Unsloth 框架 Lora 微调大模型

使用说明书

一、 引言

主要参考哔哩哔哩的教学视频：

【deepseek 基于 unsloth 本地微调整合包(windows)】

https://www.bilibili.com/video/BV14YNReqEHu/?share_source=copy_web&vd_source=264be3ecbe28dcdaaa300bab109ee0f4

代码参考：

笔记链接：

https://note.youdao.com/s/D8vlouNI

https://www.aivi.fyi/llms/Fine-Tuning-Qwen

网盘链接：

https://pan.baidu.com/s/1tLJdV5k9zwyKVxG4FMG2Wg?pwd=kdwd

二、 安装说明

下载 pycharm 编译器，anaconda，CUDA12.7 版本，提前安装好 Visual Studio（尤其是桌面 C++应用，适配 CUDA 的安装。）和 Cmake（后续用于 Hugging face 模型格式转换到 GGUF 量化格式，方便小白一键部署）

下载 LM-Studio（强烈推荐）或者 Ollama（一键部署工具，只支持 GGUF 格式，HF 格式模型需要自己编程处理）

黄色部分为采用 Ollama 部署的一些优化点可参考

【新一版丨大模型本地部署到 d 盘 丨 ollama、deepseek】

https://www.bilibili.com/video/BV1QtNmeoEe3/?share_source=copy_web&vd_source=264be3ecbe28dcdaaa300bab109ee0f4

