Dataintelliage

YOUTUBE Script generator GPT

Generating script for your youtube video with the help of Langchain and LLM.

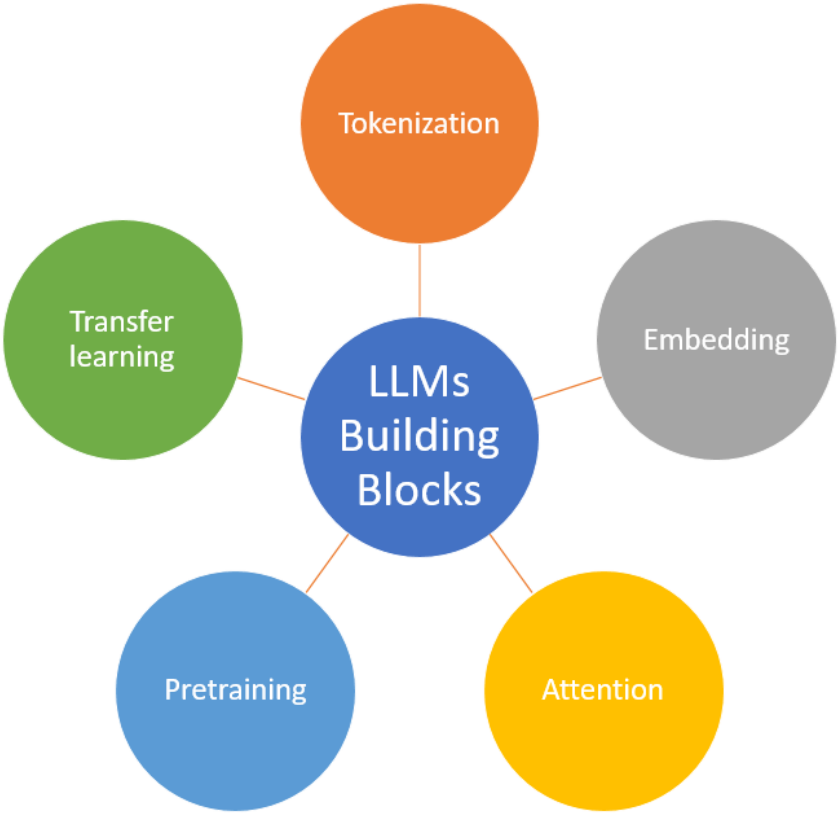
1. What is Langchain?

LangChain is an open-source framework designed to simplify the creation of applications using large language models (LLMs). It provides a standard interface for chains, lots of integrations with other tools, and end-to-end chains for common applications. It allows AI developers to develop applications based on the combined Large Language Models (LLMs) such as GPT-4 with external sources of computation and data. This framework comes with a package for both Python and JavaScript.

2.LLM (Large Language Model)

A **large language model** is a type of artificial intelligence algorithm that applies neural network techniques with lots of parameters to process and understand human

languages or text using self-supervised learning techniques. Tasks like text generation, machine translation, summary writing, image generation from texts, machine coding, chat-bots, or Conversational AI are applications of the Large Language Model. Examples of such LLM models are Chat GPT by open AI, BERT (Bidirectional Encoder Representations from Transformers) by Google, etc.



4.Procedure

-creating separate environment

-installing dependencies

-importing langchain, LLM and all other dependencies

-you have to signin to OpenAI to generate apikey

-you also have to install streamlit for the interface

YouTube GPT Creator

Plug in your prompt here

deep learning

"Unlock the Mysteries of Deep Learning: A Beginner's Guide"

Script: Welcome to the world of deep learning! In this video, we'll be exploring the mysteries of this powerful technology and how you can use it to your advantage.

We'll start by understanding two of the most important concepts in deep learning – artificial neural networks and deep reinforcement learning. Artificial neural networks are inspired by biological brains, but are static and symbolic. On the other hand, deep reinforcement learning is the combination of reinforcement learning and deep learning and allows agents to make decisions from unstructured data.

Finally, we'll take a look at Q-Learning, a model-free reinforcement learning algorithm to learn the value of an action in a particular state. This technique has been beneficial for a range of applications, including robotics, video games, natural language processing, and healthcare.

So, join us on this journey as we unlock the mysteries of deep learning and explore some of the amazing ways it can be used.

Title History

In [ ]:

```
import os
from apikey import apikey

import streamlit as st
from langchain.llms import OpenAI
from langchain.prompts import PromptTemplate
from langchain.chains import LLMChain, SequentialChain
from langchain.memory import ConversationBufferMemory
from langchain.utilities import WikipediaAPIWrapper

os.environ['OPENAI_API_KEY'] = apikey

# App framework
st.title('🦜🔗 YouTube GPT Creator')
prompt = st.text_input('Plug in your prompt here')

# Prompt templates
title_template = PromptTemplate(
    input_variables = ['topic'],
    template='write me a youtube video title about {topic}'
)

script_template = PromptTemplate(
    input_variables = ['title', 'wikipedia_research'],
    template='write me a youtube video script based on this title TITLE: {titl
)

# Memory
title_memory = ConversationBufferMemory(input_key='topic', memory_key='chat_hi
script_memory = ConversationBufferMemory(input_key='title', memory_key='chat_h


# Llms
llm = OpenAI(temperature=0.9)
title_chain = LLMChain(llm=llm, prompt=title_template, verbose=True, output_ke
script_chain = LLMChain(llm=llm, prompt=script_template, verbose=True, output_

wiki = WikipediaAPIWrapper()

# Show stuff to the screen if there's a prompt
if prompt:
    title = title_chain.run(prompt)
    wiki_research = wiki.run(prompt)
    script = script_chain.run(title=title, wikipedia_research=wiki_research)

    st.write(title)
    st.write(script)

    with st.expander('Title History'):
        st.info(title_memory.buffer)

    with st.expander('Script History'):
        st.info(script_memory.buffer)

    with st.expander('Wikipedia Research'):
```