```
!pip install  pip
!pip install  torch
!pip install  transformers
!pip install  datasets
!pip install  accelerate
!pip install  bitsandbytes
!pip install  peft
!pip install trl==0.9.4
!pip install  colored
```

Show hidden output

```
from huggingface_hub import login
login(new_session=False)
```

```
# Imports
import random
from textwrap import dedent
from typing import Dict, List

import matplotlib as mpl
import matplotlib.colors as colors
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import torch
from colored import Back, Fore, Style
from datasets import Dataset, load_dataset
from matplotlib.ticker import PercentFormatter
from peft import (
    LoraConfig,
    PeftModel,
    TaskType,
    get_peft_model,
    prepare_model_for_kbit_training,
)
from sklearn.model_selection import train_test_split
from torch.utils.data import DataLoader
from tqdm import tqdm
from transformers import (
    AutoModelForCausalLM,
    AutoTokenizer,
    BitsAndBytesConfig,
    pipeline,
)
from trl import SFTConfig, SFTTrainer
from trl.trainer.utils import DataCollatorForCompletionOnlyLM

# Plotting magic (for Jupyter Notebooks; remove if running as .py script)
# %matplotlib inline
# %config InlineBackend.figure_format='retina'

# Color palette
COLORS = ["#bae1ff", "#ffb3ba", "#ffdfba", "#ffffba", "#baffc9"]

# Seaborn and matplotlib style
sns.set(style="whitegrid", palette="muted", font_scale=1.2)
sns.set_palette(sns.color_palette(COLORS))

cmap = colors.LinearSegmentedColormap.from_list("custom_cmap", COLORS[:2])
```

```
# Matplotlib style config (fixed all key typos and line styles)
MY_STYLE = {
    "figure.facecolor": "black",
    "axes.facecolor": "black",
    "axes.edgecolor": "white",
    "axes.labelcolor": "white",
    "axes.linewidth": 0.5,
    "text.color": "white",
    "xtick.color": "white",
    "ytick.color": "white",
    "grid.color": "gray",
    "grid.linestyle": "--",
    "grid.linewidth": 0.5,
    "axes.grid": True,
    "xtick.labelsize": "medium",
    "ytick.labelsize": "medium",
    "axes.titlesize": "large",
    "axes.labelsize": "large",
    "lines.color": COLORS[0],
    "patch.edgecolor": "white",
}
mpl.rcParams.update(MY_STYLE)

# Set seed for reproducibility
SEED = 42

def seed_everything(seed: int):
    random.seed(seed)
    np.random.seed(seed)
    torch.manual_seed(seed)

seed_everything(SEED)

# Constants
PAD_TOKEN = "<|pad|>"
MODEL_NAME = "meta-llama/Meta-Llama-3-8B-Instruct"
NEW_MODEL = "Llama-3-8B-Instruct-MedQuad-MedicalQna"



quantization_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_use_double_quant=True,
    bnb_4bit_quant_type="nf4",
    bnb_4bit_compute_dtype=torch.float16
)
tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, use_fast=True)
tokenizer.add_special_tokens({"pad_token": PAD_TOKEN})
tokenizer.padding_side = "right"

max_mem = {0: "14GiB", "cpu": "32GiB"}  # leave some buffer
model = AutoModelForCausalLM.from_pretrained(
    MODEL_NAME,
    quantization_config=quantization_config,
    device_map="auto",
    max_memory=max_mem,
)


model.resize_token_embeddings(len(tokenizer), pad_to_multiple_of=8)
```

```
tokenizer_config.json: 100%                                   51.0k/51.0k [00:00<00:00, 5.59MB/s]
tokenizer.json: 100%                                          9.09M/9.09M [00:00<00:00, 33.6MB/s]
special_tokens_map.json: 100%                                 73.0/73.0 [00:00<00:00, 6.34kB/s]
config.json: 100%                                             654/654 [00:00<00:00, 79.6kB/s]
model.safetensors.index.json: 100%                            23.9k/23.9k [00:00<00:00, 1.38MB/s]
Fetching 4 files: 100%                                        4/4 [07:18<00:00, 438.72s/it]
model-00003-of-00004.safetensors: 100%                        4.92G/4.92G [06:10<00:00, 11.5MB/s]
model-00002-of-00004.safetensors: 100%                        5.00G/5.00G [05:46<00:00, 13.1MB/s]
model-00004-of-00004.safetensors: 100%                        1.17G/1.17G [00:58<00:00, 28.9MB/s]
model-00001-of-00004.safetensors: 100%                        4.98G/4.98G [07:18<00:00, 91.5MB/s]
Loading checkpoint shards: 100%                               4/4 [01:22<00:00, 17.74s/it]
generation_config.json: 100%                                  187/187 [00:00<00:00, 20.7kB/s]
The new embeddings will be initialized from a multivariate normal distribution that has old embeddings' mean and cov
The new lm_head weights will be initialized from a multivariate normal distribution that has old embeddings' mean an
Embedding(128264, 4096)
```