Ollama 使用 GPU，cmd 命令行输入 nvidia-smi -L，查看自己 GPU 的 ID，在系统环境里，输入：

变量名：CUDA-VISIBLE-DEVICES

变量值：查到的 GPU IP（例子：GPU-9b70d300-0e20-725c-f801-6addbf07b599）

变量名：OLLAMA_GPU_LAYER

变量值：cuda

然后下载模型：

Deepseek-R1:7b

Qwen3:4b

Gemma3:4b

Qwen3:4b

Qwen2.5vl:3b

拉取以及运行命令：

ollama run qwen2.5vl:7b（其他模型，参考官网命令 https://ollama.com）

LM-Studio 比较简单，下载完对应安装包后，运行，在发现里可以拉取对应的模型：

Deepseek-R1:7b

Qwen3:4b

Gemma3:4b

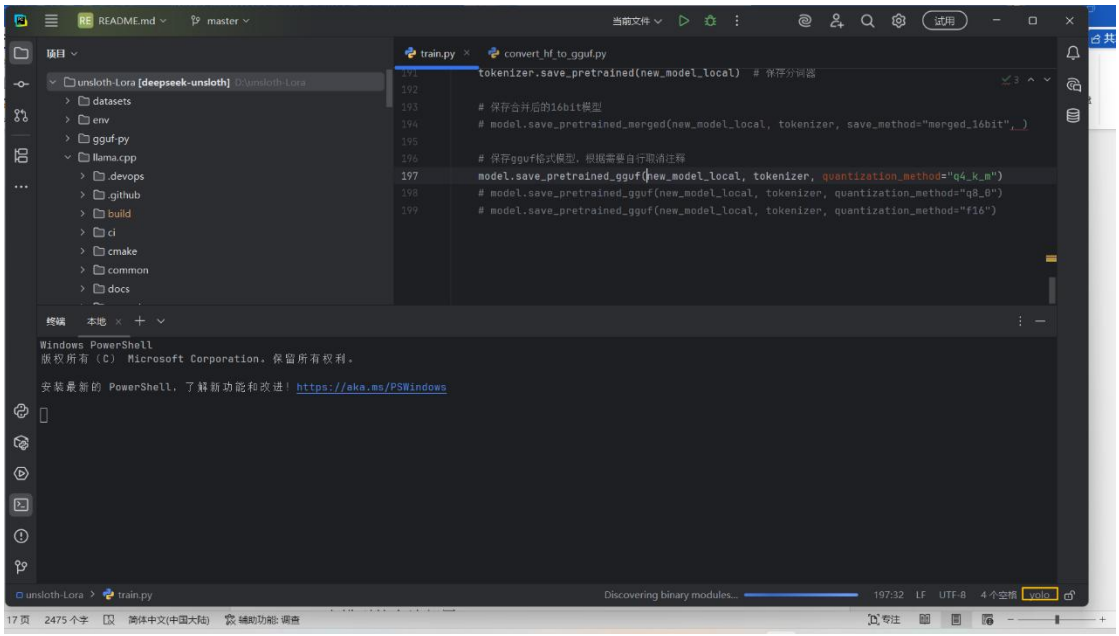
Qwen3:4b

Qwen2.5vl:3b

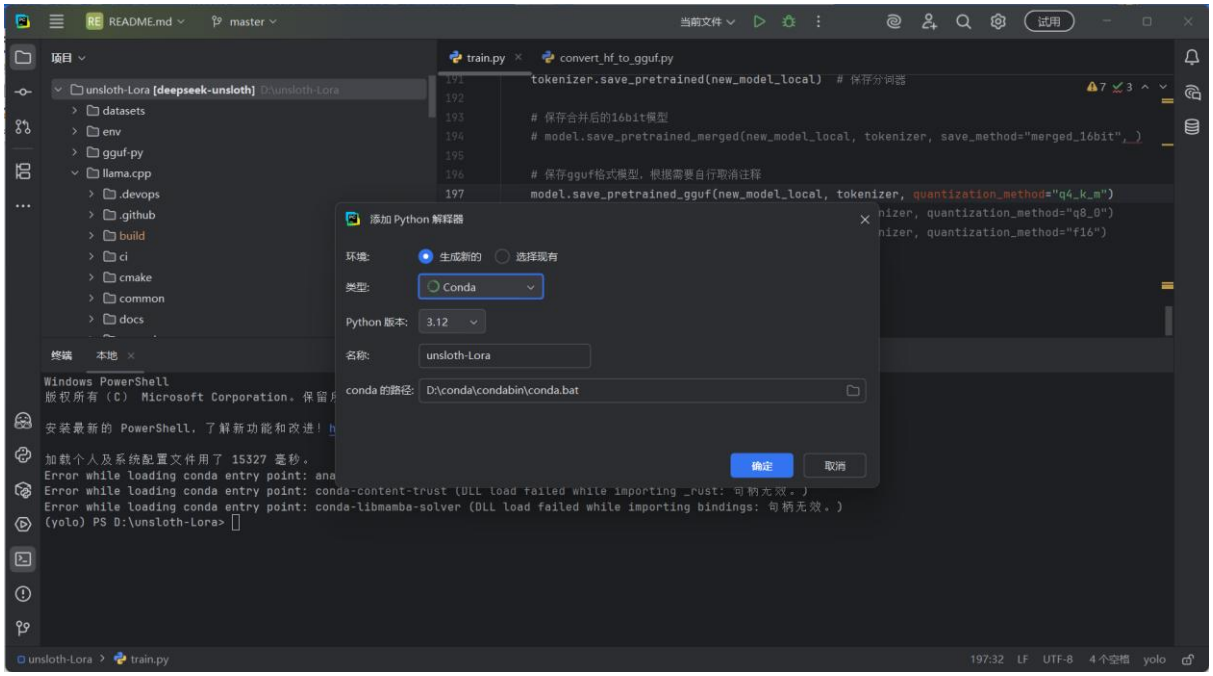
并且在 runtime 中选择配置运行环境的 CUDA 12，让模型能够调用 GPU 进行推理，然后在开发者界面中 Status:Running 打开

按钮，大模型的 API 接口即可对本机器公布调用。在 setting 中打开即时模型加载（方便你项目里自由切换模型）即可，可提前预热，加载模型，在上方下拉框选择要加载的模型里，加载你希望预热的模型即可

