Do LLMs can learn the Bible?

Eyal German, Yontan Shnitz and Eli Ben Shimol

July 13, 2024

Ben-Gurion University
of the Negev

Table of Contents

Introduction

- **In the beginning when God created the heavens and the earth...**

Introduction

- In our project we try to learn LLM the Bible using RAG
- We investigate different models in Hebrew and English

Lets start... Does LLM know Hebrew?

Figure: AI21 chat

Lets start... Does LLM know Hebrew?

LLMs that trained with Hebrew text:

- AlephBERT - Based on Google's BERT architecture
- HeBERT - Based on Google's BERT architecture.

Lets start... Does LLM know Hebrew?

LLMs that trained with Hebrew text:

- AlephBERT - Based on Google's BERT architecture
- HeBERT - Based on Google's BERT architecture.

Large models that capable of processing Hebrew text?

- Open AI (OpenAI's GPT): GPT-3.5 and GPT-4
- Claude (Anthropic's Claude AI): Claude 1 and Claude 2.
- AI21 Labs' Jurassic: Jurassic-1 and Jurassic-2
- Gemini (Google DeepMind's Gemini)

Now... Does LLM know the Hebrew Bible?

Figure: Claude chat

Now... Does LLM know the Hebrew Bible?

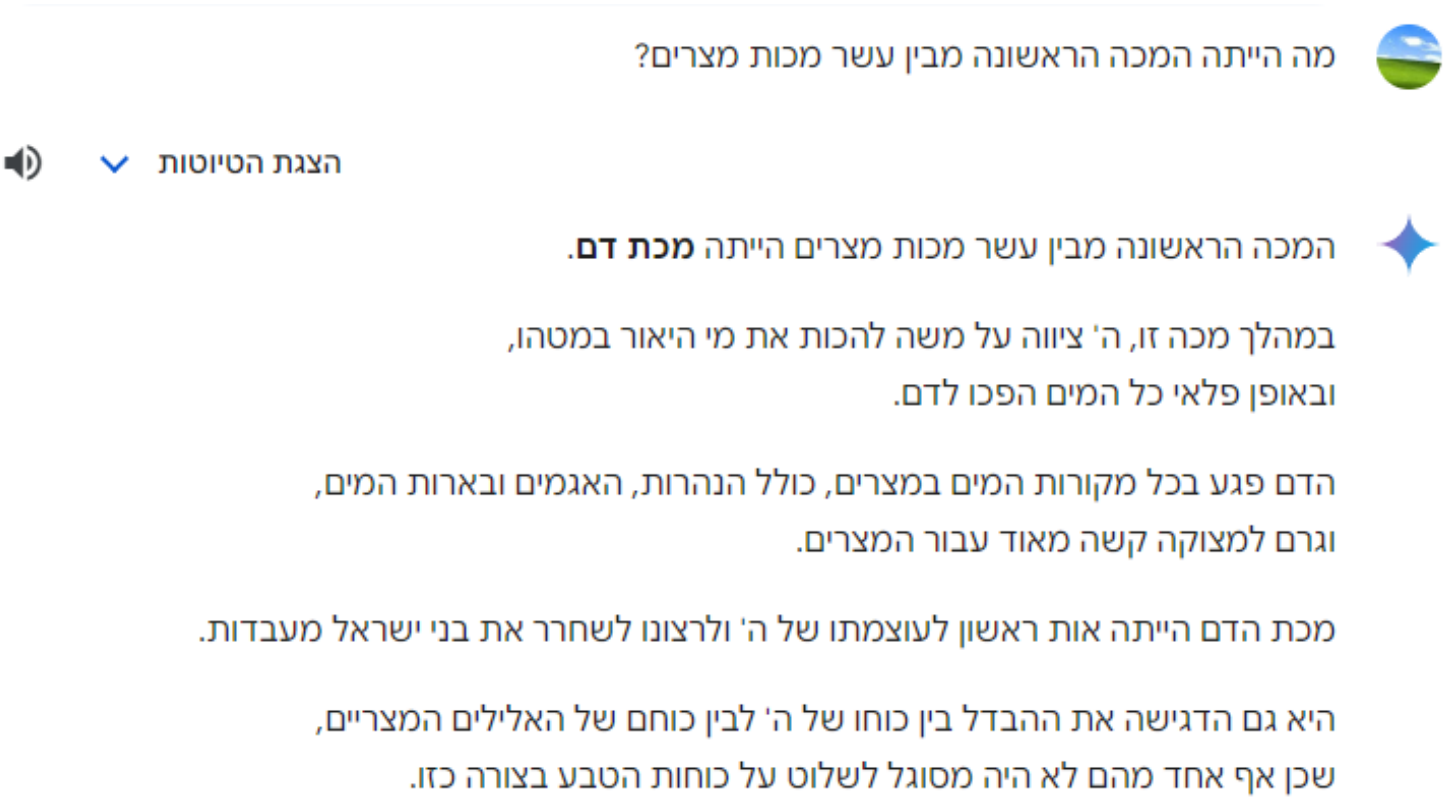
