```
%%capture
# 1. Force-remove existing conflicting packages
!pip uninstall -y unsloth bitsandbytes accelerator transformers

# 2. Install the specific 'Fast' versions for Llama 3.2
!pip install --no-cache-dir "unsloth[colab-new] @ git+https://github.com/unslothai/unsloth.git"
!pip install --no-cache-dir --no-deps trl peft accelerate bitsandbytes
```

```
import torch
import json
from datasets import Dataset
from transformers import (
    AutoTokenizer,
    AutoModelForCausalLM,
    BitsAndBytesConfig,
    set_seed
)
from peft import LoraConfig, get_peft_model, prepare_model_for_kbit_training
from trl import SFTTrainer, SFTConfig
from google.colab import userdata  # <--- Change 1: Use Colab userdata

# 0. CONFIGURATION
set_seed(42)
MODEL_NAME = "meta-llama/Llama-3.2-1B-Instruct"

# Change 2: Point this to where you uploaded your file in Colab
DATASET_PATH = "/content/history_dataset_3000_relational_patched.json"

# Securely fetch token from Colab Secrets
HF_TOKEN = userdata.get("HF_TOKEN").strip() # <--- Change 3: Pull secret from Colab vault and strip whitespace

# 1. LOAD MODEL & TOKENIZER
tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME, token=HF_TOKEN)
tokenizer.pad_token = tokenizer.eos_token

bnb_config = BitsAndBytesConfig(
    load_in_4bit=True,
    bnb_4bit_quant_type="nf4",
    bnb_44bit_compute_dtype=torch.float16,
    bnb_4bit_use_double_quant=True,
)

model = AutoModelForCausalLM.from_pretrained(
    MODEL_NAME,
    quantization_config=bnb_config,
    device_map="auto", # <--- Change 4: Use "auto" for Colab's T4 GPU
    token=HF_TOKEN,
    trust_remote_code=True
)

model.config.use_cache = False
model = prepare_model_for_kbit_training(model)

# 2. LORA CONFIG
lora_config = LoraConfig(
    r=32,
    lora_alpha=64,
    target_modules=["q_proj", "k_proj", "v_proj", "o_proj", "gate_proj", "up_proj", "down_proj"],
    lora_dropout=0.05,
    bias="none",
    task_type="CAUSAL_LM"
)
model = get_peft_model(model, lora_config)

# 3. DATA PREPARATION
with open(DATASET_PATH, 'r') as f:
    data = json.load(f)

raw_ds = Dataset.from_list(data)
split_ds = raw_ds.train_test_split(test_size=0.1, seed=42)

def format_instruction(example):
    return {"text": f"<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\nYou are a factual history assistant. Pr

formatted_ds = split_ds.map(format_instruction)

# 4. TRAINING CONFIG (Colab Optimized)
sft_config = SFTConfig(
    output_dir="./history_expert_v2",
    # max_seq_length=512, # Removed from here after previous error
```

```
    dataset_text_field="text",
    num_train_epochs=4,
    per_device_train_batch_size=2,
    gradient_accumulation_steps=8,
    learning_rate=1e-4,
    fp16=True,
    bf16=False,
    optim="paged_adamw_8bit",
    eval_strategy="epoch",
    save_strategy="epoch",
    load_best_model_at_end=True,
    logging_steps=10,
    report_to="none"
)

# 5. TRAIN
trainer = SFTTrainer(
    model=model,
    train_dataset=formatted_ds["train"],
    eval_dataset=formatted_ds["test"],
    args=sft_config
    # tokenizer=tokenizer, # Removed this line after previous error
    # max_seq_length=512 # Removed this line to fix the current error
)

print("🚀 Starting Fine-Tuning in Google Colab...")
trainer.train()

# 6. SAVE
trainer.save_model("./history_expert_final")
print("✅ Done! Best model saved locally to ./history_expert_final")
```

```
Map: 100%                                                2880/2880 [00:00<00:00, 12923.12 examples/s]
Map: 100%                                                320/320 [00:00<00:00, 5581.10 examples/s]
Adding EOS to train dataset: 100%                        2880/2880 [00:00<00:00, 19634.81 examples/s]
Tokenizing train dataset: 100%                           2880/2880 [00:01<00:00, 2208.20 examples/s]
Truncating train dataset: 100%                           2880/2880 [00:00<00:00, 99235.95 examples/s]
Adding EOS to eval dataset: 100%                         320/320 [00:00<00:00, 7814.03 examples/s]
Tokenizing eval dataset: 100%                            320/320 [00:00<00:00, 1970.58 examples/s]
Truncating eval dataset: 100%                            320/320 [00:00<00:00, 19671.08 examples/s]
The tokenizer has new PAD/BOS/EOS tokens that differ from the model config and generation config. The model config and gener
🚀 Starting Fine-Tuning in Google Colab...
```

