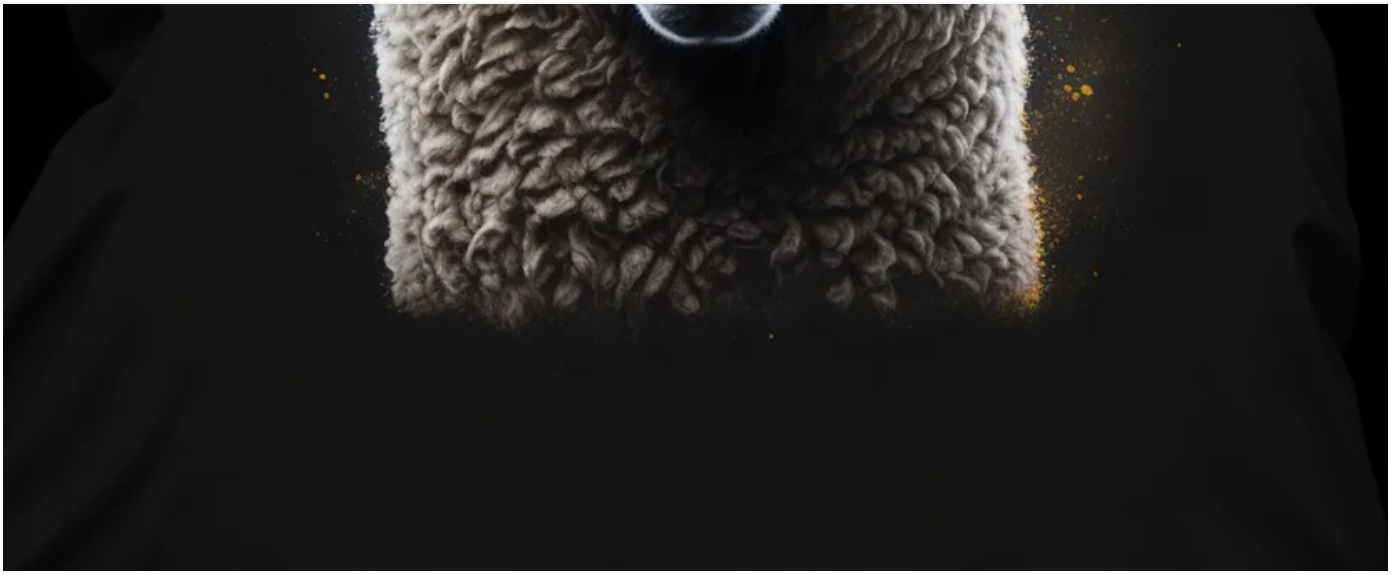
Member-only story

Text Summarization Llama2: how to Use LLama2 with Langchain

Tarik Kaoutar (高達烈) · Follow

Published in Level Up Coding
9 min read · Jul 27

Listen Share More



https://leonardo.ai/

In this Tutorial, I will guide you through how to use LLama2 with langchain for text summarization and named entity recognition using Google Colab Notebook:

Meta, better known to most of us as Facebook, has released a commercial version of Llama-v2, its open-source large language model (LLM) that uses artificial intelligence (AI) to generate text, images, and code.

What is LLama2?

Meta, better known to most of us as Facebook, has released a commercial version of Llama-v2, its open-source large language model (LLM) that uses artificial intelligence (AI) to generate text, images, and code.

Llama 2 is a successor to the Llama 1 model released earlier this year. However, Llama 1 was “closely guarded” and was only available on request.

Let’s start coding

Set Up Google Colab: Go to Google Colab (colab.research.google.com) and create a new notebook.

