Data creation:

The knowledge source was compiled by each group member scraping a subset of the referenced websites/domains/pdfs included in the lab writeup. We decided to include the contents of these as well as, for websites, the contents of all relevant linked sites on the base website. We deemed anything that wasn't a link to a social media website as relevant in order to avoid ads and potentially unrelated content. We did include some information outside of the recommended sites that felt relevant as we saw fit. For example, we searched for related wikipedia pages and allowed the web scrapers to reach some external sites that were 1 level deep away from main sites.

We extracted raw data using the BeautifulSoup with requests and PDFReader libraries to extract text from websites and pdfs respectively. For websites using AJAX responses, we just used the requests and json libraries to extract the relevant data.

In annotating data, we decided to create about 10 question/answer pairs for data within each of the webpages/documents we visited. We also made up some seemingly reasonable questions that were not strictly based on the knowledge base to avoid the bias of knowing the answer was in the knowledge base. This would ensure we would be evenly assessing our entire knowledge base while still being compact enough to run a significant number of ablations on our rag system. We made sure that each question we asked was significantly different from the others by ensuring we had one question for each of the Who, What, Where, Why, When and How starters (so long as it did not result in an awkward question) and tried to cover significantly different information for each. Overall, we ended up with 394 questions (not divisible by 10 as a result of our data quality estimation below), which we then randomly split into train and test sets, with 82 test questions and 312 train questions.

For annotations, we did not use any special interface. Each member of our time was in charge of scraping a different subset of websites, so we each independently annotated the data from our sections. So, we used a combination of excel, notepad, and VsCode as our interface for annotations, then combined the questions and reference answers before performing our train/test split.

We estimated our data quality by randomly selecting 28 questions from our test set. We then had John and Amogh independently annotate these questions, and then calculated the F1 score between answers (removing articles and punctuation and ensuring lowercase, as in the Squad paper). We achieved an average F1 score of 0.8078. This is a relatively high level of agreement, though we would expect a bit higher with more concise answers. This seems to be for two reasons. First, some of the questions were vague enough to have significantly different, technically correct answers. For example, the question "What is the highest elevation point in Pittsburgh?" was answered "1370 ft" and "Brashear Reservoir". While these answers are very different, they both answer the question, as Brashear Reservoir is the highest elevation point in Pittsburgh at an elevation of 1370 feet. Second, there was a difference between how concise the answers were. For example, the question "What vaccine was developed by Dr Jonas Salk in Pittsburgh?" was answered by John as "Polio" and by Amogh as "Polio Vaccine". In general John had shorter answers, leading to this discrepancy. We felt as though the discrepancy in how concise answers were was negligible, so we opted to only address the first issue by coming through our train and test questions, removing any that we deemed too vague.

We did not use any extra unannotated training data, as we used a prompt-based system without fine tuning. As such, the only useful training data for us would be annotated question/answer pairs for multishot learning.

Model Details:

We experimented with multiple small LLMs (between 3 and 15 billion parameters) and used prompting rather than fine tuning to get our answers. Specifically, we used a multishot prompt with huggingface models Zephyr-3B, Zephyr-7B-Beta, Ministerial-8B-Instruct-2410, and Qwen2.5-14B-Instruct. For each, we opted not to fine-tune the LLMs but instead do a prompt-based search using the following template:

```
{
    "role": "system",

    "content": """Using the information contained in the context,
    give a concise answer to the question.
    if possible limit your answer to single or a few words for who, when, where questions.
    wherever possible extract name, date, or title without additional explanations.
    Respond only to the question asked, response should be concise and relevant to the question.
    You should do short answers format responses. DO NOT PUT "ANSWER" BEFORE THE ANSWER.
    Don't answer in full sentences. For example, say "12" instead of "The answer is 12."
    """,
    },

    {
    "role": "user",
    "content": """Context:
    {context}
    ---
    Here are some examples of question answer pairs:

    Who is Pittsburgh named after?
    William Pitt

    What famous machine learning venue had its first conference in Pittsburgh in 1980?
    ICML

    What musical artist is performing at PPG Arena on October 13?
    Billie Eilish
    ---
```

Don't answer in full sentences. For example, say "12" instead of "The answer is 12."

Now here is the question for you to answer:
{question}

"""",
},

We experimented with the Zephyr models because we Zephyr is what GPT-o1 uses, so we thought its performance would be good for our RAG system. We had also tried the Ministral model because we had heard good things about Mistral, and wanted to try see how a model from knowledge distillation would perform. Finally, we tried Qwen because it was the largest model that we could find on Huggingface that would still fit on a T4. We opted for a prompt-based solution over fine tuning in order to run more ablations on each of these models, and after sufficient testing we concluded that a prompt-based solution would be sufficient for an effective RAG system.

