Project: PDF Summarizer

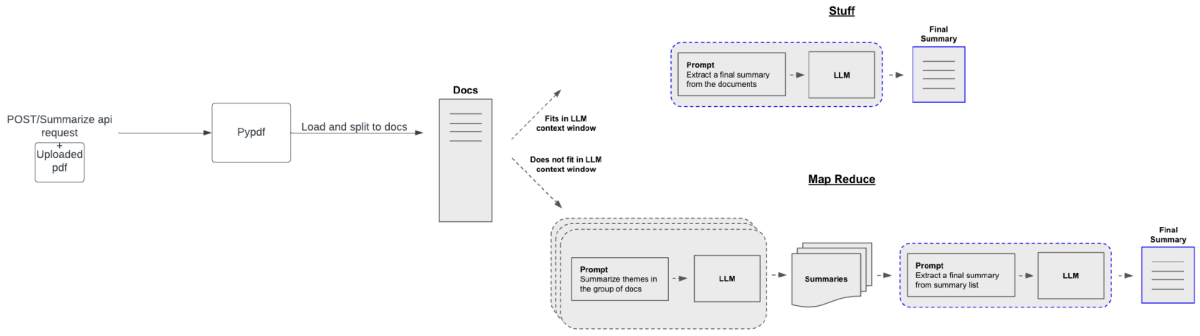
1. Scenario: Summarize a pdf file of any size.

2. Objective: Summarize pdf file of any size given the limitation of some llm's context window (here lama2 with 4096 as max number of tokens)

3. The nl query: (static)

```
"""Write a concise summary of the following:
   "{text}"
   CONCISE SUMMARY:"""
```

4. Architecture:



- The LLM: Llama2, why?
    1. Easy to run locally (with Ollama).
    2. Free.
    3. known for good summarization capabilities.

- Pypdf: document loader, loads the pdf document into the Document format, splitted.

-Chain: Based on the number of tokens, the appropriate way of summarization is chosen:

- Stuff: Simply stuff all documents into a single prompt.
- Map-reduce: Summarize each document on its own in a 'map' step and then 'reduce' the summaries into a final summary.

5. Handling concurrent requests/invalid queries:

- Number of tokens > Context window: Use map-reduce
- Empty pdf file: Returns: "I apologize, but there is no text provided for me to summarize. Could you please provide a passage or text that I can summarize?"
- Multiple pdf files: Possibility to merge multiple pdfs and give a single summarization.
- Uploaded file is not a pdf file: Returns: 'Please upload a pdf file'
- Corrupted pdf file: Returns: ''The file is corrupted''

6. How it works/Reproduce the results:
Follow README.md file

7. Evaluation:
Evaluation dataset (combining long and short texts), BillSum (random choice)
https://paperswithcode.com/dataset/billsum

Metrics:

1. ROUGE: measures the n-gram overlap between the generated summary and the reference summary.
    a. ROUGE-1: unigram,
    b. ROUGE-2: bigram,
    c. ROUGE-L: measure similarity using the longest sequence of words in common.
    d. ROUGE-Lsum: Divides the generated summary and the reference summary into sentence units and measures the similarity between the sentence units.

We can use other metrics like
BLEU (more used for translation, but can be used for evaluating summarization),
BERTScore(measures the similarity between two sentences using the BERT representation of each word and calculating the F1-score)

Results I got after evaluating on the first 10 elements of the BillSum dataset:
{'rouge-1': {'r': 0.33974400976741054, 'p': 0.4746293456743224, 'f': 0.37037122857400595},
'rouge-2': {'r': 0.14239906465798963, 'p': 0.23006445598654152, 'f': 0.16150150418410883},
'rouge-l': {'r': 0.3075088690324614, 'p': 0.4309780342289871, 'f': 0.33622694404034076}}

8. Limitations and Improvements:
The used approaches of summarization take a lot of time to process, especially map-reduce.
=> Need for a better gpu.
The used approaches require an important amount of computational power that could be optimized.
=> Other approaches could be used: clustering summarization.

The used large language model (llama2) ‘s context window is limited.
=> Other LLMs might be better for this task, especially for large files:

- 16k token OpenAI gpt-3.5-turbo-1106
- 100k token Anthropic Claude-2

We also can get better results by choosing better prompt templates.