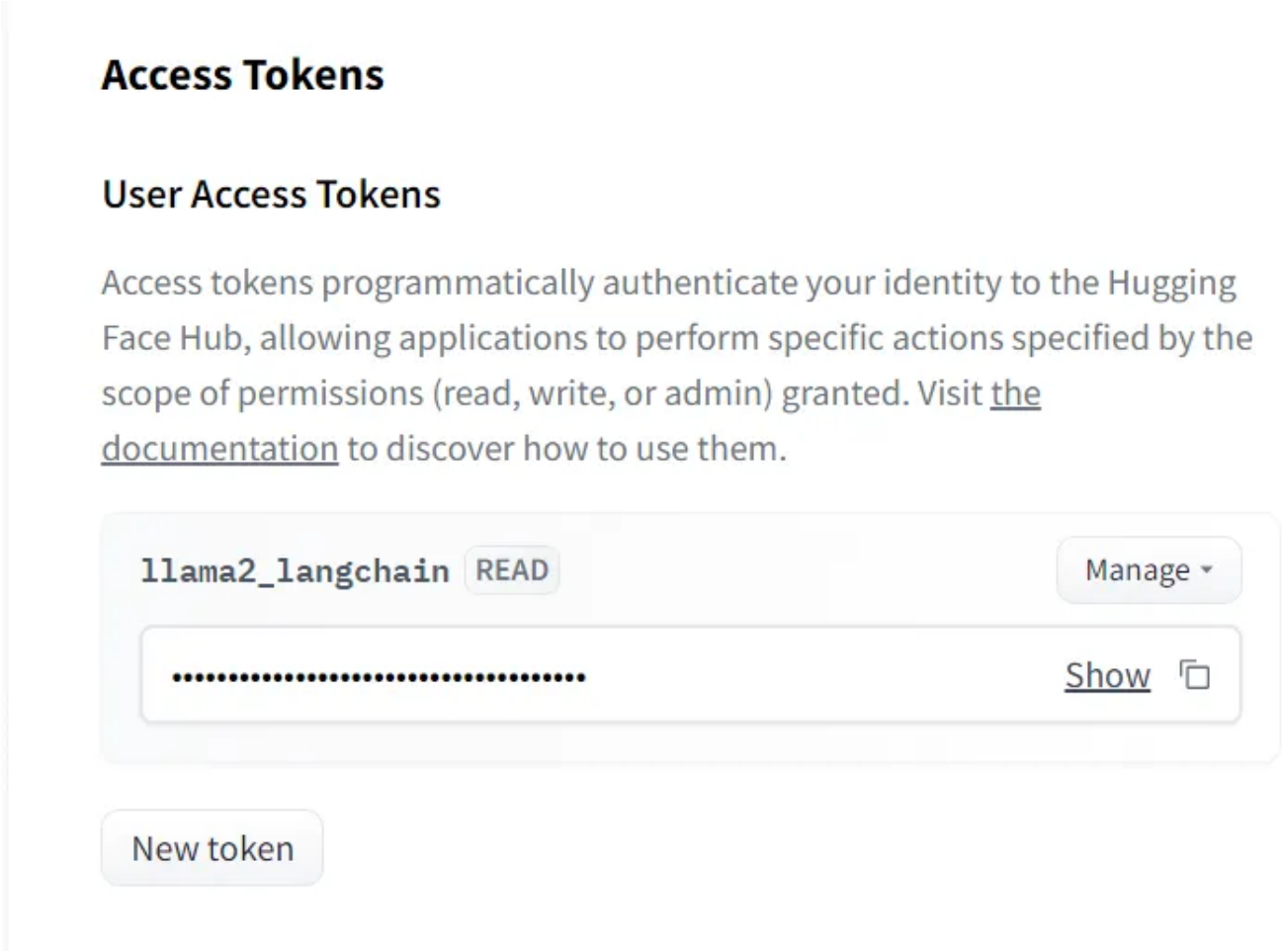
Install Required Libraries: In the first code cell of your Colab notebook, install the necessary libraries using the following code:

```
!pip install -q transformers einops accelerate langchain bitsandbytes
```

Login in !huggingface-cli login

```
!huggingface-cli login
```

If you already have a Hugging Face account, you can obtain your access token by going to the settings section on the Hugging Face website. From there, click on the “Access Tokens” tab, and you’ll be able to generate or copy your personal access token.



If you don't have a Hugging Face account yet, you can easily sign up for one on their website. Once you have an account, you can follow the same steps mentioned above to get your access token and use it to access private models and resources through the Hugging Face API or CLI.

installed the SentencePiece

If you haven't installed the SentencePiece

library, you can install it in your Colab notebook using:

```
!pip install sentencepiece
```

We going to set up a language generation pipeline using Hugging Face's transformers library and a specified model. The AutoTokenizer is used to fetch the tokenizer associated with the model. The pipeline is then configured with parameters such as

the text generation task, model, tokenizer, max_length of the generated text, and a few more.

```
from langchain import HuggingFacePipeline
from transformers import AutoTokenizer
import transformers
import torch

model = "meta-llama/Llama-2-7b-chat-hf"

tokenizer = AutoTokenizer.from_pretrained(model)

pipeline = transformers.pipeline(
    "text-generation", #task
    model=model,
    tokenizer=tokenizer,
    torch_dtype=torch.bfloat16,
    trust_remote_code=True,
    device_map="auto",
    max_length=1000,
    do_sample=True,
    top_k=10,
    num_return_sequences=1,
    eos_token_id=tokenizer.eos_token_id
)
```

We create an instance of the HuggingFacePipeline class, using the previously configured pipeline and setting the model's 'temperature' parameter, which influences the randomness of predictions.

```
llm = HuggingFacePipeline(pipeline = pipeline, model_kwargs = {'temperature':0})
```

The PromptTemplate class is used to create a template for a language model prompt. This is the instruction that the model will follow when generating text.

In this case, the template asks the model to summarize a text. The text to summarize is placed within triple backquotes (```). The model is asked to present the summary in bullet points. The {text} inside the template will be replaced by the actual text you want to summarize.

```
from langchain import PromptTemplate,  LLMChain

template = """
              Write a concise summary of the following text delimited by triple ba
              Return your response in bullet points which covers the key points of
              ```{text}```
              BULLET POINT SUMMARY:
            """

prompt = PromptTemplate(template=template, input_variables=["text"])

llm_chain = LLMChain(prompt=prompt, llm=llm)

text = """ As part of Meta's commitment to open science, today we are publicly re

Training smaller foundation models like LLaMA is desirable in the large language m

Over the last year, large language models — natural language processing (NLP) sys

Even with all the recent advancements in large language models, full research acc

Smaller models trained on more tokens — which are pieces of words — are easier to

Like other large language models, LLaMA works by taking a sequence of words as an

There is still more research that needs to be done to address the risks of bias, t

To maintain integrity and prevent misuse, we are releasing our model under a nonc

We believe that the entire AI community — academic researchers, civil society, po
"""

print(llm_chain.run(text))
```

