} as I think that {row['Wrong Explanation']}.

        Please correct any conceptual misunderstanding I have based on my explanation and explain to me why my answer is wrong.
        ## Sample format:
        Your answer is wrong. Your understanding is wrong as...
        """},

        # Wrong answer, correct explanation
        {'Title': row['Title'],
         'Question': f"""
        Given the question and the options:
        {row['Question']}
        {row['Options']}
        The correct answer is {row['Answer']}

        I chose the wrong answer {row['Wrong Answer']} as I think that {row['Correct Explanation']}.

        Please correct any conceptual misunderstanding I have based on my explanation and explain to me why my answer is wrong.
        ## Sample format:
        Your answer is wrong. Your understanding is wrong as...
        """},

        # Correct answer, wrong explanation
        {'Title': row['Title'],
         'Question': f"""
        Given the question and the options:
        {row['Question']}
        {row['Options']}
        The correct answer is {row['Answer']}

        I chose the correct answer {row['Answer']} as I think that {row['Wrong Explanation']}.

        Please correct any conceptual misunderstanding I have based on my explanation.
        ## Sample format:
        Your answer is correct. Your understanding is partially correct as...
        """},

        # Correct answer, correct explanation
        {'Title': row['Title'],
         'Question': f"""
        Given the question and the options:
        {row['Question']}
        {row['Options']}
        The correct answer is {row['Answer']}

        I chose the correct answer {row['Answer']} as I think that␣
 ↪{row['Correct Explanation']}.

        Please correct any conceptual misunderstanding I have based on my␣
 ↪explanation.
        ## Sample format:
        Your answer is correct. Your understanding is partially correct as...
        """},
    ]
```

[18]:

```
expanded_data = first_per_title.apply(split_row, axis=1)

# Flatten the list of lists into a single list of dictionaries
flattened_data = [item for sublist in expanded_data for item in sublist]

test_df = pd.DataFrame(flattened_data)
test_df
```

[18]:

```
                       Title  \
0                Easy Arrays
1                Easy Arrays
2                Easy Arrays
3                Easy Arrays
4                Easy Graphs
..                        …
91  Medium Stacks and Queues
92              Medium Trees
93              Medium Trees
94              Medium Trees
95              Medium Trees

                                             Question
0   \n        Given the question and the options:\…
1   \n        Given the question and the options:\…
2   \n        Given the question and the options:\…
3   \n        Given the question and the options:\…
4   \n        Given the question and the options:\…
..                                                …
91  \n        Given the question and the options:\…
92  \n        Given the question and the options:\…
```

```
93  \n        Given the question and the options:\…
94  \n        Given the question and the options:\…
95  \n        Given the question and the options:\…

[96 rows x 2 columns]
```

2 Testing on GPT4o

Exploring how changing the chunking of the retriever documents will impact the accuracy of the model results, and finding the optimal

- Test 1: Chunk_size = 1000, chunk_overlap = 0
- Test 2: Chunk size = 2000, chunk_overlap = 0
- Test 3: Chunk size = 2000, chunk_overlap = 100
- Test 4: Chunk size = 1000, chunk_overlap = 100

Each answer is given a score and then the scores are compared at the end

2.1 Vector store creation script

[58]:

```
# Creating function to allow for creation of vector stores with varying chunk␣
 ↪sizes and chunk overlaps
import os
import re
from dotenv import load_dotenv
from PyPDF2 import PdfReader
from langchain.text_splitter import RecursiveCharacterTextSplitter
from langchain_openai.embeddings import AzureOpenAIEmbeddings
from langchain_community.vectorstores import FAISS
from langchain_openai import OpenAIEmbeddings

load_dotenv()

def create_embeddings_from_pdf(pdf_path, embedding_path, chunk_size,␣
 ↪chunk_overlap):
    # Read PDF
    pdf_reader = PdfReader(pdf_path)

    # Extract text from all pages
    text = ""
    for page in pdf_reader.pages:
        text += page.extract_text()

    # Clean text
    text = re.sub("\s+", " ", text).strip()

    # Create text splitter
    text_splitter = RecursiveCharacterTextSplitter(
        chunk_size=chunk_size,
        chunk_overlap=chunk_overlap,
        length_function=len,
    )

    # Split text into documents
    documents = text_splitter.create_documents([text])
    print(f"Total documents created: {len(documents)}")

    # Initialize embeddings
    # embeddings = AzureOpenAIEmbeddings(
    #     azure_endpoint=os.environ['AZURE_OPENAI_ENDPOINT'],
    #     api_key=os.environ['AZURE_OPENAI_API_KEY'],
    #     model=os.environ['TEXT_EMBEDDING_MODEL_NAME'],
    #     azure_deployment=os.environ['TEXT_EMBEDDING_DEPLOYMENT_NAME']
    # )
    embeddings = OpenAIEmbeddings(model="text-embedding-ada-002")

    # Create and save vector store
    try:
        # Use create_documents method to maintain metadata
        docsearch = FAISS.from_documents(documents, embedding=embeddings)

        # Ensure embedding path exists
        os.makedirs(embedding_path, exist_ok=True)

        # Save locally
        docsearch.save_local(folder_path=embedding_path)
        print(f"Embeddings saved to {embedding_path}")

        return docsearch

    except Exception as e:
        print(f"Error creating embeddings: {e}")
        return None
```

2.2 Starting the tests...

2.3 Test 1

Chunk_size = 1000, chunk_overlap = 0

Vector store with those specifications are created first, before undergoing inference and evaluation

[104]:

```
from tqdm import tqdm

# Enable progress bar for pandas
tqdm.pandas()
```

[6]:

```
# Creating the vector store for Test 1

pdf_path = './data/(edited) DSA textbook Python.pdf'
embedding_path = './embeddings-test1'
vector_store = create_embeddings_from_pdf(pdf_path, embedding_path, 1000, 0)

if vector_store != None:
    print(f"Vector store {pdf_path} for Test 1 to {embedding_path} is a success!
  ↪")
```

