0824部署讲义：
https://docs.qq.com/doc/p/1c62f7f7181735569ca5f01c2ba10fea223c9c8d

大模型的训练和部署都很消耗硬件资源，所以硬件越给力，模型的效果越好。

IB网卡（机间互联）可以极快地加速 服务器之间的通信，比IP通信高了N个层次，属于高配。IB和IP的区别是,IB网卡通信提供了更多的数据通道，比如IB是16车道的高速公路，但是IP通信只有做多4车道。只是一般人可能买不到IB网卡，配置很高，很昂贵。

学习大模型，一定进行多机多卡的训练和部署，积累多机多卡分布式的经验。

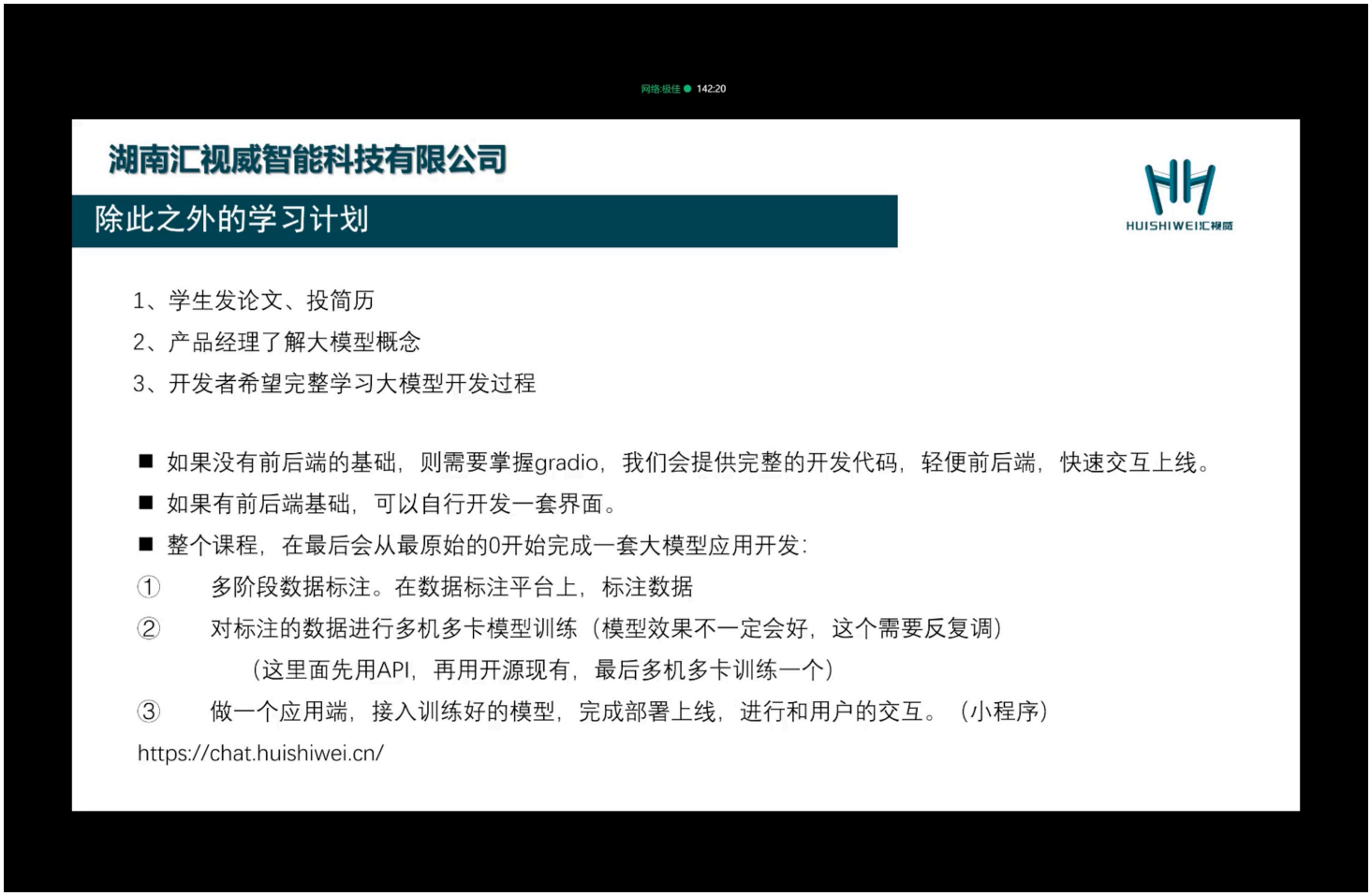
另外，做大模型应用一定要考虑多模态，因为单一的文本大模型的应用十分有限，要注意延伸到 视觉，音频等领域。

关于数据集，通常我们可以下载一些公开的数据集，然后解压看看他的内部结构，并仿照他们的结构来构建自己的数据集。

pytorch，大模型深度学习的框架。它提供了很多优化器。

需要掌握的技能如下：

最后会达成的目标是：



课程作业报告

由于没有物理机设备，所以直接从平台训练的课程复现。。

0824课程场景复现

多机多卡模型训练

步骤如下，

1. 设置IB网卡

```
export NCCL_DEBUG=INFO
export NCCL_IB_DISABLE=0
export NCCL_IB_HCA=mlx5
export NCCL_SOCKET_IFNAME=eth0
export GLOO_SOCKET_IFNAME=eth0
```

