# Differences between Langchain & LlamaIndex [closed]

Asked 1 year, 3 months ago  Modified 5 months ago  Viewed 77k times

126

**Closed**. This question is opinion-based. It is not currently accepting answers.

**Want to improve this question?** Update the question so it can be answered with facts and citations by editing this post.

Closed 10 months ago.

The community reviewed whether to reopen this question 10 months ago and left it closed:

> Original close reason(s) were not resolved

Improve this question

I'm currently working on developing a chatbot powered by a Large Language Model (LLM), and I want it to provide responses based on my own documents. I understand that using a fine-tuned model on my documents might not yield direct responses, so I'm exploring the concept of Retrieval-Augmented Generation (RAG) to enhance its performance.

In my research, I've come across two tools, Langchain and LlamaIndex, that seem to facilitate RAG. However, I'm struggling to understand the main differences between them. I've noticed that some tutorials and resources use both tools simultaneously, and I'm curious about why one might choose to use one over the other or when it makes sense to use them together.

Could someone please provide insights into the key distinctions between Langchain and LlamaIndex for RAG, and when it is beneficial to use one tool over the other or combine them in chatbot development?

**chatbot** **langchain** **large-language-model** **llama-index**

Share

Improve this question

Follow

edited Mar 9 at 0:23

desertnaut

**60.2k** ● 31 ● 151 ● 176

asked Aug 28, 2023 at 7:22

Yousif Abdalla

**1,537** ● 3 ● 8 ● 12

6 As much as I would like to see the answer to this, the question is likely to be closed as opinion based. I see it's already got one close vote. You might be able to rework it into something that's not primarily opinion based before more close votes come in. – Chris Strickland Aug 28, 2023 at 7:44

Should have been migrating to some AI SE site instead of getting closed. Similar question:

genai.stackexchange.com/q/1896/109 – Franck Dernoncourt
Jun 16 at 21:06

3 Answers Sorted by: Highest score (default)

127

# tl;dr

You'll be fine with just LangChain, however, LlamaIndex is optimized for indexing, and retrieving data.

# Here are the details

To answer your question, it's important we go over the following terms:

## Retrieval-Augmented Generation

Retrieval-Augmented Generation (or RAG) is an architecture used to help large language models like GPT-4 provide better responses by using relevant information from additional sources and reducing the chances that an LLM will leak sensitive data, or ‘hallucinate’ incorrect or misleading information.

## Vector Embeddings

Vector Embeddings are numerical vector representations of data. They are not only limited to text but can also

represent images, videos, and other types of data. They are usually created using an embedding model such as OpenAI's `text-embedding-ada-002` (see here for more information)

# LangChain vs. LlamaIndex

Let me start off by saying that it's not either LangChain or LlamaIndex. As you mentioned in your question, both tools can be used together to enhance your RAG application.

## LangChain

You can think of LangChain as a framework rather than a tool. It provides a lot of tools right out of the box that enable you to interact with LLMs. Key LangChain components include chains. Chains allow the *chaining* of components together, meaning you could use a `PromptTemplate` and a `LLMChain` to:

1. Create a prompt
2. Query a LLM

Here's a quick example:

```
...

prompt = PromptTemplate(template=template,
input_variables=["questions"])
```

```
chain = LLMChain(
    llm=llm,
    prompt=prompt
)

chain.run(query)
```

You can read more about LangChain components here.

## LlamaIndex

LlamaIndex, (previously known as GPT Index), is a data framework specifically designed for LLM apps. Its primary focus is on ingesting, structuring, and accessing private or domain-specific data. It offers a set of tools that facilitate the integration of custom data into LLMs.

Based on my experience with LlamaIndex, it is an ideal solution if you're looking to work with vector embeddings. Using its many available plugins you could load (or ingest) data from many sources easily, and generate vector embeddings using an embedding model.

One key feature of LlamaIndex is that it is optimized for index querying. After the data is ingested, an index is created. This `index` represents your vectorized data and can be easily queried like so:

```
...

query_engine = index.as_query_engine()
```

```
response = query_engine.query("Stackoverflow is
Awesome.")
```

LlamaIndex abstracts this but it is essentially taking your query `"Stackoverflow is Awesome."` and comparing it with the most relevant information from your vectorized data (or `index`) which is then provided as context to the LLM.

## Wrapping Up

It should be clear to you now why you might choose one or both technologies for your specific use case. If your app requires indexing and retrieval capabilities, and while you'll be just fine using LangChain (as it can handle that as well) I recommend integrating with LlamaIndex since it is optimized for that task and it is generally easier to ingest data using all the plugins and data connectors. Otherwise, if you just need to work with LLMs stick with only LangChain.

If you'd like to read more, I cover both LangChain and LlamaIndex on my blog. Here's a post looking at LangChain and LlamaIndex.

*Note: I am the author of this post.*

Share Improve this answer

Follow

answered Oct 18, 2023 at 16:45

jeff

1,486 ● 1 ● 4 ● 12

I am building similar to this but not with RAG, the issue I am encountering is the domain-specific data keeps on changing for me. So which should I prefer here? Also, the changing data for me is stored in an SQL database is there any way I can tell GPT to consider this data too while generating the content? – Nikhil Patel Apr 25 at 10:23

@NikhilPatel When you retrieve data dynamically from SQL database it is still RAG. Just add the dynamically changing data to the context and it's available to GPT. – Sjoerd222888 Nov 5 at 9:47

57

Langchain is a more general-purpose framework that can be used to build a wide variety of applications. It provides tools for loading, processing, and indexing data, as well as for interacting with LLMs. Langchain is also more flexible than LlamaIndex, allowing users to customize the behavior of their applications.

LlamaIndex is specifically designed for building search and retrieval applications. It provides a simple interface for querying LLMs and retrieving relevant documents. LlamaIndex is also more efficient than Langchain, making it a better choice for applications that need to process large amounts of data.

If you are building a general-purpose application that needs to be flexible and extensible, then Langchain is a good choice. If you are building a search and retrieval application that needs to be efficient and simple, then LlamaIndex is a better choice.

Share Improve this answer

Follow

answered Aug 29, 2023 at 18:50

ZKS

2,776 ● 4 ● 26 ● 37

2 @Marko, I have created an test of same two RAG algorithms with the same data, same LLM, same Embeddings and vector store. On Mac m1 machine I've noticed that llama-index is slower almost in 6 times. – Arrmlet Feb 6 at 1:14

@Arrmlet What code did you use for langchain and what code did you use for llama-index? Was the k parameter the same? – W -- Jun 24 at 1:26

@Arrmlet, but which is better in terms of accuracy? – Bennison J Jul 2 at 8:38

3

I have personally used LangChain for implementing a RAG based application. It connects pre-built repos having pdf files, a drag-drop file uploader, and a web-link. It works perfectly. I am using LLaMA 2 as the foundation model. For vector storage I used FAISS. Streamlit for the front-end. Technically, I found no limits or technical glitches while using LangChain for the task.

Share Improve this answer

Follow

edited Jul 3 at 6:34

Sangbok Lee

2,229 ● 3 ● 16 ● 33

answered Jan 22 at 12:44

Muhammad Roman

39 ● 2