llama2-final

January 4, 2024

1 AI Based Chatbot to answer DevOps Questions

1.0.1 We intend to create LLM based AI Chatbot that would answer the questions about Linux

It is simple LLM bot that answers the questions only on the trained dataset unlike RAG. We will train the data on the corpus of few Linux books like Linux Bible, etc. The vectorstore to store the learnings is FAISS database.

1.0.2 Technology Used

1. **LLM**: meta-llama/Llama-2-7b-chat-hf [https://huggingface.co/meta-llama/Llama-2-7b-chat-hf]
2. **VectorStore**: ***FAISS*** => FAISS (Facebook AI Similarity Search) is a library that allows developers to quickly search for embeddings of multimedia documents that are similar to each other [https://ai.meta.com/tools/faiss/]
3. **Embeddings**: ***sentence-transformers/all-mpnet-base-v2*** [https://huggingface.co/sentence-transformers/all-mpnet-base-v2]

1.0.3 Installing dependencies using pip

[1]:
```
!pip install -i https://test.pypi.org/simple/ bitsandbytes --quiet
!pip install -r requirements.txt --quiet
```

1.0.4 Necessary Imports

[ ]:
```
import os
from typing import Any
from langchain.globals import set_debug, set_verbose
```

1.0.5 Neccessary Settings

[3]:
```
import torch
from torch import cuda, bfloat16

if torch.cuda.is_available():
            torch.set_default_tensor_type(torch.cuda.HalfTensor)
```

```
device = f'cuda:{cuda.current_device()}' if cuda.is_available() else 'cpu'
```

1.0.6 Loading the model

[15]:
```
from transformers import AutoTokenizer, AutoModelForCausalLM
from transformers import pipeline
from langchain.llms import HuggingFacePipeline

def load_llama(model_id: Any) -> Any:
    print(f"Loading the model {model_id}")
    tokenizer = AutoTokenizer.from_pretrained(model_id)
    model = AutoModelForCausalLM.from_pretrained(model_id,
                                                 pad_token_id=tokenizer.
 ↪eos_token_id
                                                )
    model.to(device)
    llama_pipeline = pipeline(
        model=model,
        tokenizer=tokenizer,
        return_full_text=True,
        max_new_tokens=512,
        temperature=0.7,
        task="text-generation",  # LLM task
        torch_dtype=torch.float16,
        device_map="auto",
    )
    llm = HuggingFacePipeline(pipeline=llama_pipeline)
    return llm
```

1.0.7 Logging in to HuggingFace repo

[5]:
```
import huggingface_hub
def huggingface_login() -> None:
     os.environ["HF_TOKEN"] = "YOUR_HUGGINGFACE_TOKEN"
     print("------------------------------------")
     print("Huggingface login")
     huggingface_hub.login(os.environ["HF_TOKEN"])
     print("------------------------------------\n")
```

1.0.8 Setting the embeddings

[8]:
```
from langchain.embeddings import HuggingFaceEmbeddings

def set_embeddings(model_name: Any) -> Any:
     print(f"\n------------------------------------")
     print(f"Setting the embeddings")
```

```
    embeddings = HuggingFaceEmbeddings(model_name=model_name)
    print(f"Embeddings set successfully")
    print(f"------------------------------------\n")
    return embeddings
```

1.0.9 Loading the documents

[ ]:
```
from langchain.document_loaders import PyPDFLoader, DirectoryLoader

def load_documents(local_directory_path: str) -> Any:
    # For PDF files
    print(f"\n------------------------------------")
    print(f"Loading PDFs from {local_directory_path}")
    loader = DirectoryLoader(local_directory_path,
                             glob='*.pdf',
                             loader_cls=PyPDFLoader)
    print(loader)
    documents = loader.load()
    print(f"Documents Loaded")
    print(f"------------------------------------\n")
    return documents
```

1.0.10 Processing the documents

[7]:
```
from langchain.text_splitter import RecursiveCharacterTextSplitter


def process_documents(documents: Any) -> Any:
    print(f"\n------------------------------------")
    print(f"Processing the documents")
    text_splitter = RecursiveCharacterTextSplitter(chunk_size=1000,
                                                   chunk_overlap=50)
    texts = text_splitter.split_documents(documents)
    print(f"Documents processed")
    print(f"------------------------------------\n")
    return texts
```

