

以下将严格按照“**阿里云服务器 + VSCode SSH + LLaMA-Factory Web UI**”的流程，一步一步来，你只需要复制粘贴命令和点击按钮即可完成LLAMA-Factory的Web UI界面微调的学习。

注意，这是一个可以**直接跟着执行的保姆级教程**。

阶段 0：购买与配置阿里云服务器 (ECS)

这是最关键的第一步，请务必仔细操作。

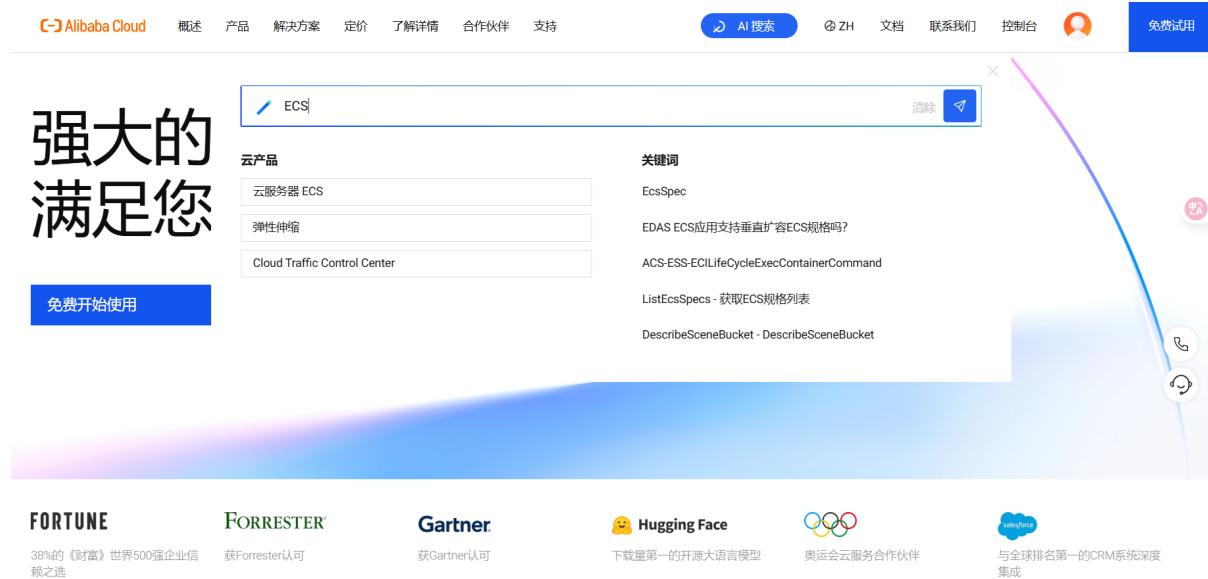
1. 注册与登录：

- 访问[阿里云官网](#)并完成注册和实名认证。
- 注意，微调使用huggingface要使用国外的节点，国内防火墙很麻烦。使用国内镜像的话又可能会造成版本不一致产生错误的麻烦问题。

那么，你可以选择使用阿里云国际站，也可选择国内的。选择国际站难点在于你需要使用护照或者驾驶执照进行验证。使用国内版本的阿里云，你需要特别注意[选择国外网点](#)，因为这会大大简化依赖包版本不匹配的问题。

2. 进入 ECS 控制台：

- 登录后，在顶部搜索框搜索“**ECS**”（即“云服务器 ECS”），点击进入控制台。



3. 创建实例（买服务器）：

- 在 ECS 概览页面，点击“**创建实例**”。

配置概要

付费类型	包年包月
地域	新加坡
可用区	随机分配可用区
网络类型	专有网络
专有网络	默认专有网络
交换机	默认交换机
实例规格	计算型 c9i / ecs.c9i.large (2 vCPU, 4 GiB, Intel)

购买实例数量: 1

购买时长: 配置费用: \$0.088864 USD/时
公网流量费用: \$0.081 USD/GB
查看明细

确认下单

4. 详细配置（关键步骤）：

- 计费方式:** 选择“**按量付费**”。（非常重要！这样你不用的时候关机就不扣费，成本最低）。
- 地域:** 选择“**香港**”。

配置概要

付费类型	包年包月
地域	中国香港
网络及可用区	默认专有网络

购买实例数量: 1

购买时长: 配置费用: \$2.167 USD/时
公网流量费用: \$0.081 USD/GB
查看明细

确认下单

- 实例规格:**

- 点击“**实例规格**”旁的“**选择**”。
- 在弹出的窗口中，左侧导航栏找到“**GPU 计算型**”。
- 新手推荐 (T4):** 选择 **ecs.gn6i-c8g1.2xlarge** (8 vCPU, 32 GiB 内存, 1 * NVIDIA T4 16GB 显存)。这个性价比最高，适合入门。

GPU 计算型 gn6i	ecs.gn6i-c8g1.2xlarge	8 vCPU	31 GiB	intel	1 * NVIDIA T4	1 *	\$2.167 USD /时
--------------	-----------------------	--------	--------	-------	---------------	-----	----------------

- 进阶推荐 (A10):** 选择 **ecs.gn7i-c8g1.2xlarge** (8 vCPU, 31 GiB 内存, 1 * NVIDIA A10 24GB 显存)。这个性能更好，可以跑更大的模型。

- 如果某个地区没有某个类别的服务器，可以查看还有哪些较近的地区。

- 选择一个后，点击“**选定**”。

- **镜像 (操作系统):**

- 这是**最关键的一步**。点击“**镜像**旁的“**选择**”。
- 选择“**公共镜像**”。
- 操作系统**: 选择 **Ubuntu**。
- 版本**: 选择 **22.04 或 20.04**。
- 关键**: 要安装CUDA和cuDNN。
- 例如: **Ubuntu 22.04 64位 + CUDA 12.1 或 Ubuntu 20.04 64位 + CUDA 11.8**。这会帮你省去安装 GPU 驱动的巨大麻烦。

- 点击“**选定**”。

- **存储**: 系统盘默认即可，不需要数据盘。

系统盘 如何选择云盘	类型	容量	数量	IOPS	性能	设置
	ESSD云盘	40	GiB	1	2280	PL0 (单盘IOPS性能上限1万)
						<input checked="" type="checkbox"/> 随实例释放
						<input type="checkbox"/> 加密

云盘性能: 不同云盘性能不同, [各云盘性能指标](#)>
云盘容量: 云盘创建总大小会受到配额限制

数据盘

+ 添加数据盘 (0 / 16)

◦ 网络与安全组：

- 网络默认的“专有网络”即可。
- **公网 IP：** 必须勾选“**分配公网 IPv4 地址**”。
- **安全组：** 点击“**新建安全组**”。
 - 在“规则”里，确保 **22 端口** 是开放的（用于 SSH）。默认就是开放的。
 - （备注：我们稍后使用 VSCode 的端口转发功能，所以你**不需要在这里开放 7860 端口，更安全。**）

带宽计费模式

按固定带宽
适用于流量较大、稳定的场景

按使用流量
适用于流量小、波动大的场景

升级至CDT计费
开通云数据传输CDT, 赠送220GB/月公网流量 (中国内地地域20GB/月, 非中国内地200GB/月), 超出部分采用阶梯计费, 查看图文说明. 价格详情 >

带宽峰值

1	2	3	5	10	50	100	Mbps
-	5	+	Mbps				

① 按使用流量计费模式下的出入带宽峰值都是带宽上限，不作为业务承诺指标。当出现资源争抢时，带宽峰值可能会受到限制。如果您的业务需要有带宽的保障，请使用按固定带宽计费模式。

阿里云免费提供最高 5Gbps 的恶意流量攻击防护。了解更多> | 提升防护能力>

安全组 ②

已有安全组 [新建安全组](#)

如何配置安全组

安全组名称

安全组类型 普通安全组 企业级安全组 [普通安全组和企业级安全组的对比差异>](#)

开通IPv4端口/协议

<input checked="" type="checkbox"/> SSH (TCP: 22)	<input type="checkbox"/> HTTP (TCP: 80)	<input type="checkbox"/> HTTPS (TCP: 443)
<input checked="" type="checkbox"/> RDP (TCP: 3389)	<input checked="" type="checkbox"/> ICMP (IPv4)	