下载好 pycharm，如图所示配置相应的 anaconda 解释器

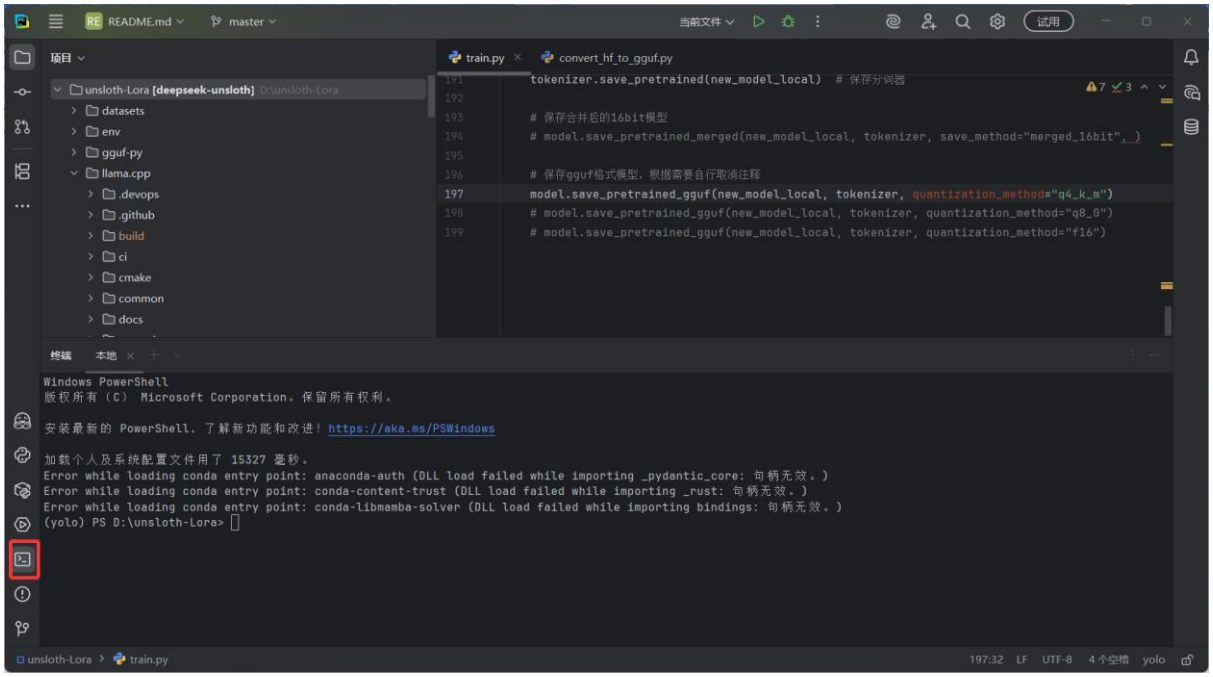


找到自己安装的 anaconda 文件夹，在 anaconda 目录中找到 bin 文件，找到 conda.bat，生成新的即可。如下图，名字可以随便命名无所谓的。



三、 安装依赖集

打开 pycharm 的运行终端如图



依次输入下面命令黄色标记：

pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple（切换国内镜像）

conda create -n unsloth python=3.11

conda activate unsloth

pip install -U ultralytics

pip uninstall torch

pip uninstall torchvision

提前下好 pytorch 的 whl（不然要等很久，可能还会中断）

Window：https://download.pytorch.org/whl/cu128/torch-2.7.0%2Bcu128-cp311-cp311-win_amd64.whl#sha256=bf88f647d76d79da9556ca55df49e45aff1d66c12797886364343179dd09a36c

Linux：https://download.pytorch.org/whl/cu128/torch-2.7.0%2Bcu128-cp311-cp311-manylinux_2_28_x86_64.whl#sha256=c4bbc0b4be60319ba1cefc90be9557b317f0b3c261eeceb96ca6e0343eec56bf

pip install D:\download\(根据自己下载的 whl 路径修改)torch-2.7.0+cu128-cp311-cp311-win_amd64.whl

pip3 install torch==2.7.0 torchvision torchaudio --index-url https://download.pytorch.org/whl/cu128

大模型 Flash-atten 注意力包下载：

Linux 系统 whl 文件下载地址：

https://github.com/Dao-AILab/flash-attention/releases

Window 系统 whl 文件下载地址：

https://github.com/bdashore3/flash-attention/releases

pip install D:\download\flash_attn-2.7.4.post1+cu128torch2.7.0cxx11abiFALSE-cp311-cp311-win_amd64.whl

pip install torch==2.7.0 xformers

pip install -U albumentations

pip install huggingface_hub

pip install pycurl

pip install datasets

pip install unsloth

pip install wandb

pip install tensorboard tensorboardX

安装完所有的依赖集后就可以准备 Lora 微调了

编译中出现 UnicodeDecodeError: 'gbk' codec can't decode byte 0x92 in position 30362: illegal multibyte sequence 这种问题，可尝试先去 llama-factory 的目录包里，参考说明书下载相关依赖集环境，补齐环境再回头试试 unsloth-Lora 运行训练代码