Let's try it out
```

```
print(llm_chain.run(text))
```

```
/usr/local/lib/python3.10/dist-packages/transformers/generation/utils.py:1270: Use
  warnings.warn(
 • The article discusses the public release of LLaMA (Large Language Model Meta AI
            • Training smaller models like LLaMA requires less computing power and
```

```
text1 = """
Tesla, Inc. (/ˈtɛslə/ TESS-lə or /ˈtɛzlə/ TEZ-lə[a]) is an American multinational

Tesla is one of the world's most valuable companies and, as of 2023, was the world

Its subsidiary Tesla Energy develops and is a major installer of photovoltaic syst

Tesla was incorporated in July 2003 by Martin Eberhard and Marc Tarpenning as Tesl

Tesla began production of its first car model, the Roadster sports car, in 2008. T

Tesla has been the subject of lawsuits, government scrutiny, and journalistic crit
"""

print(llm_chain.run(text1))
```

```
 - Tesla is an American multinational automotive and clean energy company
            - Headquartered in Austin, Texas
            - Designs and manufactures electric vehicles, battery energy storage d
            - One of the world's most valuable companies
            - Led the battery electric vehicle market in 2022
            - Installed 6.5 gigawatt-hours of battery energy storage systems in 20
            - Founded in 2003 by Martin Eberhard and Marc Tarpenning
            - Elon Musk became CEO in 2008
            - Mission is to help expedite the move to sustainable transport and en
            - Produces the Model S sedan, Model X SUV, Model 3 sedan, Model Y cros
            - Plans production of the Cybertruck light-duty pickup truck in 2023
```

```
- Cumulative sales totaled 4 million cars as of April 2023
- Market capitalization temporarily reached $1 trillion in 2
```

sets up a task to detect named entities within a given text using the PromptTemplate and LLMChain classes from the langchain library.

The PromptTemplate lays out instructions for the language model to identify named entities and return results in JSON format, with details like the entity name, its type, and its location within the text. The placeholder {text} in the template will be replaced with the actual text to analyze.

The LLMChain class then binds the prepared prompt template with the specified language model (LLM). The resulting system aims to receive a text, identify the named entities in it, and return a detailed JSON file outlining each identified entity.

```
template = """
               Detect named entities in following text delimited by triple backquo
               Return your response in json format with spans of named entities wi
               Return all entities
               ```{text}```
               json format file:
           """

prompt = PromptTemplate(template=template, input_variables=["text"])

llm_chain = LLMChain(prompt=prompt, llm=llm)
print(llm_chain.run(text2))
```

```
 {
"namedEntity": [
{
"namedEntity": "Apple Inc.",
"type": "company",
"span": [
  {"start": 0, "end": 87}
]
```
```

```
  },
  {
  "namedEntity": "Steve Jobs",
  "type": "person",
  "span": [
    {"start": 109, "end": 141}
  ]
  },
  {
  "namedEntity": "Steve Wozniak",
  "type": "person",
  "span": [
    {"start": 168, "end": 183}
  ]
  },
  {
  "namedEntity": "Ronald Wayne",
  "type": "person",
  "span": [
```

Conclusion :

🚀 You are now well-positioned to begin experimenting with the massive potential that the collaboration of LLAM 2 and LangChain will bring to the industry!

Reference :

https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
https://ai.meta.com/blog/large-language-model-llama-meta-ai/
https://python.langchain.com/docs/get_started/introduction.html
https://www.youtube.com/watch?v=fc7cAP5zrOY

> *📹 Stay tuned for more details on trending AI-related implementations and discussions on my personal blog and if you are not a medium member and you would like unlimited articles to the platform. Consider using my referral link right here to sign up — it's less than the price of a fancy coffee, only $5 a month! Dive in, the knowledge is fine!*
>
> *🧬 We are AI application experts! If you want to collaborate on a project, drop an inquiry here, stop by our website, or book a consultation with us.*

📚Feel free to check out my other articles:

AI Tutorial: How To Launch The AI Chat App In Streamlit Using Open AI And GitHub

In This Article, We are going to Talk about a service called “Streamlit”. It’s great for what we need. Even a beginner...

pub.towardsai.net

Talk To Your CSV: How To Visualize Your Data With Langchain And Streamlit

Large model models (LLMS) have progressively grown powerful and competently. These models can be used for a wide array...

levelup.gitconnected.com

Level Up Coding

Thanks for being a part of our community! Before you go:

- 👏 Clap for the story and follow the author 👉
- 📰 View more content in the Level Up Coding publication

🔔 Follow us: Twitter | LinkedIn | Newsletter

🧠 AI Tools ⇒ Become a AI prompt engineer

Data Science | Aritificial Intelligence | Machine Learning | Programming | Technology