1.0.11 Saving to FAISS Vectorstore

[9]:
```
from langchain.vectorstores import FAISS

def save_to_vectorstore(texts: Any, embeddings: Any, vectorestore_path: str) ->␣
 ↪Any:
    print(f"\n------------------------------------")
    print(f"Saving the vectorestore to {vectorestore_path}")
    vectorstore = FAISS.from_documents(texts, embeddings)
    vectorstore.save_local(vectorestore_path)
    print(f"Vectore DB stored at {vectorestore_path}")
    print(f"------------------------------------\n")
    return vectorstore
```

1.0.12 Setting the custom prompt template

[10]:
```
from langchain import PromptTemplate


custom_prompt_template = """Use the following information to answer the user's␣
 ↪question.
In case you don't know the answer, just say that you don't know, don't try to␣
 ↪make up an answer.

Context: {context}
Question: {question}

Only return the helpful answer below and nothing else.
Helpful answer:
"""

def set_custom_prompt() -> Any:
    """
    Prompt template for QA retrieval for each vectorstore
    """
    print("Setting the custom prompt")
    prompt = PromptTemplate(template=custom_prompt_template,
                            input_variables=['context', 'question'])
    return prompt
```

1.0.13 Retrieval QA Chain function

The RetrievalQAChain is a chain that combines a Retriever and a QA chain (described above). It is used to retrieve documents from a Retriever and then use a QA chain to answer a question based on the retrieved documents. Read this for more info on RetrievalQA

[11]:
```
from langchain.chains import RetrievalQA

def retrieval_qa_chain(llm: Any, vectorstore: Any) -> Any:
     chain = RetrievalQA.from_chain_type(llm=llm,
                                         chain_type='stuff',
                                         retriever=vectorstore.
 ↪as_retriever(search_kwargs={'k': 2}),
                                         return_source_documents=True,
                                         chain_type_kwargs={'prompt':␣
 ↪set_custom_prompt()},
                                         verbose=False
                                         )
     return chain
```

1.0.14 Chatbot Query

```
[12]: def chatbot(llm: Any, vectorstore: Any) -> Any:
          chain = retrieval_qa_chain(llm, vectorstore)
          exit_conditions = ("exit", "bye", "quit", ":q")
          while True:
              user_input = input("User: ")

              if user_input.lower() in exit_conditions:
                  print("Chatbot: Thanks!")
                  break
              result = chain({"query": user_input})

              response = result["result"]

              print("Chatbot: ", response)
              print("--------------------------------------------------\n\n")
```

1.0.15 Sample testing. Having the debugger output on for understanding the flow of responses

```
[ ]: if __name__ == "__main__":
         set_hf_token()
         documents = load_documents("/home/sagemaker-user/content/corpus")
         texts = process_documents(documents)
         embeddings = set_embeddings('sentence-transformers/all-mpnet-base-v2')
         vectorstore = save_to_vectorstore(texts, embeddings, "/home/sagemaker-user/
      ↪content/vectorstore/")
```

```
------------------------------------
Huggingface login
Token will not been saved to git credential helper. Pass
`add_to_git_credential=True` if you want to set the git credential as well.
Token is valid (permission: write).
Your token has been saved to /home/sagemaker-user/.cache/huggingface/token
Login successful
------------------------------------



------------------------------------
Loading PDFs from /home/sagemaker-user/content/corpus
<langchain.document_loaders.directory.DirectoryLoader object at 0x7f2cbdd4dc60>
Documents Loaded
```

```
------------------------------------

------------------------------------
Processing the documents
Documents processed
------------------------------------

------------------------------------
Setting the embeddings
Embeddings set successfully
------------------------------------

------------------------------------
Saving the vectorestore to /home/sagemaker-user/content/vectorstore/
Vectore DB stored at /home/sagemaker-user/content/vectorstore/
------------------------------------
```

[16]:
```
set_debug(True)
set_verbose(True)
chatbot(load_llama("meta-llama/Llama-2-7b-chat-hf"), vectorstore)
```

```
Loading the model meta-llama/Llama-2-7b-chat-hf

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

Setting the custom prompt

User:  What is linux

[chain/start] [1:chain:RetrievalQA] Entering Chain run with

input:

{
  "query": "What is linux"
}
[chain/start] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain] Entering Chain run with input:

[inputs]
[chain/start] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain] Entering Chain run with input:

{
  "question": "What is linux",
  "context": "other hand, was developed in a different context. Linux is a PC
version of the Unix operating system that has been used for decades on
mainframes and minicomputers and is currently the system of choice for network
servers and workstations. Linux brings the \nspeed, efficiency, scalability, and
flexibility of Unix to your PC, taking advantage of all the \ncapabilities that
PCs can now provide.\nTechnically, Linux consists of the operating system
program, referred to as the kernel,\n\nLinux
   \n \n         \n \n      "
}
[llm/start] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain > 5:llm:HuggingFacePipeline]

Entering LLM run with input:

{
  "prompts": [
    "Use the following information to answer the user's question.\nIn case you
don't know the answer, just say that you don't know, don't try to make up an
answer.\n\nContext: other hand, was developed in a different context. Linux is a
PC version of the Unix operating system that has been used for decades on
mainframes and minicomputers and is currently the system of choice for network
servers and workstations. Linux brings the \nspeed, efficiency, scalability, and
flexibility of Unix to your PC, taking advantage of all the \ncapabilities that
PCs can now provide.\nTechnically, Linux consists of the operating system
program, referred to as the kernel,\n\nLinux
   \n \n         \n \n      \nQuestion: What is linux\n\nOnly
return the helpful answer below and nothing else.\nHelpful answer:"
  ]
}
[llm/end] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain > 5:llm:HuggingFacePipeline]

[16.91s] Exiting LLM run with output:

{
  "generations": [
    [
      {
        "text": "Linux is an open-source operating system that is based on the
Unix operating system and is designed to be fast, efficient, scalable, and
flexible. It is typically used on servers and workstations, but can also be used
on personal computers.",
        "generation_info": null
      }
    ]
  ],
  "llm_output": null,
  "run": null
}
```

```
[chain/end] [1:chain:RetrievalQA >
3:chain:StuffDocumentsChain > 4:chain:LLMChain] [16.92s] Exiting Chain run with
output:
{
  "text": "Linux is an open-source operating system that is based on the Unix
operating system and is designed to be fast, efficient, scalable, and flexible.
It is typically used on servers and workstations, but can also be used on
personal computers."
}
[chain/end] [1:chain:RetrievalQA >
3:chain:StuffDocumentsChain] [16.92s] Exiting Chain run with output:
{
  "output_text": "Linux is an open-source operating system that is based on the
Unix operating system and is designed to be fast, efficient, scalable, and
flexible. It is typically used on servers and workstations, but can also be used
on personal computers."
}
[chain/end] [1:chain:RetrievalQA] [16.95s] Exiting Chain
run with output:
[outputs]
Chatbot:  Linux is an open-source operating system that is based on the Unix
operating system and is designed to be fast, efficient, scalable, and flexible.
It is typically used on servers and workstations, but can also be used on
personal computers.
--------------------------------------------------


User:  exi
[chain/start] [1:chain:RetrievalQA] Entering Chain run with
input:
{
  "query": "exi"
}
[chain/start] [1:chain:RetrievalQA >
3:chain:StuffDocumentsChain] Entering Chain run with input:
[inputs]
[chain/start] [1:chain:RetrievalQA >
3:chain:StuffDocumentsChain > 4:chain:LLMChain] Entering Chain run with input:
{
  "question": "exi",
  "context": "expressions, evaluating, 133-135, 443,\n620, 647-650\next2
filesystem\nchecking and repairing, 125\ndebugging, 107-110\nformatting devices
as, 288-290\nlabel for, displaying, 127\nprinting block and
superblock\ninformation, 124\nresizing, 356\nstoring disaster recovery
data\nfor, 126\ntuning parameters of, 457-460\next3 filesystem, 16\ndebugging,
107-110\nformatting devices as, 291\nprinting block and superblock\ninformation,
124\nextended Internet services\ndaemon, 490-493\nextended regular expressions,
searching\nwith, 128\nExtensible Filesystem (see XFS)\nextension command,
logrotate, 254\nextension() function, gawk, 740\nExterior Gateway Protocol
(EGP), 25\neXternal Data Representation (XDR), 32\n\n266, 271\nWindows Ubuntu
Installer (Wubi), 24, 42, 53Windows Vista, 27-28, 33, 36Windows XP, 28-29, 33,
36\nWinzip program, 299wireless card compatibility, 34wireless networks,
150-151, 152-153wireless signal meter gadget, 290WMV (Windows Media Video) video
format, \n266, 271\nword processing applications, 201, 219. See also \nWriter,
OpenOffi  ce.org\nWorkspace Switcher, 72, 73\nworkspaces, moving programs
between, 72, 73wrapping of command lines, 3write permissions, 116Writer,
OpenOffi  ce.org, 213, 214, 215, \n218-219, 293\nw32codec, 266Wubi (Windows
Ubuntu Installer), 24, 42, 53WWW services, 360\n• X •\nX Window System (X), 110,
111Xandros specialized distribution, 20X-Chat IRC program, 181XFS fi  lesystem,
48\nX.org.0.log fi  le, 362\n• Y •\nYouTube, 267, 293yum software installer,
282-283, 298, 300, 357\n• Z •\n.z fi les, 299\n.zip fi  les, 299\nzip program,
299, 411"
}
[llm/start] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain > 5:llm:HuggingFacePipeline]

Entering LLM run with input:

{
  "prompts": [
    "Use the following information to answer the user's question.\nIn case you
don't know the answer, just say that you don't know, don't try to make up an
answer.\n\nContext: expressions, evaluating, 133-135, 443,\n620, 647-650\next2
filesystem\nchecking and repairing, 125\ndebugging, 107-110\nformatting devices
as, 288-290\nlabel for, displaying, 127\nprinting block and
superblock\ninformation, 124\nresizing, 356\nstoring disaster recovery
data\nfor, 126\ntuning parameters of, 457-460\next3 filesystem, 16\ndebugging,
107-110\nformatting devices as, 291\nprinting block and superblock\ninformation,
124\nextended Internet services\ndaemon, 490-493\nextended regular expressions,
searching\nwith, 128\nExtensible Filesystem (see XFS)\nextension command,
logrotate, 254\nextension() function, gawk, 740\nExterior Gateway Protocol
(EGP), 25\neXternal Data Representation (XDR), 32\n\n266, 271\nWindows Ubuntu
Installer (Wubi), 24, 42, 53Windows Vista, 27-28, 33, 36Windows XP, 28-29, 33,
36\nWinzip program, 299wireless card compatibility, 34wireless networks,
150-151, 152-153wireless signal meter gadget, 290WMV (Windows Media Video) video
format, \n266, 271\nword processing applications, 201, 219. See also \nWriter,
OpenOffi  ce.org\nWorkspace Switcher, 72, 73\nworkspaces, moving programs
between, 72, 73wrapping of command lines, 3write permissions, 116Writer,
OpenOffi  ce.org, 213, 214, 215, \n218-219, 293\nw32codec, 266Wubi (Windows
Ubuntu Installer), 24, 42, 53WWW services, 360\n• X •\nX Window System (X), 110,
111Xandros specialized distribution, 20X-Chat IRC program, 181XFS fi  lesystem,
48\nX.org.0.log fi  le, 362\n• Y •\nYouTube, 267, 293yum software installer,
282-283, 298, 300, 357\n• Z •\n.z fi les, 299\n.zip fi  les, 299\nzip program,
299, 411\nQuestion: exi\n\nOnly return the helpful answer below and nothing
else.\nHelpful answer:"
  ]
}
[llm/end] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain > 5:llm:HuggingFacePipeline]

[11.14s] Exiting LLM run with output:

{
  "generations": [
    [
      {
        "text": "I don't know the answer to that question.",
        "generation_info": null
      }
    ]
  ],
  "llm_output": null,
  "run": null
}
[chain/end] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain > 4:chain:LLMChain] [11.15s] Exiting Chain run with

output:

{
  "text": "I don't know the answer to that question."
}
[chain/end] [1:chain:RetrievalQA >

3:chain:StuffDocumentsChain] [11.15s] Exiting Chain run with output:

{
  "output_text": "I don't know the answer to that question."
}
[chain/end] [1:chain:RetrievalQA] [11.18s] Exiting Chain

run with output:

[outputs]
Chatbot:  I don't know the answer to that question.
--------------------------------------------------
```

```
User:  Exit

Chatbot: Thanks!
```

1.1 Final testing

```
[18]: chatbot(load_llama("meta-llama/Llama-2-7b-chat-hf"), vectorstore)
```

```
Loading the model meta-llama/Llama-2-7b-chat-hf

Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]

Setting the custom prompt

User:  What is cloud computing

Chatbot:  Cloud computing refers to the practice of deploying applications and
services on the internet with a cloud provider.
--------------------------------------------------



User:  What is AWS

Chatbot:  AWS stands for Amazon Web Services. It is a cloud computing platform
offered by Amazon that provides a range of services for computing, storage,
networking, database, analytics, machine learning, and more.
--------------------------------------------------



User:  What are some of the major services in AWS

Chatbot:  Some of the major services in AWS include:

* Compute Services (EC2, Lambda, Elastic Beanstalk)
* Storage Services (S3, EBS, Elastic File System)
* Database Services (RDS, DynamoDB, Redshift)
* Security, Identity & Compliance Services (IAM, Cognito, Certificate Manager)
* Application Services (API Gateway, AppSync, CloudFront)
* Analytics Services (Redshift, QuickSight, Athena)
* Machine Learning Services (SageMaker, Comprehend, Rekognition)
* Networking & Connectivity Services (VPC, Elastic IP, Direct Connect)

Note: This list is not exhaustive and is based on my understanding of the
question.
--------------------------------------------------



User:  What is Devops
```

Chatbot: DevOps is a set of principles and practices that aims to bring together software developers and operations teams to collaborate and automate the software delivery process, with the goal of faster and more reliable end-to-end delivery of software systems to end customers.

User: Tell me about CI/CD

Chatbot: CI/CD stands for Continuous Integration/Continuous Deployment. It refers to the practice of integrating and testing code changes frequently, and automatically deploying those changes to production after they pass automated tests. The goal of CI/CD is to improve the efficiency and reliability of software delivery by reducing the time and effort required to move code from development to production.

User: list prominent ci/cd tools

Chatbot: Jenkins
Bamboo
GoCD
Team City
Electric Cloud

Please note that the list of prominent CI/CD tools is not exhaustive, and there are many other tools available in the market.

User: 10 important commands in linux

Chatbot:
10 important commands in Linux are:

1. ls (list files)
2. cd (change directory)
3. pwd (print working directory)
4. mv (move or rename files)
5. rm (remove files)
6. cp (copy files)
7. mkdir (create directory)
8. rmdir (remove directory)
9. echo (print text to a file)
10. cat (concatenate and display files)

Note: These are some of the most commonly used commands in Linux, but there are

many more available.

User: What is the difference between sudo -i and sudo su -

Chatbot: sudo -i and sudo su - are both used to run commands as a different user, but they have slightly different usage and behavior.

sudo -i allows you to run a command as a different user without a password, while sudo su - requires a password to switch users.

In general, it's recommended to use sudo -i when you need to run a command as a different user without a password, and to use sudo su - when you want to switch users and run a command with a password.

User: what is iphone

Chatbot: iPhone is a brand of smartphones developed by Apple Inc.

User: Who is the prime minister of INdia

Chatbot: I don't know the answer to that question. Sanjeev is a cloud and DevOps expert, not a political analyst.

User: Tell me something about networking in cloud

Chatbot: The cloud has had a significant impact on networking, as it has enabled the emergence of new networking models and technologies. For example, AWS and OpenStack provide networking services out of the box, such as VPCs, subnets, and security groups. These services have made networking a commodity much like infrastructure, allowing developers to consume networking resources on demand. Additionally, cloud networking has enabled the use of software-defined networking (SDN) and network functions virtualization (NFV) technologies, which can be used to programmatically control and automate network functions.

User: Bye

Chatbot: Thanks!

```
[ ]:
```