English-to-German Machine Translation using Falcon-LoRA

Task: Text-to-text

In this project, I fine-tuned **Falcon-1B** (developed by TII, Dubai) with **LoRA** (Low-Rank Adaptation) to build an efficient **English-to-German translation system.** The implementation leverages **parameter-efficient fine-tuning** to achieve high-quality translations while minimizing GPU resource requirements, making it suitable for environments with limited computational resources.

Data Source: English-German

Key Steps Overview:

1. **Data Preparation**: Curate a parallel English-German corpus.
2. **Tokenizer and Model Loading**: Load Falcon-1B model and tokenizer from Hugging Face, incorporating **1-bit quantization** for efficient memory usage.
3. **LoRA Configuration**: Apply **LoRA** (Low-Rank Adaptation) to inject trainable adapters into Falcon's attention layers.
4. **Training**: Fine-tune the model on a single GPU (e.g., NVIDIA T4 or A10G) using Hugging Face's Transformers library.
5. **Evaluation:** Model performance evaluation using **BLEU score**.

```
# Install required libraries
!pip install -q transformers datasets accelerate peft bitsandbytes
!pip install gradio
!pip install datasets

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 76.0/76.0 MB 9.9 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 3.9 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 105.0 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 24.5 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 49.8 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 2.0 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 6.5 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 14.7 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 8.7 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.6 MB/s eta
```

```
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 66.2 MB/s eta
0:00:00
ent already satisfied: gradio in /usr/local/lib/python3.11/dist-
packages (5.23.3)
Requirement already satisfied: aiofiles<24.0,>=22.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (23.2.1)
Requirement already satisfied: anyio<5.0,>=3.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (4.9.0)
Requirement already satisfied: fastapi<1.0,>=0.115.2 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.115.12)
Requirement already satisfied: ffmpy in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.8.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (1.8.0)
Requirement already satisfied: groovy~=0.1 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.1.2)
Requirement already satisfied: httpx>=0.24.1 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub>=0.28.1 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.30.1)
Requirement already satisfied: jinja2<4.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,>=1.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (2.0.2)
Requirement already satisfied: orjson~=3.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (3.10.16)
Requirement already satisfied: packaging in
/usr/local/lib/python3.11/dist-packages (from gradio) (24.2)
Requirement already satisfied: pandas<3.0,>=1.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (11.1.0)
Requirement already satisfied: pydantic<2.12,>=2.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (2.11.1)
Requirement already satisfied: pydub in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.25.1)
Requirement already satisfied: python-multipart>=0.0.18 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.0.20)
Requirement already satisfied: pyyaml<7.0,>=5.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (6.0.2)
Requirement already satisfied: ruff>=0.9.3 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.11.3)
Requirement already satisfied: safehttpx<0.2.0,>=0.1.6 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (2.10.0)
```

```
Requirement already satisfied: starlette<1.0,>=0.40.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.46.1)
Requirement already satisfied: tomlkit<0.14.0,>=0.12.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.13.2)
Requirement already satisfied: typer<1.0,>=0.12 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.15.2)
Requirement already satisfied: typing-extensions~=4.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (4.13.0)
Requirement already satisfied: uvicorn>=0.14.0 in
/usr/local/lib/python3.11/dist-packages (from gradio) (0.34.0)
Requirement already satisfied: fsspec in
/usr/local/lib/python3.11/dist-packages (from gradio-client==1.8.0-
>gradio) (2024.12.0)
Requirement already satisfied: websockets<16.0,>=10.0 in
/usr/local/lib/python3.11/dist-packages (from gradio-client==1.8.0-
>gradio) (15.0.1)
Requirement already satisfied: idna>=2.8 in
/usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio)
(3.10)
Requirement already satisfied: sniffio>=1.1 in
/usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio)
(1.3.1)
Requirement already satisfied: certifi in
/usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio)
(2025.1.31)
Requirement already satisfied: httpcore==1.* in
/usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio)
(1.0.7)
Requirement already satisfied: h11<0.15,>=0.13 in
/usr/local/lib/python3.11/dist-packages (from httpcore==1.*-
>httpx>=0.24.1->gradio) (0.14.0)
Requirement already satisfied: filelock in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1-
>gradio) (3.18.0)
Requirement already satisfied: requests in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1-
>gradio) (2.32.3)
Requirement already satisfied: tqdm>=4.42.1 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1-
>gradio) (4.67.1)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0-
>gradio) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0-
>gradio) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0-
>gradio) (2025.2)
```

```
Requirement already satisfied: annotated-types>=0.6.0 in
/usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0-
>gradio) (0.7.0)
Requirement already satisfied: pydantic-core==2.33.0 in
/usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0-
>gradio) (2.33.0)