⚠ 启用SSH (端口22) 时， 默认允许所有IP访问，这可能带来安全风险（如暴力破解攻击）。建议您创建实例后立即修改安全组设置，仅保留必要IP的访问权限。

弹性网卡 ③

请先指定交换机

IPv6

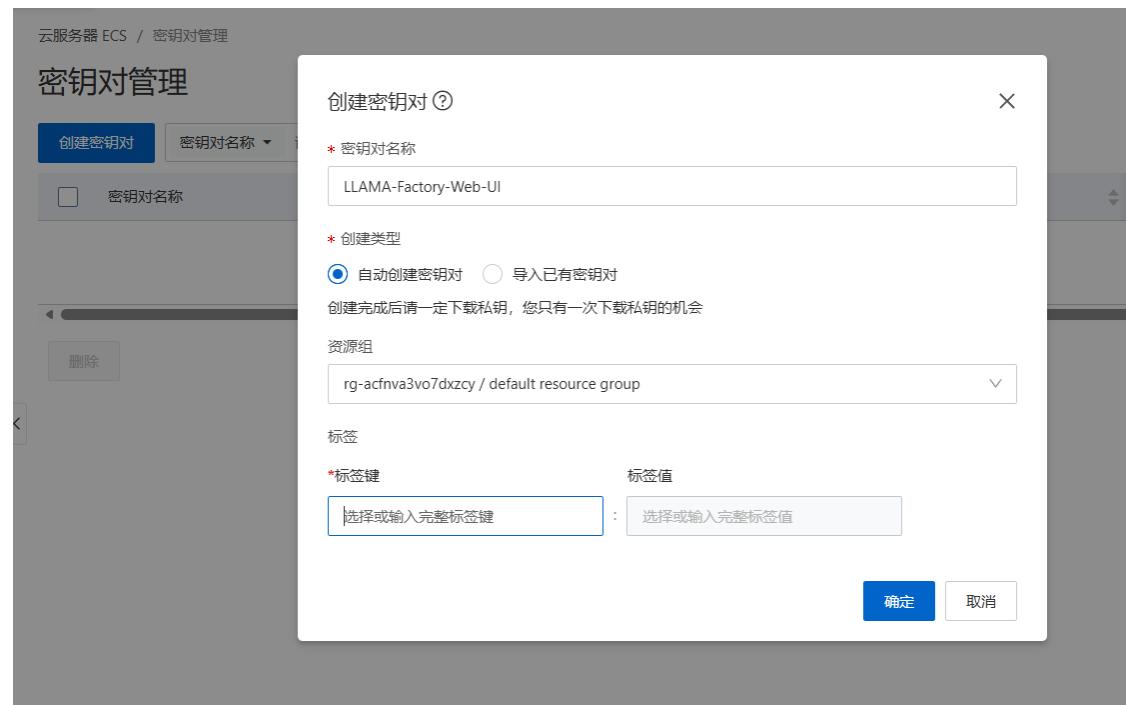
请先指定交换机

[▲ 弹性网卡 | IPv6 \(选填\)](#)

◦ 登录凭证：

- **登录凭证：** 选择“**密钥对**”。
- 点击“**新建密钥对**”。
- **密钥对名称：** 随便起一个，比如 **LLAMA-Factory-Web-UI**。
- **创建方式：** 选择“**自动新建密钥对**”。
- 点击“**确定创建**”。浏览器会自动下载一个 **.pem** 文件（例如 **my-gpu-key.pem**）。

■ 保存好这个文件！它是你登录的唯一钥匙，丢了就登不上了。



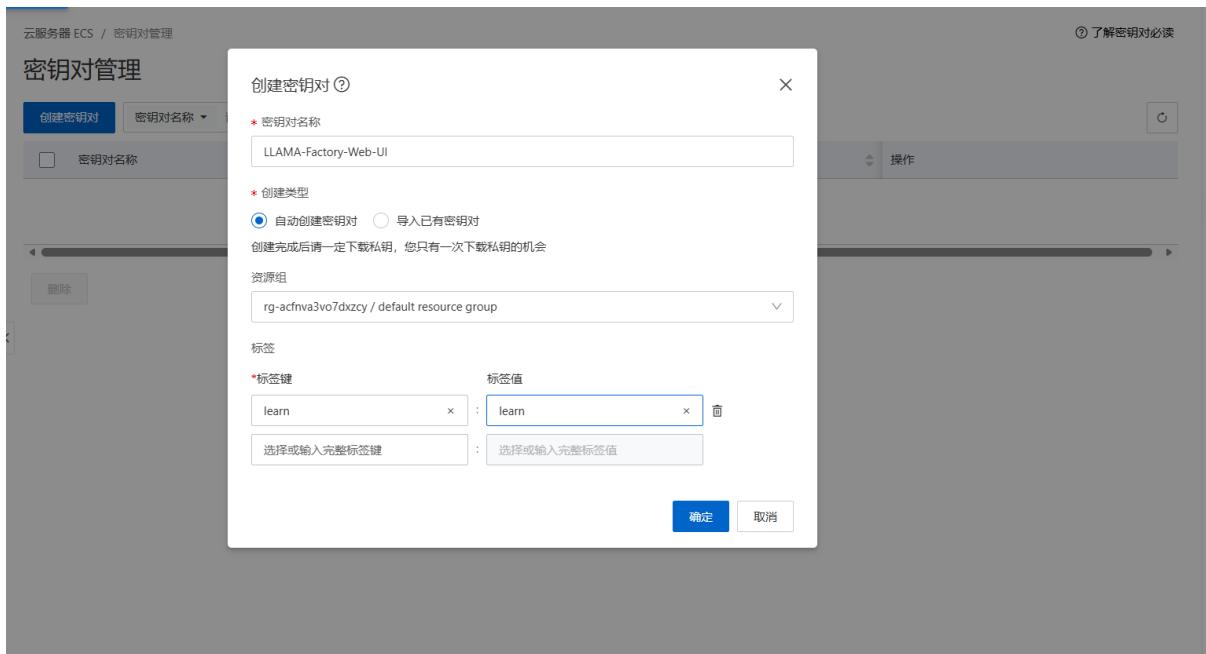
你看到的 标签键 和 标签值 是阿里云用来帮你分类资源（比如给服务器打个“开发中”或“生产中”的标签），它不影响任何功能。

这里你随便填一个就可以通过验证了。

建议你这样填：

标签键：输入 purpose （或者中文 用途）

标签值：输入 llama-factory



下载 ...

LLAMA-Factory-Web-UI.pem
[打开文件](#)

查看更多

管理设置

登录凭证

密钥对

自定义密码

创建后设置

登录名

root ecs-user

root具有操作系统的最高权限，使用root作为登录名可能会导致安全风险，建议您使用ecs-user作为登录名。[前往了解更多>](#)

密钥对

请选择密钥对

标签

如何设计标签

标签由区分大小写的键值对组成。您设置的标签将应用在本次创建的全部实例和云盘

○ 确认订单：

- 勾选“我已阅读并同意...”。

■ 点击“创建实例”。



5. 查看你的服务器 IP:

- 等待 1-3 分钟，服务器创建成功。
- 在“实例列表”中找到你的新服务器，复制它旁边的“公网 IP”地址。

阶段 1：使用 VSCode 连接服务器

1. 本地安装 VSCode:

- 如果还没有，去 [VSCode 官网](#) 下载并安装。

2. 安装 SSH 插件:

- 打开 VSCode，点击左侧的“扩展”图标 (四个方块)。
- 搜索 **Remote - SSH**，点击第一个 (微软官方出品)，安装它。

3. 配置 SSH 连接:

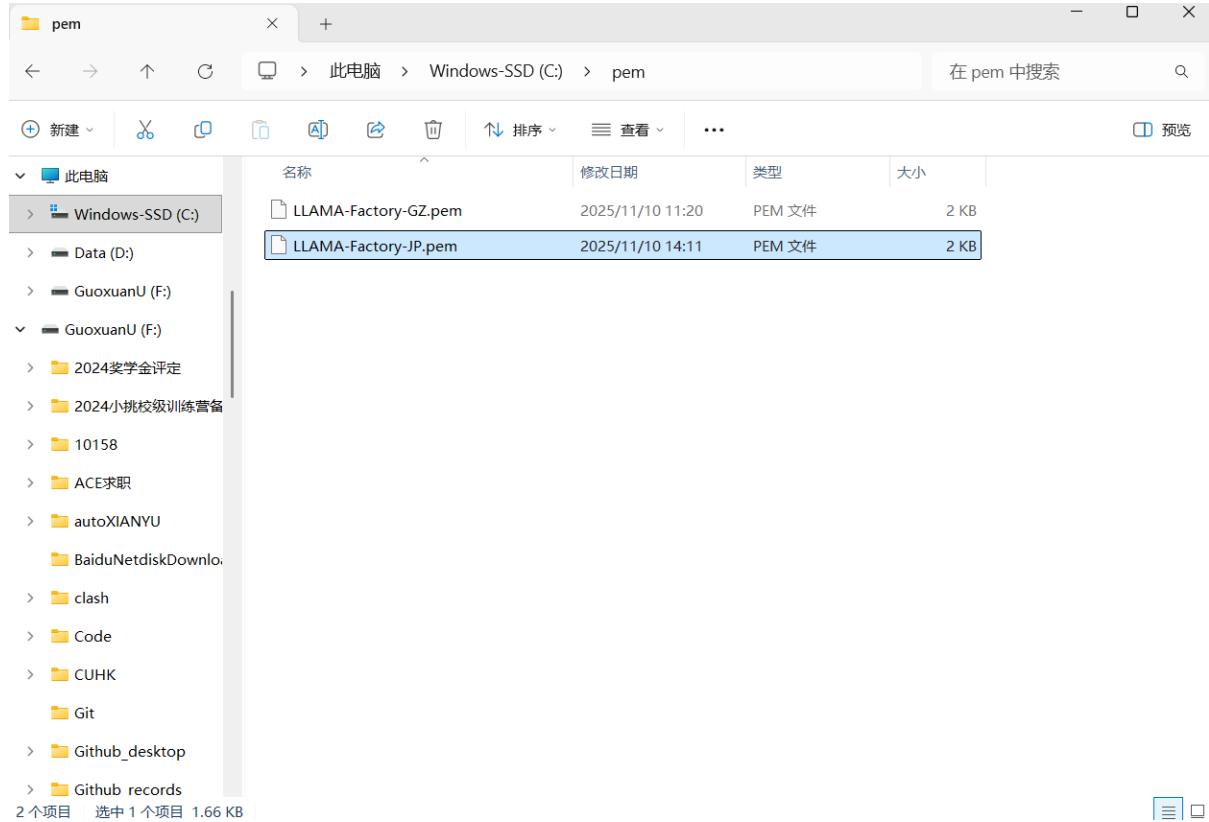
- 安装后，按 **F1** 键 (或者 **Ctrl+Shift+P**) 打开命令面板。
- 输入 **Remote-SSH: Open Configuration File...**，回车。
- 选择第一个文件，通常是 **.../.ssh/config**，打开它。
- 在文件末尾，粘贴以下内容，并替换为自己的信息：

```
Host LLAMA-Factory-GZ # 你给服务器起的别名，随便填
  HostName 你的公网ip
  User root      # 阿里云的 Ubuntu 镜像默认是 root 用户
  IdentityFile 这里是你刚才下载的 .pem 文件的完整路径
    # 这里替换成你刚才下载的 .pem 文件的【完整路径】，注意，一定要放在系统盘，例如C
    盘，因为后续要调整权限问题。
                                              # Windows 示例：C:/Users/你的用户名
名/Downloads/my-gpu-key.pem
                                              # Mac/Linux 示例：/Users/你的用户名
名/Downloads/my-gpu-key.pem
```

- **注意:** Windows 路径请使用 / 而不是 \。
- 保存并关闭这个 config 文件。

4. 调整pem文件权限

- 打开所在的地址。



- 点击高级。

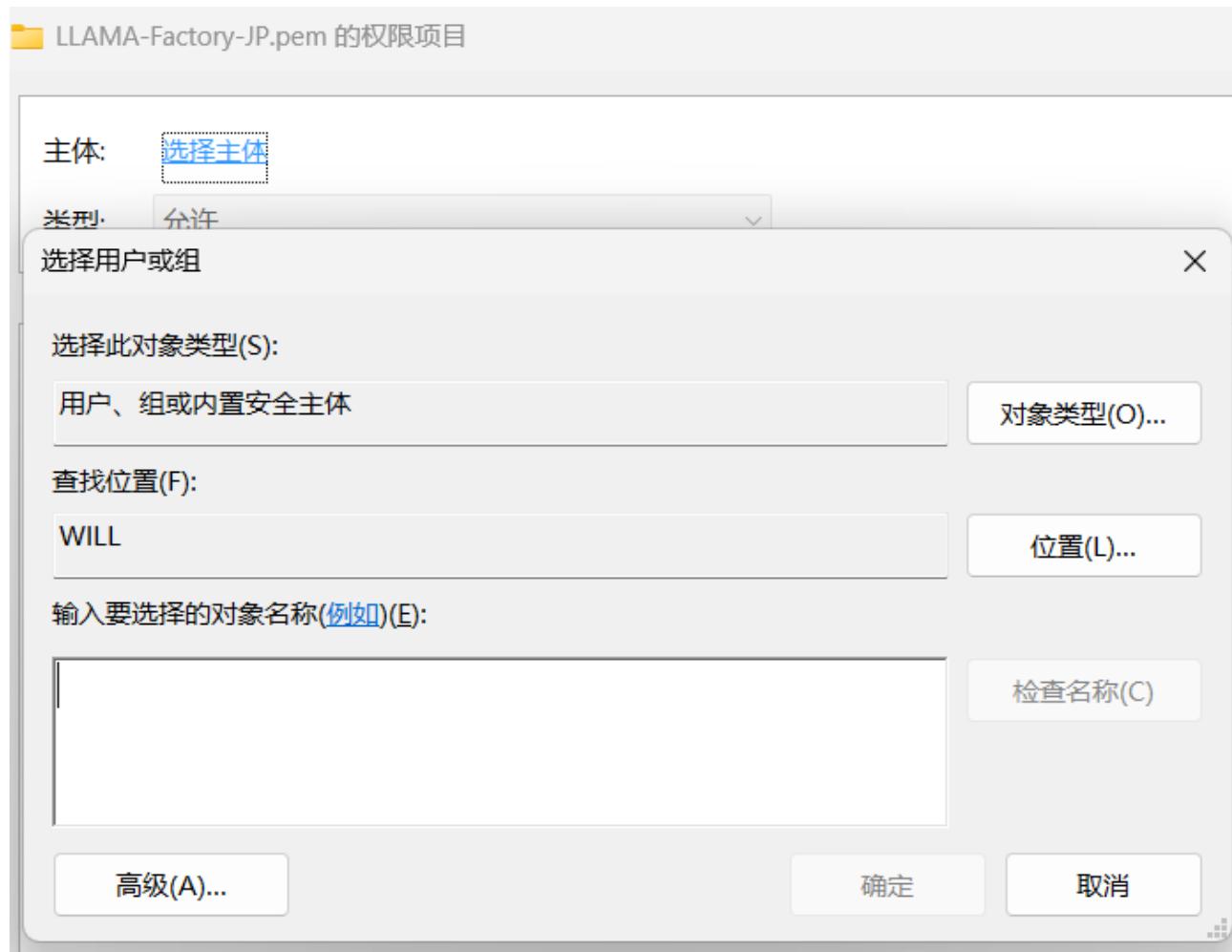


- 点击启用继承，选择第一个清除掉所有的用户的继承关系。



- 添加你这个主题。





把你的用户添加进去。

- 最后变成这样。



5. 连接!

- 点击 VSCode 左下角的绿色 >< 图标。
- 在弹出的菜单中，选择 Remote-SSH: Connect to Host...。
- 选择你刚才配置的 LLAMA-Factory-GZ。
- VSCode 会打开一个新窗口，并开始连接。第一次连接会询问你系统平台（选 Linux）和是否信任（选 Continue）。
- 连接成功后，VSCode 的左下角会显示 SSH: aliyun-gpu。

阶段 2：在服务器上安装 LLaMA-Factory

你现在的所有操作都是在你的阿里云服务器上。

1. 打开 VSCode 终端：

- 在 VSCode 菜单栏选择 Terminal -> New Terminal。

2. 检查 GPU (定心丸)：

- 在终端里输入：

```
nvidia-smi
```

- 你应该能看到一块 NVIDIA T4 (或 A10) 的信息。这表示 GPU 一切正常。



3. 安装依赖环境：

- 复制并粘贴以下命令，一次性执行：

```
# 更新包列表
apt update
# 安装 git, python3-pip 和 python3-venv
apt install git python3-pip python3-venv -y
```

```
Every 1.0s: nvidia-smi
iz6we26cuynbtmrzk7gljcZ: Mon Nov 10 14:50:27 2025
+--+
| NVIDIA-SMI 535.216.03 | Driver Version: 535.216.03 | CUDA Version: 12.2 |
+--+
| GPU Name Persistence-M Bus-Id Disp.A Volatile Uncorr. ECC |
| Fan Temp Perf Pwr:Usage/Cap | Memory-Usage | GPU-Util Compute M. |
| | | | | MIG M. |
+---+
| 0 NVIDIA A10 On 00000000:00:07.0 Off 0 |
| 0% 70C P0 138W / 150W 13062MiB / 23028MiB 51% Default N/A |
+---+
+--+
| Processes: |
| GPU GI CI PID Type Process name GPU Memory |
| ID ID |
+---+
| 0 N/A N/A 3059 C /root/LLaMA-Factory/venv/bin/python3 226MiB |
+---+
LAMA-Factory-JP ② 0 △ 0 ④ 1
```