```
Total documents created: 1431
Embeddings saved to ./embeddings-test1
Vector store ./data/(edited) DSA textbook Python.pdf for Test 1 to ./embeddings-
test1 is a success!
```

2.3.1 One-time evaluation set-up using LangSmith (LLM-as-a-Judge)

[23]:

```
# Creating the dataset (without labels/aka. reference/aka. ground truth data)
from langsmith import Client

# QA
inputs = test_df["Question"].tolist()

# outputs =
# qa_pairs = [{"question": q, "answer": a} for q, a in zip(inputs, outputs)]
qa_pairs = [{"question": q} for q in test_df["Question"]]

# Create dataset
client = Client()
dataset_name = "Algotutor_MainDataset"
dataset = client.create_dataset(
    dataset_name=dataset_name,
    description="Testset for optimising retrievers",
)
client.create_examples(
    inputs=[{"question": q} for q in inputs],
    #outputs=[{"answer": a} for a in outputs],
    dataset_id=dataset.id,
)
```

[24]:

```
### RAG
import os
import openai
from openai import AzureOpenAI
from langsmith import traceable
from langsmith.wrappers import wrap_openai
```

```python
class RagBot:

    def __init__(self, retriever, model: str = 'gpt-4o'):
        self._retriever = retriever
        # Wrapping the client instruments the LLM
        # self._client = wrap_openai(AzureOpenAI(
        #     azure_endpoint=os.environ['AZURE_OPENAI_ENDPOINT'],
        #     api_key=os.environ['AZURE_OPENAI_API_KEY'],
        #     api_version=os.environ['AZURE_OPENAI_API_VERSION']
        # ))
        self._client = wrap_openai(openai.Client())
        self._model = model

    @traceable()
    def retrieve_docs(self, question):
        return self._retriever.similarity_search(question, k=2)

    @traceable()
    def get_answer(self, question: str):
        similar = self.retrieve_docs(question)
        response = self._client.chat.completions.create(
            model=self._model,
            messages=[
                {
                    "role": "system",
                    "content": "You are a teaching assistant. Your task is to␣
 ↪answer student query about Data Structures and Algorithms in Python course.␣
 ↪If user asks any query beyond data structures and algorithms, tell the user␣
 ↪you are not an expert of the topic."
                    "Answer the question based only on the following context:
 ↪\n\n"
                    f"Context:\n\n{similar}",
                },
                {"role": "user", "content": question},
            ],
            temperature=0
        )

        # Evaluators will expect "answer" and "contexts"
        return {
            "answer": response.choices[0].message.content,
            "contexts": [str(doc) for doc in similar],
        }
```

[50]:

```python
from langsmith.evaluation import LangChainStringEvaluator, evaluate
import textwrap
```

```
# Checking whether the answer is accurate to the docs retrieved
# answer_accuracy_evaluator = LangChainStringEvaluator(
#   "labeled_score_string",
#   config={
#       "criteria": {
#           "accuracy": textwrap.dedent("""Is the Assistant's Answer grounded␣
 ↪in the Ground Truth documentation? A score of [[1]] means that the
#             Assistant answer contains is not at all based upon / grounded in␣
 ↪the Groun Truth documentation. A score of [[5]] means
#             that the Assistant answer contains some information (e.g., a␣
 ↪hallucination) that is not captured in the Ground Truth
#             documentation. A score of [[10]] means that the Assistant answer␣
 ↪is fully based upon the in the Ground Truth documentation."""
#           )
#       },
#       # If you want the score to be saved on a scale from 0 to 1
#       "normalize_by": 10,
#   },
#   prepare_data=lambda run, example: {
#         "prediction": run.outputs["answer"],
#         "reference": run.outputs["contexts"],
#         "input": example.inputs["question"],
#     },
# )

# Checking whether the retrieved documents are relevant to question
docs_relevance_evaluator = LangChainStringEvaluator(
    "score_string",
    config={
        "criteria": {
            "document_relevance": textwrap.dedent(
                """The response is a set of documents retrieved from a␣
 ↪vectorstore. The input is a question
            used for retrieval. You will score whether the Assistant's response␣
 ↪(retrieved docs) is relevant to the Ground Truth
            question. A score of [[1]] means that none of the Assistant's␣
 ↪response documents contain information useful in answering or addressing the␣
 ↪user's input.
            A score of [[5]] means that the Assistant answer contains some␣
 ↪relevant documents that can at least partially answer the user's question or␣
 ↪input.
            A score of [[10]] means that the user input can be fully answered␣
 ↪using the content in the first retrieved doc(s)."""
            )
        },
        # If you want the score to be saved on a scale from 0 to 1
        "normalize_by": 10
    },
    prepare_data=lambda run, example: {
        "prediction": run.outputs["contexts"],
        "input": example.inputs["question"],
    },
)
```

This chain was only tested with GPT-4. Performance may be significantly worse with other models.

2.3.2 Evaluation on Test 1 using first retriever

[25]:

```
from langchain_openai.embeddings import AzureOpenAIEmbeddings
from langchain_community.vectorstores.faiss import FAISS

embedding_model = AzureOpenAIEmbeddings(azure_endpoint=os.
 ↪environ['AZURE_OPENAI_ENDPOINT'],
                                   api_key=os.environ['AZURE_OPENAI_API_KEY'],
                                   model=os.
 ↪environ['TEXT_EMBEDDING_MODEL_NAME'],
                                   azure_deployment=os.
 ↪environ['TEXT_EMBEDDING_DEPLOYMENT_NAME'])

docsearch = FAISS.load_local("./embeddings-test1", embeddings=embedding_model,␣
 ↪allow_dangerous_deserialization=True)

rag_bot = RagBot(docsearch)

def predict_rag_answer_with_context(example: dict):
    """Use this for evaluation of retrieved documents and hallucinations"""
    response = rag_bot.get_answer(example["question"])
    return {"answer": response["answer"], "contexts": response["contexts"]}
```

[28]:

```
# Dataset has been split on LangSmith interface into 10 splits
# This is to bypass the token restriction on the OpenAI API calls

from time import sleep

for setnumber in range(1, 11):
    dataset_name = "Algotutor_MainDataset"
    experiment_results = evaluate(
        predict_rag_answer_with_context,
        data=client.list_examples(dataset_name=dataset_name,␣
 ↪splits=[f"set{setnumber}"]),
        evaluators=[docs_relevance_evaluator],
        experiment_prefix=f"rag-chunk1000-overlap0-doc-relevance-{setnumber}",
        # Any experiment metadata can be specified here
        metadata={
            "variant": "chunk_size=1000, chunk_overlap=0",
        },
    )
    print(f"Datasplit set{setnumber} evaluation completed")

    # Sleep in order to bypass the minute rate limit on the token calls
    if setnumber < 10:
        sleep(60)
```

```
/root/algotutor-fyp/venv/lib/python3.11/site-packages/tqdm/auto.py:21:
TqdmWarning: IProgress not found. Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
```

```
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-1-1aab139f' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=853581aa-555e-4eca-a322-25194d6372e3
```

```
10it [00:13,  1.34s/it]
```

```
Datasplit set1 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-2-ee11b364' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=c27acea5-7f7e-40c8-8ec4-4a5abca8556f
```

```
10it [00:44,  4.47s/it]
```

```
Datasplit set2 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-3-b3775221' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=321c6129-9235-40f3-92d6-76db5bcd65a7
```

```
10it [00:20,  2.02s/it]
```

```
Datasplit set3 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-4-154aa696' at:
```

```
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=da78d194-6629-498f-8a0e-ae175de3d451



10it [00:16,  1.69s/it]

Datasplit set4 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-5-f265f1b9' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=8ad6dac5-612a-4489-a0ae-869e0b6abc9c



10it [00:18,  1.85s/it]

Datasplit set5 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-6-1b2e4b31' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=559a792a-0872-428e-84ff-2f913192971c



10it [00:21,  2.11s/it]

Datasplit set6 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-7-b582aef2' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=feff2324-e68a-4759-a144-2554d03aff14



10it [00:13,  1.40s/it]

Datasplit set7 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-8-7d496f3f' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=aba28f6b-76ee-4e0e-a85a-20e3c849ac52



10it [00:16,  1.64s/it]
```

```
Datasplit set8 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-9-ef53b19a' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=fe57c17a-961f-47a9-85d1-07dff0d6cc5b
```

```
10it [00:16,  1.64s/it]
```

```
Datasplit set9 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap0-doc-
relevance-10-0aad8cb5' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/26ec1cbb-8079-4a3a-bd80-
1141f080e39f/compare?selectedSessions=b03f93b9-73c9-49bc-bead-60e816c537b2
```

```
6it [00:12,  2.05s/it]
```

```
Datasplit set10 evaluation completed
```

[30]:

```python
project_names = ['rag-chunk1000-overlap0-doc-relevance-1-1aab139f',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-2-ee11b364',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-3-b3775221',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-4-154aa696',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-5-f265f1b9',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-6-1b2e4b31',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-7-b582aef2',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-8-7d496f3f',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-9-ef53b19a',␣
  ↪'rag-chunk1000-overlap0-doc-relevance-10-0aad8cb5']

#for projectidx in range(len(project_names)):

all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for␣
  ↪projectidx in range(len(project_names))]
combined_df = pd.concat(all_dfs, ignore_index=True)

combined_df.head()
```

```
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
/tmp/ipykernel_678/4090098211.py:5: UserWarning: Function get_test_results is in
beta.
  all_dfs = [client.get_test_results(project_name=project_names[projectidx]) for
projectidx in range(len(project_names))]
```

```
[30]:                                       outputs.answer  \
      0  Your answer is incorrect. Your understanding o…
      1  Your answer is correct. Your understanding is …
      2  Your answer is correct. Your understanding is …
      3  Your answer is correct. Your understanding is …
      4  Your answer is wrong. Your understanding is co…

                                      outputs.contexts  execution_time error  \
      0  [page_content='configuration of the Queue Visu…        4.325442  None
      1  [page_content='configuration of the Queue Visu…        3.903861  None
      2  [page_content='you can see, the time required …        4.223810  None
      3  [page_content='trickiest parts isremembering w…        3.491941  None
      4  [page_content='configuration of the Queue Visu…        5.009560  None
```

```
                                     id  \
0  25e8b861-0d1f-40d8-8e68-b69fb89ae9ba
1  2c6860b7-53cb-47df-9a33-975acf198691
2  ec0b6271-e169-4056-b798-f79e8b49ebf7
3  b5f5fda2-49fd-4d24-90af-05a951f43166
4  9f8e1409-e856-4195-b487-f8d922819f4d

   feedback.score_string:document_relevance  \
0                                       0.5
1                                       0.5
2                                       1.0
3                                       0.5
4                                       0.5

                               input.example.question
0  \n        Given the question and the options:\…
1  \n        Given the question and the options:\…
2  \n        Given the question and the options:\…
3  \n        Given the question and the options:\…
4  \n        Given the question and the options:\…
```

[32]:
```
combined_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 96 entries, 0 to 95
Data columns (total 7 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   outputs.answer                            96 non-null     object
 1   outputs.contexts                          96 non-null     object
 2   execution_time                            96 non-null     float64
 3   error                                     0 non-null      object
 4   id                                        96 non-null     object
 5   feedback.score_string:document_relevance  96 non-null     float64
 6   input.example.question                    96 non-null     object
dtypes: float64(2), object(5)
memory usage: 5.4+ KB
```

[36]:
```
# Merge the two DataFrames on the matching columns
merge_df = pd.merge(
    combined_df, test_df,
    left_on="input.example.question",
    right_on="Question",
    how="inner"  # Use "inner" to include only matching rows
)

desired_column_order = [
    "id",
    "Title",
    "input.example.question",
    "outputs.answer",
    "outputs.contexts",
    "feedback.score_string:document_relevance",
    "execution_time",
    "error"
]