我们自己的机器一般是通过TCP连接来互联，但是 云平台一般是支持IB网卡。

这段配置主要是用于设置 NCCL（NVIDIA Collective Communications Library）和 GLOO 在多

机多卡训练实验中的网络通信参数。汇视威训练平台的超算集群使用方式完全兼容NVIDIA NCCL。注意，多机多卡的话，**每台机器**都要这样设置才能启用IB通信。

export NCCL_DEBUG=INFO

这个环境变量用于设置 NCCL 的调试级别。INFO 级别会输出有关 NCCL 的重要信息，有助于你跟踪和诊断 NCCL 在训练过程中的运行情况。调试级别包括：WARN、INFO、DEBUG 等，分别代表警告、信息和调试信息，INFO 是一种常见的选择，提供足够的详细信息以进行故障排除。

export NCCL_IB_DISABLE=0

这个环境变量控制 NCCL 是否使用 InfiniBand（IB）网络。如果设置为 0，表示启用 InfiniBand 网络。设置为 1 则表示禁用 InfiniBand 网络。InfiniBand 通常用于高性能计算集群中，以提供更高的带宽和低延迟的网络通信。

export NCCL_IB_HCA=mlx5

这个环境变量指定了 NCCL 使用的 InfiniBand HCA（Host Channel Adapter）。在这个例子中，mlx5 表示使用 Mellanox 的 InfiniBand HCA 设备。mlx5 是 Mellanox 的一种常见的 InfiniBand 硬件型号，它提供高带宽和低延迟的网络通信能力。

export NCCL_SOCKET_IFNAME=eth0

这个环境变量指定了 NCCL 使用的以太网接口。eth0 是常见的以太网接口名称。在多机训练中，NCCL 使用以太网进行节点之间的通信，这个变量告诉 NCCL 使用哪个网络接口来进行数据传输。

export GLOO_SOCKET_IFNAME=eth0

类似于 NCCL 的配置，GLOO_SOCKET_IFNAME 变量用于指定 GLOO（Facebook 的一个通信库）使用的以太网接口。在这个例子中，eth0 被指定为 GLOO 的网络接口。GLOO 也用于分布式训练中的进程间通信。

这些配置一起帮助优化和控制在多机多卡训练中网络通信的性能和行为。确保你的集群硬件和网络配置符合这些设置，以获得最佳的训练效果。

2. 设置网络代理以及环境变量

同样每台机器都要设置。

```
export http_proxy=http://10.10.9.50:3000
export https_proxy=http://10.10.9.50:3000
export no_proxy=localhost,127.0.0.1
export HF_HOME=/code/huggingface-cache
export HF_ENDPOINT=https://hf-mirror.com
```

`http_proxy`和`https_proxy`是为了设置网络代理，加快从外网下载资源的速度。

`no_proxy` 是设置不需要通过代理服务器的地址列表。

`HF_HOME` 是设置 以后 `huggingface-cli` 下载文件的目录。

`HF_ENDPOINT`是设置下载 `huggingface` 下载的终端节点，上面设置成镜像网站。这个地址会替代默认的 `Hugging Face` 服务器地址，从而使你的请求发送到指定的镜像服务器。也是为了加快从`huggingface`网站上下载文件的速度。

3. 启动训练

`NPROC_PER_NODE=4 NNODES=3 PORT=12345 ADDR=10.244.34.75 NODE_RANK=2 xtuner train /code/qwen1_5_1_8b_qlora_alpaca_e3_copy.py --work-dir /userhome/xtuner-workdir --deepspeed deepspeed_zero3`

逐个解释其中的参数：

`NPROC_PER_NODE=4`

每个节点上运行的进程（也就是GPU）数量，设置为4，意味着每台机器上有4个GPU参与训练。

`NNODES=3`

指定参与训练的总节点数。3个节点，也就是三台机器。

`PORT=12345`

分布式训练的通信端口，这个端口必须在所有节点上保持一致，并且在集群网络中开放，以确保各个节点能够相互通信。

`ADDR=10.244.199.246`

指定通信的主节点。分布式训练中，将会通过这个IP地址来连接主节点，进行协调和数据转换。（实操中注意用`ifconfig`查看当前节点的ipv4地址）

`NODE_RANK=0`

指定当前节点在集群中的排名或ID。主节点为0，其余的依次递增。

`xtuner train /code/qwen1_5_1_8b_qlora_alpaca_e3_copy.py --work-dir /userhome/xtuner-workdir --deepspeed deepspeed_zero3` 则是启动训练的命令。

- `xtuner` 是 分布式训练工具指令。
- `train /code/qwen1_5_1_8b_qlora_alpaca_e3_copy.py` 启动训练过程，训练的文件为

`/code`目录下的 `qwen1_5_1_8b_qlora_alpaca_e3_copy.py` 文件

- `--work-dir /userhome/xtuner-workdir` 指定训练过程中使用的工作目录,存储训练日志，模型检查点和临时文件。选择一个合适的训练目录有助于组织和训练管理数据。
- `--deepspeed deepspeed_zero3` 指定DeepSpeed训练策略，可以减少模型的内存占用和加速训练过程。