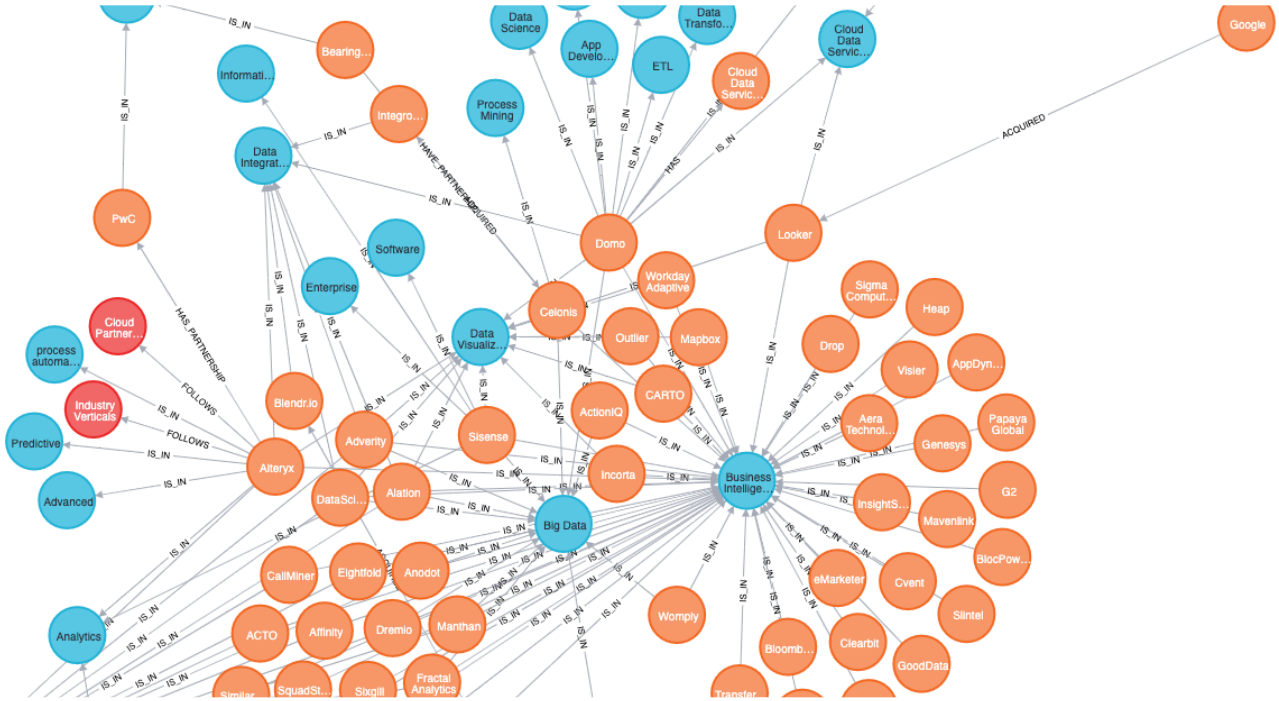
Follow

Written by Tarik Kaoutar (高達烈)

682 Followers · Writer for Level Up Coding

python software developer | Artificial Intelligent | Langchain I'll help you gain knowledge for an easier experience

More from Tarik Kaoutar (高達烈) and Level Up Coding



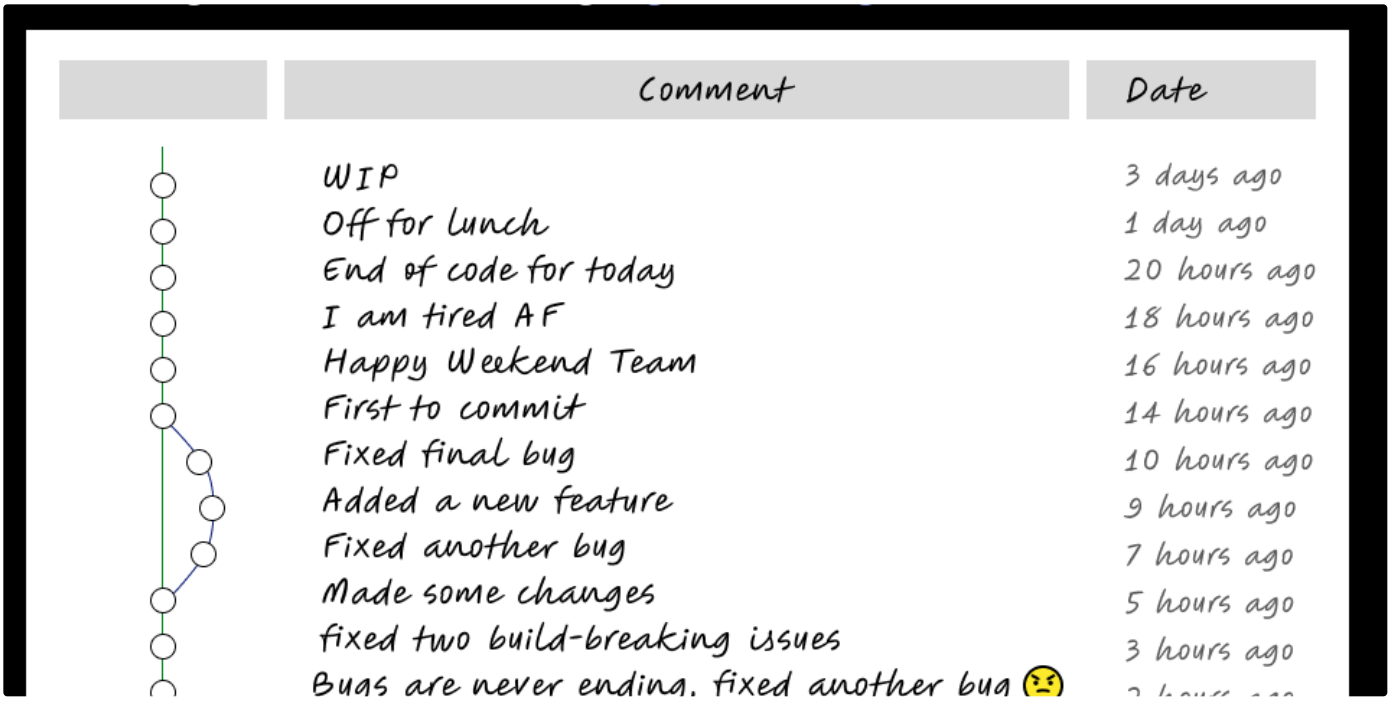
Tarik Kaoutar (高達烈) in DataDrivenInvestor

