Q&A Model with PyTorch

Marc Anthony Atanante

Introduction

In this activity, we will deploy a question-answering model using a pre-trained model from HuggingFace, FastAPI and Docker (Credits to Andre Ribeiro for the documentation).

This will make use of a given context and extracts the best answer from that input. Here, we will use the Stanford QA Dataset 2.0 which can be downloaded here.

From the said data, we will focus on the question and answer fields where the topic is “Premier League”. This will provide us with the exact answers to a specific number of questions.

Step 1: Cloning the repository

The first step is to clone the following repository to your local machine. Make sure that you review and understand each file.

There are four main files required to create the Q&A API:

1. **app/main.py** - Main app file and docker entrypoint. This defines the FastAPI logic.
2. **app/utils.py** - Utility file for the model logic.
3. **download_model.sh** - Shell file that defines the model and required steps.
4. **Dockerfile** - Defines the steps to install all the required libraries, download the pretrained model and run the FastAPI app.
5. **test/test_app.ipynb** - Test the app set_context and get_answer endpoints.

Step 2: Download and Install Docker

Download Docker Desktop so that you can build and run the container. Since I am using Windows, I first installed and configured my Linux Subsystem to do the task.

After installation make sure that Docker will complete the setup with no errors. Otherwise, make sure that your Linux Subsystem is updated and WSL version is set to 2.0.

Step 3: Build the Docker Container

For someone using a Linux Subsystem, here are the steps that I did to complete the task using the Command Prompt:

1. Navigate to your local Windows directory using the **cd /mnt/c/Users** command.
2. Go to the folder where you cloned the repository and type **docker build . -t qamodel**. It should look like this:

```
[+] Building 7.3s (10/10) FINISHED
 => [internal] load build definition from Dockerfile                                                    0.0s
 => => transferring dockerfile: 38B                                                                     0.0s
 => [internal] load .dockerignore                                                                       0.0s
 => => transferring context: 2B                                                                         0.0s
 => [internal] load metadata for docker.io/library/python:3.8.12-slim-buster                            7.1s
 => [1/5] FROM docker.io/library/python:3.8.12-slim-buster@sha256:26ab58f6b8936fe59303b0ca0e915d5f9e071ef8b9bf7b3  0.0s
 => [internal] load build context                                                                       0.0s
 => => transferring context: 125B                                                                       0.0s
 => CACHED [2/5] RUN apt-get update &&    apt-get install -y --no-install-recommends wget &&    rm -rf /var/lib/a  0.0s
 => CACHED [3/5] COPY download_model.sh .                                                               0.0s
 => CACHED [4/5] RUN bash download_model.sh                                                             0.0s
 => CACHED [5/5] COPY app/ app/                                                                         0.0s
 => exporting to image                                                                                  0.0s
 => => exporting layers                                                                                 0.0s
 => => writing image sha256:c9bd1692c988fca4f5b4993cabaac46f0ba7a0b1318e844b99cc6c862362a919             0.0s
 => => naming to docker.io/library/qamodel                                                              0.0s
```