如何下载数据集

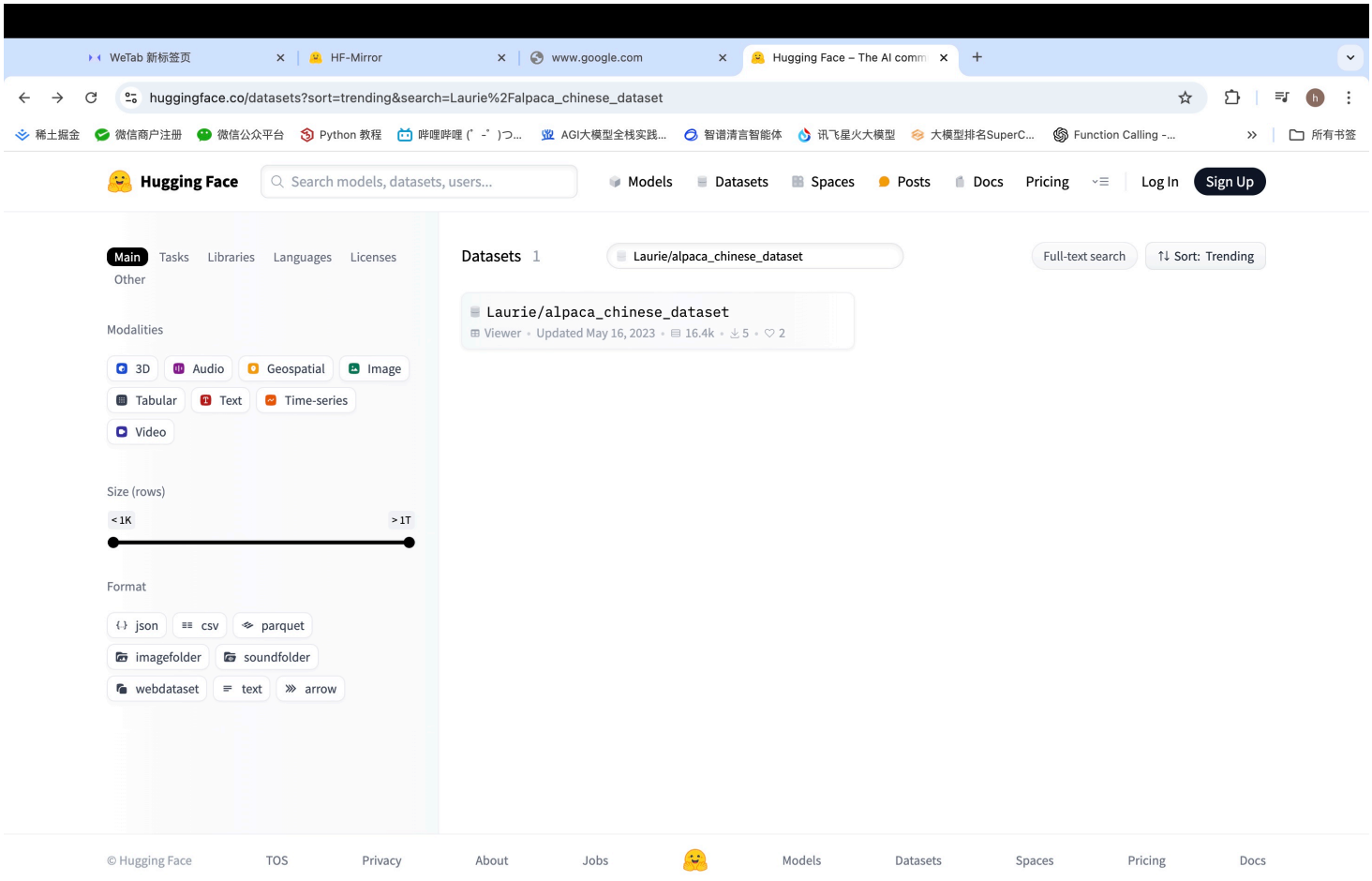
```
huggingface-cli download Laurie/alpaca_chinese_dataset --repo-type dataset --revision main --local-dir-use-symlinks False --local-dir /code/Laurie_alpaca_chinese_dataset
```

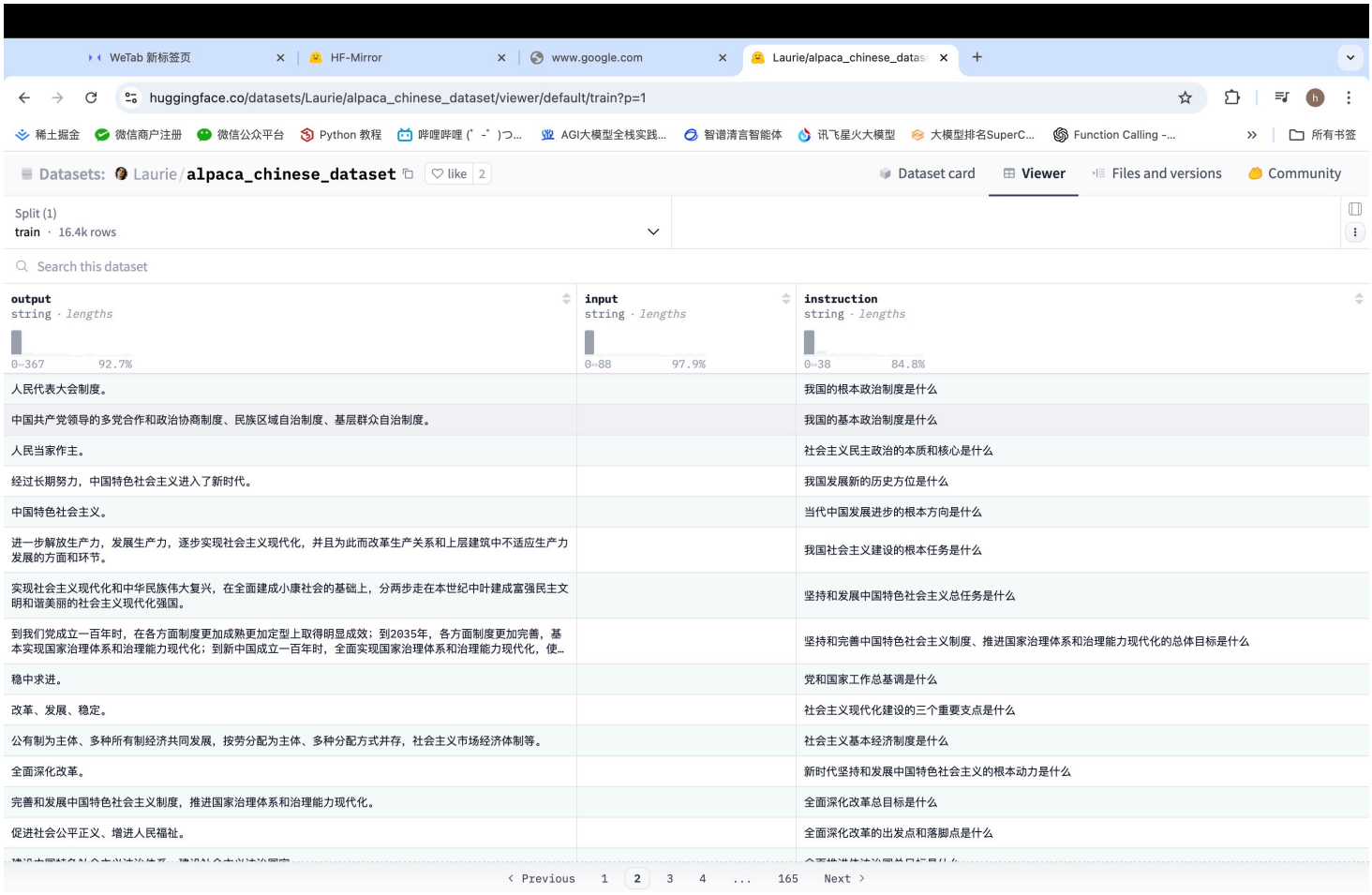
`huggingface-cli download`

它是 `huggingface-cli` 工具的下载命令，用于从 huggingface hub（`https://huggingface.co/docs/hub/index`）中去下载资源。