Graph your Document: How To Use Langchain and Matplotlib To Build a Knowledge Graph

In this easy-to-follow guide, we are going to see a complete example of how to use Langchain and Matplotlib To Build a Knowledge Graph.

· 4 min read · Sep 23

70 1



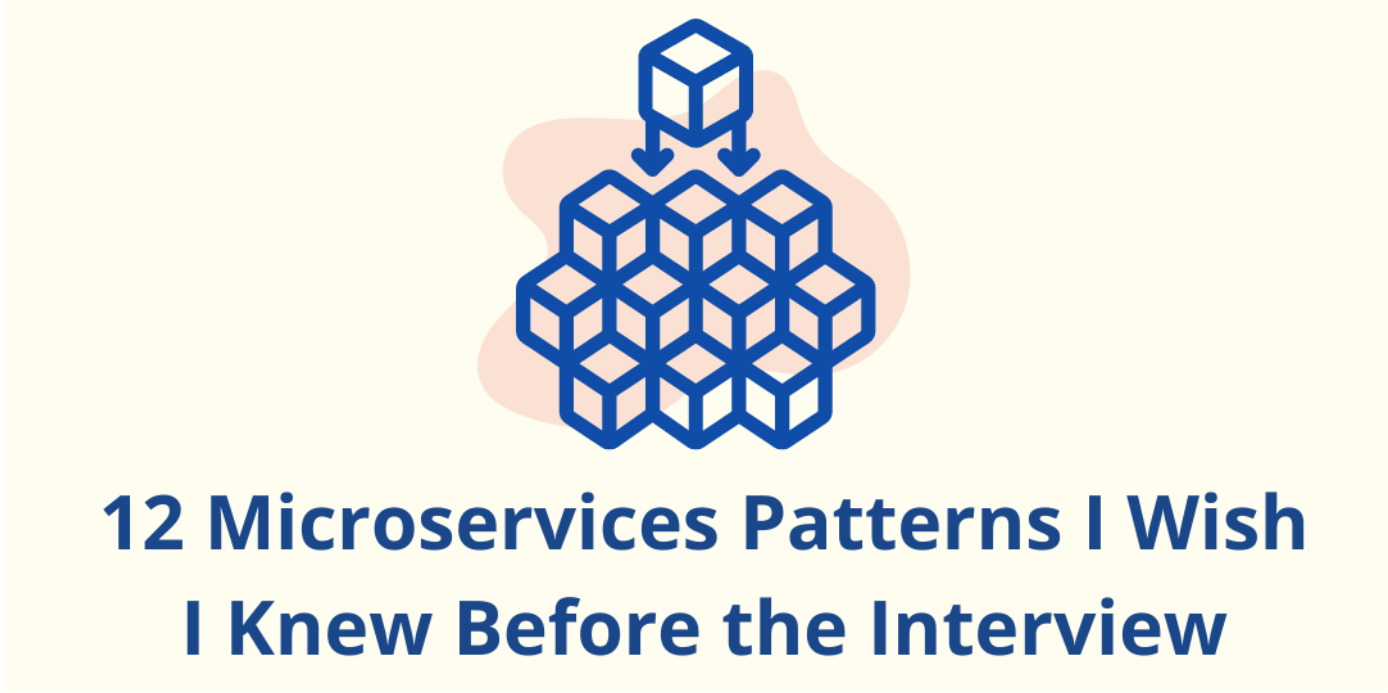
Victor Timi in Level Up Coding

“Good Commit” vs “Your Commit”: How to Write a Perfect Git Commit Message

A good commit shows whether a developer is a good collaborator—Peter Hutterer, Linux.

