#### lab4

## 环境配置

- 1. 在阿里云平台购买A10加速 GPU, 硬盘设置60G, 选择对应的N卡驱动, 弹性公网 (可以没有)
- 2. pycharm链接远程解释器
- 3. 在服务器安装git, conda, 配置对应环境, 安装相关包
- 4. 按照swift官网, clone项目, 并执行安装命令

## 模型选择

qwen1.5-1.8b模型,在A10加速GPU上训练,应用实验文档中的参数微调2小时左右

# 数据集准备

采用单轮对话数据, 做二分类任务

由于框架本身支持HC3-zh的数据集,可以不用特意修改swift/llm/data文件夹下的dataset\_json文件,划分采用命令行自动划分,设置数据集5000.

### 微调

采用LORA微调 具体参数设置如下,基于swift的

```
--train dataset sample 5000 \
--num train epochs 2 \
--max length 512 \
--check dataset strategy warning \
--lora rank 8 \
--lora alpha 32 \
--lora dropout p 0.05 \
--lora target modules ALL \
--gradient checkpointing true \
--batch size 1 \
--weight decay 0.1 \
--learning rate 1e-4 \
--gradient accumulation steps 16 \
--max grad norm 0.5 \
--warmup ratio 0.03 \
--eval steps 100 \
--save steps 100 \
--save total limit 2 \
--logging steps 10 \
--use flash attn false \
--self cognition sample 1000 \
--model name Lab4 \
--model author 宋业鑫 \
```

#### 解释:

后端调优方法: peft

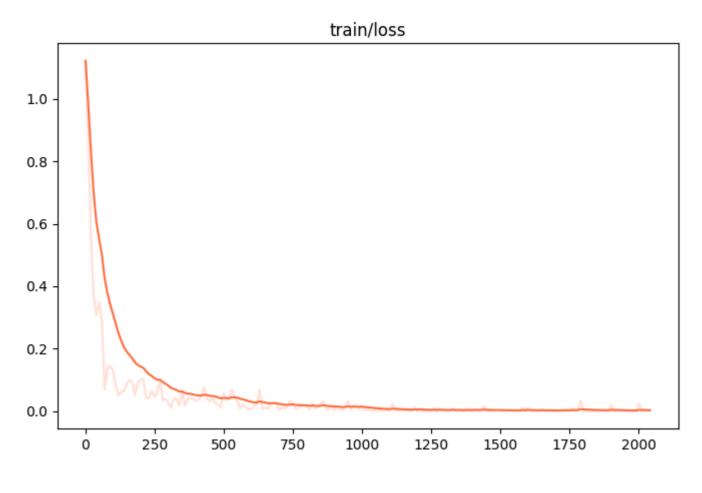
rank, alpha, max\_len参考的是实验文档的参数,通过开启检查点技术可以减少内存消耗。且只保存两个模型,即最优结果,和最后一次训练结果

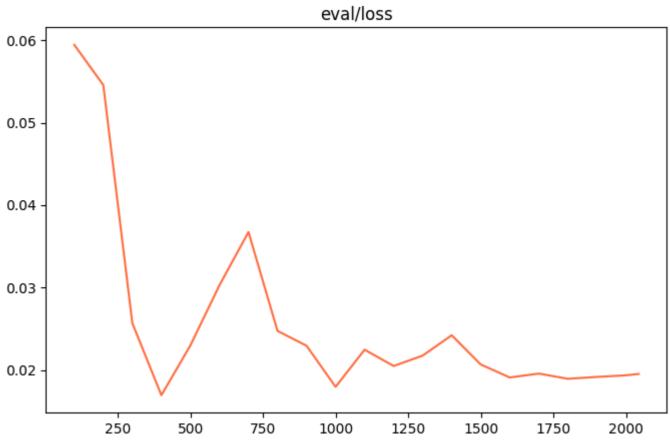
#### 训练过程截图

```
{'loss': 0.00280725, 'acc': 0.9989727, 'grad_norm': 0.00125885, 'learning_rate': 5.621e-05, 'memory(GiB)': 5.17, 'train_speed(iter/s)': 0.291725, 'epoch': 0.91, 'global_step': 930}
{'loss': 0.01831999, 'acc': 0.99457836, 'grad_norm': 0.01538086, 'learning_rate': 5.57e-05, 'memory(GiB)': 4.98, 'train_speed(iter/s)': 0.291739, 'epoch': 0.92, 'global_step': 940}
{'loss': 0.03225512, 'acc': 0.9953658, 'grad_norm': 0.00674438, 'learning_rate': 5.52e-05, 'memory(GiB)': 5.04, 'train_speed(iter/s)': 0.291779, 'epoch': 0.93, 'global_step': 950}
{'loss': 0.0043891, 'acc': 0.99855022, 'grad_norm': 0.05249023, 'learning_rate': 5.469e-05, 'memory(GiB)': 5.38, 'train_speed(iter/s)': 0.291769, 'epoch': 0.94, 'global_step': 960}
{'loss': 0.01931686, 'acc': 0.99497623, 'grad_norm': 2.390625, 'learning_rate': 5.419e-05, 'memory(GiB)': 5.24, 'train_speed(iter/s)': 0.29184, 'epoch': 0.95, 'global_step': 970}
{'loss': 0.01263758, 'acc': 0.99575024, 'grad_norm': 0.02783203, 'learning_rate': 5.368e-05, 'memory(GiB)': 4.98, 'train_speed(iter/s)': 0.29184, 'grad_norm': 0.00866699, 'learning_rate': 5.318e-05, 'memory(GiB)': 5.11, 'train_speed(iter/s)': 0.29189, 'epoch': 0.97, 'global_step': 970}
B. *(epoch': 0.97, 'global_step': 970}
Train: 49%1
```

swift 自动保存两个checkpoint,一个是最后的,一个是最佳的,下面测试使用最佳点,即400

### 微调结果





应用微调成果-bestmodel CUDA\_VISIBLE\_DEVICES=0 swift export

#### 评测

1. 使用vllm加速

```
# 原始模型
    CUDA VISIBLE DEVCIES=0 swift eval --model type qwen1half-1 8b-chat \
        --eval dataset no \
        --infer backend vllm \
        --custom eval config ../data/eval config.json
    # LoRA微调后
    CUDA VISIBLE DEVICES=0 swift eval --ckpt dir output/qwen1half-1 8b-chat/v0-2
        --eval dataset no --infer backend vllm \
        --merge lora true \
        --custom eval config ../data/eval config.json
2. 自定义评测集的格式, prompt
 csv的格式为id,question,A,B,C,D,answer,explanation
 question = f"针对一个计算机知识的提问所给出的回答: {answer}这句话是由 回答的。"
 A为human
 B为chatgpt
3. 修改对应配置文件
        {
            "name": "ceval",
            "pattern": "ceval",
            "dataset": "../data/ceval",
            "subset list": ["default"]
        }
    1
```

### 结果

原始模型

#### 正确率0.38

#### 微调后

#### 正确率0.44