`Laurie/alpaca_chinese_dataset`

它是要下载的数据集的名称。网页截图如下：



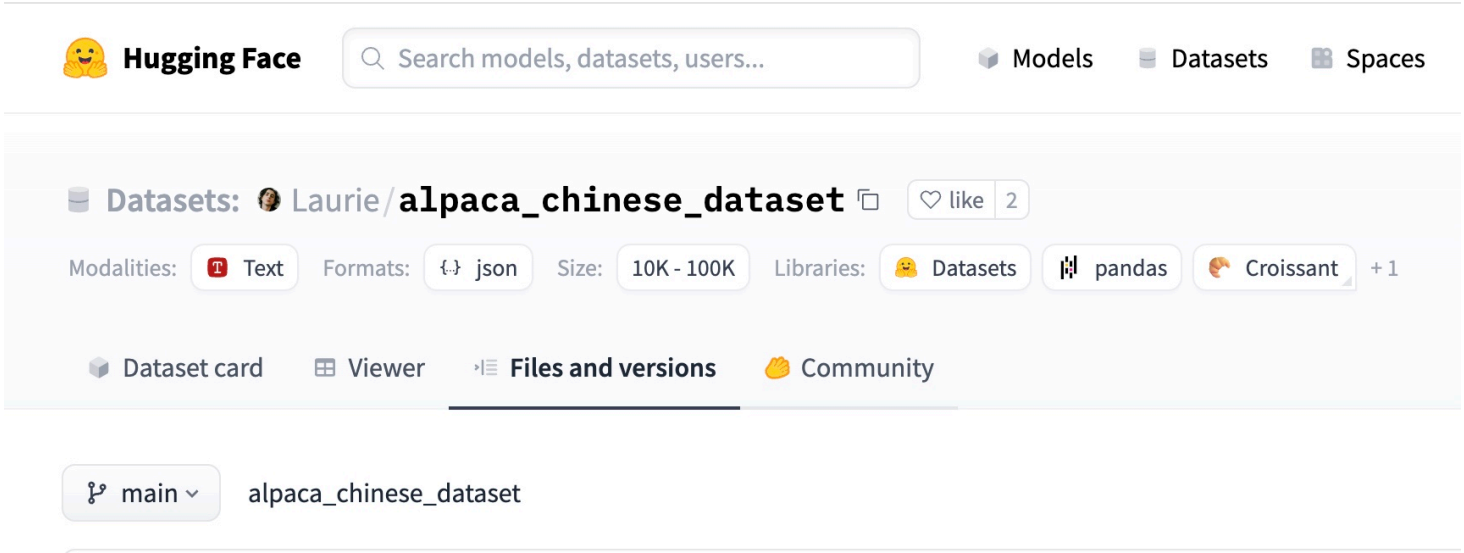


`--repo-type dataset`

指定要下载的是 数据集，而不是别的类型（比如模型或其他）。除了上面指定的 `dataset` 之外，还可以是 `model` 和 `space`.

`--revision main`

指定要下载的资源的版本号。比如下图中的 `main` 。



`--local-dir-use-symlinks False`

这个选项控制是否在本地目录中使用符号链接（symlinks）。False 表示不使用符号链接，而是

将数据集直接下载到本地目录中。符号链接是一种文件系统功能，它允许你创建指向另一个位置的快捷方式，False 设置会将所有文件实际下载到指定目录中。

`--local-dir /code/Laurie_alpaca_chinese_dataset`

这个选项指定了数据集在本地存储的目录。/code/Laurie_alpaca_chinese_dataset 是本地目录的路径，数据集将被下载并存储在这个目录下。指定目录是为了便于管理和访问下载的数据集。

如何下载模型

`huggingface-cli download Qwen/Qwen1.5-1.8B --resume-download --local-dir-use-symlinks False --local-dir /code/Qwen1.5-1.8B  --exclude "*.bin"`