DATASET PREPROCESSING

```
dataset = load_dataset("keivalya/MedQuad-MedicalQnADataset")
```

```
README.md: 100%                                   233/233 [00:00<00:00, 14.6kB/s]
medDataset_processed.csv: 100%                    22.5M/22.5M [00:00<00:00, 55.0MB/s]
Generating train split: 100%                      16407/16407 [00:00<00:00, 31770.78 examples/s]
```

```
dataset
```

```
DatasetDict({
    train: Dataset({
        features: ['qtype', 'Question', 'Answer'],
        num_rows: 16407
    })
})
```

```
rows = []
for item in dataset ["train"]:
    rows. append(
    {
    "qtype": item["qtype"],
    "question": item["Question"],
    "answer": item["Answer"],
    }
    )
df = pd.DataFrame(rows)

df.head()
```

	qtype	question	answer
0	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	LCMV infections can occur after exposure to fr...
1	symptoms	What are the symptoms of Lymphocytic Choriomen...	LCMV is most commonly recognized as causing ne...
2	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	Individuals of all ages who come into contact ...
3	exams and tests	How to diagnose Lymphocytic Choriomeningitis (...	During the first phase of the disease, the mos...
4	treatment	What are the treatments for Lymphocytic Chorio...	Aseptic meningitis, encephalitis, or meningoen...

```
df.isnull().value_counts()
```

			count
qtype	question	answer	
False	**False**	**False**	16407

dtype: int64

```
def format_example(row: dict):
    prompt=dedent(
        f"""
        {row["question"]}
        Type:

        '''
        {row["qtype"]}
        '''

        """
    )
    messages = [
        {
            "role": "system",
            "content": f"You are a helpful medical assistant. The question type is: {row['qtype']}."
        },
        {
            "role": "user",
            "content": row["question"]
        },
        {
            "role": "assistant",
            "content": row["answer"]
        }
    ]
    return tokenizer.apply_chat_template(messages, tokenize=False)
```

```
df["text"]=df.apply(format_example,axis=1)
```

```
df.head()
```

	qtype	question	answer	text
0	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	LCMV infections can occur after exposure to fr...	<|begin_of_text|><|start_header_id|>system<|en...
1	symptoms	What are the symptoms of Lymphocytic Choriomen...	LCMV is most commonly recognized as causing ne...	<|begin_of_text|><|start_header_id|>system<|en...
2	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	Individuals of all ages who come into contact ...	<|begin_of_text|><|start_header_id|>system<|en...
3	exams and tests	How to diagnose Lymphocytic Choriomeningitis (...	During the first phase of the disease, the mos...	<|begin_of_text|><|start_header_id|>system<|en...
4	treatment	What are the treatments for Lymphocytic Chorio...	Aseptic meningitis, encephalitis, or meningoen...	<|begin_of_text|><|start_header_id|>system<|en...

```
def count_tokens(row: Dict) -> int:
    return len(
        tokenizer(
            row["text"],
            add_special_tokens=True,
            return_attention_mask=False,
        ) ["input_ids"]
    )


df["token_count"] = df.apply(count_tokens, axis=1)

df.head()
```

