Summarization of Harry Potter Books Assignment

Task:

Given a pdf file containing 4 harry potter books, namely

1. Harry Potter and the Sorcerer's Stone
2. Harry Potter and the Chamber of Secrets
3. Harry Potter and the Prisoner of Azkaban
4. Harry Potter and the Goblet of Fire

Summarize them into maximum 10 lines paragraph and benchmark the below LLMs according to my observation.
LLMs specified are:

1. gemini -pro
2. falcon 7b
3. Mistral 7b

Approach:

General:
Since the task at hand is to summarize more than 1500 pages into maximum 10 lines, the first thing that comes to mind is to break the pages into chunks, summarize them and then summarize the collective summaries. According to Langchain documentation on summarisation, there are two methods regarding this.

1. Map-reduce
2. Refine

There is a third one called stuff, but that is a general one where all the text is fed directly into the LLM.

Map-reduce

Chunks are summarized and stored which are used again to summarize the whole document, repeating the summarization part until we get our desired length.

Refine

This method iteratively summarizes the chunks, adding the latest result to the next chunk for summarization. Hence, it is a kind of rolling summary of the whole document.

These are some general methods of summarization.

Some approaches that I came up with for doing the assignment:

RAG pipeline:

Using retrieval augmented question answer, and asking the document 5 most relevant questions that captures the essence of the books would seem cost effective and fast once the pipeline is placed.
I asked Chat-GPT to give me 5 such generic questions and these were the results:

Introduction and Setting:
- "What is the primary setting and context established in this chapter?"

Character Development:
- "How do the characters evolve or face challenges in this chapter?"

Plot Advancements:
- "What significant events or revelations propel the plot forward in this chapter?"

Theme Exploration:
- "Which themes or motifs are explored, and how do they contribute to the overall narrative?"

Conflict and Resolution:
- "What conflicts arise, and how are they resolved or left unresolved in this chapter?"

We can either take the whole book and ask these questions or do it chapter by chapter.

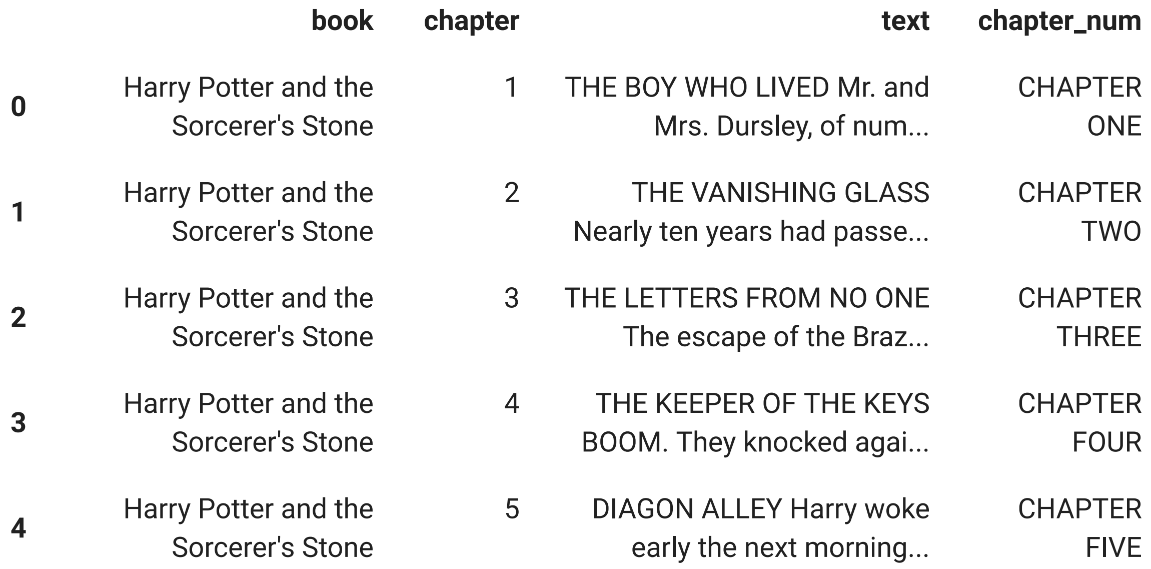
Approach used:

Step 1: Preprocess the document

Preprocess the document to make a dataframe consisting of

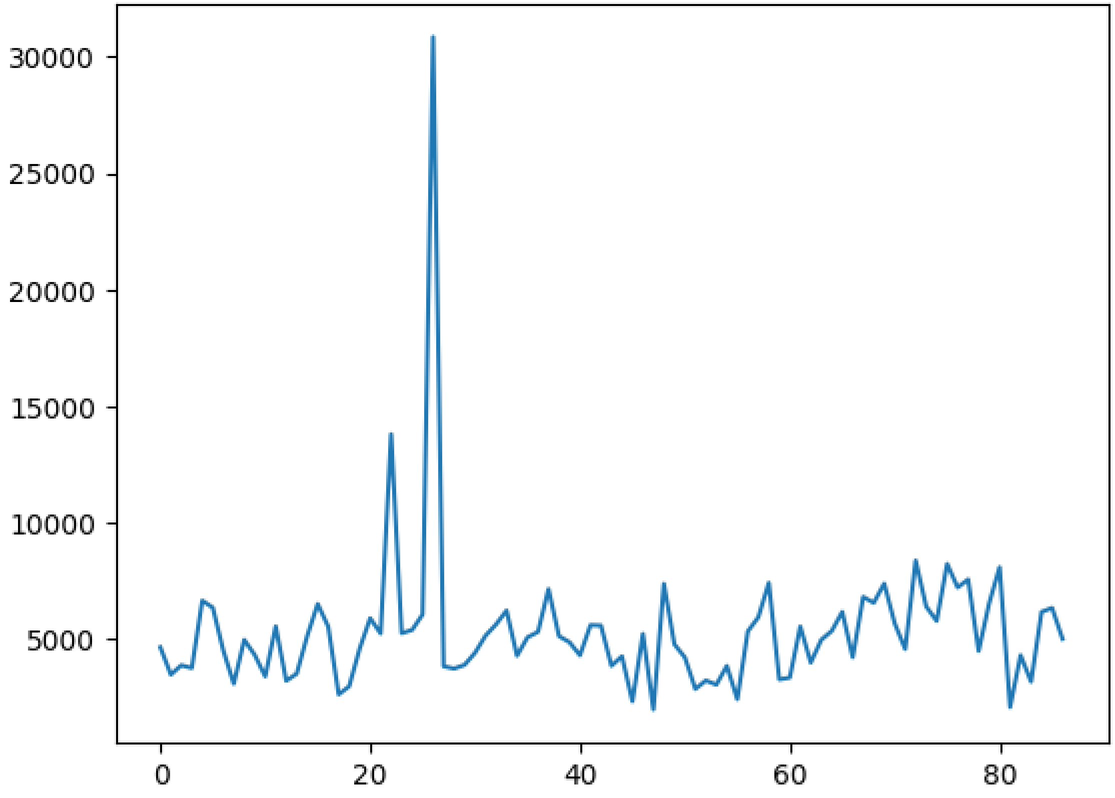
1. Book name
2. Chapter
3. Text

In the following format:



	book	chapter	text	chapter_num
0	Harry Potter and the Sorcerer's Stone	1	THE BOY WHO LIVED Mr. and Mrs. Dursley, of num...	CHAPTER ONE
1	Harry Potter and the Sorcerer's Stone	2	THE VANISHING GLASS Nearly ten years had passe...	CHAPTER TWO
2	Harry Potter and the Sorcerer's Stone	3	THE LETTERS FROM NO ONE The escape of the Braz...	CHAPTER THREE
3	Harry Potter and the Sorcerer's Stone	4	THE KEEPER OF THE KEYS BOOM. They knocked agai...	CHAPTER FOUR
4	Harry Potter and the Sorcerer's Stone	5	DIAGON ALLEY Harry woke early the next morning...	CHAPTER FIVE

Analyzing the number of words vs chapter numbers below shows that there are missing chapter names in the data.



The code used for splitting the chapters can be found in the data_processing.ipynb file where `'CHAPTER' , 'C H A P T E R'` was used for text splitting. I manually also checked for other patterns but this worked out the best.

Step 2: Chunking

Chunking of each chapter text was done.
For falcon, the input context window is 2048, so being on the safe side, 1400 tokens were taken for each chunk.

Step 3: Summarization on each chunk

Summarized each chunk and combined the results to form a bigger chunk just like map reduce from langchain. These chunks are again summarized until their size becomes less than 250 tokens and stored as the “Chapter Summary” in the dataframe.

Prompt used:

```
prompt = '''Summarize the below chunk of text, taken from a Harry Potter
book, in strictly one sentence. \nKeep only the most relevant information,
in the context of the whole book. \nText chunk: "<text>" \nSummarized
output:'''
```

Step 4: Chapter Summary dataframe creation

Chapter Summary dataframe is created and stored, since the whole setup is done on colab and once session expires we can’t access the data.