和上面下载数据集不一样的地方是：

`--resume-download`：允许断点续传。

`--exclude "*.bin"`：下载文件时，所有扩展名为.bin的，将会被忽略。

训练过程中查看显卡情况

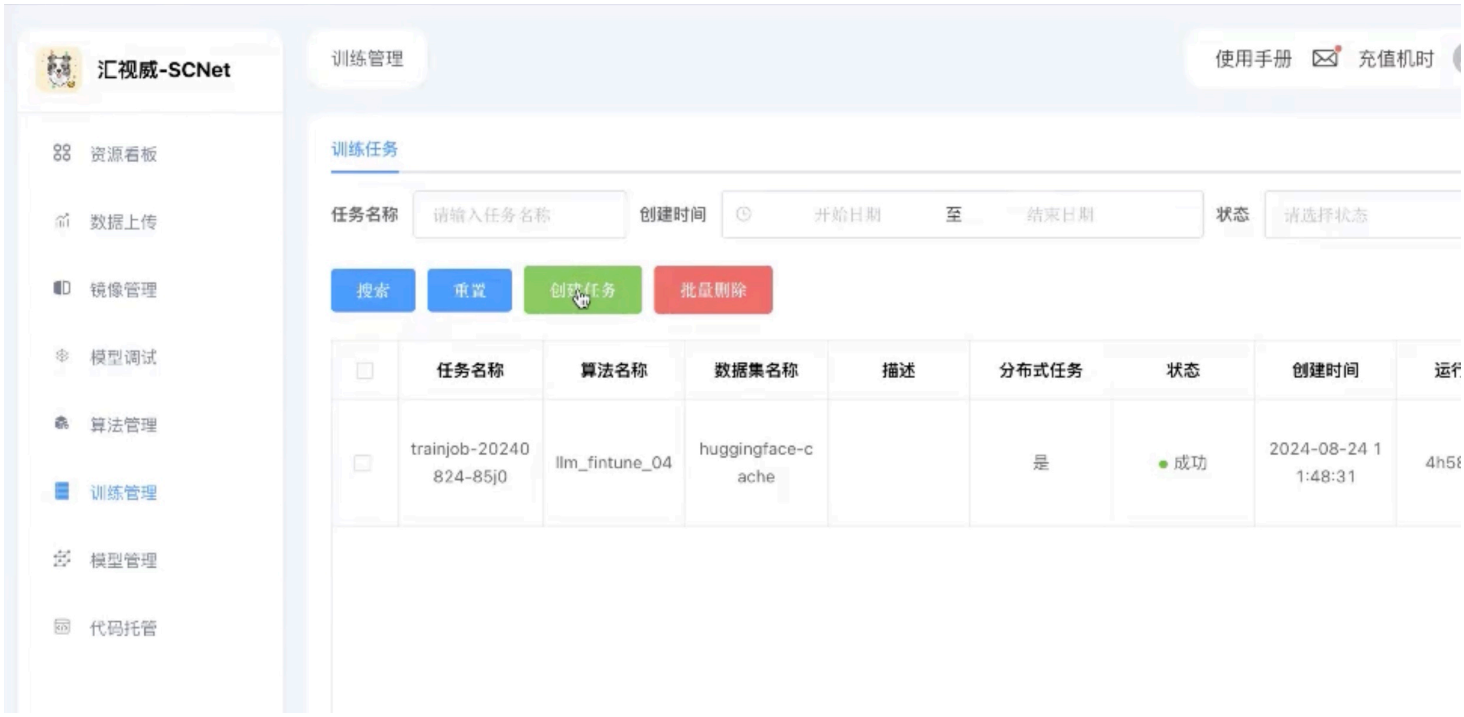
`watch -n 0.5 rocm-smi`： 周期性自动刷新训练过程中的显卡情况。

也可以单独使用 `rocm-smi` 来查看显卡情况。

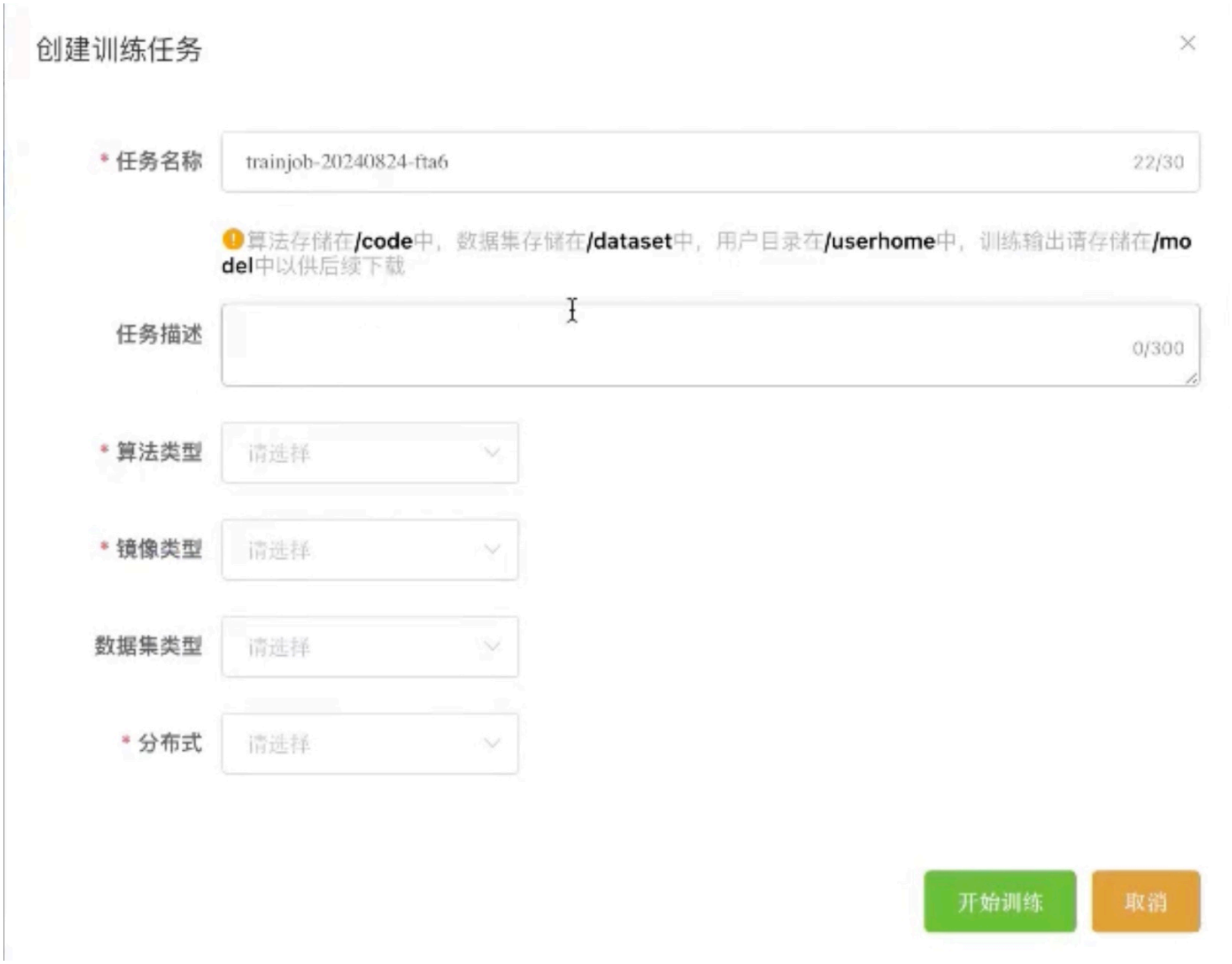
此章节录屏：
三机12卡模型训练录屏.mp4

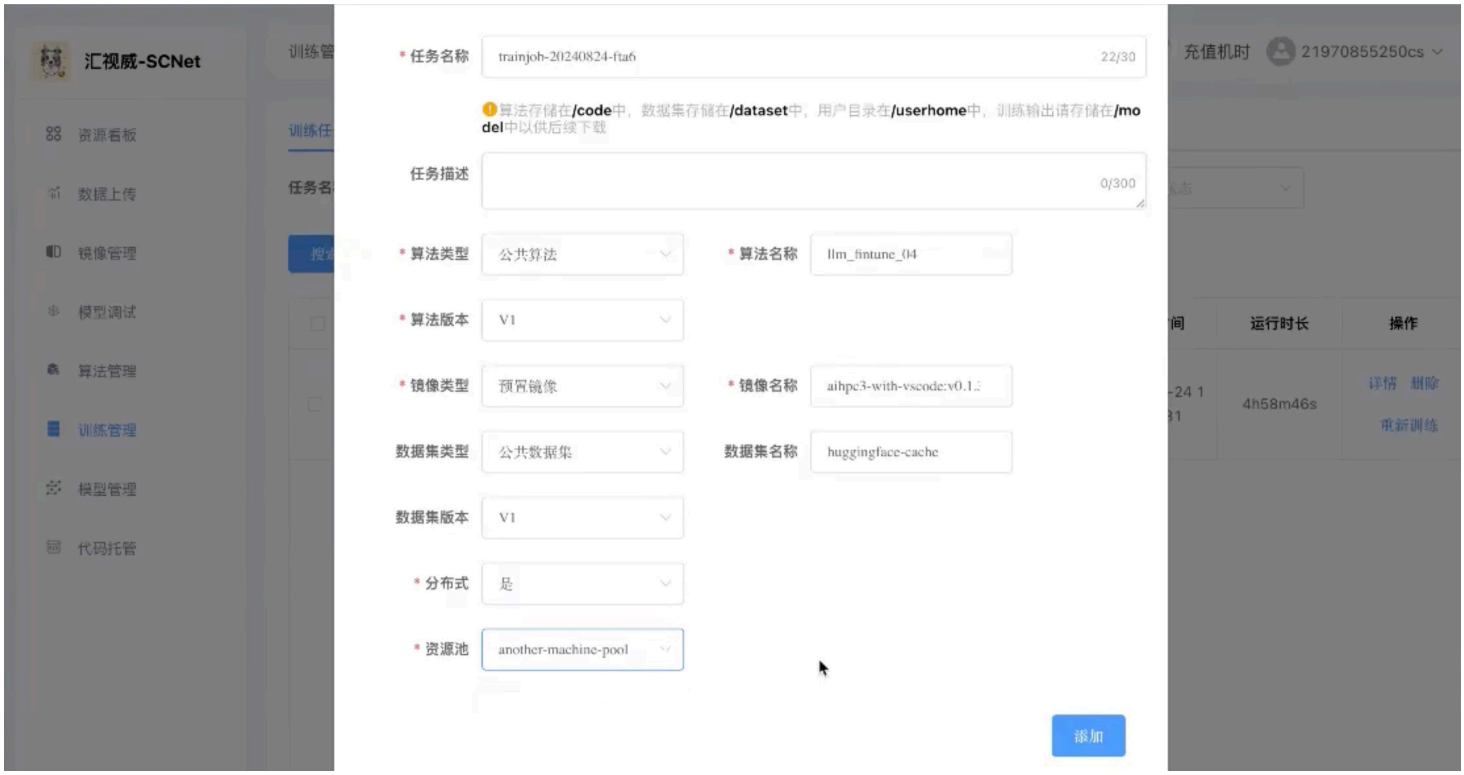
如何直接开启训练任务

点击创建任务



指定算法，镜像，数据集，分布式。





指定分布式的每个任务

主任务，下方`是否主任务`选择`是`。

下面两个是非主任务，下方`是否主任务`选择`否`。

添加分布式任务

* 任务名称 worker01

* 运行命令 bash distributed_finetune_job.sh qwen1_5_1_8b_qlora_alpaca_e3_copy.py 3

运行参数 增加 预览

* 资源规格 24核-192G-4DCU(64G

* 副本个数 1

* 最小副本成功数 1

* 最小副本失败数 1

* 是否是主任务 否

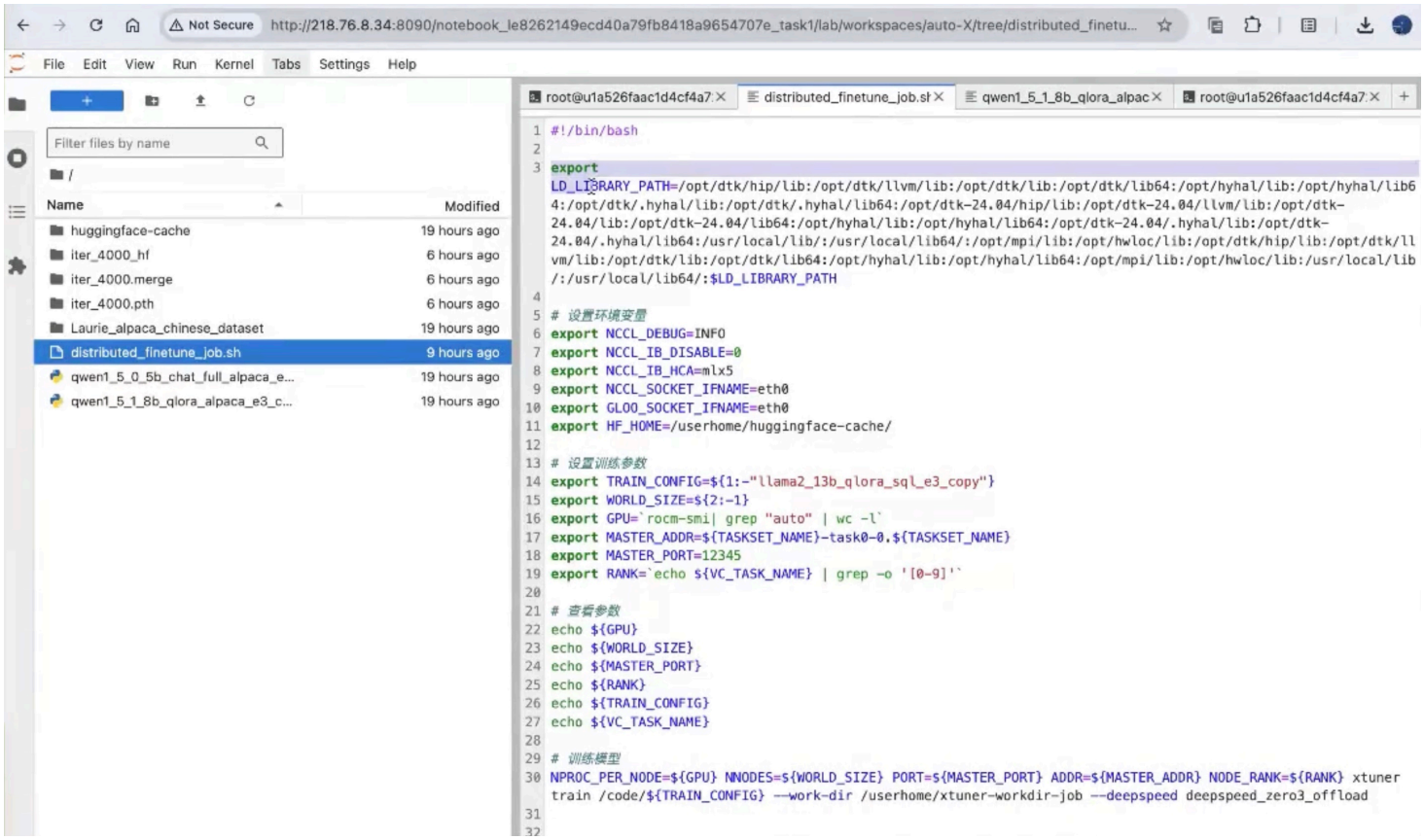
取消 确定

执行命令的解释

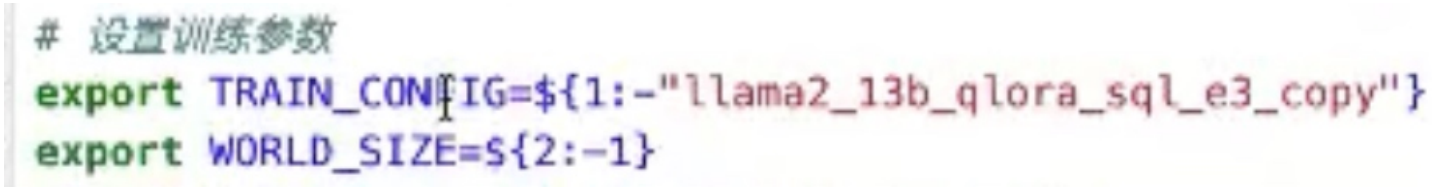
`bash distributed_finetune_job.sh qwen1_5_1_8b_qlora_alpaca_e3_copy.py 3`

`bash` 执行bash指令

`distributed_finetune_job.sh` 是一个shell文件，内容如下：



这个shell文件中的脚本，要接收两个外部参数，也就是上面的 `qwen1_5_1_8b_qlora_alpaca_e3_copy.py` 和 `3`



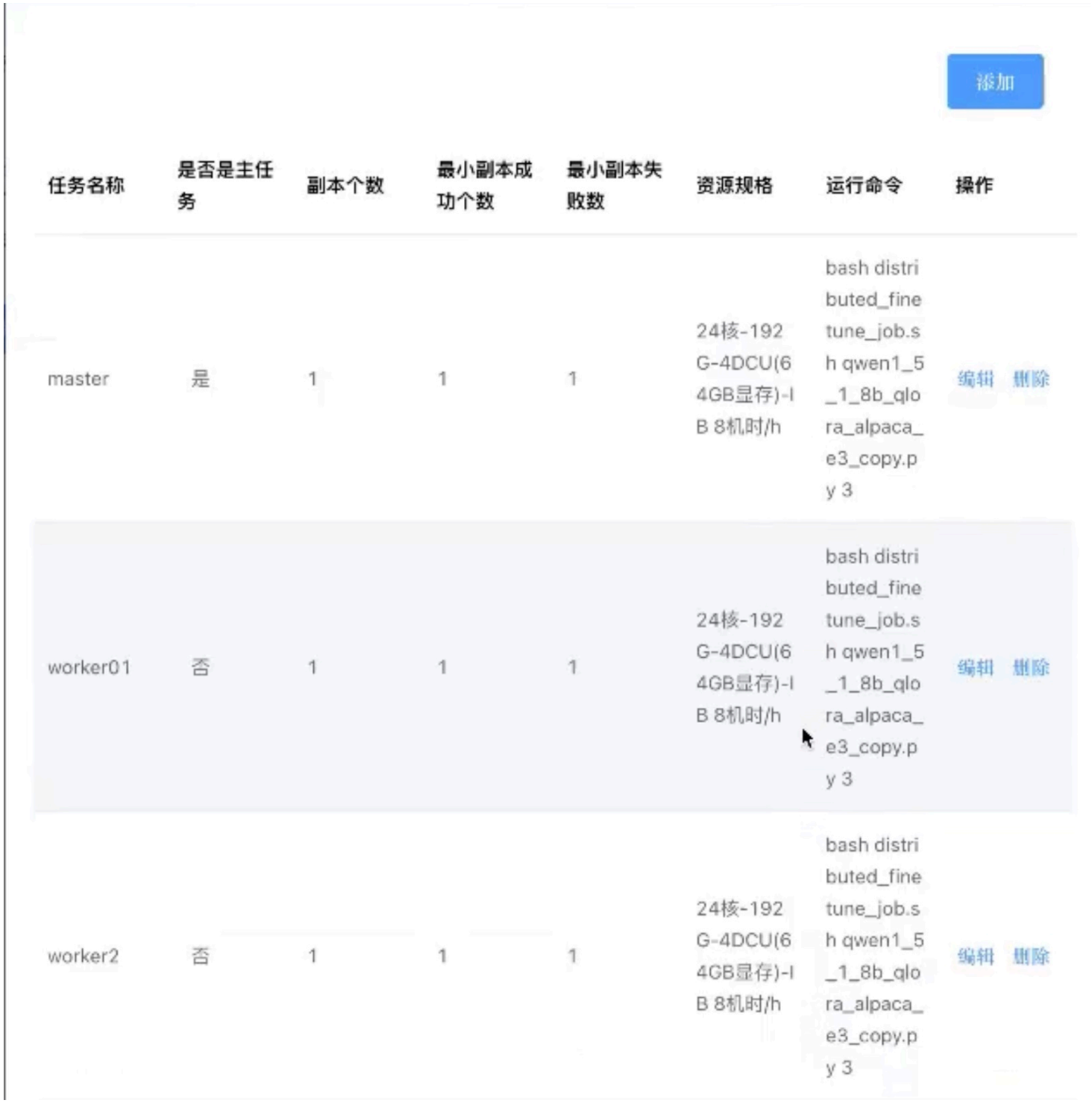
前者，指定训练算法文件：`qwen1_5_1_8b_qlora_alpaca_e3_copy.py`