	qtype	question	answer	text	token_count
0	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	LCMV infections can occur after exposure to fr...	<|begin_of_text|><|start_header_id|>system<|en...	139
1	symptoms	What are the symptoms of Lymphocytic Choriomen...	LCMV is most commonly recognized as causing ne...	<|begin_of_text|><|start_header_id|>system<|en...	578
2	susceptibility	Who is at risk for Lymphocytic Choriomeningiti...	Individuals of all ages who come into contact ...	<|begin_of_text|><|start_header_id|>system<|en...	182
3	exams and tests	How to diagnose Lymphocytic Choriomeningitis (...	During the first phase of the disease, the mos...	<|begin_of_text|><|start_header_id|>system<|en...	194
4	treatment	What are the treatments for Lymphocytic Chorio...	Aseptic meningitis, encephalitis, or	<|begin_of_text|><|start_header_id|>system<|en...	142

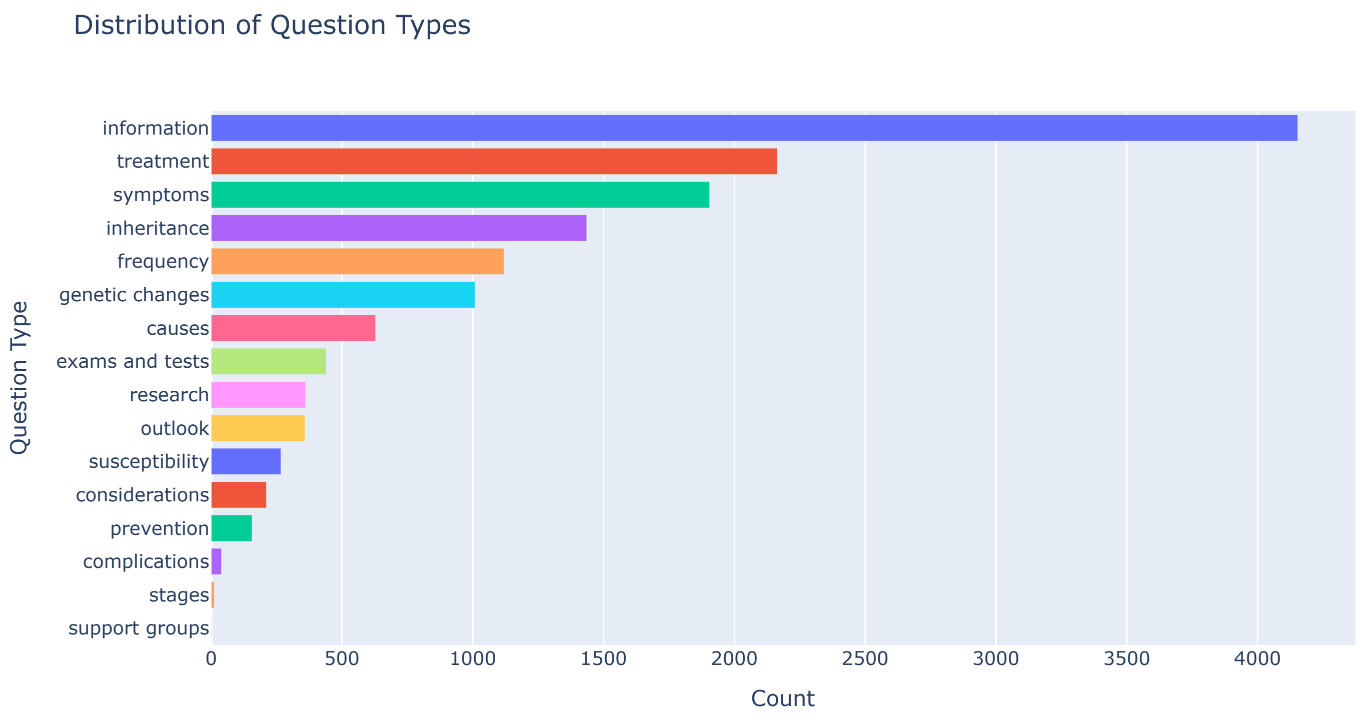
```
len(df[df.token_count<512]) ,len(df)
```

```
(14271, 16407)
```

```
df=df[df.token_count<512]
len(df)
```

```
14271
```

```
!pip install -q plotly
import plotly.express as px
import plotly.graph_objects as go



fig = px.bar(
    df["qtype"].value_counts().reset_index(),
    x="count",
    y="qtype",
    orientation="h",
    color="qtype",
    title="Distribution of Question Types",
    labels={"qtype":"Question Type", "count":"Count"}
)
fig.update_layout(showlegend=False)
fig.show()
```



```
rare = df.groupby("qtype").filter(lambda x: len(x) < 2)
df_rest = df.drop(rare.index)

train, temp = train_test_split(
    df_rest,
    test_size=0.2,
    random_state=42,
    stratify=df_rest["qtype"]
)
val,test=train_test_split(temp,test_size=0.2)
# add rare categories back into train
train = pd.concat([train, rare]).reset_index(drop=True)
```

```
len(df) , len(train), len(val), len(test)
```

```
(14271, 11417, 2283, 571)
```

```
print(train['qtype'].value_counts())
print(val['qtype'].value_counts())
print(test['qtype'].value_counts())
```

```
qtype
information        3323
treatment          1732
symptoms           1525
inheritance        1149
```

```
frequency           896
genetic changes     807
causes              503
exams and tests     353
