```
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM,
BitsAndBytesConfig
import logging
from datasets import load_dataset
import pandas
from peft import prepare_model_for_kbit_training, LoraConfig,
get_peft_model

model_name = "meta-llama/Llama-2-7b-chat-hf"
dataset = load_dataset("lamini/lamini_docs")
use_hf = True
token = "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"

bnb_config = BitsAndBytesConfig(
    load_in_8bit=True,
    bnb_8bit_use_double_quant=True,
    bnb_8bit_quant_type="nf4",
    bnb_8bit_compute_dtype=torch.bfloat16
)

tokenizer = AutoTokenizer.from_pretrained(model_name, token = token)
tokenizer.pad_token = tokenizer.eos_token

def tokenize_function(examples):
    if "question" in examples and "answer" in examples:
      text = examples["question"][0] + examples["answer"][0]
    elif "input" in examples and "output" in examples:
      text = examples["input"][0] + examples["output"][0]
    else:
      text = examples["text"][0]

    tokenizer.pad_token = tokenizer.eos_token
    tokenized_inputs = tokenizer(
        text,
        return_tensors="np",
        padding=True,
    )

    max_length = min(
        tokenized_inputs["input_ids"].shape[1],
        2048
    )
    tokenizer.truncation_side = "left"
    tokenized_inputs = tokenizer(
        text,
        return_tensors="np",
        truncation=True,
        max_length=max_length
    )
    return tokenized_inputs

finetuning_dataset_loaded = dataset

tokenized_dataset = finetuning_dataset_loaded.map(
    tokenize_function,
    batched=True,
    batch_size=1,
    drop_last_batch=True
)

print(tokenized_dataset)

DatasetDict({
    train: Dataset({
        features: ['question', 'answer', 'input_ids',
'attention_mask', 'labels'],
        num_rows: 1260
    })
    test: Dataset({
        features: ['question', 'answer', 'input_ids',
'attention_mask', 'labels'],
        num_rows: 140
    })
})

train_data = tokenized_dataset["train"]
test_data = tokenized_dataset["test"]

print(train_data)
print(test_data)

Dataset({
    features: ['question', 'answer', 'input_ids', 'attention_mask',
'labels'],
    num_rows: 1260
})
Dataset({
    features: ['question', 'answer', 'input_ids', 'attention_mask',
'labels'],
    num_rows: 140
})

base_model = AutoModelForCausalLM.from_pretrained(model_name,
quantization_config=bnb_config, device_map={"":0}, token=token)


===================================BUG
REPORT===================================
Welcome to bitsandbytes. For bug reports, please submit your error
```

```
trace to: https://github.com/TimDettmers/bitsandbytes/issues
==================================================================================
binary_path: C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\
bitsandbytes\cuda_setup\libbitsandbytes_cuda116.dll
CUDA SETUP: Loading binary C:\Users\adity\miniconda3\envs\tfgpu\lib\
site-packages\bitsandbytes\cuda_setup\libbitsandbytes_cuda116.dll...

{"model_id":"cd9e8414396544008f0e609211c01659","version_major":2,"vers
ion_minor":0}

base_model.gradient_checkpointing_enable()
base_model = prepare_model_for_kbit_training(base_model)

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}"
    )

config = LoraConfig(
    r=8,
    lora_alpha=32,
    target_modules=["q_proj", "v_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)

base_model = get_peft_model(base_model, config)
print_trainable_parameters(base_model)

trainable params: 4194304 || all params: 6742609920 || trainable%:
0.06220594176090199

import transformers
trainer = transformers.Trainer(
    model=base_model,
    train_dataset=train_data,
    args=transformers.TrainingArguments(
        per_device_train_batch_size=1,
        gradient_accumulation_steps=4,
        warmup_steps=2,
        max_steps=3000,
        learning_rate=1.5e-4,
        fp16=True,
        logging_steps=10,
        output_dir="outputs",
        optim="adafactor"
    ),

data_collator=transformers.DataCollatorForLanguageModeling(tokenizer,
mlm=False),
)
base_model.config.use_cache = False  # silence the warnings. Please
re-enable for inference!
trainer.train()
```

```
C:\Users\adity\AppData\Roaming\Python\Python310\site-packages\
accelerate\accelerator.py:437: FutureWarning: Passing the following
arguments to `Accelerator` is deprecated and will be removed in
version 1.0 of Accelerate: dict_keys(['dispatch_batches',
'split_batches', 'even_batches', 'use_seedable_sampler']). Please pass
an `accelerate.DataLoaderConfiguration` instead:
dataloader_config = DataLoaderConfiguration(dispatch_batches=None,
split_batches=False, even_batches=True, use_seedable_sampler=True)
  warnings.warn(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")

<IPython.core.display.HTML object>

Checkpoint destination directory outputs\checkpoint-500 already exists
and is non-empty. Saving will proceed but saved results may be
invalid.
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-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(
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\bitsandbytes\
autograd\_functions.py:298: UserWarning: MatMul8bitLt: inputs will be
cast from torch.float32 to float16 during quantization
  warnings.warn(f"MatMul8bitLt: inputs will be cast from {A.dtype} to
float16 during quantization")
```

```
TrainOutput(global_step=3000, training_loss=0.7388844806353251,
metrics={'train_runtime': 16568.175, 'train_samples_per_second':
0.724, 'train_steps_per_second': 0.181, 'total_flos':
4.029814345310208e+16, 'train_loss': 0.7388844806353251, 'epoch':
9.52})
```

```python
from huggingface_hub import notebook_login

notebook_login()
```

{"model_id":"01d1951a95fd44bcb1c9ac14a180dacf","version_major":2,"vers
ion_minor":0}

```python
base_model.push_to_hub("AdityaSingh312/Llama-7b-lamini-docs",
                  use_auth_token=True,
                  commit_message="basic training",
                  private=True)
```

```
C:\Users\adity\AppData\Roaming\Python\Python310\site-packages\
transformers\utils\hub.py:834: FutureWarning: The `use_auth_token`
argument is deprecated and will be removed in v5 of Transformers.
Please use `token` instead.
  warnings.warn(
```

{"model_id":"0a278da498af4f86b74b1201f9d33261","version_major":2,"vers
ion_minor":0}

```
C:\Users\adity\miniconda3\envs\tfgpu\lib\site-packages\
huggingface_hub\file_download.py:149: UserWarning: `huggingface_hub`
cache-system uses symlinks by default to efficiently store duplicated
files but your machine does not support them in C:\Users\adity\.cache\
huggingface\hub\models--AdityaSingh312--Llama-7b-lamini-docs. Caching
files will still work but in a degraded version that might require
more space on your disk. This warning can be disabled by setting the
`HF_HUB_DISABLE_SYMLINKS_WARNING` environment variable. For more
details, see https://huggingface.co/docs/huggingface_hub/how-to-
cache#limitations.
To support symlinks on Windows, you either need to activate Developer
Mode or to run Python as an administrator. In order to see activate
developer mode, see this article:
```

```
https://docs.microsoft.com/en-us/windows/apps/get-started/enable-your-
device-for-development
  warnings.warn(message)
```

{"model_id":"6cd1bce6c1c94b25868cc4639a7b2586","version_major":2,"version_minor":0}

```
CommitInfo(commit_url='https://huggingface.co/AdityaSingh312/Llama-7b-
lamini-docs/commit/40e85684fac130bf658d8ae37bbca873053a25ce',
commit_message='basic training', commit_description='',
oid='40e85684fac130bf658d8ae37bbca873053a25ce', pr_url=None,
pr_revision=None, pr_num=None)
```