微调 deepseek-R1 模型应用实例-小红书文案推理数据集-变身文案大师

MindsRiverPonder
中国科学技术大学 电子信息硕士在读
关注他

1. 前言

单纯和deepseek聊天，那很乏味了，让deepseek做某个垂直领域的专家，那可太有意思了。本期教程是在小红书推理思考数据集上微调DeepSeek-R1-Distill-Qwen-14B，让deepseek变成小红书文案大师，帮我们自动生成文案。当然，本教程仅仅作为一个引例，大家大可尝试不同的数据集，采用不同大小的蒸馏模型（如果你有几万张英伟达的GPU，建议你尝试更大的模型，站得高，看得远嘛doge）。此外，本篇文章的配套B站视频如下所示，大家可以一起讨论，如有错误，恳请批评指正。

还有一件事情，**下载数据集**到当前目录

```
from datasets import load_dataset
import json

# 加载数据集，使用 streaming=True 避免一次性下载全部数据
dataset = load_dataset("Congliu/Chinese-DeepSeek-R1-Distill-data-110k-SFT", strea

# 过滤数据，只保留 repo_name 为 'xhs/xhs' 的数据
filtered_dataset = dataset.filter(lambda example: example['repo_name'] == 'xhs/xh
#filtered_dataset = dataset['train'].filter(lambda example: example['repo_name'] =


# 将过滤后的数据保存为 JSON 文件
def save_to_json(dataset, filename):
    with open(filename, 'w', encoding='utf-8') as f:
        for example in dataset:
            #filtered_dataset是一个可迭代对象，需要逐条example提取
            json.dump(example, f, ensure_ascii=False)
            f.write('\n')  # 每条数据之间添加换行符


save_to_json(filtered_dataset['train'], 'xhs_data.json')
```

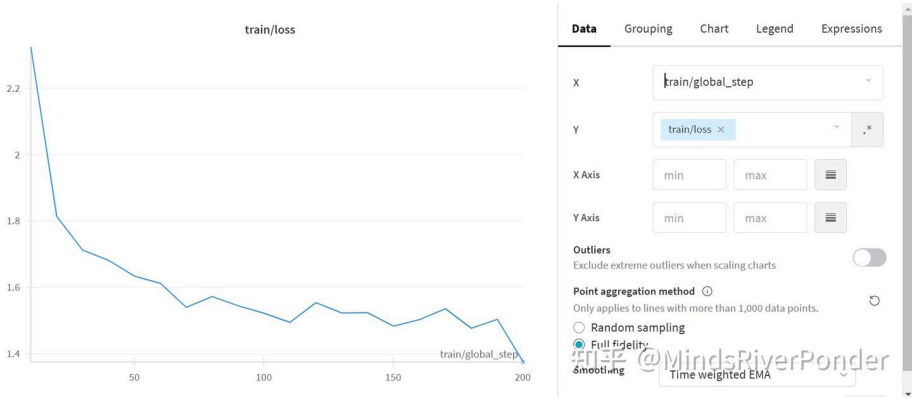
2. 手把手微调代码讲解

2.1 第一步，安装必要的库

其中unsloth这个框架可以使得微调任务更快，更省内存，wandb用来跟踪并可视化微调过程，比如train_loss、GPU利用率

```
!pip install unsloth   # 高效微调框架，优化内存和训练速度
!pip install datasets
!pip install -U huggingface_hub
!pip install wandb    # 实验跟踪工具，如追踪train_loss，并可视化
```

下图是wandb追踪train_loss的图片（我在4090上花了30分钟跑了200步）



wandb-train_loss

2.2 第二步，配置hugging face的Access Tokens

配置这玩意，待会要从hugging face下载模型，要申请token，可以到官网免费申请

```
# 登录HuggingFace Hub，待会要下载模型
from huggingface_hub import login
hf_token = "XXX"  #到官网免费申请一个
login(hf_token)
```

2.3 第三步，初始化wandb

配置这玩意，跟踪训练过程并可视化，又要申请token，可以到官网免费申请

```
import wandb

wb_token = "XXX"  #到官网免费申请一个
wandb.login(key=wb_token)
run = wandb.init(
    project='fine-tune-DeepSeek-R1-Distill-Qwen-14B on xhs Dataset',
    job_type="training",
    anonymous="allow"
)
```

