Modules ■ Chains ■ Popular ■ Retrieval QA

Retrieval QA

This example showcases question answering over an index.

```
from langchain.chains import RetrievalQA
from langchain.document_loaders import TextLoader
from langchain.embeddings.openai import OpenAIEmbeddings
from langchain.llms import OpenAI
from langchain.text_splitter import CharacterTextSplitter
from langchain.vectorstores import Chroma
```

```
loader = TextLoader("../../state_of_the_union.txt")
documents = loader.load()
text_splitter = CharacterTextSplitter(chunk_size=1000, chunk_overlap=0)
texts = text_splitter.split_documents(documents)

embeddings = OpenAIEmbeddings()
docsearch = Chroma.from_documents(texts, embeddings)

qa = RetrievalQA.from_chain_type(llm=OpenAI(), chain_type="stuff", retriever=docsearch.as_retriever())
```

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

```
    " The president said that she is one of the nation's top legal minds, a former top litigator in private practice, a former federal public defender, and from a family of public school educators and police officers. He also said that she is a consensus builder and has received a broad range of support, from the Fraternal Order of Police to former judges appointed by Democrats and Republicans."
```

Chain Type

You can easily specify different chain types to load and use in the RetrievalQA chain. For a more detailed walkthrough of these types, please see this notebook.

There are two ways to load different chain types. First, you can specify the chain type argument in the `from_chain_type` method. This allows you to pass in the name of the chain type you want to use. For example, in the below we change the chain type to `map_reduce`.

```
qa = RetrievalQA.from_chain_type(llm=OpenAI(), chain_type="map_reduce", retriever=docsearch.as_retriever())
```

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

```
    " The president said that Judge Ketanji Brown Jackson is one of our nation's top legal minds, a former top litigator in private practice and a former federal public defender, from a family of public school educators and police officers, a consensus builder and has received a broad range of support from the Fraternal Order of Police to former judges appointed by Democrats and Republicans."
```

The above way allows you to really simply change the chain_type, but it doesn't provide a ton of flexibility over parameters to that chain type. If you want to control those parameters, you can load the chain directly (as you did in this notebook) and then pass that directly to the the RetrievalQA chain with the `combine_documents_chain` parameter. For example:

```
from langchain.chains.question_answering import load_qa_chain
qa_chain = load_qa_chain(OpenAI(temperature=0), chain_type="stuff")
qa = RetrievalQA(combine_documents_chain=qa_chain, retriever=docsearch.as_retriever())
```

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

```
    " The president said that Ketanji Brown Jackson is one of the nation's top legal minds, a former top litigator in private practice, a former federal public defender, and from a family of public school educators and police officers. He also said that she is a consensus builder and has received a broad range of support from the Fraternal Order of Police to former judges appointed by Democrats and Republicans."
```

Custom Prompts

You can pass in custom prompts to do question answering. These prompts are the same prompts as you can pass into the base question answering chain

```
from langchain.prompts import PromptTemplate
prompt_template = """Use the following pieces of context to answer the question at the end. If you don't know the answer, just say that you don't know, don't try to make up an answer.
```

```
{context}

Question: {question}
Answer in Italian:"""
PROMPT = PromptTemplate(
    template=prompt_template, input_variables=["context", "question"]
)
```

```
chain_type_kwargs = {"prompt": PROMPT}
qa = RetrievalQA.from_chain_type(llm=OpenAI(), chain_type="stuff", retriever=docsearch.as_retriever(),
chain_type_kwargs=chain_type_kwargs)
```

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

```
    " Il presidente ha detto che Ketanji Brown Jackson è una delle menti legali più importanti del paese, che
continuerà l'eccellenza di Justice Breyer e che ha ricevuto un ampio sostegno, da Fraternal Order of Police a
ex giudici nominati da democratici e repubblicani."
```