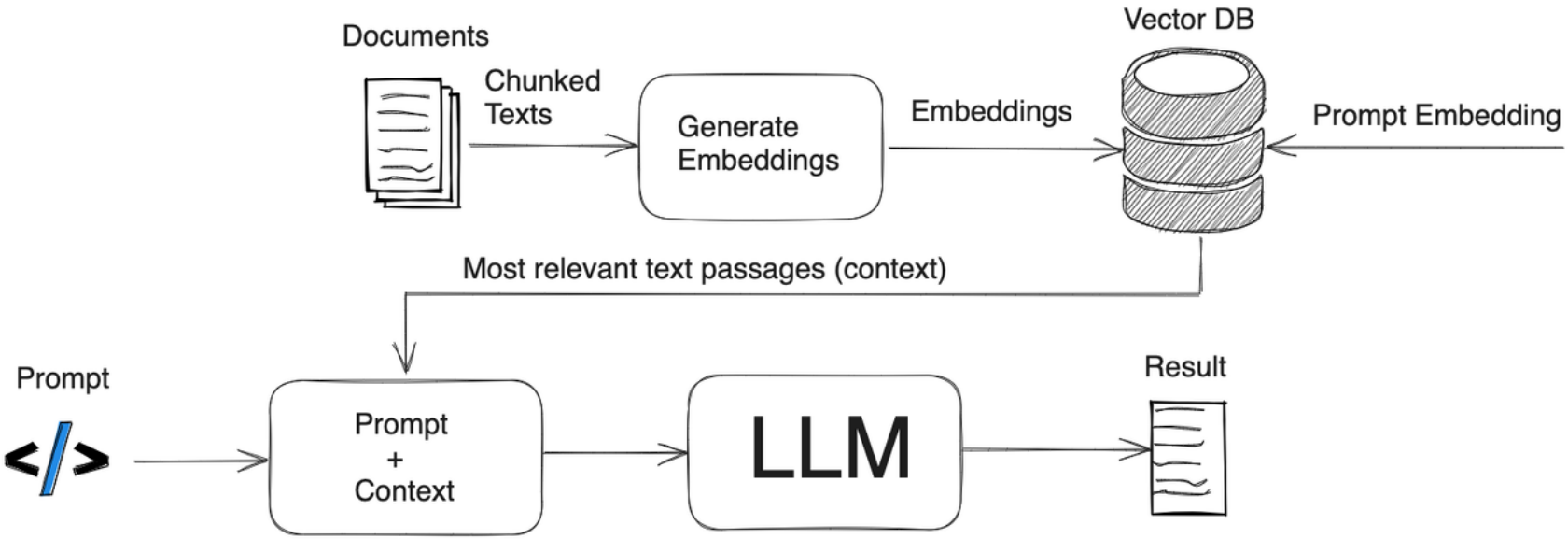


Figure: Gemini chat

Retrieval-Augmented Generation (RAG)



Research Questions

1. Can LLMs learn the Hebrew Bible and answer questions using RAG?
2. Can the retrieval model find the relevant chapter for a given question?
3. Can LLMs learn the English Bible and answer questions using RAG?

Table of Contents

Approach

- In this project, we aim to investigate how well LLMs can answer different types of questions.
- To answer the given questions, the model will use RAG on the Bible.
- We want to evaluate the results on different models and various types of questions.

Approach - RAG

- For the documents of the RAG, we downloaded the full Bible using the Sefaria API.
- We split the data by chapters and for each question retrieve the top-2 relevant chapters.
- We checked two kinds of retrieval models:
  1. Retriever based on Embeddings: Uses embeddings to find relevant passages based on the similarity of the text.
  2. BM25: Determines the similarity between Documents and the query based on the BM25 algorithm, which computes a weighted word overlap between the two strings.