research            289
outlook             287
susceptibility      214
considerations      170
prevention          126
complications        32
stages               10
support groups        1
Name: count, dtype: int64
qtype
information        679
treatment          340
symptoms           299
inheritance        233
frequency          180
genetic changes    162
causes              95
exams and tests     66
outlook             60
research            56
susceptibility      41
considerations      38
prevention          27
complications        5
stages               2
Name: count, dtype: int64
qtype
information        152
treatment           93
symptoms            82
inheritance         54
frequency           44
genetic changes     40
causes              31
exams and tests     22
research            17
susceptibility      12
outlook             12
considerations       4
prevention           4
complications        3
stages               1
Name: count, dtype: int64
```

```
train.sample(n=4000).to_json("/content/train.json", orient="records", lines=True)
val.sample(n=500).to_json("/content/val.json", orient="records", lines=True)
test.sample(n=100).to_json("/content/test.json", orient="records", lines=True)

dataset = load_dataset(
    "json",
    data_files={
        "train": "/content/train.json",
        "validation": "/content/val.json",
        "test": "/content/test.json"
    }
)
```

```
Generating train split:    4000/0 [00:00<00:00, 33126.96 examples/s]

Generating validation split:    500/0 [00:00<00:00, 12381.86 examples/s]

Generating test split:    100/0 [00:00<00:00, 1774.37 examples/s]
```

```
print(dataset)
print(dataset["train"][0])
```

```
DatasetDict({
    train: Dataset({
        features: ['qtype', 'question', 'answer', 'text', 'token_count'],
        num_rows: 4000
    })
    validation: Dataset({
        features: ['qtype', 'question', 'answer', 'text', 'token_count'],
        num_rows: 500
    })
    test: Dataset({
        features: ['qtype', 'question', 'answer', 'text', 'token_count'],
        num_rows: 100
    })
})
{'qtype': 'treatment', 'question': 'What are the treatments for Imerslund-Grsbeck syndrome ?', 'answer': 'These reso
```