2.4 第四步，加载模型和对应的分词器

```
from unsloth import FastLanguageModel

max_seq_length = 2048
dtype = None
load_in_4bit = True
model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = "unsloth/DeepSeek-R1-Distill-Qwen-14B",
    max_seq_length = max_seq_length,
    dtype = dtype,
    load_in_4bit = load_in_4bit,  # 4bit量化加载以节省显存
    token = hf_token,
)
```

2.5 第五步，定义训练时prompt模板

本例中，我的任务是让deepseek变成小红书文案专家。当然，你如果有不同的数据集，做不同的任务，这里需要更改。

```
train_prompt_style = """以下是一项任务说明，并附带了更详细的背景信息。
请撰写一个满足完成请求的回复。
在回答之前，请仔细考虑问题，并创建一个逐步的思考链，以确保逻辑和准确的回答。

### Instruction:
你是一个资深的小红书文案专家
请你根据以下问题完成写作
### Question:
{}
### Response:
<think>
{}
</think>
{}"""
EOS_TOKEN = tokenizer.eos_token
```

2.6 第六步，对数据进行处理与标准化（重点）

这一部分跟你的数据集息息相关，不同的数据集可能数据结构不一样，数据处理的函数需要你自己编写。

我这个小红书推理数据集+是一个**已经下载到当前目录的json文件**，数据结构如下所示：


```
import re  # 导入正则表达式模块
# 使用正则提取思考链和最终响应
def formatting_prompts_func(examples):
    instructions = examples["instruction"]
```

```
        # 使用正则表达式提取 <think> 和 </think> 之间的内容作为 cots
        match = re.search(r"<think>(.*?)</think>", output_text, re.DOTALL)
        cot = match.group(1).strip() if match else ""

        # 提取 </think> 之后的内容作为 outputs
        output = output_text.replace(match.group(0), "").strip() if match else ou

        text = train_prompt_style.format(instruction_text, cot, output) + EOS_TOKI
        texts.append(text)
    return {"text": texts}
```

2.7 第七步，加载数据集，并使用上面的函数处理

```
# 加载数据集
from datasets import load_dataset

dataset = load_dataset("json", data_files="xhs_data.json", split="train")

# 使用 map 函数进行数据处理
dataset = dataset.map(formatting_prompts_func, batched=True)
```

2.8 第八步，设置 LoRA⁺ 进行微调

其中r是lora_rank，值越大，更新的参数量越多，训练越慢，效果更好。lora_alpha用来控制参数值的更新幅度

```
model = FastLanguageModel.get_peft_model(
    model,
    r=32,  # LoRA秩（矩阵分解维度）
    target_modules=[
        "q_proj",
        "k_proj",
        "v_proj",
        "o_proj",
        "gate_proj",
        "up_proj",
        "down_proj",
    ],
    lora_alpha=32,  # 缩放因子
    lora_dropout=0,
    bias="none",
    use_gradient_checkpointing="unsloth",
    random_state=6666,
    use_rslora=False,
    loftq_config=None,
)
```

2.9 第九步，配置训练器，准备训练

能调的一些参数我都注释出来了

```
from trl import SFTTrainer
from transformers import TrainingArguments
from unsloth import is_bfloat16_supported

trainer = SFTTrainer(
```

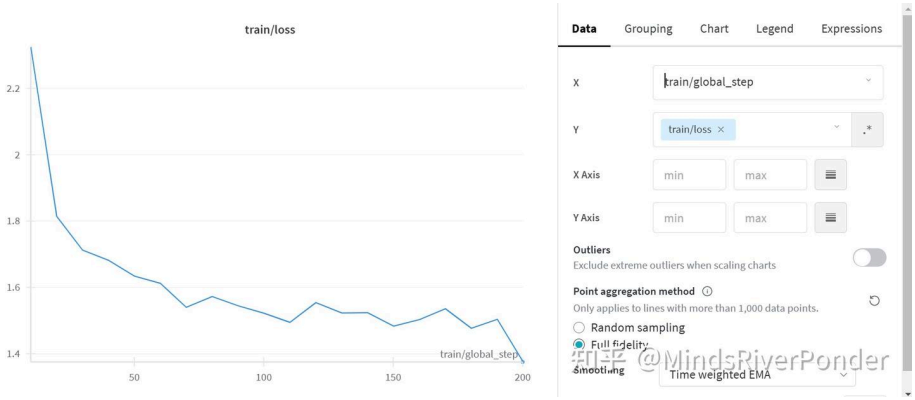
```
    dataset_text_field="text",
    max_seq_length=max_seq_length,        #输入序列的最大长度
    dataset_num_proc=2,                  #预处理数据集的进程数
    args=TrainingArguments(
        per_device_train_batch_size=2,
        gradient_accumulation_steps=4,  #梯度累积步数（变相模拟更大batch_size）
        warmup_steps=5,          #学习率预热步数。
        max_steps=200,          # 总训练步数
        learning_rate=2e-4,     #学习率
        fp16=not is_bfloat16_supported(),
        bf16=is_bfloat16_supported(),
        logging_steps=10,
        optim="adamw_8bit",
        weight_decay=0.01,           # 权重衰减系数
        lr_scheduler_type="linear", #学习率调度器
        seed=6666,                  #选一个自己喜欢的随机数种子
        output_dir="outputs",
        report_to="wandb",        # 使用 wandb 进行报告，实验指标可视化
    ),
)
```