# Reorder the DataFrame columns
results_df = merge_df[desired_column_order]

new_column_names = {
    "id": "ID",
    "Title": "Title",
    "input.example.question": "Question",
    "outputs.answer": "Model Answer",
    "outputs.contexts": "Retrieved Context",
    "feedback.score_string:document_relevance": "Relevance Score",
    "execution_time": "Execution Time",
    "error": "Error"
}

results_df.rename(columns=new_column_names, inplace=True)
```

```
/tmp/ipykernel_678/3690702879.py:34: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  results_df.rename(columns=new_column_names, inplace=True)
```

[37]:

```
results_df.head()
```

[37]:

```
                                     ID                     Title  \
0  25e8b861-0d1f-40d8-8e68-b69fb89ae9ba  Medium Stacks and Queues
1  2c6860b7-53cb-47df-9a33-975acf198691  Medium Stacks and Queues
2  ec0b6271-e169-4056-b798-f79e8b49ebf7              Medium Trees
3  b5f5fda2-49fd-4d24-90af-05a951f43166          Medium Recursion
4  9f8e1409-e856-4195-b487-f8d922819f4d  Medium Stacks and Queues

                                            Question  \
0  \n        Given the question and the options:\…
1  \n        Given the question and the options:\…
2  \n        Given the question and the options:\…
3  \n        Given the question and the options:\…
4  \n        Given the question and the options:\…
```

```
                                        Model Answer  \
0  Your answer is incorrect. Your understanding o…
1  Your answer is correct. Your understanding is …
2  Your answer is correct. Your understanding is …
3  Your answer is correct. Your understanding is …
4  Your answer is wrong. Your understanding is co…

                                   Retrieved Context  Relevance Score  \
0  [page_content='configuration of the Queue Visu…              0.5
1  [page_content='configuration of the Queue Visu…              0.5
2  [page_content='you can see, the time required …              1.0
3  [page_content='trickiest parts isremembering w…              0.5
4  [page_content='configuration of the Queue Visu…              0.5

   Execution Time Error
0        4.325442  None
1        3.903861  None
2        4.223810  None
3        3.491941  None
4        5.009560  None
```

[38]:
```
results_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 96 entries, 0 to 95
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   ID                 96 non-null     object
 1   Title              96 non-null     object
 2   Question           96 non-null     object
 3   Model Answer       96 non-null     object
 4   Retrieved Context  96 non-null     object
 5   Relevance Score    96 non-null     float64
 6   Execution Time     96 non-null     float64
 7   Error              0 non-null      object
dtypes: float64(2), object(6)
memory usage: 6.1+ KB
```

2.3.3 Score for Test 1

Calculating the score by taking the mean of all the relevance scores for each question/retrieved context pair

[40]:
```
print(f"Score for Test 1 retriever: {results_df['Relevance Score'].mean()}")
```

```
Score for Test 1 retriever: 0.7145833333333332
```

2.4 Test 2

Chunk_size = 2000, chunk_overlap = 0

Vector store with those specifications are created first, before undergoing inference and evaluation

[46]:

```python
# Creating the vector store for Test 1

pdf_path = './data/(edited) DSA textbook Python.pdf'
embedding_path = './embeddings-test2'
vector_store = create_embeddings_from_pdf(pdf_path, embedding_path, 2000, 0)

if vector_store != None:
    print(f"Vector store {pdf_path} for Test 2 to {embedding_path} is a success!
 ↪")
```

```
Total documents created: 715
Embeddings saved to ./embeddings-test2
Vector store ./data/(edited) DSA textbook Python.pdf for Test 2 to ./embeddings-
test2 is a success!
```

2.4.1 Evaluation on Test 2 using second retriever

[48]:

```python
from langchain_openai.embeddings import AzureOpenAIEmbeddings
from langchain_community.vectorstores.faiss import FAISS

embedding_model = AzureOpenAIEmbeddings(azure_endpoint=os.
 ↪environ['AZURE_OPENAI_ENDPOINT'],
                                   api_key=os.environ['AZURE_OPENAI_API_KEY'],
                                   model=os.
 ↪environ['TEXT_EMBEDDING_MODEL_NAME'],
                                   azure_deployment=os.
 ↪environ['TEXT_EMBEDDING_DEPLOYMENT_NAME'])

docsearch = FAISS.load_local("./embeddings-test2", embeddings=embedding_model,␣
 ↪allow_dangerous_deserialization=True)

rag_bot = RagBot(docsearch)

def predict_rag_answer_with_context(example: dict):
    """Use this for evaluation of retrieved documents and hallucinations"""
    response = rag_bot.get_answer(example["question"])
    return {"answer": response["answer"], "contexts": response["contexts"]}
```

[52]:

```python
# Dataset has been split on LangSmith interface into 20 splits
# This is to bypass the token restriction on the OpenAI API calls

from time import sleep
```

```python
for setnumber in range(1, 21):
    dataset_name = "Algotutor_Dataset_20split"
    experiment_results = evaluate(
        predict_rag_answer_with_context,
        data=client.list_examples(dataset_name=dataset_name, splits=[f"set{setnumber}"]),
        evaluators=[docs_relevance_evaluator],
        experiment_prefix=f"rag-chunk2000-overlap0-doc-relevance-{setnumber}",
        # Any experiment metadata can be specified here
        metadata={
            "variant": "chunk_size=2000, chunk_overlap=0",
        },
    )
    print(f"Datasplit set{setnumber} evaluation completed")

    # Sleep in order to bypass the minute rate limit on the token calls
    if setnumber < 20:
        sleep(30)
```

```
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-1-f68e44f4' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=c67bb198-6bc0-45b2-b254-7a3a559fd558