Return Source Documents

Additionally, we can return the source documents used to answer the question by specifying an optional parameter when constructing the chain.

```
qa = RetrievalQA.from_chain_type(llm=OpenAI(), chain_type="stuff", retriever=docsearch.as_retriever(), return_source_documents=True)
```

```
query = "What did the president say about Ketanji Brown Jackson"
result = qa({"query": query})
```

```
result["result"]
```

```
    " The president said that Ketanji Brown Jackson is one of the nation's top legal minds, a former top litigator in private practice and a former federal public defender from a family of public school educators and police officers, and that she has received a broad range of support from the Fraternal Order of Police to former judges appointed by Democrats and Republicans."
```

```
result["source_documents"]
```

```
    [Document(page_content='Tonight. I call on the Senate to: Pass the Freedom to Vote Act. Pass the John Lewis Voting Rights Act. And while you’re at it, pass the Disclose Act so Americans can know who is funding our elections. \n\nTonight, I’d like to honor someone who has dedicated his life to serve this country: Justice Stephen Breyer—an Army veteran, Constitutional scholar, and retiring Justice of the United States Supreme Court. Justice Breyer, thank you for your service. \n\nOne of the most serious constitutional responsibilities a President has is nominating someone to serve on the United States Supreme Court. \n\nAnd I did that 4 days ago, when I nominated Circuit Court of Appeals Judge Ketanji Brown Jackson. One of our nation’s top legal minds, who will continue Justice Breyer’s legacy of excellence.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='A former top litigator in private practice. A former federal public defender. And from a family of public school educators and police officers. A consensus builder. Since she’s been
nominated, she’s received a broad range of support—from the Fraternal Order of Police to former judges
appointed by Democrats and Republicans. \n\nAnd if we are to advance liberty and justice, we need to secure
the Border and fix the immigration system. \n\nWe can do both. At our border, we’ve installed new technology
like cutting-edge scanners to better detect drug smuggling.  \n\nWe’ve set up joint patrols with Mexico and
Guatemala to catch more human traffickers.  \n\nWe’re putting in place dedicated immigration judges so
families fleeing persecution and violence can have their cases heard faster. \n\nWe’re securing commitments
and supporting partners in South and Central America to host more refugees and secure their own borders.',
lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='And for our LGBTQ+ Americans, let’s finally get the bipartisan Equality Act to my
desk. The onslaught of state laws targeting transgender Americans and their families is wrong. \n\nAs I said
last year, especially to our younger transgender Americans, I will always have your back as your President,
so you can be yourself and reach your God-given potential. \n\nWhile it often appears that we never agree,
that isn’t true. I signed 80 bipartisan bills into law last year. From preventing government shutdowns to
protecting Asian-Americans from still-too-common hate crimes to reforming military justice. \n\nAnd soon,
we’ll strengthen the Violence Against Women Act that I first wrote three decades ago. It is important for us
to show the nation that we can come together and do big things. \n\nSo tonight I’m offering a Unity Agenda
for the Nation. Four big things we can do together.  \n\nFirst, beat the opioid epidemic.', lookup_str='',
metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0),
     Document(page_content='Tonight, I’m announcing a crackdown on these companies overcharging American
businesses and consumers. \n\nAnd as Wall Street firms take over more nursing homes, quality in those homes
has gone down and costs have gone up.  \n\nThat ends on my watch. \n\nMedicare is going to set higher
standards for nursing homes and make sure your loved ones get the care they deserve and expect. \n\nWe’ll
also cut costs and keep the economy going strong by giving workers a fair shot, provide more training and
apprenticeships, hire them based on their skills not degrees. \n\nLet’s pass the Paycheck Fairness Act and
paid leave.  \n\nRaise the minimum wage to $15 an hour and extend the Child Tax Credit, so no one has to
raise a family in poverty. \n\nLet’s increase Pell Grants and increase our historic support of HBCUs, and
invest in what Jill—our First Lady who teaches full-time—calls America’s best-kept secret: community
colleges.', lookup_str='', metadata={'source': '../../state_of_the_union.txt'}, lookup_index=0)]
```

Alternatively, if our document have a "source" metadata key, we can use the `RetrievalQAWithSourceChain` to cite our sources:

```
docsearch = Chroma.from_texts(texts, embeddings, metadatas=[{"source": f"{i}-pl"} for i in
range(len(texts))])
```

```
from langchain.chains import RetrievalQAWithSourcesChain
from langchain import OpenAI

chain = RetrievalQAWithSourcesChain.from_chain_type(OpenAI(temperature=0), chain_type="stuff",
retriever=docsearch.as_retriever())
```

```
chain({"question": "What did the president say about Justice Breyer"}, return_only_outputs=True)
```

```
    {'answer': ' The president honored Justice Breyer for his service and mentioned his legacy of
excellence.\n',
     'sources': '31-pl'}
```