· 8 min read · Sep 5

2.3K 27



Arslan Ahmad in Level Up Coding

12 Microservices Patterns I Wish I Knew Before the System Design Interview

Mastering the Art of Scalable and Resilient Systems with Essential Microservices Design Patterns

13 min read · May 16

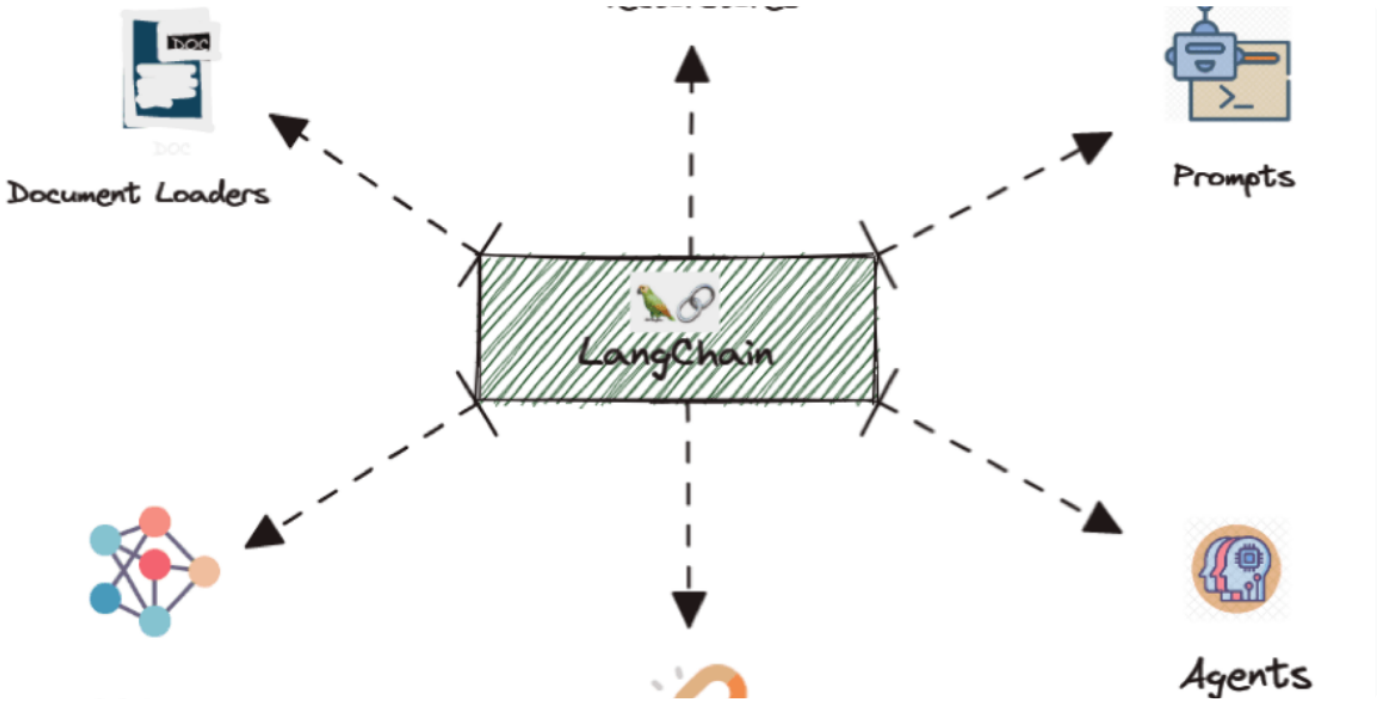
Tarik Kaoutar (高達烈) in Level Up Coding

LlamaIndex Last Version: From Basics To Advanced Techniques In Python - (Part-1)

In we will dig deeper into the intricate processes of LlamaIndex, using sample code as an example, this will help you to understand how to...

· 5 min read · Aug 22

Recommended from Medium



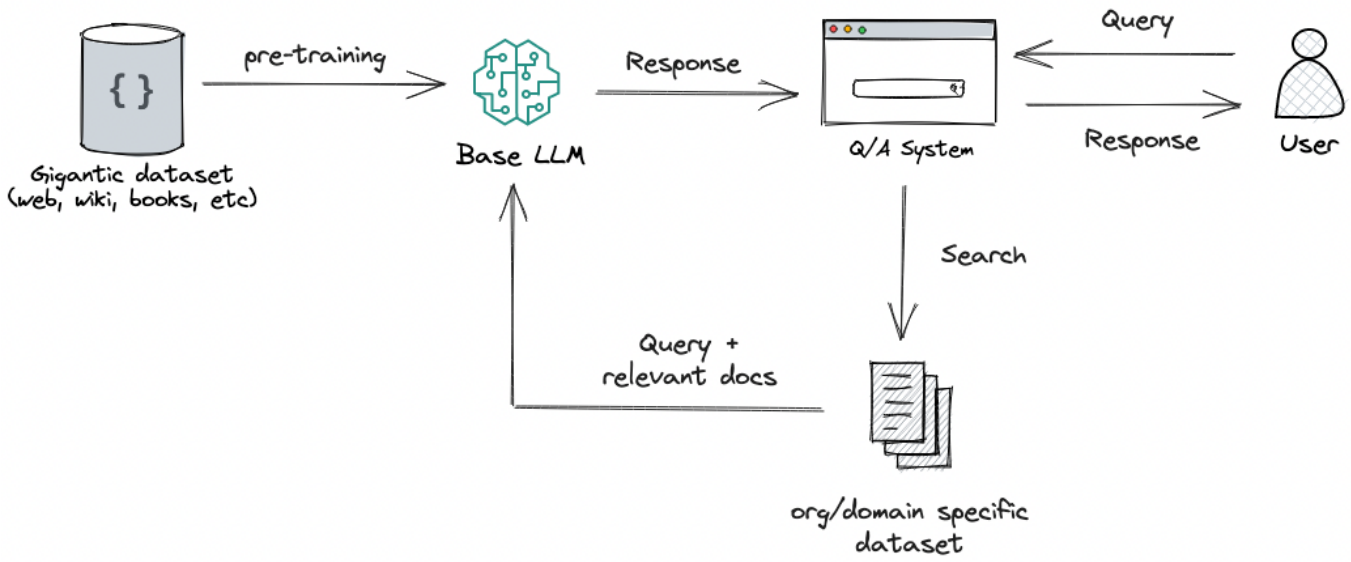
Zeeshan Malik

Connecting ChatGPT with your own Data using Llama Index and LangChain

In the last three months, there has been a rapid increase in the use of Large Language Models (LLMs) for a variety of applications, such as...