Requirement already satisfied: typing-inspection>=0.4.0 in
/usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0-
>gradio) (0.4.0)
Requirement already satisfied: click>=8.0.0 in
/usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12-
>gradio) (8.1.8)
Requirement already satisfied: shellingham>=1.3.0 in
/usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12-
>gradio) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in
/usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12-
>gradio) (13.9.4)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2-
>pandas<3.0,>=1.0->gradio) (1.17.0)
Requirement already satisfied: markdown-it-py>=2.2.0 in
/usr/local/lib/python3.11/dist-packages (from rich>=10.11.0-
>typer<1.0,>=0.12->gradio) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in
/usr/local/lib/python3.11/dist-packages (from rich>=10.11.0-
>typer<1.0,>=0.12->gradio) (2.18.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests->huggingface-
hub>=0.28.1->gradio) (3.4.1)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests->huggingface-
hub>=0.28.1->gradio) (2.3.0)
Requirement already satisfied: mdurl~=0.1 in
/usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0-
>rich>=10.11.0->typer<1.0,>=0.12->gradio) (0.1.2)
Requirement already satisfied: datasets in
/usr/local/lib/python3.11/dist-packages (3.5.0)
Requirement already satisfied: filelock in
/usr/local/lib/python3.11/dist-packages (from datasets) (3.18.0)
Requirement already satisfied: numpy>=1.17 in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
```

```
/usr/local/lib/python3.11/dist-packages (from datasets) (2.32.3)
Requirement already satisfied: tqdm>=4.66.3 in
/usr/local/lib/python3.11/dist-packages (from datasets) (4.67.1)
Requirement already satisfied: xxhash in
/usr/local/lib/python3.11/dist-packages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in
/usr/local/lib/python3.11/dist-packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec<=2024.12.0,>=2023.1.0 in
/usr/local/lib/python3.11/dist-packages (from
fsspec[http]<=2024.12.0,>=2023.1.0->datasets) (2024.12.0)
Requirement already satisfied: aiohttp in
/usr/local/lib/python3.11/dist-packages (from datasets) (3.11.15)
Requirement already satisfied: huggingface-hub>=0.24.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (0.30.1)
Requirement already satisfied: packaging in
/usr/local/lib/python3.11/dist-packages (from datasets) (24.2)
Requirement already satisfied: pyyaml>=5.1 in
/usr/local/lib/python3.11/dist-packages (from datasets) (6.0.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(2.6.1)
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.3.2)
Requirement already satisfied: attrs>=17.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(25.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(6.3.0)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.18.3)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.24.0-
>datasets) (4.13.0)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
```

```
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (2025.1.31)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2025.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2025.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2-
>pandas->datasets) (1.17.0)
```

```python
import torch
from transformers import (
    AutoTokenizer,
    AutoModelForCausalLM,
    TrainingArguments,
    Trainer,
    DataCollatorForLanguageModeling,
    BitsAndBytesConfig
)
from datasets import Dataset
from peft import get_peft_model, LoraConfig, TaskType

# Quantization configuration for reduced memory usage (helpful for
Colab)
quantization_config = BitsAndBytesConfig(
    load_in_8bit=True,  # Enable 8-bit quantization
    llm_int8_threshold=6.0
)

class FalconWrapper(AutoModelForCausalLM):
    def forward(self, *args, **kwargs):
        # Remove unsupported arguments
        kwargs.pop("num_items_in_batch", None)
        return super().forward(*args, **kwargs)

# Configure model and tokenizer (I am trying distilgpt2 as an
alternative for lower GPU usage)
model_name = "distilgpt2"
tokenizer = AutoTokenizer.from_pretrained(model_name,
trust_remote_code=True)
tokenizer.pad_token = tokenizer.eos_token

# Load model with quantization and mixed precision
model = FalconWrapper.from_pretrained(model_name,
torch_dtype=torch.float16, device_map="auto")

def load_and_prepare_data(file_path, sample_fraction=0.3):
    raw_lines = []
    # Read all lines
    with open(file_path, 'r', encoding='utf-8') as f:
        lines = f.readlines()

    # Calculate sample size and slice
    sample_size = int(len(lines) * sample_fraction)
    lines = lines[:sample_size]

    # Process lines into input-output pairs
    for line in lines:
        parts = line.strip().split('\t')
        if len(parts) >= 2:
            raw_lines.append({'input': parts[0], 'output': parts[1]})

    return Dataset.from_list(raw_lines)

# Load and split data (80% train, 20% validation)
dataset =
load_and_prepare_data('deu.txt').train_test_split(test_size=0.2,
seed=42)

# Formatting function
def format_dataset(example):
    return {
        "text": f"Translate English to German:\nEnglish:
{example['input']}\nGerman:",
        "labels": example['output']
    }

train_dataset = dataset['train'].map(format_dataset)
val_dataset = dataset['test'].map(format_dataset)

{"model_id":"8835dedcd7ca4360828ac9469a964a03","version_major":2,"vers
ion_minor":0}

{"model_id":"0ebd709ab86e4974bb4c6689e1c5db8d","version_major":2,"vers
ion_minor":0}

# Tokenization with proper padding/truncation
def tokenize_function(examples):
    model_inputs = tokenizer(
        examples["input"],
        max_length=30,
        padding="max_length",
        truncation=True
    )

    with tokenizer.as_target_tokenizer():
        labels = tokenizer(
            examples["output"],
            max_length=30,
            padding="max_length",
            truncation=True
        )

    model_inputs["labels"] = labels["input_ids"]
    return model_inputs

# Apply tokenization
tokenized_train = train_dataset.map(tokenize_function, batched=True)
tokenized_val = val_dataset.map(tokenize_function, batched=True)
```