Results:

Embedding Model	K	PCA n_components	FAISS {clusters, m, nbits}	chunk_size	Inference Model	Exact Match	F1	Recall	Notes
mpnet-base-v2	10	512	1000, 8, 8	384-64	stabilityai/stablelm-zephyr-3b	7.49%	20.72%	34.11%	
mpnet-base-v2	20	512	1000, 8, 8	384-64	stabilityai/stablelm-zephyr-3b	7.67%	20.96%	35.17%	
mpnet-base-v2	10	768	1000, 8, 8	384-64	stabilityai/stablelm-zephyr-3b	7.14%	19.20%	31.15%	
mpnet-base-v2	10	512	2000, 16, 16	384-64	stabilityai/stablelm-zephyr-3b	7.49%	19.85%	33.48%	
sentence-t5-large	10	skipped	176, 16, 8	384-64	stabilityai/stablelm-zephyr-3b	9.58%	21.57%	33.88%	
mpnet-base-v2	10	skipped	1000, 8, 8	384-64	mistralai/Ministral-8B-Instruct-2410	17.25%	27.58%	26.52%	
sentence-t5-large	20	skipped	199, 16, 8	264	HuggingFaceH4/zephyr-7b-beta	2.44%	12.61%	31.43%	
mpnet-base-v2	10	skipped	1000, 8, 8	384-64	Qwen/Qwen2.5-14B-Instruct	16.72%	28.26%	29.57%	
sentence-t5-large	10	skipped	200, 16, 8	384-64	mistralai/Ministral-8B-Instruct-2410	15.16%	26.35%	25.67%	Turned off do_sample and removed temperature

mpnet-base-v2	10	skipped	1000, 8, 8	384-64	mistralai/Ministral-8B-Instruct-2410	17.25%	27.51%	26.46%	Turned off do_sample and removed temperature
sentence-t5-large	10	skipped	200, 16, 8	264	mistralai/Ministral-8B-Instruct-2410	16.90%	27.17%	26.42%	Turned off do_sample and removed temperature
mpnet-base-v2	10	skipped	1000, 8, 8	384-64	mistralai/Ministral-8B-Instruct-2410	17.25%	26.26%	25.36%	Turned off do_sample and removed temperature, updated knowledge base
mpnet-base-v2	10	skipped	1000, 8, 8	384-64	mistralai/Ministral-8B-Instruct-2410	17.25%	26.35%	25.45%	Updated knowledge base
mpnet-base-v2	20	skipped	1000, 8, 8	384-64	mistralai/Ministral-8B-Instruct-2410	16.03%	25.68%	24.99%	Updated knowledge base

Here we see the results for each of our ablations. In some cases, these are statistically significant. For example, getting 35% recall from k=20 on zephyr-3b is significantly higher than any other entry barring maybe some other zephyr models. Now, while Qwen2.5-14B does give the best F1 score, it isn't significant as an ablation of Ministral-8B comes within a percent of that recall, and the runtime/improved metrics of Ministral make it more impressive. Ministerial does yield the highest exact match by a significant margin in comparison to the other two, so we take that in general as our best model as we feel exact match is the most important factor here. One thing to note, with better prompt engineering Qwen may have outperformed, the issue is it is multilingual and for some reason liked to output chinese characters or kanji. But overall, the results we have are in general statistically significant.

Analysis:

We particularly had an issue with getting the correct dates output for many questions before we updated our knowledge base, so to investigate we specifically looked at questions we wrote starting with "What date" or "When". On our train set, we got Exact Match: 13.33%, F1 Score: 24.45%, and Recall: 24.71%, and on our test set we got Exact Match: 9.09%, F1 Score: 21.35%, and Recall: 23.86%. For a point of comparison, we also test results on which: train - Exact Match: 34.62%, F1 Score: 41.13%, Recall: 39.95%, test - Exact Match: 42.86%, F1 Score: 45.61%, Recall: 42.53%, where: train - Exact Match: 24.29%, F1 Score: 27.97%, Recall: 24.43% test - Exact Match 23.31%, F1 Score: 26.82%, Recall: 22.33% and what: train - Exact Match: 17.16%, F1 Score: 30.71%, Recall: 29.22%. test - Exact Match: 16.30%, F1 Score:

29.89%, Recall: 25.49% questions. Overall, we can see _. Overall this tells us that our RAG system does indeed struggle heavily with providing dates, but does a great job of identifying answers to questions beginning with Which.

In a closed book, we saw these metrics: Exact Match: 4.88%, F1 Score: 24.82%, Recall: 37.61%. In the open book system with the same hyperparameters, we saw Exact Match: 17.25%, F1 Score: 26.35%, and Recall: 25.45%. This significant improvement in exact match indicates our RAG system is much better at actually retrieving information, though may be less robust compared to the closed book solution as its recall is significantly better.

In terms of systems, we'll show some example outputs of both Ministral and Qwen. From our test set:

Q. What vaccine was developed by Dr Jonas Salk in Pittsburgh?
Ministral A. Polio
Qwen A. Polio疫苗

Q. What dinosaur-related experiences can visitors enjoy at the museum?
Ministral A. Life-sized dinosaur skeletons and learn about prehistoric life
Qwen A. Nqwenqwuia化石名称未直接给出，根据描述推断可能需要更具体信息，原文未明确指出名字。若依据发现地及机构推测，可能指代Tyrannosaurus或Gualicho等，

Now this didn't hold for all answers with Qwen, but it did somewhat frequently use different languages. This is because Qwen2.5-14B is multilingual, so something in the context saying "Only use English in your responses" may have addressed this, as this was likely the biggest difference between exact matching.