5 min read · Jun 11

102 2



Heiko Hotz in Towards Data Science

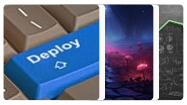
RAG vs Finetuning — Which Is the Best Tool to Boost Your LLM Application?

The definitive guide for choosing the right method for your use case

· 19 min read · Aug 25

2.2K 16

Lists



Predictive Modeling w/ Python

20 stories · 431 saves

Practical Guides to Machine Learning

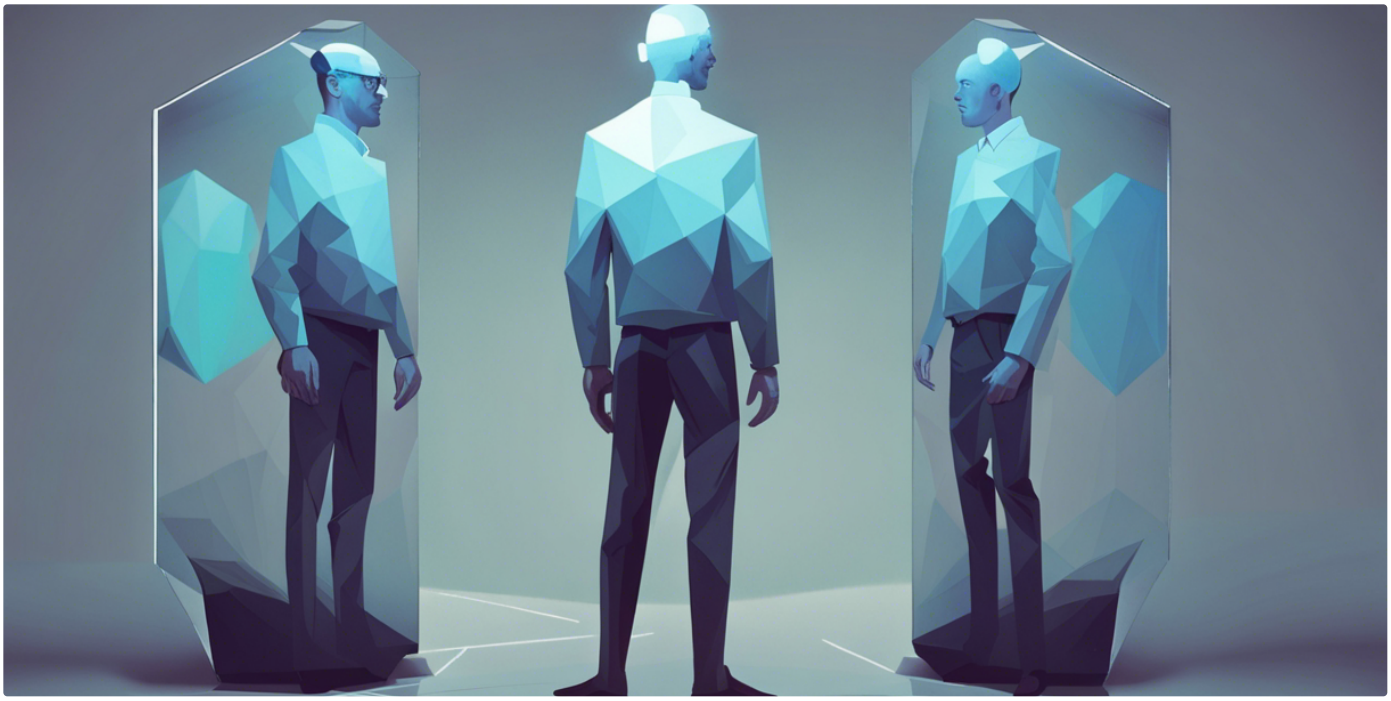
10 stories · 499 saves

ChatGPT prompts

24 stories · 439 saves

Coding & Development

11 stories · 197 saves



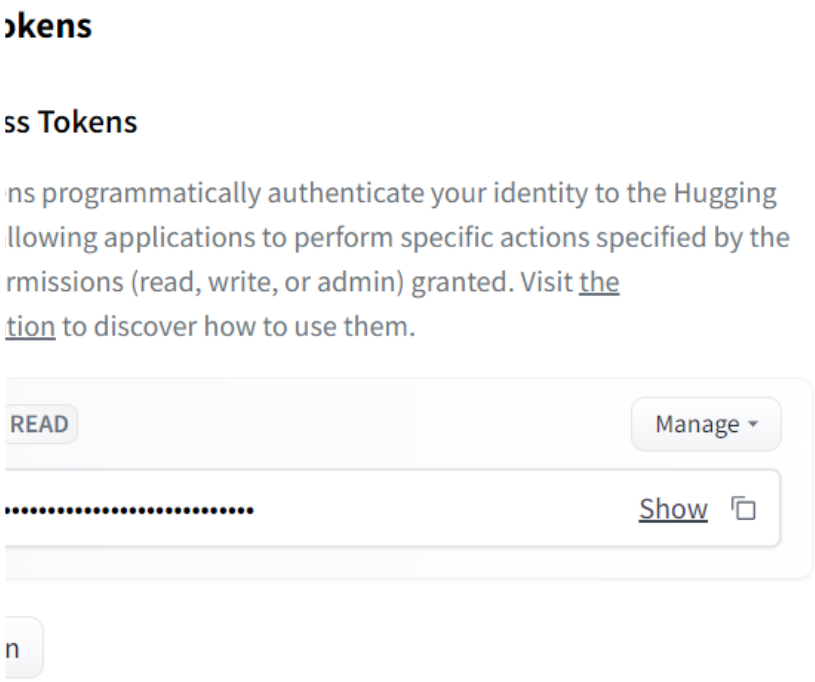
Sergei Savvov in Better Programming

Create a Clone of Yourself With a Fine-tuned LLM

Unleash your digital twin

