Name Bealu Girma Gebresilassie

Roll no: 21053273 CSE-2

In [ ]:

```
!pip install transformers "datasets[s3]==2.18.0" "sagemaker>=2.190.0" "hug
```

```
Requirement already satisfied: transformers in /usr/local/lib/python
3.10/dist-packages (4.38.2)
Collecting transformers
  Downloading transformers-4.39.1-py3-none-any.whl (8.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 8.8/8.8 MB 18.6 MB/s et
a 0:00:00
Collecting datasets[s3]==2.18.0
  Downloading datasets-2.18.0-py3-none-any.whl (510 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 510.5/510.5 kB 31.3 MB/
s eta 0:00:00
Collecting sagemaker>=2.190.0
  Downloading sagemaker-2.214.1-py3-none-any.whl (1.4 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.4/1.4 MB 36.1 MB/s et
a 0:00:00
Requirement already satisfied: huggingface_hub[cli] in /usr/local/li
b/python3.10/dist-packages (0.20.3)
Collecting huggingface_hub[cli]
  Downloading huggingface_hub-0.22.1-py3-none-any.whl (388 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 388.6/388.6 kB 28.4 MB/
```

In [ ]:

```
!huggingface-cli login --token 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
```

```
Token has not been saved to git credential helper. Pass `add_to_git_cred
ential=True` if you want to set the git credential as well.
Token is valid (permission: read).
Your token has been saved to /root/.cache/huggingface/token
Login successful
```

In [ ]:
```
!aws configure
```

```
Requirement already satisfied: sagemaker in /usr/local/lib/python3.10/di
st-packages (2.214.1)
Requirement already satisfied: attrs<24,>=23.1.0 in /usr/local/lib/pytho
n3.10/dist-packages (from sagemaker) (23.2.0)
Requirement already satisfied: boto3<2.0,>=1.33.3 in /usr/local/lib/pyth
on3.10/dist-packages (from sagemaker) (1.34.72)
Requirement already satisfied: cloudpickle==2.2.1 in /usr/local/lib/pyth
on3.10/dist-packages (from sagemaker) (2.2.1)
Requirement already satisfied: google-pasta in /usr/local/lib/python3.1
0/dist-packages (from sagemaker) (0.2.0)
Requirement already satisfied: numpy<2.0,>=1.9.0 in /usr/local/lib/pytho
n3.10/dist-packages (from sagemaker) (1.25.2)
Requirement already satisfied: protobuf<5.0,>=3.12 in /usr/local/lib/pyt
hon3.10/dist-packages (from sagemaker) (3.20.3)
Requirement already satisfied: smdebug-rulesconfig==1.0.1 in /usr/local/
lib/python3.10/dist-packages (from sagemaker) (1.0.1)
Requirement already satisfied: importlib-metadata<7.0,>=1.4.0 in /usr/lo
cal/lib/python3.10/dist-packages (from sagemaker) (6.11.0)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python
3.10/dist-packages (from sagemaker) (24.0)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-
packages (from sagemaker) (1.5.3)
Requirement already satisfied: pathos in /usr/local/lib/python3.10/dist-
packages (from sagemaker) (0.3.2)
Requirement already satisfied: schema in /usr/local/lib/python3.10/dist-
packages (from sagemaker) (0.7.5)
Requirement already satisfied: PyYAML~=6.0 in /usr/local/lib/python3.10/
dist-packages (from sagemaker) (6.0.1)
Requirement already satisfied: jsonschema in /usr/local/lib/python3.10/d
ist-packages (from sagemaker) (4.19.2)
Requirement already satisfied: platformdirs in /usr/local/lib/python3.1
0/dist-packages (from sagemaker) (4.2.0)
Requirement already satisfied: tblib<4,>=1.7.0 in /usr/local/lib/python
3.10/dist-packages (from sagemaker) (3.0.0)
Requirement already satisfied: urllib3<3.0.0,>=1.26.8 in /usr/local/lib/
python3.10/dist-packages (from sagemaker) (2.0.7)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dis
t-packages (from sagemaker) (2.31.0)
Requirement already satisfied: docker in /usr/local/lib/python3.10/dist-
packages (from sagemaker) (7.0.0)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-pa
ckages (from sagemaker) (4.66.2)
Requirement already satisfied: psutil in /usr/local/lib/python3.10/dist-
packages (from sagemaker) (5.9.5)
Requirement already satisfied: botocore<1.35.0,>=1.34.72 in /usr/local/l
ib/python3.10/dist-packages (from boto3<2.0,>=1.33.3->sagemaker) (1.34.7
2)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /usr/local/lib/
python3.10/dist-packages (from boto3<2.0,>=1.33.3->sagemaker) (1.0.1)
Requirement already satisfied: s3transfer<0.11.0,>=0.10.0 in /usr/local/
lib/python3.10/dist-packages (from boto3<2.0,>=1.33.3->sagemaker) (0.10.
1)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.10/di
st-packages (from importlib-metadata<7.0,>=1.4.0->sagemaker) (3.18.1)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/li
b/python3.10/dist-packages (from requests->sagemaker) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.1
```

```
0/dist-packages (from requests->sagemaker) (3.6)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/pyth
on3.10/dist-packages (from requests->sagemaker) (2024.2.2)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-pac
kages (from google-pasta->sagemaker) (1.16.0)
Requirement already satisfied: jsonschema-specifications>=2023.03.6 in /
usr/local/lib/python3.10/dist-packages (from jsonschema->sagemaker) (202
3.12.1)
Requirement already satisfied: referencing>=0.28.4 in /usr/local/lib/pyt
hon3.10/dist-packages (from jsonschema->sagemaker) (0.34.0)
Requirement already satisfied: rpds-py>=0.7.1 in /usr/local/lib/python3.
10/dist-packages (from jsonschema->sagemaker) (0.18.0)
Requirement already satisfied: python-dateutil>=2.8.1 in /usr/local/lib/
python3.10/dist-packages (from pandas->sagemaker) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.1
0/dist-packages (from pandas->sagemaker) (2023.4)
Requirement already satisfied: ppft>=1.7.6.8 in /usr/local/lib/python3.1
0/dist-packages (from pathos->sagemaker) (1.7.6.8)
Requirement already satisfied: dill>=0.3.8 in /usr/local/lib/python3.10/
dist-packages (from pathos->sagemaker) (0.3.8)
Requirement already satisfied: pox>=0.3.4 in /usr/local/lib/python3.10/d
ist-packages (from pathos->sagemaker) (0.3.4)
Requirement already satisfied: multiprocess>=0.70.16 in /usr/local/lib/p
ython3.10/dist-packages (from pathos->sagemaker) (0.70.16)
Requirement already satisfied: contextlib2>=0.5.5 in /usr/local/lib/pyth
on3.10/dist-packages (from schema->sagemaker) (21.6.0)
```

In [ ]:

```python
import sagemaker
import boto3
sess = sagemaker.Session()

sagemaker_session_bucket=None
if sagemaker_session_bucket is None and sess is not None:

    sagemaker_session_bucket = sess.default_bucket()

try:
    role = sagemaker.get_execution_role()
except ValueError:
    iam = boto3.client('iam')
    role = iam.get_role(RoleName='SageMakerTest')['Role']['Arn']

sess = sagemaker.Session(default_bucket=sagemaker_session_bucket)

print(f"sagemaker role arn: {role}")
print(f"sagemaker bucket: {sess.default_bucket()}")
print(f"sagemaker session region: {sess.boto_region_name}")
```

```
WARNING:sagemaker:Couldn't call 'get_role' to get Role ARN from role nam
e collab to get Role path.

sagemaker role arn: arn:aws:iam::381491942612:role/SageMakerTest
sagemaker bucket: sagemaker-ap-southeast-2-381491942612
sagemaker session region: ap-southeast-2
```

In [ ]:

```
from datasets import load_dataset

# Convert dataset to OAI messages
system_message = """You are an text to SQL query translator. Users will as
SCHEMA:
{schema}"""

def create_conversation(sample):
  return {
    "messages": [
      {"role": "system", "content": system_message.format(schema=sample["c
      {"role": "user", "content": sample["question"]},
      {"role": "assistant", "content": sample["answer"]}
    ]
  }

# Load dataset from the hub
dataset = load_dataset("b-mc2/sql-create-context", split="train")
dataset = dataset.shuffle().select(range(12500))

# Convert dataset to OAI messages
dataset = dataset.map(create_conversation, remove_columns=dataset.features
# split dataset into 10,000 training samples and 2,500 test samples
dataset = dataset.train_test_split(test_size=2500/12500)

print(dataset["train"][345]["messages"])
```

```
Downloading readme:   0%|          | 0.00/4.43k [00:00<?, ?B/s]

Downloading data:   0%|          | 0.00/21.8M [00:00<?, ?B/s]

Generating train split: 0 examples [00:00, ? examples/s]

Map:   0%|          | 0/12500 [00:00<?, ? examples/s]

[{'content': 'You are an text to SQL query translator. Users will ask yo
u questions in English and you will generate a SQL query based on the pr
ovided SCHEMA.\nSCHEMA:\nCREATE TABLE table_12962773_4 (no INTEGER, play
er VARCHAR)', 'role': 'system'}, {'content': 'What No is the player Zora
n Erceg', 'role': 'user'}, {'content': 'SELECT MIN(no) FROM table_129627
73_4 WHERE player = "Zoran Erceg"', 'role': 'assistant'}]
```

In [ ]:

```
training_input_path = f's3://{sess.default_bucket()}/datasets/text-to-sql'


dataset["train"].to_json(f"{training_input_path}/train_dataset.json", orie
dataset["test"].to_json(f"{training_input_path}/test_dataset.json", orient

print(f"Training data uploaded to:")
print(f"{training_input_path}/train_dataset.json")
print(f"https://s3.console.aws.amazon.com/s3/buckets/{sess.default_bucket(
```

```
Creating json from Arrow format:   0%|          | 0/10 [00:00<?, ?ba/s]

Creating json from Arrow format:   0%|          | 0/3 [00:00<?, ?ba/s]

Training data uploaded to:
s3://sagemaker-ap-southeast-2-381491942612/datasets/text-to-sql/train_da
taset.json
https://s3.console.aws.amazon.com/s3/buckets/sagemaker-ap-southeast-2-38
1491942612/?region=ap-southeast-2&prefix=datasets/text-to-sql/ (https://
s3.console.aws.amazon.com/s3/buckets/sagemaker-ap-southeast-2-3814919426
12/?region=ap-southeast-2&prefix=datasets/text-to-sql/)
```

In [ ]:

```
# hyperparameters, which are passed into the training job
hyperparameters = {
  ### SCRIPT PARAMETERS ###
  'dataset_path': '/opt/ml/input/data/training/train_dataset.json', # path
  'model_id': "codellama/CodeLlama-7b-hf",
  'max_seq_len': 3072,
  'use_qlora': True,
  ### TRAINING PARAMETERS ###
  'num_train_epochs': 3,
  'per_device_train_batch_size': 1,
  'gradient_accumulation_steps': 4,
  'gradient_checkpointing': True,
  'optim': "adamw_torch_fused",
  'logging_steps': 10,
  'save_strategy': "epoch",
  'learning_rate': 2e-4,
  'bf16': False,
  'tf32': True,
  'max_grad_norm': 0.3,
  'warmup_ratio': 0.03,
  'lr_scheduler_type': "constant",
  'report_to': "tensorboard",
  'output_dir': '/tmp/tun',
  'merge_adapters': True,
}
```

In [ ]:
```
from sagemaker.huggingface import HuggingFace

# define Training Job Name
job_name = f'codellama-7b-hf-text-to-sql-exp1'

# create the Estimator
huggingface_estimator = HuggingFace(
    entry_point          = 'run_sft.py',    # train script
    source_dir           = '/content/',#'https://github.com/philschmid/llm
    instance_type        = 'ml.t3.medium',   # instances type used for the
    instance_count       = 1,                 # the number of instances us
    max_run              = 2*24*60*60,        # maximum runtime in seconds
    base_job_name        = job_name,          # the name of the training j
    role                 = role,              # Iam role used in training
    volume_size          = 300,               # the size of the EBS volume
    transformers_version = '4.36',            # the transformers version u
    pytorch_version      = '2.1',             # the pytorch_version versic
    py_version           = 'py310',           # the python version used ir
    hyperparameters      =  hyperparameters,  # the hyperparameters passec
    disable_output_compression = True,        # not compress output to sav
    environment          = {
                            "HUGGINGFACE_HUB_CACHE": "/tmp/.cache", # set
                             "HF_TOKEN": "hf_jwOUrKKKVGoOUrnBQZZQgJTPhfyTH
                            },
)
```

In [ ]:
```
# define a data input dictonary with our uploaded s3 uris
data = {'training': training_input_path}

# starting the train job with our uploaded datasets as input
huggingface_estimator.fit(data, wait=True)
```

In [ ]:
```
from sagemaker.huggingface import get_huggingface_llm_image_uri

# retrieve the llm image uri
llm_image = get_huggingface_llm_image_uri(
  "huggingface",
  version="1.4.0",
  session=sess,
)

# print ecr image uri
print(f"llm image uri: {llm_image}")
```

```
llm image uri: 763104351884.dkr.ecr.ap-southeast-2.amazonaws.com/hugging
face-pytorch-tgi-inference:2.1.1-tgi1.4.0-gpu-py310-cu121-ubuntu20.04
```

In [ ]:
```
import json
from sagemaker.huggingface import HuggingFaceModel

# s3 path where the model will be uploaded
# if you try to deploy the model to a different time add the s3 path here
model_s3_path = huggingface_estimator.model_data["S3DataSource"]["S3Uri"]

# sagemaker config
instance_type = "ml.g5.2xlarge"
number_of_gpu = 1
health_check_timeout = 300

# Define Model and Endpoint configuration parameter
config = {
  'HF_MODEL_ID': "/opt/ml/model", # path to where sagemaker stores the moc
  'SM_NUM_GPUS': json.dumps(number_of_gpu), # Number of GPU used per repli
  'MAX_INPUT_LENGTH': json.dumps(1024), # Max length of input text
  'MAX_TOTAL_TOKENS': json.dumps(2048), # Max length of the generation (ir
}

# create HuggingFaceModel with the image uri
llm_model = HuggingFaceModel(
  role=role,
  image_uri=llm_image,
  model_data={'S3DataSource':{'S3Uri': model_s3_path,'S3DataType': 'S3Pref
  env=config
)
```

In [ ]:
```

# Deploy model to an endpoint
# https://sagemaker.readthedocs.io/en/stable/api/inference/model.html#sage
llm = llm_model.deploy(
  initial_instance_count=1,
  instance_type=instance_type,
  container_startup_health_check_timeout=health_check_timeout, # 10 minute
)
```

In [ ]:

```
from transformers import AutoTokenizer
from sagemaker.s3 import S3Downloader

# Load the tokenizer
tokenizer = AutoTokenizer.from_pretrained("codellama/CodeLlama-7b-hf")

# Load the test dataset from s3
S3Downloader.download(f"{training_input_path}/test_dataset.json", ".")
test_dataset = load_dataset("json", data_files="test_dataset.json",split='
random_sample = test_dataset[345]

def request(sample):
    prompt = tokenizer.apply_chat_template(sample, tokenize=False, add_ger
    outputs = llm.predict({
      "inputs": prompt,
      "parameters": {
        "max_new_tokens": 512,
        "do_sample": False,
        "return_full_text": False,
        "stop": ["<|im_end|>"],
      }
    })
    return {"role": "assistant", "content": outputs[0]["generated_text"].s

print(random_sample["messages"][1])
request(random_sample["messages"][:2])
```

In [ ]:

```
from transformers import AutoTokenizer
from sagemaker.s3 import S3Downloader

# Load the tokenizer
tokenizer = AutoTokenizer.from_pretrained("codellama/CodeLlama-7b-hf")

# Load the test dataset from s3
S3Downloader.download(f"{training_input_path}/test_dataset.json", ".")
test_dataset = load_dataset("json", data_files="test_dataset.json",split='
random_sample = test_dataset[345]

def request(sample):
    prompt = tokenizer.apply_chat_template(sample, tokenize=False, add_ger
    outputs = llm.predict({
      "inputs": prompt,
      "parameters": {
        "max_new_tokens": 512,
        "do_sample": False,
        "return_full_text": False,
        "stop": ["<|im_end|>"],
      }
    })
    return {"role": "assistant", "content": outputs[0]["generated_text"].s

print(random_sample["messages"][1])
request(random_sample["messages"][:2])
```

In [ ]:

```
from tqdm import tqdm

def evaluate(sample):
    predicted_answer = request(sample["messages"][:2])
    if predicted_answer["content"] == sample["messages"][2]["content"]:
        return 1
    else:
        return 0

success_rate = []
number_of_eval_samples = 1000
# iterate over eval dataset and predict
for s in tqdm(test_dataset.shuffle().select(range(number_of_eval_samples))
    success_rate.append(evaluate(s))

# compute accuracy
accuracy = sum(success_rate)/len(success_rate)

print(f"Accuracy: {accuracy*100:.2f}%")
```

In [ ]:

```
llm.delete_model()
llm.delete_endpoint()
```