BASELINE

```
pipe = pipeline(
task="text-generation",
model=model,
tokenizer=tokenizer,
max_new_tokens=128,
return_full_text=False,
)
```

```
Device set to use cuda:0
```

```
def create_test_prompt(data_row: dict):

    messages = [
        {
            "role": "system",
            "content": f"You are a helpful medical assistant. The question type is: {data_row['qtype']}."
        },
        {
            "role": "user",
            "content": data_row["question"]
        }
    ]
    return tokenizer.apply_chat_template(
        messages,
        tokenize=False,
        add_generation_prompt=True
    )
```

```
row=dataset["test"][0]
prompt=create_test_prompt(row)
print(prompt)
```

```
<|begin_of_text|><|start_header_id|>system<|end_header_id|>

You are a helpful medical assistant. The question type is: research.<|eot_id|><|start_header_id|>user<|end_header_id

what research (or clinical trials) is being done for Prostate Cancer ?<|eot_id|><|start_header_id|>assistant<|end_he
```

```
%%time
outputs=pipe(prompt)
print(outputs[0]["generated_text"])
```

```
Parasites - Loiasis is caused by the filarial parasite Loa loa, which is typically spread through the bite of an inf

The following groups are at increased risk for Loiasis:

1. Travelers to areas where Loa loa is endemic, such as:
        * Central and West Africa, particularly in countries like Cameroon, Democratic Republic of Congo, Republic o
        * West Africa, including countries like Nigeria, Ghana, and
CPU times: user 15.8 s, sys: 268 ms, total: 16 s
Wall time: 23.3 s
```

```
response_template = "<|end_header_id|>"
collator = DataCollatorForCompletionOnlyLM(response_template, tokenizer=tokenizer)

examples = [dataset ["train"] [0] ["text"]]
encodings = [tokenizer(e) for e in examples]

dataloader = DataLoader(encodings, collate_fn=collator, batch_size=1)
```

```
batch = next(iter(dataloader))
batch.keys()
```

```
KeysView({'input_ids': tensor([[128000, 128000, 128006,   9125, 128007,    271,   2675,    527,    264,
          11190,   6593,  18328,     13,    578,   3488,    955,    374,     25,
           6514,     13, 128009, 128006,    882, 128007,    271,   3923,    527,
            279,  22972,    369,   2417,    388,  85833,  12279,   5544,  55177,
          28439,    949, 128009, 128006,  78191, 128007,    271,   9673,   5070,
           2686,    279,  23842,    477,   6373,    315,   2417,    388,  85833,
          12279,   5544,  55177,  28439,     25,    220,    482,   3344,   1074,
          22560,  68198,     25,   1556,  22689,    482,    426,    717,  48294,
            256,   4314,   5070,    505,   3344,   1074,  22560,   3085,   2038,
            922,    279,  23842,    323,   6373,    315,   5370,   2890,   4787,
             25,    220,    482,  51088,  20756,    220,    482,  26166,  40143,
            220,    482,  48190,    323,  81318,    220,    482,  75226,  89549,
            256,    482,  72460,  54679,  10852, 128009]]), 'attention_mask': tensor([[1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
         1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]]), 'labels': tensor([[  -100,   -100,   -100,
-100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,    271,   9673,   5070,
           2686,    279,  23842,    477,   6373,    315,   2417,    388,  85833,
          12279,   5544,  55177,  28439,     25,    220,    482,   3344,   1074,
          22560,  68198,     25,   1556,  22689,    482,    426,    717,  48294,
            256,   4314,   5070,    505,   3344,   1074,  22560,   3085,   2038,
            922,    279,  23842,    323,   6373,    315,   5370,   2890,   4787,
             25,    220,    482,  51088,  20756,    220,    482,  26166,  40143,
            220,    482,  48190,    323,  81318,    220,    482,  75226,  89549,
            256,    482,  72460,  54679,  10852, 128009]])})
```

```
batch["labels"]
```

```
tensor([[  -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
           -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,   -100,
          -100,   -100,   -100,   -100,   -100,   -100,    271,   9673,   5070,
          2686,    279,  23842,    477,   6373,    315,   2417,    388,  85833,
         12279,   5544,  55177,  28439,     25,    220,    482,   3344,   1074,
         22560,  68198,     25,   1556,  22689,    482,    426,    717,  48294,
           256,   4314,   5070,    505,   3344,   1074,  22560,   3085,   2038,
           922,    279,  23842,    323,   6373,    315,   5370,   2890,   4787,
            25,    220,    482,  51088,  20756,    220,    482,  26166,  40143,
           220,    482,  48190,    323,  81318,    220,    482,  75226,  89549,
           256,    482,  72460,  54679,  10852, 128009]])
```