{"model_id":"bb56b98c126343adb905d467a898948a","version_major":2,"version_minor":0}

```
/usr/local/lib/python3.11/dist-packages/transformers/
tokenization_utils_base.py:3980: UserWarning: `as_target_tokenizer` is
deprecated and will be removed in v5 of Transformers. You can tokenize
your labels by using the argument `text_target` of the regular
`__call__` method (either in the same call as your input texts if you
use the same keyword arguments, or in a separate call.
  warnings.warn(
```

{"model_id":"7fa832b1f2ca42698348335c11edba20","version_major":2,"version_minor":0}

```
# Remove unnecessary columns
tokenized_train = tokenized_train.remove_columns(["text", "input",
"output"])
tokenized_val = tokenized_val.remove_columns(["text", "input",
"output"])

# Configure LoRA (lightweight)
peft_config = LoraConfig(
    task_type=TaskType.CAUSAL_LM,
    r=4,
    lora_alpha=16,
    lora_dropout=0.05,
    bias="none"

)
```

```
model = get_peft_model(model, peft_config)
model.print_trainable_parameters()
```

```
trainable params: 73,728 || all params: 81,986,304 || trainable%:
0.0899
```

```
/usr/local/lib/python3.11/dist-packages/peft/tuners/lora/
layer.py:1264: UserWarning: fan_in_fan_out is set to False but the
target module is `Conv1D`. Setting fan_in_fan_out to True.
  warnings.warn(
```

```
# Training arguments
training_args = TrainingArguments(
    output_dir="lora-falcon-output",
    max_steps=300,
    per_device_train_batch_size=1,
    gradient_accumulation_steps=4,  # Increase accumulation steps to
maintain effective batch size
    learning_rate=5e-5,
    fp16=True,  # Enable mixed precision training
    logging_steps=50,
    save_strategy="no",
    evaluation_strategy="no",
    report_to="none",
    remove_unused_columns=False,
    warmup_steps=10,
    dataloader_num_workers=2  # Increase workers for better I/O
throughput
)
```

```
/usr/local/lib/python3.11/dist-packages/transformers/
training_args.py:1611: FutureWarning: `evaluation_strategy` is
deprecated and will be removed in version 4.46 of □ Transformers. Use
`eval_strategy` instead
  warnings.warn(
```

```
# Data collator
data_collator = DataCollatorForLanguageModeling(
    tokenizer=tokenizer,
    mlm=False
)

# Trainer
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_train,
    eval_dataset=tokenized_val,
    tokenizer=tokenizer,
    data_collator=data_collator
)
```

```
<ipython-input-48-d5f8dc6f8b5a>:2: FutureWarning: `tokenizer` is
deprecated and will be removed in version 5.0.0 for
`Trainer.__init__`. Use `processing_class` instead.
  trainer = Trainer(
No label_names provided for model class `PeftModelForCausalLM`. Since
`PeftModel` hides base models input arguments, if label_names is not
given, label_names can't be set automatically within `Trainer`. Note
that empty label_names list will be used instead.
```

```
# Start training
print("Starting training...")
trainer.train()
```

```
Starting training...
```

