Aim

To fine-tune the Llama-2-7b model using the QLoRA method and upload the fine-tuned model to Hugging Face Hub.

Objectives

1. Mount Google Drive and install necessary libraries.
2. Configure QLoRA parameters for fine-tuning.
3. Load and preprocess the dataset.
4. Train the Llama-2-7b model with the configured parameters.
5. Evaluate and save the fine-tuned model.
6. Push the fine-tuned model to Hugging Face Hub.

Summary

This project demonstrates the process of fine-tuning the Llama-2-7b model using the QLoRA method. It involves setting up the environment, configuring model and training parameters, loading the dataset, and performing the fine-tuning. The fine-tuned model is then evaluated, saved, and uploaded to Hugging Face Hub.

Tools and Libraries Used

- **Google Colab**: Cloud-based Jupyter notebook environment.
- **Hugging Face Transformers**: Library for working with transformer models.
- **BitsAndBytes**: Library for efficient 4-bit quantization.
- **TRL (Transformer Reinforcement Learning)**: Library for fine-tuning transformer models.
- **PyArrow, Requests, Peft**: Additional libraries for data processing and fine-tuning.

Procedure

1. **Setup and Installation**:
    - Mount Google Drive to access and save data.
    - Install required libraries such as pyarrow, requests, transformers, trl, bitsandbytes, and peft.

CODE:

```
from google.colab import drive

drive.mount('/content/drive')

!pip install pyarrow==14.0.1 requests==2.31.0
```

!pip install accelerate==0.21.0 peft==0.4.0 bitsandbytes==0.40.2 transformers==4.31.0 trl==0.4.7

2. **Configuration**:
    - Define QLoRA, bitsandbytes, and training parameters.
    - Load the dataset and preprocess it.
    - Load the base Llama-2-7b model and tokenizer with specific configurations.

CODE:

```
model_name = "NousResearch/Llama-2-7b-chat-hf"

dataset_name = "mlabonne/guanaco-llama2-1k"

new_model = "Llama-2-7b-chat-finetune"


lora_r = 64

lora_alpha = 16

lora_dropout = 0.1

use_4bit = True

bnb_4bit_compute_dtype = "float16"

bnb_4bit_quant_type = "nf4"

use_nested_quant = False

output_dir = "./results"

num_train_epochs = 1

per_device_train_batch_size = 4

per_device_eval_batch_size = 4

gradient_accumulation_steps = 1

gradient_checkpointing = True

max_grad_norm = 0.3

learning_rate = 2e-4

weight_decay = 0.001

optim = "paged_adamw_32bit"

lr_scheduler_type = "cosine"

max_steps = -1
```

```
warmup_ratio = 0.03

group_by_length = True

save_steps = 0

logging_steps = 25


dataset = load_dataset(dataset_name, split="train")
```

3. **Training**:
    - Create the training arguments and fine-tune the model using the SFTTrainer.
    - Save the trained model.

CODE:

```
training_arguments = TrainingArguments(
    output_dir=output_dir,
    num_train_epochs=num_train_epochs,
    per_device_train_batch_size=per_device_train_batch_size,
    gradient_accumulation_steps=gradient_accumulation_steps,
    optim=optim,
    save_steps=save_steps,
    logging_steps=logging_steps,
    learning_rate=learning_rate,
    weight_decay=weight_decay,
    fp16=fp16,
    bf16=bf16,
    max_grad_norm=max_grad_norm,
    max_steps=max_steps,
    warmup_ratio=warmup_ratio,
    group_by_length=group_by_length,
    lr_scheduler_type=lr_scheduler_type,
    report_to="tensorboard"
)
```

```
trainer = SFTTrainer(
    model=model,
    train_dataset=dataset,
    peft_config=peft_config,
    dataset_text_field="text",
    max_seq_length=max_seq_length,
    tokenizer=tokenizer,
    args=training_arguments,
    packing=packing,
)

trainer.train()
trainer.model.save_pretrained(new_model)
```

4. **Evaluation and Upload**:
    - Run a text generation pipeline to evaluate the fine-tuned model.
    - Reload the model in FP16, merge with LoRA weights, and upload to Hugging Face Hub.

CODE:

```
prompt = "What is a large language model?"
pipe = pipeline(task="text-generation", model=model, tokenizer=tokenizer, max_length=200)
result = pipe(f"<s>[INST] {prompt} [/INST]")
print(result[0]['generated_text'])

!huggingface-cli login
model.push_to_hub("entbappy/Llama-2-7b-chat-finetune", check_pr=True)
tokenizer.push_to_hub("entbappy/Llama-2-7b-chat-finetune",check_pr=True)
```

Highlights

- **Quantization with QLoRA**: Efficient 4-bit precision quantization for model loading.
- **LoRA Configuration**: Specific parameters to enhance the model's attention mechanism.

- **Training Optimization**: Use of gradient checkpointing, gradient accumulation, and cosine learning rate scheduling.
- **Model Evaluation**: Generative text evaluation pipeline to verify the fine-tuned model's performance.
- **Seamless Integration**: Uploading the fine-tuned model to Hugging Face Hub for easy access and sharing.

Conclusion

This project successfully demonstrates the process of fine-tuning a large language model, Llama-2-7b, using the QLoRA method. The use of efficient quantization and fine-tuning techniques results in a robust model that performs well on the specified dataset. The final model is saved and uploaded to Hugging Face Hub, making it accessible for further use and experimentation.