四、 Lora 微调

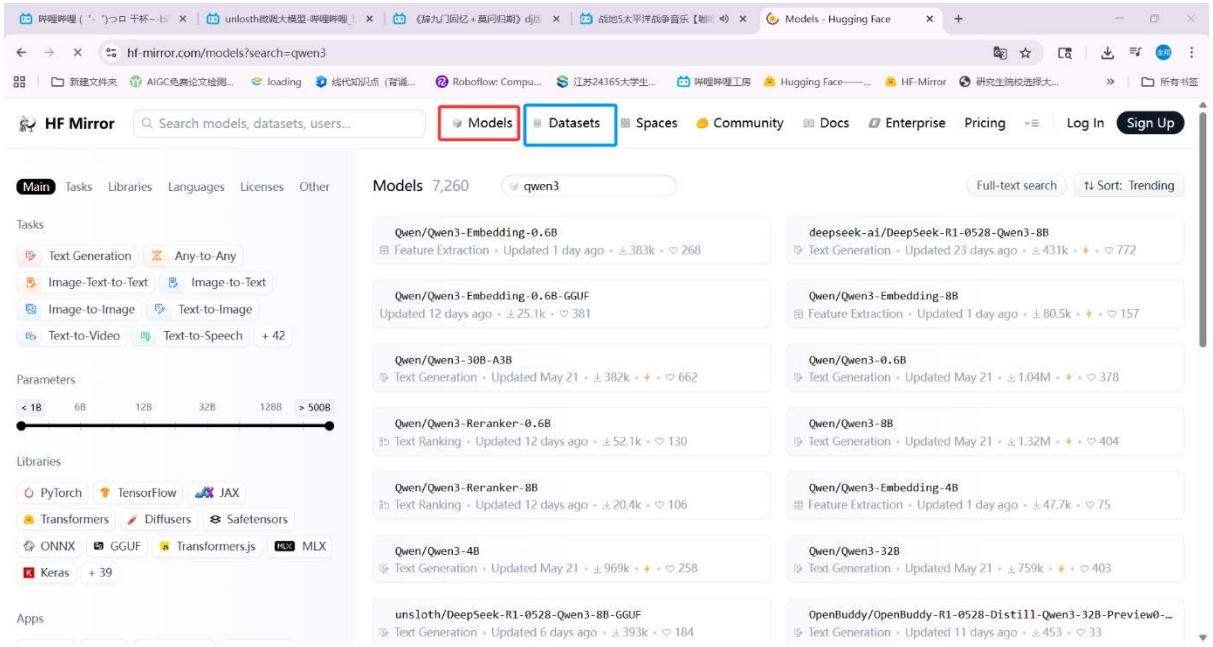
打开代码包，可以看到有 datasets 文件夹（数据集存放，文件为）和 models（原模型文件，文件为阿里通意千问 3-1.4B），可以根据自己的实际任务场景和硬件进行模型替换