🦙ビル LLaMA Factory: Unified Efficient Fine-Tuning of 100+ LLMs

Visit [GitHub Page Documentation](#)

Language	Model name	Model path	Hub name
en	Qwen1.5-1.8B-Chat	Path to pretrained model or model identifier from Hugging Face. Qwen/Qwen1.5-1.8B-Chat	Choose the model download source. huggingface
Finetuning method	Checkpoint path		
lora			
Quantization bit Enable quantization (QLoRA).	Quantization method Quantization algorithm to use.	Chat template The chat template used in constructing prompts. qwen	RoPE scaling RoPE scaling method to use. none
		Booster Approach used to boost training speed. auto	

4. 下载 LLaMA-Factory:

- cd /root (进入你的主目录)
- git clone <https://github.com/hiyouga/LLaMA-Factory.git>

Train	Evaluate & Predict	Chat	Export
Stage The stage to perform in training. Supervised Fine-Tuni	Data dir Path to the data directory. data	Dataset alpaca_gpt4_zh	Preview dataset
Learning rate Initial learning rate for AdamW.	Epochs Total number of training epochs to perform. 5e-5	Maximum gradient norm Norm for gradient clipping. 3.0	Max samples Maximum samples per dataset. 1.0
Cutoff length Max tokens in input sequence. 2048	Batch size Number of samples processed on each GPU. 2	Gradient accumulation Number of steps for gradient accumulation. 1024	Val size Percentage of validation set from the entire dataset. bf16
LR scheduler Name of the learning rate scheduler. cosine	0	0	1

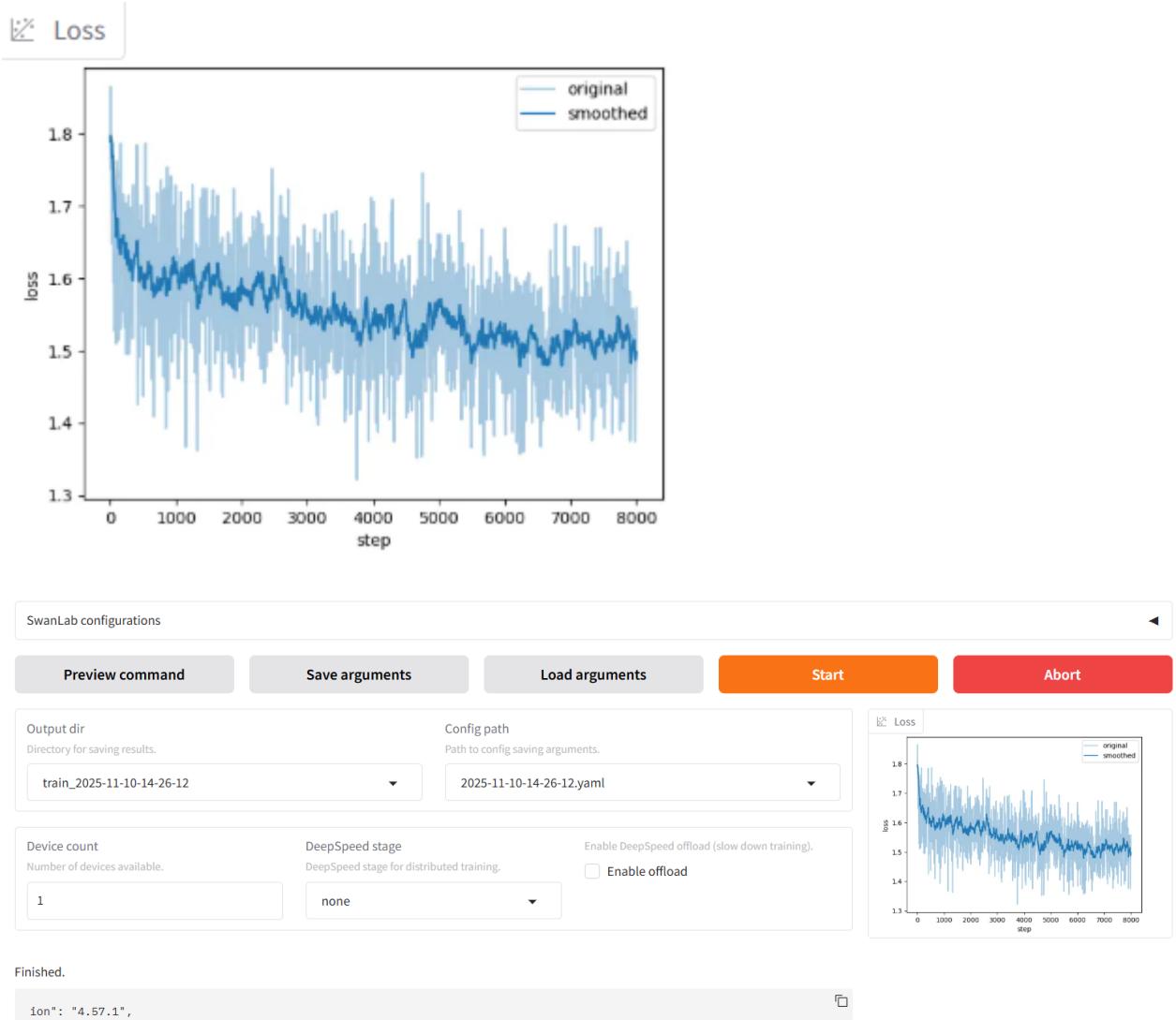
5. 安装 LLaMA-Factory:

```
# 1. 进入刚刚下载的目录
cd LLaMA-Factory
```

```
# 2. 创建一个 Python 虚拟环境, 命名为 venv
python3 -m venv venv

# 3. 激活这个虚拟环境 (非常重要!)
source venv/bin/activate
# (激活后, 你的命令行前面会多一个 (venv) 标记)

# 4. 安装 LLaMA-Factory 核心依赖和 4-bit 量化支持
pip install -e .[torch,bitsandbytes]
```



阶段 3：运行 LLaMA-Factory Web UI 并实战微调

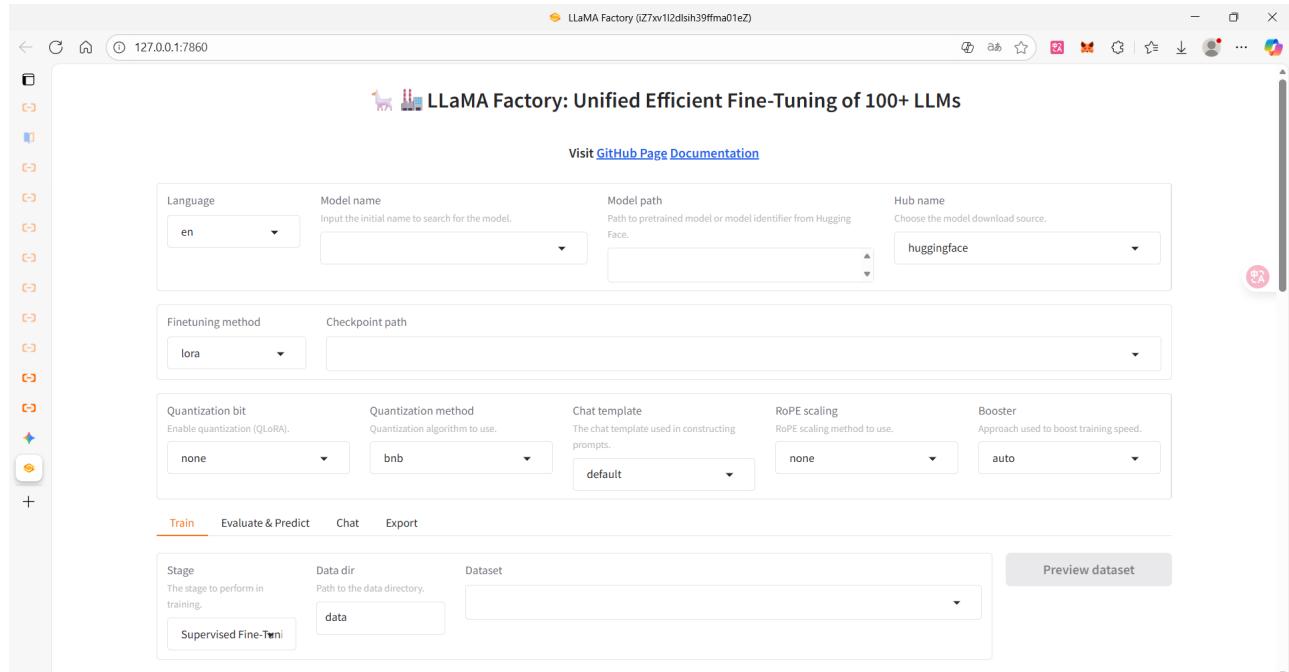
1. 启动 Web UI:

- 确保你仍处于 LLaMA-Factory 目录且 (venv) 虚拟环境已激活。
- 在 VSCode 终端中运行：

```
CUDA_VISIBLE_DEVICES=0 llamafactory-cli webui
```

- 终端会开始运行，并显示 Running on local URL: http://127.0.0.1:7860。

运行之后可能会直接在你的本地弹出浏览器界面：



这时候你就不用进行2. 访问 Web UI (VSCode 端口转发)的任务了。

2. 访问 Web UI (VSCode 端口转发):

- 不要在你的本地浏览器打开 127.0.0.1:7860 (那是无效的)。
- 在 VSCode 窗口中，找到“端口 (Ports)”标签页（它和“终端”在同一个面板）。
- 点击“转发端口 (Forward a Port)”。
- 输入 7860，回车。
- VSCode 会自动为你设置好转发。在“端口”列表里，7860 端口后面会出现一个“在浏览器中打开 (Open in Browser)”的小地球图标。
- 点击这个小地球图标！
- VSCode 会在你的本地浏览器中打开一个 http://localhost:7860 的地址。
- 成功！你现在看到了 LLaMA-Factory 的 Web UI 界面。

3. 执行 SFT 微调任务 (Qwen1.5-1.8B)

在你的本地浏览器打开的 Web UI 界面上，按顺序点击设置：

- (1) Model (模型) 标签页：**
 - Model name (模型名称):** 从下拉框中选择 Qwen/Qwen1.5-1.8B-Chat。 (它会在运行时自动下载)
 - Finetuning method (微调方法):** 选择 lora。
 - Quantization (量化):** 选择 4-bit。 (这是 QLoRA，极大节省显存)
- (2) Dataset (数据集) 标签页：**
 - 在 Dataset (数据集) 下拉框中，找到并勾选 alpaca_gpt4_zh (一个常用的中文指令数据集)。
 - (你可以点一下 Preview dataset 预览数据格式)

- (3) Hyperparams (参数) 标签页:

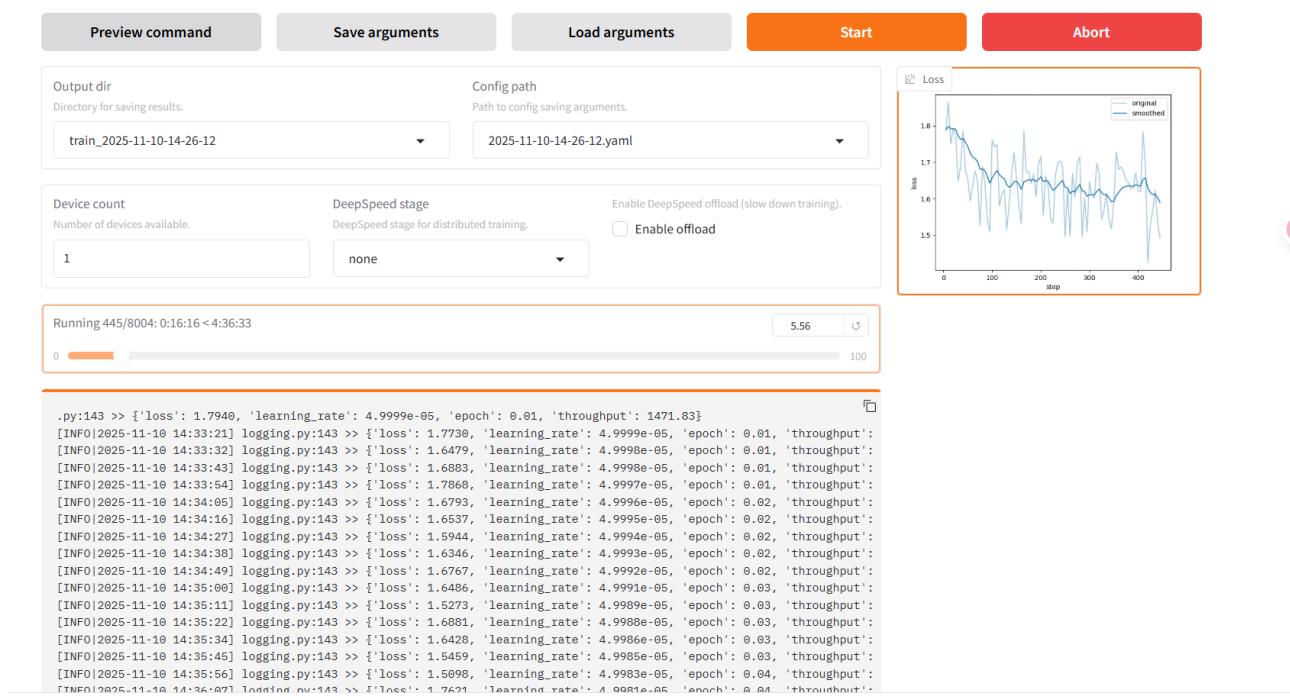
- Epochs (训练轮数): 设为 3.0。
- Batch size (批处理大小): 设为 2。 (如果你的显存是 T4 16GB, 2 是绝对安全的值)。
- Learning rate (学习率): 保持默认的 5e-5 即可。

- (4) Run (运行) 标签页:

- Output directory (输出目录): 没有办法自己设置。Output dir Directory for saving results: train_2025-11-10-12-36-55 Config path Path to config saving arguments: 2025-11-10-12-36-55.yaml
- 点击 Start 按钮!

4. 监控训练:

- 看 Web UI: 界面下方的 Logs (日志) 框会开始滚动。首先它会下载模型, 然后开始训练, 你会看到进度条和 Loss (损失) 值出现。



- 看 GPU (推荐): 在 VSCode 里, 点击终端面板右上角的 + 号, 新开一个终端。



- 在新终端里输入 watch -n 1 nvidia-smi (每秒刷新一次 nvidia-smi)。

- 你将看到 GPU 的 **Volatile GPU-Util** (GPU使用率) 飙升到 90% 以上, **Memory-Usage** (显存占用) 也会很高。这证明你的 GPU 正在全力“炼丹”!

```
Every 1.0s: nvidia-smi
Mon Nov 10 14:50:27 2025
+-----+
| NVIDIA-SMI 535.216.03      Driver Version: 535.216.03    CUDA Version: 12.2 |
+-----+
| GPU Name Persistence-M Bus-Id Disp.A Volatile Uncorr. ECC |
| Fan Temp Perf Pwr:Usage/Cap | Memory-Usage GPU-Util Compute M. |
|                               | MIG M. |
+-----+
| 0  NVIDIA A10          On  00000000:00:07.0 off | 51%   Default |
| 0% 70C P0           138W / 150W | 13062MiB / 23028MiB | N/A |
+-----+
| Processes:                   GPU Memory |
| GPU GI CI PID Type Process name     Usage |
| ID ID   ID   |                         |
+-----+
| 0 N/A N/A 3059 C   /root/LLaMA-Factory/venv/bin/python3 226MiB |
+-----+
LAMA-Factory-JP @ 0 △ 0 ━━ 1
```

- 根据 T4 的性能, 这个 1.8B 模型的训练任务大概需要 15-30 分钟。

解释一下这个监控图：

📈 关键指标解读

1. **`Process name: .../LLaMA-Factory/venv/bin/python3`**
* **含义:** 在下方的“进程”表里, 你可以看到 `python3` 正在运行, 它正是 LLaMA-Factory 的主程序。
 2. **`GPU Name: NVIDIA A10`**
* **含义:** 你使用的是 A10 GPU, 这比教程中的 T4 更强, 拥有 24GB 显存 (图上显示为 23028MiB)。
 3. **`Pwr:Usage/Cap: 141W / 150W`**
* **含义:** 这是**最重要的信号!** 你的 GPU 正在消耗 141 瓦的电力, 已经**接近其 150 瓦的功率上限**。这表明 GPU 正在全力工作。
 4. **`Perf: P0`**
* **含义:** `P0` 状态代表“最高性能”。你的 GPU 已经火力全开。
 5. **`Memory-Usage: 13062MiB / 23028MiB`**
* **含义:** 你的程序已经占用了 13GB 的显存。这包括:
 - * 4-bit 量化后的 `Qwen-1.8B` 模型。
 - * LoRA 适配器参数。
 - * 优化器状态 (比如 AdamW)。
 - * 当前正在处理的数据批次 (`Batch size: 2`)。* 占用 13GB 显存是**完全正常**的, 说明模型和数据都已成功加载。
-

？ 常见疑问：为什么 GPU-Util (使用率) 只有 35%?