5it [00:15,  3.03s/it]

Datasplit set1 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-2-8c3ce496' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=14aca239-b155-4c9c-9a37-e213553a5866



5it [00:11,  2.32s/it]

Datasplit set2 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-3-ef140a31' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=09cff7d1-20bb-4bcd-95b9-f49211700f73



5it [00:14,  2.82s/it]
```

```
Datasplit set3 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-4-ae6d3aee' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=f8ad8abd-9806-481f-908b-d89a267728cd
```

```
5it [00:11,  2.35s/it]
```

```
Datasplit set4 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-5-4560bbce' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=19732a25-0e0a-460b-a8f3-a83089ebd9a8
```

```
5it [00:13,  2.60s/it]
```

```
Datasplit set5 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-6-57993c74' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5a8207a6-dcb8-4cbf-a78e-a19b5c49adaf
```

```
5it [00:10,  2.11s/it]
```

```
Datasplit set6 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-7-10cae52f' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=132a80ac-1f95-46ea-a47d-0245da4dd967
```

```
5it [00:10,  2.13s/it]
```

```
Datasplit set7 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-8-e46801b5' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=457109ac-bcab-451f-8b67-6cc68785d295
```

```
5it [00:15,  3.00s/it]

Datasplit set8 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-9-02eef999' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=c38b9ce6-a2ef-41e0-a4a4-75ad6f1889f3
```

```
5it [00:12,  2.54s/it]

Datasplit set9 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-10-15e48fb8' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=253a5448-6f4c-47e9-9573-e3c4d7fbd32f
```

```
3it [00:13,  4.37s/it]

Datasplit set10 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-11-9b60f173' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=c8d65dc6-e163-4c2a-b575-b6530fc71042
```

```
5it [00:11,  2.31s/it]

Datasplit set11 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-12-d20e1afa' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=3348c757-aca3-450e-80a3-1af263d86713
```

```
5it [00:15,  3.05s/it]

Datasplit set12 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-13-af0a3865' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=68de185d-1a12-4822-b48c-94a525fdab26
```

```
5it [00:13,  2.68s/it]
```

```
Datasplit set13 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-14-5f083228' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=1cf4530a-5af1-4350-b9c3-37bd9c8ee401
```

```
5it [00:15,  3.00s/it]
```

```
Datasplit set14 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-15-74f68302' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=27e87d38-25ec-43ae-b9bc-8954fe174674
```

```
5it [00:13,  2.73s/it]
```

```
Datasplit set15 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-16-c09df657' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=d5a8cc8b-5586-4c56-b815-b4b7ffe94603
```

```
5it [00:12,  2.55s/it]
```

```
Datasplit set16 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-17-0ccdea9e' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5fc9b7e2-ab61-4397-b875-9fdfbdf24ba7
```

```
5it [00:10,  2.09s/it]
```

```
Datasplit set17 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-18-af9cb8ce' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
```

```
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=3d2a98e4-3479-4477-a07d-f57900abc431
```

```
5it [00:10,  2.11s/it]
```

```
Datasplit set18 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-19-43de23fd' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=fff93734-4565-4eab-9d32-06e9af003f59
```

```
5it [00:13,  2.73s/it]
```

```
Datasplit set19 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap0-doc-
relevance-20-05145500' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=f7242759-6acc-4ee6-8bf1-b9f1d72041ec
```

```
3it [00:11,  3.75s/it]
```

```
Datasplit set20 evaluation completed
```

[53]:

```
project_names_test2 = ['rag-chunk2000-overlap0-doc-relevance-1-f68e44f4',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-2-8c3ce496',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-3-ef140a31',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-4-ae6d3aee',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-5-4560bbce',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-6-57993c74',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-7-10cae52f',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-8-e46801b5',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-9-02eef999',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-10-15e48fb8',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-11-9b60f173',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-12-d20e1afa',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-13-af0a3865',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-14-5f083228',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-15-74f68302',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-16-c09df657','rag-chunk2000-overlap0-doc-relevance-17
  ↪'rag-chunk2000-overlap0-doc-relevance-18-af9cb8ce',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-19-43de23fd',␣
  ↪'rag-chunk2000-overlap0-doc-relevance-20-05145500']

all_dfs_test2 = [client.
  ↪get_test_results(project_name=project_names_test2[projectidx]) for␣
  ↪projectidx in range(len(project_names_test2))]
combined_df_test2 = pd.concat(all_dfs_test2, ignore_index=True)

# Merge the two DataFrames on the matching columns
merge_df_test2 = pd.merge(
    combined_df_test2, test_df,
    left_on="input.example.question",
    right_on="Question",
    how="inner"  # Use "inner" to include only matching rows
)

# Reorder the DataFrame columns
results_df_test2 = merge_df_test2[desired_column_order]

results_df_test2.rename(columns=new_column_names, inplace=True)
```

```
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
```

```
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
/tmp/ipykernel_678/690879135.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test2 =
[client.get_test_results(project_name=project_names_test2[projectidx]) for
projectidx in range(len(project_names_test2))]
```

```
Score for Test 2 retriever: 0.671875
```

```
/tmp/ipykernel_678/690879135.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame
```

```
See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  results_df_test2.rename(columns=new_column_names, inplace=True)
```

2.4.2 Score for Test 2

```
[54]: print(f"Score for Test 2 retriever: {results_df_test2['Relevance Score'].
        ↪mean()}")
```

```
Score for Test 2 retriever: 0.671875
```

2.5 Test 3

Chunk_size = 2000, chunk_overlap = 100

Vector store with those specifications are created first, before undergoing inference and evaluation

```
[59]: # Creating the vector store for Test 1

      pdf_path = './data/(edited) DSA textbook Python.pdf'
      embedding_path = './embeddings-test3'
      vector_store = create_embeddings_from_pdf(pdf_path, embedding_path, 2000, 100)

      if vector_store != None:
          print(f"Vector store {pdf_path} for Test 3 to {embedding_path} is a success!
        ↪")
```