原模型下载地址：

https://huggingface.co/（抱抱脸）

https://hf-mirror.com/（抱抱脸-国内镜像）

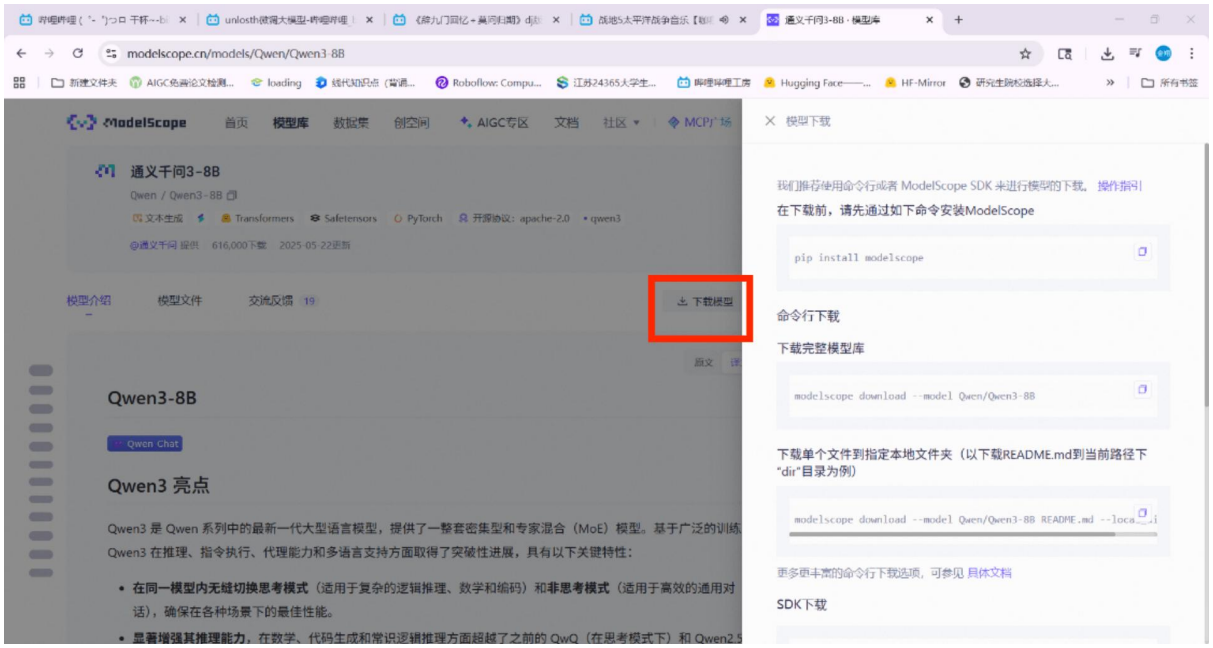
搜索自己想要的模型和数据集



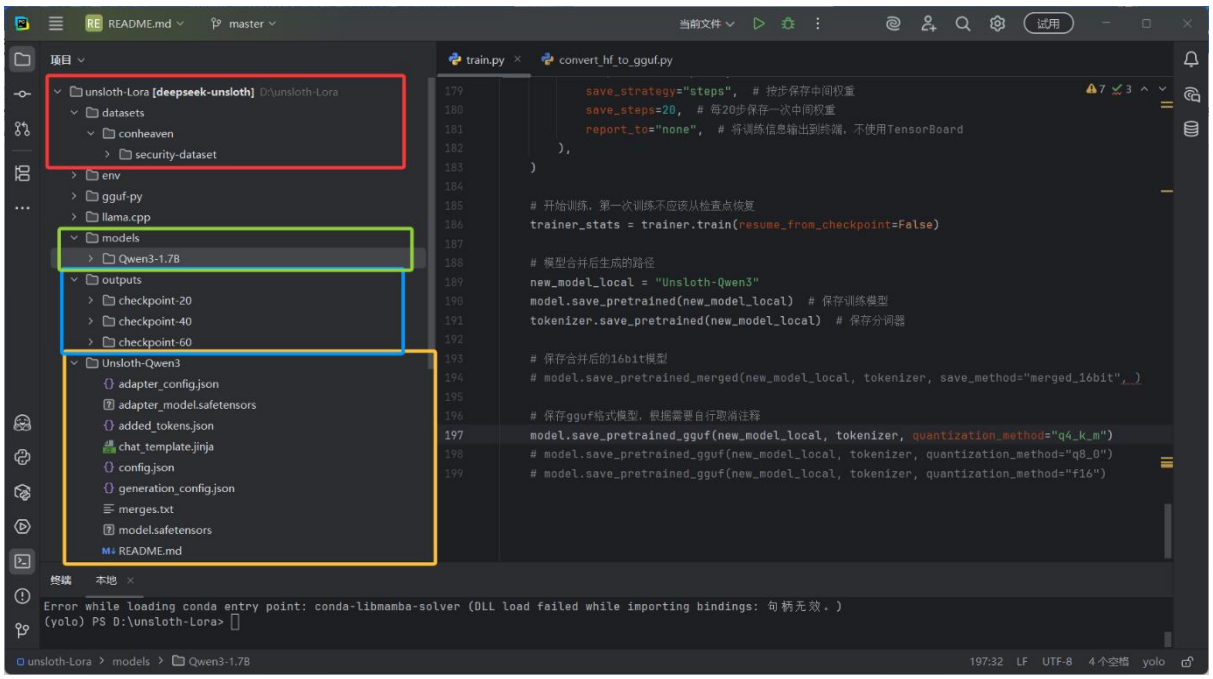
红色为模型基底模型，蓝底为数据集

https://modelscope.cn/（魔搭社区-国内大模型交流社区平台）

（强烈推荐）



可以在魔搭社区下载模型，点击，查看下载命令（可根据命令，稍作修改，更改下载位置，要有一定动手能力），数据集的下载也是一样



完成模型和数据集拉取后，可以对文件进行替换，红色标记为数据集替换区，可能要根据数据集不同，对相关代码，提示词进行处理：

```
train_prompt_style = """以下是描述任务的指令，以及提供更多上下文的输入。
                    请写出恰当完成该请求的回答。
                    在回答之前，请仔细思考问题，并创建一个逐步的思维链，以确保回答合乎逻辑且准确。

                    ### Instruction:
                    你是一位在网络安全、网络攻防、信息保护和安全架构设计方面具有专业知识的网络安全专家。
                    请回答以下网络安全相关问题。

                    ### Question:
                    {}

                    ### Response:
                    <think>
                    {}
                    </think>
                    {}"""

EOS_TOKEN = tokenizer.eos_token # 添加结束符标记

# 格式化提示函数,用于处理数据集中的示例
def formatting_prompts_func(examples):
    # 从 examples 中提取问题和回答
```

```
    inputs = examples["instruction"]  # 网络安全问题列表
    outputs = examples["output"]  # 回答列表
    texts = []  # 存储格式化后的文本
    # 遍历每个示例,将问题和回答组合成指定格式
    for input, output in zip(inputs, outputs):
        # 为思维链部分提供空字符串，使用 train_prompt_style 模板格式化文本,并添加结束符
        # 提供三个参数：问题、思维链(空字符串)、回答
        text = train_prompt_style.format(input, "", output) + EOS_TOKEN
        texts.append(text)
    # 返回格式化后的文本字典
    return {
        "text": texts,
    }

# 加载数据集并应用格式化
dataset = load_dataset("datasets/NetworkSecurity", split="train",
                       trust_remote_code=True)
dataset = dataset.map(formatting_prompts_func, batched=True, )
```

