Project 1 — Tiny GPT-2 Q&A; Chatbot

Why This Project Matters

This is your “Hello World” of fine-tuning. GPT-2 is small enough to run on Google Colab (free tier) or a low-end GPU. You’ll learn the entire workflow: dataset prep → model loading → training → inference. Skills from here carry forward to bigger models.

Prerequisites

- Basic Python (loops, functions, variables) - Hugging Face Transformers installation: pip install transformers datasets accelerate - What is a tokenizer? Tokenizers split text into tokens the model understands. - GPU access (Colab: Runtime → Change runtime type → GPU)

Dataset

```
For Q&A, we’ll use the SQuAD v2 dataset from Hugging Face.
Link: https://huggingface.co/datasets/squad_v2
Size: ~150k question-answer pairs.

Download & load:
from datasets import load_dataset
dataset = load_dataset("squad_v2")
print(dataset)
```

Step-by-Step Build

```
Step 1: Load tokenizer & model:
from transformers import GPT2Tokenizer, GPT2LMHeadModel
model_name = "gpt2"
tokenizer = GPT2Tokenizer.from_pretrained(model_name)
model = GPT2LMHeadModel.from_pretrained(model_name)

Step 2: Preprocess dataset:
def preprocess(example):
    prompt = f"Question: {example['question']} Answer: {example['answers']['text'][0] if example
    tokens = tokenizer(prompt, truncation=True, padding="max_length", max_length=128)
    tokens["labels"] = tokens["input_ids"].copy()
    return tokens

tokenized_dataset = dataset.map(preprocess, batched=True)

Step 3: Training loop:
from transformers import Trainer, TrainingArguments
training_args = TrainingArguments(
    output_dir="./gpt2-qa",
    evaluation_strategy="epoch",
    learning_rate=5e-5,
    per_device_train_batch_size=2,
    num_train_epochs=3,
    weight_decay=0.01,
    push_to_hub=False
)
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    eval_dataset=tokenized_dataset["validation"]
)
trainer.train()

Step 4: Save & test:
trainer.save_model("./gpt2-qa")
input_text = "Question: What is fine-tuning?"
inputs = tokenizer(input_text, return_tensors="pt")
outputs = model.generate(**inputs, max_length=50)
print(tokenizer.decode(outputs[0]))
```

Design Thinking

- GPT-2 small fits in <2GB VRAM, fast iterations. - Concatenating Q&A; into one prompt trains GPT-2 on our format. - 128 tokens keeps training fast while covering most questions and answers.

Troubleshooting

Problem	Cause	Fix
CUDA out of memory	Batch too big	Reduce per_device_train_batch_size
Tokenizer error	Missing pad token	tokenizer.pad_token = tokenizer.eos_token
Model outputs nonsense	Not enough epochs	Increase num_train_epochs

Flowchart

[Dataset] → [Preprocess Q&A] → [Tokenization] → [Model Load] → [Training Loop] → [Save Model]

Mini Quiz

1. Why set labels = input_ids in fine-tuning?
2. Purpose of truncation?
3. Why start with GPT-2 small?

Answers:
1. Model learns to predict next tokens.
2. Prevents input from exceeding max length.
3. Faster and less VRAM heavy.

Side Quest

- Use your own Q&A dataset.
- Try distilgpt2 for faster runs.
- Experiment with temperature in generate() for creative answers.