Retriever based Embeddings I

```
1 # Load the tokenizer and model for Hebrew and English
2 hebrew_tokenizer =
  ↪  AutoTokenizer.from_pretrained("onlplab/alephbert-base")
3 hebrew_model =
  ↪  AutoModel.from_pretrained("onlplab/alephbert-base")
4
5 english_tokenizer =
  ↪  AutoTokenizer.from_pretrained("bert-base-uncased")
6 english_model =
  ↪  AutoModel.from_pretrained("bert-base-uncased")
7
8
9
```

Retriever based Embeddings II

```
10 def get_most_similar_embeddings(question_embedding,
   ↪  chapter_embeddings, top_k=2):
11     similarities = cosine_similarity([question_embedding],
       ↪  chapter_embeddings)
12     most_similar_indices =
       ↪  similarities.argsort()[0][-top_k:][::-1]
13     return most_similar_indices
14
15
```

Retriever based BM25 I

```
# Convert DataFrame to list of Documents for Hebrew and English texts
hebrew_documents = [Document(page_content=row['hebrew_text']) for _, row in df.iterrows()]
english_documents = [Document(page_content=row['english_text']) for _, row in df.iterrows()]
# Create BM25 retriever for Hebrew text
hebrew_retriever = BM25Retriever.from_documents(hebrew_documents, k=2)
# Create BM25 retriever for English text
english_retriever = BM25Retriever.from_documents(english_documents, k=2)
```

Retriever based BM25 II

```
8
9  def get_question_context_bm25(question, language='Hebrew'):
10     if language == "Hebrew":
11         retriever = hebrew_retriever
12     else:
13         retriever = english_retriever
14     texts = retriever.invoke(question)
15     texts = [text.page_content for text in texts]
16     context = "\n\n ".join(texts)
17     return context
18
```

Types of Questions

Explanation of Question Types:

- **Multiple Choice Questions:** These questions offer several options from which the correct answer must be selected.
- **Open Questions:** These require more detailed responses and allow for a range of possible answers.
- **Who Told Whom:** These questions focus on identifying who communicated specific information to whom.
- **True or False:** These statements must be evaluated as either true or false.

Table of Contents

Experimental Setup

- **Evaluation Metrics**
  - **Accuracy**: Measures the correctness of the answers generated by the models.
  - **Context Error**: The percentage of questions the model was unable to answer because the information was not found in the context.
  - **Accuracy on Answerable Questions**: Measures the percentage of correct answers from the questions the model was able to answer.
- We export the model's answers to a CSV file and check the results manually.

Models

Evaluation of Results on Three Models:

- **j2-ultra** - by AI21
- **Gemini-pro** - by Google
- **claude-2.1** - by Anthropic

Questions Creation

Datasets:

- **Hebrew:**
  - Used questions from the International Bible Competition.
  - Exported these questions into a CSV file.
  - Source: International Bible Competition
- **English:**
  - Translated questions from the International Bible Competition.
  - Augmented the dataset with additional questions from an Bible Quiz.

Run Function

```
1  def get_hebrew_answer_with_find_context(question, model,
   ↪  retriver="alephbert"):
2      prompt = PromptTemplate.from_template(hebrew_template)
3      if retriver == "alephbert":
4          context = get_question_embedding(question, "Hebrew")
5      else:
6          context = get_question_context_bm25(question,
       ↪  language='Hebrew')
7      print("Context:", context)
8      # Create the chain with a string output parser
9      chain = prompt | model | StrOutputParser()
10     # Invoke the chain with the context and question
11     response = chain.invoke({"context": context, "question":
       ↪  question})
12     print("response:", response)
13     return response, context
```