绿色标记为模型源文件，可更换自己拉取文件，对应代码如下：

```
# 加载预训练模型和分词器
model, tokenizer = FastLanguageModel.from_pretrained(
```

```
    model_name="./models/Qwen3-1.7B",
    max_seq_length=max_seq_length,
    dtype=dtype,
    load_in_4bit=load_in_4bit,
)
```

完成数据集和模型替换后，运行 train.py，模型开始进行学习训练，参数的更改可查看 readme.md 文件，对训练次数等可进行更改优化。

注：代码如果实在不会调整，推荐下载 cursor，注册一个新账号（每个新账号可在 14 天内免费使用 50 次智联体模型，比如更改查询解析创作代码，使用 Agent 让其自动更改，之后想用可借助 B 站，GitHub，淘宝等渠道，找免费或者低价续杯软件使用）使用 claude 系列模型（不要选择自动，自动智商过低）进行自动识别项目包和代码进行更换，Trae 也可以，不过不推荐，性能不及 Cursor。

蓝色 outpoint 为微调模型的相关检查点，可对一个时间段进行模型检查

黄色为最终 Lora 微调完模型，可对微调完的模型使用 inference.py 验证模型。

五、 进阶，GGUF 格式转换

所有微调的模型，应该都是 HF 格式的原始模型，需要一定编程水平才能部署，当然可以选择 GGUF 量化格式的模型后导入到 ollama 或者 LM-studio 中一键部署，默认量化为 FP16，如果需要量化到 Q4_K_M 或其他量化,可对下面代码进行取消注释等

```
# 保存合并后的 16bit 模型
# model.save_pretrained_merged(new_model_local, tokenizer, save_method="merged_16bit", )

# 保存 gguf 格式模型，根据需要自行取消注释
model.save_pretrained_gguf(new_model_local, tokenizer, quantization_method="q4_k_m")
# model.save_pretrained_gguf(new_model_local, tokenizer, quantization_method="q8_0")
# model.save_pretrained_gguf(new_model_local, tokenizer, quantization_method="f16")
```

并且需要下载 cmake（官方下载），对 llama.cpp 进行重新编译，编译命令如下（windows）：

```
cd llama.cpp
mkdir build
cd build
cmake -G "Visual Studio 17 2022" -A x64 -DLLAMA_CURL=OFF ..
cmake --build . --config Release
```

Linux：

```
git clone --recursive https://github.com/ggerganov/llama.cpp
    cd llama.cpp && make clean && make all -j
```

把 build/bin/Release/llama-quantize.exe, llama.dll ggml-cpu.dll,ggml-base.dll,ggml.dll 移动到 llama.cpp 目录下，运行 train.py 训练，即可，会发现最后转换的 GGUF 格式的大模型也在黄色 Lora 微调完的模型文件当中。