后者，指定 总节点数。

其余文件，都是根据当前机器的参数自动生成或者干脆写死的。无需外部传参指定。

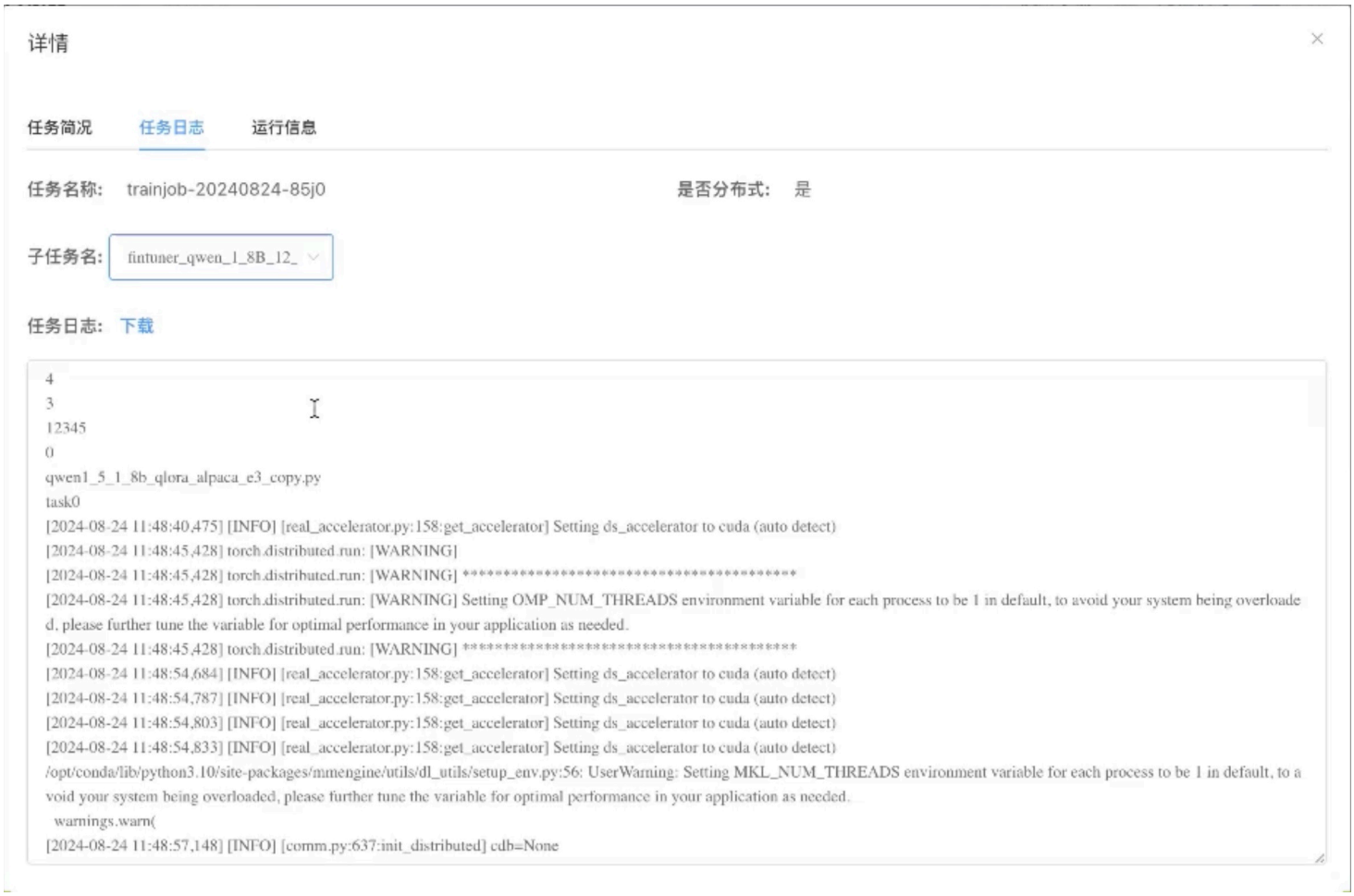
开启训练

在设置任务完成之后，点击开始训练



训练任务完成之后，点击详情，可以看到如下训练报告

一般看master节点即可。



至此，训练任务完成。

训练的成果文件，随便打开一个notebook，成果文件都保存在共享目录中。

此章节视频：
直接训练录屏.mp4