```
Total documents created: 751
Embeddings saved to ./embeddings-test3
Vector store ./data/(edited) DSA textbook Python.pdf for Test 3 to ./embeddings-
test3 is a success!
```

2.5.1 Evaluation on Test 3 using third retriever

```
[60]: from langchain_openai.embeddings import AzureOpenAIEmbeddings
      from langchain_community.vectorstores.faiss import FAISS

      embedding_model = AzureOpenAIEmbeddings(azure_endpoint=os.
        ↪environ['AZURE_OPENAI_ENDPOINT'],
                                             api_key=os.environ['AZURE_OPENAI_API_KEY'],
                                             model=os.
        ↪environ['TEXT_EMBEDDING_MODEL_NAME'],
                                             azure_deployment=os.
        ↪environ['TEXT_EMBEDDING_DEPLOYMENT_NAME'])

      docsearch = FAISS.load_local("./embeddings-test3", embeddings=embedding_model,␣
        ↪allow_dangerous_deserialization=True)

      rag_bot = RagBot(docsearch)
```

```python
def predict_rag_answer_with_context(example: dict):
    """Use this for evaluation of retrieved documents and hallucinations"""
    response = rag_bot.get_answer(example["question"])
    return {"answer": response["answer"], "contexts": response["contexts"]}
```

[61]:

```python
# Dataset has been split on LangSmith interface into 20 splits
# This is to bypass the token restriction on the OpenAI API calls

from time import sleep

for setnumber in range(1, 21):
    dataset_name = "Algotutor_Dataset_20split"
    experiment_results = evaluate(
        predict_rag_answer_with_context,
        data=client.list_examples(dataset_name=dataset_name,␣
  ↪splits=[f"set{setnumber}"]),
        evaluators=[docs_relevance_evaluator],
        experiment_prefix=f"rag-chunk2000-overlap100-doc-relevance-{setnumber}",
        # Any experiment metadata can be specified here
        metadata={
            "variant": "chunk_size=2000, chunk_overlap=100",
        },
    )
    print(f"Datasplit set{setnumber} evaluation completed")

    # Sleep in order to bypass the minute rate limit on the token calls
    if setnumber < 20:
        sleep(30)
```

```
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-1-db735c22' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=ea4af3ec-de2c-4f8a-8cf2-de5ede7ca681
```

```
5it [00:13,  2.79s/it]
```

```
Datasplit set1 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-2-e33de573' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=41d07cba-fd39-483c-8ad7-ffa2cd58fae0
```

```
5it [00:15,  3.14s/it]

Datasplit set2 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-3-777f0047' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=1800b542-3590-4d6e-a7cd-ff81ffe2728d



5it [00:14,  2.82s/it]

Datasplit set3 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-4-2cf804c5' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=7d308d5d-efe4-4a12-a810-256994367cb1



5it [00:11,  2.29s/it]

Datasplit set4 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-5-63f09e84' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=f93dcc50-f601-4845-9e13-76aa20304983



5it [00:14,  2.80s/it]

Datasplit set5 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-6-b79c1e51' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=a135dc34-de08-4ce8-9904-a342d3356a0a



5it [00:11,  2.35s/it]

Datasplit set6 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-7-9163e970' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5e41088c-9f09-49d4-a46c-7d6dde0f3fdb
```

```
5it [00:12,  2.52s/it]

Datasplit set7 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-8-958562b8' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=1ffd705b-52df-4836-9de3-a6dc8a2c8711


5it [00:11,  2.37s/it]

Datasplit set8 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-9-4091d4a2' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=05727576-4133-4390-91ff-3d078ab115cf


5it [00:14,  2.86s/it]

Datasplit set9 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-10-232412a1' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=762ee04c-c11c-4a9f-889e-2e4d36c7763d


3it [00:15,  5.29s/it]

Datasplit set10 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-11-cdf454c6' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5f255ada-12a5-4f7c-a0de-7d8020cc53f1


5it [00:15,  3.06s/it]

Datasplit set11 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-12-fa301475' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
```

```
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=8858b5b9-3ae8-4e87-bc51-403e2e4e323f


5it [00:14,  2.97s/it]

Datasplit set12 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-13-217659a1' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=439da68b-2710-40db-9bdb-dbeb823166e9


5it [00:15,  3.05s/it]

Datasplit set13 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-14-7781698d' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=e3bdb1af-13b7-4b1f-97a6-f7516faa3e5f


5it [00:11,  2.40s/it]

Datasplit set14 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-15-df4d8554' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=828193f2-b409-445f-807d-9d3740051ce1


5it [00:14,  2.86s/it]

Datasplit set15 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-16-627ac9e4' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=1494c849-bf3b-478c-abbf-3699473990b3


5it [00:15,  3.16s/it]

Datasplit set16 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
```

```
relevance-17-9cdaf4a1' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=033f9461-d617-471b-9700-65aaa60a5ff2
```

```
5it [00:12,  2.54s/it]
```

```
Datasplit set17 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-18-c717e573' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=c07407ee-ebf6-4e82-be4d-e187f126f817
```

```
5it [00:12,  2.43s/it]
```

```
Datasplit set18 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-19-7c6a558a' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=cf170818-8bff-480f-b1a8-a00f834d8ee2
```

```
5it [00:10,  2.12s/it]
```

```
Datasplit set19 evaluation completed
View the evaluation results for experiment: 'rag-chunk2000-overlap100-doc-
relevance-20-94845d60' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5527c247-30dd-4a07-bd41-27ee14cb0405
```

```
3it [00:11,  3.96s/it]
```

```
Datasplit set20 evaluation completed
```

[64]:

```

```

```
project_names_test3= ['rag-chunk2000-overlap100-doc-relevance-1-db735c22',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-2-e33de573',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-3-777f0047',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-4-2cf804c5',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-5-63f09e84',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-6-b79c1e51',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-7-9163e970',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-8-958562b8',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-9-4091d4a2',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-10-232412a1',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-11-cdf454c6',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-12-fa301475',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-13-217659a1',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-14-7781698d',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-15-df4d8554',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-16-627ac9e4',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-17-9cdaf4a1',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-18-c717e573',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-19-7c6a558a',␣
 ↪'rag-chunk2000-overlap100-doc-relevance-20-94845d60']

all_dfs_test3 = [client.
 ↪get_test_results(project_name=project_names_test3[projectidx]) for␣
 ↪projectidx in range(len(project_names_test3))]
combined_df_test3 = pd.concat(all_dfs_test3, ignore_index=True)

# Merge the two DataFrames on the matching columns
merge_df_test3 = pd.merge(
    combined_df_test3, test_df,
    left_on="input.example.question",
    right_on="Question",
    how="inner"  # Use "inner" to include only matching rows
)

# Reorder the DataFrame columns
results_df_test3 = merge_df_test3[desired_column_order]

results_df_test3.rename(columns=new_column_names, inplace=True)
```

```
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
```

```
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test3 =
[client.get_test_results(project_name=project_names_test3[projectidx]) for
projectidx in range(len(project_names_test3))]
/tmp/ipykernel_678/92923380.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
```

```
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  results_df_test3.rename(columns=new_column_names, inplace=True)
```

2.5.2 Score for Test 3

[65]:

```
print(f"Score for Test 3 retriever: {results_df_test3['Relevance Score'].
 ↪mean()}")
```

```
Score for Test 3 retriever: 0.6479166666666667
```

2.6 Test 4

Chunk_size = 1000, chunk_overlap = 100

Vector store with those specifications are created first, before undergoing inference and evaluation

[66]:

```
# Creating the vector store for Test 1

pdf_path = './data/(edited) DSA textbook Python.pdf'
embedding_path = './embeddings-test4'
vector_store = create_embeddings_from_pdf(pdf_path, embedding_path, 1000, 100)

if vector_store != None:
    print(f"Vector store {pdf_path} for Test 4 to {embedding_path} is a success!
 ↪")
```

```
Total documents created: 1585
Embeddings saved to ./embeddings-test4
Vector store ./data/(edited) DSA textbook Python.pdf for Test 4 to ./embeddings-
test4 is a success!
```

2.6.1 Evaluation on Test 4 using fourth retriever

[67]:

```
from langchain_openai.embeddings import AzureOpenAIEmbeddings
from langchain_community.vectorstores.faiss import FAISS

embedding_model = AzureOpenAIEmbeddings(azure_endpoint=os.
 ↪environ['AZURE_OPENAI_ENDPOINT'],
                                   api_key=os.environ['AZURE_OPENAI_API_KEY'],
                                   model=os.
 ↪environ['TEXT_EMBEDDING_MODEL_NAME'],
                                   azure_deployment=os.
 ↪environ['TEXT_EMBEDDING_DEPLOYMENT_NAME'])

docsearch = FAISS.load_local("./embeddings-test4", embeddings=embedding_model,␣
 ↪allow_dangerous_deserialization=True)

rag_bot = RagBot(docsearch)
```

```python
def predict_rag_answer_with_context(example: dict):
    """Use this for evaluation of retrieved documents and hallucinations"""
    response = rag_bot.get_answer(example["question"])
    return {"answer": response["answer"], "contexts": response["contexts"]}
```

[68]:

```python
# Dataset has been split on LangSmith interface into 20 splits
# This is to bypass the token restriction on the OpenAI API calls

from time import sleep

for setnumber in range(1, 21):
    dataset_name = "Algotutor_Dataset_20split"
    experiment_results = evaluate(
        predict_rag_answer_with_context,
        data=client.list_examples(dataset_name=dataset_name,␣
 ↪splits=[f"set{setnumber}"]),
        evaluators=[docs_relevance_evaluator],
        experiment_prefix=f"rag-chunk1000-overlap100-doc-relevance-{setnumber}",
        # Any experiment metadata can be specified here
        metadata={
            "variant": "chunk_size=1000, chunk_overlap=100",
        },
    )
    print(f"Datasplit set{setnumber} evaluation completed")

    # Sleep in order to bypass the minute rate limit on the token calls
    if setnumber < 20:
        sleep(30)
```

```
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-1-39242bf8' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=5b070019-0bde-48cc-a008-3d2f72641467



5it [00:13,  2.74s/it]

Datasplit set1 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-2-ffdc9b39' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=4946b109-a453-48e1-8c26-c9bc8369c59b



5it [00:14,  2.81s/it]
```

```
Datasplit set2 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-3-2029bff0' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=4c8e8c2c-d9f3-470a-858f-b19c4940e1f1
```

```
5it [00:39,  7.87s/it]
```

```
Datasplit set3 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-4-5028a9fa' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=e41c242f-4a58-4b7f-8ace-3f4b7d3123de
```

```
5it [00:13,  2.68s/it]
```

```
Datasplit set4 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-5-f19d72c9' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=4e6c2aab-7235-432a-8e05-50a701587e6e
```

```
5it [00:15,  3.18s/it]
```

```
Datasplit set5 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-6-fb109294' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=b99288c1-403d-4841-9ee6-37404e1c018d
```

```
5it [00:12,  2.51s/it]
```

```
Datasplit set6 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-7-a0350f08' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=65242bd5-5e7f-4d8e-9fc6-da007563b2cc
```

```
5it [00:14,  2.87s/it]
```

```
Datasplit set7 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-8-dadf82ec' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=56a4184b-52b2-40f2-ba80-5ec487c6d3b2
```

```
5it [00:15,  3.15s/it]
```

```
Datasplit set8 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-9-866a1bd6' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=6d170513-065a-4c3c-8226-ccb1ab5a0143
```