[720/720 39:28, Epoch 4/4]

Epoch	Training Loss	Validation Loss	Entropy	Num Tokens	Mean Token Accuracy
1	0.091400	0.089988	0.097915	283803.000000	0.963955
2	0.079800	0.083926	0.086748	567606.000000	0.964350
3	0.076800	0.080645	0.083192	851409.000000	0.965259
4	0.079900	0.079833	0.083378	1135212.000000	0.965464

```
✅ Done! Best model saved locally to ./history_expert_final
```

```
from transformers import pipeline

# Load the fine-tuned model
history_pipe = pipeline("text-generation", model="./history_expert_final", torch_dtype=torch.float16, device=0)

def test_model(prompt):
    # Use the same format as your training
    formatted_prompt = f"<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\nYou are a factual history assistant

    output = history_pipe(formatted_prompt, max_new_tokens=150, do_sample=False)
    print(output[0]['generated_text'].split("assistant<|end_header_id|>\n\n")[1])

# Run your tests
print("--- Factual Test ---")
test_model("Who founded the Indian National Congress?")
print("\n--- Domain Test ---")
test_model("Tell me about latest cricket matches.")
```

```
`torch_dtype` is deprecated! Use `dtype` instead!
Device set to use cuda:0
The following generation flags are not valid and may be ignored: ['temperature', 'top_p']. Set `TRANSFORMERS_VERBOSITY=info`
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
```

```
--- Factual Test ---
Setting `pad_token_id` to `eos_token_id`:128001 for open-end generation.
The Indian National Congress was founded in 1885 by A.O. Hume. The first session of the Congress was presided over by W.C. B

--- Domain Test ---
I specialize in history and cannot answer questions outside historical topics.
```

```
import ipywidgets as widgets
from IPython.display import display, clear_output

# 1. Setup the generation function
def generate_history_response(prompt):
    # This must match your training prompt format exactly
    formatted_prompt = (
        f"<|begin_of_text|><|start_header_id|>system<|end_header_id|>\n\n"
        f"You are a factual history assistant. Provide precise, verified historical data.<|eot_id|>"
        f"<|start_header_id|>user<|end_header_id|>\n\n{prompt}<|eot_id|>"
        f"<|start_header_id|>assistant<|end_header_id|>\n\n"
    )
    # Using the 'trainer.model' if you just finished training,
    # or load your saved model via pipeline as shown before.
    inputs = tokenizer(formatted_prompt, return_tensors="pt").to("cuda")
    outputs = model.generate(**inputs, max_new_tokens=150, eos_token_id=tokenizer.eos_token_id)

    # Decode only the new assistant text
    full_text = tokenizer.decode(outputs[0], skip_special_tokens=True)
    return full_text.split("assistant")[-1].strip()

# 2. Create UI Elements
text_input = widgets.Text(
    placeholder='Ask a history question...',
    description='Question:',
    layout=widgets.Layout(width='70%')
)
button = widgets.Button(description="Ask Assistant", button_style='primary')
output_area = widgets.Output(layout={'border': '1px solid black', 'padding': '10px', 'margin': '10px 0'})
def on_button_clicked(b):
    with output_area:
        clear_output()
        question = text_input.value
        if question:
            print(f"User: {question}")
            print("-" * 30)
            response = generate_history_response(question)
            print(f"Assistant: {response}")
        else:
            print("Please enter a question.")

button.on_click(on_button_clicked)

# 3. Display the Interface
display(widgets.VBox([text_input, button, output_area]))
```

Question: Explain the causes and impact of the French Revolution

Ask Assistant

```
Setting `pad_token_id` to `eos_token_id`:128009 for open-end generation.
User: Explain the causes and impact of the French Revolution
------------------------------
Assistant: The French Revolution began in 1789 and challenged monarchy and social hierarchy in France. Events such as
the storming of the Bastille symbolized popular uprising. Revolutionary ideas emphasized liberty, equality, and
fraternity. Economic crisis, social inequality, and political dissatisfaction under the ancien régime fueled unrest.
Enlightenment ideas questioned absolute monarchy. Fiscal problems and food shortages intensified tensions. The
revolution transformed political thought and influenced global movements for democracy. Feudal privileges were abolished
and new political institutions emerged. Its legacy shaped modern concepts of citizenship and rights.
```