11 min read · Jul 28

2.3K 19



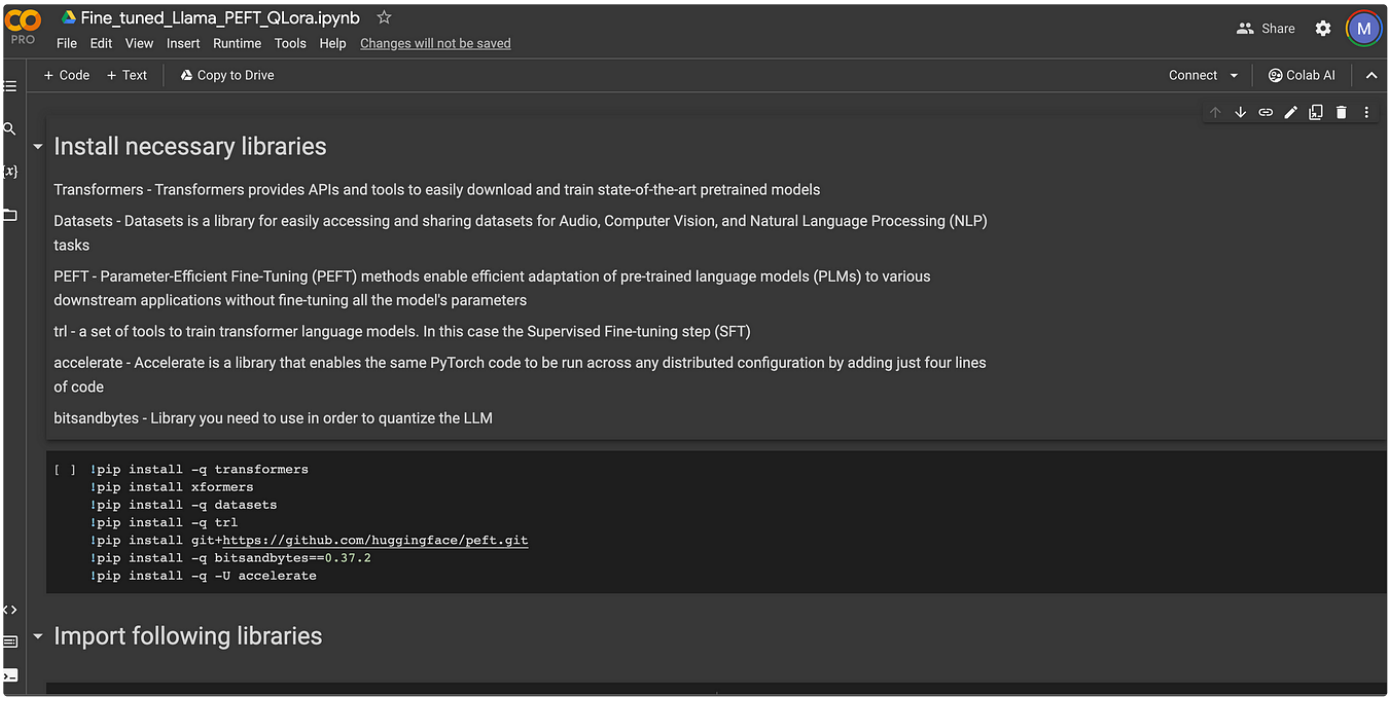
Ankit

Generating Summaries for Large Documents with Llama2 using Hugging Face and Langchain

Introduction

11 min read · Aug 28

99 3



Maya Akim

Complete Guide to LLM Fine Tuning for Beginners

Fine-tuning a model refers to the process of adapting a pre-trained, foundational model (such as Falcom or Llama) to perform a new task or...

5 min read · Aug 14

94 1

Maximilian Vogel in MLearning.ai

The ChatGPT list of lists: A collection of 3000+ prompts, examples, use-cases, tools, APIs...

Updated Sep-09, 2023. Added new prompt engineering courses, masterclasses and tutorials.

10 min read · Feb 8

8.7K 114

See more recommendations