Fine Tune Llama-2
Quote Generation Model

February 19, 2024

0.1 step 1: Loading dataset and preprocessing it

[1]:
```
import pandas as pd
import numpy as np
```

[2]:
```
data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/NLP/LLM/Data/quotes.
↪csv')
data
```

[2]:
```
      Unnamed: 0                                              quote  \
0              0     "Be yourself; everyone else is already taken."
1              1  "I'm selfish, impatient and a little insecure…
2              2  "Two things are infinite: the universe and hum…
3              3                    "So many books, so little time."
4              4  "A room without books is like a body without a…
…            …                                                 …
2503        2995  "Morality is simply the attitude we adopt towa…
2504        2996  "Don't aim at success. The more you aim at it …
2505        2997  "In life, finding a voice is speaking and livi…
2506        2998  "Winter is the time for comfort, for good food…
2507        2999                      "Silence is so freaking loud"

                     author                                               tags
0              Oscar Wilde  ['be-yourself', 'gilbert-perreira', 'honesty',…
1            Marilyn Monroe  ['best', 'life', 'love', 'mistakes', 'out-of-c…
2          Albert Einstein  ['human-nature', 'humor', 'infinity', 'philoso…
3              Frank Zappa                                ['books', 'humor']
4     Marcus Tullius Cicero                       ['books', 'simile', 'soul']
…                       …                                                 …
2503          Oscar Wilde,                        ['morality', 'philosophy']
2504      Viktor E. Frankl,                          ['happiness', 'success']
2505           John Grisham                            ['inspirational-life']
2506         Edith Sitwell                     ['comfort', 'home', 'winter']
2507         Sarah Dessen,  ['just-listen', 'loud', 'owen', 'sara-dessen',…

[2508 rows x 4 columns]
```

[3]:
```
# data[data['quote']=='Life is the flower of which love is the honey. The more you give, the more you get.']
data['tags'].unique()
```

[3]:
```
array(["['be-yourself', 'gilbert-perreira', 'honesty', 'inspirational',
'misattributed-oscar-wilde', 'quote-investigator']",
       "['best', 'life', 'love', 'mistakes', 'out-of-control', 'truth',
'worst']",
       "['human-nature', 'humor', 'infinity', 'philosophy', 'science',
'stupidity', 'universe']",
       …, "['morality', 'philosophy']", "['comfort', 'home', 'winter']",
       "['just-listen', 'loud', 'owen', 'sara-dessen', 'silence']"],
      dtype=object)
```

[5]:
```
data = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/NLP/LLM/Data/quotes.csv')
data['quote'] = data['quote'].str.lower()
data['tags'] = data['tags'].str.lower()
data = data.drop('Unnamed: 0',axis=1)
data
```

[5]:

	quote	author	tags
0	"be yourself; everyone else is already taken."	Oscar Wilde	['be-yourself', 'gilbert-perreira', 'honesty',…
1	"i'm selfish, impatient and a little insecure…	Marilyn Monroe	['best', 'life', 'love', 'mistakes', 'out-of-c…
2	"two things are infinite: the universe and hum…	Albert Einstein	['human-nature', 'humor', 'infinity', 'philoso…
3	"so many books, so little time."	Frank Zappa	['books', 'humor']
4	"a room without books is like a body without a…	Marcus Tullius Cicero	['books', 'simile', 'soul']
…	…	…	…
2503	"morality is simply the attitude we adopt towa…	Oscar Wilde,	['morality', 'philosophy']
2504	"don't aim at success. the more you aim at it …	Viktor E. Frankl,	['happiness', 'success']
2505	"in life, finding a voice is speaking and livi…	John Grisham	['inspirational-life']
2506	"winter is the time for comfort, for good food…	Edith Sitwell	['comfort', 'home', 'winter']
2507	"silence is so freaking loud"	Sarah Dessen,	['just-listen', 'loud', 'owen', 'sara-dessen',…

```
[2508 rows x 3 columns]
```

[7]:
```
from datasets import load_dataset,Dataset
# data_path="/kaggle/input/english-quotes/quotes.jsonl"
# data = load_dataset("json",data_files={
#     "train":data_path
# })
# data = data.map(lambda samples: tokenizer(samples["quote"]), batched=True)
data = data[['quote','tags']]
data1 = Dataset.from_pandas(data, split='train')
data1
```