Step 3: Build the Docker Container

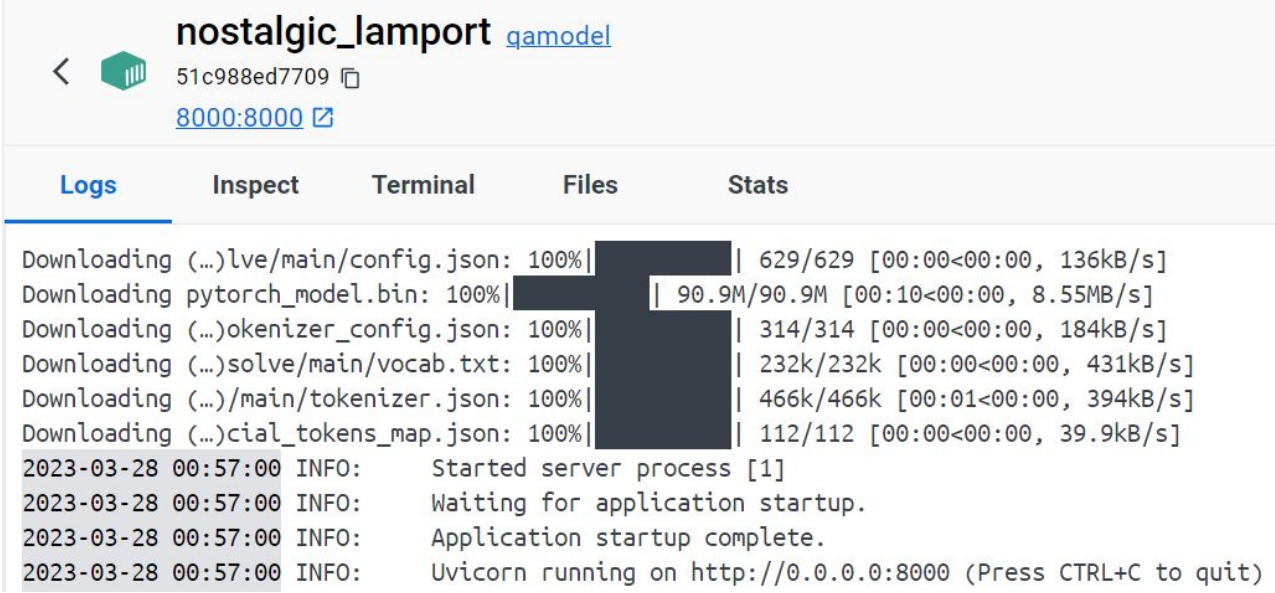
It is important to note that the process will pass or fail depending on your Dockerfile.

For this task, I encountered many errors (specifically 404) because of the **download_model.sh** shell file.

The model directory used way back in 2020 was "https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L6-v2/resolve/main" but it is not working anymore. I used "https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L6-v2/blob/main" instead and finally the docker container was built.

Step 3: Run the Docker Image

To run the docker image, type **docker run -p 8000:8000 qamodel**. A successful run looks like this:



Step 4: Other requirements and fixes

Make sure to download and install the necessary packages you need for the testing procedure (transformers, torch and requests).

Review the **model_name** parameter if it is consistent with the HuggingFace links. In our case, the model name should be "**sentence-transformers/paraphrase-MiniLM-L6-v2**".

To test the model locally, run first the shell file using the command **bash download_model.sh**. This will ensure that you have the model files in your local machine.

Step 5: Test the running app

To demonstrate the use of the Q&A model, I created a file called **test_file.py** instead of using the **test_app.ipynb** file that the author provided (see Code Snippets section).

Run the command **python3 test_file.py** and the output should look like this:

```
Type: <class 'dict'>
Length: 2
{'question': 'When did Beyonce start becoming popular?', 'id': '56be85543aeaaa14008c9063', 'answers': [{'text': 'in the late 1990s',
'answer_start': 269}], 'is_impossible': False}
Number of available questions: 357
Embeddings shape: torch.Size([3, 384])
tensor([71.4029, 59.8726, 23.9430])

 Testing the set_context and get_answer endpoints
{'message': 'Search context set'}

 Q&A Demo:
orig_q : How many teams compete in the Premier League ?
best_q : How many clubs are currently in the Premier League?
best_a : 20

orig_q : When does the Premier League starts and finishes ?
best_q : When does the Premier League have its playing season?
best_a : During the course of a season (from August to May)

orig_q : Who has the highest number of goals in the Premier League ?
best_q : Who has the record for most goals in the Premier League?
best_a : Newcastle United striker Alan Shearer holds the record for most Premier League goals with 260
```

Step 6: Hosting on AWS EC2

You can sign up and use Amazon's AWS EC2 virtual machines. However, if you do it for free you can only launch the eligible instance t3.micro.

1. Create an EC2 instance. In my case, I chose the free tier eligible one.
2. Install Docker using the command **sudo yum install docker -y**.
3. Start the docker service using **sudo service docker start**.
4. Add ec2-user to the docker-group using **sudo usermod -a -G docker ec2-user.**
5. In your local machine, push the Docker image to the Docker Hub. Login with your credentials through the command **docker login,** tag your image using the command **docker tag qamodel docker-username/qamodel** and then type **docker push docker-username/qamodel**.
6. I used **sudo growpart /dev/nvme0n1 1** to resize my partition and then extended the partition size using the command **sudo xfs_grow -d /**.
7. Pull the Docker image in EC2 using **docker pull docker-username/qamodel:latest**.
8. Launch the Docker image using **docker run -d -p 80:80 docker-username/qamodel**.

Step 6: Hosting on AWS EC2

aws Services Search [Alt+S]

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Mar 28 16:20:10 2023 from 13.48.4.202
[ec2-user@ip-172-31-10-73 ~]$ docker login
```

Step 6: Hosting on AWS EC2

```
[ec2-user@ip-172-31-10-73 ~]$ docker pull marcatanante/qamodel
Using default tag: latest
latest: Pulling from marcatanante/qamodel
15115158dd02: Pull complete
4d445d10bda3: Pull complete
7801333f6f71: Pull complete
f740bae08aac: Pull complete
1611ec958526: Pull complete
db75ed0c461a: Pull complete
8712ee46917f: Pull complete
b02f9958c64c: Pull complete
e2c95804422b: Pull complete
Digest: sha256:eed58a7f11bce4d5b8855bd62fb26e769907b2a7a7cd75c90a9b136d1d44a519
Status: Downloaded newer image for marcatanante/qamodel:latest
docker.io/marcatanante/qamodel:latest
[ec2-user@ip-172-31-10-73 ~]$ docker images
REPOSITORY              TAG       IMAGE ID       CREATED         SIZE
marcatanante/qamodel    latest    c9bd1692c988   42 hours ago    4.71GB
hello-world             latest    feb5d9fea6a5   18 months ago   13.3kB
```

Code Snippets: test_file.py

test_file.py > ...

```python
import json
import transformers
from transformers import AutoTokenizer
from transformers import AutoModel
import torch
import requests

with open("train-v2.0.json", 'r') as f:
  data = json.load(f)

print("Type:", type(data))
print("Length:", len(data))
print(data['data'][0]['paragraphs'][0]['qas'][0])

def get_qa(topic, data):
    q = []
    a = []
    for d in data['data']:
        if d['title']==topic:
            for paragraph in d['paragraphs']:
                for qa in paragraph['qas']:
                    if not qa['is_impossible']:
                        q.append(qa['question'])
                        a.append(qa['answers'][0]['text'])
            return q,a

questions, answers = get_qa(topic='Premier_League', data=data)

print("Number of available questions: {}".format(len(questions)))
```

test_file.py > ...

```python
def get_model(model_name):
    model = AutoModel.from_pretrained(model_name)
    tokenizer = AutoTokenizer.from_pretrained(model_name)
    return model, tokenizer

model, tokenizer = get_model(model_name="sentence-transformers/paraphrase-MiniLM-L6-v2")

# Mean Pooling - Take attention mask into account for correct averaging
# source: https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L6-v2
def mean_pooling(model_output, attention_mask):
    token_embeddings = model_output[0]

    input_mask_expanded = (
      attention_mask
      .unsqueeze(-1)
      .expand(token_embeddings.size())
      .float()
    )

    pool_emb = (
      torch.sum(token_embeddings * input_mask_expanded, 1)
      / torch.clamp(input_mask_expanded.sum(1), min=1e-9)
    )

    return pool_emb
```

Code Snippets: test_file.py

```python
def get_embeddings(questions, tokenizer, model):
    # Tokenize sentences
    encoded_input = tokenizer(questions, padding=True, truncation=True, return_tensors='pt')

    # Compute token embeddings
    with torch.no_grad():
        model_output = model(**encoded_input)

    # Average pooling
    embeddings = mean_pooling(model_output, encoded_input['attention_mask'])

    return embeddings

embeddings = get_embeddings(questions[:3], tokenizer, model)
print("Embeddings shape: {}".format(embeddings.shape))

new_question = 'Which days have the most events played at?'
new_embedding = get_embeddings([new_question], tokenizer, model)

# squared Euclidean distance between sample questions and new_question
print(((embeddings - new_embedding)**2).sum(axis=1))
```

```python
json_data = {
  'questions':questions,
  'answers':answers,
}

response = requests.post(
  'http://0.0.0.0:8000/set_context',
  json=json_data
)

print(response.json())
# Input new questions and expect the best answer
new_questions = [
    'How many teams compete in the Premier League ?',
    'When does the Premier League starts and finishes ?',
    'Who has the highest number of goals in the Premier League ?',
]

json_data = {
  'questions':new_questions,
}

response = requests.post(
  'http://0.0.0.0:8000/get_answer',
  json=json_data
)

print("Q&A Demo:")
for d in response.json():
  print('\n'.join(["{} : {}".format(k, v) for k,v in d.items()])+'\n')
```