Table of Contents

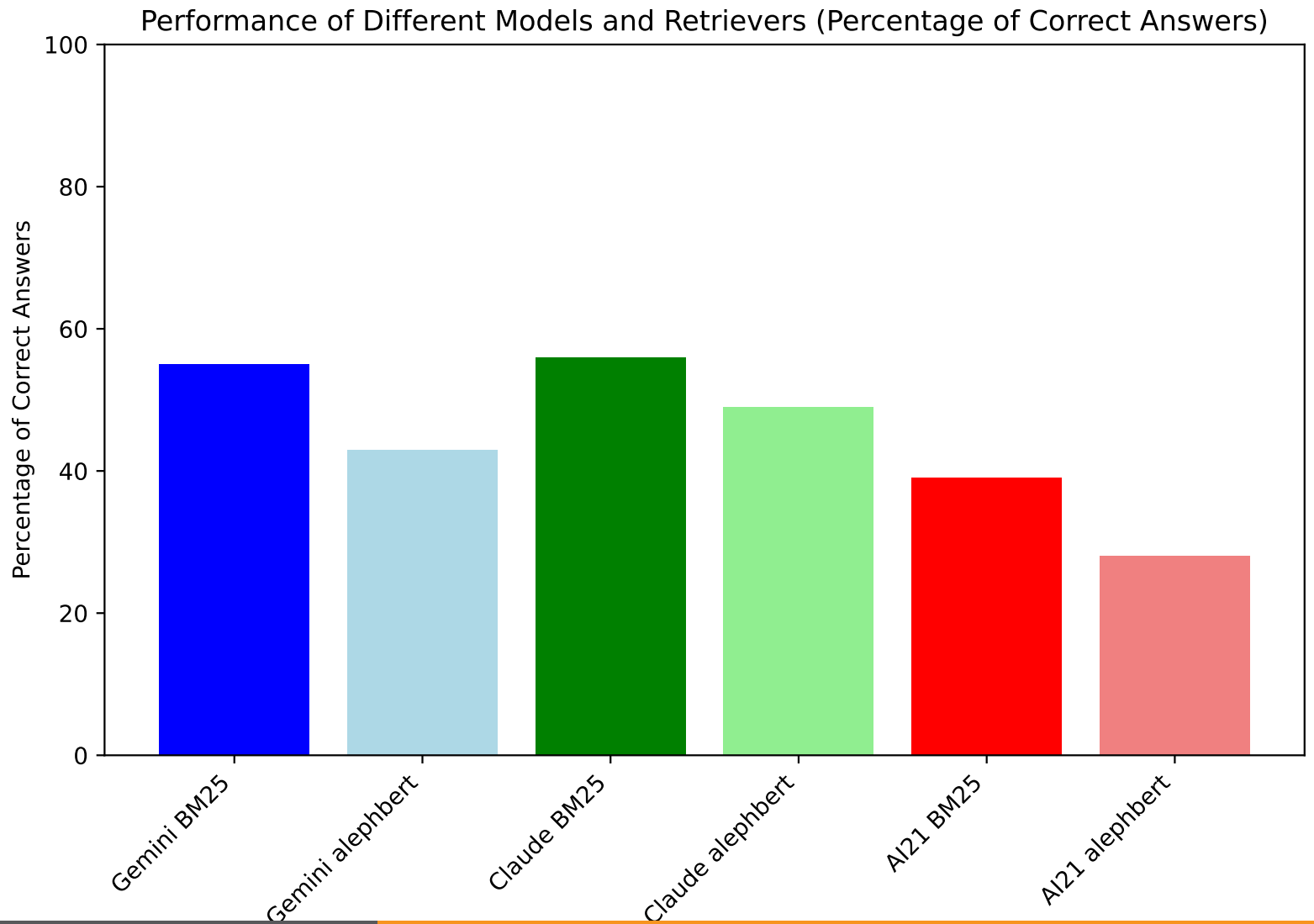
English Results

Table: Performance comparison of different models and retrievers across various datasets in English.

Model	**Gemini**		**Claude**		**AI21**	
Retriever	**alephbert**	**BM25**	**alephbert**	**BM25**	**alephbert**	**BM25**
Correct	43	55	49	56	28	39
Context Error	29	20	23	17	38	45

