Laboratório 23: Fine-Tuning do Llama com LoRA

Objetivo do Laboratório

O objetivo desta atividade é realizar o ajuste fino (*Fine-Tuning*) de um modelo gigantesco (Llama-3-8B) para aprender um dialeto ou comportamento específico (ex: respostas no sotaque Mineirês). Utilizaremos a técnica LoRA por meio da biblioteca **Unsloth** para viabilizar o treinamento em uma GPU de nível básico no Google Colab.

Atividade Prática

1. Configuração do Ambiente (Google Colab)

1 Acesse o Google Colab.

2 Crie um novo Notebook e vá em **Runtime > Change runtime type**. Selecione a aceleração por hardware **T4 GPU**.

3 Instale as dependências do Unsloth (copie e cole no primeiro bloco):

```
!pip install unsloth
!pip install "unsloth[colab-new] @ git+https://github.com/unslothai/unsloth.git"
!pip install --no-deps "xformers<0.0.27" "trl<0.9.0" peft accelerate bitsandbytes
```

2. O Código: Carregamento do Modelo e Preparo

Crie um arquivo JSON com cerca de 20 a 50 exemplos de treinamento, no formato `instrução` e `resposta`. Em seguida, carregue o modelo base e configure os adaptadores LoRA.

```
from unsloth import FastLanguageModel
import torch

max_seq_length = 2048
```

```
dtype = None # Autodetect
load_in_4bit = True # Redução massiva de uso de VRAM

# 1. Carrega o modelo base otimizado
model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = "unsloth/llama-3-8b-bnb-4bit",
    max_seq_length = max_seq_length,
    dtype = dtype,
    load_in_4bit = load_in_4bit,
)

# 2. Configura os adaptadores LoRA
model = FastLanguageModel.get_peft_model(
    model,
    r = 16, # Tamanho do adaptador (Rank)
    target_modules = ["q_proj", "k_proj", "v_proj", "o_proj",
                      "gate_proj", "up_proj", "down_proj",],
    lora_alpha = 16,
    lora_dropout = 0,
    bias = "none",
    use_gradient_checkpointing = "unsloth",
    random_state = 3407,
)
print("LoRA configurado com sucesso!")
```

3. O Treinamento (Trainer)

Carregue os seus dados e inicie o treinamento do modelo.

```
from trl import SFTTrainer
from transformers import TrainingArguments
from datasets import load_dataset

# Dataset fictício para exemplo:
# dataset = load_dataset("json", data_files="meus_dados_mineiros.json", split="train")

trainer = SFTTrainer(
    model = model,
    tokenizer = tokenizer,
    train_dataset = dataset, # Seu dataset formatado
    dataset_text_field = "text",
    max_seq_length = max_seq_length,
    args = TrainingArguments(
        per_device_train_batch_size = 2,
        gradient_accumulation_steps = 4,
        warmup_steps = 5,
        max_steps = 60, # Número curto para aula
```

```
        learning_rate = 2e-4,
        fp16 = not torch.cuda.is_bf16_supported(),
        bf16 = torch.cuda.is_bf16_supported(),
        logging_steps = 1,
        output_dir = "outputs",
    ),
)

trainer_stats = trainer.train()
```

4. Entrega via Git e Colab

1 Exporte o Notebook do Colab (`.ipynb`) que contém a prova da execução (logs de perda/loss diminuindo).

2 Adicione o arquivo no seu repositório Git: `git add fine_tuning_lora.ipynb`.

3 Crie o commit: `git commit -m "Laboratório de Fine-Tuning LoRA Llama-3"`.

4 Faça o push para o seu branch principal.

Critério de Avaliação

A aprovação ocorre pela submissão do Notebook comprovando a queda na curva de Loss durante as épocas do Hugging Face Trainer e um exemplo de inferência gerada ao final do notebook demonstrando a aquisição da nova habilidade pelo modelo.