[7]:
```
Dataset({
    features: ['quote', 'tags'],
    num_rows: 2508
})
```

0.2 step 2: Loading the Llama-2 LLM for downstreaming task

[4]:
```
!pip install -q -U bitsandbytes
!pip install transformers==4.31
!pip install -q -U git+https://github.com/huggingface/peft.git
!pip install -q -U git+https://github.com/huggingface/accelerate.git
!pip install -q datasets
```

```
Requirement already satisfied: transformers==4.31 in
/usr/local/lib/python3.10/dist-packages (4.31.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from transformers==4.31) (3.13.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.14.1 in
/usr/local/lib/python3.10/dist-packages (from transformers==4.31) (0.20.3)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from transformers==4.31) (1.25.2)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from transformers==4.31) (23.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from transformers==4.31) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.10/dist-packages (from transformers==4.31) (2023.12.25)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-
packages (from transformers==4.31) (2.31.0)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in
/usr/local/lib/python3.10/dist-packages (from transformers==4.31) (0.13.3)
Requirement already satisfied: safetensors>=0.3.1 in
/usr/local/lib/python3.10/dist-packages (from transformers==4.31) (0.4.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-
packages (from transformers==4.31) (4.66.2)
```

```
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.14.1->transformers==4.31) (2023.6.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.10/dist-packages (from huggingface-
hub<1.0,>=0.14.1->transformers==4.31) (4.9.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers==4.31)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests->transformers==4.31) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers==4.31)
(2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests->transformers==4.31)
(2024.2.2)
  Installing build dependencies … done
  Getting requirements to build wheel … done
  Preparing metadata (pyproject.toml) … done
  Installing build dependencies … done
  Getting requirements to build wheel … done
  Preparing metadata (pyproject.toml) … done
```

[5]:
```
# !pip uninstall pyarrow
!pip install pyarrow==12.0.0
```

```
Requirement already satisfied: pyarrow==12.0.0 in
/usr/local/lib/python3.10/dist-packages (12.0.0)
Requirement already satisfied: numpy>=1.16.6 in /usr/local/lib/python3.10/dist-
packages (from pyarrow==12.0.0) (1.25.2)
```

[6]:
```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig
# EleutherAI/gpt-neo-125m
model_id = "meta-llama/Llama-2-7b-chat-hf"
# model_id = "meta-llama/Llama-2-13b-chat-hf"
bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16
)

tokenizer = AutoTokenizer.from_pretrained(model_id)
model = AutoModelForCausalLM.from_pretrained(model_id,␣
 ↪quantization_config=bnb_config, device_map={"":0})
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_token.py:88:
UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab
(https://huggingface.co/settings/tokens), set it as secret in your Google Colab
and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access
public models or datasets.
  warnings.warn(
```

```
Loading checkpoint shards:   0%|          | 0/7 [00:00<?, ?it/s]
```

0.3 step 3: converting normal model to PEFT model using LORA config

[7]:
```
from peft import prepare_model_for_kbit_training

model.gradient_checkpointing_enable()
model = prepare_model_for_kbit_training(model)
```

[8]:
```
def print_trainable_parameters(model):
    """
    Prints the number of trainable parameters in the model.
    """
    trainable_params = 0
    all_param = 0
    for _, param in model.named_parameters():
        all_param += param.numel()
        if param.requires_grad:
            trainable_params += param.numel()
    print(f"trainable params: {trainable_params} || all params: {all_param} ||␣
 ↪trainable%: {100 * trainable_params / all_param}")
```

[9]:
```
from peft import LoraConfig, get_peft_model

config = LoraConfig(r=8,
    lora_alpha=32,
    target_modules=["self_attn.q_proj", "self_attn.k_proj", "self_attn.v_proj",␣
 ↪"self_attn.o_proj"], #specific to Llama models.
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM")

model = get_peft_model(model, config)
print_trainable_parameters(model)
```

```
trainable params: 8388608 || all params: 3508801536 || trainable%:
0.23907331075678143
```

