Homework1: SFT Practice

Goal

Conduct an end-to-end SFT experiment with Qwen2.5-0.5B, familiarize with the "data-training-evaluation" loop.

Required Outputs

Homework1_Upload_url

Please package the following required files into a .zip (or .rar) file and upload it to Google Drive. File naming format: `StudentID_Name_AIMS5740_Homework1.zip`

(Please ensure your uploaded file is less than 10MB)

1. A complete training script (.ipynb or .py), including:
    a. SFT training script + logs (with key hyperparameters)
    b. Evaluation score (base vs SFT model comparison)
2. A brief report (2-4 pages): data cleaning strategy, training tuning, evaluation results, conclusions & analysis

Model & Dataset

- Model: Qwen/Qwen2.5-0.5B
- Dataset: Open-Orca/OpenOrca

The original dataset is too large, so we used a deterministic method to select the 1M subset as the initial dataset. Please perform your data filtering based on this subset. **Note: For a 0.5B model, the ideal SFT dataset size is 200k to 350k entries.**

```
# Load a deterministic subset of Open-Orca/OpenOrca (for 0.5B SFT)
# Pin a specific commit for full reproducibility (shown on HF commit history page)
REVISION = "e9c87b4"  # 2025-02-19

from datasets import load_dataset, DatasetDict

SUBSET_ROWS = 1_000_000
train_ds = load_dataset(
    "Open-Orca/OpenOrca",
```

```
    split=f"train[:{SUBSET_ROWS}]",
    revision=REVISION,
)

dataset = DatasetDict({"train": train_ds})
```

Data Cleaning

e.g. formatting, quality, consistency; explore more cleaning methods.

SFT Training

- Framework: llama-factory
- Method: full finetuning
- Tunable hyperparameters: learning rate, batch size, epochs, warmup steps, weight decay, max sequence length, optimizer, etc.

Evaluation Requirements

1. Verify two points: model learns data distribution (loss/perplexity); output behavior improves (manual alignment + automatic metrics).
2. Evaluate 8 datasets (use standard libraries): `mmlu, arc_easy, arc_challenge, hellaswag, winogrande, truthfulqa_mc2, piqa, boolq.`
3. Key evaluation settings (all apply_chat_template=True):
   (For reproducibility, please ensure these key parameters for generation/evaluation are aligned.)
   - mmlu: 5-shot
   - arc_easy: 0-shot
   - arc_challenge: 25-shot
   - hellaswag: 10-shot
   - winogrande: 5-shot
   - truthfulqa_mc2: 0-shot
   - piqa: 0-shot
   - boolq: 0-shot

4. Note that `llamafactory-cli eval` feature will be deprecated in later versions of llamafactory. You can try it, but it's not recommended.

Evaluation Examples

In our example, we provide two evaluation examples:
`lm_eval for mmlu_test`
`lm_eval for ARC-E`

Setup and preparation

```
!pip -q install -U "lm_eval[hf]" tiktoken sentencepiece
```

```
import os, json, time
import torch

# TODO: Change it to the output directory after your SFT training is complete.
SFT_MODEL_DIR = "/content/your_sft_model_dir"
OUT_DIR = "/content/lm_eval_results/your_output_subdir"
os.makedirs(OUT_DIR, exist_ok=True)

MODEL_ARGS = f"pretrained={SFT_MODEL_DIR},trust_remote_code=True,dtype=float16"
DEVICE = "cuda:0"
BATCH_SIZE = 4
```

Example for MMLU (5-shot for default)

```
!python -m lm_eval \
  --model hf \
  --model_args "{MODEL_ARGS}" \
  --tasks mmlu \
  --num_fewshot 5 \
  --device {DEVICE} \
  --batch_size {BATCH_SIZE} \
  --apply_chat_template \
  --output_path "{OUT_DIR}/mmlu_5shot.json"
```