2.10第十步，训练

```
trainer_stats = trainer.train()
```

3. 结果及效果演示

使用4090训练了200步，train_loss曲线如下所示，我们可以看到，损失一直在降低，仍有下降空间，继续训练效果会更好一些。



光有这个train_loss不过瘾啊，我们看看模型微调后的回答。

我们先设置一个用于提问的prompt。

```
prompt_style = """以下是一项任务说明，并附带了更详细的背景信息。
请撰写一个满足完成请求的回复。
在回答之前，请仔细考虑问题，并创建一个逐步的思考链，以确保逻辑和准确的回答。

### Instruction:
你是一个资深的小红书文案专家
请你根据以下问题完成写作
### Question:
{}
```

接下来，调用我们微调后的模型，问问题（提要求）。

我提的要求是：“写一篇小红书风格的帖子，标题是男生变帅只需三步丨分享逆袭大干货”

```
question = "写一篇小红书风格的帖子，标题是男生变帅只需三步丨分享逆袭大干货  "


FastLanguageModel.for_inference(model)
inputs = tokenizer([prompt_style.format(question, "")], return_tensors="pt").to("
outputs = model.generate(
    input_ids=inputs.input_ids,
    attention_mask=inputs.attention_mask,
    max_new_tokens=1200,
    use_cache=True,
)
response = tokenizer.batch_decode(outputs)
print(response[0].split("### Response:")[1])
```

我们看看模型的思考过程，这考虑的是面面俱到啊

<think>
嗯，用户让我写一篇小红书风格的帖子，标题是“男生变帅只需三步丨分享逆袭大干货”。首先，我需要理解用户的需求。他们可能希望这篇帖子能吸引男性读者，帮助他们提升形象，变得更有魅力。小红书的用户群体主要是年轻女性，但男性变帅的指南可能也会吸引她们，或者她们的男友、兄弟等男性朋友。所以，内容需要既实用又带有鼓励和亲切感。

接下来，我要考虑小红书的风格。通常这类帖子会用活泼的语气，加上表情符号和分段清晰的结构，可能还会用一些热门标签。标题里的可能代表快速、逆袭，所以内容需要突出简单易行的三个步骤。

用户提到的三个步骤是关键。我需要确定这三个步骤是什么。常见的变帅方法包括护肤、发型、穿搭、体态调整等。可能需要涵盖这些方面，但用户希望是三步，所以可能需要合并或简化。比如，第一步可能是护肤，第二步是发型，第三步是穿搭，或者体态和表情管理等。

然后，每个步骤需要详细但不过于复杂，比如护肤部分，可以强调清洁、保湿和防晒，但要简单说明，避免专业术语太多。发型部分可能建议剪清爽发型，或者避免油腻的发型，可能提到打理方法。穿搭部分可能强调颜色搭配、整洁感，避免过于花哨。