你可能在想, 教程里说会飙到 90%, 为什么我这里只有 35%?

回答: 这是完全正常的。**

`GPU-Util` 显示的是**计算核心**的使用率。在训练时, GPU 不仅仅在“计算”, 它还在“等待数据”。

在你的任务中 (`Batch size: 2`)，GPU (A10) 的计算速度**非常快**，而 CPU 准备下一批数据（读取、分词、打包）的速度相对较慢。

这会导致一个常见的“瓶颈”现象：

1. CPU 把一批数据 (batch 1) 交给 GPU。
2. GPU **瞬间** (比如 0.1 秒) 完成计算 (Util 翻到 100%)。
3. GPU **等待** (比如 0.2 秒) CPU 把下一批数据 (batch 2) 准备好 (Util 降到 0%)。
4. `nvidia-smi` 每秒刷新一次，它看到的只是这个过程的*平均值* $((100\% * 0.1s + 0\% * 0.2s) / 0.3s \approx 33\%)$ 。

总结：

不要看 `GPU-Util`，**请看 `Pwr:Usage` (功率)**。你的功率 (141W) 已经跑满了，这证明 GPU 正在全力工作。

结论

一切顺利。你现在只需要：

1. **回到 LLaMA-Factory 的 Web UI 浏览器页面。**
2. 查看下方的 `Logs (日志)` 框。
3. 你很快就会看到**训练进度条**开始滚动，显示 `Step`、`Loss` (损失值) 和 `Epoch` (轮数)。

恭喜你，你已经成功开始了你的第一次微调！

我这里记录这个任务所需要的具体的时间：开始：[INFO]2025-11-10 14:27:30] 结束：

我这个任务的具体的参数是：

LLaMA Factory: Unified Efficient Fine-Tuning of 100+ LLMs

Visit [GitHub Page Documentation](#)

Language	Model name	Model path	Hub name	
en	Input the initial name to search for the model. Qwen1.5-1.8B-Chat	Path to pretrained model or model identifier from Hugging Face. Qwen/Qwen1.5-1.8B-Chat	Choose the model download source. huggingface	
Finetuning method	Checkpoint path			
lora				
Quantization bit	Quantization method	Chat template	RoPE scaling	Booster
Enable quantization (QLoRA).	Quantization algorithm to use. bnb	The chat template used in constructing prompts. qwen	RoPE scaling method to use. none	Approach used to boost training speed. auto

Train Evaluate & Predict Chat Export

Stage Stage to perform in training. **Data dir** Path to the data directory. **Dataset** alpaca_gpt4_zh **Preview dataset**

Supervised Fine-Tune

Learning rate Initial learning rate for AdamW. **Epochs** Total number of training epochs to perform. **Maximum gradient norm** Norm for gradient clipping. **Max samples** Maximum samples per dataset. **Compute type** Whether to use mixed precision training.

5e-5 3.0 1.0 100000 bf16

Cutoff length Max tokens in input sequence. **Batch size** Number of samples processed on each GPU. **Gradient accumulation** Number of steps for gradient accumulation. **Val size** Percentage of validation set from the entire dataset. **LR scheduler** Name of the learning rate scheduler.

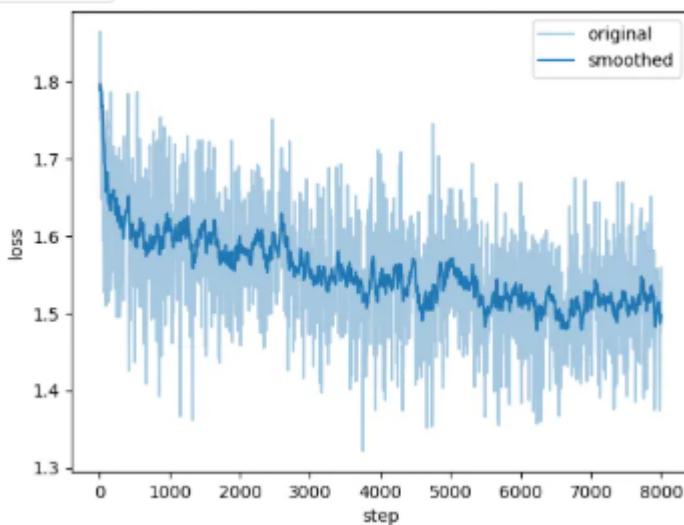
2048 2 8 0 cosine

4 131072 1 1024 1

阶段 4：测试与导出你训练的模型

训练完成后，日志会显示 "Finetuning finished."。

Loss



SwanLab configurations

Preview command

Save arguments

Load arguments

Start

Abort

Output dir

Directory for saving results.

train_2025-11-10-14-26-12

Config path

Path to config saving arguments.

2025-11-10-14-26-12.yaml

Device count

Number of devices available.

1

DeepSpeed stage

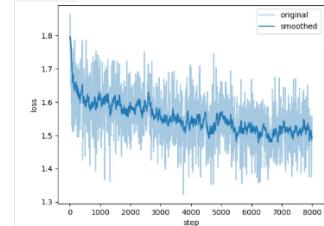
DeepSpeed stage for distributed training.

none

Enable DeepSpeed offload (slow down training).

Enable offload

Loss



Finished.

```
ion": "4.57.1",
"max_workers": 4096
```

1. 在线聊天测试：

首先你需要去加载一个检查点，这记录了你最优模型的位置。然后点load model。

Visit [GitHub Page Documentation](#)

Language: en

Model name: Qwen1.5-1.8B-Chat

Model path: Qwen/Qwen1.5-1.8B-Chat

Hub name: huggingface

Finetuning method: lora

Checkpoint path: train_2025-11-10-14-26-12

Quantization bit: 4

Quantization method: bnb

Chat template: qwen

RoPE scaling: none

Booster: auto

Inference engine: huggingface

Extra arguments: {"vllm_enforce_eager": true}

Train Evaluate & Predict Chat Export

Load model Unload model

这就是你微调之后的模型。

Role: user

System prompt (optional)

Tools (optional)

Input...

Maximum new tokens: 1024

Top-p: 0.7

Temperature: 0.95

Skip special tokens

Escape HTML tags

Enable thinking

Clear history

Submit

Device memory: Current memory usage of the device (GB). 0 / 22.19

让我们来进行几次测试：

期望的差别（微调前 vs 微调后）：微调前 (Base Qwen-Chat): 作为通义千问，它已经能回答你的问题。但它的回答可能是常规的聊天风格，比如一段话，或者一个简单的列表，不一定会严格遵守你设定的“数量”或“格式”要求。

微调后 (你训练的模型): 你的模型被 alpaca_gpt4_zh (GPT-4 级别的数据) “教导”过，它应该更严格地遵守指令的结构。当你要求“三点”时，它会非常精确地给出三点，并且格式工整（比如使用 1. 2. 3.），回答的质量也应该更高。

微调前：

我应该如何开始学习冥想？请给我三个简单的入门步骤。



学习冥想可以帮助你放松身心，提高注意力和集中力，同时也有助于提升情绪稳定性和内在平静。以下是一些简单的冥想入门步骤：

1. 找一个安静的环境：选择一个安静、无干扰的环境，可以是你的卧室、客厅、公园或者户外的任何地方。确保你的环境舒适，温度适宜，光线适中，可以让你更容易集中注意力。

2. 选择一个舒适的坐姿：冥想时，建议采用坐姿，如盘腿坐或半坐姿，使你的脊柱保持自然的曲线。可以选择一个舒适的椅子，但不要过于软或过于硬，以避免影响你的坐姿和呼吸。

3. 关闭眼睛：在冥想开始之前，先关闭眼睛，让自己的思绪从外部世界暂时转移至自己的内心世界。这可以帮助你集中注意力，减轻杂念和干扰。

4. 深呼吸：开始冥想时，可以先进行深呼吸练习，慢慢地吸气，然后慢慢地呼气，重复几次。深呼吸可以帮助你放松身心，平静呼吸的节奏可以有助于引导你的思绪集中到冥想的中心。



5. 开始冥想：找到一个安静的时刻，然后开始冥想。你可以选择从5分钟开始，逐渐增加冥想的时间，直到你感觉身心完全放松，可以保持5-10分钟。开始时，你可以先从简单的呼吸练习开始，然后逐渐加入冥想的技巧和内容。

6. 专注于呼吸：在冥想过程中，将注意力集中在你的呼吸上，感受你的呼吸在你的身体中流动。这可以帮助你降低心率，放松身体，从而达到冥想的效果。尝试将注意力从你的呼吸转移到周围的环境或事物上，但不要过于执着于这种转移，保持注意力的稳定性。

7. 慢慢增加：开始时，你可以尝试在几分钟内进行冥想，然后逐渐增加冥想的时间。每次增加冥想时间时，你可以尝试在冥想开始时设定一个目标，如专注于5分钟的呼吸，然后逐渐增加到10分钟，15分钟，20分钟等。这样可以帮助你逐渐适应冥想的过程，并逐渐提高冥想的持久性。

8. 持续练习：冥想需要持续练习才能达到良好的效果。每天花一些时间进行冥想，即使只有几分钟，也能帮助你逐渐提高冥想的技巧和效果。在日常生活中，可以将冥想作为一种放松身心、提高注意力和集中力的方式，每天花几分钟进行冥想，以此来提升自己的身心健康。

以上就是学习冥想的三个简单入门步骤，希望对你有所帮助。记住，冥想是一种实践，需要时间和耐心，但只要坚持下去，你一定能从中获得许多益处。

微调后：

我应该如何开始学习冥想？请给我三个简单的入门步骤。



以下是三个简单的冥想入门步骤：

1. 找一个安静的地方：找一个安静、没有干扰的地方，可以是卧室、公园、客厅等，让你可以集中注意力，减少外界干扰。

2. 坐姿端正：坐姿要端正，背部挺直，双脚平放在地上，不要交叉或翘起。调整坐姿，使脊柱保持自然直立。

3. 注意呼吸：闭上眼睛，专注于你的呼吸。吸气时，慢慢地吸气，让腹部膨胀，然后慢慢地呼气，让腹部收缩。每次呼吸时，都可以想象自己在大自然中呼吸，感受清新的空气。注意，不要去控制呼吸，只需专注于当下呼吸，享受这种平静的感觉。



2. 导出完整模型：

导出模型的正确步骤：

请你现在按照以下步骤，在你的 **Export** 页面上补全设置即可：

1. **Model name:** Qwen1.5-1.8B-Chat

2. **Hub name:** huggingface

3. **Finetuning method:** lora

4. **【关键步骤 1：选择你的补丁】**

■ 在 **Finetuning method** 下方，找到 **Checkpoint path** (检查点路径) 这个下拉框。

■ 点击它。

■ 在弹出的列表中，勾选你刚刚训练好的那个适配器（补丁），它的名字应该类似于 **train_2025-11-10_14-51-09** (即你训练时 **Output dir** 的名字)。

■ (这就是教程中所说的 **Adapter path**)

5. **【关键步骤 2：命名你的模型】**

- 找到 **Export dir** (导出目录) 这个输入框。
- 在框里输入一个名字, 比如教程中推荐的: **my_first_merged_model**

6. 【最后一步：开始导出】

- 点击页面中间那个大大的、灰色的 **Export** 按钮。
- (这就是教程中所说的 **Start** 按钮)

```
[INFO]configuration_utils.py:986] 2025-11-10 19:55:39,786 >> Generate config GenerationConfig {
    "bos_token_id": 151643,
    "do_sample": true,
    "eos_token_id": [
        151645,
        151643
    ],
    "pad_token_id": 151643,
    "repetition_penalty": 1.1,
    "top_p": 0.8
}

[INFO]dynamic_module_utils.py:423] 2025-11-10 19:55:39,953 >> Could not locate the custom generate/generate.py inside Qwen/Qwen1.5-1.8B-Chat.
[INFO]2025-11-10 19:55:39] llmfactory.model.model_utils.attention:143 >> Using torch SDPA for faster training and inference.
[INFO]2025-11-10 19:55:43] llmfactory.model.adapter:143 >> Merged 1 adapter(s).
[INFO]2025-11-10 19:55:43] llmfactory.model.adapter:143 >> Loaded adapter(s): saves/Qwen1.5-1.8B-Chat/lora/train_2025-11-10-14-26-12
[INFO]2025-11-10 19:55:43] llmfactory.model.loader:143 >> all params: 1,836,828,672
[INFO]2025-11-10 19:55:43] llmfactory.train.tuner:143 >> Convert model dtype to: torch.bfloat16.
[INFO]configuration_utils.py:491] 2025-11-10 19:55:43,453 >> Configuration saved in my_first_merged_model/config.json
[INFO]configuration_utils.py:757] 2025-11-10 19:55:43,453 >> Configuration saved in my_first_merged_model/generation_config.json
```

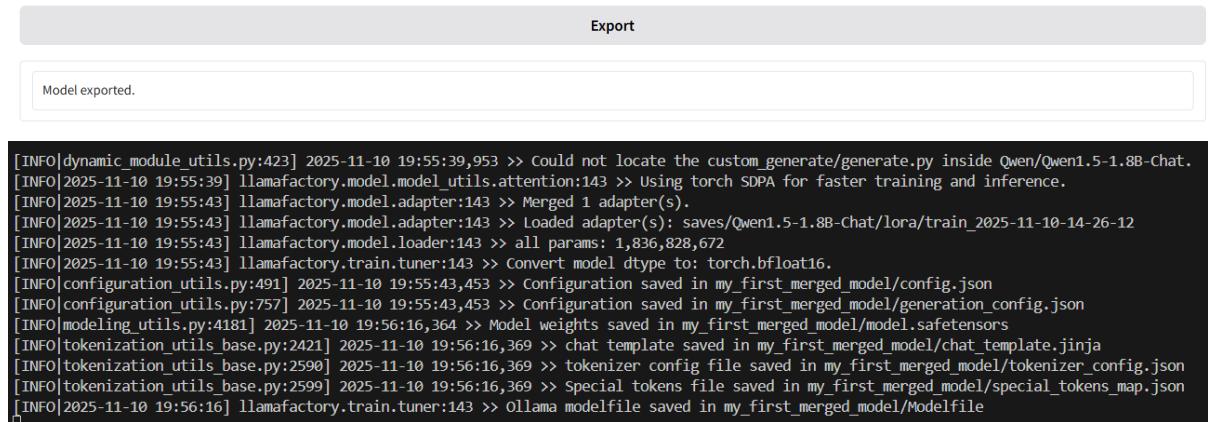
注意, 你可能会遇到空间不够的情况, 请你新开一个终端, 执行下面的命令来清除缓存:

```
# 1. 清理 apt 软件包缓存
apt-get clean

# 2. 清理 pip 库缓存
pip cache purge
```

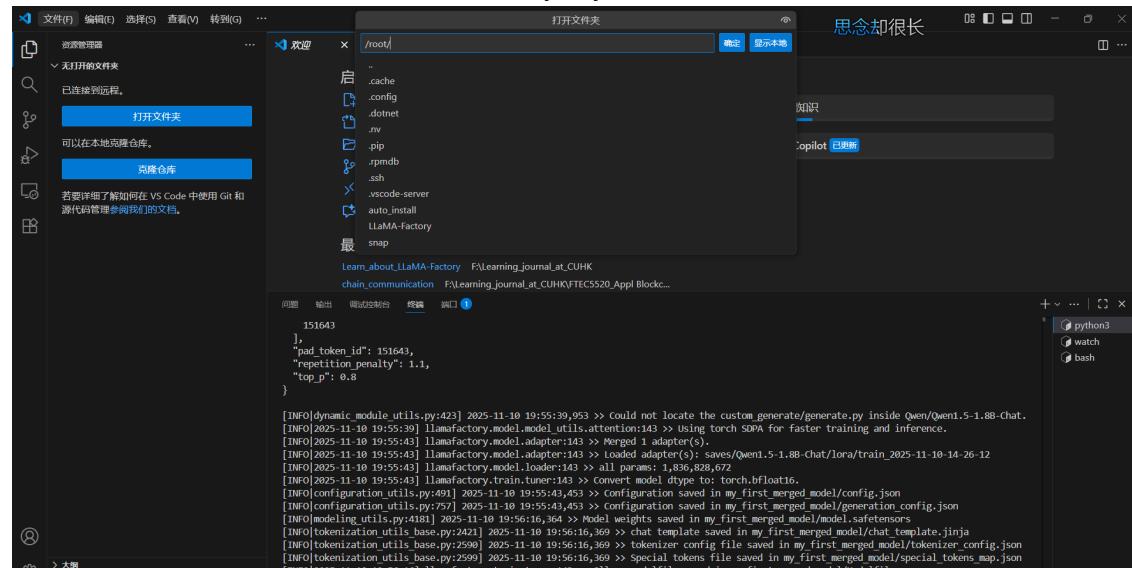
接下来会发生什么?

- 点击 **Export** 后, 请回到你的 VSCode 终端查看日志。

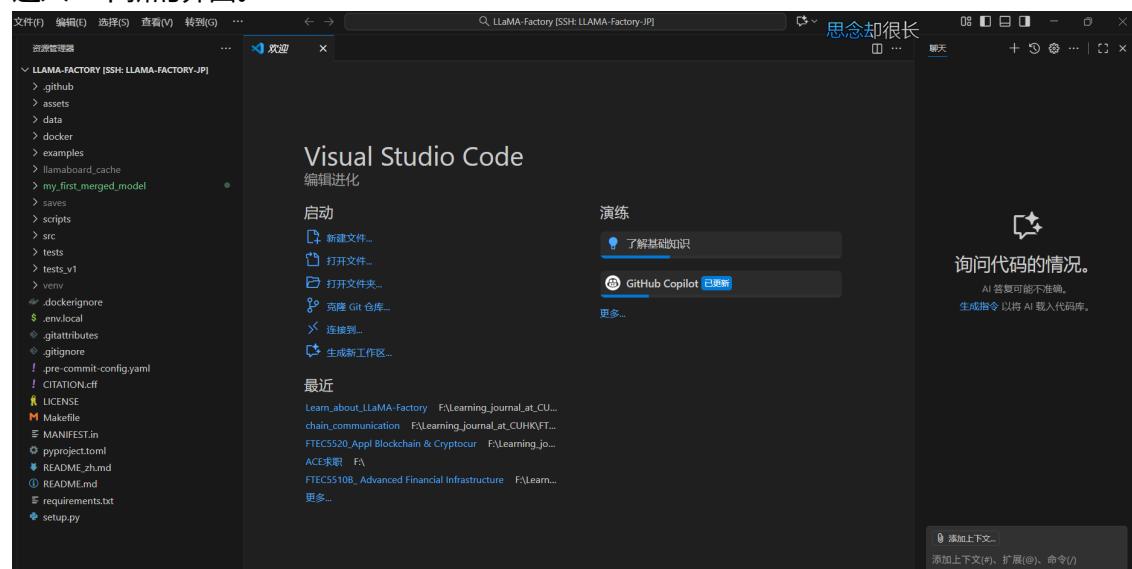


- 你会看到 LLaMA-Factory 开始加载基础模型 (Qwen), 然后加载你的 LoRA 补丁, 将它们合并, 最后保存到你指定的 **my_first_merged_model** 文件夹中。
- 这个过程会占用一些时间。当日志显示完成后, 你就可以在 VSCode 左侧的文件浏览器中看到你导出的完整模型了!
- 如何找到这个下载下来的模型:

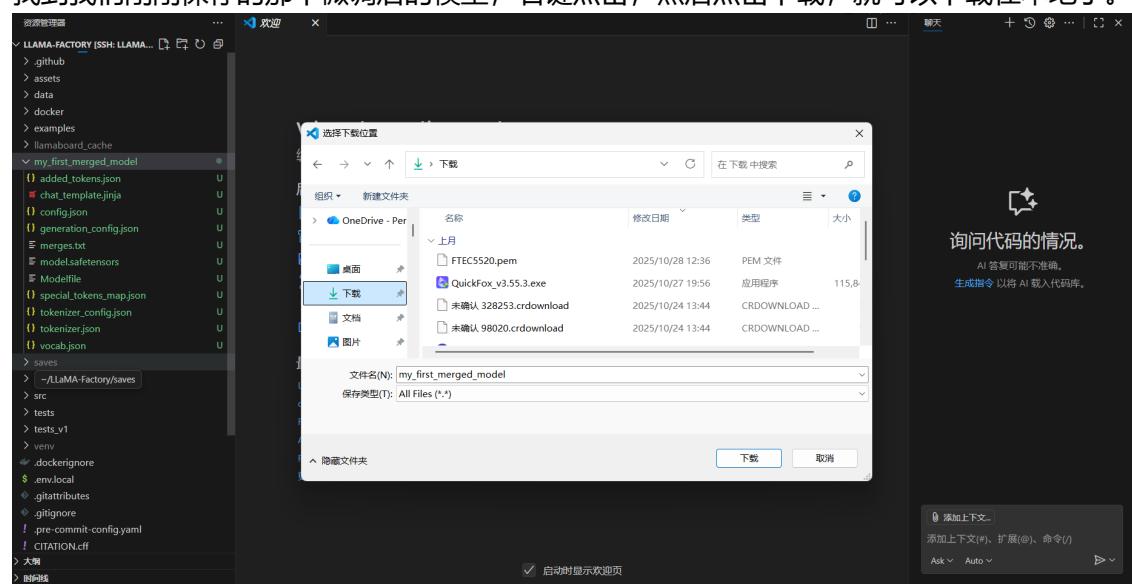
■ 你需要打开文件夹，点击创建的LLAMA-Factory这个文件夹。



■ 进入一个新的界面。



■ 找到我们刚刚保存的那个微调后的模型，右键点击，然后点击下载，就可以下载在本地了。



阶段 5：清理战场（非常重要！）

1. 停止 Web UI:

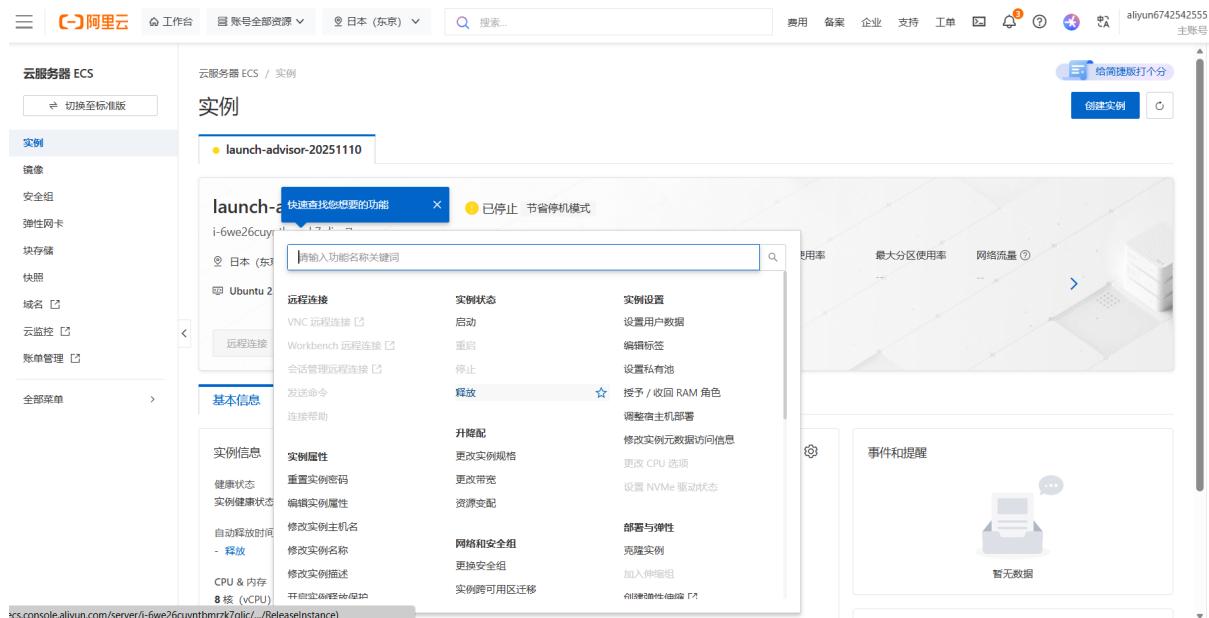
- 在 VSCode 里运行 `llamafactory-cli webui` 的那个终端里，按 `Ctrl + C` 来停止服务。

```
问题 输出 调试控制台 终端 端口 1
+ bash - LLaMA-Factory ... ×

root@iZ6we26cuyntbmrzk7g1jcZ:~/LLaMA-Factory# llamafactory-cli webui
llamafactory-cli: command not found
root@iZ6we26cuyntbmrzk7g1jcZ:~/LLaMA-Factory# source venv/bin/activate
(venv) root@iZ6we26cuyntbmrzk7g1jcZ:~/LLaMA-Factory# CUDA_VISIBLE_DEVICES=0 llamafactory-cli webui
Visit http://ip:port for Web UI, e.g., http://127.0.0.1:7860
* Running on local URL: http://0.0.0.0:7860
* To create a public link, set `share=True` in `launch()`.
^Keyboard interruption in main thread... closing server.
(venv) root@iZ6we26cuyntbmrzk7g1jcZ:~/LLaMA-Factory#
```

2. 停止服务器（省钱！）：

- 回到阿里云 ECS 控制台。**
- 找到你的实例，点击“操作” -> “停止”。
- 一定要选择“停止”而不是“重启”！**
- （注意：停止后保留实例，系统盘还是会收很小一笔存储费，但 GPU 和 CPU 的大头费用就停止了。如果你确定几天内都不用了，可以选择“释放”，这会彻底删除服务器。）
- 你可以给释放掉。



恭喜你！你已经完整地跑通了在云服务器上微调 AI 大模型的全过程！