Finetuning

```
model
```

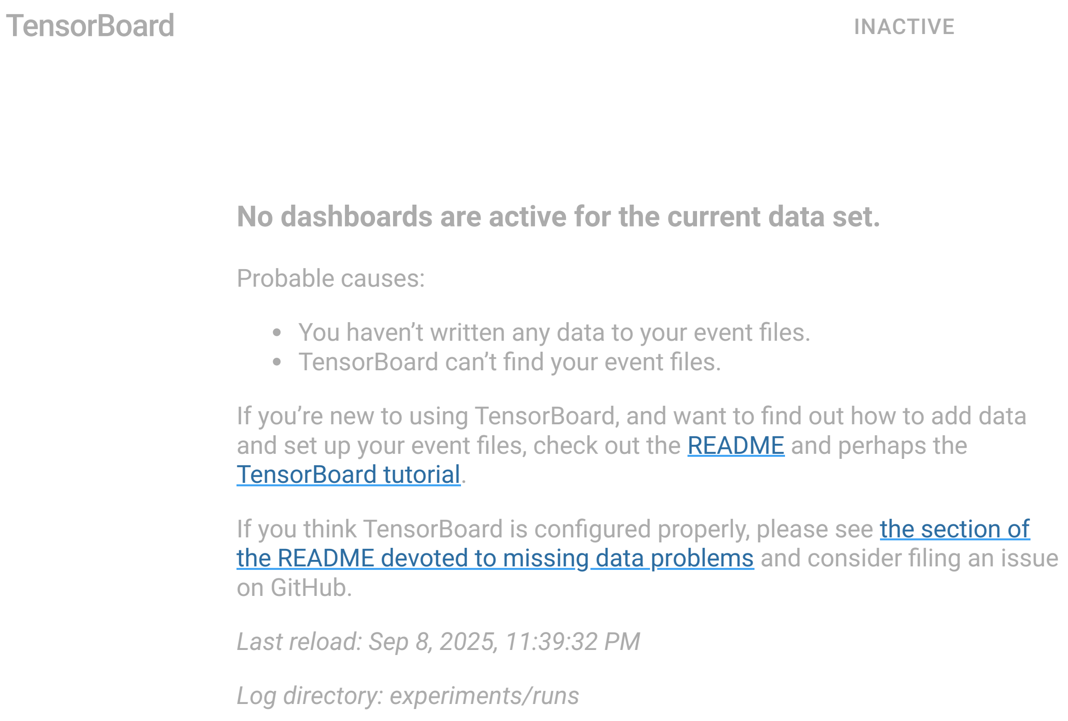
```
                (default): Linear(in_features=4096, out_features=8, bias=False)
              )
              (lora_B): ModuleDict(
                (default): Linear(in_features=8, out_features=4096, bias=False)
              )
              (lora_embedding_A): ParameterDict()
              (lora_embedding_B): ParameterDict()
              (lora_magnitude_vector): ModuleDict()
            )
            (k_proj): lora.Linear4bit(
              (base_layer): Linear4bit(in_features=4096, out_features=1024, bias=False)
              (lora_dropout): ModuleDict(
                (default): Dropout(p=0.2, 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=1024, bias=False)
              )
              (lora_embedding_A): ParameterDict()
              (lora_embedding_B): ParameterDict()
              (lora_magnitude_vector): ModuleDict()
            )
            (v_proj): lora.Linear4bit(
              (base_layer): Linear4bit(in_features=4096, out_features=1024, bias=False)
              (lora_dropout): ModuleDict(
                (default): Dropout(p=0.2, 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=1024, bias=False)
              )
              (lora_embedding_A): ParameterDict()
              (lora_embedding_B): ParameterDict()
              (lora_magnitude_vector): ModuleDict()
            )
            (o_proj): Linear4bit(in_features=4096, out_features=4096, bias=False)
          )
          )
        )
      )

lora_config = LoraConfig (
r=8,
lora_alpha=16,
target_modules=[
"self_attn.q_proj",
"self_attn.k_proj",
"self_attn.v_proj"
,],
lora_dropout=0.2,
bias="none",
task_type=TaskType.CAUSAL_LM,
)
model = prepare_model_for_kbit_training(model)
model = get_peft_model(model, lora_config)
```

```
/usr/local/lib/python3.12/dist-packages/peft/mapping_func.py:73: UserWarning:

You are trying to modify a model with PEFT for a second time. If you want to reload the model with a different confi

/usr/local/lib/python3.12/dist-packages/peft/tuners/tuners_utils.py:196: UserWarning:

Already found a `peft_config` attribute in the model. This will lead to having multiple adapters in the model. Make
```

```
model.print_trainable_parameters()
```

```
trainable params: 4,718,592 || all params: 8,035,045,376 || trainable%: 0.0587
```

```
OUTPUT_DIR = "experiments"
%load_ext tensorboard
%tensorboard --logdir "experiments/runs"
```