[10]:
```
!pip install datasets
```

```
Requirement already satisfied: datasets in /usr/local/lib/python3.10/dist-
packages (2.17.1)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-
packages (from datasets) (3.13.1)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-
packages (from datasets) (1.25.2)
Requirement already satisfied: pyarrow>=12.0.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (12.0.0)
Requirement already satisfied: pyarrow-hotfix in /usr/local/lib/python3.10/dist-
packages (from datasets) (0.6)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages
(from datasets) (1.5.3)
Requirement already satisfied: requests>=2.19.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (2.31.0)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.10/dist-
packages (from datasets) (4.66.2)
Requirement already satisfied: xxhash in /usr/local/lib/python3.10/dist-packages
(from datasets) (3.4.1)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.10/dist-
packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec[http]<=2023.10.0,>=2023.1.0 in
/usr/local/lib/python3.10/dist-packages (from datasets) (2023.6.0)
Requirement already satisfied: aiohttp in /usr/local/lib/python3.10/dist-
packages (from datasets) (3.9.3)
Requirement already satisfied: huggingface-hub>=0.19.4 in
/usr/local/lib/python3.10/dist-packages (from datasets) (0.20.3)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from datasets) (23.2)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-
packages (from datasets) (6.0.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.3.1)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp->datasets) (23.2.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (1.4.1)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (6.0.5)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.10/dist-
packages (from aiohttp->datasets) (1.9.4)
Requirement already satisfied: async-timeout<5.0,>=4.0 in
/usr/local/lib/python3.10/dist-packages (from aiohttp->datasets) (4.0.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
```

```
/usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.19.4->datasets)
(4.9.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests>=2.19.0->datasets) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets)
(2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from requests>=2.19.0->datasets)
(2024.2.2)
Requirement already satisfied: python-dateutil>=2.8.1 in
/usr/local/lib/python3.10/dist-packages (from pandas->datasets) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-
packages (from pandas->datasets) (2023.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.8.1->pandas->datasets) (1.16.0)
```

[11]:
```
data1 = data1.map(lambda samples: tokenizer(samples["quote"]), batched=True)
data1
```

```
Map:   0%|          | 0/2508 [00:00<?, ? examples/s]
```

[11]:
```
Dataset({
    features: ['quote', 'tags', 'input_ids', 'attention_mask'],
    num_rows: 2508
})
```

0.4 step 4: training the model

[13]:
```
import transformers

# needed for Llama tokenizer
tokenizer.pad_token = tokenizer.eos_token # </s>

trainer = transformers.Trainer(
    model=model,
    train_dataset=data1,
    args=transformers.TrainingArguments(
        per_device_train_batch_size=1,
        gradient_accumulation_steps=4,
        warmup_steps=2,
        max_steps=10,
        learning_rate=2e-4,
        fp16=True,
```

```
        logging_steps=1,
        output_dir="outputs",
        optim="paged_adamw_8bit"),
    data_collator=transformers.DataCollatorForLanguageModeling(tokenizer,␣
 ↪mlm=False),)
model.config.use_cache = False  # silence the warnings. Please re-enable for␣
 ↪inference!
trainer.train()
```

```
You're using a LlamaTokenizerFast tokenizer. Please note that with a fast
tokenizer, using the `__call__` method is faster than using a method to encode
the text followed by a call to the `pad` method to get a padded encoding.
/usr/local/lib/python3.10/dist-packages/torch/utils/checkpoint.py:429:
UserWarning: torch.utils.checkpoint: please pass in use_reentrant=True or
use_reentrant=False explicitly. The default value of use_reentrant will be
updated to be False in the future. To maintain current behavior, pass
use_reentrant=True. It is recommended that you use use_reentrant=False. Refer to
docs for more details on the differences between the two variants.
  warnings.warn(
```

```
<IPython.core.display.HTML object>
```

[13]:
```
TrainOutput(global_step=10, training_loss=2.091718888282776,
metrics={'train_runtime': 61.574, 'train_samples_per_second': 0.65,
'train_steps_per_second': 0.162, 'total_flos': 36398413479936.0, 'train_loss':
2.091718888282776, 'epoch': 0.02})
```