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

```
TrainOutput(global_step=300, training_loss=4.166671040852864,
metrics={'train_runtime': 37.6468, 'train_samples_per_second': 31.875,
'train_steps_per_second': 7.969, 'total_flos': 9202139136000.0,
'train_loss': 4.166671040852864, 'epoch': 0.022570391408204337})
```

```
# Save the model
model.save_pretrained("lora-falcon-finetuned")
tokenizer.save_pretrained("lora-falcon-finetuned")
```

```
('lora-falcon-finetuned/tokenizer_config.json',
 'lora-falcon-finetuned/special_tokens_map.json',
 'lora-falcon-finetuned/vocab.json',
 'lora-falcon-finetuned/merges.txt',
 'lora-falcon-finetuned/added_tokens.json',
 'lora-falcon-finetuned/tokenizer.json')
```

Data Preparation:

- I used a curated parallel **English-German corpus** for training the translation model. This dataset consists of pairs of English sentences and their German translations, formatted into a tab-separated text file.

LoRA Configuration:

- To make the model more parameter-efficient, I applied **LoRA** (Low-Rank Adaptation). LoRA injects trainable low-rank adapters into the attention layers, allowing for fine-tuning with fewer parameters and reduced memory consumption.
- This method reduces computational cost while maintaining high performance for the task.

Simple Training Setup (Free Google Colab)

I'm using the **free version of Google Colab**, so the training setup is simplified for low GPU resources:

- **Epochs**: 1
- **Batch size**: 1
- **Gradient accumulation**: 2
- **FP16**: Enabled to save memory
- **Warmup steps**: 10
- **Save & Eval**: Once per epoch
- **DataLoader workers**: 0

This setup helps avoid out-of-memory errors and runs smoothly on limited GPUs like Tesla T4.

Inference:

- Finally, I used the trained model to perform **inference** on a few sample sentences. The model generates the German translation for an English input, which can be evaluated by comparing it to the ground truth translation.

```
# Inference function
def generate_translation(model, tokenizer, english_text):
    prompt = f"Translate English to German:\nEnglish: {english_text}\nGerman:"
    inputs = tokenizer(prompt, return_tensors="pt").to(model.device)

    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_new_tokens=50,
            num_beams=5,
            early_stopping=True,
            temperature=0.7
        )

    # Decode only the generated German text
    full_output = tokenizer.decode(outputs[0], skip_special_tokens=True)
    german_translation = full_output.split("German:")[1].strip()
    return german_translation

# Test translation
test_text = "Where is the train station?"
translation = generate_translation(model, tokenizer, test_text)
print(f"\nTest Translation:\nEnglish: {test_text}\nGerman: {translation}")
```

```
/usr/local/lib/python3.11/dist-packages/transformers/generation/
configuration_utils.py:628: UserWarning: `do_sample` is set to
`False`. However, `temperature` is set to `0.7` -- this flag is only
used in sample-based generation modes. You should set `do_sample=True`
or unset `temperature`.
  warnings.warn(
Setting `pad_token_id` to `eos_token_id`:50256 for open-end
generation.


Test Translation:
English: Where is the train station?
German: English: Where is the train station?
```

```python
from transformers import pipeline # Import the pipeline function

# Initialize the translator
translator = pipeline(task="translation", model="lora-falcon-
finetuned", tokenizer="lora-falcon-finetuned")
```

```
Device set to use cuda:0
The model 'GPT2LMHeadModel' is not supported for translation.
Supported models are ['BartForConditionalGeneration',
'BigBirdPegasusForConditionalGeneration',
'BlenderbotForConditionalGeneration',
'BlenderbotSmallForConditionalGeneration', 'EncoderDecoderModel',
'FSMTForConditionalGeneration',
'GPTSanJapaneseForConditionalGeneration',
'LEDForConditionalGeneration', 'LongT5ForConditionalGeneration',
'M2M100ForConditionalGeneration', 'MarianMTModel',
'MBartForConditionalGeneration', 'MT5ForConditionalGeneration',
'MvpForConditionalGeneration', 'NllbMoeForConditionalGeneration',
'PegasusForConditionalGeneration', 'PegasusXForConditionalGeneration',
'PLBartForConditionalGeneration',
'ProphetNetForConditionalGeneration',
'Qwen2AudioForConditionalGeneration', 'SeamlessM4TForTextToText',
'SeamlessM4Tv2ForTextToText',
'SwitchTransformersForConditionalGeneration',
'T5ForConditionalGeneration', 'UMT5ForConditionalGeneration',
'XLMProphetNetForConditionalGeneration'].
```

```python
import gradio as gr
```

```python
import gradio as gr
from transformers import pipeline

# Load the fine-tuned Falcon model with LoRA adapters
translator = pipeline(
    task="translation",
    model="lora-falcon-finetuned",
    tokenizer="lora-falcon-finetuned"
)
```

```
Device set to use cuda:0
The model 'GPT2LMHeadModel' is not supported for translation.
Supported models are ['BartForConditionalGeneration',
'BigBirdPegasusForConditionalGeneration',
'BlenderbotForConditionalGeneration',
'BlenderbotSmallForConditionalGeneration', 'EncoderDecoderModel',
'FSMTForConditionalGeneration',
'GPTSanJapaneseForConditionalGeneration',
'LEDForConditionalGeneration', 'LongT5ForConditionalGeneration',
'M2M100ForConditionalGeneration', 'MarianMTModel',
'MBartForConditionalGeneration', 'MT5ForConditionalGeneration',
'MvpForConditionalGeneration', 'NllbMoeForConditionalGeneration',
'PegasusForConditionalGeneration', 'PegasusXForConditionalGeneration',
'PLBartForConditionalGeneration',
'ProphetNetForConditionalGeneration',
'Qwen2AudioForConditionalGeneration', 'SeamlessM4TForTextToText',
'SeamlessM4Tv2ForTextToText',
'SwitchTransformersForConditionalGeneration',
'T5ForConditionalGeneration', 'UMT5ForConditionalGeneration',
'XLMProphetNetForConditionalGeneration'].

# Define the translation function
def translate_text(text):
    if not text.strip():
        return "Please enter some English text to translate."
    result = translator(text)[0]['translation_text']
    return result

# Interface styling
title = "□ English-to-German Translator"
description = """
□ This web application uses a **fine-tuned Falcon-1B model with LoRA
(Low-Rank Adaptation)** to translate English sentences into German
efficiently.□ The model was trained by **Subrat Kumar** with a focus
on performance and low GPU consumption using parameter-efficient fine-
tuning techniques.□ Just enter any English sentence below and click
**Translate** to get its German equivalent.
"""

# Launch the Gradio Interface
interface = gr.Interface(
    fn=translate_text,
    inputs=gr.Textbox(
        label="□ Enter English Text",
        placeholder="Type something like: 'How are you today?'",
        lines=3
    ),
    outputs=gr.Textbox(
        label="□□ German Translation",
        lines=3
    ),
    title=title,
    description=description,
    theme="soft",  # Optional: switch to a soft color palette
    allow_flagging="never",
    examples=["Good morning!", "What is your name?", "I love machine
learning."]
)
```

```
/usr/local/lib/python3.11/dist-packages/gradio/interface.py:415:
UserWarning: The `allow_flagging` parameter in `Interface` is
deprecated.Use `flagging_mode` instead.
  warnings.warn(
```

```
interface.launch()
```

```
Running Gradio in a Colab notebook requires sharing enabled.
Automatically setting `share=True` (you can turn this off by setting
`share=False` in `launch()` explicitly).

Colab notebook detected. To show errors in colab notebook, set
debug=True in launch()
* Running on public URL: https://abd6ca85751074680a.gradio.live

This share link expires in 72 hours. For free permanent hosting and
GPU upgrades, run `gradio deploy` from the terminal in the working
directory to deploy to Hugging Face Spaces
(https://huggingface.co/spaces)

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

Challenges Faced:

Due to the limited dataset and smaller model size, early outputs weren't very accurate. I noticed it sometimes repeated phrases or gave generic responses.

Next Steps: I will improve this by tweaking the **prompt format**, using **beam search**, and will adjust **temperature** for more diverse and meaningful German translations. I will also consider the whole data instate of 30 parcent.