No dashboards are active for the current data set.

Probable causes:

- You haven’t written any data to your event files.
- TensorBoard can’t find your event files.

If you’re new to using TensorBoard, and want to find out how to add data and set up your event files, check out the README and perhaps the TensorBoard tutorial.

If you think TensorBoard is configured properly, please see the section of the README devoted to missing data problems and consider filing an issue on GitHub.

Last reload: Sep 8, 2025, 11:39:32 PM

Log directory: experiments/runs

```
sft_config = SFTConfig(
    output_dir=OUTPUT_DIR,              # where to save checkpoints + final model
    dataset_text_field="text",
    max_seq_length=512,
    num_train_epochs=1,
    per_device_train_batch_size=1,
    per_device_eval_batch_size=2,
    gradient_accumulation_steps=8,
    optim="paged_adamw_8bit",

    # ✅ Correct checkpointing + evaluation
    eval_strategy="steps",      # correct name
    eval_steps=200,                     # evaluate every 200 steps
    save_strategy="steps",              # save based on steps
    save_steps=200,                     # save every 200 steps
    save_total_limit=3,                 # keep only last 3 checkpoints

    logging_steps=10,                   # log training progress
    learning_rate=1e-4,
    fp16=True,                          # or bf16 if supported
    warmup_ratio=0.1,
    lr_scheduler_type="constant",
    report_to="tensorboard",            # enable logging to TensorBoard
    save_safetensors=True,              # use safetensors (smaller + safer)

    dataset_kwargs={
        "add_special_tokens": False,
        "append_concat_token": False
    },
    seed=SEED,
)



trainer = SFTTrainer(
model=model,
args=sft_config,
train_dataset=dataset ["train"],
eval_dataset=dataset ["validation"],
tokenizer=tokenizer,
data_collator=collator,
)
```

```
Map: 100%                                                4000/4000 [00:02<00:00, 1732.67 examples/s]
Map: 100%                                                500/500 [00:00<00:00, 1608.67 examples/s]
/usr/local/lib/python3.12/dist-packages/trl/trainer/sft_trainer.py:402: FutureWarning:

`tokenizer` is deprecated and will be removed in version 5.0.0 for `SFTTrainer.__init__`. Use `processing_class` ins
```

```
trainer.train()
```

```
/usr/local/lib/python3.12/dist-packages/torch/_dynamo/eval_frame.py:929: UserWarning:

torch.utils.checkpoint: the use_reentrant parameter should be passed explicitly. In version 2.5 we will raise an exc
```

[500/500 1:41:32, Epoch 1/1]