Step 5: Book Summary

Using the Chapter summaries created, we create a book wise summary using the concept of “refine” from langchain as its easier to track the summaries as they are less in number and much concise. So using the rolling chunk mechanism, we create the book summary and store in a dataframe for further use.

```
prompt = """Summarize the below chunk of chapter summaries taken from
Harry Potter book in strictly below 3 lines, keeping only the most
relevant information and theme, to get the essence of the whole book.
\nChapter wise text chunk: "<text>" \nSummarized output:"""
```

	book	summary
0	HARRY POTTER AND THE CHAMBER OF SECRETS ...	\nIn the Harry Potter book, the main plot revo...
1	Harry Potter and the Prisoner of Azkaban ...	\nHarry & friends met up at Hogwarts to prote...
2	Harry Potter and the Goblet of Fire by	\n- Harry is a brave, loyal wizard who has bee...
3	Harry Potter and the Sorcerer's Stone	Harry Potter and friends face challenges in t...

Step 6: Final Summary

Combining all the book summaries we finally get the chunk of text to generate the final summary.
Prompt used:

```
    final_prompt = "Summarize the below summary of Harry Potter books capturing the primary setting, themes, context, characters development, challenges faced, significant events, revelation, conflicts and resolution in utmost detail in strictly 10 lines. \nText: '<text>' \nSummarized output:"
```

For falcon, using the above prompt was giving better result as I tried other prompts as well and experimented.
I was skeptical if the model is using its available knowledge of harry potter from training data and giving a response from there, so I used a simple prompt as well.

Results

Gemini-Pro:

Details of the model:

Context length: 32,768 tokens
Usage via api

Comments:

1. The results were very impressive as the fluidity and clarity of the summarization was very good.
2. Due to large context window, fed larger chunk of text easily.
    a. On providing token length > 25k, the model's output was hallucinating and even giving back the same input once..
    b. With token length < 15k, the results were clear and stable. Hence, in the code a limit of 15k token was put to chunk any text having more than 15k tokens and passed separately and later on combining the outputs to produce a single summary of the chapter.
3. Following the approach given, book summary and then final summary was created.

Final Summary:

```
- Harry Potter discovers his wizarding identity and embarks on a journey
at Hogwarts, facing challenges and revealing secrets.
- He uncovers dark conspiracies and confronts Lord Voldemort, the evil
wizard who seeks to return to power.
- Harry and his friends overcome obstacles, including Voldemort's
resurgence and threats to the wizarding world.
- Through friendship, loyalty, and courage, Harry confronts his destiny
and ultimately defeats Voldemort, restoring peace.
- Harry's journey from an ordinary boy to a powerful wizard highlights the
importance of friendship, courage, and perseverance in overcoming
challenges.
- The conflict between good and evil culminates in a final showdown, where
Harry's determination and sacrifice ultimately prevail.
```

This summary is very interesting as it narrates the whole story in a concise way, which came out without much experimentation with prompt. Hence, this model performed the best.

Falcon-7B

Details:

Context length: 2048 tokens
Loading it via transformers library in a GPU on Colab
Space needed : 15 GB

Comments:

1. While any generation, be it chapter summary or book or overall summary, a lot of experimentation had to be done for prompt designing.
2. At the start, used ‘book name’ as well in the prompt for giving better context to the model during chapter summary. But 30-50% times, the model focused more on the book name and instead of summarizing the chunk of text, was giving an overall summary of the book from its knowledge of Harry Potter books. Hence, needed to experiment with the prompt removing the book name and started getting better results.
3. Since, total context window was 2048, used maximum 1400 tokens per chunk to summarize first, followed by merging these summaries to form a chapter summary of 1 sentence.
    a. If the chapter summary was 2-3 sentences long, then probably the model could have captured more from each chapter.
    b. Due to time constraints, was not able to experiment with these.

Final Summary

After much experimenting with prompt, the best output that was achieved is:

```
- Harry Potter and friends face challenges in the wizarding world.
- Themes, characters, and the plot of the book revolve around challenges
involving danger, the wizarding hierarchy, and friendships.
- The protagonists must face deathly hallows, centaur, dangerous
creatures, and Voldemort, while the story concludes with their victory and
personal growth.
- Ron gets injured, and the plot revolves around the protection of Harry
and their friends while facing Dementors and Death Eaters.
```

Prompt used:

```
prompt = "Summarize the below text in detail in strictly 6 lines. \nText: '<text>' \nSummarized output:"
```

Previously, asking to generate in 10 lines was giving low quality summary in 1-2 lines, but changing it to 6 lines gave better result, reducing the randomness to get more from provided sentences, giving 4 sentences.

Other Summaries are as follows:

```
Harry's friends face challenges in the wizarding world while he sacrifices
of his own life to save them, and the story concludes with their school's
safety. Themes involve friendship and responsibility, and new characters
help them develop and grow.
```

- When asked for 10 lines

Summary 1:

```
# The context involves a wizarding world with Death Eaters, centaurs, and
the wizarding hierarchy. Themes include friendship, courage, and
responsibility. Characters are Harry Potter, Hermione Granger, Ron
Weasley, and their wizarding peers. Main challenges involve Harry's quest
to protect his friends and the wizarding world from Death Eaters and
centaurs. Significant events involve their escape from the wizarding
world, and Voldemort's resurrection. The plot resolution involves Harry's
recovery and Voldemort's death. The summary has 40 words.
```

Summary 2:

```
# Harry Potter series is set in a wizarding world where a young wizard,
Harry, makes new friends and learns powerful magic. They face challenging
life-threatening situations and must navigate through dangerous chess
pieces as they attempt to save the wizarding world from the evil
Voldemort. Harry's character development, friendships, and learning
powerful spells throughout the series lead to his ultimate triumph over
the Dark Lord.
```

- Above summaries are when I mentioned Harry Potter books in the final prompt below

```
final_prompt = "Summarize the below summary of Harry Potter books
capturing the primary setting, themes, context, characters development,
challenges faced, significant events, revelation, conflicts and resolution
in utmost detail in strictly 6 lines. \nText: '<text>' \nSummarized
output:"
```

Where it used its previous knowledge and came up with a better answer.

Depending on the quality of summary it’s not close to Gemini-pro but for a 7B model, it does the job.

Mistral-7B

Details:

Context length: 8k tokens
Loading it via transformers library in a GPU on Colab
Space needed : >15 GB

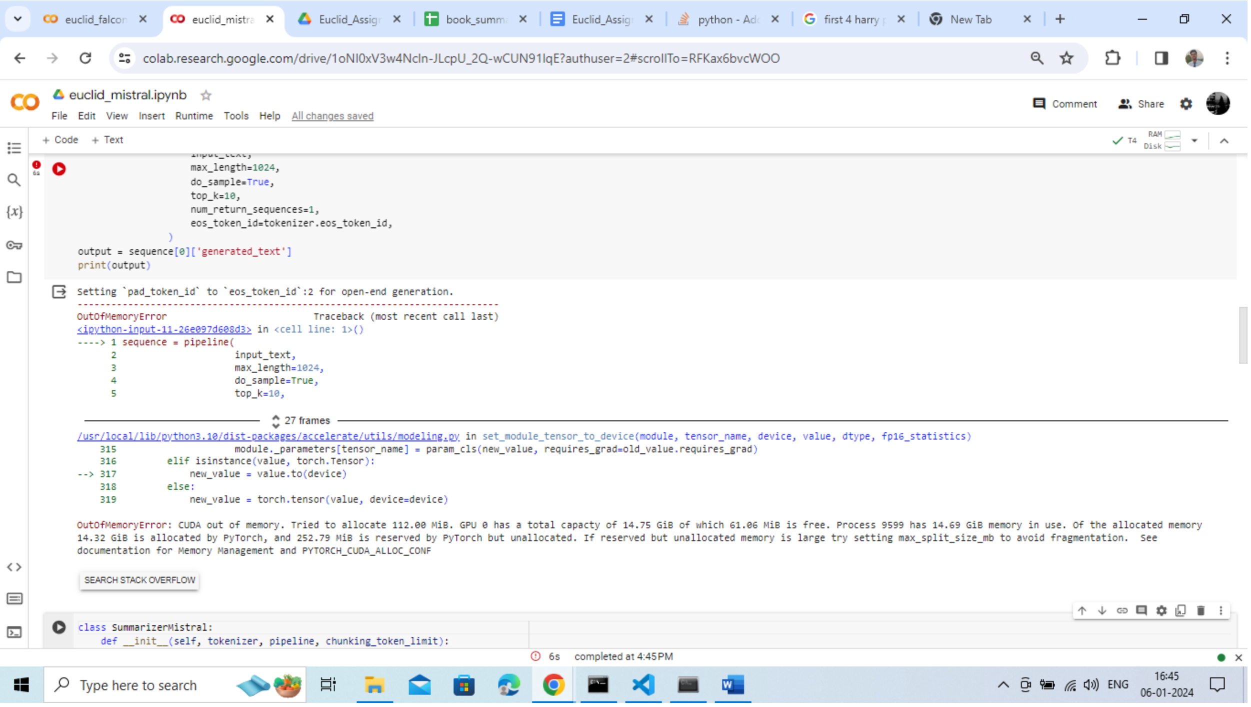
Comments:

Unfortunately, the 15 GB space given in colab was not sufficient to inference this model. The model was loaded fine, but during inference, kept getting “cuda out of memory error”. Hence, was not able to move forward with the experimentation of this model.
I tried to work with both

1. `mistralai/Mistral-7B-v0.1`
2. `mistralai/Mistral-7B-v0.2`

But both were unable to have inference on the colab.

Screenshots of the error are shown below:



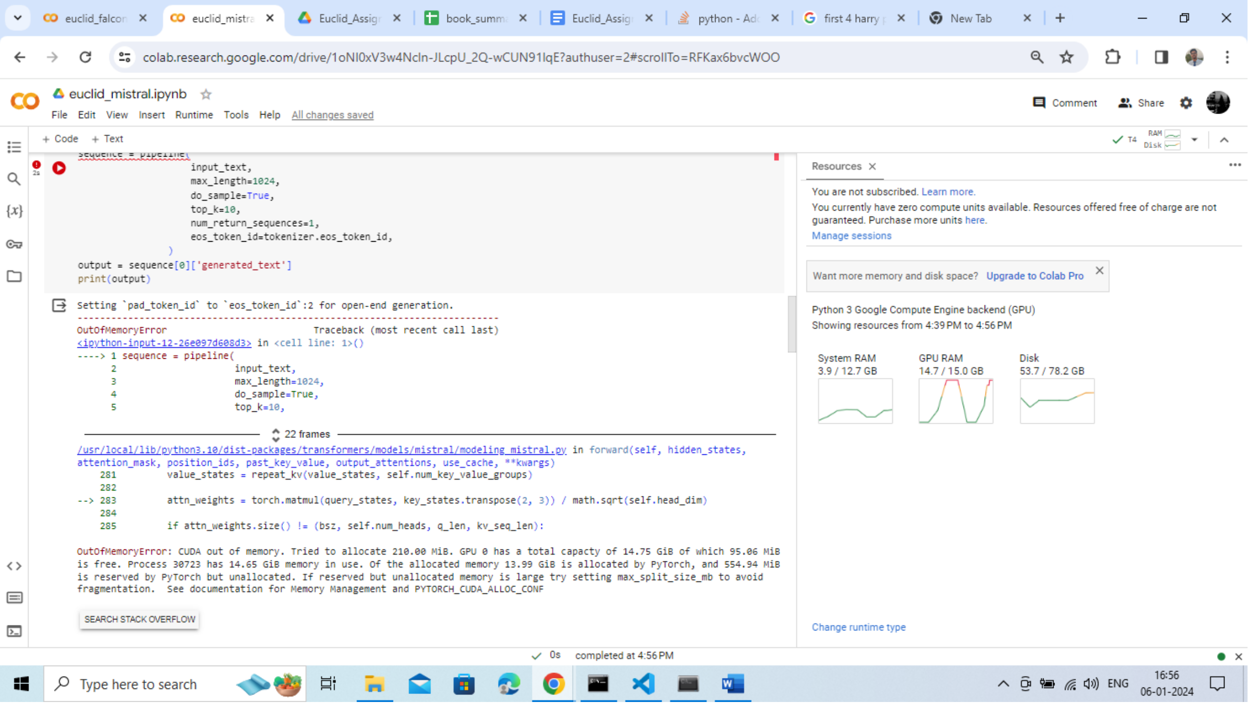
```
                     input_text,
                     max_length=1024,
                     do_sample=True,
                     top_k=10,
                     num_return_sequences=1,
                     eos_token_id=tokenizer.eos_token_id,
                 )
output = sequence[0]['generated_text']
print(output)
```

```
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
---------------------------------------------------------------------------
OutOfMemoryError                          Traceback (most recent call last)
<ipython-input-11-26e097d608d3> in <cell line: 1>()
----> 1 sequence = pipeline(
      2                     input_text,
      3                     max_length=1024,
      4                     do_sample=True,
      5                     top_k=10,

27 frames
/usr/local/lib/python3.10/dist-packages/accelerate/utils/modeling.py in set_module_tensor_to_device(module, tensor_name, device, value, dtype, fp16_statistics)
    315                     module._parameters[tensor_name] = param_cls(new_value, requires_grad=old_value.requires_grad)
    316         elif isinstance(value, torch.Tensor):
--> 317             new_value = value.to(device)
    318         else:
    319             new_value = torch.tensor(value, device=device)

OutOfMemoryError: CUDA out of memory. Tried to allocate 112.00 MiB. GPU 0 has a total capacty of 14.75 GiB of which 61.06 MiB is free. Process 9599 has 14.69 GiB memory in use. Of the allocated memory 14.32 GiB is allocated by PyTorch, and 252.79 MiB is reserved by PyTorch but unallocated. If reserved but unallocated memory is large try setting max_split_size_mb to avoid fragmentation.  See documentation for Memory Management and PYTORCH_CUDA_ALLOC_CONF
```

```
class SummarizerMistral:
    def __init__(self, tokenizer, pipeline, chunking_token_limit):
```



```
sequence = pipeline(
                    input_text,
                    max_length=1024,
                    do_sample=True,
                    top_k=10,
                    num_return_sequences=1,
                    eos_token_id=tokenizer.eos_token_id,
                )
output = sequence[0]['generated_text']
print(output)
```

```
Setting `pad_token_id` to `eos_token_id`:2 for open-end generation.
---------------------------------------------------------------------------
OutOfMemoryError                          Traceback (most recent call last)
<ipython-input-12-26e097d608d3> in <cell line: 1>()
----> 1 sequence = pipeline(
      2                     input_text,
      3                     max_length=1024,
      4                     do_sample=True,
      5                     top_k=10,

                              22 frames
/usr/local/lib/python3.10/dist-packages/transformers/models/mistral/modeling_mistral.py in forward(self, hidden_states,
attention_mask, position_ids, past_key_value, output_attentions, use_cache, **kwargs)
    281         value_states = repeat_kv(value_states, self.num_key_value_groups)
    282
--> 283         attn_weights = torch.matmul(query_states, key_states.transpose(2, 3)) / math.sqrt(self.head_dim)
    284
    285         if attn_weights.size() != (bsz, self.num_heads, q_len, kv_seq_len):

OutOfMemoryError: CUDA out of memory. Tried to allocate 210.00 MiB. GPU 0 has a total capacity of 14.75 GiB of which 95.06 MiB
is free. Process 30723 has 14.65 GiB memory in use. Of the allocated memory 13.99 GiB is allocated by PyTorch, and 554.94 MiB
is reserved by PyTorch but unallocated. If reserved but unallocated memory is large try setting max_split_size_mb to avoid
fragmentation.  See documentation for Memory Management and PYTORCH_CUDA_ALLOC_CONF
```

You are not subscribed. Learn more.

You currently have zero compute units available. Resources offered free of charge are not guaranteed. Purchase more units here.

Manage sessions

Want more memory and disk space? Upgrade to Colab Pro

Python 3 Google Compute Engine backend (GPU)
Showing resources from 4:39 PM to 4:56 PM

System RAM 3.9 / 12.7 GB
GPU RAM 14.7 / 15.0 GB
Disk 53.7 / 78.2 GB

Change runtime type

✓ 0s completed at 4:56 PM

Comparison of different models:

Prompting: Easier in Gemini-pro
Inference time: Faster for Gemini-pro api
Summary output: Better with Gemini-pro

Though, I was not not able to experiment with Mistral-7B, I believe that the encoder-decoder architecture of Mistral would have been a better choice compared to Falcon's decoder only architecture to capture intricate details and nuances to carry forward in the summarization process. Also, the larger summarization window using Sliding-Window Attention in Mistral would have been beneficial in capturing a better understanding of the chunk by feeding larger chunks, losing lesser information on every iteration as compared to Falcon.

Next Steps:

1. Experimenting with various prompts and style to capture more elements like characters development, challenges faced, significant events, revelation, conflicts and resolution during each iteration in chapter chunks could be done
2. Using libraries like Bitsnbytes to use quantised version of the models to save on the hardware resources and experimenting to check the overall performance, making it faster and cost effective
3. Trying to see if getting more sentences per chapter works better, to not lose out on information
4. Trying a RAG based approach while asking the 5 most important question per chapter to get the most important plots and sequences per chapter
5. Trying and experimenting with various cost effective simpler model like T5 and others for abstractive summarization at the start to get the most important sequences and then summarizing them, saving time during initial