还需要加入一些鼓励的话语，比如“逆袭不是魔法，是每天坚持的魔法”，这样能激励读者。可能还需要一些个人经历或见证，比如“亲测有效”，增加可信度。

另外，要注意使用小红书常用的标签，比如#男生变帅 #逆袭攻略 #男生穿搭 等，提高帖子的可见性。结尾部分可以引导互动，比如让读者在评论区分享自己的经验或提问。

可能还需要考虑用户的深层需求。他们可能希望内容不仅实用，还能让读者感到被理解和激励，避免负面情绪。所以语气要积极，避免批评，而是强调改变的可能性和好处。

最后，检查结构是否清晰，每个部分是否有足够的细节，同时保持轻松易读。可能需要添加一些表情符号和分段符号，[illegible]理解。
</think>

再来看一下最后回答，（图片没截完，下面还有一大堆），我们可以看到，这满满的小红书味道啊！

🌟【男生变帅只需三步丨分享逆袭大干货】🌟

姐妹们！今天这篇是给想让“他”变帅的你们看的！也是给想自救的男同胞的逆袭攻略！
亲测有效，简单粗暴，三步到位，直接抄作业👇

🔥【Step 1：拒绝“油头油腻男”！】
发型决定第一印象！
❌别留刘海！别留杂乱刘海！
✅清爽短发/背头/韩式V字刘海（避开圆头！）
👂打理技巧：
▪发蜡薄涂发根+喷定型喷雾（油头选干爽型！）
▪理发店挑“剪发师”而不是“理发师”（别信理发师的“流行款”）
▪拒绝碎发！剪发时强调“干净利落”

🔥【Step 2：拒绝“垮脸男”！】
体态+表情=气场开挂！
❌别低头！别驼背！别表情木头！
✅下巴微抬30°（显脸小！显牙齿白！）
✅双肩下沉（别耸肩！）
✅走路带风（但别像机器人！）
👂隐藏技能：
▪握手时掌心对掌心（巨显真诚！）
▪微笑时露出8颗牙齿（牙齿黄？看【Step3】）

没啥总结的，大家有啥有意思的数据集不？doge。

送礼物

还没有人送礼物，鼓励一下作者吧

编辑于 2025-02-20 11:36 · 广东

DeepSeek-R1 大模型微调 大模型微调

理性发言，友善互动

16 条评论 默认 最新

amor-romantico
两百步的时候train loss降到0.002左右正常吗？试了一下推理感觉不如题主训的效果，是不是过拟合了？(参数里只改了lora alpha=64)
04-13 · 北京 回复 喜欢

昨日的世界
数据集必须要有思考过程吗
03-27 · 江苏 回复 喜欢

昨日的世界 ‣ **卡牌大师**
那意思是基座模型选非推理模型，qwen2.5可以吗
03-31 · 江苏 回复 喜欢

卡牌大师 ‣ **昨日的世界**
没有思考过程，你就会破坏蒸馏模型的思维模式，不如微调普通的模型。
03-30 · 四川 回复 喜欢

展开其他 3 条回复 〉

小陈
您好，我看到你的两个项目。请问 qwen-3B + GPRO 和 DeepSeek-R1-Distill-Qwen-14B + sft 二者有没有什么关系，我想用DeepSeek-R1-Distill-Qwen-14B + GPRO，可以吗?
03-17 · 北京 回复 喜欢

淡定
博主请问一下您的python版本是什么，我这边会报错
03-12 · 湖北 回复 喜欢

Nous
如何让模型数据更加贴近数据集呢，我训练成功后用数据集的问题提问，回答不太准确
03-03 · 陕西 回复 喜欢

憨憨坑
牛逼
02-20 · 上海 回复 喜欢

MindsRiverPonder 作者
谢谢
02-20 · 广东

02-20 · 湖南 回复 喜欢

Helper ‣ **MindsRiverPonder**
感谢！！！！
02-20 · 湖南 回复 喜欢

MindsRiverPonder 作者
嗯嗯
new_model_local = "路径" model.save_pretrained(new_model_local)
tokenizer.save_pretrained(new_model_local)
model.save_pretrained_merged(new_model_local, tokenizer, save_method="merged_16bit")
02-20 · 广东 回复 喜欢

令人暖心的赵火龙
大佬，为什么 SFT 的时候 prompt 格式是这样的，最后变成了 text 字段，question answer 和 think 全都塞进去了呢？不能用{role:user，content:....}{role:assisntant, content: xxxx}的格式吗？ 另外为啥你的结果里面开头是<think>。。我直接跑 qwen 蒸馏的 r1 没有开头的<think>但是 apply_chat_template 后最后一个 token 是 <think> 🤣
02-20 · 四川 回复 喜欢

理性发言，友善互动

推荐阅读

如何用DeepSeek做运营？给你10个爆款模板
挖塘人 发表于挖塘人

有赞埋点设计方案学习
辰菇凉 发表于从零开始数...

Python数据分析案例--运用K-Means聚类分析广告效果
爱迪生 发表于Pytho...

Flume在有赞大数据的实践
有赞技术团队