Out of 100 Multiple Choice Questions

English Results



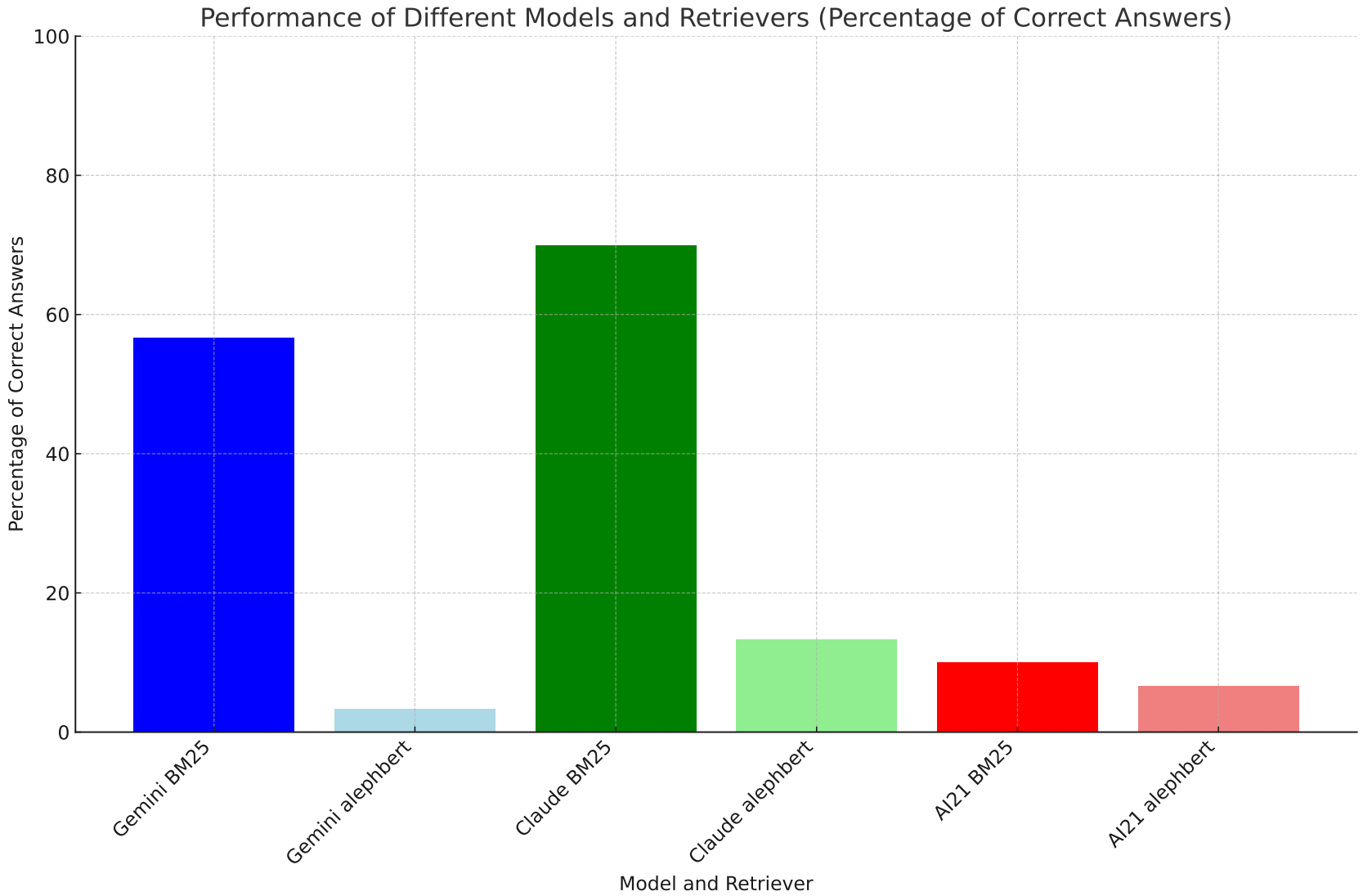
Hebrew Results

Table: Performance comparison of different models and retrievers across various datasets in Hebrew.

Model	**Gemini**		**Claude**		**AI21**	
Retriever	**alephbert**	**BM25**	**alephbert**	**BM25**	**alephbert**	**BM25**
Correct	1	17	4	21	2	3
Context Error	26	7	20	5	0	1

Out of 30 Multiple Choice Questions

Hebrew Results



Results Analysis

- Other types of questions did not achieve good results in our tests and required extensive manual checking.
- **Hebrew**
  - The results across all the models were not very good.
  - Using the BM25 retriever significantly improved the results.
- **English**
  - For English questions, we found that in many cases, the model was unable to answer due to the context it received.
  - In the translated questions, we found that the translation quality was poor, which affected performance.

Table of Contents

Medium articles

During the project we found 2 articles from Medium that are similar to our:

- When Llama learns Bible: Fine tune Llama 2 7B on questions on the bible.
-

GPTs - Bible Expert

- We also create custom GPTs to answer your questions on the bible ☺
- https://chatgpt.com/g/g-7qVTXRMhG-bible-expert

Conclusion

- **Challenges:**
  - Poor context handling by models led to incorrect answers.
  - Translation quality issues negatively impacted the performance of English models on translated questions.
  - Need for manual checking to verify the correctness of the models' answers.
- **Future Directions:**
  - Improve context retrieval methods to enhance model performance.
  - Continue to explore different model architectures and retrievers for optimal results.
  - Develop an automated method to verify the correctness of the models' answers.
- **Overall:**
  - While there are challenges, the use of RAG shows promise for teaching LLMs to understand and answer questions about the Bible.

Thank You!

Questions?

Thank you for your attention!