```
5it [00:14,  2.97s/it]
```

```
Datasplit set9 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-10-a9cf247d' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=a747f15e-487d-4c01-be86-5bf3486b7e7c
```

```
3it [00:15,  5.08s/it]
```

```
Datasplit set10 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-11-39fb5df0' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=6bc4e71d-e29e-46ce-86e7-b4e77c522fd0
```

```
5it [00:12,  2.54s/it]
```

```
Datasplit set11 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-12-f82155a2' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=370b463b-1cb7-4023-ad40-3af64d680872
```

```
5it [00:17,  3.54s/it]

Datasplit set12 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-13-838b4ef8' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=53d54cd1-9466-4ccd-b478-274e98a27b7c


5it [00:14,  2.95s/it]

Datasplit set13 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-14-2ec3c3c0' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=4ea3a5a0-33bb-40b9-935a-633adea81496


5it [00:13,  2.65s/it]

Datasplit set14 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-15-28119639' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=2978c026-550f-4f32-8eb2-1f4e4b5acf87


5it [00:15,  3.09s/it]

Datasplit set15 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-16-dc20cbe5' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=a5ff52ff-e600-43cc-903e-1823fc52809c


5it [00:13,  2.78s/it]

Datasplit set16 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-17-f0ff6a90' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
```

```
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=aba3c07b-13fa-4a63-b735-d8792af699c5
```

```
5it [00:13,  2.64s/it]
```

```
Datasplit set17 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-18-08aaafb9' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=0af3d50f-21fd-48cc-a164-9dc12b207fd6
```

```
5it [00:12,  2.48s/it]
```

```
Datasplit set18 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-19-202cd5df' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=33bb1b67-3b2b-4477-b29e-985b25f5788d
```

```
5it [00:13,  2.68s/it]
```

```
Datasplit set19 evaluation completed
View the evaluation results for experiment: 'rag-chunk1000-overlap100-doc-
relevance-20-2c5009fe' at:
https://smith.langchain.com/o/65a167b9-d4dd-594a-9fef-
747869e69109/datasets/2ffa90d0-c4c7-4365-b7f2-
db681b2d2c8a/compare?selectedSessions=38bdda40-c87d-4c55-8a2a-492484e195b0
```

```
3it [00:12,  4.06s/it]
```

```
Datasplit set20 evaluation completed
```

[69]:

```

```

```
project_names_test4= ['rag-chunk1000-overlap100-doc-relevance-1-39242bf8',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-2-ffdc9b39',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-3-2029bff0',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-4-5028a9fa',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-5-f19d72c9',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-6-fb109294',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-7-a0350f08',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-8-dadf82ec',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-9-866a1bd6',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-10-a9cf247d',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-11-39fb5df0',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-12-f82155a2',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-13-838b4ef8',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-14-2ec3c3c0',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-15-28119639',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-16-dc20cbe5',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-17-f0ff6a90',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-18-08aaafb9',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-19-202cd5df',␣
 ↪'rag-chunk1000-overlap100-doc-relevance-20-2c5009fe']

all_dfs_test4 = [client.
 ↪get_test_results(project_name=project_names_test4[projectidx]) for␣
 ↪projectidx in range(len(project_names_test4))]
combined_df_test4 = pd.concat(all_dfs_test4, ignore_index=True)

# Merge the two DataFrames on the matching columns
merge_df_test4 = pd.merge(
    combined_df_test4, test_df,
    left_on="input.example.question",
    right_on="Question",
    how="inner"  # Use "inner" to include only matching rows
)

# Reorder the DataFrame columns
results_df_test4 = merge_df_test4[desired_column_order]

results_df_test4.rename(columns=new_column_names, inplace=True)
```

```
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
```

```
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:3: UserWarning: Function get_test_results is in
beta.
  all_dfs_test4 =
[client.get_test_results(project_name=project_names_test4[projectidx]) for
projectidx in range(len(project_names_test4))]
/tmp/ipykernel_678/4238294787.py:17: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
```

```
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  results_df_test4.rename(columns=new_column_names, inplace=True)
```

2.6.2 Score for Test 4

[71]:

```
print(f"Score for Test 4 retriever: {results_df_test4['Relevance Score'].
 ↪mean()}")
```

```
Score for Test 4 retriever: 0.7177083333333334
```

3 Conclusion

Recap on the retriever's specifications for each test: - Test 1: Chunk_size = 1000, chunk_overlap = 0 - Test 2: Chunk size = 2000, chunk_overlap = 0 - Test 3: Chunk size = 2000, chunk_overlap = 100 - Test 4: Chunk size = 1000, chunk_overlap = 100

3.0.1 Final Table Summary of all the retriever's relevance scores

[80]:

```
scores = [
    results_df['Relevance Score'].mean(),
    results_df_test2['Relevance Score'].mean(),
    results_df_test3['Relevance Score'].mean(),
    results_df_test4['Relevance Score'].mean(),
]

final_df = pd.DataFrame({
    'Test': [1, 2, 3, 4],
    'Score': scores,
    'Chunk_Size': [1000, 2000, 2000, 1000],
    'Chunk_Overlap': [0, 0, 100, 100]
}, index=None)

final_df
```

[80]:

```
   Test     Score  Chunk_Size  Chunk_Overlap
0     1  0.714583        1000              0
1     2  0.671875        2000              0
2     3  0.647917        2000            100
3     4  0.717708        1000            100
```

[79]:

```
# Getting the excel file results

results_df['Test'] = 1
results_df_test2['Test'] = 2
results_df_test3['Test'] = 3
results_df_test4['Test'] = 4
```

```python
all_results_df = pd.concat([results_df, results_df_test2, results_df_test3, results_df_test4], ignore_index=True)

# Saving the concatenated DataFrame to an Excel file
all_results_df.to_excel('concatenated_evaluation_results.xlsx', index=False)
```