Question Answering over Docs

Contents

> Conceptual Guide

Question answering in this context refers to question answering over your document data. For question answering over other types of data, please see other sources documentation like SQL database Question Answering or Interacting with APIs.

For question answering over many documents, you almost always want to create an index over the data. This can be used to smartly access the most relevant documents for a given question, allowing you to avoid having to pass all the documents to the LLM (saving you time and money).

See this notebook for a more detailed introduction to this, but for a super quick start the steps involved are:

Load Your Documents

```
from langchain.document_loaders import TextLoader
loader = TextLoader('../state_of_the_union.txt')
```

See here for more information on how to get started with document loading.

Create Your Index

```
from langchain.indexes import VectorstoreIndexCreator
```

Skip to main content

The best and most popular index by far at the moment is the VectorStore index.

Query Your Index

```
query = "What did the president say about Ketanji Brown Jackson"
index.query(query)
```

Alternatively, use `query_with_sources` to also get back the sources involved

```
query = "What did the president say about Ketanji Brown Jackson"
index.query_with_sources(query)
```

Again, these high level interfaces obfuscate a lot of what is going on under the hood, so please see this notebook for a lower level walkthrough.

Document Question Answering

Question answering involves fetching multiple documents, and then asking a question of them. The LLM response will contain the answer to your question, based on the content of the documents.

The recommended way to get started using a question answering chain is:

```
from langchain.chains.question_answering import load_qa_chain
chain = load_qa_chain(llm, chain_type="stuff")
chain.run(input_documents=docs, question=query)
```

The following resources exist:

- Question Answering Notebook: A notebook walking through how to accomplish this task.
- VectorDB Question Answering Notebook: A notebook walking through how to do question answering over a vector database. This can often be useful for when you have a LOT of documents, and you don’t want to pass them all to the LLM, but rather first want to do some semantic search over embeddings.

Skip to main content

Adding in sources

There is also a variant of this, where in addition to responding with the answer the language model will also cite its sources (eg which of the documents passed in it used).

The recommended way to get started using a question answering with sources chain is:

```
from langchain.chains.qa_with_sources import load_qa_with_sources_chain
chain = load_qa_with_sources_chain(llm, chain_type="stuff")
chain({"input_documents": docs, "question": query}, return_only_outputs=True)
```

The following resources exist:

- QA With Sources Notebook: A notebook walking through how to accomplish this task.
- VectorDB QA With Sources Notebook: A notebook walking through how to do question answering with sources over a vector database. This can often be useful for when you have a LOT of documents, and you don't want to pass them all to the LLM, but rather first want to do some semantic search over embeddings.

Additional Related Resources

Additional related resources include:

- Utilities for working with Documents: Guides on how to use several of the utilities which will prove helpful for this task, including Text Splitters (for splitting up long documents) and Embeddings & Vectorstores (useful for the above Vector DB example).
- CombineDocuments Chains: A conceptual overview of specific types of chains by which you can accomplish this task.