Step	Training Loss	Validation Loss
200	1.277600	1.117799
400	1.011900	1.091459

```
/usr/local/lib/python3.12/dist-packages/torch/_dynamo/eval_frame.py:929: UserWarning:

torch.utils.checkpoint: the use_reentrant parameter should be passed explicitly. In version 2.5 we will raise an exc

/usr/local/lib/python3.12/dist-packages/torch/_dynamo/eval_frame.py:929: UserWarning:

torch.utils.checkpoint: the use_reentrant parameter should be passed explicitly. In version 2.5 we will raise an exc

TrainOutput(global_step=500, training_loss=1.112665370941162, metrics={'train_runtime': 6101.7852,
'train_samples_per_second': 0.656, 'train_steps_per_second': 0.082, 'total_flos': 4.07747021962199e+16,
'train_loss': 1.112665370941162, 'epoch': 1.0})
```

```
OUTPUT_DIR = "/content/llama3-medquad-qlora"

# Save adapters
model.save_pretrained(OUTPUT_DIR)
tokenizer.save_pretrained(OUTPUT_DIR)
```

```
('/content/llama3-medquad-qlora/tokenizer_config.json',
 '/content/llama3-medquad-qlora/special_tokens_map.json',
 '/content/llama3-medquad-qlora/chat_template.jinja',
 '/content/llama3-medquad-qlora/tokenizer.json')
```

```
from huggingface_hub import HfApi
from huggingface_hub import Repository

# Upload using HF API
model.push_to_hub("Arushp1/llama3-medquad-qlora")
tokenizer.push_to_hub("Arushp1/llama3-medquad-qlora")
```

```
Processing Files (1 / 1)              : 100%     18.9MB / 18.9MB, 8.59MB/s

New Data Upload                          : 100%     18.9MB / 18.9MB, 8.59MB/s

  ...uad-qlora/adapter_model.safetensors: 100%                                        18.9MB / 18.9MB

README.md:      5.17k/? [00:00<00:00, 268kB/s]

Processing Files (1 / 1)              : 100%     17.2MB / 17.2MB, 1.89MB/s

New Data Upload                          : 100%     16.9MB / 16.9MB, 1.89MB/s

  llama3-medquad-qlora/tokenizer.json    : 100%                                        17.2MB / 17.2MB
```

```
CommitInfo(commit_url='https://huggingface.co/Arushp1/llama3-medquad-qlora/commit/adb739aa327f67a212ccd5e76fa5dc17b12404c8', commit_message='Upload tokenizer', commit_description='',
oid='adb739aa327f67a212ccd5e76fa5dc17b12404c8', pr_url=None,
repo_url=RepoUrl('https://huggingface.co/Arushp1/llama3-medquad-qlora', endpoint='https://huggingface.co',
```

```
model.push_to_hub("Arushp1/llama3-8b-medquad-qlora",tokenizer=tokenizer,max_shard_size="5GB")
```

```
Processing Files (1 / 1)              : 100%     18.9MB / 18.9MB, 3.60MB/s

New Data Upload                          :         0.00B /   0.00B,   0.00B/s

  ...p1xlamjfs/adapter_model.safetensors: 100%                                        18.9MB / 18.9MB
```

```
CommitInfo(commit_url='https://huggingface.co/Arushp1/llama3-8b-medquad-qlora/commit/02c308ffd51ae035a46d747687c014a8555c1584', commit_message='Upload model', commit_description='',
oid='02c308ffd51ae035a46d747687c014a8555c1584', pr_url=None,
repo_url=RepoUrl('https://huggingface.co/Arushp1/llama3-8b-medquad-qlora', endpoint='https://huggingface.co',
repo_type='model', repo_id='Arushp1/llama3-8b-medquad-qlora'), pr_revision=None, pr_num=None)
```

```
import shutil

# Zip the model folder
shutil.make_archive("/content/llama3-medquad-qlora", 'zip', OUTPUT_DIR)
```

```
'/content/llama3-medquad-qlora.zip'
```

```
from google.colab import files
files.download("/content/llama3-medquad-qlora.zip")
```

```
from peft import PeftModel

base_model = AutoModelForCausalLM.from_pretrained("meta-llama/Meta-Llama-3-8B-Instruct", device_map="auto")
model = PeftModel.from_pretrained(base_model, "Arushp1/llama3-medquad-qlora")
```