[14]:
```
model.config.use_cache = True
model.eval()
```

[14]:
```
PeftModelForCausalLM(
  (base_model): LoraModel(
    (model): LlamaForCausalLM(
      (model): LlamaModel(
        (embed_tokens): Embedding(32000, 4096, padding_idx=0)
        (layers): ModuleList(
          (0-31): 32 x LlamaDecoderLayer(
            (self_attn): LlamaAttention(
              (q_proj): lora.Linear4bit(
                (base_layer): Linear4bit(in_features=4096, out_features=4096,
bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=4096, out_features=8,
bias=False)
                )
```

```
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=4096,
bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
              )
              (k_proj): lora.Linear4bit(
                (base_layer): Linear4bit(in_features=4096, out_features=4096,
bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=4096, out_features=8,
bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=4096,
bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
              )
              (v_proj): lora.Linear4bit(
                (base_layer): Linear4bit(in_features=4096, out_features=4096,
bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
                (lora_A): ModuleDict(
                  (default): Linear(in_features=4096, out_features=8,
bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=4096,
bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
              )
              (o_proj): lora.Linear4bit(
                (base_layer): Linear4bit(in_features=4096, out_features=4096,
bias=False)
                (lora_dropout): ModuleDict(
                  (default): Dropout(p=0.05, inplace=False)
                )
```

```
                (lora_A): ModuleDict(
                  (default): Linear(in_features=4096, out_features=8,
bias=False)
                )
                (lora_B): ModuleDict(
                  (default): Linear(in_features=8, out_features=4096,
bias=False)
                )
                (lora_embedding_A): ParameterDict()
                (lora_embedding_B): ParameterDict()
              )
              (rotary_emb): LlamaRotaryEmbedding()
            )
            (mlp): LlamaMLP(
              (gate_proj): Linear4bit(in_features=4096, out_features=11008,
bias=False)
              (up_proj): Linear4bit(in_features=4096, out_features=11008,
bias=False)
              (down_proj): Linear4bit(in_features=11008, out_features=4096,
bias=False)
              (act_fn): SiLUActivation()
            )
            (input_layernorm): LlamaRMSNorm()
            (post_attention_layernorm): LlamaRMSNorm()
          )
        )
        (norm): LlamaRMSNorm()
      )
      (lm_head): Linear(in_features=4096, out_features=32000, bias=False)
    )
  )
)
```

[ ]:
```
# output_dir='/content/drive/MyDrive/Colab Notebooks/NLP/LLM/
  ↪quote_generation_adaptive_model'
# model.save_pretrained(adapter_model)
```

##step 6 : converting normal model to PEFT model using LORA config

[ ]:
```
from transformers import AutoModelForCausalLM
model_id = "/content/drive/MyDrive/Colab Notebooks/NLP/LLM/
  ↪Llama-2-7b-chat-hf-sharded-bf16"
model_1 = AutoModelForCausalLM.from_pretrained(model_id,␣
  ↪trust_remote_code=True, torch_dtype=torch.float16, cache_dir="cache")

from peft import PeftModel
```

```
# load perf model with new adapters
model_peft = PeftModel.from_pretrained(model_1,model)
model = model.merge_and_unload()
model_peft.save_pretrained('/content/drive/MyDrive/Colab Notebooks/NLP/LLM/
 ↪quote_generation')
```

##step 7 : loading the final saved model

[ ]:
```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM, BitsAndBytesConfig

finetuned_model_checkpoint = '/content/drive/MyDrive/Colab Notebooks/NLP/LLM/
 ↪quote_generation'

bitsandbytes_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.bfloat16
)

tokenizer = AutoTokenizer.from_pretrained(finetuned_model_checkpoint)
model = AutoModelForCausalLM.from_pretrained(finetuned_model_checkpoint,␣
 ↪quantization_config=bitsandbytes_config)
```

0.5 creating the prompts and streaming the generated text

[ ]:
```
from transformers import TextStreamer
def stream(user_prompt,model):
    runtimeFlag = "cuda:0"
    system_prompt = 'You are a helpful assistant that provides accurate and␣
 ↪concise responses'

    B_INST, E_INST = "[INST]", "[/INST]"
    B_SYS, E_SYS = "<<SYS>>\n", "\n<</SYS>>\n\n"

    prompt = f"{B_INST} {B_SYS}{system_prompt.strip()}{E_SYS}{user_prompt.
 ↪strip()} {E_INST}\n\n"

    inputs = tokenizer([prompt], return_tensors="pt").to(runtimeFlag)

    streamer = TextStreamer(tokenizer)
    print('the quote is generating')
    # Despite returning the usual output, the streamer will also print the␣
 ↪generated text to stdout.
    _ = model.generate(**inputs, streamer=streamer, max_new_tokens=100)
```

0.6 English Quote Genereted

[21]:
```
stream('Provide a quote on love and wind it should contains less words but␣
↪interesting',model)
```

```
the quote is generating
<s> [INST] <<SYS>>
You are a helpful assistant that provides accurate and concise responses
<</SYS>>

Provide a quote on love and wind it should contains less words but interesting
[/INST]

"Love is like a gentle breeze, it can calm the soul and lift the spirit. Just as
wind can carry the scent of blooming flowers, love carries the essence of
connection and belonging."</s>
```

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]:

[ ]: