TI-ONE 训练平台

产品文档

目录

产品简介
产品概述

什么是 TI-ONE

TI-ONE 是为 AI 工程师打造的一站式机器学习平台，为用户提供从数据准备、模型训练、模型评测到模型服务部署的全流程支持。TI-ONE 支持多种训练方式和算法框架，并已全面支持 LLM 大模型的增训（Post-Pretrain）和有监督精调（SFT），满足不同 AI 场景的需求。

产品架构

核心功能

- 训练工坊：提供开发机和任务式建模两种训练方式，可基于内置镜像或自定义镜像快速、灵活发起训练任务，并基于 Angel 框架提供训练加速。其中：
  - 开发机：提供交互式的开发功能，支持 Jupyter Notebook 和 VSCode 两种在线编码 IDE，内置主流框架，支持 SSH 远程连接、Git 存储库。不仅支持算法调试与模型训练，也可以进行数据准备和预处理。
  - 任务式建模：提供向导式的训练任务提交、管理功能，特别适用于多机多卡大规模训练。基于训练任务

优先级管理以及多层容错机制，保障训练任务高效、稳定运行。

- 模型服务：支持将模型快速发布为推理服务，同时也支持离线批量预测。其中：
  - 在线服务：在一键部署之外，还支持丰富的服务管理和监控能力，包括热更新、手动/自动扩缩容、流量分配、在线测试、服务监控。

客户价值

TI-ONE 训练平台的客户价值包括技术价值和业务价值两个方面。

技术价值

- 云端的高可用 GPU 分布式集群服务器，满足大规模深度学习模型训练对性能的要求。
- 基于 GPU 的分布式机器学习平台，兼容 TensorFlow、Pytorch、PySpark 等主流开源机器学习框架，用户可在平台上灵活地定义算法模块。

业务价值

- TI-ONE 对 GPU 分布式集群服务器上的深度学习模型训练算法进行优化，能够大幅提升训练速度，从而缩短模型训练的时间。
- 使用 TI-ONE ，用户可以节省搭建机器学习平台和管理物理资源的时间，把精力聚焦在更有业务价值的建模工作上。
- 平台提供的模型一键部署功能让用户训练的模型与实际场景业务无缝对接，同时服务版本的灰度升级与流量分配功能，能帮助用户在实际的业务中灵活地进行升级与发布操作，大幅降低版本切换风险。

应用场景

TI-ONE 训练平台提供完善的框架与内置算法支持，能轻松应对各种机器学习和深度学习的定制建模的场景。以下为本产品协助各企业机构完成的一些应用场景。

金融风控

随着不法分子的作业手段日益更新，滞后的风险识别与居高不下的坏账率损失一直是各大金融机构的痛点。 TI 平台 TI-ONE 可以基于金融机构大量与风险有关的高质量数据搭建风险监控模型，提高风控的时效性、准确率和覆盖率。从贷前的额度审批、贷中的交易反欺诈到贷后的催收，覆盖各个环节，大幅减小金融机构的风险损失和管理成本。

营销推荐

如何精准触达目标消费者，提高购买转换率一直是各大商业主体都关心的问题。 TI 平台 TI-ONE 可以根据历史成交数据训练匹配模型，预测各个场景下客户和商品的最优匹配，从而实现提升营销效果、降低营销成本、挖掘潜在客户、实现交叉销售等目的。

工业质量检测

传统的工业质检依赖大量人力，成本高且漏检率难以降低。 TI 平台 TI-ONE 可以基于设备参数数据与生产图像对产品进行缺陷检测与缺陷分类，大大降低人力成本、提升缺陷检出率的同时帮助企业进行质量控制数字化管理。

算法大赛

随着人工智能行业的兴起，各类 AI 算法大赛层出不穷，如何提供满足各参赛队伍的使用习惯的工具，同时又能支撑数千人的高并发一直是各举办单位的痛点。 TI 平台 TI-ONE 内置的丰富算法与框架组件可以满足不同用户的使用习惯，高性能集群稳定性可以支持大批量的训练任务。

物业智能化管理

随着生活水平的提高，业主对物业的管理要求也日益升高，同时面临居高不下的人力成本挑战。 TI 平台 TI-ONE 基于图像识别算法，智能识别进出小区的车辆，以及所有垃圾堆放点的情况，打造智能化物业管理方式，降低人力成本、提升业主满意度。

快速入门
平台使用准备
配置账号和权限

总览

TI-ONE 训练平台使用过程需要使用其他云产品的API（例如: COS、CFS 等），因此在正式使用前，需要提前开通对应访问授权。本文档介绍主账号、子账号和对应支持的权限策略。

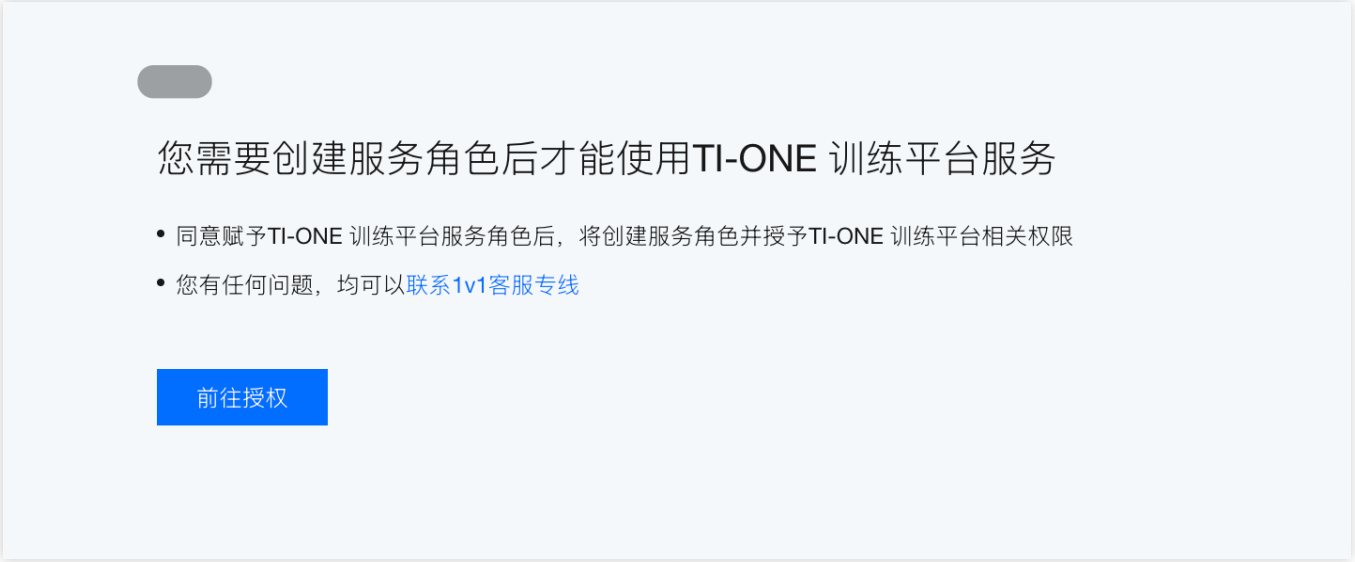
主账号授权

前提条件

> 说明
>
> 当您注册账号后，系统默认为您创建了一个主账号，用于快捷访问平台资源。

操作步骤

1. 使用 主账号登录 TI-ONE 控制台，提示需要创建服务角色权限以正常访问其他云产品资源，使 TI-ONE 正常运行。



2. 单击前往授权， 进入 CAM 控制台授权，单击同意授权，则为 TI-ONE 平台授权服务角色访问您其他云产品资源。

子账号授权

1. TI-ONE 支持子账号、协作者账号登录，主账号可以授权子账号或协作者访问管理权限。

2. 为了满足主账号便捷给子账号授予常用范围的权限，TI 平台预设了如下三个预设策略，主账号可根据对子账号的权限定位在CAM控制台进行授权。

预设策略名称	功能描述	常用场景
QcloudTIONEFullAccessContainMultiservice	TI-ONE 平台全读写访问权限，以及其他云产品包括CAM、COS、CFS、VPC、监控、标签等和下单交易权限。 拥有该策略的子账号/协作者可以完整使用TI-ONE平台的所有功能	管理员
QcloudTIONEResourceGroupFullAccessContainMultiservice	TI-ONE 资源组管理模块的读写权限，平台其他模块的只读权限，部分云产品包括CAM、COS、CFS、VPC、监控、标签等的只读权限，和下单交易权限。 拥有该策略的子账号/协作者有 TI-ONE 平台的只读权限和资源组管理模块的全读写权限	资源管理员
QcloudTIONEReadOnlyAccessContainMultiservice	TI-ONE 平台只读权限，以及其他云产品包括CAM、COS、CFS、VPC、监控、标签等的只读权限。	全平台只读

预设策略名称	功能描述	常用场景
	拥有该策略的子账号/协作者 拥有 TI-ONE 平台的只读权限	

3. 当 2 中的预设策略无法满足主账号对子账号的自定义权限管控需求时，可以使用自定义策略为子账号授权，相关操作指引请查看 CAM 策略授权使用说明，详细接口说明请查看 CAM 业务接口说明。

4. 如果您已对资源绑定了标签，并希望给子账号控制标签属性资源的访问权限，您需要通过 按标签授权 创建自定义策略。

开通关联产品

概览

使用 TI-ONE 训练平台过程中，不同的场景下会使用其他产品，如 COS、CFS、CLS、TCR 等。因此在正式使用前，需要根据场景需要提前开通对应的其他云产品并做好授权，保障后续的使用顺利进行。本文介绍各场景下依赖的其他云产品列表及权限要求。

TI-ONE 产品关联云产品

主要产品功能	关联云产品	和 TI-ONE 的关系
全平台	访问管理-CAM	账号权限控制
训练工坊	对象存储-COS	任务式建模中的代码包、训练输出存储存储模型，代码，训练数据
	CFS/TurboCFS	任务式建模的数据集 Notebook 实例中的数据或代码文件存储
	GooseFS/GooseFSx	任务式建模的数据集
	镜像服务	上传自定义镜像用于训练
	日志服务-CLS	投递日志用于日志分析，监控告警
	可观测平台-TCOP	训练任务资源使用监控告警
模型服务	CFS/TurboCFS	在线服务，将存储在 CFS 中的模型部署为在线服务
	对象存储-COS	批量预测，数据从 COS 输入、输出
	镜像服务	在线服务，上传自定义模型和运行环境镜像
	日志服务-CLS	投递日志用于日志长期归档，日志分析，监控告警
	可观测平台-TCOP	在线服务资源使用监控告警
资源组管理	弹性公网 IP	资源组节点机器访问公网资源需要，二者开通一个即可
	NAT 网关	
	云联网	TI-ONE 平台通过云联网管理资源组节点
	可观测平台-TCOP	资源组资源使用监控告警

混元大模型系列
Hunyuan-Large x TI 上手指南

总览

Hunyuan-Large 拥有 3890 亿总参数量、520 亿激活参数量，并支持 256K 上下文长度，是目前业界参数规模最大、性能领先的开源 MoE 模型。基于 MoE（Mixture of Experts）结构的优越性，Hunyuan-Large 在推理速度和参数规模之间取得平衡，显著提升了模型的处理能力。测试结果显示，Hunyuan-Large 在 CMMLU、MMLU、CEval、AGIEval 等多学科评测集以及中英文 NLP 任务、代码、数学等9大维度表现出色，超越 Llama3 和 Mixtral 等主流开源模型。

对应您使用 Hunyuan-Large 的不同场景，TI 平台能满足您的需求。

场景1：在您的大模型应用接入 Hunyuan-Large 基础模型 API

如果未经精调的 Hunyuan-Large 的效果已能满足您的需求，我们推荐您使用 Hunyuan-Large 公有 API 。您可以将 Hunyuan-Large 公有 API 接入您的大模型应用，感受 Hunyuan-Large 基础模型在您的实际应用中产生的作用。
请注意：精调后的专属 Hunyuan-Large 大模型 API 需通过 TI 平台发布。

场景2：基于自有数据精调 Hunyuan-Large 并发布为 API

您可以逐步完成 Hunyuan-Large 模型 SFT 精调 + 部署全流程，包括数据准备、模型精调训练、精调后模型直接发布为 API 等步骤。
基于 TI 平台精调出的专属 Hunyuan-Large 大模型，可以在平台内快速发布为 API，以供您接入应用增强生产力。

传统 AI 模型系列
使用任务式建模构建手写体分类模型

任务式建模简介

任务式建模提供通过向导式的训练任务提交方式进行模型构建，支持基于多种算法来源进行训练任务提交，可直接通过代码包绑定主流训练框架启动训练任务，快速使用主流高性能及分布式训练框架提交训练任务。下面将由一个简单的 PyTorch MPIJob 演示如何使用任务式建模快速创建任务。

数据准备

数据集

本案例使用 mnist 数据集，下载地址为 数据集。

代码包

本案例的训练脚本是使用 PyTorch 框架撰写的，代码包下载地址为 代码包。

上传数据到 COS 对象存储中

您可以进入COS 控制台，在存储桶列表页面创建存储桶，详情请参考创建存储桶。
创建好的存储桶将用于平台任务数据的存放，包括数据集、代码包等，文件上传操作详情请参考上传对象。

> 注意
>
> 1. COS 为对象存储产品，独立计费，详细可见对象存储-计费概述。
> 2. 创建 COS 存储桶时，所属地域需跟训练任务所在的地域一致。

操作步骤

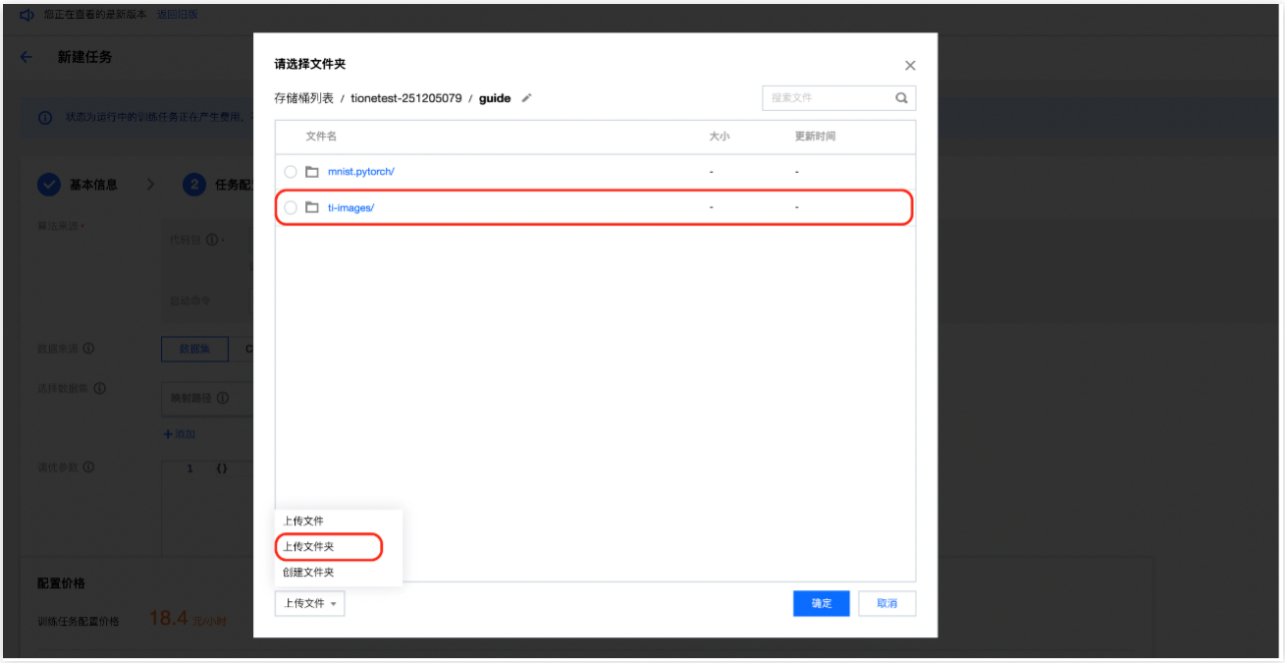
新建任务第一步

1. 进入训练工坊 > 任务式建模，单击新建，开始进入向导式训练任务创建。
2. 在基本信息页，填写如下信息：
    - 任务名称：mnist_train
    - 训练镜像选择：内置镜像 / PyTorch / torch1.9-py3.8-cuda11.1-gpu
    - 训练模式：MPI
    - 算力规格：8C40G V100*1
    - 节点数量：1个
    - 标签和描述：无需填写

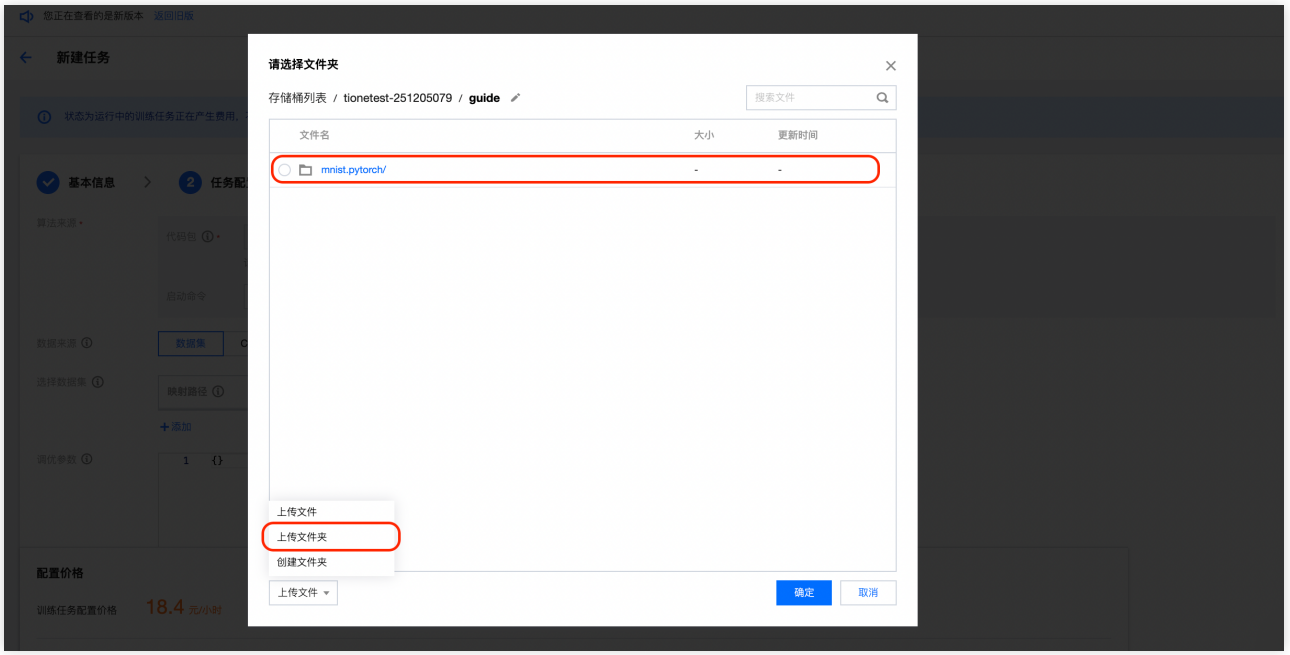
新建任务第二步

在任务配置页，填写如下信息：

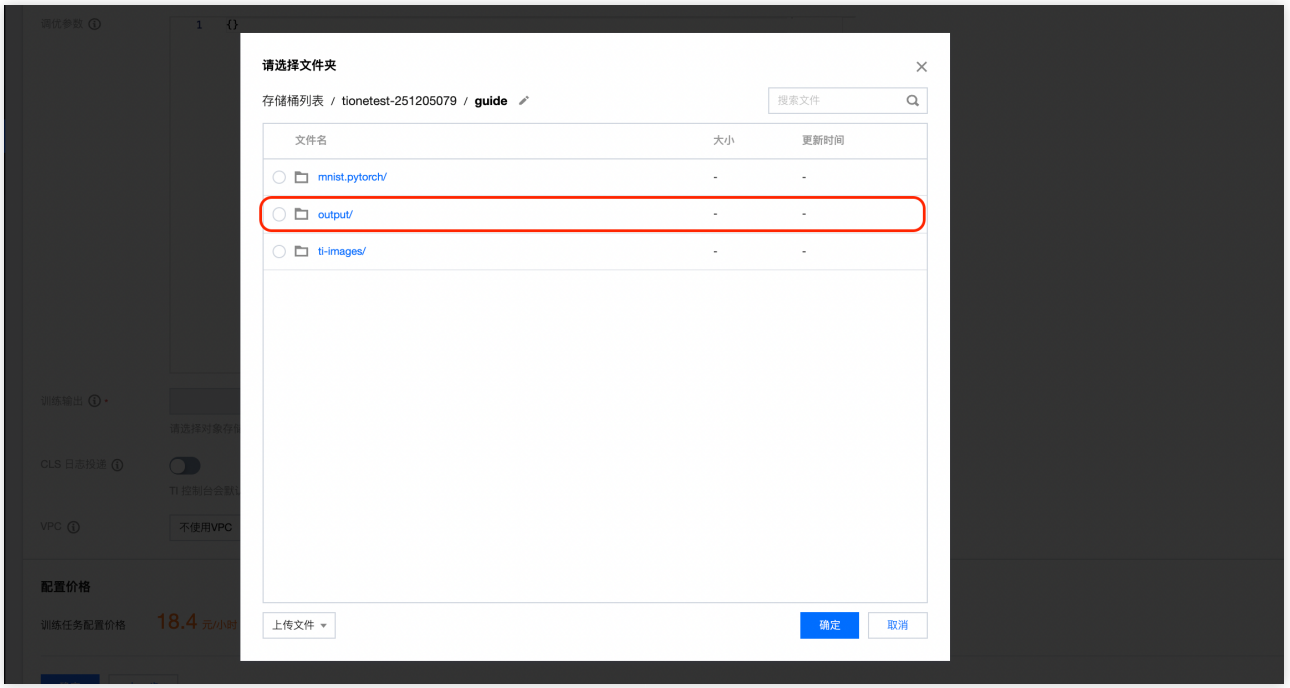
1. 数据配置：选择 COS 数据
   - 本地存储路径：填写 train
   - 数据所在路径：单击选择文件，在弹出的COS对话框中，选择需要使用的存储桶，单击左下方上传文件夹，将数据集解压后的文件夹ti-images上传，上传完成后选中文件夹路径，如下图所示：



2. 代码包：
   单击选择文件，在弹出的 COS 对话框中，选择需要使用的存储桶，单击左下方上传文件夹将准备好的代码包（需要先解压）文件夹mnist.pytorch上传至COS存储桶中，并选定代码包所在路径。



3. 启动命令：填写 sh start.sh
4. 训练输出：单击选择文件，在弹出的COS对话框中，选择需要使用的存储桶，选择训练输出数据需要保存的路径，如下图所示：

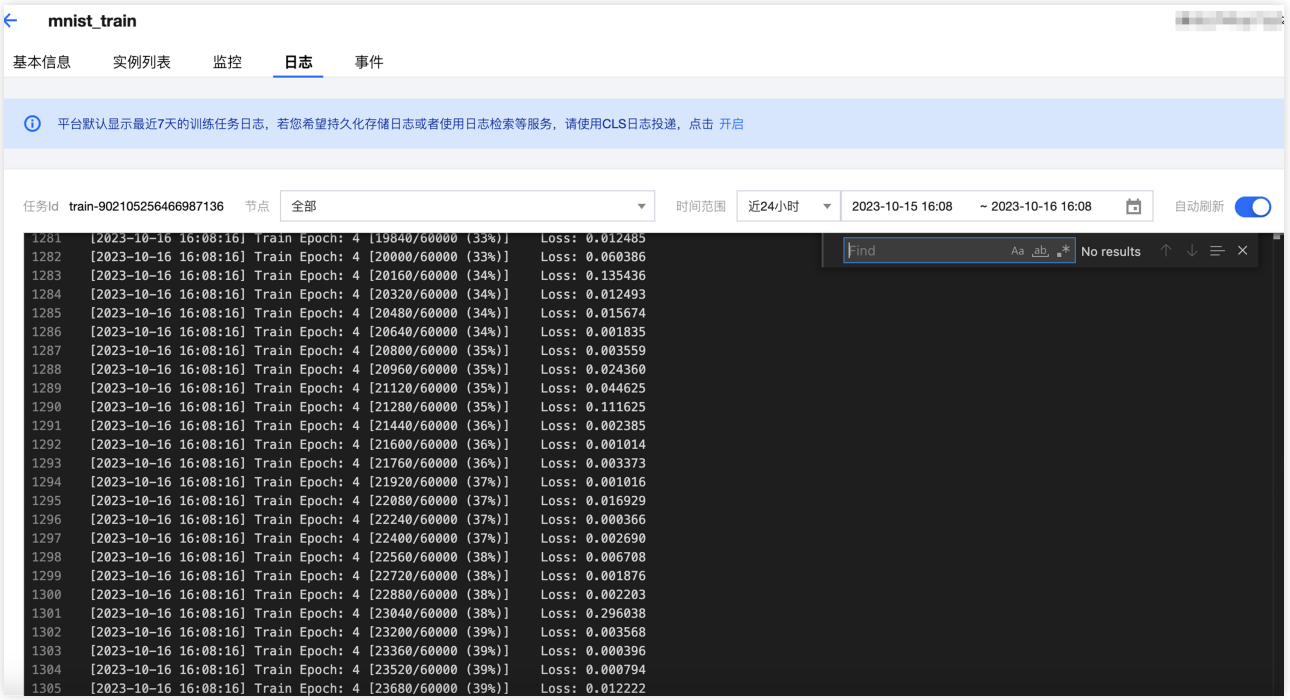


5. 调优参数：无
6. 私有化网络：无
7. CLS 日志：选择不投递

   配置完成后，可在页面底部查看本次训练任务的每小时收费价格，单击确定，即完成任务提交。

查看和监控任务

1. 提交成功后，可在任务列表页面看到任务记录。
2. 点击任务名称，可进入任务详情页查看日志和监控信息。


mnist_train
基本信息
实例列表
监控
日志
事件
平台默认显示最近7天的训练任务日志，若您希望持久化存储日志或者使用日志检索等服务，请使用CLS日志投递，点击 开启
任务Id train-902105256466987136
节点 全部
时间范围 近24小时 2023-10-15 16:08 ~ 2023-10-16 16:08
自动刷新
Find
No results
[2023-10-16 16:08:16] Train Epoch: 4 [19840/60000 (33%)] Loss: 0.012485
[2023-10-16 16:08:16] Train Epoch: 4 [20000/60000 (33%)] Loss: 0.060386
[2023-10-16 16:08:16] Train Epoch: 4 [20160/60000 (34%)] Loss: 0.135436
[2023-10-16 16:08:16] Train Epoch: 4 [20320/60000 (34%)] Loss: 0.012493
[2023-10-16 16:08:16] Train Epoch: 4 [20480/60000 (34%)] Loss: 0.015674
[2023-10-16 16:08:16] Train Epoch: 4 [20640/60000 (34%)] Loss: 0.001835
[2023-10-16 16:08:16] Train Epoch: 4 [20800/60000 (35%)] Loss: 0.003559
[2023-10-16 16:08:16] Train Epoch: 4 [20960/60000 (35%)] Loss: 0.024360
[2023-10-16 16:08:16] Train Epoch: 4 [21120/60000 (35%)] Loss: 0.044625
[2023-10-16 16:08:16] Train Epoch: 4 [21280/60000 (35%)] Loss: 0.111625
[2023-10-16 16:08:16] Train Epoch: 4 [21440/60000 (36%)] Loss: 0.002385
[2023-10-16 16:08:16] Train Epoch: 4 [21600/60000 (36%)] Loss: 0.001014
[2023-10-16 16:08:16] Train Epoch: 4 [21760/60000 (36%)] Loss: 0.003373
[2023-10-16 16:08:16] Train Epoch: 4 [21920/60000 (37%)] Loss: 0.001016
[2023-10-16 16:08:16] Train Epoch: 4 [22080/60000 (37%)] Loss: 0.016929
[2023-10-16 16:08:16] Train Epoch: 4 [22240/60000 (37%)] Loss: 0.000366
[2023-10-16 16:08:16] Train Epoch: 4 [22400/60000 (37%)] Loss: 0.002690
[2023-10-16 16:08:16] Train Epoch: 4 [22560/60000 (38%)] Loss: 0.006708
[2023-10-16 16:08:16] Train Epoch: 4 [22720/60000 (38%)] Loss: 0.001876
[2023-10-16 16:08:16] Train Epoch: 4 [22880/60000 (38%)] Loss: 0.002203
[2023-10-16 16:08:16] Train Epoch: 4 [23040/60000 (38%)] Loss: 0.296038
[2023-10-16 16:08:16] Train Epoch: 4 [23200/60000 (39%)] Loss: 0.003568
[2023-10-16 16:08:16] Train Epoch: 4 [23360/60000 (39%)] Loss: 0.000396
[2023-10-16 16:08:16] Train Epoch: 4 [23520/60000 (39%)] Loss: 0.000794
[2023-10-16 16:08:16] Train Epoch: 4 [23680/60000 (39%)] Loss: 0.012222

LLM 大模型系列
导入和部署自定义 LLM 大模型（平台内置推理镜像）

总览

本文以【Qwen2-7B-Instruct】模型为例，指导如何将自定义大模型导入到 TI-ONE 训练平台，并使用平台内置推理镜像部署大模型对话推理服务。

前置要求

申请 CFS

在导入和部署自定义 LLM 大模型中，您的大模型文件使用到的存储可以为 CFS，所以需要您首先申请 CFS。

操作步骤

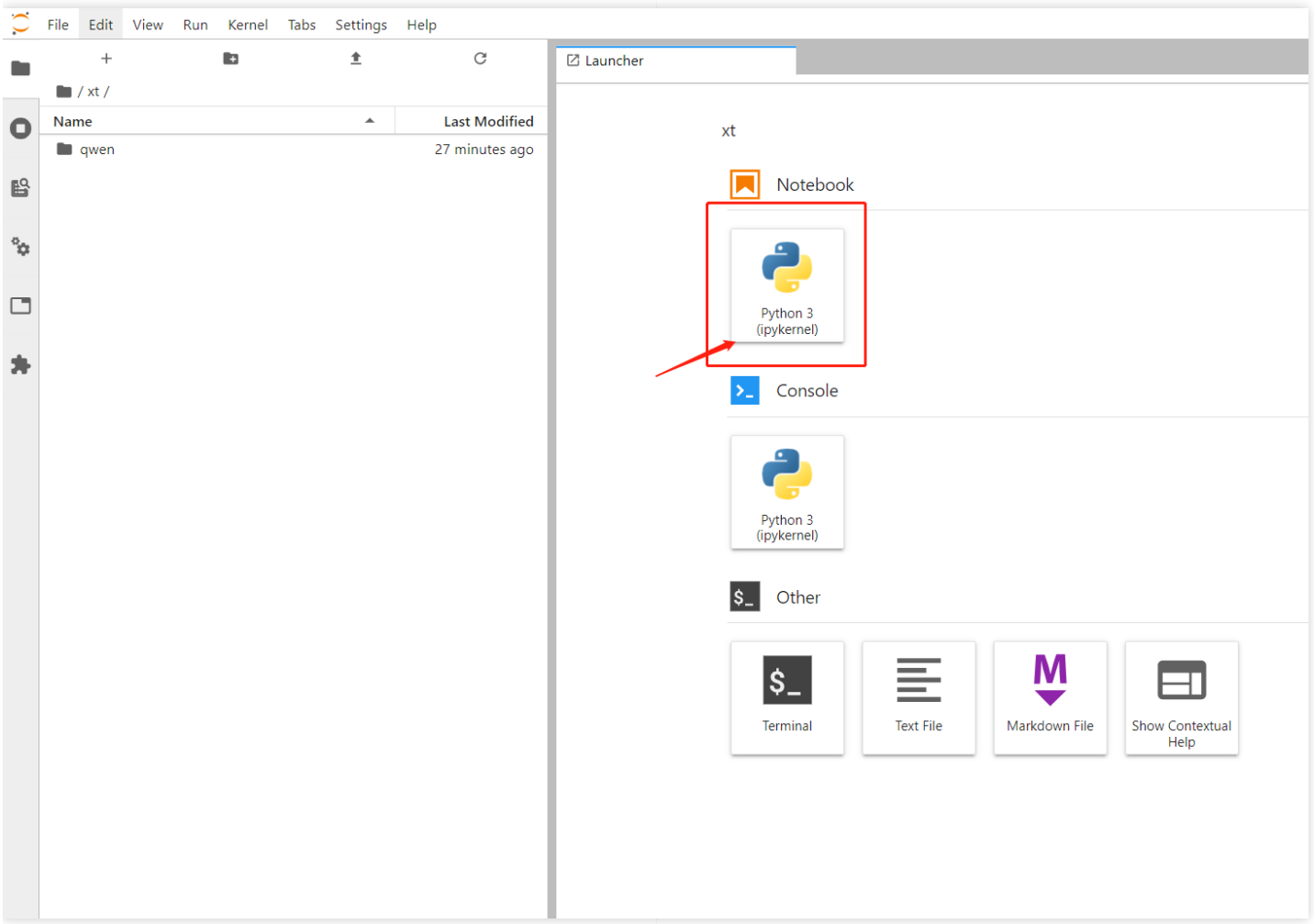
1. 上传模型文件到 CFS

登录 TI-ONE 控制台训练工坊 > 开发机，单击新建，其中各字段的填写说明如下：

- 镜像：选择任意内置镜像即可。
- **资源组和资源申请：**选择您已经创建好的资源组，并配置足够的 CPU 资源即可。
- **存储配置：**选择 CFS 文件系统，路径默认为根目录 /，用于指定保存用户自定义大模型位置。
- **其它设置：**默认不需要填写。

> 说明：
>
> 本 Notebook 实例仅用于上传或下载大模型文件。

新建成功后启动 Notebook，单击 Notebook > Python3(ipykernel) 通过脚本下载所需模型；

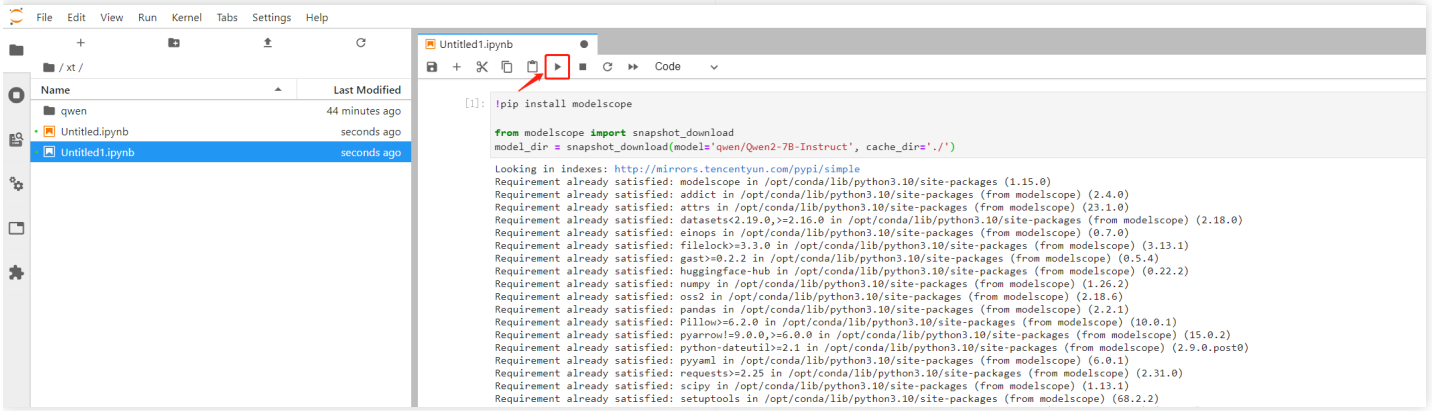


您可在魔搭社区或Hugging Face检索需要用到的大模型，通过社区中 Python 脚本自行下载模型并保存到CFS中，本文以【Qwen2-7B-Instruct】模型为例，下载代码如下：

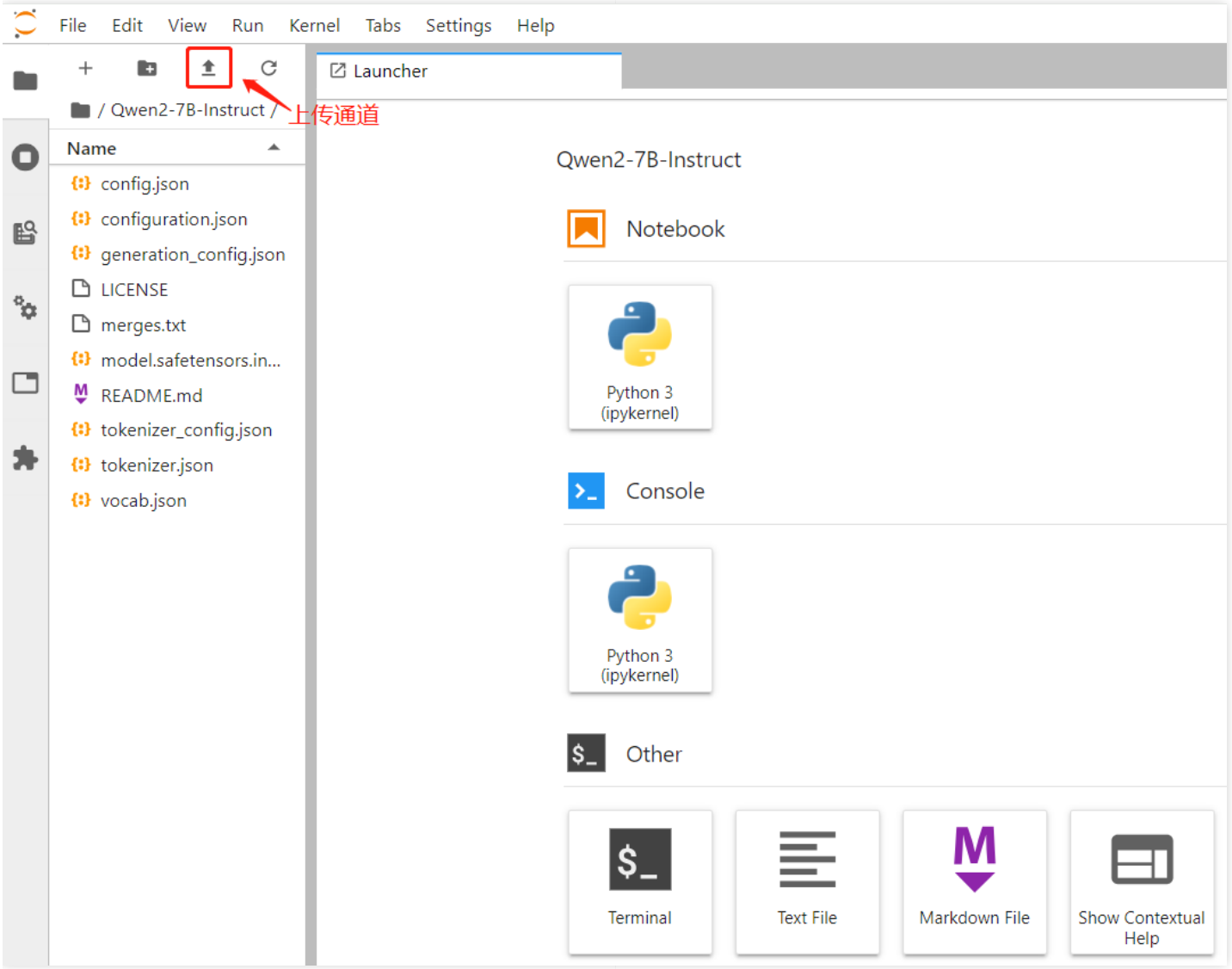
```
!pip install modelscope

from modelscope import snapshot_download
#qwen/Qwen2-7B-Instruct为需要下载的模型名称，cache_dir为下载模型保存的地址，这里'./'表示将下载模型保存在CFS的根目录中
model_dir = snapshot_download('qwen/Qwen2-7B-Instruct', cache_dir='./')
```

> 说明：
>
> 指定下载模型的地址 cache_dir（例如path/to/local/dir）后，后续在线服务 CFS 中指定模型地址为 /path/to/local/dir/qwen/Qwen2-7B-Instruct。

复制上述下载脚本并更换需要下载的模型后，粘贴到新建的 ipynb 文件中，点击运行按钮即可开始下载模型；



此外您也可以在本地下载或微调后，通过开发机上传通道将模型文件保存至 CFS 中，上传接口如图所示：



2. 创建在线服务

点击平台的模型服务 > 在线服务，单击新建服务来启动推理服务，以下是服务实例配置的指引。

- **模型来源：**选择 CFS。
- **选择模型：**指定申请的 CFS，模型路径为 CFS 中下载或上传的模型路径，此处为【/qwen/Qwen2-7B-Instruct】。
- **运行环境：**选择【内置 / LLM / angel-vllm】。
- **算力规格：**根据实际的模型大小或拥有的资源情况选择，大模型推理时需要的机器资源与模型的参数量相关，推荐按如下规则配置推理服务资源。

模型参数量	GPU 卡类型和数量
6 ~ 8B	L20 * 1 / A10 * 1 / A100 * 1 / V100 * 1
12 ~ 14B	L20 * 1 / A10 * 2 / A100 * 1 / V100 * 2
65 ~ 72B	L20 * 8 / A100 * 8

- 【高级设置 > 环境变量】：需要设置模型名称MODEL_ID(魔搭社区或Hugging Face上开源模型 ID)，以及对话模板名称CONV_TEMPLATE (若 MODEL_ID与开源模型相同，可以不添加CONV_TEMPLATE参数)，常用的对话模板名称如下表所示，本文使用qwen-chat 系列，故设置为qwen-7b-chat。

对话模板名称CONV_TEMPLATE	支持的模型系列MODEL_ID
generate	非对话模型（直接生成，无对话模板）
llama-3	llama-3-8b-instruct、llama-3-70b-instruct 模型
llama-2	llama-2-chat 系列模型
qwen-7b-chat	qwen-chat 系列模型（chatml格式）
baichuan2-chat	baichuan2-chat 系列模型
baichuan-chat	baichuan-13b-chat 模型
chatglm3	chatglm3-6b 模型
chatglm2	chatglm2-6b 模型

> 注意：
>
> - 如果对推理速度有较高要求，推荐您开启量化加速，通过环境变量 QUANTIZATION 设置，可选值有"none", "ifq", "smoothquant", "auto"。
> - none：表示关闭量化加速
> - ifq：表示开启在线 Int8 Weight-Only 量化，可以在效果基本不损失的情况下加速推理，并减少模型权重的显存占用。

    - smoothquant：表示开启 LayerwiseSearchSMQ 量化 ，可以在效果略微损失的情况下进一步加速推理（依赖提前准备量化后的模型文件，当前仅部分模型支持）
    - auto：表示自动判断量化模式，其中：
        - 若机型的显卡不支持量化，自动关闭量化。
        - 若模型目录中包含 smoothq_model-8bit-auto.safetensors 文件，会自动开启 LayerwiseSearchSMQ 量化加速。
        - 其他情况下，默认开启在线 Int8 Weight-Only 量化加速(ifq)。
- 若开启服务后日志报错 CUDA out of memory，此处由于模型max-model-len参数默认值32k较大（推理服务支持的最大上下文token数，默认为自动读取模型配置信息的上下文长度，若模型加载默认的上下文长度较大会导致显存不足），可通过环境变量 MAX_MODEL_LEN 来设置较小的数值（例如16k或8k），也可以通过开启量化加速减少模型权重所占用的显存。

3. 前端在线体验

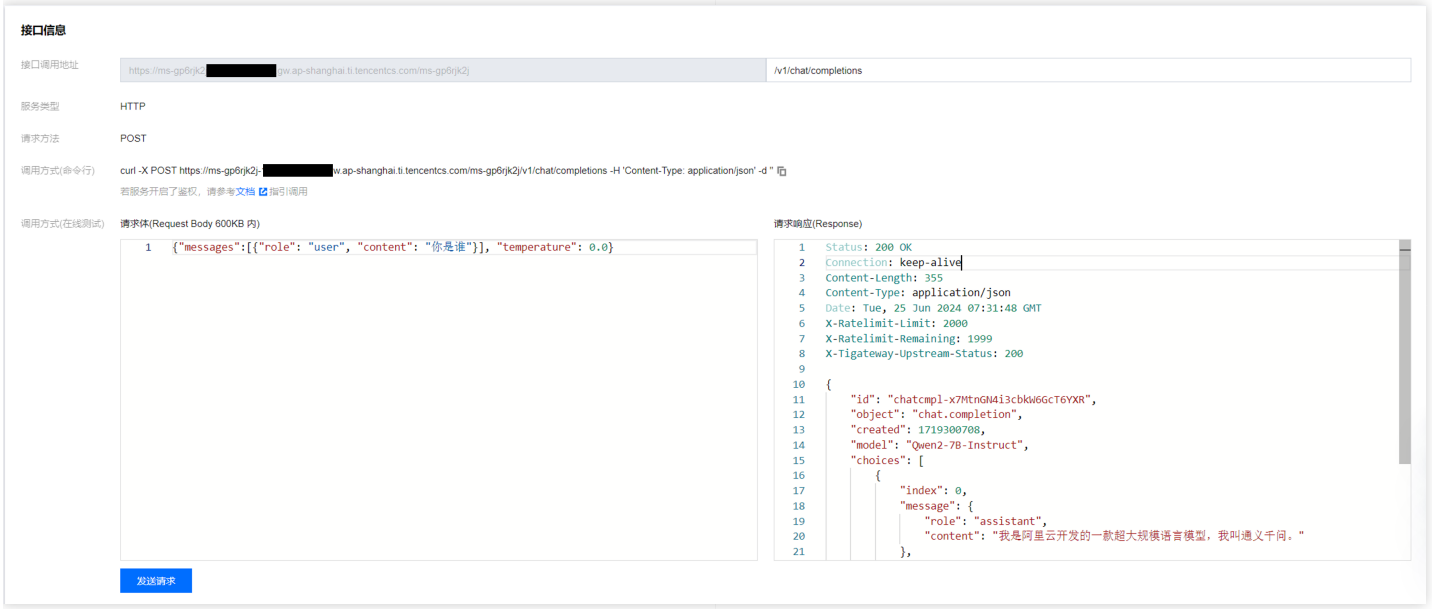
进入创建的在线服务详情，通过点击在线体验 Tab 页即可与部署的大模型进行交互体验。

4. 接口服务调用

可通过服务调用 Tab 页中的**接口信息 > 调用方式(在线测试)**进行访问，接口的调用地址为 ${SERVER_URL}/v1/chat/completions，请求体的格式：

```
{"messages":[{"role": "user", "content": "你是谁"}]}
```

字段 content 为具体的消息内容。



公网访问地址可从在线服务实例服务调用中获取，API 调用示例如下：

```
# 公网访问地址
SERVER_URL = https://ms-gp6rjk2jj-**********/ms-gp6rjk2j
```

```
# 非流式调用
curl -H "content-type: application/json" ${SERVER_URL}/v1/chat/completions -d '{"messages":[{"role": "user", "content": "你好"}], "temperature": 0.0}'
# 流式调用
curl -H "content-type: application/json" ${SERVER_URL}/v1/chat/completions -d '{"messages":[{"role": "user", "content": "你好"}], "temperature": 0.0, "stream": true}'
```

非流式返回结果：


```
{"id":"chatcmpl-4aeRgYwnaYe4RzmmcyKtYs","object":"chat.completion","created":1698291242,"model":"baichuan-13b-chat","choices":[{"index":0,"message":{"role":"assistant","content":"你好！有什么我能帮到你的吗？"},"finish_reason":"stop"}],"usage":{"prompt_tokens":4,"total_tokens":16,"completion_tokens":12}}
```


流式返回结果：


```
data: {"id": "chatcmpl-hn5mCVt4szVVZBa4fVFZWF", "model": "baichuan-13b-chat", "choices": [{"index": 0, "delta": {"role": "assistant"}, "finish_reason": null}]}

data: {"id": "chatcmpl-hn5mCVt4szVVZBa4fVFZWF", "model": "baichuan-13b-chat", "choices": [{"index": 0, "delta": {"content": " 你"}, "finish_reason": null}], "usage": {"prompt_tokens": 4, "total_tokens": 5, "completion_tokens": 1}}

data: {"id": "chatcmpl-hn5mCVt4szVVZBa4fVFZWF", "model": "baichuan-13b-chat", "choices": [{"index": 0, "delta": {"content": "好"}, "finish_reason": null}], "usage": {"prompt_tokens": 4, "total_tokens": 6, "completion_tokens": 2}}

.....此处省略中间结果.....

data: {"id": "chatcmpl-hn5mCVt4szVVZBa4fVFZWF", "model": "baichuan-13b-chat", "choices": [{"index": 0, "delta": {"content": "？"}, "finish_reason": null}], "usage": {"prompt_tokens": 4, "total_tokens": 15, "completion_tokens": 11}}

data: {"id": "chatcmpl-hn5mCVt4szVVZBa4fVFZWF", "object": "chat.completion.chunk", "created": 1714120317, "model": "baichuan-13b-chat", "choices": [{"index": 0, "delta": {}, "finish_reason": "stop"}], "usage": {"prompt_tokens": 4, "total_tokens": 16, "completion_tokens": 12}}

data: [DONE]
```


另外也可以通过 python 常用的 requests 库来使用服务，下面是一个命令行与 Qwen2-7B-Instruct 大模型推理服务进行对话交互的 Demo 示例：

```
import argparse
import requests
import json
```

```
def chat(messages):
    data = {
        "messages": messages,
        "temperature": args.temperature,
        "max_tokens": args.max_tokens,
        "top_p": args.top_p,
        "stream": True,  # 开启流式输出
    }
    header = {
        "Content-Type": "application/json",
    }
    if args.token:
        header["Authorization"] = f"Bearer {args.token}"

    response = requests.post(f"{args.server}/v1/chat/completions", json=data, headers=header, stream=
True)  # 设置 stream=True 参数获取实时数据流
    if response.status_code != 200:
        print(response.json())
        exit()

    result = ""
    print("Assistant: ", end = "", flush = True)
    for part in response.iter_lines():
        if part:
            if "content" in part.decode("utf-8"):
                content = json.loads(part.decode("utf-8")[5:])["choices"][0]["delta"]["content"]  # 字符串过滤 d
ata: 之后转成 json 格式再取文本
                result += content
                print(content, end = "", flush = True)
    print("\n")

    return result

if __name__ == "__main__":
    parser = argparse.ArgumentParser(description="Chat CLI Demo.")
    # --server替换为实际的调用地址
    parser.add_argument("--server", type=str, default="https://ms-gp6rjk2j-*******/ms-gp6rjk2j")
    parser.add_argument("--max-tokens", type=int, default=512)
    parser.add_argument("--temperature", type=float, default=0.7)
    parser.add_argument("--top_p", type=float, default=1.0)
    parser.add_argument("--system", type=str, default=None)
    parser.add_argument("--token", type=str, default=None)
    args = parser.parse_args()

    messages = []
    if args.system:
```

```
    messages.append({"role": "system", "content": args.system})

while True:
    user_input = input("User: ")
    messages.append({"role": "user", "content": user_input})
    response = chat(messages)
    messages.append({"role": "assistant", "content": response})
```

实践教程
LLM 部署及推理
快速部署和体验 DeepSeek 系列模型

总览

DeepSeek 是由深度求索公司推出的大语言模型。其中：

- DeepSeek-V3 是在14.8万亿高质量 token 上完成预训练的一个强大的混合专家 (MoE) 语言模型，拥有6710亿参数。作为通用大语言模型，其在知识问答、内容生成、智能客服等领域表现出色。
- DeepSeek-R1 是基于 DeepSeek-V3-Base 训练生成的高性能推理模型，在数学、代码生成和逻辑推断等复杂推理任务上表现优异。
- DeepSeek-R1-Distill 是使用 DeepSeek-R1 生成的样本对开源模型进行有监督微调（SFT）得到的小模型，即蒸馏模型。拥有更小参数规模，推理成本更低，基准测试同样表现出色。
  本文将介绍如何通过 TI 平台，快速部署 DeepSeek 系列模型。完成模型部署后，即可与模型进行对话体验；或以 API 形式进行调用，接入 AI 应用中。

模型列表及资源/价格参考

TI 平台已上架 DeepSeek 全系模型，详见下表。

> 注意：
>
> 1. 在部署V3或R1模型时，如仅需短时体验，对并发/上下文窗口要求不高，可使用单节点部署，部署方式请选择标准部署；其他情况下更推荐多节点部署，部署方式请选择多机分布式部署，节点数至少配置为2个。
>   - 单节点部署：最大支持64K上下文
>   - 多节点部署：最大支持128K上下文
>   - 上下文窗口：针对 DeepSeek V3/R1 模型，最大上下文窗口默认配置为16K。在部署时，可通过修改环境变量 MAX_MODEL_LEN 来进行扩展。例如：MAX_MODEL_LEN=131072 对应最大上下文窗口为128K。
> 2. 在部署 1.5B 至 70B 的 DeepSeek 蒸馏版模型时，仅需单节点，部署方式请选择标准部署。

模型名称	**参数量**	**最大上下文窗口**
DeepSeek-R1	671B	单节点 64K
		多节点 128K
DeepSeek-V3	671B	单节点 64K
		多节点 128K
DeepSeek-R1-Distill-Qwen-1.5B	1.5B	-

模型名称	**参数量**	**最大上下文窗口**
DeepSeek-R1-Distill-Qwen-7B	7B	-
DeepSeek-R1-Distill-Llama-8B	8B	128K
DeepSeek-R1-Distill-Qwen-14B	14B	-
DeepSeek-R1-Distill-Qwen-32B	32B	-
DeepSeek-R1-Distill-Llama-70B	70B	128K

模型部署实践

如果您需要部署专属的 DeepSeek 大模型推理服务，可参考如下指引进行操作。下文我们将选用尺寸相对最小的 DeepSeek-R1-Distill-Qwen-1.5B 模型进行部署实践。其他模型的操作流程类似，仅需注意算力资源的配置差异。

前置准备工作

- 模型：TI 平台已在大模型广场中内置了 DeepSeek 模型，您可直接选择模型并一键部署。
- 资源：1.5B 的 DeepSeek 模型对算力需求较小，单卡A10即可支持其推理服务。本实践也将使用该模式进行展开。

> 注意：
>
> 模型加载提示：由于 R1、V3 模型的参数量较大，其模型体积达到 641 GB，仅从平台存储加载到机器就需要相当长时间（达2小时以上），因此在模型未提前存储到机器的情况下，模型部署时间整体也会较长。

步骤一：部署模型服务

1. 登录 TI 平台 ，在大模型广场页面，您可看到 TI 平台内置的开源大模型卡片。
2. 单击进入“DeepSeek 系列模型”卡片，查看模型详细介绍。

3. 在模型详情页面，单击新建在线服务，跳转至“模型服务 > 在线服务 > 创建服务”页面配置部署参数。

4. 按页面提示填写配置信息，参考如下：

- 服务名称：输入您的自定义的服务名称。如："demo-DeepSeek-R1-Distill-Qwen-1_5B"。
- 部署方式：选择"标准部署"。
- 服务实例：
  - 模型来源：选择 "镜像"类型。
  - 模型和运行环境：选择"内置大模型/DeepSeek 系列模型/DeepSeek-R1-Distill-Qwen-1.5B"。
  - 算力规格：选择资源组

5. 单击底部启动服务，正式发起服务部署。

步骤二：体验模型效果

1. 服务部署完成后，在"模型服务 > 在线服务"页面的列表中，其状态将显示为"运行中"。DeepSeek-R1-Distill-Qwen-1.5B 模型的部署时长预计为1-2分钟。
2. 单击列表中的在线体验，进入模型快速体验页面。可通过前端页面直接提问，体验模型效果。

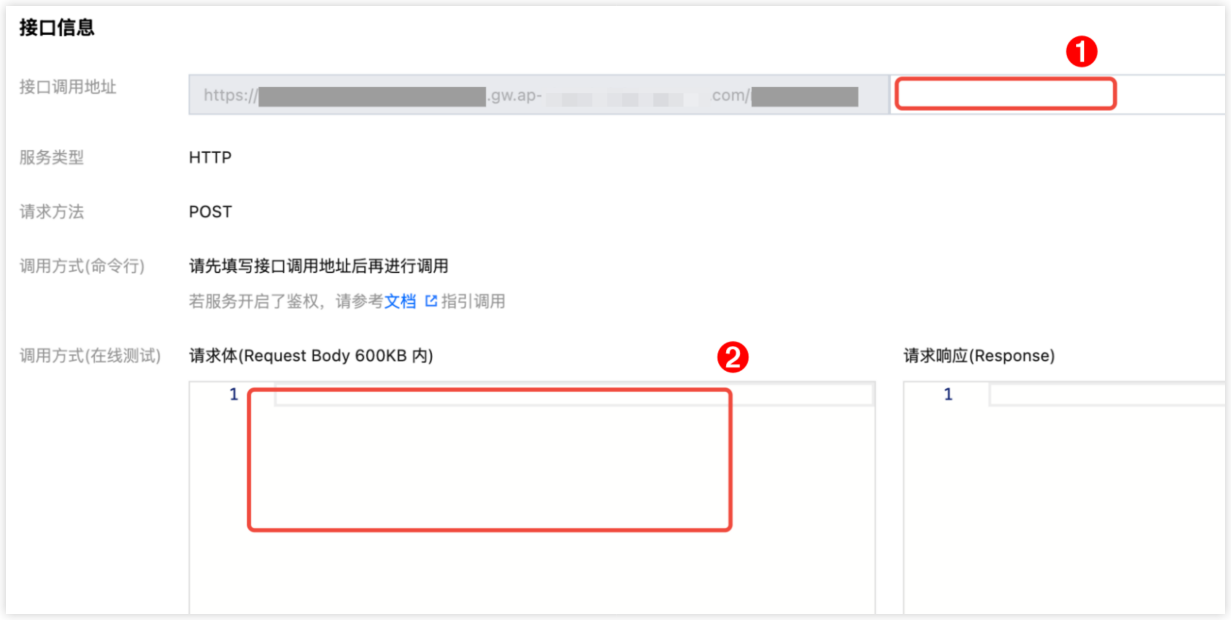
步骤三：调用模型推理 API

TI-ONE 平台在线服务模块内置了接口调用测试功能。此外，您还可以使用命令行等工具测试调用 API。测试完成后，您可以以 API 调用方式将模型接入 AI 应用。下文将对模型推理 API 的测试及接入进行示例说明。

方式一：使用 TI 平台内置工具测试 API 调用

1. 在"模型服务 > 在线服务"页面的列表中，单击刚部署的服务的名称，跳转到服务详情页。
2. 进入服务详情页的"服务调用"Tab，在页面底部可看到"接口信息"版块。

3. 在“接口信息”版块的输入框中，输入接口和请求信息，进行接口测试。



- 接口名：在上图中位置1处输入接口名，对话接口请填写 /v1/chat/completions。
  - 备注：TI 平台为 DeepSeek 大模型配备的推理框架为 sglang，兼容 OpenAI 接口规范，除对话接口以外的更多接口请参考 sglang 官方文档。
- 请求体（Request Body）：在上图中位置 2 处输入请求体，Chat Completion 接口的请求体格式请参考下方代码（请注意，下方代码中的“model”字段值“ms-xxxxxxxx”仅为示例，请在您自己的请求体中替换为真实有效的值）：

```
{
  "model": "ms-xxxxxxxx",
  "messages":
  [
```

```
        {
            "role": "user",
            "content": "描述一下你对人工智能的理解。"
        }
    ]
}
```

- 对于“model”字段，请输入服务组 ID，即页面上方“调用地址”的最后一部分。可参考下图，红框中标记的字符串即为服务组 ID，可看到该字符串以“ms-”作为前缀：

- 对于“content”字段，请输入您想对模型提出的具体问题。

4. 完成信息输入后，单击发送请求，稍作等待，“请求响应”框中将显示模型返回的响应结果：

> 注意：
>
> 按上述方法通过 TI 平台内置工具测试 API 时，受控制台统一规则约束，请求的响应时间如超过 15s 则会被判定为超时。如您遇到此类情形，可按下述方式二通过命令行测试 API 。

方式二：使用命令行工具测试 API 调用

1. 在上述的“接口信息”版块中，输入接口名。复制完整的 API 调用命令头。

2. 在命令头最后追加参数 -d'{REQ_BODY}'，得到完整命令。其中 {REQ_BODY} 为请求体，请按照上文中“使用平台在线测试功能调用 API”的第 3 点给出的格式填写。最终编写成的完整命令应如下方代码所示（如您未开启鉴权，则命令头中不会有 Authorization 参数）：

```
curl -X POST https://ms-xxxxxxxx-xxxxxxxx.gw.ap-xxxxx.ti.xxxxx.com/ms-xxxxxxxx/v1/chat/completions -H 'Authorization: xxxxxxxx' -H 'Content-Type: application/json' -d'{
    "model": "ms-xxxxxxxx",
    "messages":
    [
        {
            "role": "user",
            "content": "描述一下你对人工智能的理解。"
        }
    ]
}'
```

3. 将完整命令输入到已连接到公网的计算设备的命令行工具中并执行，命令行中将返回模型的输出。

```
[redacted] ~ % curl -X POST https://ms-[redacted].gw.ap-[redacted] com/
ms-[redacted]/v1/chat/completions  -H 'Content-Type: application/json' -d'{
    "model": "ms-[redacted]",
    "messages":
    [
        {
            "role": "user",
            "content": "描述一下你对人工智能的理解。"
        }
    ]
}'
{"id":"[redacted]","object":"chat.completion","created"[redacted],"model
":"ms-[redacted]","choices":[{"index":0,"message":{"role":"assistant","content":"<think>\n嗯，用户让我描述
一下我对人工智能的理解。首先，我得明白用户的需求是什么。可能他们是刚开始学习AI，或者是想了解AI的相关领域
。我需要从几个方面来展开，确保信息全面且易于理解。\n\n首先，定义人工智能。AI是什么，为什么被定义这么重要
？然后，人类和AI WHO Whose relationship is important。这部分应该突出AI作为人类的辅助工具和替代手段。\n\n
接下来，AI的作用领域。科学、教育、医疗、金融这些领域。用户可能有特定的领域需求，所以这些例子能帮助他们更
好地应用AI知识。\n\n然后，AI的定义。系统智能化方面，设备自适应处理任务模式，像优化系统。这一部分需要技术
术语，但又不避免解释清楚。\n\n技术意义和原理。在数据处理方面，比如无人江，强化学习的概念；用 Animated图
像来展示AI精确捕捉动物行为是什么意思。这部分要具体，让读者容易理解。\n\n应用前景，KolASS的应用以及量子计
算。这种方式展示了AI的未来可能，吸引对科技感兴趣的用户。\n\n探索和挑战。开源社区，如PyTorch和TensorFlow
，是重要的让用户考虑的因素。用户可能想知道AI的发展趋势在哪里。\n\n总结部分，AI的人机对弈关系的走向和AI伦
理。这些都是我们需要强调的，确保内容全面。\n\n最后，检查是否有遗漏的领域，比如医疗诊断、金融监控或者农业
优化。验证这些例子是否符合用户的需求，并给出可选解决方案或建议。\n\n总的来说，我需要结构清晰，涵盖定义、
现状、未来、探索和挑战，同时提供实际的应用例子和技术细节，确保内容全面且易于理解和应用。\n</think>\n\n人
工智能（AI）是一个强大的技术领域，它不仅模拟了人类思考和社交活动的特点，还具有强大的潜力。从基础到应用，
```

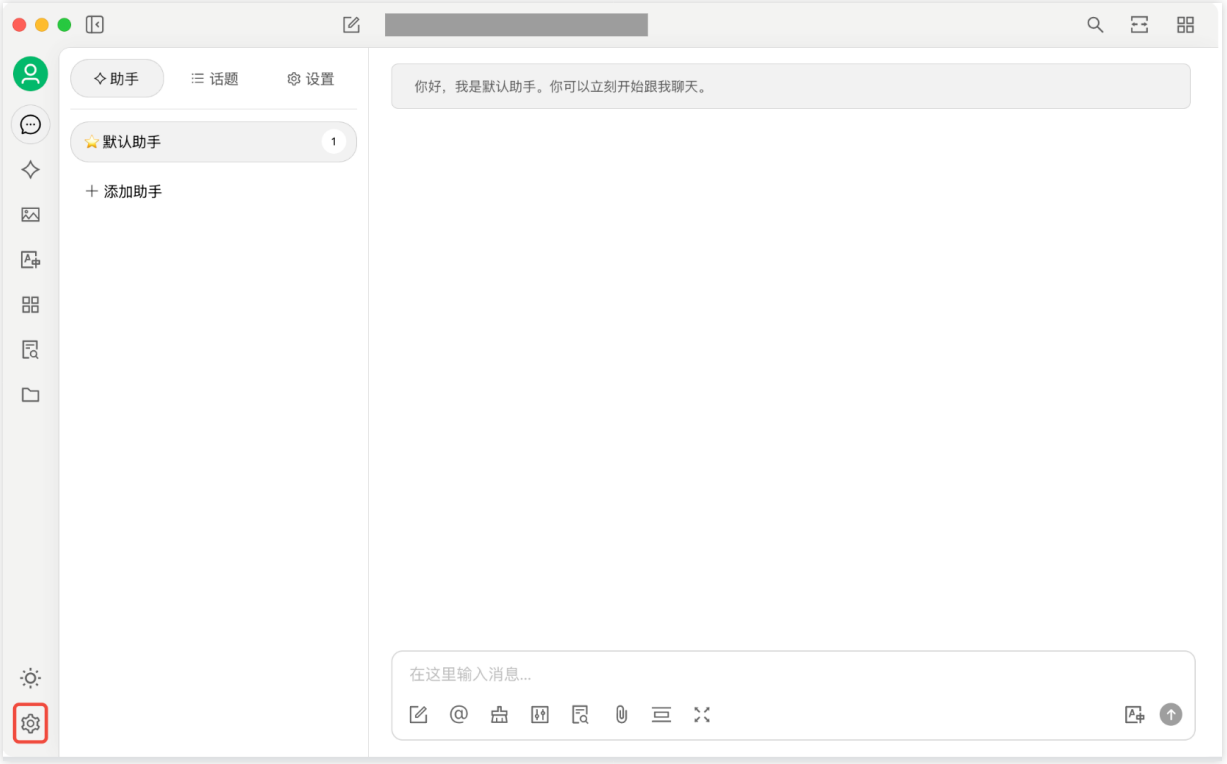
方式三：使用第三方应用开发工具调用 API

完成模型部署后，如果您需要在您的 AI 应用中接入已部署的模型服务，可以将服务 API 的信息配置到相关平台或系统中。下文以Cherry Studio为例，介绍如何将服务 API 接入应用中。

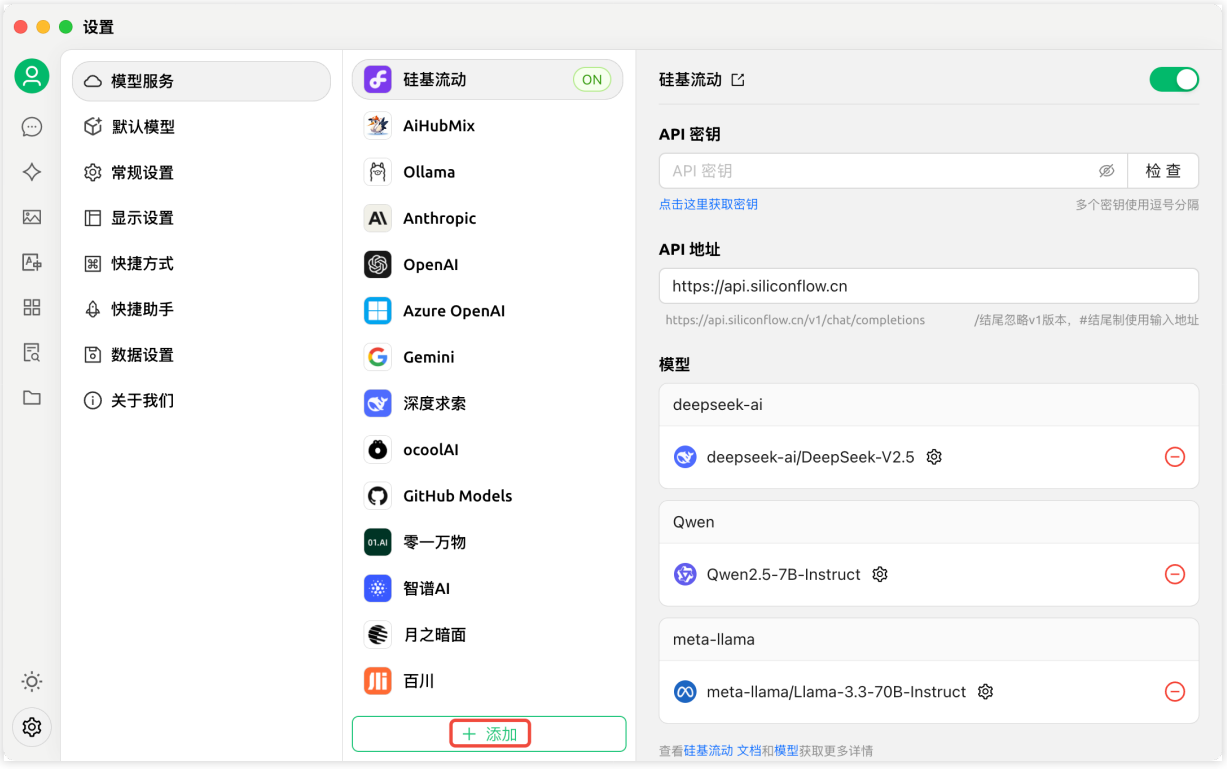
Cherry Studio 是一个支持多模型服务的开源桌面客户端，可以将多服务集成至桌面 AI 对话应用中。本文仅以此为例介绍 API 调用。如您需要商用 Cherry Studio，请仔细阅读其开源软件协议。

1. 进入您在 TI 平台已部署模型服务的“服务详情页 > 服务调用”Tab，在页面较上方位置找到“调用地址”字段，并单击最右侧复制按钮复制。

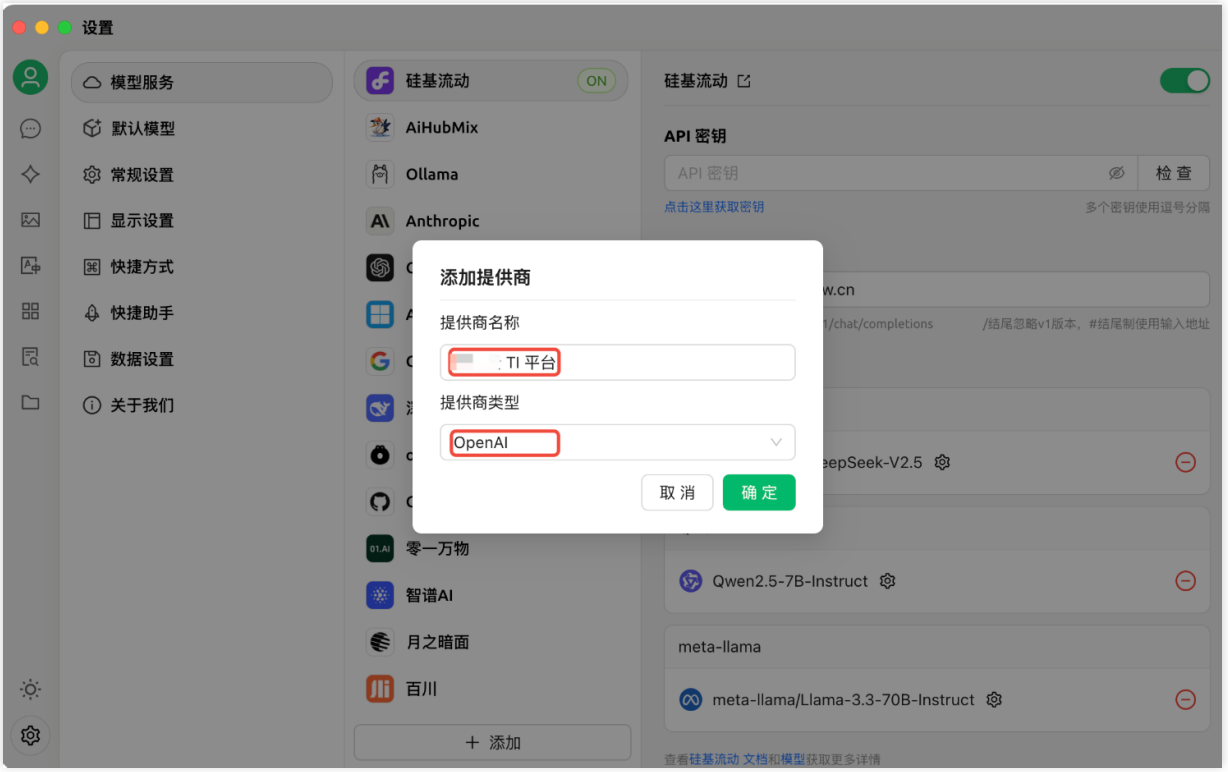
2. 下载并安装 Cherry Studio。完成安装后，打开 Cherry Studio，进入产品主页，并单击左下角的设置按钮，跳转到产品设置页。



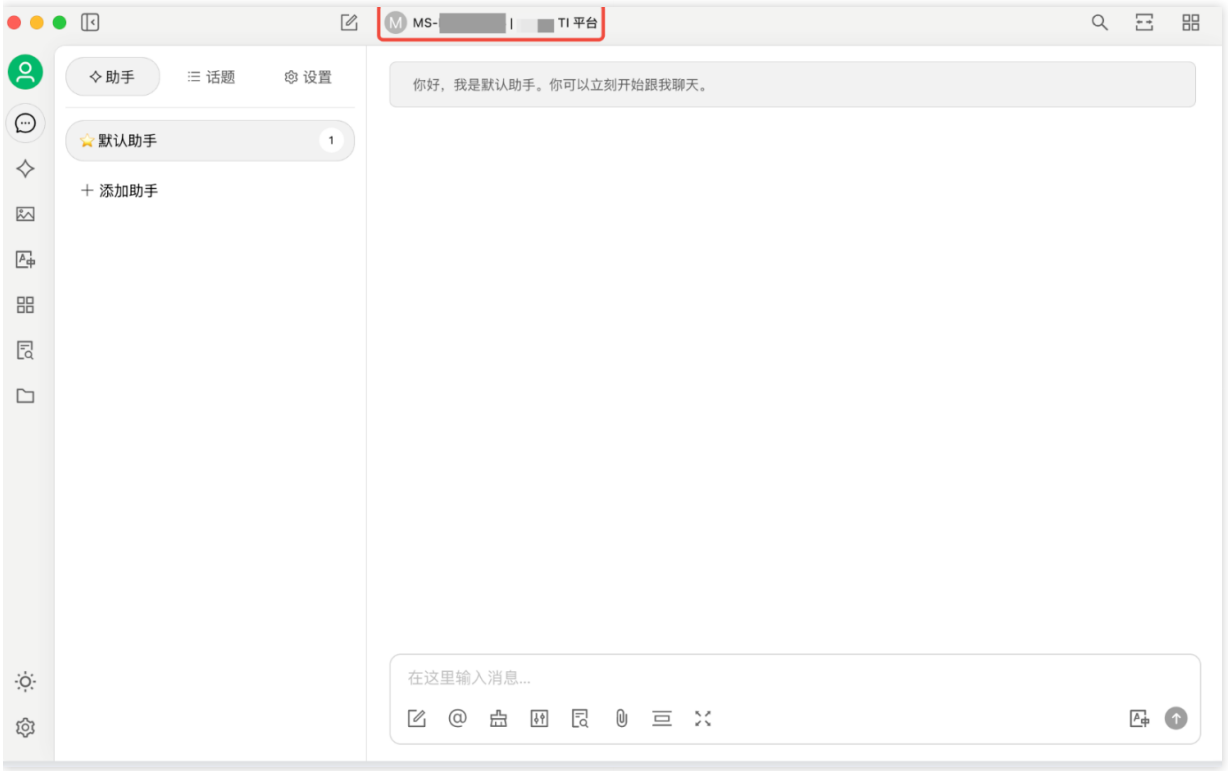
3. 进入产品设置页后，单击页面中间下方的添加：

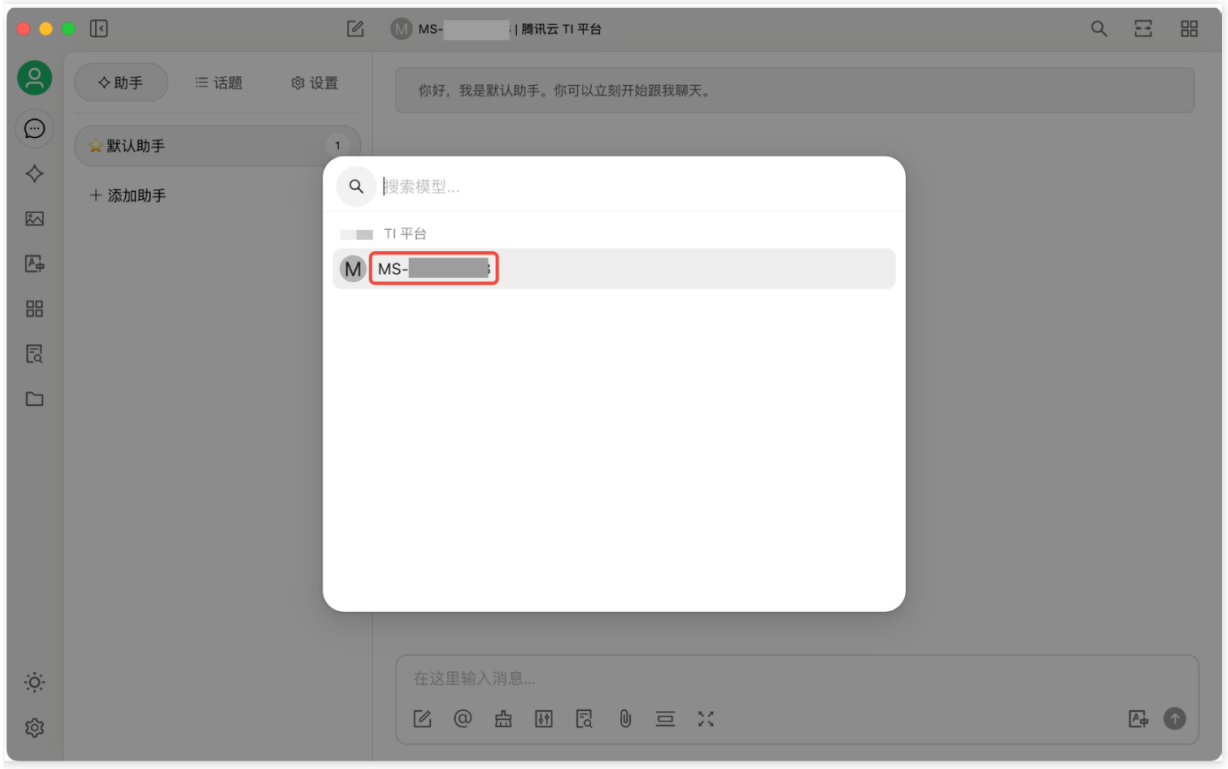


4. 单击添加后，需要在弹出的“添加提供商”对话框中输入信息，其中，“提供商名称”可自由填写，提供商类型需选择“OpenAI”，填好后单击确定：



5. 按照第4点要求成功添加提供商后，将自动跳转到该提供商的配置菜单，本文中示例为“TI 平台”。在菜单中进行如下配置：
   - API 密钥：
     - 已部署服务开启了鉴权：进入服务的“服务详情页 > 服务调用”Tab，在页面较上方位置找到“AuthToken”字段，复制字段值并粘贴到此处。
     - 已部署服务未开启鉴权：可任意填写，但不可以不填。
   - API 地址：粘贴第1点中复制的调用地址。
   - 配置完成后，单击下方“模型”版块的添加。
6. 单击添加后，将弹出“添加模型”对话框，在对话框中输入要求的信息。其中，模型 ID 需配置为 TI 平台已部署服务的服务组 ID（获取方式见“使用平台在线测试功能调用 API”第 3 点，该字段以“ms-”作为前缀），接着单击添加模型：
7. 按照第6点的要求成功添加模型后，单击左上方对话按钮，回到对话页面
8. 单击对话页面顶部的模型选择按钮，单击后弹出模型列表，选择刚刚添加的模型：





9. 现在，可以通过 Cherry Studio 和在 TI 平台部署的模型进行对话了：

步骤四：管理推理服务

您可以通过访问"模型服务 > 在线服务 > 服务详情"页面查看并管理推理服务。包括但不限于：停止/重启/删除服务、查看服务配置信息、实例列表、监控图表、容器事件、日志、更新记录等。详细操作指引可参考在线服务运营。

不同模型部署的注意事项

对于 DeepSeek 的其他模型，部署流程与前述类似，主要区别在于填写服务参数时配置的资源规格。

大小模型的效果对比

基于已部署的"DeepSeek-R1-Distill-Qwen-1.5B"和"DeepSeek-R1"模型服务，我们尝试使用一个相同的问题，简要对比一下大小模型的推理效果。
拥有更大参数量的 DeepSeek-R1 模型在推理效果上更胜一筹，其正确推理出了杯子倒扣时球会掉出并留在床上，即使杯子随后被移动至房间。而参数量较小的 DeepSeek-R1-Distill-Qwen-1.5B 模型仍然认为球在杯中。
另一方面，相比 DeepSeek-R1 模型，更小参数的 DeepSeek-R1-Distill-Qwen-1.5B 模型的响应速度更快、占用资源更少、部署时长更短，在处理较为简单的任务时，仍是不错的选择。
其中，DeepSeek-R1-Distill-Qwen-1.5B 的部署时长预计为1-2分钟，DeepSeek-R1 预计为9-10分钟（模型需预加载到节点的本地数据盘中）。

使用 TensorRT-LLM 进行推理加速

总览

本文以 Baichuan2-13B-Chat 模型为例，展示如何将一个 LLM 使用 TensorRT-LLM 做推理加速并部署。

TensorRT-LLM 介绍

TensorRT-LLM 是一款由 NVIDIA 推出的大语言模型（LLMs）推理加速框架，为用户提供了一个易于使用的 Python API，并使用最新的优化技术将大型语言模型构建为 TensorRT 引擎文件，以便在 NVIDIA GPU 上高效地进行推理。

TensorRT-LLM 也提供了支持被 NVIDIA Triton Inference Server 集成的后端，用于将模型部署成在线推理服务，并且支持 In-Flight Batching 技术，可以显著提升推理服务的吞吐率并降低时延。

TensorRT-LLM 模型转换

创建模型转换开发机

您可以拉取 TI-ONE 提供的 TensorRT-LLM 镜像，并保存到自己的容器镜像服务个人版或企业版镜像仓库实例中：

```
MY_IMAGE="<你的仓库地址>"
docker pull tione-public-hub.xx.com/xxx/tritonserver:23.10-py3-trtllm-0.7.1
docker tag tione-public-hub.xx.com/xxx/tritonserver:23.10-py3-trtllm-0.7.1 ${MY_IMAGE}
docker push ${MY_IMAGE}
```

使用上面的自定义镜像来打开一个开发机实例，挂载已申请的 CFS 或 GooseFS 存储。请注意这里开发机实例需要使用 1 卡推理用的 GPU 用于构建 TensorRT 引擎文件。

构建 TensorRT-LLM 模型

进入开发机后，镜像在 /workspace/TensorRT-LLM-examples 目录里已内置好了模型转换的示例代码，可以按示例进行操作：

1. 下载 Baichuan2-13B-Chat 模型

   您可以自行下载模型保存到 CFS 的路径中，这里提供一个参考方式：

```
apt update && apt install git-lfs
git lfs install
GIT_LFS_SKIP_SMUDGE=1 git clone https://www.modelscope.cn/baichuan-inc/Baichuan2-13B-Chat.git
cd Baichuan2-13B-Chat
git lfs pull
```

2. 按注释指引修改 build_triton_repo_baichuan2_13b.sh 文件的内容，然后执行该脚本：

```
#!/bin/bash
set -ex
# 指定模型并行数
TP=1
# 【请修改】指定原始 huggingface 模型本地目录
HF_MODEL=/home/tione/notebook/triton-example/hf_model/Baichuan2-13B-Chat
# 【请修改】指定 Triton 模型包输出目录（推荐cfs中新建一个目录）
TRITON_REPO=/home/tione/notebook/triton-example/triton_model/Baichuan2-13B-Chat/trt-${TP}-gpu
# 指定 TensorRT-LLM Engine 构建脚本路径
BUILD_SCRIPT=tensorrtllm_backend/tensorrt_llm/examples/baichuan/build.py

# 创建输出目录
mkdir -p ${TRITON_REPO}
cp -r tensorrtllm_backend/all_models/inflight_batcher_llm/* ${TRITON_REPO}/
# 拷贝 Tokenizer 相关文件到输出目录
cp ${HF_MODEL}/*token* ${MODEL_PATH}/tensorrt_llm/1/

# 构建 TensorRT-LLM Engine 文件，参数详见`tensorrt_llm/examples/baichuan/README.md`
# 示例1: baichuan V2 13B 参数量模型，使用 FP16，开启 in-flight batching 支持
#python3 $BUILD_SCRIPT --model_version v2_13b \
#                --model_dir ${HF_MODEL} \
#                --output_dir ${TRITON_REPO}/tensorrt_llm/1/ \
#               --world_size ${TP} \
#                --max_batch_size 32 \
#                --dtype float16 \
#                --use_gemm_plugin float16 \
#                --use_gpt_attention_plugin float16 \
#                --remove_input_padding \
#                --paged_kv_cache

# 示例2: baichuan V2 13B 参数量模型，使用 INT8 weight-only 量化，开启 in-flight batching 支持
python3 $BUILD_SCRIPT --model_version v2_13b \
```

```
        --model_dir ${HF_MODEL} \
        --output_dir ${TRITON_REPO}/tensorrt_llm/1/ \
        --world_size ${TP} \
        --max_batch_size 32 \
        --dtype float16 \
        --use_weight_only \
        --use_gemm_plugin float16 \
        --use_gpt_attention_plugin float16 \
        --remove_input_padding \
        --paged_kv_cache


# Triton config.pbtxt 配置文件修改
# options.txt 文件可以按需修改，一般推荐使用默认值
OPTIONS=options.txt
python3 tensorrtllm_backend/tools/fill_template.py -i ${TRITON_REPO}/preprocessing/config.pbt
xt ${OPTIONS}
python3 tensorrtllm_backend/tools/fill_template.py -i ${TRITON_REPO}/postprocessing/config.p
btxt ${OPTIONS}
python3 tensorrtllm_backend/tools/fill_template.py -i ${TRITON_REPO}/tensorrt_llm_bls/config.p
btxt ${OPTIONS}
python3 tensorrtllm_backend/tools/fill_template.py -i ${TRITON_REPO}/ensemble/config.pbtxt
${OPTIONS}
python3 tensorrtllm_backend/tools/fill_template.py -i ${TRITON_REPO}/tensorrt_llm/config.pbtxt
${OPTIONS}

# 建立 /data/model 的软链（TIONE在线服务中，模型默认挂载到此处）
mkdir -p /data
ln -s ${TRITON_REPO} /data/model

# 本地启动 Triton 推理服务调试
# launch_triton_server
```

转换完的模型目录结构如下

```
# tree
.
├── ensemble
│   ├── 1
│   └── config.pbtxt
├── postprocessing
│   ├── 1
│   │   └── model.py
│   └── config.pbtxt
├── preprocessing
```

```
│   ├── 1
│   │   └── model.py
│   └── config.pbtxt
├── tensorrt_llm
│   ├── 1
│   │   ├── baichuan_float16_tp1_rank0.engine
│   │   ├── config.json
│   │   ├── model.cache
│   │   ├── special_tokens_map.json
│   │   ├── tokenization_baichuan.py
│   │   ├── tokenizer_config.json
│   │   └── tokenizer.model
│   └── config.pbtxt
└── tensorrt_llm_bls
    ├── 1
    │   └── model.py
    └── config.pbtxt
```

您可以在开发机中直接执行 launch_triton_server 命令启动 Triton Inference Server，并参考 api_test.sh 进行本地调用，若您希望发布正式的推理服务并允许公网或 VPC 内调用，请参考下面的章节。

Triton Inference Server 推理服务部署

创建在线服务

创建服务时，模型来源选择 CFS 或 GooseFS，选择模型选择 CFS 或 GooseFS 上转换好的 Triton 模型包路径。
运行环境选择刚才的自定义镜像或内置镜像内置 / TRION(1.0.0) / 23.10-py3-trtllm-0.7.1。
算力资源根据实际拥有的资源情况选择，CPU 不低于 8 核，内存不小于 40 G，GPU 推荐使用 A100 或 A800。
看到类似如下日志，说明服务启动完成：

```
[2024-01-31 16:51:52] I0131 08:51:51.374852 100 tritonserver.cc:2458]
[2024-01-31 16:51:52] +----------------------------------+------------------------------------------------------------------------------------------------------------------------------
[2024-01-31 16:51:52] | Option                           | Value
[2024-01-31 16:51:52] +----------------------------------+------------------------------------------------------------------------------------------------------------------------------
[2024-01-31 16:51:52] | server_id                        | triton
[2024-01-31 16:51:52] | server_version                   | 2.39.0
[2024-01-31 16:51:52] | server_extensions                | classification sequence model_repository model_repository(unload_dependents) schedule_policy model_configuration system_shared
[2024-01-31 16:51:52] | model_repository_path[0]         | /data/model
[2024-01-31 16:51:52] | model_control_mode               | MODE_NONE
[2024-01-31 16:51:52] | strict_model_config              | 1
[2024-01-31 16:51:52] | rate_limit                       | OFF
[2024-01-31 16:51:52] | pinned_memory_pool_byte_size     | 268435456
[2024-01-31 16:51:52] | cuda_memory_pool_byte_size{0}    | 67108864
[2024-01-31 16:51:52] | min_supported_compute_capability | 6.0
[2024-01-31 16:51:52] | strict_readiness                 | 1
[2024-01-31 16:51:52] | exit_timeout                     | 30
[2024-01-31 16:51:52] | cache_enabled                    | 0
[2024-01-31 16:51:52] +----------------------------------+------------------------------------------------------------------------------------------------------------------------------
[2024-01-31 16:51:52] I0131 08:51:51.376420 100 grpc_server.cc:2513] Started GRPCInferenceService at 0.0.0.0:8001
[2024-01-31 16:51:52] I0131 08:51:51.376618 100 http_server.cc:4497] Started HTTPService at 0.0.0.0:8501
[2024-01-31 16:51:52] I0131 08:51:51.418100 100 http_server.cc:270] Started Metrics Service at 0.0.0.0:8002
```

接口调用

文本生成接口API可以参考 Triton 的文档，示例如下：

```
# 公网访问地址可从在线服务实例网页前端的【服务调用】Tab 页获取
SERVER_URL=https://service-********.sh.xxxx.com:443/tione

# 非流式调用
curl -X POST ${SERVER_URL}/v2/models/tensorrt_llm_bls/generate -d '{"text_input": "<reserved_106>你是谁？<reserved_107>", "max_tokens": 64}'

# 流式调用
curl -X POST ${SERVER_URL}/v2/models/tensorrt_llm_bls/generate_stream -d '{"text_input": "<reserved_106>你是谁？<reserved_107>", "max_tokens": 64, "stream": true}'
```

非流式返回结果：

{"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":[0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0
ext_output":"我是百川大模型，是由百川智能的工程师们创造的大语言模型，我可以和人类进行自然交流、解答问题、协助创作，帮助大众轻松、普惠的获得世界知识和专业服务。如果你有任何问题，可以随时向我提问"}

流式返回结果：

data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":0.0,"text_output":"我是"}
data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":[0.0,0.0],"text_output":"我是百川"}
data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":[0.0,0.0,0.0],"text_output":"我是百川大"}
data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":[0.0,0.0,0.0,0.0],"text_output":"我是百川大模型"}

data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":[0.0,0.0,0.0,0.0,0.0],"text_output":"我是百川大模型，"}

... 省略多行 ...

data: {"cum_log_probs":0.0,"model_name":"tensorrt_llm_bls","model_version":"1","output_log_probs":
[0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0
ext_output":"我是百川大模型，是由百川智能的工程师们创造的大语言模型，我可以和人类进行自然交流、解答问题、协助创作，帮助大众轻松、普惠的获得世界知识和专业服务。如果你有任何问题，可以随时向我提问"}

大模型推理所需资源指南

本文旨在介绍 TI-ONE 训练平台进行大模型推理时，可保障模型正常运行的配置资源，仅供您参考。

内置大模型的推理资源指南

> 注意：
>
> 在部署 DeepSeek V3 或R1模型时，如仅需低并发体验，可使用单节点部署；如果您对推理性能以及上下文长度有较高要求，且算力资源充足，推荐使用至少2节点部署。

内置大模型	模型清单	推理资源推荐
显存推荐		
Hunyuan-Large	hunyuan-large-chat	显存：不小于96GB*4
DeepSeek 系列模型	DeepSeek-V3	显存：不小于96GB*8
	DeepSeek-R1	显存：不小于96GB*8
	DeepSeek-V3-0324	显存：不小于96GB*8
	DeepSeek-R1-AngelACC	显存：不小于96GB*8
	DeepSeek-V3-AngelACC	显存：不小于96GB*8
	DeepSeek-V3-0324-AngelACC	显存：不小于96GB*8
	DeepSeek-R1-Distill-Qwen-1.5B	显存：不小于24GB
	DeepSeek-R1-Distill-Qwen-7B	显存：不小于24GB
	DeepSeek-R1-Distill-Qwen-14B	显存：不小于48GB
	DeepSeek-R1-Distill-Qwen-32B	显存：不小于96GB

基于内置 Angel-vLLM 镜像进行推理加速

总览

> **说明：**本文以 qwen2-7b-instruct 为例，展示如何使用TI-ONE 训练平台部署并推理加速自定义大模型，最后通过测试脚本给出加速量化性能测试使用方法；您也可以基于本文相关脚本，测试通过TI-ONE 训练平台内置推理镜像 Angel-vLLM 推理加速的效果。

Angel-vLLM 是 AI 加速团队基于开源 vLLM 深度优化的大模型推理加速框架。在保持和社区 vLLM 相同使用接口和完全兼容社区 vLLM 功能的同时，具备如下特点：

1. 功能更丰富。相比 vLLM 社区开源版本，Angel-vLLM 提供了 INT8，NF4，FP8 在线量化，lookahead 并行解码等功能。
2. 性能更强大。相比 vLLM 社区开源版本，Angel-vLLM 量化不仅节省显存，也可以降低延迟，提升吞吐。lookahead 并行解码经过实际业务打磨，在 RAG 场景提升明显。
3. 精度更对齐。Angel-vLLM 生成结果经过大量上线业务检验，可以做到和 HuggingFace 生成结果完全对齐或精度保持基本不变。
   TI-ONE 平台内置了已集成好 Angel-vLLM 推理加速框架的推理镜像，方便您一键部署兼容 openai 接口协议的 LLM 推理服务。
   本文将介绍如何在 TI-ONE 平台使用内置的 Angel-vLLM 镜像来部署 LLM 推理服务，开启 Angel-vLLM 加速能力，并进行接口调用。

前置条件

用户通过TI-ONE 训练平台部署并通过内置推理加速镜像 Angel-vLLM 使用自有模型，首先需要准备如下资源（包含存储模型等文件的 CFS 以及推理模型使用的算力资源 GPU 等）。

文件存储

申请 CFS：在自定义大模型部署推理过程中模型文件使用到的存储为 CFS，所以需要您首先申请 CFS，CFS 实例区别详见文件存储-存储类型及性能规格。请保证购买的 CFS 实例与上述算力资源机器的网络互通。CFS 使用详情请查看文件存储-创建文件系统及挂载点。备注：若您的基础存储服务是 GooseFSx，整体操作和本文介绍的 CFS 类似可直接参考。

算力资源

- 自行购买算力：若您首次使用TI-ONE 训练平台，请参考资源组管理指引，根据推理的模型参数购买合理的算力资源。
- 根据实际的模型大小或拥有的资源情况选择，大模型推理时需要的机器资源与模型的参数量及上下文长度相关，推荐按如下表格快速预估推理服务资源，详细的显存占用大小请参考推理资源要求。

模型参数量	GPU 卡类型和数量
6 ~ 8B	L20 * 1 / A10 * 1 / A100 * 1 / V100 * 1
12 ~ 14B	L20 * 1 / A10 * 2 / A100 * 1 / V100 * 2
65 ~ 72B	L20 * 8 / A100 * 8

模型准备

> 说明：这里我们通过开发机下载或上传需要使用的模型文件，示例使用的资源为 CPU 8 核，内存 16 GiB（只用于开发机实例，与后续推理服务无关，可适当减少资源）

选择训练工坊 > 开发机 > 新建，创建开发机，核心参数配置建议如下：

- 镜像：选择任意内置镜像即可，本开发机实例仅用于下载模型文件。
- 资源组和资源申请：选择您已创建好的资源组并配置适量的CPU资源即可。
- 存储配置：选择 CFS 文件系统，名称为上文前置要求中申请配置好的 CFS，路径默认为根目录 /，用于指定保存用户自定义大模型位置。

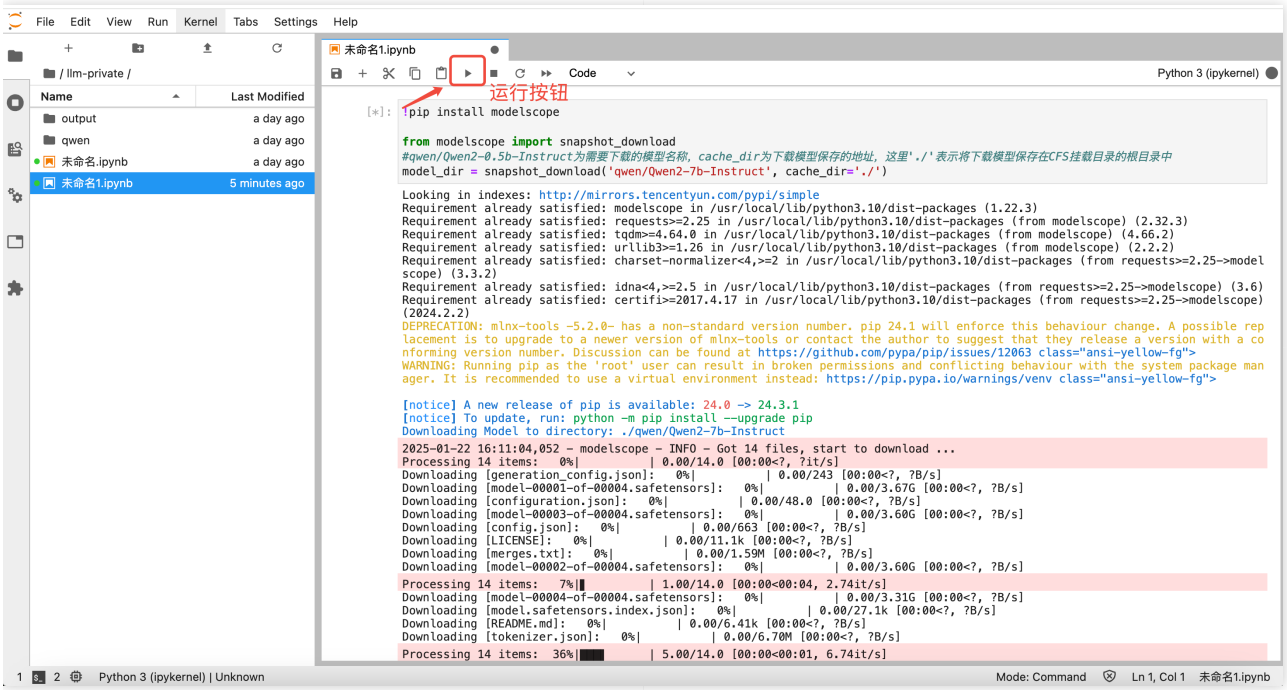
模型文件

新建成功后启动开发机，单击开发机下 python3 的 kernel 来新建一个 ipynb 页面通过脚本下载所需模型；您可在魔搭社区或 Hugging Face检索需要用到的大模型，通过社区中 Python 脚本自行下载模型并保存到 CFS 中，本文以【Qwen2-7b-Instruct】模型为例，下载代码如下：

```
!pip install modelscope

from modelscope import snapshot_download
#qwen/Qwen2-7b-Instruct为需要下载的模型名称，cache_dir为下载模型保存的地址，这里'/home/tione/notebook'表示将下载模型保存在CFS挂载目录的根目录中
model_dir = snapshot_download('qwen/Qwen2-7b-Instruct', cache_dir='/home/tione/notebook')
```

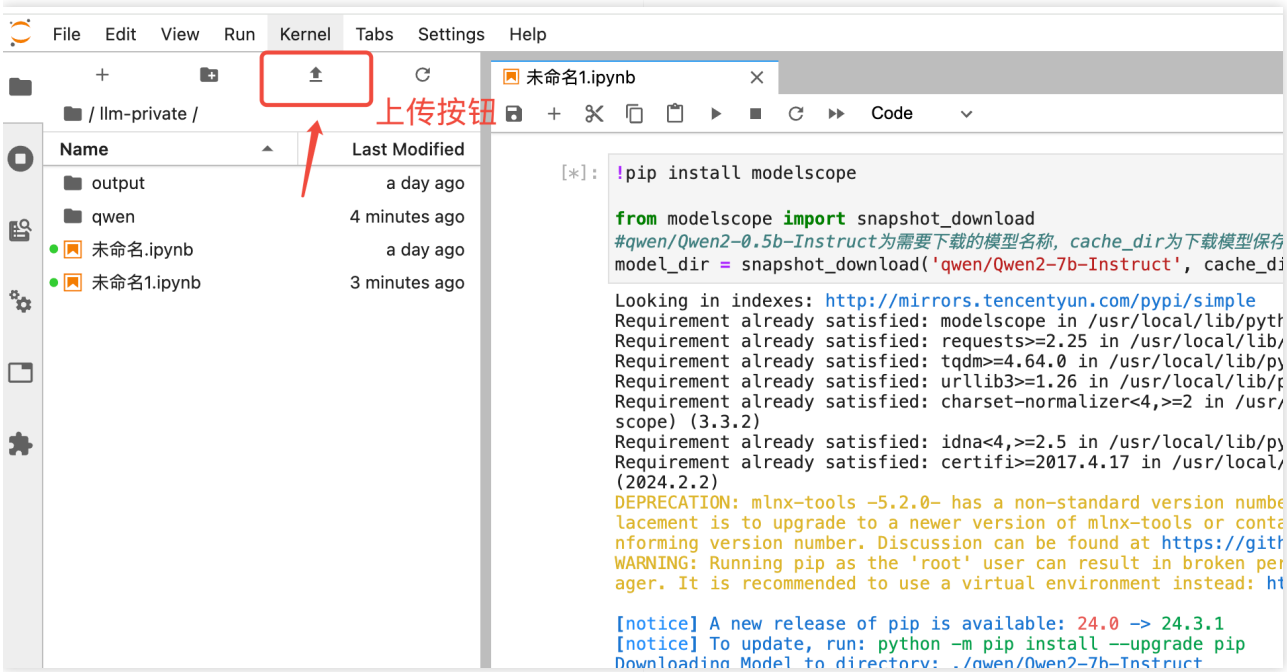
> 注意：这里路径映射的关系如下，以 CFS 挂载时容器挂载路径为 "/home/tione/notebook" 为例，若 CFS 源路径为 "/"，cache_dir 为 "/home/tione/notebook"，则模型实际保存在 CFS 的根路径下；若 CFS 路径为 "/dir"，cache_dir 为 "/home/tione/notebook/model"，则模型实际保存在 CFS 的 "/dir/model" 目录下。

复制上述下载脚本并替换 "qwen/Qwen2-7b-Instruct" 为您需要下载的模型后，粘贴到新建的 ipynb 文件中，单击运行（或单击 cell 格后，键入"enter+shift"）即可开始下载模型；



此外，你也可将本地已有大模型文件，通过本地上传，如下图：

> 注意：这种上传方式带宽有限，建议只传10MB以内的文件，若您的模型较大，建议通过 COS 中转上传（推荐上传到与 CFS 相同地域的 COS 存储桶中）。桌面端使用 COS 可以参考对象存储-桌面端使用说明，命令行使用 cos 可以参考对象存储-COSCMD 工具，然后在开发机中安装 coscmd 或 coscli 工具来下载文件到 CFS 中。



服务部署指引

部署流程

进入产品控制台的模型服务 > 在线服务页面，单击新建服务：

- 服务名称：自定义填写；
- 部署方式：若单机资源足够部署模型，推荐选择标准部署；若需要部署超大模型，单机资源不足，可以选择多机分布式部署；
- 资源组：请选择您已创建好的资源组；
- 副本设置：
  - 模型来源：选择 CFS，并选中模型存储的 CFS 实例，路径按实际训练输出路径填写，若需要选择其中的某一个 checkpoint，则填写到 checkpoint 这一级目录；
  - 选择模型：选择模型所在的 CFS 实例，然后填写模型目录在 CFS 上的源路径；
  - 运行环境：选择 内置 / LLM / angel-vllm(2.1)；
  - 资源申请 / 算力规格：按需选择模型需要的算力资源，资源要求可参考附录推理资源要求；
  - 高级设置：启动命令和环境变量请参考服务部署参数。

单击启动服务，服务会进入创建流程。此时服务的状态可能会在就绪中状态维持一段时间，表示服务还没有启动完成，您可以通过单击具体服务的事件或日志标签页来查看服务的具体启动进程。

服务部署参数

启动命令说明

镜像的默认启动命令（即不填写启动命令的默认值）为 run ，整体功能基本等价于 vLLM 自带的启动命令 vllm serve 或 python3 -m vllm.entrypoints.openai.api_server

默认启动命令相比 vLLM 自带的启动命令的主要区别是：

- 新增部分参数通过环境变量设置功能，详见下面的启动命令参数和环境变量说明；
- 调整了部分参数的默认值，支持部分参数自动配置，详见下面的启动命令参数和环境变量说明；
- 当使用多机分布式部署方式时，自动建立多机 ray 通信环境；
- 支持 TI-ONE 平台内置大模型训练的 LoRA 模型快速启动模型服务（自动开启 multi-lora 或做 LoRA 权重合并）；
- 对 CFS Turbo 存储介质上的模型的内存预读加载速度优化；
  以下所有启动命令参数对于 run 、 vllm serve 、 python3 -m vllm.entrypoints.openai.api_server 三种启动命令均兼容，环境变量仅默认的 run 启动命令支持。

启动命令参数和环境变量说明

参数详解

Angel-vLLM 核心加速功能参数：

启动命令参数	环境变量	含义
--quantization	QUANTIZATION	量化方式，默认未设置，相比开源 vLLM 0.6.2 版本新增以下量化方式： - fp8 ：W8A8 FP8 精度量化，支持 NVIDIA 计算能力 >= 8.9 的显卡系列（Ada Lovelace、Hopper），模型权重的显存占用降低 50%，提升推理速度，精度几乎无损； - ifq_nf4 ：W4A16 NF4 精度量化，支持 NVIDIA 计算能力 >= 8.0 的显卡系列（Ampere、Ada Lovelace、Hopper），模型权重的显存占用降低 75%，提升推理速度，相比 INT4 量化精度更好（该量化方式仅支持基于 float16 精度量化，即 --dtype float16 ）； - ifq ：W8A16 INT8 精度量化，支持 NVIDIA 计算能力 >= 7.0 的显卡系列（Volta、Turing、Ampere、Ada Lovelace、Hopper），模型权重的显存占用降低 50%，提升推理速度，精度几乎无损； 这 3 种量化方式都支持一键在线量化，无需提前对模型做转换或离线校准。 显存占用：ifq_nf4 < fp8 = ifq < 非量化 推理速度：fp8 >= ifq_nf4 > ifq > 非量化 模型精度：非量化 >= fp8 >= ifq > ifq_nf4
--use-lookahead	USE_LOOKAHEAD	默认为"0"，设置为"1"表示开启Lookahead并行解码，很多场景下可以显著加快解码速度；相比开源实现，Lookahead 并行解码无需额外的小模型或模型头，可以做到并行解码的结果和非并行解码的结果一致，对于输出文本中有大部分在输入文本中都出现过（例如 RAG 场景），或是大量请求中有相似的请求或答案的情况下有明显的加速效果；（加速效果可以见附录）
--num-speculative-tokens	NUM_SPECULATIVE_TOKENS	默认为"6"，表示Lookahead并行解码一次解码长度，若实际需要支持的并发数较大，可以调小此值，并发数小，可以调大此值；

为了兼容更多场景，平台内置镜像调整了部分启动参数默认值，以下启动参数与 vLLM 的默认启动参数不同：

启动命令参数	环境变量	含义
--max-model-len	MAX_MODEL_LEN	模型最大上下文长度，为兼容显存较小的机型，平台默认为上下文长度大于 8K 的模型调小此参数为 8192 ，若您实际需要支持更长上下文，

启动命令参数	环境变量	含义
		您可以手动配置此参数来调整。
--dtype	DTYPE	默认为 float16 ，若您希望使用 bfloat16 精度推理，请手动修改为 bfloat16。
--enable-prefix-caching	ENABLE_PREFIX_CACHING	默认为 true ，开启 prefix caching 功能，针对包含重复前缀的长文本输入或者连续多轮对话的首字延迟有显著加速效果。您可以设置环境变量 ENABLE_PREFIX_CACHING=false 来手动关闭。（V100显卡由于暂不支持此功能，默认为 false ）
--tensor-parallel-size	TP	默认为创建在线服务时指定的卡数，表示模型并行大小。
--trust-remote-code	TRUST_REMOTE_CODE	默认为 true ，以兼容需要开启此选项的模型，您可以设置环境变量 TRUST_REMOTE_CODE=false 来手动关闭。
--model	MODEL	模型名称或路径，默认为 /data/model ，对应平台默认容器内模型挂载路径。
--port	-	服务端口，默认为 8501 ，对应平台默认推理服务端口。
--use-v2-block-manager	-	默认为 true ，使用 v2 block manager。
--chat-template	-	HuggingFace 对话模板， .jinja 对话模板文件路径或模板字符串；（若您未设置此参数，但是指定了 MODEL_ID 或 CONV_TEMPLATE 环境变量，或模型目录中包含 ti_model_config.json 文件，平台会尝试做自动匹配设置，详见对话模板 ）
--tool-call-parser	-	工具调用解析器，可选值 ["llama3_json", "hermes", "mistral", "internlm"]；（若您未设置此参数，但是指定了 MODEL_ID 或 CONV_TEMPLATE 环境变量，或模型目录中包含 ti_model_config.json 文件，平台会尝试做自动匹配设置，详见对话模板 ）
--enable-auto-tool-choice	-	开启自动工具调用能力，需和 --tool-call-parser 配合使用；（若您未设置此参数，但是指定了 MODEL_ID 或 CONV_TEMPLATE 环境变量，或模型目录中包含 ti_model_config.json 文件，平台会尝试做自动匹配设置，详见对话模板 ）

其他平台镜像额外支持环境变量配置的参数：

启动命令参数	环境变量	含义
--gpu-memory-utilization	GPU_MEMORY_UTILIZATION	显存使用率，默认 0.9
--enforce-eager	ENFORCE_EAGER	是否强制开启 Pytorch 的 eager 模式，默认 false ，此时会额外使用 CUDA graph 做进一步加速，但会占用额外显存，并增加一些服务启动耗时。

启动命令参数	环境变量	含义
-	MODEL_ID	模型名称，默认为 model ，推荐直接设置成模型在 HuggingFace 上的名称，这样会触发对话模板自动匹配；
-	CONV_TEMPLATE	对话模板名称，默认未设置，主要用于对话模板自动匹配；
-	DISABLE_MEM_CACHE	禁用内存缓存预读，默认 false ；

剩余服务支持的启动参数请参考 vLLM 的官方文档：OpenAI Compatible Server。

各个场景的推荐参数

在大多数情况下，您可以不填写任何启动命令及环境变量直接启动推理服务，针对部分典型场景，我们在这里提供一些启动参数的配置指引。

案例 - 模型体验 - 最小资源需求

- 场景：推理算力资源较少，默认参数部署需要的模型报显存不足，仅作模型快速体验用，对模型精度和推理速度没有特定要求。
- 推荐启动命令： run --quantization ifq_nf4 --enforce-eager --gpu-memory-utilization 0.95 --max-model-len 2048
- 解释：
  - --quantization ifq_nf4 ：开启 4bit 量化，模型权重占用显存降低 75%（此功能需要Ampere、Ada Lovelace、Hopper系列显卡型号，若您使用的是V100或T4显卡，可以开启 ifq 量化模式）；
  - --enforce-eager ：关闭 CUDA Graph，减少额外显存占用，从而可以设置更大的 --gpu-memory-utilization；
  - --gpu-memory-utilization 0.95 ：调大显存使用率到 95%，增加显存利用率；
  - --max-model-len 2048 ：调小上下文长度到 2048，减少 KV Cache 所需显存大小；

案例 - 生产环境部署 - 简历解析

- 场景：对模型精度和推理速度均有较高的要求。简历解析场景输入文本较长，生成内容在输入中出现过的概率较高，较适合开启并行解码加速。
- 推荐启动命令： run --dtype auto --quantization fp8 --use-lookahead --num-speculative-tokens 4 --max-model-len 32768 --enable-prefix-caching --disable-log-requests --api-keys mock_api_key --served-model-name model_123
- 解释：
  - --dtype auto ：模型计算精度自动，即采用模型 config.json 中配置的精度（部分模型 fp16 和 bf16 效果有一些差异，可以根据实际测试情况调整）；
  - --quantization fp8 ：采用 FP8 W8A8 量化，在保证精度的同时极大加速推理，并减少 50% 的模型权重显存占用（此功能需要Ada Lovelace、Hopper系列显卡型号，若您使用的其他显卡，可以开启 ifq 量化模式）；
  - --use-lookahead ：开启并行解码，加速该场景下生成速度，提高吞吐；
  - --num-speculative-tokens 4 ：并行解码参数，当需要支持多并发时推荐设置稍微小一些，可按实际调整测试吞吐；
  - --max-model-len 32768 ：按需设置需要支持的最大上下文长度；
  - --enable-prefix-caching ：开启prefix caching 能力，该选项在 run 启动命令中默认已开启，也可以不填；
  - --disable-log-requests ：禁用日志记录请求详情，避免日志中包含太多太长的请求信息；
  - --api-keys mock_api_key ：设置兼容 openai API 调用的鉴权 Token，避免公网接口被随意调用；
  - --served-model-name model_123 ：设置模型名称，方便客户端区分推理结果是哪个模型返回的；

案例 - 使用 vllm 原生启动命令 - 吞吐优化

- 场景：希望使用 vllm 原生的启动命令部署服务，并尽可能地优化服务吞吐。
- 推荐启动命令： vllm serve /data/model --port 8501 -tp 2 --quantization fp8 --max-model-len 32768 --enable-prefix-caching --disable-log-requests --served-model-name model_123 --num-scheduler-steps 8
- 解释：
  - /data/model ：设置模型路径为默认的容器挂载路径；
  - --port 8501 ：设置服务端口为平台默认的8501端口；
  - -tp 2 ：设置模型TP并行数为2，这里需要根据实际服务使用的卡数调整，单卡推理可以不设置；
  - --quantization fp8 ：采用 FP8 W8A8 量化，在保证精度的同时极大加速推理，并减少 50% 的模型权重显存占用（此功能需要Ada Lovelace、Hopper系列显卡型号，若您使用的其他显卡，可以开启 ifq 量化模式）；
  - --max-model-len 32768 ：按需设置需要支持的最大上下文长度；
  - --enable-prefix-caching ：开启 prefix caching 能力，优化首字延迟；
  - --disable-log-requests ：禁用日志记录请求详情，避免日志中包含太多太长的请求信息；
  - --served-model-name model_123 ：设置模型名称，方便客户端区分推理结果是哪个模型返回的；
  - --num-scheduler-steps 8 ：开启多步调度能力，优化GPU利用率，提高服务吞吐（此选项与并行解码有冲突，建议根据实际业务情况与并行解码二选一来开启）；

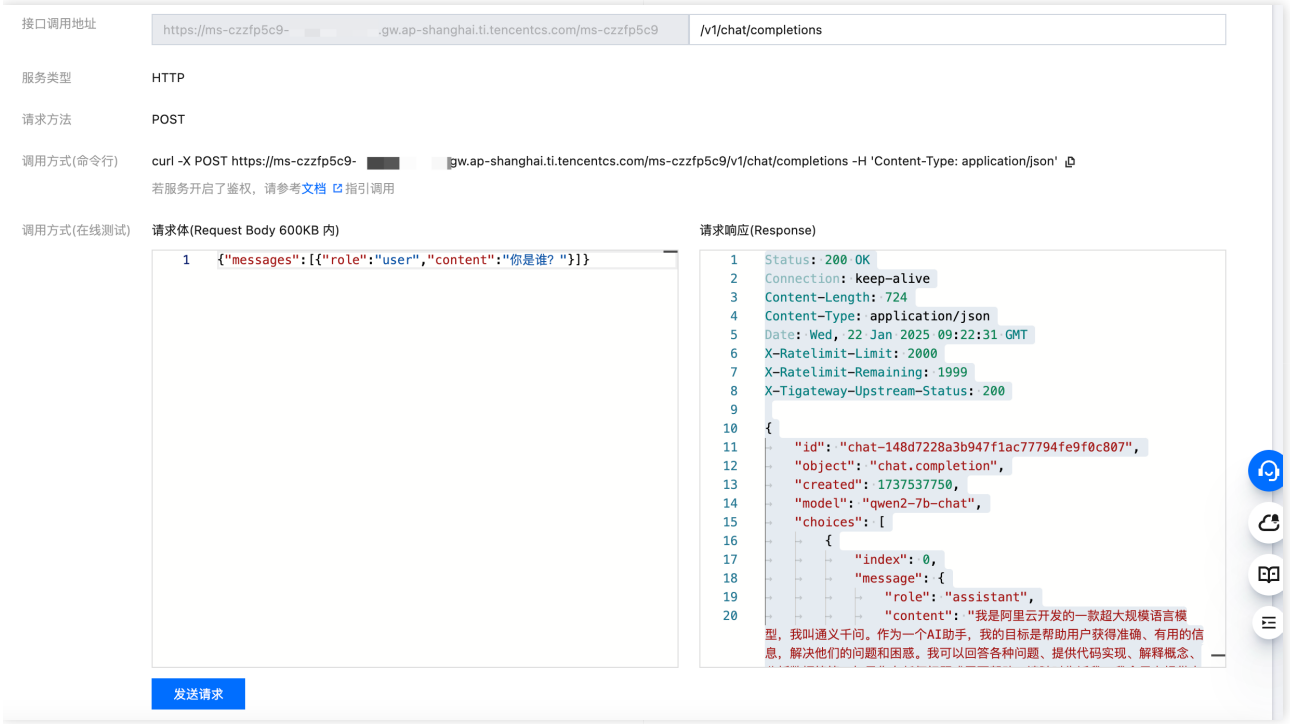
服务调用指引

对话体验

服务就绪后，可以看到“对话体验”的标签页，单击进入页面可以进行在线对话体验。

对话 API 调用

服务也支持通过 http/https 协议直接请求，单击“服务调用”页面，在接口调用地址一栏后输入对话 API 接口 /v1/chat/completions ，接口格式可以参考 OpenAI Chat Completions 接口。



下面为几个常见的调用场景：

纯文本对话

示例请求：

```
{"messages":[{"role":"user","content":"你好"}],"max_tokens":128}
```

工具调用

仅服务部署时开启工具调用能力时接口才支持，详见常见问题。

示例请求：

```
{
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_current_weather",
                "description": "Get the current weather in a given location",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "city": {
                            "type": "string",
                            "description": "The city to find the weather"
                        }
                    },
                    "required": [
                        "city"
```

```
                ]
            }
        }
    }
  ],
  "messages": [
    {"role": "user", "content": "what's the weather of xx today?"}
  ],
  "max_tokens": 512
}
```

多模态

仅模型本身支持多模态能力时才支持，例如 Qwen/Qwen2-VL-7B-Instruct ， meta-llama/Llama-3.2-11B-Vision-Instruct 模型。
示例请求：

```
{
  "messages": [
    {
      "role": "user",
      "content": [
        {
          "type": "text",
          "text": "What's in this image?"
        },
        {
          "type": "image_url",
          "image_url": {
            "url": "https://tione-public-cos-xxxx/test/cat.jpg"
          }
        }
      ]
    }
  ],
  "max_tokens": 512
}
```

第三方 LLM 应用接入

以 Dify 为例，可以添加类型为 OpenAI-API-compatible 的模型，然后 API endpoint URL 填写服务的在线调用地址，后加 /v1 即可。

服务性能测试

> **说明：**以下测试使用随机数据，若需要测试并行解码加速能力，建议使用真实业务数据测试。

启动上述模型准备中的开发机实例（确保有挂载大模型的 CFS），在 vllm 官方项目中vllm/benchmarks at main · vllm-project/vllm · GitHub下载测试脚本（包括 benchmark_serving.py 以及 backend_request_func.py ），在开发机中新建工作目录；新建

启动脚本如下（可根据自身需求修改相应参数）。

```
#!/bin/bash
RESULTS_FOLDER="results"
HOST="https://ms-cxxxxxxx.com/ms-czzfp5c9/"
MODEL="../Qwen2-7B-Instruct"

QPS="inf"
INPUT_LEN=128
OUTPUT_LEN=128
NUM_PROMPTS=20
CONCURRENCY=16

mkdir -p $RESULTS_FOLDER

client_command="python3 benchmark_serving.py \
--port 8501 \
--base-url $HOST \
--endpoint /v1/chat/completions \
--backend openai-chat \
--model $MODEL \
--dataset-name random \
--random-input-len $INPUT_LEN \
--random-output-len $OUTPUT_LEN \
--ignore-eos \
--request-rate $QPS \
--num-prompts $NUM_PROMPTS \
--save-result \
--result-dir $RESULTS_FOLDER \
--metadata input_len=$INPUT_LEN output_len=$OUTPUT_LEN qps=$QPS concurrency=$CONCURRENCY"

eval $client_command
```

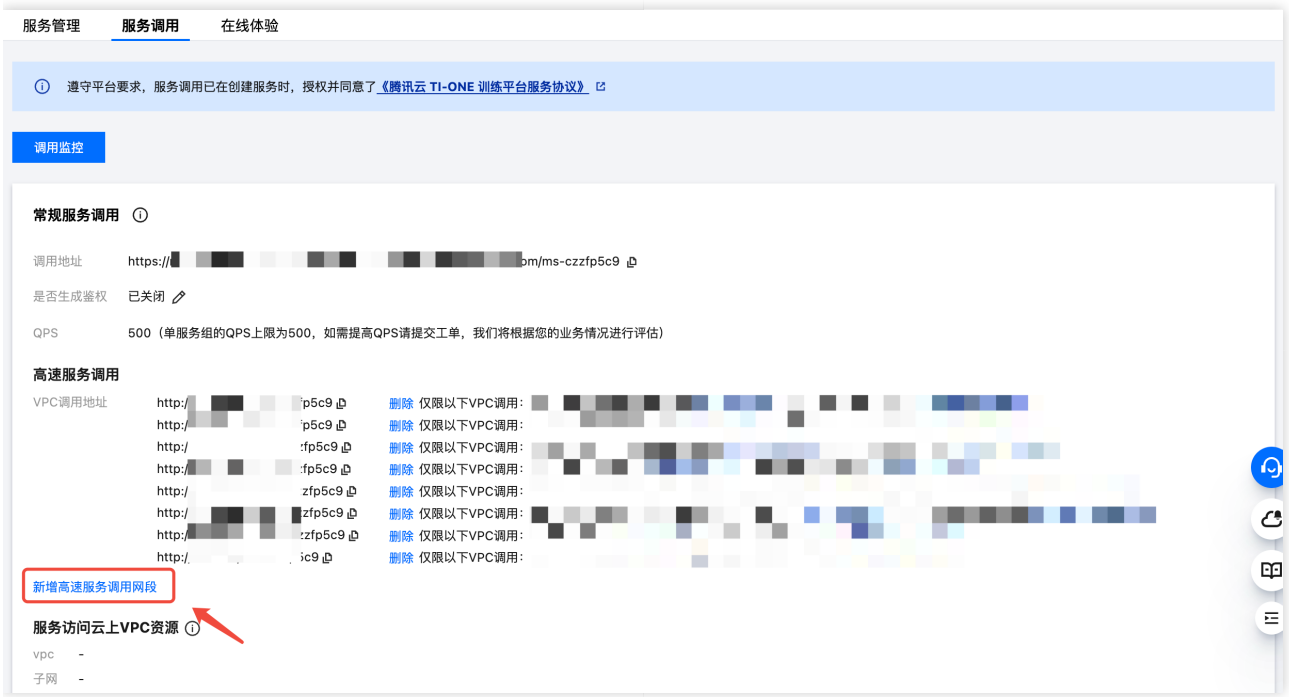
各参数含义说明如下：

参数名称	参数含义
MODEL	CFS 中挂载的模型路径
RESULTS_FOLDER	最终结果的保存路径
INPUT_LEN	模型输入的 token 数
OUTPUT_LEN	模型输出的最大 token 数
NUM_PROMPTS	请求总数
CONCURRENCY	并发数量
HOST	调用地址

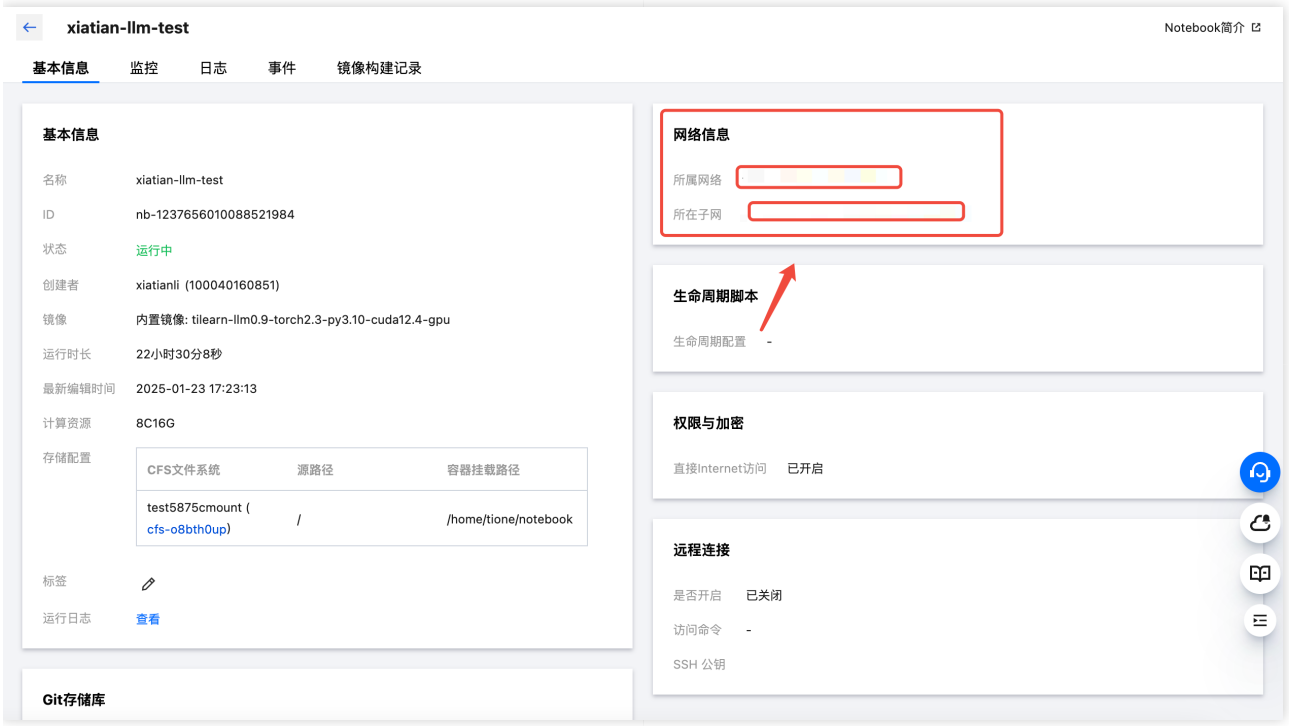
> **注意：**常规服务调用地址由于调用链路耗时可能受到 WAF 防火墙等影响产生波动，建议您使用高速服务调用地址来压测

> 服务性能。即将 HOST 配置成高速服务调用地址。您可将开发机实例所在的 VPC 网络新增到在线服务的高速服务调用网段，操作说明如下示例

首先在服务调用处单击新增高速服务调用网段：

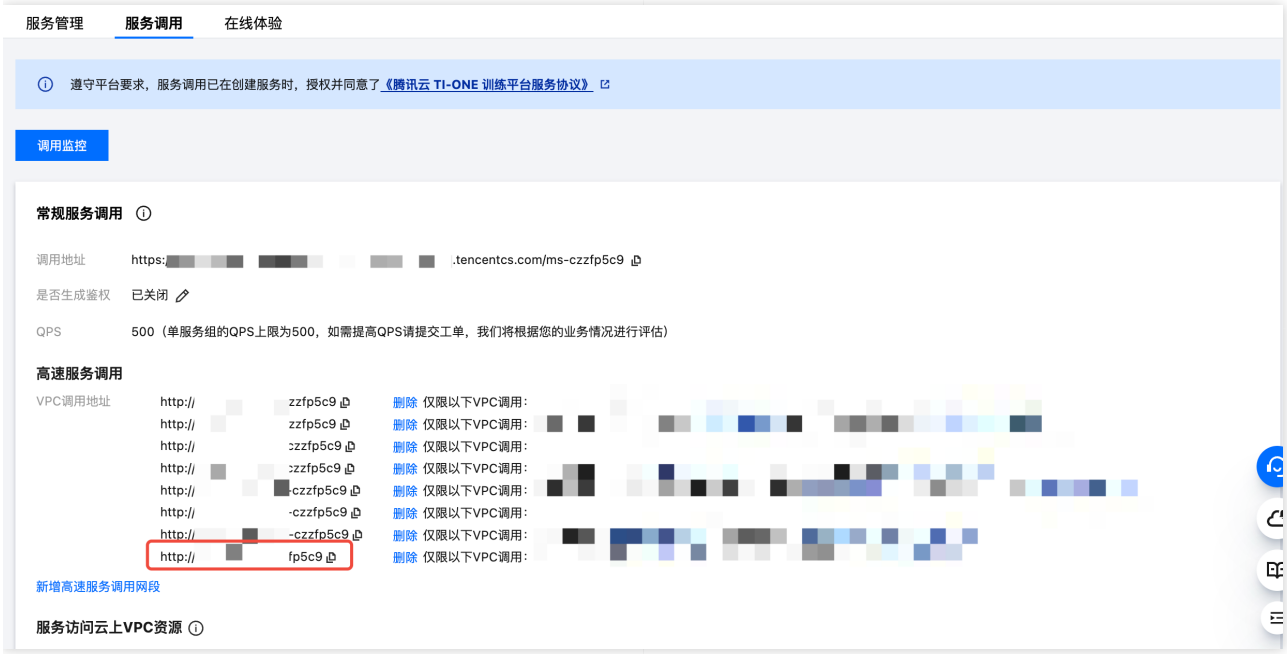


再将 开发机的网络信息（所属网络以及所在子网）复制如下：

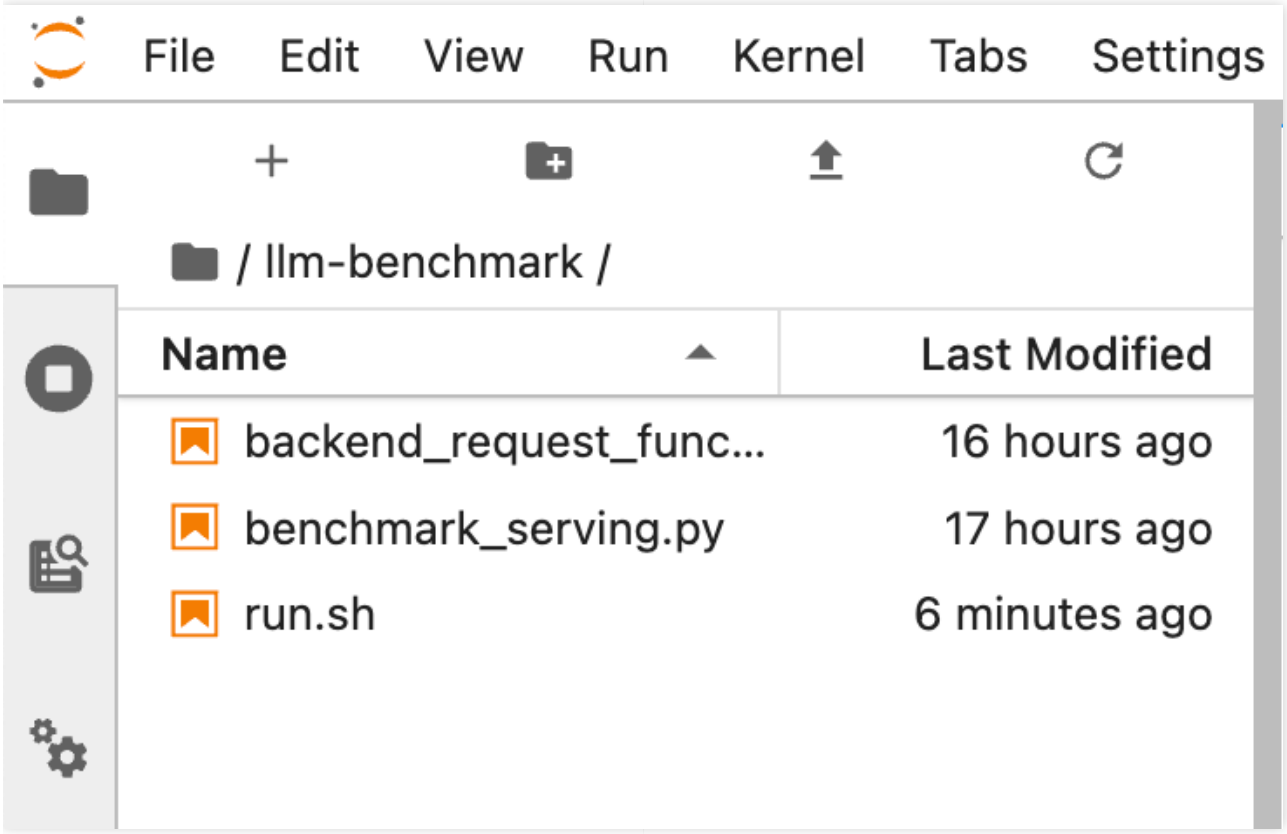


对应填写并确认添加：

最终的调用地址如下：



最终的测试文件目录组织如下：



新建终端 Terminal，执行 bash run.sh 即可开始执行测试脚本，可通过终端上的输出内容查看性能测试结果：

```
Starting initial single prompt test run...
Initial test run completed. Starting main benchmark run...
Traffic request rate: inf
Burstiness factor: 1.0 (Poisson process)
Maximum request concurrency: None
100%|██████████████████████████████████████████████████████████████████████████████████████████| 20/20 [00:27<00:00,  1.37s/it]
============ Serving Benchmark Result ============
Successful requests:                     20
Benchmark duration (s):                  27.37
Total input tokens:                      2560
Total generated tokens:                  7910
Request throughput (req/s):              0.73
Output token throughput (tok/s):         289.03
Total Token throughput (tok/s):          382.57
---------------Time to First Token----------------
Mean TTFT (ms):                          248.81
Median TTFT (ms):                        244.84
P99 TTFT (ms):                           261.74
-----Time per Output Token (excl. 1st token)------
Mean TPOT (ms):                          15.45
Median TPOT (ms):                        15.56
P99 TPOT (ms):                           15.94
----------------Inter-token Latency----------------
Mean ITL (ms):                           15.05
Median ITL (ms):                         15.14
P99 ITL (ms):                            67.92
==================================================
```

各指标含义如下：

指标名称	指标含义
Successful requests	完成总请求数
Benchmark duration (s)	所有请求完成时间
Total input tokens	所有请求输入的token数
Total generated tokens	所有请求输出的token数

指标名称	指标含义
Request throughput (req/s)	请求吞吐量
Output token throughput (tok/s)	生成结果吞吐量
Time to First Token（ms）	首token时延
Time per Output Token（ms）	每token时延
Inter-token Latency（ms）	token间时延

实际数据集的推理加速结果对比可以参考推理加速性能测试结果。

常见问题（FAQ）

问：Angel-vLLM 的并行解码能力使用有什么限制？

答：并行解码与 vLLM 的 --enable-chunked-prefill 功能有冲突，vLLM 默认会在 --max-model-len 超过 32K 时自动打开 --enable-chunked-prefill 功能，若此时需要使用 Lookahead 并行解码加速，请在启动命令中添加 --enable-chunked-prefill false 来手动关闭；此外，若启动 vLLM 时配置了 --num-scheduler-steps 参数大于1，也与并行解码不兼容。

问：Angel-vLLM 的量化能力和开源的 GPTQ 及 AWQ 量化有什么区别？

答：Angel-vLLM 的 INT8、NF4、FP8 量化，都支持一键在线量化，不需要用数据集做离线校准，且相比其他开源实现速度更快，其中 INT8 与 FP8 精度几乎无损，NF4 量化实验结果比 INT4 精度更好。开源的 GPTQ 和 AWQ 都是离线校准算法，这两个算法主要用于 INT4 和 INT8 居多。Angel-vLLM 不支持已经经过 GPTQ 或 AWQ 校准后的模型使用 ifq、ifq_nf4、fp8 量化部署，这些模型请使用 vllm 原生的 gptq、gptq_marlin、awq、awq_marlin 量化方式部署。

> 注意：
>
> 若需要推理 qwen2 或 qwen2.5 的 GPTQ 或 AWQ 量化模型，请添加环境变量 VLLM_ALLOW_QUANT_LM_HEAD=0

问：为什么我显卡有 40G 显存，部署 7B 的模型，显存占用率也会到 90%？

答：vLLM 框架会对显存进行预分配，具体会使用 显卡显存 * gpu_memory_utilization - 单次推理的峰值显存占用（包括模型权重 + KV Cache） 剩余的显存用于分配 GPU KV Cache blocks，以提高服务支持的批处理大小。要想估算模型对应的权重及 KV Cache 的最少显存占用，可以参考推理资源要求。

问：如何开启 function call 工具调用能力？

答：开启工具调用需要添加 --enable-auto-tool-choice 和 --tool-call-parser <parser_name> 参数。您也可以直接设置 MODEL_ID 为对应模型在 HuggingFace 上的名称，我们默认为部分支持工具调用的模型开启这些参数，详见对话模板。

附录

推理加速性能测试结果

FP8 量化加速测试

测试环境：

配置项	配置内容
GPU类型	L20
测试模型名称	Qwen2-7B-Instruct
测试数据集	ShareGPT_V3_unfiltered_cleaned_split.json
测试代码	vllm/benchmark
测试项目	对比不开加速与fp8加速

测试脚本：

```
python benchmark_serving.py \
        --port xxxx \
        --base-url http://xxxxxxxx/ms-rpbpr92c/ \
        --backend openai-chat \
        --model ../Qwen2-7B-Instruct \
        --dataset-name sharegpt \
        --endpoint /v1/chat/completions \
        --tokenizer ../Qwen2-7B-Instruct \
        --max-concurrency 10 \
        --save-result \
        --result-dir results \
        --dataset-path /path/to/ShareGPT_V3_unfiltered_cleaned_split.json
```

基于Angel-vLLM推理加速测试结果如下（concurrency=10，其他测试环境均一致）：

监控指标	无Angel-vLLM特性	FP8量化	效果提升
Successful requests	1000	1000	-
Total input tokens	217393	217393	-
Request throughput (req/s)	0.68	1.08	+58.82%
Output token throughput (tok/s)	383.75	597.04	+55.58%
Total Token throughput (tok/s)	531.74	832.25	+56.51%
Mean TTFT (ms)	113.27	86.90	+23.28%
Median TTFT (ms)	76.52	55.64	+27.29%
P99 TTFT (ms)	874.30	708.10	+19.01%
Mean TPOT (ms)	24.33	16.56	+31.94%
Median TPOT (ms)	24.17	16.46	+31.90%
P99 TPOT (ms)	27.34	18.68	+31.68%

监控指标	无Angel-vLLM特性	FP8量化	效果提升
Mean ITL (ms)	24.22	16.52	+31.79%
Median ITL (ms)	23.45	15.83	+32.49%
P99 ITL (ms)	69.92	50.20	+28.20%

Lookahead 并行解码加速效果

由于 Lookahead 并行解码的加速原理是服务见过的 token 会有加速效果，一般情况下加速效果会随着会话次数增多越来越好，下图为默认参数与开启lookahead并行解码后相同请求第二次请求时的生成速度直观对比。

对话模板

对话模板用于将用户输入的对话转换为输入给大模型的 prompt 文本，可以参考 huggingface 上的介绍。
对话模板一般采用 Jinja 模板来描述，对于比较新的开源对话模型（一般带 Instruct 或 Chat 后缀），您一般可以在模型目录的 tokenizer_config.json 文件或 chat_template.json 文件中找到 chat_template 字段，即为该模型的对话模板，若未做特殊设置我们默认会使用此对话模板。

为了便于一些未自带 chat_template 字段的模型部署，及便于一些支持 function call 工具调用的模型开启工具调用能力，若您指定了 MODEL_ID 或 CONV_TEMPLATE 环境变量，或模型目录中包含 ti_model_config.json 文件，并使用了默认启动命令启动的镜像，推理框架会按以下规则和顺序依次自动匹配模型的对话模板。（其中 CONV_TEMPLATE 和 MODEL_ID 匹配规则是"或"的关系）

模型系列	CONV_TEMPLATE	MODEL_ID (忽略大小写)	默认增加的启动命令	默认额外配置的 stop_token_ids
非对话模型	generate	-	--chat-template examples/template_generate.jinja	-
混元 Large	hunyuan	包含 hunyuan-large	--chat-template examples/template_hunyuan.jinja	[127960, 127967]
行业大模型	shennong_chat	包含 shennong 或 sn-	--chat-template examples/template_shennong.jinja	-
Llama-3.2 Vision Instruct 模型	llama-3.2-vision	包含 llama-3.2 且含 vision 且含 instruct 或 chat	--chat-template examples/tool_chat_template_llama3.2_vision.jinja --enable-auto-tool-choice --tool-call-parser llama3_json --enforce-eager --max-num-seqs 8	-
Llama-3.2 Instruct 模型	llama-3.2	包含 llama-3.2 不含 vision 且含 instruct 或 chat	--chat-template examples/tool_chat_template_llama3.2_json.jinja --enable-auto-tool-choice --tool-call-parser llama3_json	-
Llama-3.1 Instruct 模型	llama-3.1	包含 llama-3.1 且含 instruct 或 chat	--chat-template examples/tool_chat_template_llama3.1_json.jinja --enable-auto-tool-choice --tool-call-parser llama3_json	
Qwen2.5 Instruct 模型	qwen2.5	包含 qwen2.5 且含 instruct 或 chat	--enable-auto-tool-choice --tool-call-parser hermes	-
Baichuan2 Chat 模型	baichuan2-chat	包含 baichuan2 且含 instruct 或 chat	--chat-template examples/template_baichuan.jinja	-
Llama2 Chat 模型	llama-2	包含 llama-2 且含 instruct 或 chat	--chat-template examples/template_llama2.jinja	-
Llama3 Chat 模型	llama-3	包含 llama-3- 且含 instruct 或 chat	--chat-template examples/template_llama3.jinja	[128001, 128009]
Qwen Chat 模型	qwen 或 qwen-7b-chat	包含 qwen 且不含 vl 且含 instruct 或 chat	--chat-template examples/template_qwen.jinja	[151643, 151644, 151645]

推理资源要求

推理资源要求主要包括 CPU、内存、GPU，我们主要根据 GPU 显存计算需要的算力资源，CPU、内存简易按机型和卡数等比例分配，其中内存建议大于模型权重所占存储空间；
大模型推理对 GPU 显存要求较高，我们可以通过以下方式计算使用本镜像推理大模型所需要的显存大小及最少显卡数量：

模型权重所需显存 (GiB) ≈ 模型权重参数量(B) * 模型权重精度(Byte)
KV Cache所需显存 (GiB) = 模型层大小(hidden_size) * 模型层数(num_hidden_layers) * 模型KV头数(num_key_value_heads) / 模型attention头数(num_attention_heads) * 模型上下文长度(max_position_embeddings) * 模型KV-Cache精度(Byte) * 2 / 1024 / 1024 / 1024
所需显卡显存 > (模型权重所需显存 + KV Cache所需显存) / 显存使用率(gpu_memory_utilization)

所需最少显卡数量 = ceil(所需显卡显存 / 单卡显存大小)
其中要求 num_attention_heads % 显卡数量 == 0

其中，模型权重精度和KV Cache精度在未做量化前一般是 float16 或 bfloat16，即 2 字节；若进行了 int8 或 fp8 量化，则为 1 字节；若进行了 int4 或 nf4 量化，则为 0.5 字节；其他关键参数可以在模型目录的 config.json 文件中找到。
示例：

开源模型名称	Meta-Llama-3.1-8B-Instruct	Qwen2-72B-Instruct
模型权重参数量 (Billion)	8	72
模型权重精度 (Byte) （可通过开启 --quantization 量化调整）	2	2
合计模型权重占用(GiB)	16	144
模型层大小 (hidden_size)	4096	8192
模型层数 (num_hidden_layers)	32	80
模型KV头数 (num_key_value_heads)	8	8
模型attention头数 (num_attention_heads)	32	64
模型上下文长度 (max_position_embeddings) （可通过 --max-model-len 参数调整）	131072	32768
模型KV-Cache精度 (Byte)	2	2
合计模型KV-Cache占用(GiB)	16	10

开源模型名称	Meta-Llama-3.1-8B-Instruct	Qwen2-72B-Instruct
显存使用率（可通过 --gpu-memory-utilization 参数调整）	0.9	0.9
所需最少显卡显存(GiB)	35.56	171.11
24G显存显卡最少数量	2	8
40G显存显卡最少数量	1	8
48G显存显卡最少数量	1	4

LLM 训练及评测
精调满血版 DeepSeek-R1 全流程实践

总览

TI-ONE 平台已内置了全系列 DeepSeek 模型，包含满血版 V3 和 R1 模型，以及基于 DeepSeek-R1 蒸馏后的六个小模型。您可以在平台任务式建模模块单击新建任务，在训练镜像中选择内置大模型，即可一键发起全系列 DeepSeek 模型精调。
本文将介绍如何使用 TI 平台来完成 DeepSeek-R1-671B 模型的有监督精调，本实践的总体步骤如下：

- 步骤一：上传数据集
- 步骤二：启动精调任务
- 步骤三：模型转换
- 步骤四：部署服务
- 步骤五：验证推理结果
  本实践的结果是您将通过微调改变模型的自我认知。
  下文简称 DeepSeek-R1-671B 模型为R1模型。

前置准备条件

算力和存储资源准备

1. R1模型精调需要使用 GPU 算力资源，推荐高性能 GPU 算力。本文以对模型做全参 SFT 精调为例，需要最少 32 台 高性能 GPU 算力。因此在实践开始前，建议纳管至少32台高性能 GPU 算力，该模式下需要您提前配置好 CVM 机器并添加至 TI 平台资源组，详细操作步骤请参考资源组管理。
2. 模型精调过程中，数据集和模型 CheckPoint 存储都依赖 CFS，请前往 CFS 控制台开通 CFS。由于 R1 模型文件非常大，为保证训练保存 checkpoint 及后续使用模型部署推理服务和启动评测的速度，推荐使用 Turbo 系列存储，CFS 实例区别详见存储类型及规格；另外，请注意保证配置的 CFS 实例与上述算力资源机器的网络互通。

物料准备

模型、镜像和训练代码

本文用于精调的R1模型实践涉及的模型、镜像、代码等均平台内置，您无须额外下载。

数据集

本文精调的示例场景为针对模型的自我认知进行精调，用于精调的数据集下载地址如下：
identity.jsonl

详细步骤

步骤一：上传数据集

> **说明：**目前 TI-ONE 平台不支持在控制台直接进行数据上传操作，为了解决此问题，需要创建一个运维开发机以挂载

> CFS 并使用开发机服务进行上传或下载大模型、训练代码等文件。

进入训练工坊 > 开发机 ，单击新建按钮，创建开发机，

- 镜像：选择任意内置镜像即可，本开发机实例仅用于下载数据集。
- 存储配置：选择 CFS 文件系统，名称为上文前置要求中申请配置好的 CFS，路径默认为根目录 /，用于指定训练数据保存路径。
- 其它设置：默认不需要填写
  将前文已下载好的数据集上传至开发机中。

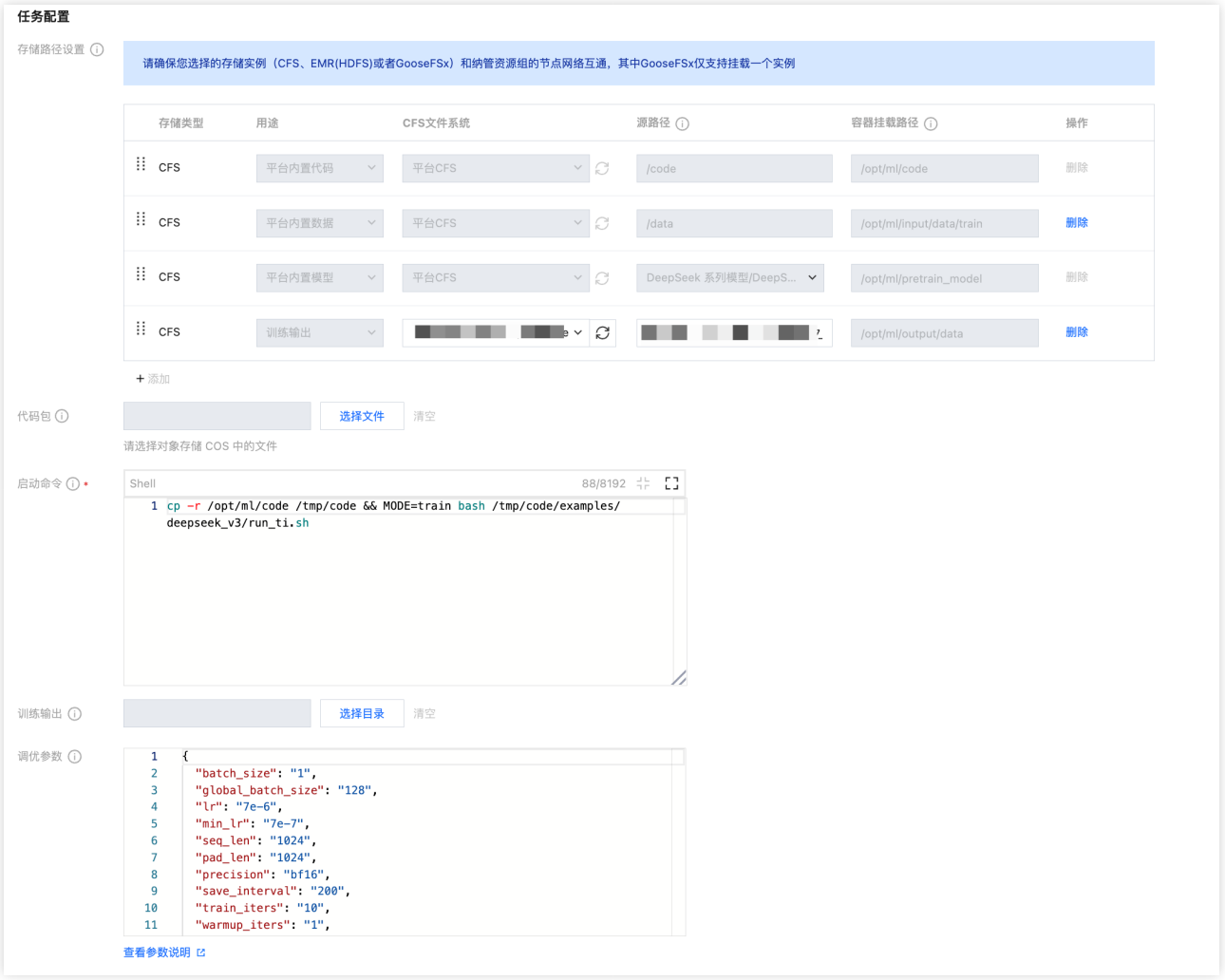
步骤二：启动精调任务

1. 进入训练工坊 > 任务式建模，单击新建任务，填写任务配置信息：
   - 任务名称：输入您的自定义任务名称。
   - 训练镜像：选择内置大模型/DeepSeek系列模型/DeepSeek-R1-671B。
   - 资源组：选择32台节点所在的资源组。
   - 资源申请：选择单节点的 GPU、CPU、内存资源尽量用满（某高性能机型单节点整机可用CPU380核，内存2214GB，若整机为空闲状态，推荐填写整机资源CPU380核，内存2214GB），如下图所示。
   - 标签：可根据实际情况填写，本实践不涉及。
   - 描述：可根据实际情况填写，本实践不涉及。
   - CLS 日志投递：默认关闭。
   - 自动重启：默认关闭。
   - 存储路径设置：本页面重点参数配置如下：
     - 存储路径设置：默认选中“CFS”（目前大模型精调的平台内置代码、平台内置数据和平台内置模型仅支持 CFS ）。
     - 选择存储路径
       - 第一行 “平台CFS”：系统默认为您配置了精调该大模型的配套训练代码。
       - 第二行 “平台CFS”：
         - 系统默认为您配置了一份精调该大模型的示例数据；
         - 本案例使用自定义业务数据精调该大模型，可删除此行，并在底部添加一行，选择用途为 **用户自有数据 ，**选择您的数据集所在的CFS和对应源路径，这里假设您的数据集所在路径为<您的CFS_id>/dataset。
       - 第三行 “平台CFS”：系统默认为您配置了平台内置模型（由于 R1 模型尺寸非常大，直接从平台默认共享存储拉取时间较慢，为了加速模型拉取，建议您通过销售或售前架构师联系 TI 平台团队为您提供手动缓存加速支持）。
       - 第四行 “用户CFS”：此处需选择您的训练输出所在的CFS 文件系统和对应源路径，这里假设您选择的输出路径为<您的CFS_id>/output。
   - 代码包：无需选择（大模型精调场景中您无需选择代码包）。
   - 启动命令：系统默认填充您无需修改（用于启动大模型默认配套的训练代码）。
   - 训练输出：无需选择。
   - 调优参数：为系统默认填充，本案例中填写的超参如下。实际精调场景中，可根据训练数据量来调整 train_iters，warmup_iters，seq_len，pad_len等参数。

{

```
  "batch_size": "1",
  "global_batch_size": "128",
  "lr": "7e-6",
  "min_lr": "7e-7",
  "seq_len": "1024",
  "pad_len": "1024",
  "precision": "bf16",
  "save_interval": "200",
  "train_iters": "10",
  "warmup_iters": "1",
  "sft": "true",
  "sft_packing": "false",
  "tp_size": "1",
  "pp_size": "8",
  "cp_size": "1",
  "ep_size": "32",
  "enable_sp": "true",
  "enable_do": "true",
  "enable_fl": "false",
  "ac": "full",
  "optimizer_offload": "false"
}
```


完整的超参说明如下：



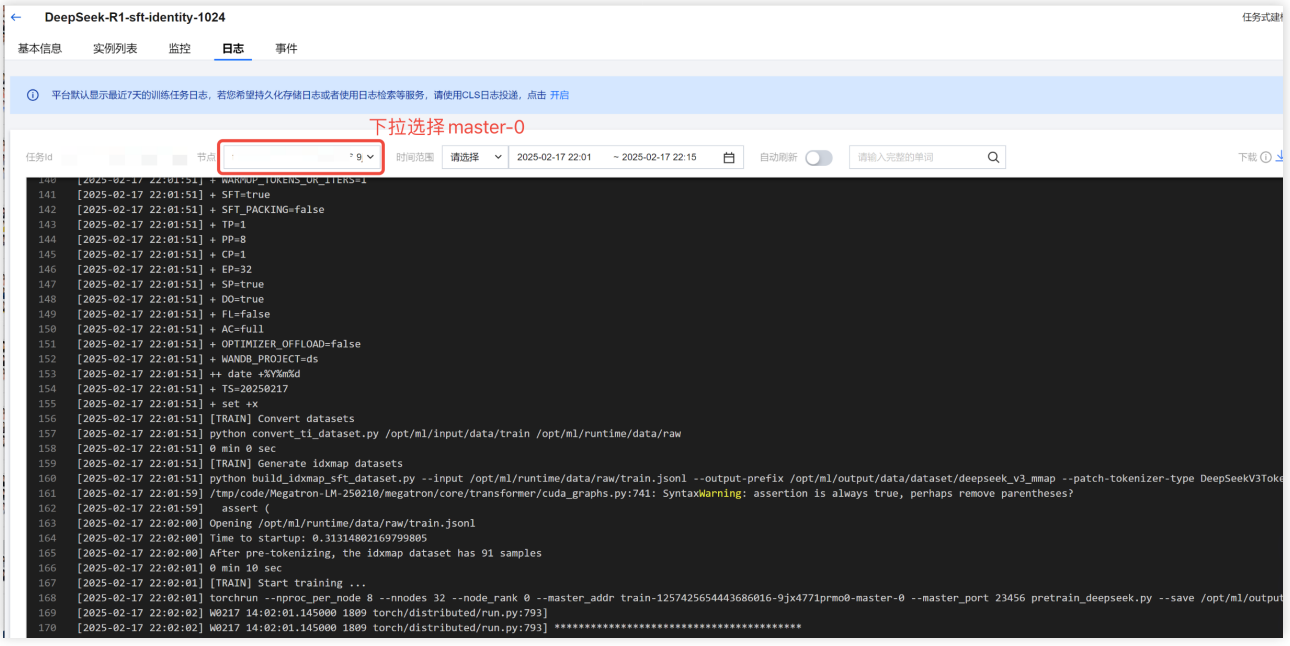
> 超参说明：
>
> 满血版V3/R1的精调超参列表是基于 Megatron 框架的，与其他模型的超参列表范围有区别，其他模型超参列表请查看 预置调优参数。

- batch_size: 一次迭代一个数据并行内的样本数
- global_batch_size: 一次迭代多个数据并行的总样本数
- lr: 学习率
- min_lr: 最小学习率
- seq_len: 序列长度
- pad_len: Padding长度
- precision: 训练精度: fp16, bf16, fp8
- save_interval: 保存ckpt的间隔
- train_iters: 训练迭代步数
- warmup_iters: 预热迭代步数
- sft: 是否执行微调训练: true, false
- sft_packing: SFT 时是否开启数据 packing
- tp_size: 模型并行度，当前只能设置为1
- pp_size: 流水并行度

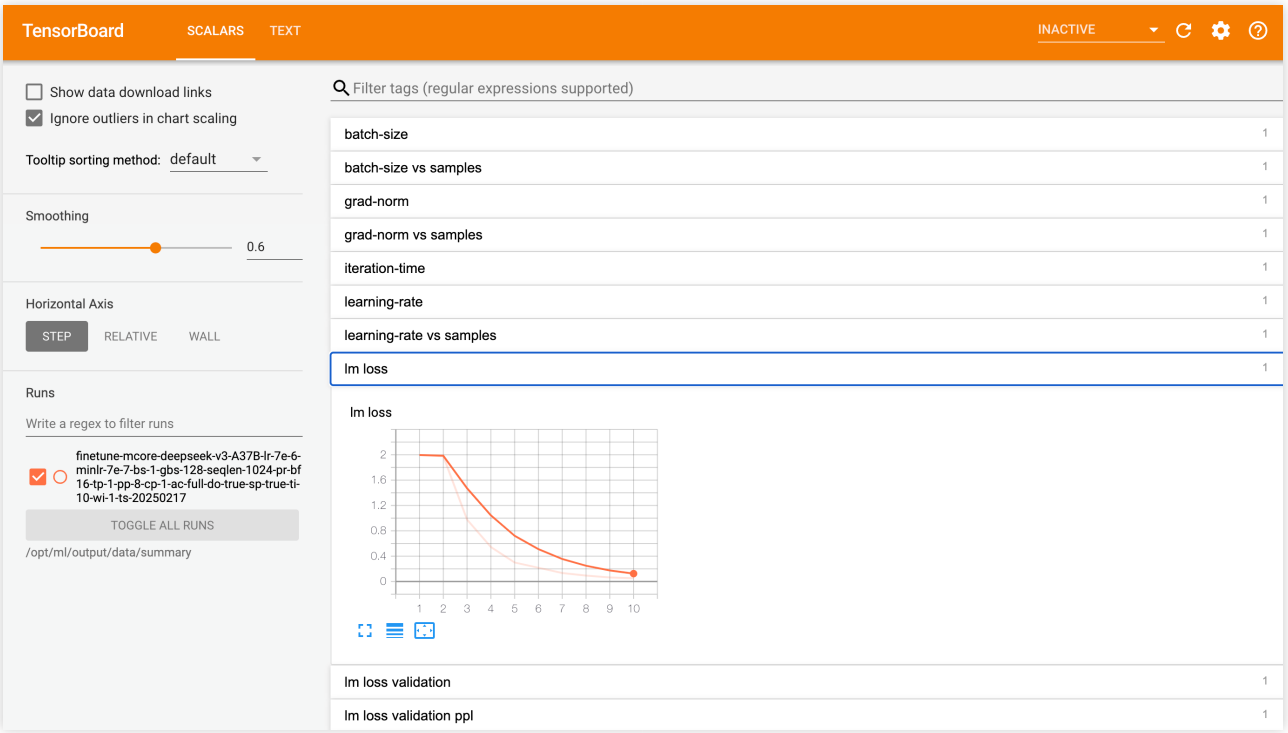
- cp_size: 上下文并行度
- ep_size: 专家并行度
- enable_sp: 是否使用序列并行: true, false
- enable_do: 是否使用Megatron版Zero-1降显存优化器: true, false
- enable_fl: 是否优先使用Flash Attention: true, false
- ac: 激活检查点模式: sel, full, offload, false
- optimizer_offload: 是否启用Offload optimizer: false, static, auto

2. 任务配置完成后，单击确定即可返回任务列表，您可以在此页面查看模型训练进度、训练日志、监控训练资源消耗和训练指标收敛情况等信息。

3. 训练任务运行过程中，可单击任务名称，进入日志 Tab 页，开始观察训练状态，训练任务启动后会有以下几个阶段：
   - 数据预处理：第一个节点（节点下拉选择 master-0）会打印数据预处理的信息：



   - 开始训练：最后一个节点（"节点"下拉框选择 worker-30 节点）会打印训练迭代步数、learning rate、lm loss 等信息，这里默认每 1 步打印一次训练信息日志。
     单击 Tensorboard，可查看训练指标监控。



训练完成后，模型即自动保存在训练配置的输出CFS路径中。

步骤三：模型转换

> 说明：
>
> 仅精调满血版V3/R1模型需要进行模型转换步骤，精调其他模型（例如 R1 Distill 系列模型）不需要进行模型转换，训练后的模型可以直接进行评测或者推理。

由于 TI 平台针对 R1 模型使用基于 Megatron-LM 的训练框架，其保存的 CheckPoint 格式为非 huggingface 格式，因此在进行模型推理前，需要先对模型进行格式转换。为了简化用户操作，平台将模型格式转换步骤集成到了任务式建模内置训练代码中，您只需要复制精调训练任务，修改部分参数即可完成模型转换。具体步骤如下：

1. 复制精调训练任务，修改节点数为 1，修改启动命令中的 MODE 变量为 "mg2hf"：
2. 其他参数设置不变，单击确定运行任务，任务会将对应精调训练任务最后输出的模型转换为 huggingface 格式，保存到原始 checkpoint 目录同级目录下，以 _hf 后缀命名。如果 cfs 为 turbo 类型，预计模型转换时间为 1 小时。

步骤四：部署服务

进入 模型服务 > 在线服务，单击新建服务按钮：
服务名称可以自定义，其中部署方式选择多机分布式部署模型；模型来源选择 CFS，并选中模型存储的 CFS 实例，路径选择 CFS 中模型转换后的 hf 格式模型目录，运行环境选择内置的 LLM / sglang 镜像，资源配置使用2机16卡高性能 GPU 机型，同时为了加速启动，添加环境变量 ENABLE_TORCH_COMPILE=0，如下图所示。

步骤五：验证推理结果

服务部署成功后，可进入在线体验页面进行模型对话。通过对话，可发现模型的自我认知已发生改变。
注意：本实践重在提升 R1 的自我认知，如需提升其他通用能力，需要合理筛选、清洗和配比微调数据集。考虑到 R1 本身能力较强，如何通过精调提升其通用能力，具有一定门槛和挑战。欢迎广大算法工程师使用 TI-ONE 平台持续进行探索和实践。


在线服务
上海
在线服务简介
新建服务
多个关键字用竖线 "|" 分隔，多个过滤标签用回车键分隔
名称
状态
机器来源
运行中/总版本数
运行/期望副本数量
标签
创建时间
操作
已停止
从CVM机器中选择
0/1
0/0
-
2025-02-18 16:51:09
调用API 编辑标签 在线体验 调用 删除
R1-精调后推理
运行中
从CVM机器中选择
1/1
1/1
-
2025-02-18 16:38:48
调用API 编辑标签 在线体验 调用 删除
运行中
从CVM机器中选择
1/1
1/1
-
2025-02-18 16:08:33
调用API 编辑标签 在线体验 调用 删除
已停止
从CVM机器中选择
0/1
0/0
-
2025-02-18 15:46:57
调用API 编辑标签 在线体验 删除

使用任务式建模精调自定义大模型

总览

本文以 qwen2-0.5b-instruct 为例，展示如何使用 TI-ONE 平台精调用户自定义大模型，并通过在线服务将精调后的模型推理部署。这里我们使用到的训练框架为开源 LLaMA-Factory，并对 qwen2-0.5b-instruct 进行全参 sft 微调，最后使用平台内置镜像 Angel-vLLM 对训练后的模型进行推理部署。

前置要求

申请 CFS ：在自定义大模型精调过程中训练数据、模型、代码等使用到的存储可以为 CFS ，所以需要您首先申请 CFS 。

操作步骤

物料准备

> 说明：
>
> 目前 TI-ONE 平台不支持在控制台直接进行数据上传操作，为了解决此问题，需要创建一个运维开发机以挂载 CFS 并使用开发机服务进行上传或下载大模型、训练代码等文件。

单击训练工坊 > 开发机 > 新建按钮，创建调试训练代码用的开发机，

- 镜像：选择任意内置镜像即可，本开发机实例仅用于下载模型文件以及训练代码。
- 资源组：下拉选择资源组，请参考资源组简介管理您的机器资源。
- 存储配置：选择 CFS 文件系统，名称为上文前置要求中申请配置好的 CFS ，路径默认为根目录 /，用于指定保存用户自定义大模型位置。
- 其它设置：默认不需要填写。
  最终开发机服务配置如下：

模型文件

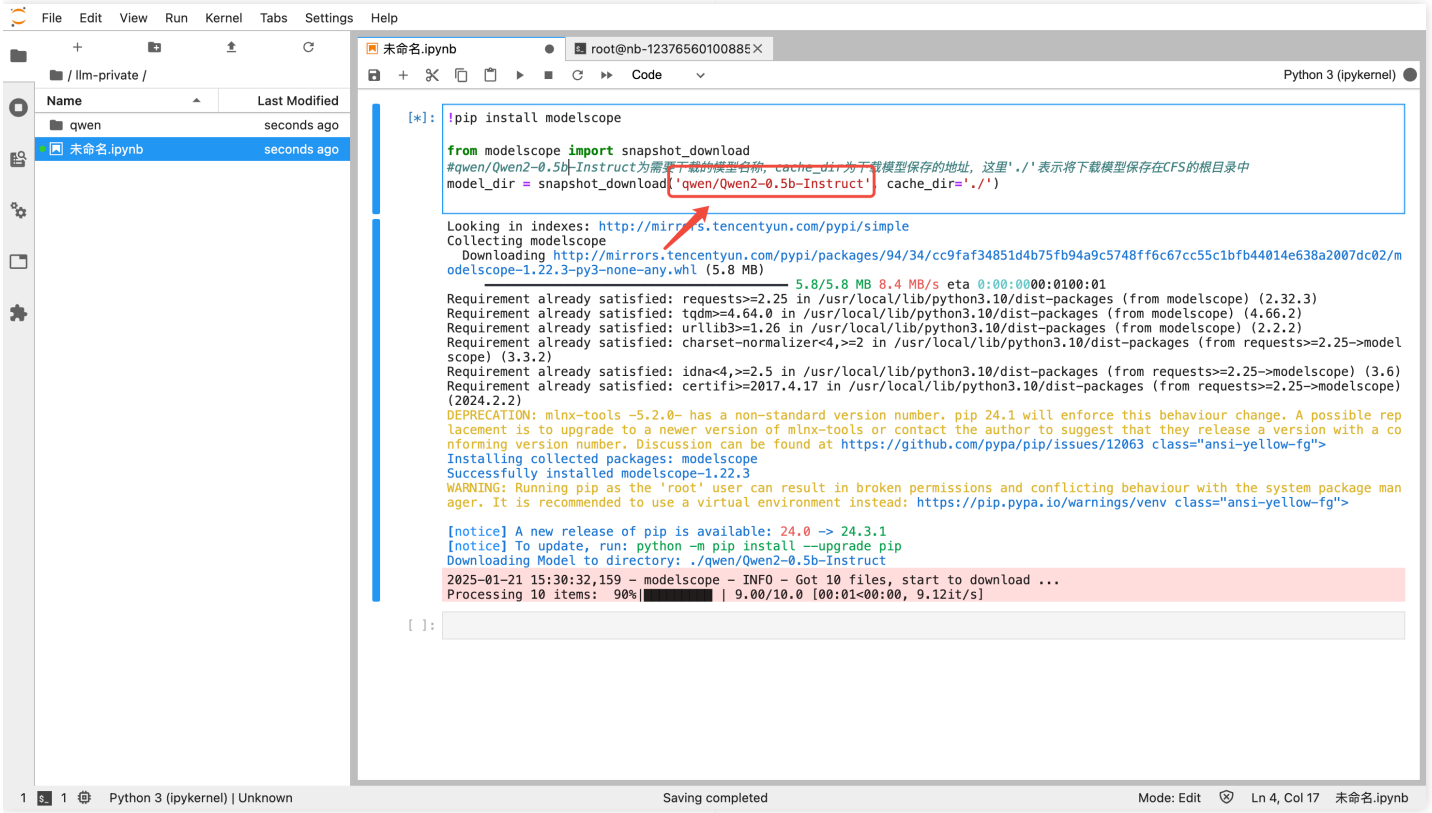
新建成功后，单击开发机 > Python3(ipykernel) 通过脚本下载所需模型；您可在 魔搭社区 或 Hugging Face 检索需要用到的大模型，通过社区中 Python 脚本自行下载模型并保存到 CFS 中，本文以 Qwen2-0.5B-Instruct 模型为例，下载代码如下：

```
!pip install modelscope
```

```
from modelscope import snapshot_download
#qwen/Qwen2-0.5b-Instruct为需要下载的模型名称，cache_dir为下载模型保存的地址，这里'./'表示将下载模型保存在CFS挂载目录的根目录中
model_dir = snapshot_download('qwen/Qwen2-0.5b-Instruct', cache_dir='./')
```

> 说明：
>
> 指定下载模型的地址 cache_dir 在后续任务是建模中的位置为挂载开发机的 path+cache_dir，例如：挂载开发机的路径为 /dir1，cache_dir 为 /dir2，则文件在 CFS 中的位置为 /dir1/dir2。

复制上述下载脚本并更换需要下载的模型后，粘贴到新建的 ipynb 文件中，单击运行按钮，即可开始下载模型。



训练代码

下载 LLaMA-Factory 代码

直接通过 git clone 下载到 CFS 中，单击 Terminal 输入如下命令，如果需要用最新代码，请使用 main 分支。

```
git clone -b v0.9.2 https://github.com/hiyouga/LLaMA-Factory.git
```

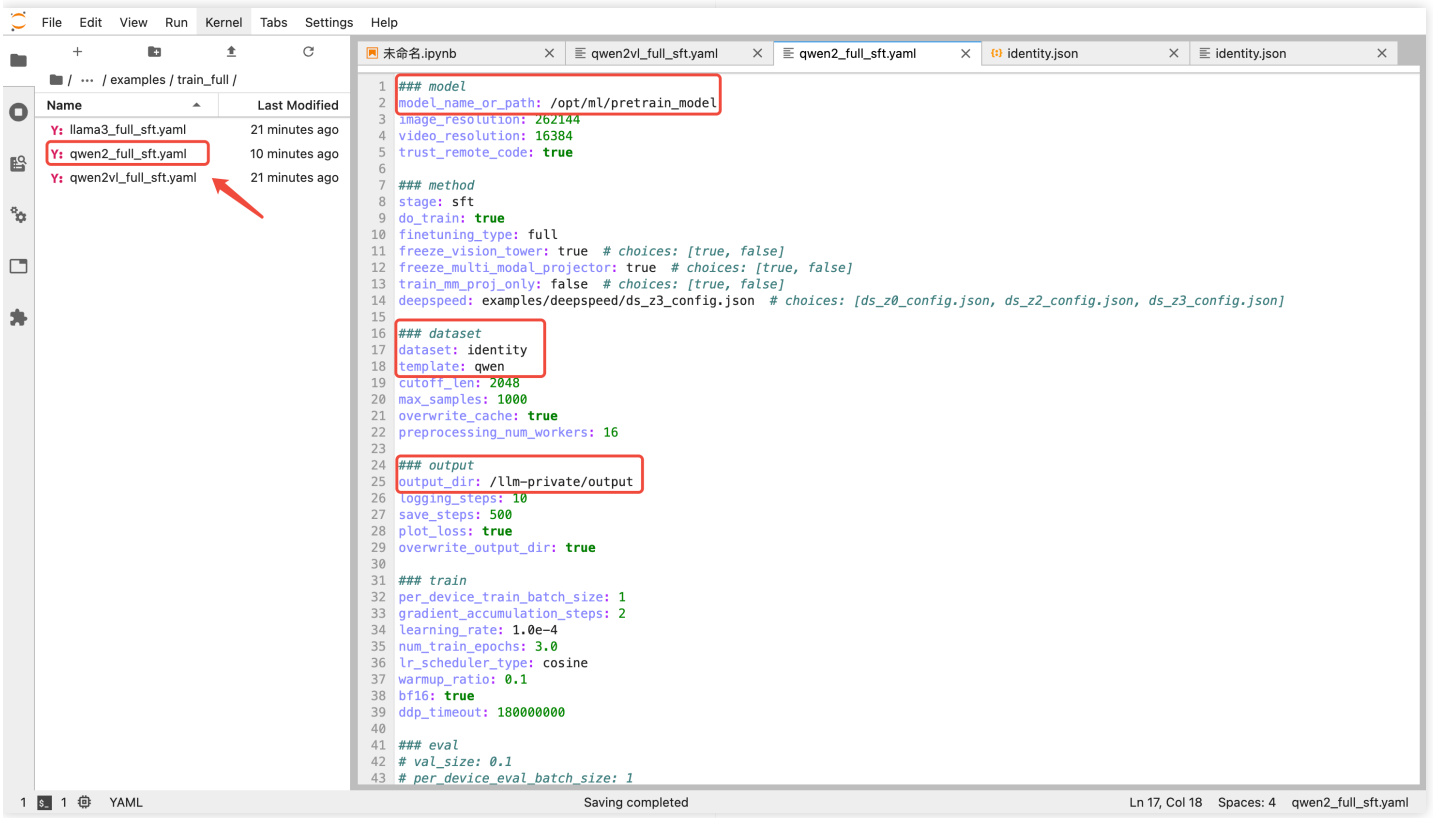
或者先将开源代码 下载 到本地再上传，上传开源 LLaMA-Factory 的源码到开发机中并解压。
如果您创建时打开了"SSH 连接"，您还可以使用 scp 将本地物料上传至开发机。

```
scp -r -P <port> Qwen2-0.5B-Instruct root@host:/home/tione/notebook/workspace
```

Tips: 如您的文件由 git 下载，您可以尝试删除目录下的 .git 目录，可减少上传的文件，提升上传速度。

修改训练配置文件

进入 LLaMa-Factory/examples/train_full 目录，复制 qwen2vl_full_sft.yaml 重命名为 qwen2_full_sft.yaml 并编辑相应内容。



训练使用的参考数据集在 LLaMa-Factory/data 下，这里用到的数据集为 identity.json (您可将其中的"{{name}}"和"{{author}}"替换以验证精调效果)，下面为主要需要修改的超参内容，您也可以根据官方文档按需调整：

```
# 预训练模型路径
model_name_or_path: /opt/ml/pretrain_model

# 训练数据
dataset：identity
template：qwen

# 训练输出路径
output_dir: /llm-private/output

# 训练轮数（为了快速验证可以适当调小）
num_train_epochs: 3.0
```

> 说明：

> 这里的目录对应到后续任务式建模相关容器挂载目录。

- 若需要自定义数据集，请修改 dataset 字段，并进入 LLaMa-Factory/data 目录修改 dataset_info.json 文件配置对应的数据集信息。
- 若需要修改其他训练参数，请自行修改。
- 若模型较小，容易过拟合，建议调小 learning_rate。

创建启动脚本

在 LLaMA-Factory 目录内创建一个 start.sh 文件，作为训练启动脚本，内容如下：

```
#!/bin/bash
TRAIN_CONFIG=$1
DISTRIBUTED_ARGS="
    --nproc_per_node ${NPROC_PER_NODE:-$(nvidia-smi -L | wc -l)} \
    --nnodes ${WORLD_SIZE:-1} \
    --node_rank ${RANK:-0} \
    --master_addr ${MASTER_ADDR:-127.0.0.1} \
    --master_port ${MASTER_PORT:-23456}
    "

set -ex
torchrun $DISTRIBUTED_ARGS src/train.py $TRAIN_CONFIG
```

上述启动脚本兼容任务式建模分布式训练启动。

任务式建模启动精调任务

创建训练任务

单击训练工坊 > 任务式建模 > 创建任务按钮，训练镜像选择一开始准备的镜像，训练模式选择“DDP”，资源配置按需选择。如果需要多机分布式训练，可以选择8卡并配置节点数大于1。

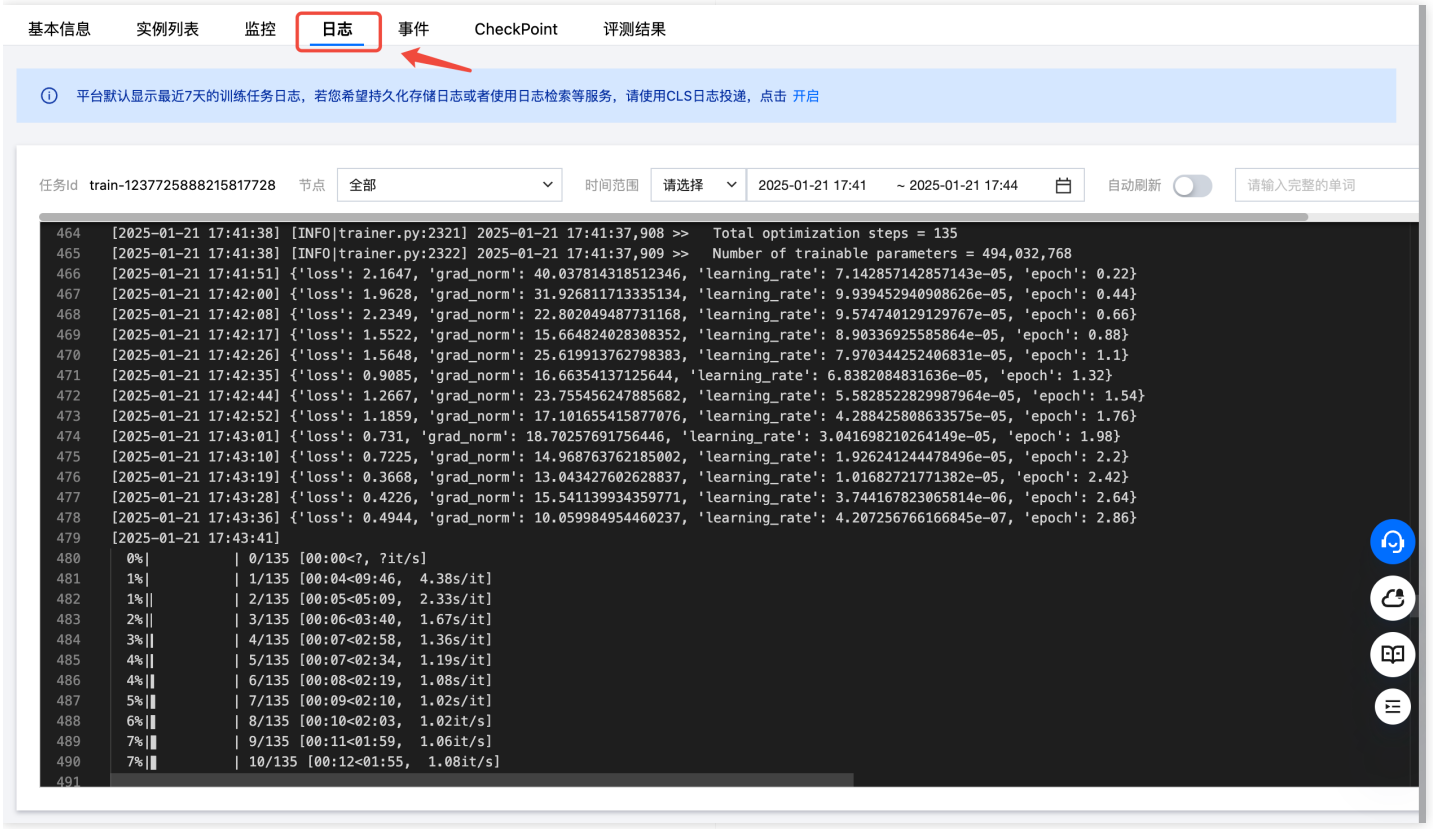
- 训练镜像：选择内置镜像 /PyTorch/tilearn-llm0.9-torch2.3-py3.10-cuda12.4-gpu，该镜像已默认配置了大模型训练运行时环境。
- 算力规格：请合理选择训练资源，如果需要多机分布式训练，可以选择8卡并配置节点数大于1。
- 存储路径配置：这里需要挂载的内容有训练代码、预训练模型与训练输出路径，其中容器的挂载路径对应上文训练配置文件中的相关参数（您也可将训练代码、预训练模型与训练输出放在同一个大目录下，并将根目录挂入容器，最后修改启动命令 cd 到 LLaMA-Factory 下即可）。
- 启动命令：（由于 LLaMA-Factory 主分支依赖比较高版本的 transformers 库，因此我们在启动命令中在线升级平台训练镜像的对应第三方库版本，最终写入如下启动命令）：

```
cd /opt/ml/code
pip3 install -r requirements.txt
```

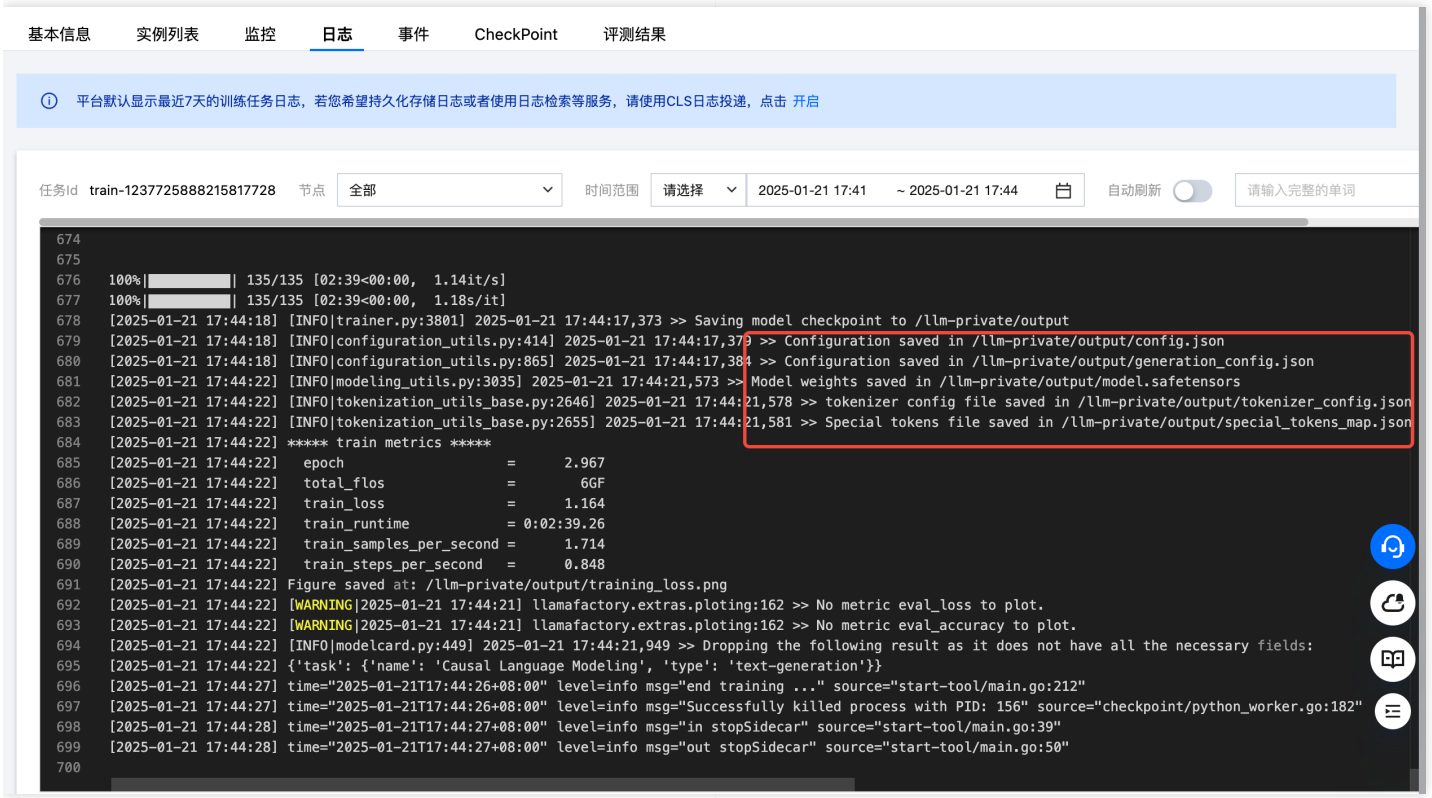
```
bash start.sh examples/train_full/qwen2_full_sft.yaml
```

查看训练状态

任务启动后，可以单击日志查看训练日志等信息。

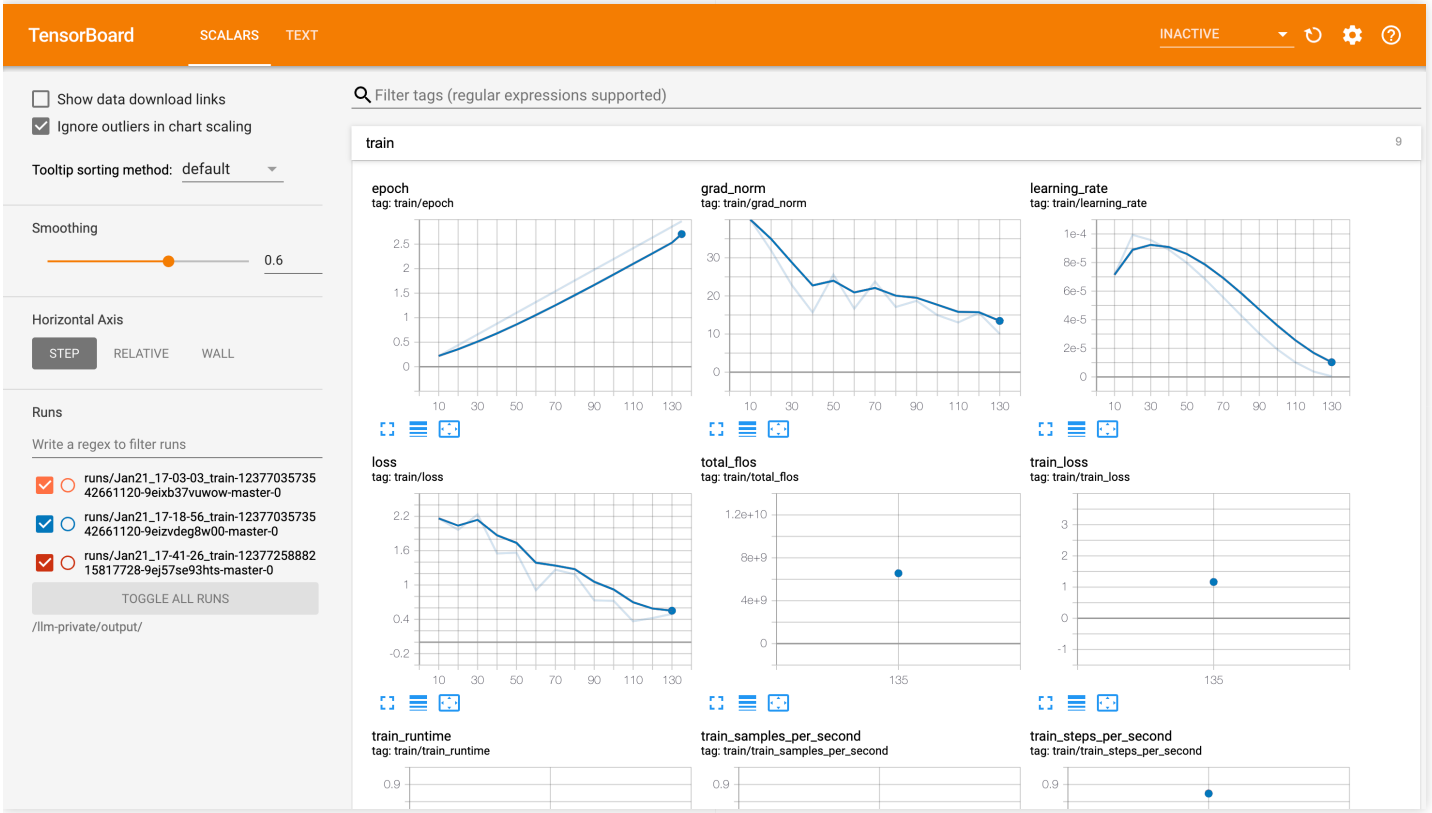


训练结束后，会将最终的 checkpoint 保存到挂载的输出目录 /project/output 中，该目录用于后续部署精调后的模型。



此外，您还可以单击"任务列表 > 操作栏中的 Tensorboard 按钮"，配置 Tensorboard 任务，路径为 CFS 中模型训练的输出路径 /project/output。

确定后即可查看训练过程损失梯度等状态。



精调后部署模型

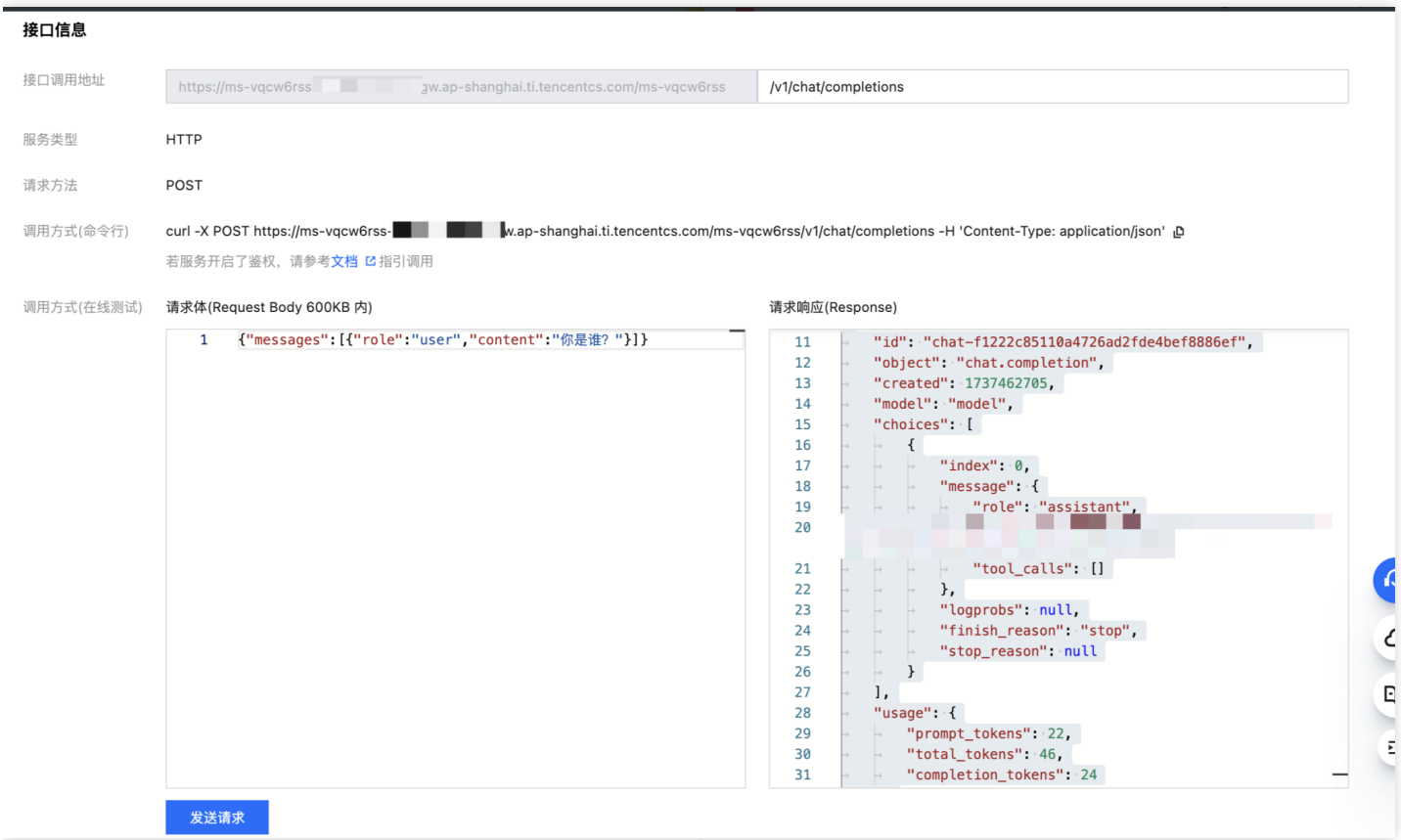
部署服务

进入模型服务 > 在线服务 > 新建服务按钮。

- 服务名称：自定义即可。
- 模型来源：选择 CFS，并选中模型存储的 CFS 实例，路径按实际训练输出路径填写，若需要选择其中的某一个 checkpoint，则填写到 checkpoint 这一级目录。
- 运行环境：选择内置的 LLM / angel-vllm(2.1) 镜像。
- 算力规格：根据实际的模型大小或拥有的资源情况选择，大模型推理时需要的机器资源与模型的参数量相关，推荐参考推理资源要求来评估。

调用服务——在线体验

回到上级菜单，单击服务的服务调用标签页面：在接口调用地址处输入 /v1/chat/completions 然后输入请求体，接口格式兼容目前流行的 OpenAI Chat Completions 接口，如图所示：



调用服务——本地测试脚本

调用地址：

本地测试脚本：

> 说明：
>
> 需要修改调用地址并准备测试数据。

```
# -*- coding: UTF-8 -*-
import time
import requests
import argparse
import json

from concurrent.futures import ThreadPoolExecutor, as_completed

def generate(prompt, log_stream=False):
    start = time.time()
    header = {
        "content-type": "application/json",
    }

    # print(prompt,type(prompt))
    data = {
        "messages": [{"role":"user","content":prompt}],        "temperature": 0.7,
        "max_tokens": 256,
    }
    result = requests.post(f"{args.server}/v1/chat/completions",
        headers=header,
        json=data,
        stream=True    )
    response = ""
    for part in result.iter_lines():
        if part:
            if "content" in part.decode("utf-8"):
                print(part.decode("utf-8"))
                content = json.loads(part.decode("utf-8"))["choices"][0]["message"]["content"]  # 字符串过滤
data: 之后转成 json 格式再取文本
                response += content
                if log_stream:
                    print(content, end = "", flush = True)
    if log_stream:
        print()
    end = time.time()
    return {"prompt": prompt, "generated_text": response, "time_cost": end - start}

def main():
    # 准备测试集
```

```
    prompts = []
    with open(args.test_data, "r", encoding="utf8") as data:
        for line in data.readlines():
            line_dict = json.loads(line.strip())
            prompt = line_dict['question']
            prompts.append(prompt)
    # 或者使用几个简单的测试样例：
    prompts = ["你好！","你是谁？"]
    task_list = []
    t1 = time.time()
    with ThreadPoolExecutor(max_workers=args.concurrency) as executor:
        for prompt in prompts:
            task_list.append(executor.submit(generate, prompt, args.verbose))
        for item in as_completed(task_list):
            result = item.result()
            print(result)
    total_time = time.time() - t1
    req_per_sec = len(prompts) / total_time
    print(f"total_time={total_time:.4f}s, requests/s={req_per_sec:.4f}")

if __name__ == '__main__':
    parser = argparse.ArgumentParser()
    # 调用地址换为在线服务的地址
    parser.add_argument("-s", "--server", type=str, default="https://ms-xxxxxxxx-yyyyyy.cs.com/ms-xxxx
xxx", help="推理服务地址")
    parser.add_argument("-d", "--test-data", type=str, default="./test.jsonl", help="测试jsonl文件")
    parser.add_argument("-c", "--concurrency", type=int, default=4, help="请求并发数")
    parser.add_argument("-v", "--verbose", action="store_true", help="打印流式详情")
    args = parser.parse_args()
    main()
```

回答示例：

```
{'prompt': '你是谁？', 'generated_text': '您好，我是 xiatian，一个人工智能助手，我可以帮人们解决各种语言
相关的问题和任务。', 'time_cost': 1.1000895500183105}
total_time=1.1006s, requests/s=0.9086
```

内置训练镜像列表

简介

TI-ONE 内置了主流的 Pytorch 深度学习框架，其中 tilearn-llm 是为大模型定制的训练加速组件，已内置在平台内置镜像中，同时支持开发机和任务式建模。

内置通用镜像

框架	镜像名称	支持的训练模式	备注
PyTorch	tilearn-llm0.9-torch2.3-py3.10-cuda12.4-gpu	DDP、MPI、Ray	支持的核心库：Python 3.10，CUDA 12.4，jupyterlab 2.3.2，torch 2.3.0a0+40ec155e58.nv24.3，transformers 4.39.3，deepspeed 0.13.4，tilearn-llm 0.9.9，tilearn.ops 0.2.2.175，angel-vllm 0.4.2，ray 2.42.0 支持的模块：任务式建模和开发机

自定义训练镜像规范

若平台内置镜像不满足您的需求，您也可以使用自定义镜像创建训练任务和开发机实例，以下是自定义镜像的 Dockerfile 示例：

基本镜像规范

若想要自定义镜像能在任务式建模启动训练任务，要求镜像安装 openssh-server 组件。若需要使用 Git 存储，同时要求镜像安装git组件，示例如下：

```
# 基础镜像请自行修改
FROM ubuntu:20.04

# 安装 openssh-server
RUN apt-get update && apt-get install -y openssh-server git && apt-get clean && mkdir -p /var/run/sshd
```

> 说明：
>
> 若基础镜像是 centos 系统的话，使用 yum/dnf 进行包管理，请自行调整安装命令。

开发机镜像规范

若想要自定义镜像能在开发机启动实例，除了要满足上述基本镜像规范以外，建议安装python环境，示例如下：

```
# 基础镜像请自行修改
FROM ubuntu:20.04

# 安装 openssh-server
RUN apt-get update && apt-get install -y openssh-server git && apt-get clean && mkdir -p /var/run/sshd

# 安装 python3、pip3
RUN apt-get update && apt-get install -y python3.8 python3.8-distutils curl && \
    curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py && \
    python3.8 get-pip.py && rm -f get-pip.py
```

> 说明：
>
> - 若基础镜像已经包含了对应的包，请自行跳过对应安装命令。
> - 这里 /home/tione/notebook 是 Notebook 默认的磁盘挂载路径，镜像中是否有该路径不影响平台中使

用。

说明：

- 推荐自定义镜像统一使用云平台的软件源，以支持更快的安装速度，上述示例已包含配置方法。若要配置其他软件源，可进一步参考：云平台软件源加速软件包下载和更新。
- 若需要使用 HCC-GPU 型机型进行多机训练，推荐镜像安装 TCCL 插件以优化云平台星脉网络下的 RDMA 通信，上述示例已包含配置方法。
- 当前不支持训练镜像在 bashrc 等 bash 配置文件声明的变量。

Angel 推理加速功能介绍

LLM推理加速

Angel-vLLM 是 AI 加速团队基于开源 vLLM 深度优化的推理加速框架。在保持和社区 vLLM 相同使用接口和完全兼容社区 vLLM 功能的同时，具备如下特点：
第一，功能更丰富。相比 vLLM 社区开源版本，Angel-vLLM 提供了 INT8，INT4 在线量化，KV Cache INT8 量化，lookahead 并行解码，PP+TP 模型并行等功能(部分功能需要0.4.2或更新版本框架支持)。
第二，性能更强大。相比 vLLM 社区开源版本，Angel-vLLM 量化不仅节省显存，也可以降低延迟，提升吞吐。lookahead 并行解码经过实际业务打磨，在 RAG 场景提升明显。
第三，精度更对齐。Angel-vLLM 生成结果经过大量上线业务检验，可以做到和 HuggingFace 生成结果完全对齐或精度保持基本不变。

Angel-vLLM 一键式模型量化加速

支持多种量化加速方案，包括：

- int8 weight-only 量化
- int8 smoothquant 量化
- int4 weight-only 量化

结合 int8 kv-cache 量化技术，在保持算法效果无明显下降情况下，大幅提升推理性能，降低部署成本。

使用方法

相比原始推理代码，只需要额外配置 quantization 和 kv_cache_dtype 两个参数：

```
# 加载大模型
from vllm import LLM
model = LLM(model=args.model,
          trust_remote_code=True,
          quantization=args.quantization,     # "ifq代表int8 weight-only量化，ifq_int4代表int4 weight-only量化，smoothquant代表int8 smoothquant量化，默认None代表不使用量化加速"
          kv_cache_dtype=args.kv_cache_dtype, # "int8代表使用int8精度压缩kv-cache，默认auto代表不使用kv-cache量化"
          dtype=torch.float16)

# 定义采样参数
sampling_params = SamplingParams(top_k=50, top_p=0.8, max_tokens=args.output_len)

# 推理
outputs = model.generate(prompt_token_ids=input_ids, sampling_params=sampling_params)
```

备注：

- ifq 量化下，输入为原始 fp16 模型，在模型加载过程中完成量化。
- ifq_int4 和 smoothquant 量化下，输入为 PTQ 校准后的模型。
- kv-cache 量化当前仅在 smoothquant 下支持。

加速效果

参考落地业务典型使用场景，性能测试配置：

- Prompt 长度：1000
- 输出 Token 数：100
- 并发数：8
- 硬件：A800、A100、L20、A10
- 模型参数量：70B，13B，7B

  性能测试数据：

测试环境	测试模型	FP16		INT8 weight-only				INT8 smoothquant			
		首Token延迟 (ms)	吞吐 (tokens/s)	首Token延迟 (ms)	首Token加速比	吞吐 (tokens/s)	吞吐加速比	首Token延迟 (ms)	首Token加速比	吞吐 (tokens/s)	吞吐加速比
A100-40GB	Baichuan2-13B	954.1	210	1037.8	0.92	267	1.27	596.8	1.60	321	1.53
L20-48GB	Baichuan2-13B	1940.1	117	2075.5	0.93	153	1.31	1111.1	1.75	201	1.72
A10-24GB	Baichuan2-7B	1933.3	129	1982.7	0.98	169	1.32	1065.2	1.81	224	1.74

测试环境	测试模型	INT8 weight-only		INT4 weight-only			
		首Token延迟 (ms)	吞吐 (tokens/s)	首Token延迟 (ms)	首Token加速比	吞吐 (tokens/s/卡)	吞吐加速比
A800-80GB*2	Qwen2-70B	1631.4	78	1441.5	1.13	102	1.31
A800-80GB	Qwen2-70B	OOM	OOM	2875.3	0.57	118	1.52

Angel-vLLM lookahead 并行解码

将 vLLM 推理代码改造成使用 lookahead 并行解码，只需要添加3个参数。
例如，原始推理代码如下:

```
sampling_params = SamplingParams(seed = 10, top_k=50, use_beam_search = False)
llm = LLM(model = model_path,
        enforce_eager = True,
        quantization = None)
output = llm.generate(prompts, sampling_params = sampling_params)
```

改造成使用 lookahead 并行解码，只需要添加三个选项：

```
llm = LLM(model = model_path,
        enforce_eager = True,
        use_lookahead = True,       #开启lookahead并行解码
```

use_v2_block_manager = True, #lookahead并行解码需要使用block_manager v2
num_speculative_tokens = 12, #一次并行解码长度，单BS可以设置成12， 多BS可以适当降低一次解码长度，例如设置成8或6
quantization = None) #并行解码也支持int8量化，以及smoothquant量化，设置quantization='ifq' or 'smoothquant'打开相应
量化。模型量化使用方法参考模型量化部分。

lookahead 解码在客服 RAG 场景（输入1000左右，输出100左右）性能测试数据：

测试环境	测试模型	量化方法	并发数	解码长度	正常解码		lookahead解码		QPM加速比
					首token时间	QPM	首token时间	QPM	
L40-48G	Baichuan2-13B	weight-only	1	12	0.39	16.2	0.39	28	1.73
L40-48G	Baichuan2-13B	weight-only	8	6	0.69	66.7	0.73	91.4	1.37
L40-48G*8	Qwen-72B	weight-only	1	12	0.69	17.4	0.7	29	1.67
L40-48G*8	Qwen-72B	weight-only	8	4	1.19	62.6	1.25	71.9	1.15
A100-40G	Baichuan2-13B	weight-only	1	12	0.27	29.6	0.28	50	1.69
A100-40G	Baichuan2-13B	weight-only	8	4	0.45	106	0.49	125	1.18
A100-40G*4	Qwen-72B	weight-only	1	12	0.42	14	0.43	28.6	2.04
A100-40G*4	Qwen-72B	weight-only	8	6	0.67	64.2	0.73	91.7	1.43
L40-48G	Qwen1.5-14B	smoothquant	1	12	0.17	19.2	0.23	40.7	2.12
L40-48G	Qwen1.5-14B	smoothquant	8	7	0.2	114.2	0.36	167	1.46

Angel-vLLM NGram 并行解码

将 vLLM 推理代码改造成使用 NGram 并行解码，可以通过添加如下选项开启 NGram 并行解码能力：

```
llm = LLM(model = model_path,
    enforce_eager = True,
    use_v2_block_manager = True,
    num_speculative_tokens = 12,
    ngram_prompt_lookup_max = 12,
    ngram_prompt_lookup_min = 2,
    speculative_model = "[ngram]",
    quantization = None)
```

NGram 通过单次请求的数据对未生成 Token 进行匹配填充。
相比开源实现，Angel-vLLM NGram 解码可以做到生成结果和 HF 完全对齐。

操作指南
大模型广场

概述

大模型广场是 TI 平台 的内置大模型库，预置多种预训练大模型及指令微调大模型，覆盖各类下游任务，如多轮对话、逻辑推理、内容创作等。
用户可一键将内置大模型部署为在线服务，通过网页问答快速直观地体验大模型的效果；也可一键发起基于内置大模型的精调任务，将模型优化为可满足垂直场景需求的生产级模型。

内置大模型

大模型广场已内置如下大模型，最新支持的模型清单以大模型广场页面展示为准。

- Hunyuan大模型：包括开源的 Hunyuan-Large 系列，以及闭源版本。
- 精选通用大模型：包括主流的开源大模型，如 DeepSeek 系列等。

功能说明

大模型广场提供以下主要功能。

浏览内置大模型

在大模型广场页面，以卡片形式展示模型清单。

- 模型标签：按任务类型（如文本分类、翻译、问答等）、语言（如中文、英文）、框架（如PyTorch）等维度对模型进行分类，以标签形式展示，方便快速分类查找。
- 模型搜索：支持基于关键字的搜索功能，用户可以通过模型名称进行模糊查找。

发起部署和精调

单击广场中的卡片，可进入模型详情页，展示模型的更多详细信息，并支持快速试一试、精调训练等功能。

- 模型介绍：包括模型描述、系列模型清单、模型归属等信息。
- 快速试一试：单击新建在线服务，可跳转至在线服务模块，一键发起模型部署。
- 精调训练：单击新建训练任务，可跳转至任务式建模模块，一键发起精调训练。模型精调可显著提升模型在特定场景和任务中的性能，从而进一步满足生产级需求。

任务式建模
任务式建模简介

任务式建模提供通过向导式的训练任务提交方式进行模型构建，可直接基于平台内置镜像快速使用主流高性能及分布式训练框架提交训练任务，也可通过自定义训练镜像启动任务，其功能模块详细描述为：

- 训练任务创建：基于内置框架或者自定义训练镜像进行训练任务创建提交，支持使用CFS文件系统路径或者COS存储桶作为训练输入路径，支持选择Git存储库作为代码输入，支持填写训练任务算法参数，支持日志投递和 VPC 绑定。
- 任务运行管理：提供启停，删除等训练任务常规管理操作，同时提供任务复制功能，快速启动多种不同配置的训练任务，比对模型训练效果。
- 任务运行监控：提供训练运行日志，训练资源消耗可视化监控。
- Tensorboard：任务支持启动 Tensorboard 监控。

创建任务

创建步骤

填写基本信息

1. 登录TI-ONE 控制台，进入训练工坊 > 任务式建模，单击新建，开始创建训练任务。
2. 在基本信息配置页，填写如下信息：

- 任务名称：仅支持中英文、数字、下划线"_"、短横"-"，只能以中英文、数字开头。
- 地域：训练任务所在的地域，默认为当前列表页所在的地域。
- 训练镜像：可选择平台内置训练镜像、自定义镜像和内置大模型，其中内置训练镜像请查看内置训练镜像列表；自定义镜像支持选择容器镜像服务的镜像或者填写外部镜像地址（若为私有镜像，需要输入用户名和密码），自定义镜像规范请查看自定义训练镜像规范；内置大模型训练使用方式请查看精调内置开源大模型。
- 训练模式：不同训练框架支持的训练模式请查看内置训练镜像列表。
- 资源组：请选择已创建的资源组

  > 说明：
  >
  > a. 选择资源组后会立即展示该资源组所剩的 GPU 概览信息，包括各卡型号的 GPU 总卡数，整机卡数和非整机卡数（碎卡数），可帮助用户快速了解选中资源组的 GPU 分布情况，根据当前任务场景选择使用整机资源还是非整机（碎卡）资源，可有效降低整体资源的碎片化情况，提升 GPU 总体利用率。
  > b. 点击 查看详情 即会在当前页面右侧展示详细资源看板，看板中展示了各个卡类型的节点剩余可用和总资源；点击下拉后可展示当前节点正在运行的所有任务/服务，可帮助用户快速了解节点资源的占用情况，以便团队间进行资源使用协商。

- 标签：可为任务创建标签，一个任务可添加多个标签。
- 描述：可添加最多500字的备注描述。
- CLS 日志投递：CLS 日志投递默认关闭，TI 控制台会默认展示 15 天的日志，若您期望持久化存储日志，获得日志检索等服务，可以开启 CLS 日志投递，打开 CLS 后可以将日志投递至 CLS 服务（需要确保 CLS 服务已完成开通），CLS 产品介绍请查看 CLS 介绍。

- 自动重启：您可以对任务配置自动重启策略，您需要配置最大重启次数，最高为10次，超过最大重启次数后，会将任务直接标记为异常；当前任务自动重启的触发条件为任务运行过程中发生异常退出。任务自动重启的事件信息可在 任务详情 > 事件 页面中查看。

填写任务配置信息

任务配置页面需要配置本次训练任务的算法、数据、输入输出等信息，具体配置项说明如下：

1. 存储路径设置：存储类型支持 COS和CFS（包含 CFS Turbo 类型），每设置一个存储路径，均支持选择该路径的用途，包含用户自有模型，用户自有代码，用户自有数据，训练数据及其他（其中当选择存储类型为 CFS 时，用途还支持选择平台内置模型，可将 TIONE 平台内置 CFS 中的模型直接挂载到训练容器）：
   - 若选择 COS，则需要选择数据所在的 COS 路径，选择的路径下的文件会自动下载到容器内路径；
   - 若选择 CFS ，则需要下拉选择 CFS 文件系统，同时填写该CFS系统上的源路径，以及容器内挂载路径；

2. Git 存储：可以选择 Git 存储库，并且配置容器内存储路径，任务启动时会将该存储库下的文件下载到指定的存储路径中。您需要首先在训练工坊-Git 存储库中创建好存储库。

3. 启动命令：您需要填写程序入口命令，支持填写多行，默认工作目录为/opt/ml/code。

4. 调优参数：填写的超参数 JSON 会保存为 /opt/ml/input/config/hyperparameters.json 文件，您的代码需自行解析。

5. 训练输出：选择您需要保存训练输出的 COS 路径，平台会默认将 /opt/ml/output 路径下的数据定期上传至输出 COS 路径；若您需要将训练输出的模型一键发布至模型仓库，则需要将模型输出保存至 /opt/ml/model 路径下，平台会在训练结束后将该路径下的数据上传至 COS 路径；若您选择的是 CFS 等文件系统作为训练存储，您也可以不配置训练输出，直接将训练输出写到挂载的 CFS 文件路径中。

   另外，在您配置任务的过程中，底部会实时显示您当前任务配置的价格，请注意关注。所有信息配置完成后，即完成了任务创建。

内置大模型预置流程说明

任务式建模内置了多种大模型精调模板，您可以直接一键启动内置大模型精调任务，详细的最佳实践可以参考精调内置开源大模型。以下是内置字段说明：

预置存储路径设置

第一行“平台 CFS”：系统默认为您配置了精调该大模型的配套训练代码。
第二行“平台 CFS”：系统默认为您配置了一份精调该大模型的示例数据；若您希望使用自定义业务数据精调该大模型，可删除此行，并在底部添加其他存储来源。
第三行“平台 CFS”：系统默认为您配置了平台内置模型。

第四行 “用户 CFS”：此处需选择您的 CFS 文件系统和源路径，“容器挂载路径”为系统默认填充您无需修改。若您需要使用 其他CFS文件系统 作为训练输出，则您可以删除本行再添加。

注意：若您使用自己的业务数据进行精调，需要使用如下平台约定的格式或者遵循 llamafactory 的 dataset_info.json 数据配置文件描述，详情请 单击跳转 查看。

预置启动命令

平台默认填充了启动命令，一般情况下您无需修改。

预置调优参数

平台提供多个预置参数，您可以直接修改超参 json 迭代模型。以下是超参释义：

超参	默认值	解释
Epoch	2	训练过程中的迭代轮次
BatchSize	1	每轮训练迭代中的样本数。BatchSize 越大，训练速度越快同时内存占用越高
LearningRate	1e-5	梯度下降过程中更新权重的超参，过高导致模型难以收敛，过低导致模型收敛速度过慢
Step	750	每跑多少个Step保存一次模型的checkpoint，保存checkpoint越多需要的存储空间越大

超参	默认值	解释
UseTilearn	true	是否要开启加速，"true/false"，设置为"true"会默认启用加速框架训练，其中3d并行加速需8卡以上，需要进行PP和TP参数设置，可参考angel-tilearn文档；设置为"false"会默认使用开源加速框架进行训练。仅部分模型开放
FinetuningType	Lora	用户可自定义选择精调训练模式“Lora/Full”，LoRA 在固定预训练大模型本身参数的基础上，对权重矩阵进行低秩分解，训练过程中只更新低秩部分参数；FULL 在精调过程中会全量更新模型参数，需要的训练资源更多
MaxSequenceLength	2048	最大文本序列长度，可根据您的业务数据长度进行合理设置。例如，如果大部分业务数据长度都在2048以下，则可设置 MaxSequenceLength 为 2048，大于2048的数据都会被截断为2048，可降低 GPU 显存压力
GradienAccumulationSteps	1	huggingface trainer参数，默认为1，提升batchsize
GradientCheckPointing	True	huggingface trainer参数，默认True，时间换显存的策略，开启后优化显存但训练速度变慢
DeepspeedZeroStage	z3	DeepSpeed ZeRO 阶段配置，可选值["z0", "z2", "z2_offload", "z3", "z3_offload"]，默认值z3；仅部分模型开放
ResumeFromCheckpoint	Ture	是否自动从已有的 checkpoint 文件恢复训练，默认值为 True，表示若输出目录存在 checkpoint 文件，从最新的 checkpoint 恢复继续训练；设置为 False 表示将重新训练。设置为 False 且输出目录非空，则会报错，建议训练输出路径使用空目录，若需要开启强制覆盖需要手动增加一条"overwrite_output_dir": true 参数开启覆盖文件
TilearnHybridTPSize	1	Tilearn 3D 并行参数，TP并行的维度，默认为1；仅部分模型开放
TilearnHybridPPSize	1	Tilearn 3D 并行参数，PP并行的维度，默认为1；仅部分模型开放

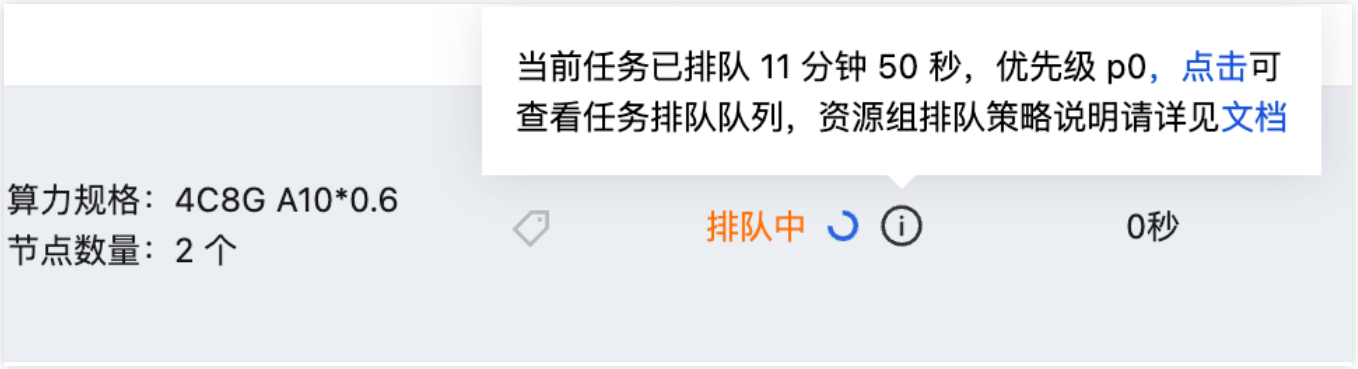
任务管理

简介

任务创建完成后，会在任务列表页面展示该条训练任务记录，列表展示了任务名称，训练镜像，机器来源，占用资源，标签，状态（任务状态有提交中，排队中，启动中，运行中，异常，停止中，已停止，已完成），运行时长，创建者，训练开始时间，更新时间，监控和操作。操作包括复制任务，Tensorboard，停止/启动任务和删除任务等 。

任务各个状态简介：

- 提交中：任务成功发起提交请求至进入排队队列前。
- 排队中：任务正常提交后进入排队队列。当状态为排队中时，点击可展示任务的排队时长和优先级。



- 启动中：任务成功出队，调度到了资源，开始初始化任务。
- 运行中：任务成功加载，进入运行状态。
- 异常：任务异常退出。
- 停止中：用户手动停止任务，回收资源等过程。
- 已停止：用户手动停止任务，任务正常终止。
- 已完成：任务训练完成，正常退出。

查看我的任务

列表页右上角有查看我的任务快捷选项，勾选后仅展示当前登录的子账号创建的任务。

复制任务

当用户需要进行多个训练任务进行对比，以比较不同数据集或者不同超参配置的训练效果时，可选择复制训练任务，单击复制训练任务将跳转到创建任务窗口，用户可在原任务配置信息基础上进行简单修改，即可快速创建一个任务。

Tensorboard

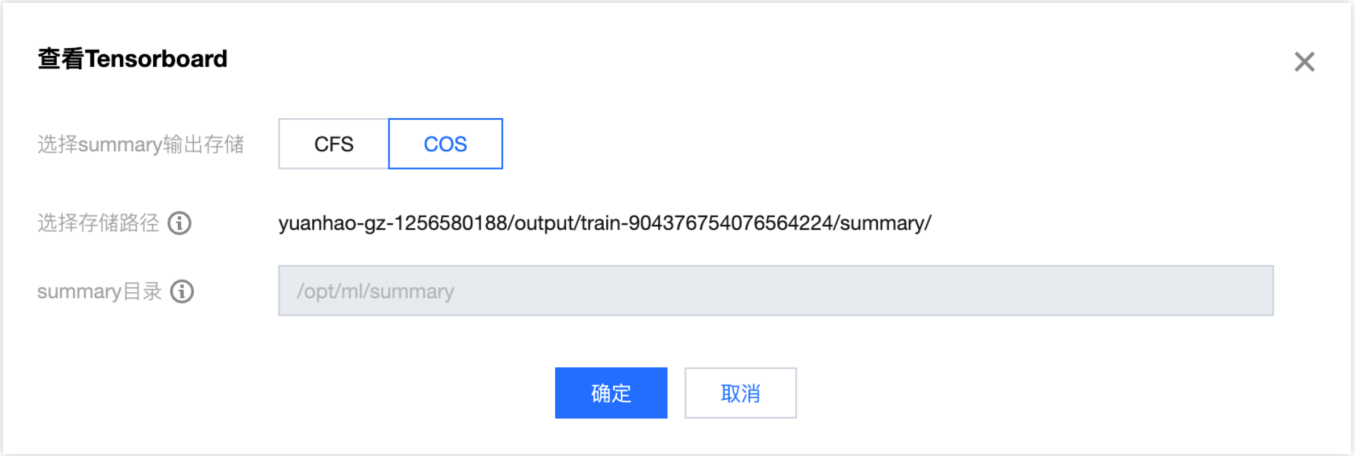
平台支持读取 CFS 和 COS 中的 summary 数据生成 Tensorboard 监控面板，具体操作步骤如下：

- 单击任务列表 > 操作栏中的 Tensorboard 按钮，开始配置 Tensorboard 任务。

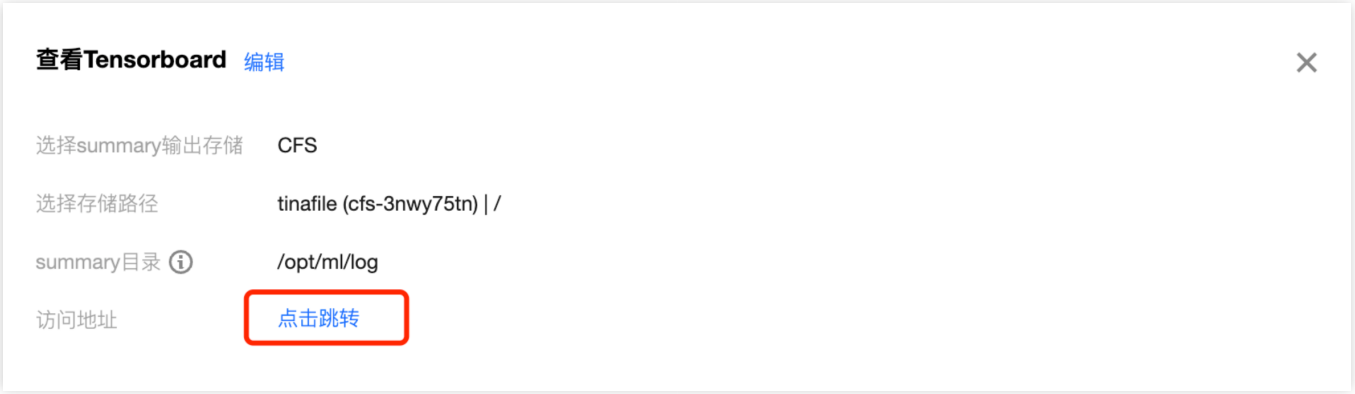
- 若您的训练输出文件存储在 CFS 中，则选择 CFS 为 summary 输出存储，选择存储路径为您当前训练任务配置的 CFS 文件系统和源路径（若您配置了多个，则需要选择其中一个），填写您的 summary 数据所在的容器内目录（平台默认会将您的 CFS 系统挂载的容器本地路径展示出来，您只需要填写 summary 文件具体所在的子目录），如下图所示：

- 若您的训练输出文件存储在 COS 中，则选择 COS 为 summary 输出存储，需要注意的是，如果选择了 COS，存储路径默认为任务配置的训练输出 COS 路径 + /< job_id> + /summary，summary 目录默认为/opt/ml/summary/

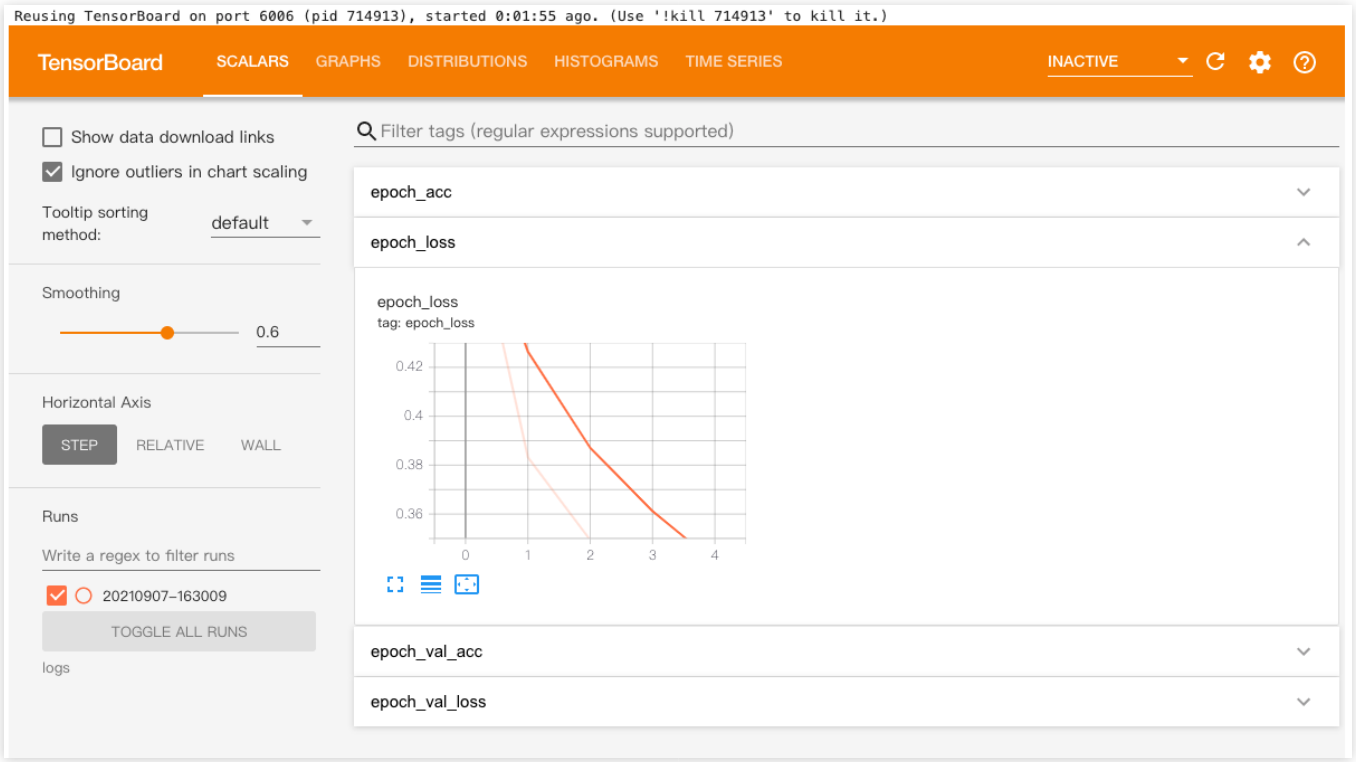
（容器内路径），用户无需配置，因此您需要在训练代码中提前将 summary 监控数据输出到/opt/ml/summary/中，如若不然，则无法创建 Tensorboard 面板。



- 完成配置后，单击确定，即可进入 Tensorboard 信息页，此时单击页面中的 点击跳转 ，即可跳转至 Tensorboard 面板。



启动Tensorboard 可能需要1 - 2分钟，在此期间，请不要关闭重定向页面。



停止/启动任务

正在训练的任务用户可以选择手停，停止后的任务可进行重新启动。

删除任务

训练完成/已停止的任务用户可手动删除记录。

任务详情

单击训练任务名称，可进入任务详情页。

基本信息

基本配置页显示了训练任务的基本配置信息、作业参数信息和资源配置信息。其中任务的标签可以进行更新。

实例列表

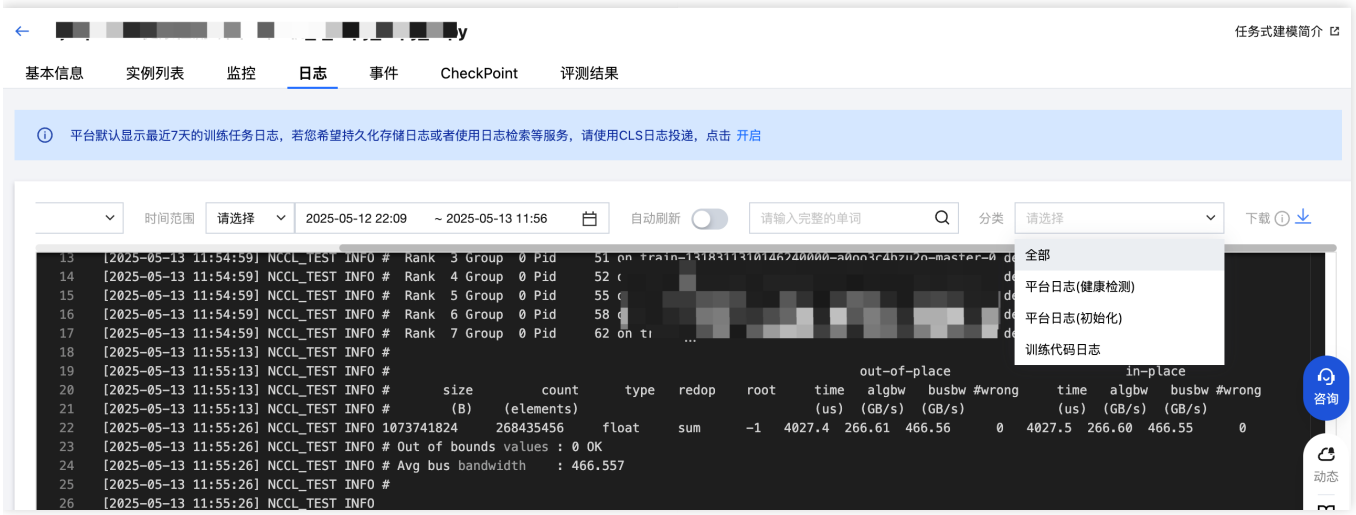
实例列表页展示了当前训练任务的实例清单，在此页面可查看实例 ID，占用资源，状态。此外，训练实例可支持直接进入实例容器，用户可在页面上单击进入容器，进入 webshell 页面，用户可执行命令行操作训练实例容器。

监控

资源监控信息展示了训练任务的 CPU 使用率，MEM 使用率，MEM 使用量，GPU 使用率，显存使用率和显存使用量信息；整卡任务会展示细粒度卡维度监控指标，同时若您提交了整机训练任务，还可以查看节点 RDMA 使用监控信息。

训练日志

训练任务页展示了实例维度的训练日志，日志可实时刷新查看，同时支持全文检索，亦可通过时间选择查看该训练任务的历史日志。平台默认为用户存储最长15天的日志，若您需要持久化存储使用，请使用 CLS 日志投递。用户也可在当前页选择开启 CLS 日志投递或者关闭 CLS 日志服务。同时，平台支持日志进行分类展示，可按照 平台日志(初始化)、训练代码日志进行日志过滤，方便用户针对性的查看所关心的日志分类。



事件

平台支持查看任务运行过程中产生的事件，用户可基于事件进行问题的诊断。

分布式训练使用指引

任务式建模支持提交多机多卡分布式训练任务，TI-ONE 平台支持多种分布式训练模式，包含 DDP、Ray、MPI、PS-Worker 等，本文档将阐述不同训练方式在 TI-ONE 平台中的使用方法。此外，大规模分布式训练任务需要使用 RDMA 技术来获得高吞吐、低延迟的网络通信，从而提升训练效率。本文最后也将介绍如何在 TI-ONE 平台上基于 RDMA 的高性能 GPU 实例进行分布式训练。

一、支持的分布式训练模式及使用说明

TIONE 任务式建模支持多种分布式训练模式，包含 DDP、Ray、MPI、PS-Worker 等，以下是平台支持的分布式训练模式及其使用场景概述：

分布式训练模式	模式介绍
DDP	PyTorch DistributedDataParallel（DDP）是一种数据并行的分布式训练方法。通过 DDP 创建多个进程进行模型训练，通过 ring-all-reduce 的方法做进程通讯，完成梯度的交换及参数更新。
MPI	MPI 是一种基于信息传递的并行编程技术。平台支持用户发起 MPI 的分布式训练任务，同时也支持常见的 Horovod、DeepSpeed 等基于 MPI 的训练框架。
Ray	Ray 是一个开源的分布式计算框架，能够简化分布式机器学习的开发和部署。Ray 提供了一套 API 和基础设施，使得开发人员可以轻松地将单机训练代码扩展到分布式环境，支持数据并行和模型并行等多种并行策略。Ray在大模型领域也常用作加速强化学习训练过程的分布式框架。
PS-Worker	PS（ParameterServer）参数服务器训练是一种常见的数据并行方法，用于在多台机器上扩展模型训练。训练集群由 Worker 和 ParameterServer(ps) 组成。参数保存在ps上，在每一轮训练中，ps 将参数分发给 worker，worker 完成计算后将梯度回传给 ps 进行更新。

DDP 模式使用说明

PyTorch DistributedDataParallel（DDP）训练模式支持在 PyTorch 中进行数据并行训练。数据并行模式可以跨多个进程同时处理多个数据批次，每个进程的输入数据批次不重叠，每个进程计算梯度并使用 ring-all-reduce 算法完成与其他进程的同步。

使用方式

1. 在“任务式建模”界面创建训练任务时，选择训练模式为 DDP，并配置单节点资源和节点个数。其中启动命令会在每个节点上被执行。

2. DDP 训练模式包含两种角色 Master 和 Worker。其中编号为0的是 Master（对应环境变量中 RANK=0），承担保存模型的任务。

3. TI 平台会根据任务配置创建对应的实例，并注入相关的环境变量，如任务中包含的实例组信息，以及当前实例的角色。Worker 会等待 Master 正常启动，网络通畅。以下是任务式建模启动时默认注入的环境变量列表：
内置环境变量

变量名	变量描述	示例
NODE_LIST	训练任务公共环境变量：任务节点的列表和节点的 GPU 卡数信息	NODE_LIST=timaker-xxxyyy-launcher.training-job.svc.cluster.local:1,timaker-xxxyyy-worker-0.training-job.svc.cluster.local:1
INDEX	训练任务公共环境变量：当前节点信息在 NODE_LIST 的索引，从0开始	INDEX=1
MASTER_ADDR	DDP 训练任务的 master 节点IP	MASTER_ADDR=10.35.110.11
MASTER_PORT	DDP 训练任务的 master 节点端口	MASTER_PORT=23456
WORLD_SIZE	DDP 训练任务的节点数	WORLD_SIZE=2
RANK	DDP 训练任务的当前节点	RANK=1
GPU_NUM	任务包含的 GPU 卡总数	GPU_NUM=2
GPU_NUM_PER_NODE	单个节点的 GPU 卡数	GPU_NUM_PER_NODE=1

4. 训练过程中任意实例退出码非0则训练任务失败。所有实例成功则训练任务成功。

示例启动方式

启动 DDP 分布式训练的命令示例如下：

```
python -m torch.distributed.launch --nproc_per_node $GPU_NUM_PER_NODE --nnodes $WORLD_SIZE --node_rank $RANK --master_addr $MASTER_ADDR --master_port $MASTER_PORT
```

DDP 分布式训练参数与平台环境变量对应关系如下表所示：

变量名	变量描述
nproc_per_node	单个实例（机器）上运行的进程数，使用 GPU 时通常为每台机器上的 GPU 卡数，对应环境变量 GPU_NUM_PER_NODE 的值。
nnodes	对应环境变量 WORLD_SIZE 的值。
node_rank	对应环境变量 RANK 的值。
master_addr	对应环境变量 MASTER_ADDR 的值。
master_port	对应环境变量 MASTER_PORT 的值。

更多 DDP 训练参数请参考官网文档 TORCHRUN。

对于启动命令中涉及的内置环境变量，平台会在启动任务式建模任务时注入。而在 Notebook 或者本地调试代码时，需要开发人员先为对应环境变量赋值。为了方便调试使用，可以为对应环境变量设置默认值，示例如下：

```
MASTER_ADDR=${MASTER_ADDR:-localhost}
MASTER_PORT=${MASTER_PORT:-23456}
NNODES=${WORLD_SIZE:-1}
NODE_RANK=${RANK:-0}
GPU_PER_NODE=${GPU_NUM_PER_NODE:-$(nvidia-smi -L | wc -l)}
python -m torch.distributed.launch --nproc_per_node $GPU_PER_NODE --nnodes $NNODES --node_ra
nk $NODE_RANK --master_addr $MASTER_ADDR --master_port $MASTER_PORT
```

MPI/Horovod 模式使用说明

MPI 是一种用于分布式并行训练的消息传递标准。平台支持用户发起 MPI 模式的分布式训练任务，并也支持常见的基于 MPI 通信的训练框架如 Horovod。而 Horovod 训练模式则是更原生适配了以 Horovod 框架进行训练的任务。本文以上述两种训练模式为例，介绍如何在机器学习平台上发起分布式训练任务。

使用方式

1. 在“任务式建模”界面创建训练任务时，选择训练模式为 MPI/Horovod，并配置单节点资源和节点个数。
2. MPI 训练模式和 Horovod 模式均包含 Launcher 和 Worker 两种角色，但两种角色均可执行训练任务，当任务仅配置一个实例时默认创建 Launcher 实例。
3. MPI 模式启动命令会在每个实例上执行。而 Horovod 模式启动命令仅会在 Launcher 实例上执行，Worker 实例的命令被配置为 sleep infinity 等待 Launcher 的命令。以下是任务式建模启动时默认注入的环境变量列表：
   内置环境变量

变量名	变量描述	示例

变量名	变量描述	示例
OMPI_MCA_orte_default_hostfile	MPI/Horovod 训练任务的节点信息文件	OMPI_MCA_orte_default_hostfile=/etc/mpi/hostfile
GPU_NUM	任务包含的GPU卡总数	GPU_NUM=2
GPU_NUM_PER_NODE	单个节点的GPU卡数	GPU_NUM_PER_NODE=1
NODE_IP_SLOT_LIST	任务包含的节点IP和对应卡数信息（仅支持用于配置启动命令）	NODE_IP_SLOT_LIST=9.0.255.56:1,9.0.255.118:1

4. TI 平台会根据任务配置创建对应的实例组，并注入相关环境变量，给出任务中包含的实例组信息，以及当前实例的角色。
5. 训练过程中任意实例退出码非0则训练任务失败。所有实例成功则训练任务成功。

示例启动方式

其中 /etc/mpi/hostfile 的内容示例如下：

```
train-960258573108964736-7an39bddmfpc-launcher slots=1
train-960258573108964736-7an39bddmfpc-worker-0 slots=1
```

内容分为两列，第一列是实例的域名，第二列是实例上的进程个数。

启动 MPI/Horovod 分布式训练的命令示例如下：

```
# MPI方式启动
mpirun --allow-run-as-root -np $GPU_NUM -H $NODE_IP_SLOT_LIST python3 train.py --data-dir /opt/ml/input/data

# horovod方式启动
horovodrun -np $GPU_NUM -H $NODE_IP_SLOT_LIST --network-interface eth0 python3 train.py --data-dir /opt/ml/input/data
```

此外， DeepSpeed 框架支持使用 OpenMPI 格式的 hostfiles 来配置多节点计算资源，使用MPI启动 DeepSpeed 分布式训练操作实践请查看 使用任务式建模运行 DeepSpeed 分布式训练指引。

PS-worker 模式使用说明

PS（ParameterServer）参数服务器训练是一种常见的数据并行方法，用于在多台机器上扩展模型训练。训练集群由

Worker 和 ParameterServer(ps) 组成。参数保存在 ps 上，在每一轮训练中，ps 将参数分发给 worker，worker 完成计算后将梯度回传给 ps 进行更新。

使用方式

1. 在“任务式建模”界面创建训练任务时，选择训练模式为 PS-Worker，并配置单节点资源和节点个数。

2. 平台提供的 PS-Worker 训练模式包含两种角色：ps 和 worker。ps 保存和更新参数，实例数量应>=1，worker 负责执行训练，实例数量应>=1。

3. TI 平台会根据任务配置创建对应的实例，并注入对应的环境变量 TF_CONFIG，给出了任务中包含的实例组信息，以及当前实例的角色。实例通过读取 TF_CONFIG 得到任务中 ps/worker 的数量和地址，并通过 task 中的 type 得知当前实例所属的角色和编号。
   环境变量
   TF_CONFIG

```
{
   "cluster": {
      "ps": [
         "train-960252492096760832-7an13ppfli80-ps-0.train-100031385875.svc:2222",
         "train-960252492096760832-7an13ppfli80-ps-1.train-100031385875.svc:2222"
      ],
      "worker": [
         "train-960252492096760832-7an13ppfli80-worker-0.train-100031385875.svc:2222",
         "train-960252492096760832-7an13ppfli80-worker-1.train-100031385875.svc:2222"
      ]
   },
   "task": {
      "type": "ps",
      "index": 0
   },
   "environment": "cloud"
}
```

4. 训练过程中任意实例退出码非0则训练任务失败。所有实例成功则训练任务成功。

Ray 模式使用说明

Ray 是一个开源的分布式计算框架，能够简化分布式机器学习的开发和部署。Ray 提供了一套 API 和基础设施，使得开发人员可以轻松地将单机训练代码扩展到分布式环境，支持数据并行和模型并行等多种并行策略。

使用方式

1. 在"任务式建模"界面创建训练任务时，选择训练模式为 Ray，并配置 Head 节点资源和各组 Worker 节点资源

和个数，Worker 最多可配置5组。

2. Ray 训练模式包含两种角色：Head 和 Worker。其中编号为0的是 Head 节点（对应环境变量中 RANK=0），负责协调整个集群的计算资源和任务调度，Worker 节点则执行具体的训练任务。
3. TI 平台会根据任务配置创建对应的实例，并注入相关的环境变量，以下是任务式建模启动时默认注入的环境变量列表：

变量名	变量描述	示例
HEAD_ADDR	Ray 集群的 Head 节点地址	HEAD_ADDR=train-1282671078021627392-9qu5j2n90b9c-head-0
HEAD_PORT	Ray 集群的 Head 节点端口	HEAD_PORT=6379
RANK	当前节点在集群中的序号，RANK0为HEAD节点	RANK=0

示例启动方式

您只需要在您的代码中，使用 ray.init() 默认初始化即可，并把您的脚本执行命令配置到启动命令，我们会默认在 Head 节点上提交您的任务到集群中，您无需指定 Head 节点的地址。
以一个简单的计数任务为例，将如下代码保存为 job.py。

```
import ray

ray.init()

# 定义 Actor 类
@ray.remote
class Counter:
    def __init__(self):
        self.value = 0

    def increment(self):
        self.value += 1
        return self.value

# 创建 Actor 实例
counter = Counter.remote()

# 并发调用 Actor 方法
futures = [counter.increment.remote() for _ in range(10)]
results = ray.get(futures)  # [1, 2, 3, ..., 10]
```

```
print("计数器结果:", results)
```

job.py 作为训练任务，放到您的 CFS 指定目录，并在任务式建模中选择挂载到 /opt/ml/code 目录，然后指定启动命令为:

```
cd /opt/ml/code; python job.py
```

在训练任务中，可以通过 INDEX 指定节点，以指定在 INDEX = 1 的节点执行如下函数为例：

```
@ray.remote(resources={"Rank:1": 0.001})
def f(a, b, c):
    return a + b + c
```

> 注意：
>
> 1. 暂时不支持 Ray tensorboard 的查看。
> 2. Ray 的设计是基于整数资源调度的。例如，一个任务声明需要 1 个 CPU，Ray 会确保它独占一个完整的物理核心，避免资源竞争。 因此建议在配置 Ray 资源组的 CPU 资源时避免使用小数核数，例如设置0.7核，Ray 会将您的设置向下取整，例如 0.7 会被截断为 0，这可能导致程序行为不符合预期（例如任务无法运行）。
> 3. 由于 TIONE 平台支持 GPU 碎片调度，例如0.2卡资源。在 Ray 集群内部，Worker 会把0.2卡当成一块完整的GPU来使用，可以通过 nums_gpu=1 来设置。

二、发起 RDMA 网络加速训练

RDMA 是 kernel by pass 的一种通信技术，在多机通信场景可显著提升通信带宽。本文将介绍如何在 TIONE 平台任务式建模使用 RDMA 网络。

使用前提

1. 资源组中至少包含 2 台支持 RDMA 的高性能 GPU 节点。
2. 提交的分布式任务配置为大于等于2个节点，且每个节点配置为8卡整机 GPU 资源，平台会默认为该资源配置 RDMA 资源。
3. 平台 llm 内置镜像默认支持常见的 HCC 高性能 GPU 机型，自定义镜像需要安装用户态 RDMA 驱动，安装文档参考 容器安装用户态 RDMA 驱动。

如何确认 RDMA 是否生效

在平台上运行的多机任务，如果开启了 RDMA，则会有以下日志：

```
[0] NCCL INFO Channel 00/0 : 8[0] -> 0[0] [receive] via NET/IBext/0/GDRDMA
```

平台内置的环境变量

针对 HCC 机型多机训练场景会开启 RDMA，并且会内置以下 NCCL 环境变量，用户使用 TI 平台的时候，无需显式设置。

```
NCCL_IB_GID_INDEX=3
NCCL_IB_SL=3
NCCL_CHECK_DISABLE=1
NCCL_P2P_DISABLE=0
NCCL_IB_DISABLE=0
NCCL_LL_THRESHOLD=16384
NCCL_IB_CUDA_SUPPORT=1
NCCL_IB_HCA=mlx5_bond
NCCL_NET_GDR_LEVEL=2
NCCL_IB_QPS_PER_CONNECTION=4
NCCL_IB_TC=160
NCCL_PXN_DISABLE=1
NCCL_IB_TIMEOUT=24
NCCL_DEBUG=INFO
NCCL_SOCKET_IFNAME=eth0
GLOO_SOCKET_IFNAME=eth0
TCCL_TOPO_AFFINITY=4
```

NCCL 支持的环境变量参考文档 Environment Variables。

开发机
开发机简介

概述

开发机是 TI-ONE 为开发者量身打造的灵活的在线开发工具，支持通过 Jupyter Notebook 或者 VSCode 多种 IDE 进行在线编码，也支持通过 SSH 连接开发机远程开发。您可以在开发机中完成数据准备、数据预处理、算法调试与模型训练，同时我们也提供了多种内置主流开发环境，您可以一键创建，开箱即用。

核心特性

- 提供多种资源规格供用户自由选择，支持各类资源灵活切换，降低使用成本。
- 内置多种 WebIDE，且支持 SSH 连接远程开发。
- 内置多种镜像，支持自定义安装第三方库；支持导出镜像，一键保存自定义环境。
- 支持生命周期脚本，用户可以在创建/启动开发机实例时运行预设的 shell 脚本。
- 支持与 Git 存储库对接，方便协同开发与版本控制。

创建开发机

操作场景

本文档将向您演示如何在 TI-ONE 中创建一个开发机实例。

操作步骤

1. 登录 TI-ONE 控制台，单击菜单栏的开发机，页面将跳转至开发机的实例列表页面，此页面将罗列用户创建的所有开发机实例。
2. 在开发机实例列表页，单击左上角新建，跳转至创建开发机实例的设置页面。填写说明如下：
    - 名称：开发机名称，不超过60个字符，仅支持中英文、数字、下划线 "_"、短横 "-" ，只能以中英文、数字开头。
    - 地域：此字段不可修改，将自动显示平台选择的地区。
    - 镜像：您可以选择启动开发机的镜像，支持的镜像列表请查看内置训练镜像列表。其中tilearn-llm相关训练镜像内置了最新版本Angel加速组件 tilearn-llm，可直接用于部分开源大模型训练的加速，详细使用指引请 Angel 训练加速功能介绍。
    - 资源组：请选择已创建的资源组
    - 存储路径设置：可选择云硬盘和CFS（包含 CFS Turbo ）文件系统。存储路径可配置多个，为了确保开发机实例可以正常使用，选择第一个存储路径会挂载到开发机的默认工作目录 /home/tione/notebook 下。
        - 选择云硬盘，平台会将申请的100G硬盘挂载到用户开发机容器目录中.
        - 选择 CFS 文件系统，需要选择 CFS 文件系统，填写 CFS 文件系统的源目录和容器挂载路径，平台会将该 CFS 文件系统的源目录挂载到用户指定的容器挂载路径中。
    - 标签：可为开发机添加标签，支持添加多个。
    - 高级设置（默认收起）：
        - 环境变量：可以添加多个环境变量。
        - CLS 日志服务：用户可以自行选择是否开通 CLS 日志服务。
        - 生命周期配置：选择是否使用生命周期脚本。
        - Git 存储：此为可选项，用户可以前往 Git 存储库-新增存储库进行配置。
        - 自动停止：开启该选项后，该实例将在运行时长超过您选择的时长后自动停止，停止状态计算资源不再收费，存储资源仍会收费，请注意费用产生。自动停止时间以小时为单位，最小为1小时，最大为24小时。
        - SSH 连接：您可以选择是否启用 SSH 连接，启用后您可以在其他机器上访问本实例。您需要填写发起 SSH 登录机器上的 ~/.ssh/id_rsa.pub 文件内容。若该文件不存在，可用 ssh-keygen 命令生成。发起 SSH 登录时，请注意检查私钥是否配对。如需从多台机器发起 SSH 登录需要填写多

个公钥，您可以添加多个（按回车键可输入多个）。

- VPC 和子网：平台支持为开发机容器配置 VPC 和子网，当选择了 VPC 和子网后，开发机容器将可访问同一网络配置下的其他云产品。
- 直接访问 internet：开发机默认为不使用 VPC，即不为开发机容器配置 VPC，该情况下默认可访问公网；若选择了 VPC 和子网，则会为用户的 开发机容器配置该私有网络，配置后开发机容器可访问同一 VPC 网络下的其他云服务。选择 VPC 和子网后，可选择关闭直接访问 internet，此时开发机容器将无法访问公网。

3. 单击创建，开发机列表中将新增一条实例记录。当实例“状态”由启动中变为运行中时，单击打开进入开发机实例内部。

管理开发机

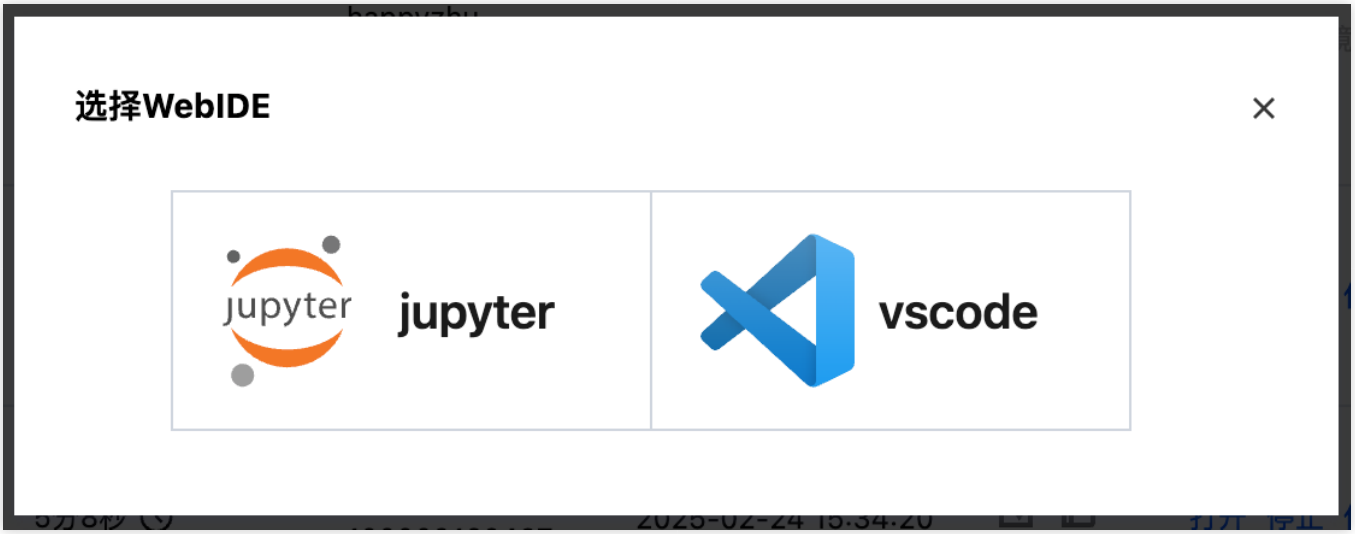
操作场景

本文档将向您演示在 TI-ONE 中，如何对开发机实例进行实例管理、资源管理和数据管理。

实例管理

完成新增实例后，开发机列表中将新增一条实例记录。您可以在此页面查看实例的名称、占用资源、标签、状态、运行时长、最近编辑时间、监控与日志等。您还可以对开发机实例进行打开、停止、编辑、复制、删除等操作。

- 单击监控与日志栏，可查看 开发机 资源监控情况和日志详情。
  - 监控指标支持 CPU 使用率、MEM 使用率、GPU 使用率、显存使用率、系统磁盘使用率（当实例所在节点有本地数据盘时，指的是当前开发机实例所在 Pod 所分配的系统磁盘使用情况；当实例所在节点没有本地数据盘时，指的开发机实例所使用的节点系统盘使用率）
- 单击打开进入开发机实例内部，当前选择通过 Jupyter Notebook 或者 VSCode 两种 IDE 打开进行在线编码。



- 单击停止开发机将会停止运行。实例停止后非挂载的持久化存储路径下的数据将被清空，请注意数据保存，可将数据转存到持久化路径，或者在实例停止前保存镜像。
- 单击编辑可以对部分配置信息进行修改。
- 单击删除此实例记录销毁。
- 单击复制可一键复制实例配置，快速新建开发机。
- 单击保存镜像可以将该开发机实例保存为镜像，您可以选择其中一个 kernel 环境进行保存，支持保存到容器镜像服务个人版和企业版（请注意保存的镜像中不会包含挂载的外部存储设备，如云硬盘、CFS、CFS Turbo路径下的文件和数据下的文件和数据）。镜像构建记录可单击开发机 实例名称，在镜像构建记录页面可查看。
- 单击远程连接可以查看该开发机实例的远程访问地址。
- 实例支持自动停止设置，开启该选项后，该实例将在运行时长超过您选择的时长后自动停止，停止状态计算资源不再

收费，存储资源仍会收费，请注意费用产生。自动停止时间范围为1 - 24小时，数值为整数。新增实例时，您可以设置自动停止时间，在实例列表中，您可以单击运行时长旁的小时钟，开启、禁止或重设自动停止时间。

SSH 网络配置指引

背景介绍

开发机实例支持通过 SSH 远程连接，远程连接可提供公网访问地址和 VPC 内访问地址，其中公网访问地址无需配置，平台会默认生成，如下图所示：

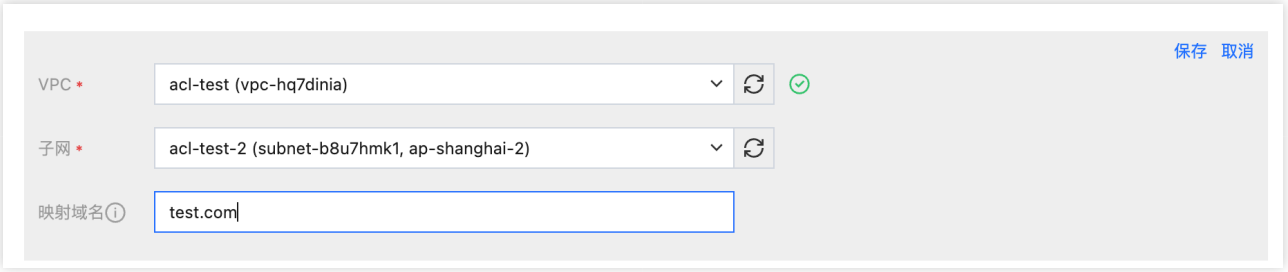
VPC 内访问地址一般提供给具有内网访问要求的客户使用，平台默认展示实例 Pod 内 IP 地址，该 IP 会随着开发机实例重启而改变，长期稳定的连接建议用户前往 SSH 网络配置进行私有连接打通，以下是详细的配置指引。

私有连接配置

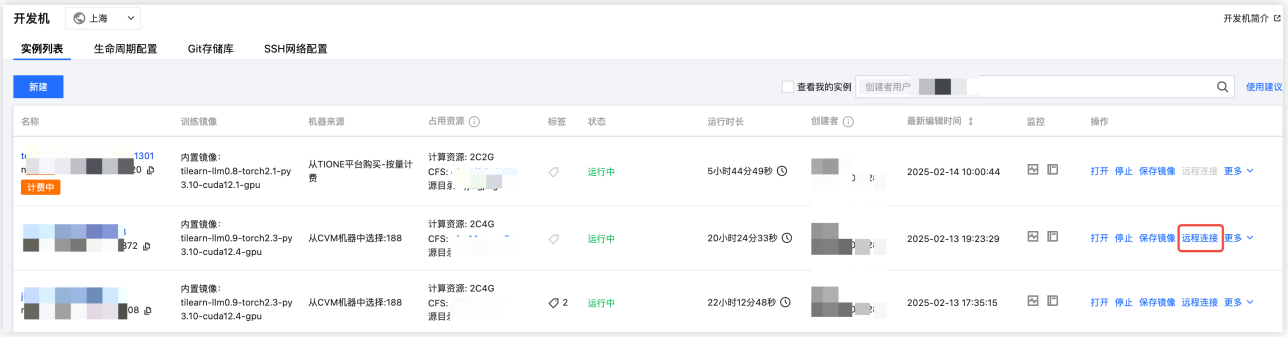
1. 进入 SSH 网络配置 Tab页，私有连接配置要求该账号拥有 CreateVpcPrivateLink、DeleteVpcPrivateLink、DescribeVpcPrivateLinks、DescribeVpcPrivateLink、ModifyVpcPrivateLink 接口权限，可联系主账号或者 CAM 管理员进行配置。进入后单击添加私有连接：
    i. 选择需要打通的 VPC 和子网。
    ii. 您也可以为当前私有连接地址配置自定义映射域名，注意，需要用户访问终端自行进行域名解析。

1.3. 配置完成后单击保存，则会在用户账号下自动创建一个终端节点，请注意：

- 该终端节点需要独立计费，详细计费方式请查看终端节点计费。
- 私有连接配置一旦完成，则对当前地域下的所有开发机实例均生效。



2. 配置完成后，在开发机实例列表页，单击远程连接，则可查看当前实例的 VPC 内访问地址和所生效的 VPC 和子网信息，如下图所示。您可以在本地客户端直接复制该地址连接该实例。
   - 一个地域下可创建多个私有连接，一个私有连接对应一个访问地址。
   - 一个实例可以关联多个访问地。
   - 一个 VPC 和子网仅能创建一个私有连接。
   - 一个实例可以生成多个访问地址。




远程连接地址
公网访问地址
VPC内网访问地址
访问命令：ssh -p 10329 root@1 .1.2
仅限以下VPC调用：VPC vpc冲 (vpc-c ic)、子网 测试
3(sub aet, 上海五区)
访问命令：ssh -p 10329 root@h .com
仅限以下VPC调用：VPC ac dinia)、子网 acl 8gbbr, 上海五
区)
确定
取消

闲置回收策略配置指引

背景介绍

为了提升资源利用率，平台支持为开发机实例配置闲置自动回收策略，即管理员可以设置当前地域的开发机闲置策略，当资源利用率未达到要求时，会强制回收以释放计算资源。

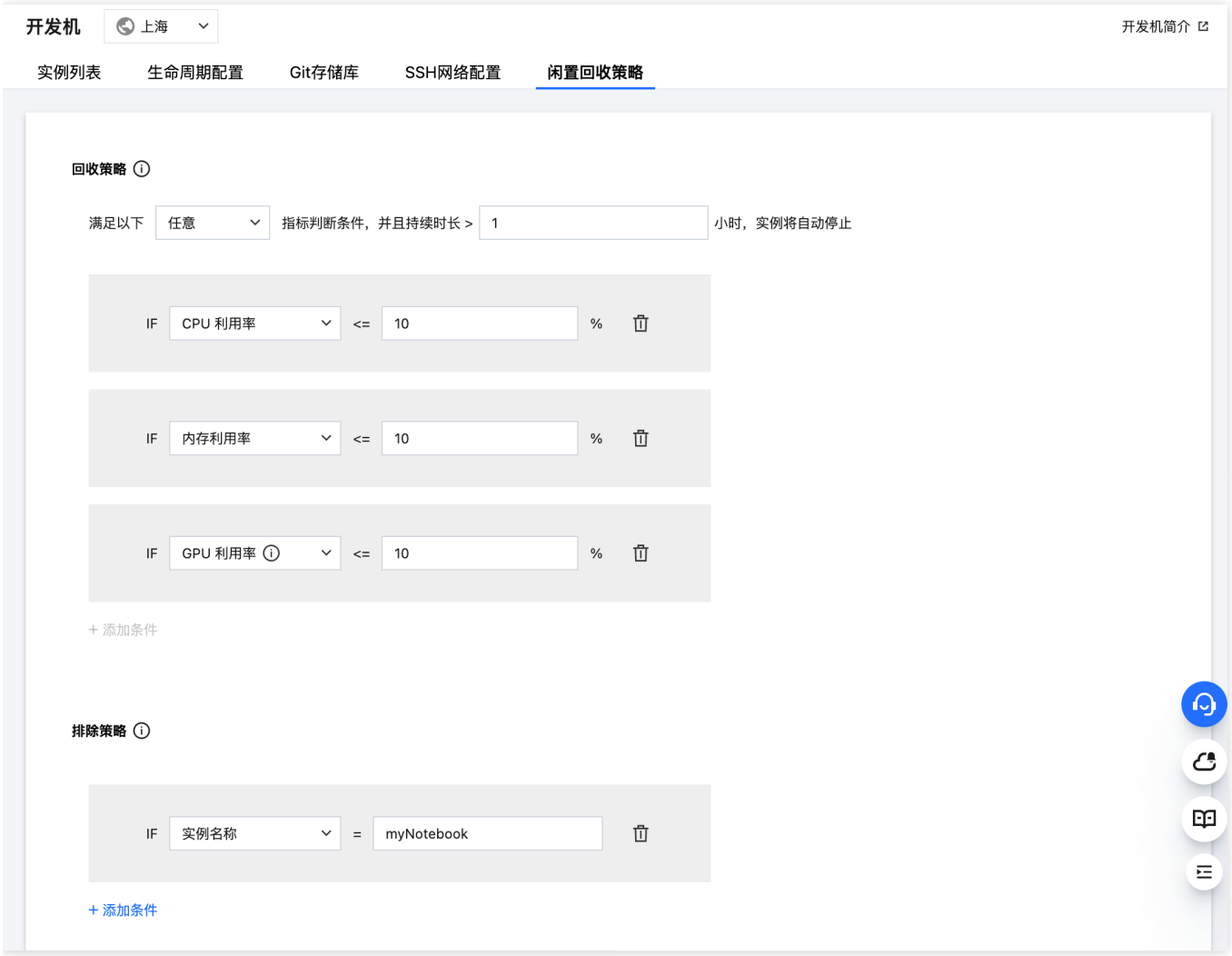
配置指引

进入 闲置回收策略 Tab 页，配置和查看闲置回收策略要求该账号拥有 CreateRecyclePolicy、DescribeRecyclePolicies、DeleteRecyclePolicy、ModifyRecyclePolicy 接口权限，可联系主账号或者 CAM 管理员进行配置。单击新建策略，可支持配置回收策略和排除策略。其中回收策略表示当实例满足回收条件时，实例将会自动停止，释放计算资源；排除策略表示满足排除条件的实例，不受回收策略控制，也就是即使符合闲置回收条件，也不会被强制停止。

配置回收策略

支持按照 CPU 利用率、内存利用率和 GPU 利用率配置实例的资源指标条件，其中这些指标条件可按照任意或者所有进行逻辑组合。当选择任意时，代表当满足配置的任意指标条件时，即可触发自动停止；当选择所有时，代表当满意配置的所有指标条件时，才可触发自动停止。

> 如下图策略所示，当配置了如下策略时：
>
> 满足以下任意指标判断条件，并且持续时长>1小时，实例将自动停止。
>
> IF CPU 利用率<= 10%
>
> IF 内存利用率<= 10%
>
> IF GPU 利用率<= 10%
>
> 表示当某一开发机实例的 CPU 利用率或者内存利用率或者 GPU 利用率持续1小时以上小于等于10%，该实例即会触发自动停止。



配置排除策略

支持按照 实例名称 过滤选择被排除的实例，也就是在该排除策略中配置的开发机实例，将不受上述回收策略管控，如上图所示，实例名称为 myNotebook 的开发机将不受已配置的回收策略管控，也就是即使该实例达到了回收的条件，也不会被强制停止。

实例列表

1. 当该地域配置了闲置回收策略以后，则在实例列表上方展示如下提示文案，单击自动停止策略，用户可查看配置的指标条件和持续时长（用户需要有DescribeRecyclePolicies 权限）
2. 当开发机实例由于闲置自动停止后，状态为转为已停止 ，同时 hover 可展示“已触发闲置实例自动停止”文案提示，如下图所示。


指标条件
持续时长
CPU 利用率<=10%||
>1小时
内存利用率<=10%||
GPU 利用率<=10%
开发机
开发机简介
实例列表
网络配置
闲置回收策略
当前地域已配置了闲置实例自动停止策略，触发规则的实例会自动停止，请合理管理实例运行时间并保存关键数据，若您需要退出闲置自动停止队列，请联系管理员。
新建
查看我的实例
多个关键字用竖线 "|" 分隔，多个过滤标签用回车键分隔
使用建议
名称
训练镜像
机器来源
占用资源
标签
状态
运行时长
创建者
最新编辑时间
监控
操作
已触发闲置实例自动停止
内置镜像：
tilearn-llm0.9-torch2.3-py3.10-cuda12.4-gpu
从TIONE平台购买-按量计费
计算资源: 2C2G
已停止
10分11秒
2025-02-24 14:23:11
打开 启动 保存镜像 远程连接 更多

使用生命周期脚本

生命周期脚本配置规则

生命周期配置提供 SHELL 脚本，在用户创建 发机实例或每次启动开发机实例时运行，可以帮助用户安装自定义依赖，个性化配置开发机环境。
生命周期配置遵循以下规定：

- 创建脚本：第一次新建后启动发机实例会运行的脚本，只会运行一次。
- 启动脚本：每次启动开发机实例时都会运行的脚本，包括第一次创建时。
- 每个脚本 BASE64 编码后不能超过16384个字符。
- 每个脚本将以 root 用户的角色运行。
- 每个脚本的 $PATH 环境变量为 /usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/snap/bin
- 每个脚本最长运行时间为5分钟，超过5分钟开发机将启动失败，请避免在脚本中安装大型依赖包。可在详情页中查看失败原因如“启动脚本超时”。
- 如果脚本出错，开发机也将启动失败，可在详情页中查看具体失败原因。
- 如果脚本是从自己的编辑器复制到 TI-ONE 网页上的，请确保编辑脚本的编辑器使用 Unix 风格的编排。

生命周期脚本最佳实践

以下是使用生命周期配置的一个实践案例：

- 生命周期脚本以 root 用户权限运行，开发机进程以 tione 用户运行。如果需要切换用户，可以在脚本中运行 sudo -u tione 切换到 tione 用户。
- 开发机使用 conda 管理多内核，可以激活 conda env 来为不同的内核安装依赖包。
  例如：在 conda_python3 的内核中安装 Python 依赖包 fire，可以编写如下启动脚本：

```
#!/bin/bash
sudo -u tione -i <<'EOF'

# This will affect only the Jupyter kernel called "conda_python3".
source /opt/conda/bin/activate python3

# Replace fire with the name of the package you want to install.
pip install fire
# You can also perform "conda install" here as well.

source /opt/conda/bin/deactivate
```

```
EOF
```

例如：在所有内核中都安装 fire 依赖包，可以这样编写脚本：

```
#!/bin/bash
sudo -u tione -i <<'EOF'

# Note that "base" is special environment name, include it there as well.

for env in base /opt/conda/envs/*; do
  source /opt/conda/bin/activate $(basename "$env")

  # Installing packages in the Jupyter system environment can affect stability of your tione
  # Notebook Instance.  You can remove this check if you'd like to install Jupyter extensions, etc.
  if [ $env = 'JupyterSystemEnv' ]; then
    continue
  fi

  # Replace myPackage with the name of the package you want to install.
  pip install fire
  # You can also perform "conda install" here as well.

  source /opt/conda/bin/deactivate
done

EOF
```

Git 存储库

简介

进行模型训练和代码调试时，往往需要直接对接用户的远端代码仓库，TI-ONE 为用户提供统一的 Git 存储库管理能力，用户可以在实际使用任务式建模或者开发机时关联配置好的 Git 存储库，从而能够快速拉取代码或者连接代码仓库。

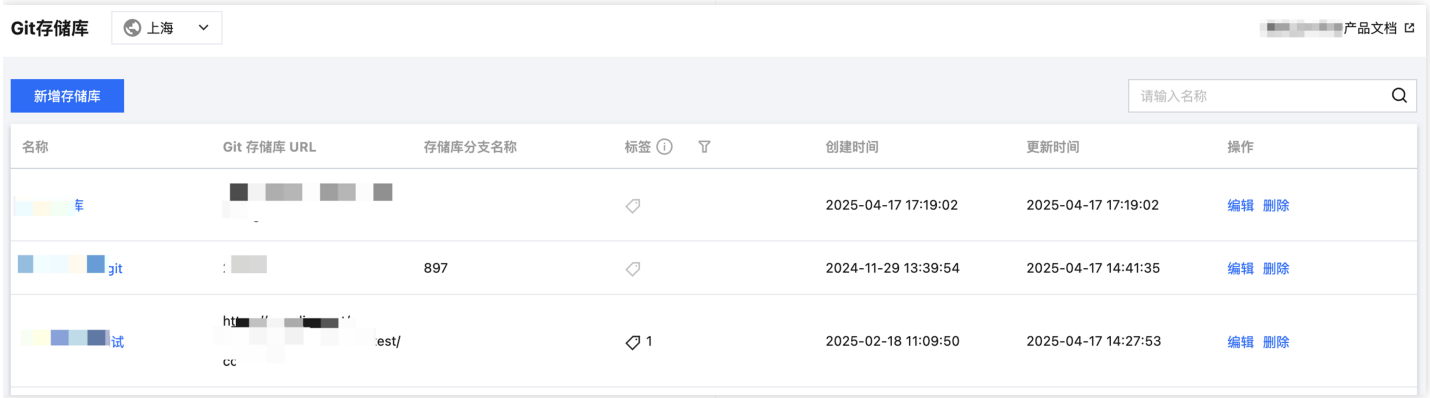
功能描述

存储库管理

新增存储库

前往训练工坊 > Git 存储库页面，单击新增存储库 ，在弹窗中输入存储库的名称，目标存储库的 URL 以及存储库分支的名称（可选）。如果目标存储库为需要验证的私有存储库，您必须准确输入所需的用户名（或邮箱）与密码，用于凭证验证。

创建完成后可以在列表页看到配置的存储库列表。



编辑已创建的 Git 存储库

前往训练工坊 > Git 存储库页面，单击编辑 ，可以对已创建的 Git 存储库的密钥进行编辑。

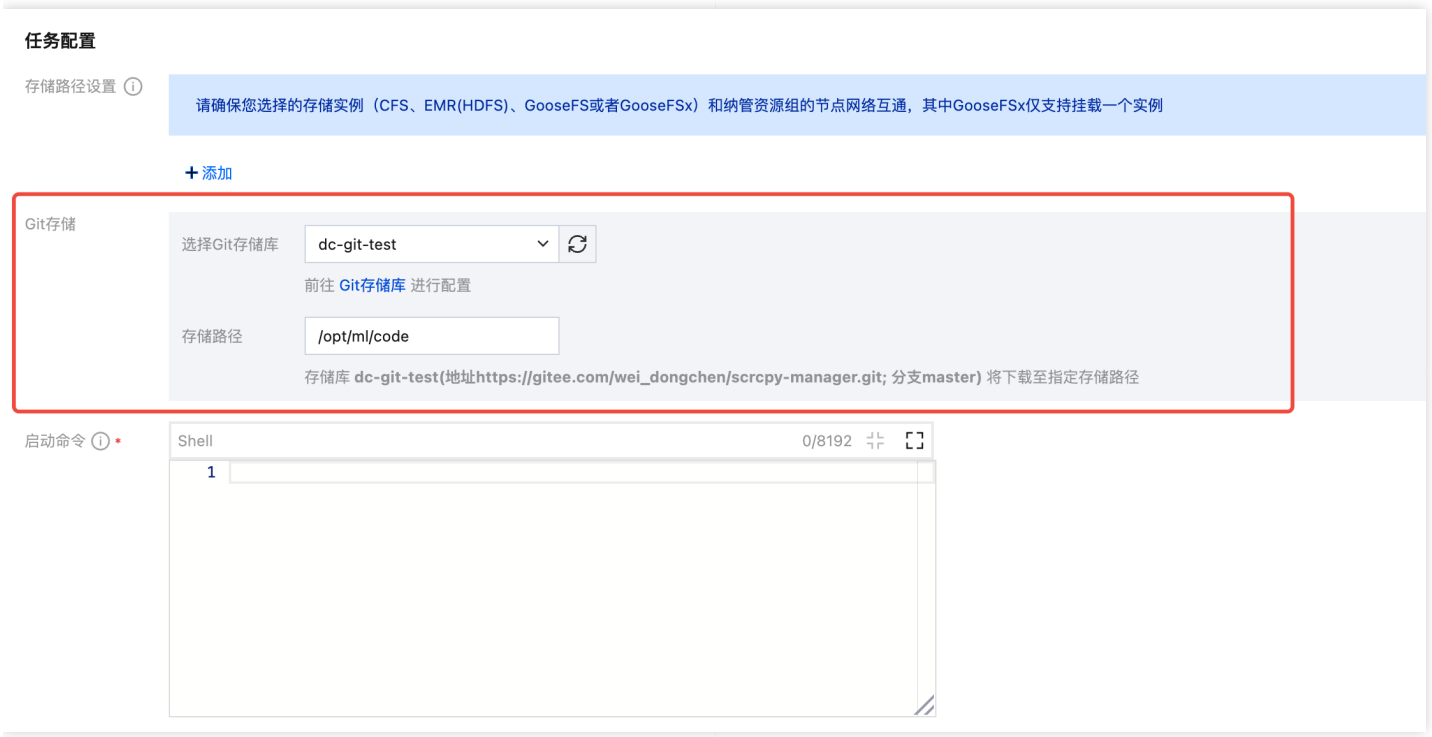
> 注意
>
> 一旦创建，存储库的名称和 URL 将不能修改，如果输入错误，建议删除重新新建。

在开发机中关联 Git 存储库

前往开发机页面，单击新建实例 ，您可以下拉选择已创建的 Git 存储库，可以添加除默认存储库以外的其他存储库，最多添加3个其他存储库。
配置完成并且启动开发机后，首先会将关联的 Git 存储库中的文件下载到默认工作目录中，此外，用户也可以在开发机内免密连接该存储库，方便提交和同步代码。

在任务式建模中关联 Git 存储库

前往任务式建模页面，单击新建任务 ，您可以下拉选择已创建的 Git 存储库，同时需要指定存储路径，配置完成并且启动任务以后，会将关联的 Git 存储库下载到指定存储路径中。


任务配置
存储路径设置
请确保您选择的存储实例（CFS、EMR(HDFS)、GooseFS或者GooseFSx）和纳管资源组的节点网络互通，其中GooseFSx仅支持挂载一个实例
+ 添加
Git存储
选择Git存储库
dc-git-test
前往 Git存储库 进行配置
存储路径
/opt/ml/code
存储库 dc-git-test(地址https://gitee.com/wei_dongchen/scrcpy-manager.git; 分支master) 将下载至指定存储路径
启动命令
Shell
0/8192
1

在线服务
在线服务简介

模块概述

TI-ONE 的在线服务模块，提供将模型部署为在线推理服务的能力，供用户通过 API 接口调用的方式对接自身业务应用。在线服务支持虚拟化异构算力和弹性扩缩容能力，帮助用户解决模型部署复杂、资源浪费、手工扩展资源效率低下等问题。同时，在线服务还支持部署多种模型格式、支持服务流量分配与滚动更新，以支撑在线推理场景中的多元应用诉求。

模块特点

- 算力虚拟化：支持为服务分配小至0.1卡 GPU 算力，通过细粒度算力分配，让您随时随地享受高性价比服务体验。
- 自动弹性扩缩容：您可以选择手动或自动调整弹性实例扩展策略，模型部署会根据业务负载情况，动态实时自动管理实例数量，帮助您以最合适的实例数量应对业务情况，免去人工部署负担。
- 丰富的管理能力：提供丰富的多模型支持、多版本管理、流量分配、滚动更新等能力，支持服务及调用信息的多维度监控及事件查看，为您的各类业务保驾护航。

应用场景

支持用户将推荐、图像处理、自然语言处理、语音识别等各类机器学习场景的模型部署为在线服务。

在线服务部署

在完成模型的训练或自定义镜像的开发后，可以使用模型服务模块部署为在线服务。

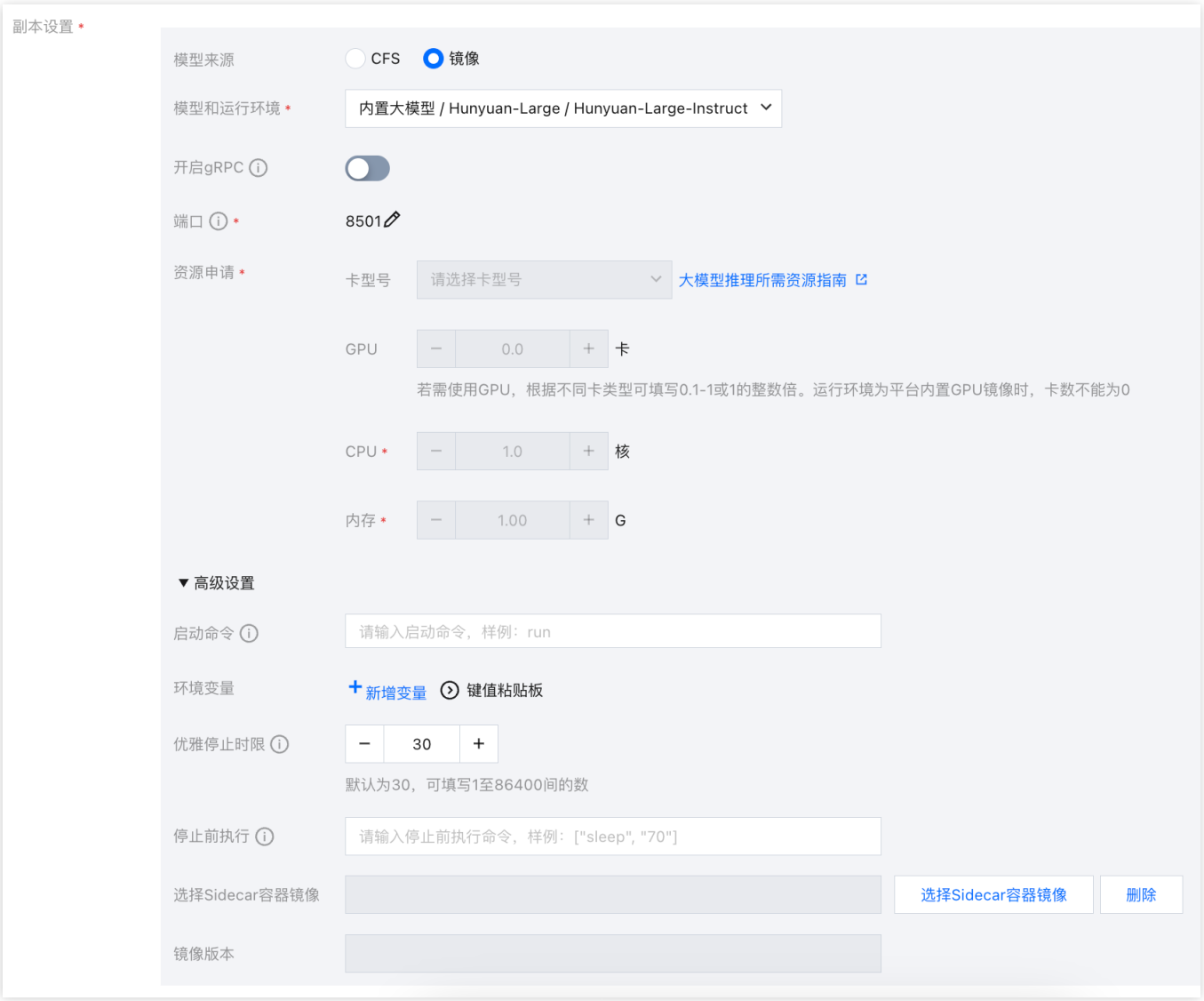
操作步骤

登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面。 在服务列表页面单击新建服务，进入服务启动页面。 在服务启动页面，配置在线服务的相关参数。

1. 服务基础信息
   基础页面填写的参数说明：

参数	说明
服务名称	服务的名称，按照界面提示的规则填写即可。
服务版本	版本号系统自动生成。
服务描述	可按需为服务配置描述信息。
地域	同账号下的服务按地域进行隔离，地域字段取值根据您在服务列表页面所选择的地域自动带入。
部署方式	支持多种部署方式： - 标准部署：单副本下有1个实例运行，适用于大多数标准场景。 - 多机分布式部署：单副本下有多个实例协调运行，适用于模型需要多机并行的场景。 注意： 服务新建后，更新服务、新增版本均无法修改部署方式，请新建的时候谨慎选择。
资源组	可选择资源组管理模块的资源组。

2. 副本设置信息



参数填写说明：

参数	说明
模型来源	支持多种模型来源（不同部署方式下支持的模型来源也不一致）： - 选择 CFS 适用于，部署服务所需的模型文件已放在 CFS 文件系统里的场景，选择模型所在的 CFS 实例，路径输入到模型所在路径的层级（如模型为精调出来的 checkpoint500，则路径输入到/a/b/checkpoint500这一层级）。CFS Turbo 仅支持机器来源为“从 CVM 机器中选择”时使用。 - 选择镜像适用于，部署服务所需的自定义镜像已封装模型文件，不需要再进行模型文件挂载，且自定义镜像已上传至容器镜像服务 TCR 的场景；或者内置大模型的场景。
运行环境	- 若从 CFS 选择模型，则运行环境需要根据模型文件选择对应的内置镜像。 - 若从镜像选择模型，则运行环境可以选择已上传至容器镜像服务 TCR 的自定义镜像、输入镜像地址或者内置大模型镜像。
开启 gRPC	开关默认关闭。关闭时仅支持 HTTP 协议调用。开启后支持 gRPC 协议调用。
端口	支持配置容器对外暴露的端口，可填范围 1024-65535，但不包括 8502-8510,6006,9092。 额外注意两个特殊端口号：8500 是 gRPC 的默认端口，8501 是 REST 的默认端口，不可混淆使用。

参数	说明
资源申请/算力规格	可设置从所选资源组中申请多少资源用于启动当前服务。
启动命令	支持配置容器的启动命令，选填。
环境变量	支持配置容器的环境变量，选填。
优雅停止时限	即 Kubernetes 的 terminationGracePeriodSeconds，停止时间超过该时限的 Pod 将被强制销毁。默认值为30s。
停止前执行	即 Kubernetes 的 PreStop 命令，Pod 在销毁前会先执行该命令以实现优雅停止，选填。命令格式为字符串数组，例如："["sleep", "70"]"。
选择Sidecar容器镜像	支持用户自定义配置 Sidecar 容器镜像，选填。

3. 服务特性配置

参数说明：

参数	说明
请求限流	支持配置服务限流值： - 不限流默认单个服务最大的 QPS 为500，限流值大于500时，按照500进行限流，当服务升级包后，服务的总限流值以升级包的上限为准。

参数	说明
	- 单副本 QPS 的限流值为单个实例的限流。 - 单副本最大并发数为单个实例的限流。 注意： 该限流值为单个副本的限流，当服务进行扩缩容时，服务整体限流值将按照设置的值 * 副本数进行更新；单个服务组最大的 QPS 为500，当服务组下设置的服务总限流值大于500时，按照500进行限流。
副本调节	- 手动调节：可以自定义设置服务的副本数量，副本数量最小为1。 - 自动调节：可以选择基于时间或者基于 HPA 的调节策略，该部分详细说明请查看 在线服务运营。
是否生成鉴权	若开启，则服务调用时会进行签名认证，已启动的服务可在服务调用页面查看签名密钥及签名计算指引。开启鉴权后，将会为您自动生成服务的首个密钥，您可在服务详情页-服务鉴权中查看并创建更多密钥。
CLS 日志投递	平台为用户提供近15日服务日志存储，若需要持久化日志存储以及更灵活的日志检索能力、日志监控告警能力 ，可开启 CLS 日志投递，开启后服务日志会根据日志集与日志主题投递至日志服务 CLS。
重试策略	配置服务部署失败时采用的重试逻辑，支持“有限次重试”或“无限次重试”。默认值为有限重试5次。只有在新部署服务时才会使用该逻辑；更新服务或启动已停止服务时，系统将采用“无限次重试”。
滚动更新策略	支持设置服务滚动更新策略，确保服务平滑升级： - MaxSurge：表示滚动更新期间，允许超出所需规模的最大副本数量。 - MaxUnavailable：表示滚动更新期间，允许不可用副本数量的上限。
健康检测	Kubernetes 的健康检查机制，支持自动检测并恢复失败的容器，确保流量分发到健康的实例上。 - 存活检测：Liveness Probes，验证服务进程是否存活，容器是否正常运行。触发阶段：容器启动后持续运行（整个生命周期）。 - 就绪检测：Readiness Probes，确认服务已具备处理请求的能力，容器是否准备好接受流量。触发阶段：容器启动后持续运行（整个生命周期）。 - 启动检测：Startup Probes，对慢启动容器进行监控检查，检测容器内应用是否完成。触发阶段：仅在容器启动阶段运行（成功后停止）。 检查方法支持三种：HTTPGet、TCPSocket、Exec。
自动停止	平台支持自动停止模型服务，当开启该开关后，在线服务将在指定的停止时间自动停止，同时停止服务算力计费。
标签	支持为服务添加标签，用于按照标签进行授权等。

4. 启动服务

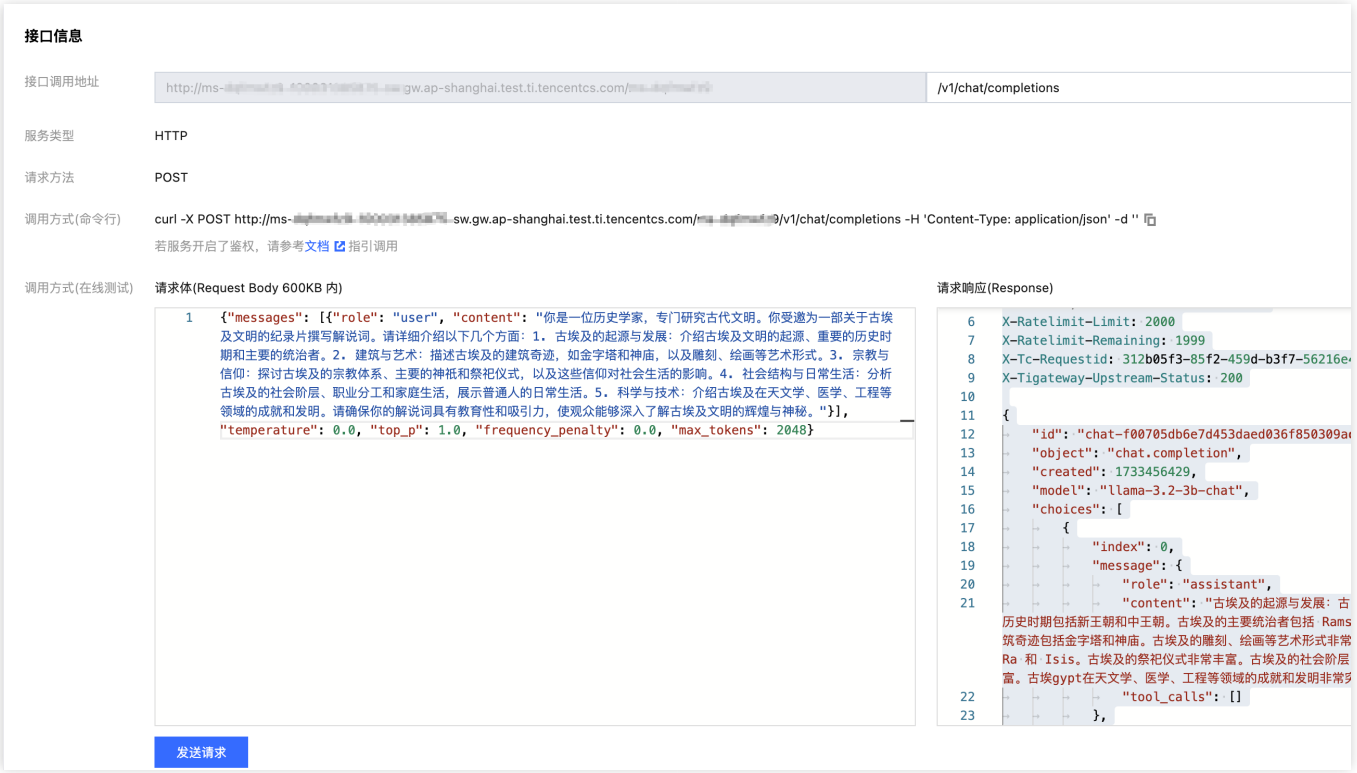
确认服务配置信息无误后，单击启动服务进行服务部署。服务部署过程中将为您创建网关并调度计算资源，需要等待一段时间，待服务成功完成部署时，服务状态将变为运行中。

在线服务调用

接口在线测试

在线服务为您提供接口的在线测试能力，完成部署的在线服务可进行接口的调用测试。

1. 登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面。
2. 在服务列表页单击调用操作按钮，进入服务调用页面，查看服务的接口信息。
3. 单击接口列表中的在线测试操作按钮，打开调用测试页面。
4. 在接口调用地址栏输入您要测试的接口 URL，部分内置镜像会提供默认 URL，如需测试其他接口直接修改即可。
5. 在请求体模块录入 JSON 格式的请求信息并单击发送请求后，可在请求响应模块查看预测结果。



公网调用

在 TI-ONE 完成在线服务部署后，系统会自动注册 API 网关生成公网地址用于服务请求调用。您可以在服务列表页单击调用操作按钮，进入服务调用页面，查看公网访问地址。通过该调用信息可以向在线服务发起预测请求，curl 命令示例如下，其中 Authorization 仅在鉴权开启时需要：

```
curl -X POST https://ms-9******j-100******5.gw.ap-xxx.xx.xxxxx.com/ms-9******j/v1/models/m:predict -H 'Authorization: P*********pg' -H 'Content-Type: application/json' -d ''
```

- 开启服务鉴权，则在调用 API 时，需要使用签名密钥（ApiAppKey 和 ApiAppSecret）对请求内容进行签名计算。

- 如您希望屏蔽在线服务的公网调用方式，可联系您对接的架构师或其他服务团队提供支持。

内网调用

高速服务调用（HTTP）

公网链路收到诸多因素的限制，如果您对请求的性能和稳定性有更高的要求，推荐使用内网高速服务调用的方式来访问您的服务。高速服务调用通过 私有连接建立用户 VPC 与服务的高速通道，具体操作路径如下：

1. 登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面。

2. 在服务列表页单击**服务名称，进入服务版本列表页面，单击服务调用，**查看配置服务调用相关信息。

3. 高速服务调用模块，首次配置，单击开启新增高速服务调用网段。

4. 在配置弹窗里，您选择需要打通内网的 VPC、子网。

5. 配置好 VPC、子网后，高速服务调用模块，展示 VPC 调用地址，以及可调用的 VPC、子网信息，配置好后支持删除后重新新增。如果在线服务有2个版本，则内网地址有2条。

6. 服务调用时，如需要在终端里内网接口调用，可执行以下命令：

```
//内网地址，请您参考服务调用/高速服务调用/VPC调用地址
//URL为服务调用/接口信息/接口调用地址的右边URL内容
curl -X POST -H 'C ontent-Type: application/json' 内网地址/URL -d '{ "prompt": "puppy dog", "steps": 5}'
```

> 说明：
>
> 一次开启配置，本地域下的全部服务均生效。本能力使用私有连接，配置后，将在您账户的该 VPC 和子网下，创建一个终端节点，独立计费，您可参考 终端节点计费文档 。

高速服务调用（GRPC）

高速服务调用除支持 HTTP 协议外，用户可在新建在线服务时选择打开开启 GRPC 的开关按钮支持 GRPC 调用协议。用户开启 GRPC 协议后和 HTTP 协的区别是：

- 只支持 VPC 地址调用，所以公网访问会被屏蔽，前端不再展示常规服务调用的调用地址；
- 由于接口在线测试依赖公网下的 HTTP 协议调用，所以 GRPC 协议下也不支持【接口信息】的模块。
  GRPC 调用说明：发送请求时，请求中需携带 Header x-ti-service = ms-hx7c5srv 和 虚拟 Authority（ms-hx7c5srv-100031385875-sw-grpc.gw.ap-xxxx.com）。

服务访问云上 VPC 资源

如果您的服务需要通过内网访问其他 VPC 资源，可通过服务访问云上 VPC 资源进行配置，具体操作路径为：

1. 登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面。
2. 在服务列表页单击新建服务，在新建弹窗里，开启服务访问云上 VPC 资源。

3. 开启后，选择需要打通内网的 VPC、子网。

4. 配置好 VPC、子网后，服务访问云上 VPC 资源模块，展示配置好的 VPC、子网。

5. 此时 tione 可以内网访问该 VPC 下的资源。

在线服务鉴权和限流

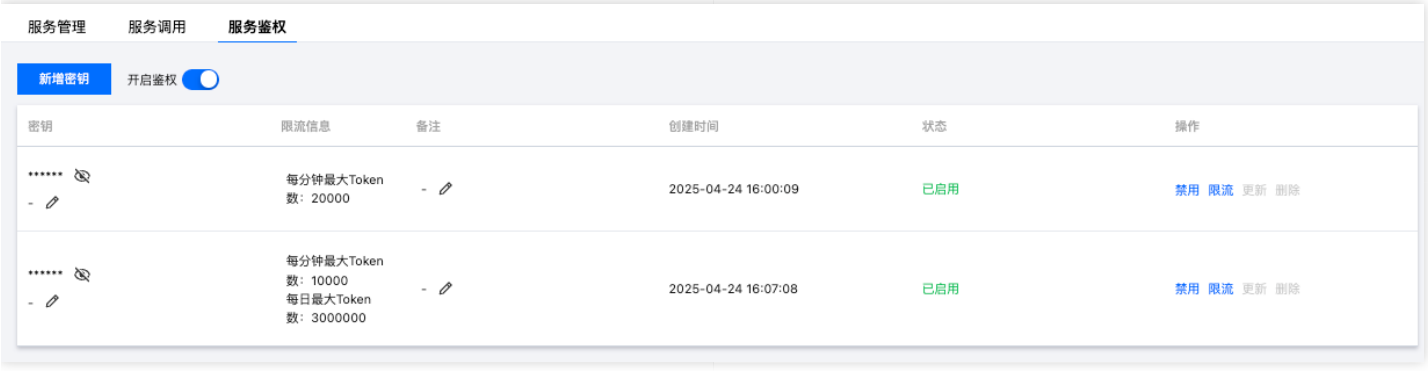
TI-ONE 训练平台在线服务模块为用户提供大模型服务鉴权和限流的产品功能，支持用户针对单个服务配置多个密钥，并支持基于 Tokens 计数进行大模型流量控制，以此实现精细化的调用方管理和流量控制**。**

开启服务鉴权

用户在新建在线服务的参数配置页面，开启**【是否生成鉴权】**开关即可开启服务鉴权功能。因为鉴权开关的状态与部署的服务本身无关，所以服务创建后，用户若需切换鉴权开关的状态，无需停止或更新服务即可在服务详情页的【服务鉴权】Tab 中进行编辑修改。

管理鉴权密钥

开启鉴权密钥后，用户可单击在线服务名称，进入服务详情页面，通过**【服务鉴权】**的 Tab 页对鉴权密钥进行统一管理。



该页面列表字段解释如下：

- 密钥：展示用户访问服务所需的密钥信息，当前仅支持 AuthToken 的密钥类型。开启鉴权后用户需在服务请求中需携带该 AuthToken 进行鉴权。支持用户单击列表页上的新增密钥来添加一个新的自定义密钥。

- 限流信息：展示用户对每个密钥单独设置的限流信息，限流类型支持两种：每分钟最大 Token 数（TPM）和每日最大 Token 数（TPD）。

- 备注：展示用户给密钥添加的备注信息，方便用户详细记录各密钥用途。
- 创建时间：密钥被创建的时间。
- 状态：密钥当前的状态，有“已启用/已禁用”两个枚举值，可通过操作 > 禁用/启用按钮切换密钥状态，禁用密钥后，TI 平台将拒绝此密钥的所有服务请求，请谨慎操作。

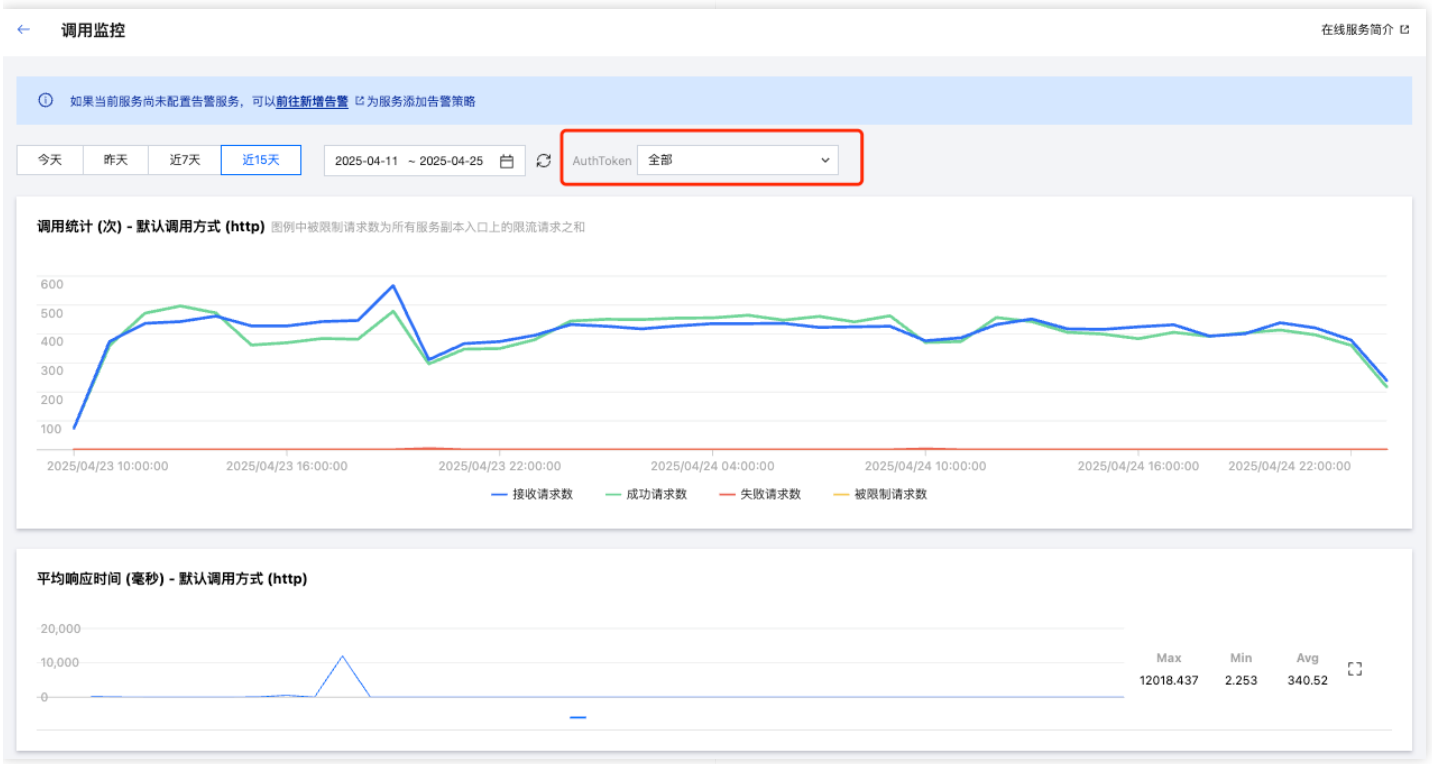
服务限流

用户可通过服务鉴权列表页的【操作-限流】按钮设置对应密钥的流量控制。由于非大模型服务无 token 概念所以无

法进行 token 限流，所以限流功能仅支持“模型来源”为"镜像-内置大模型"，或“运行环境”为“内置-LLM”的大模型服务，且有一个前提条件是：模型的 response 需满足 openai 规范 必须包含标准的 usage 字段信息。
目前支持的限流方式有两种：每分钟最大 Token 数（通过该密钥，每分钟能请求的最大 Token 数）和 每日最大 Token数（通过该密钥，每日能请求的最大 Token 数）。对密钥设置限流后，一旦该密钥的服务请求触发了流量上限则会导致请求报错。

服务鉴权调用监控

由于平台支持用户针对单个大模型服务配置多个密钥鉴权，所以在大模型服务的调用监控页面，平台也支持用户通过切换鉴权 AuthToken 详细的查看每个调用方的调用信息，实现精细化的流量监控。


调用监控
在线服务简介
如果当前服务尚未配置告警服务，可以前往新增告警 为服务添加告警策略
今天
昨天
近7天
近15天
2025-04-11 ~ 2025-04-25
AuthToken
全部
调用统计 (次) - 默认调用方式 (http) 图例中被限制请求数为所有服务副本入口上的限流请求之和
600
500
400
300
200
100
2025/04/23 10:00:00
2025/04/23 16:00:00
2025/04/23 22:00:00
2025/04/24 04:00:00
2025/04/24 10:00:00
2025/04/24 16:00:00
2025/04/24 22:00:00
接收请求数
成功请求数
失败请求数
被限制请求数
平均响应时间 (毫秒) - 默认调用方式 (http)
20,000
10,000
0
Max
12018.437
Min
2.253
Avg
340.52

在线服务运营

自动扩缩容

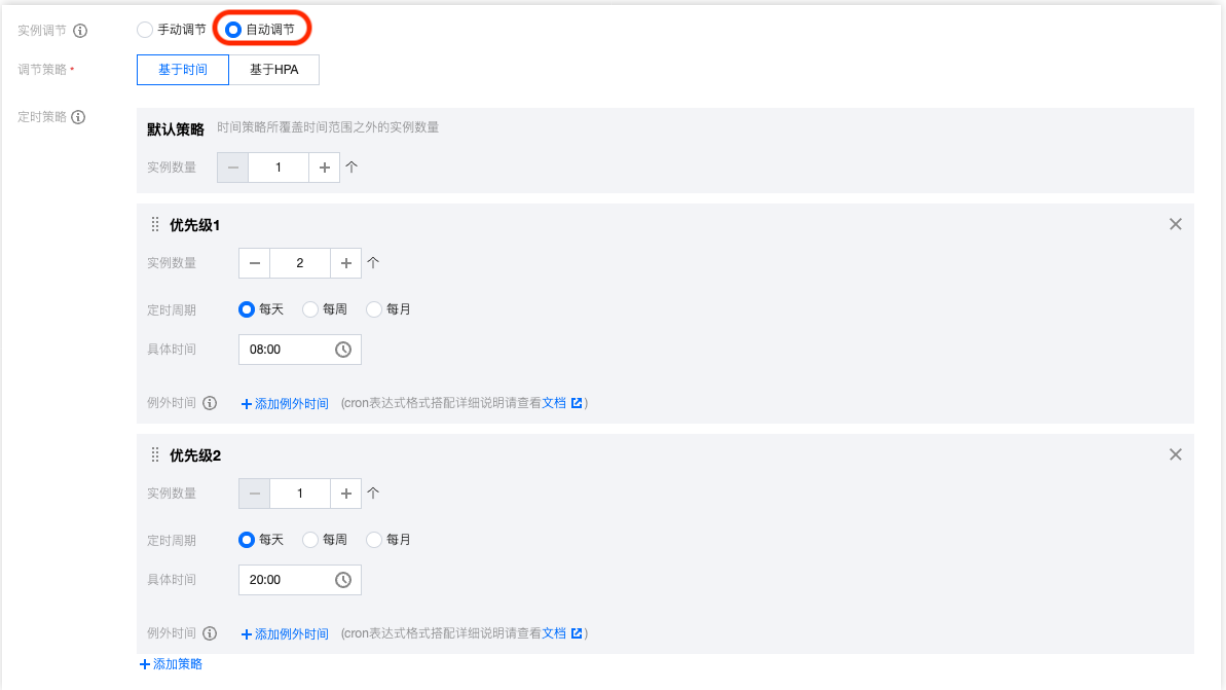
如果您的业务负载有显著的峰谷特征，为了提升推理算力资源的利用效率，您可以使用在线服务模块的自动扩缩容能力。该功能支持在线服务的实例数量根据您配置的扩缩容策略自动调整，从而实现在业务负载高峰时实例数量自动扩容，在业务负载低谷时实例数量自动缩容。
自动扩缩容支持两种类型的调节策略：基于时间调节、基于 HPA 调节，下文将详细介绍两种调节策略的使用方法。

基于时间调节

如果您的业务负载有显著的时间特征，则可以根据时间进行自动扩缩容策略的配置。

1. 如何开启定时扩缩容
   i. 登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面；
   ii. 在服务列表中找到需要开启定时扩缩容策略的服务，单击服务名称，进入版本列表页面，单击更新进入服务详细配置更新页面，或者单击扩缩容进入实例调节弹窗；

   iii. 服务详细配置中，将“实例调节”字段的选项设置为“自动调节”，调节策略类型选择“基于时间”，即可进行时间调节策略的规则配置；
   iv. 您可以根据实际业务负载的时间特征自行配置多条定时策略规则，例如若8:00至20:00为业务高峰时段，20:00至8:00为业务低谷时段，则可以配置如下图的定时策略，每日8:00将实例数扩容至2，每日20:00将实例数缩容为1（默认策略为服务启动后的初始实例数量）；



v. 若您配置了多条定时策略规则，且多条规则之间存在时间冲突，则会以优先级级别较高（即优先级排序靠前）的策略为准；
vi. 完成扩缩容策略的内容配置后，单击更新服务即可进行配置信息保存，待服务完成更新后，您所配置的自动扩缩容策略即会生效。

2. 例外时间配置规则
i. 若某个定时策略希望在特定的时间不执行，则可以为该定时策略规则配置例外时间，支持添加多个；
ii. 例外时间需通过 Cron 表达式进行配置，Cron 表达式共包含6位，分别代表“秒”“分”“时”“日”“月”“星期”，若特定位的取值为任意值则使用星号(*)即可，若特定位取值需包含连续多个数值则可以使用连字符(-)，若特定位取值需包含多个离散数值则可以使用逗号(,)；
iii. 例外时间的最小配置粒度是日，因此 Cron 表达式的前三位取值需要使用“*”（前三位配置其他值不会生效），后三位取值可按需配置，第4位“日”的可用值范围为1-31，第5位“月”的可用值范围为1-12或 JAN-DEC，第6位“星期”的可用值范围为0-6或 SUN-SAT；
iv. 例如：每年10月1日至10月7日的 Cron 表达式为“*** 1-7 10 *”。

基于 HPA 调节

如果定时调节不适合于您的业务模式，您也可以选择“基于 HPA” 的自动扩缩容调节策略，在该策略下，服务实例数量可根据您所配置的策略指标与指标阈值，在实例数的最小值与最大值之间自动进行调节。策略指标支持 CPU 使用率、内存使用率、GPU 使用率、单实例 QPS、最大并发数使用率 等。
其中策略指标需要配置“最大并发数使用率”时，请先在请求限流处配置“单实例最大并发数”。

流量分配

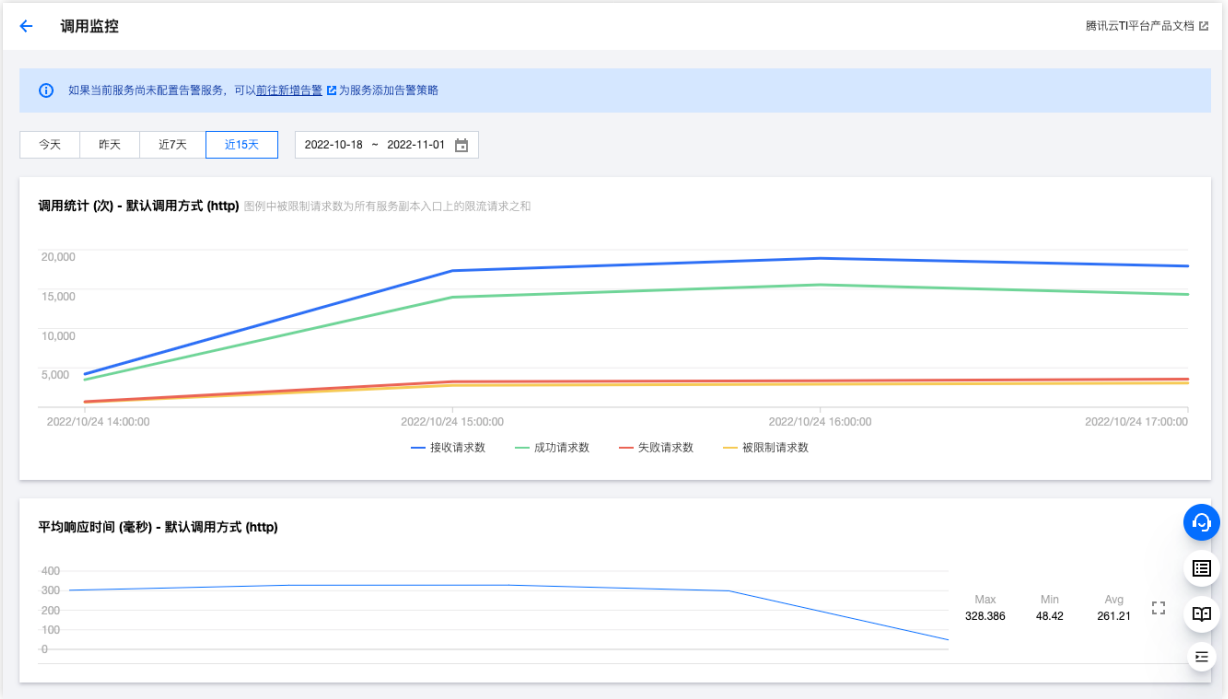
为了满足灰度验证或者A/B测试类的服务使用诉求，平台支持用户为单个服务添加多个版本，并进行流量分配。

1. 登录 TI-ONE 控制台，在左侧导航栏中选择模型服务 > 在线服务，进入在线服务列表页面。
2. 找到需要测试的服务，单击服务的新增版本操作，打开服务版本创建页，按需配置当前服务版本的容器信息及实例调节信息。
3. 单击启动服务。
4. 创建新的服务版本后，系统将为您创建网关后端并调度计算资源，需要等待一段时间，待服务版本成功完成部署时，状态将变为运行中。
5. 此时可单击服务版本列表上方的流量分配操作，进行多版本流量比例的设置。

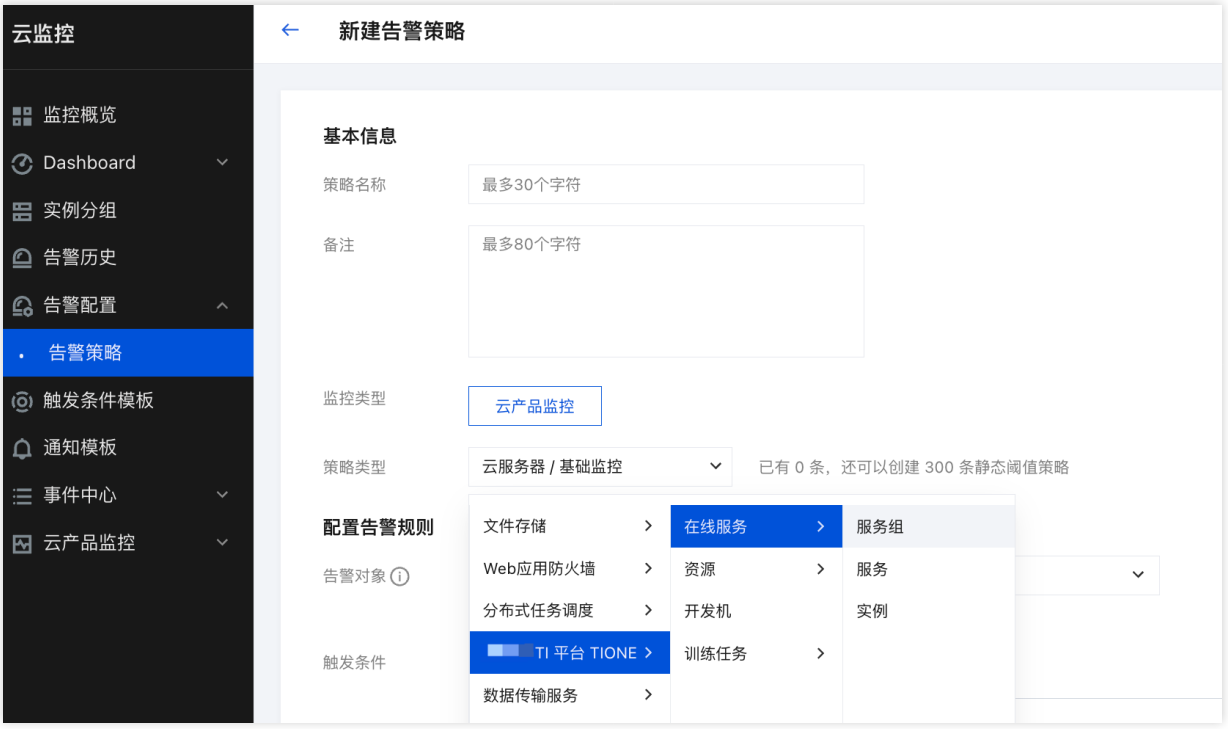
服务监控

为了满足服务运行情况追踪的诉求，平台提供服务数据监控、调用数据监控、事件与日志查看能力。

1. 在线服务列表页面，单击服务名称进入版本列表页后，单击服务调用>调用监控，可查看服务调用情况的统计信息，包括接收请求数、成功请求数、失败请求数、被限制请求数、平均响应时间。

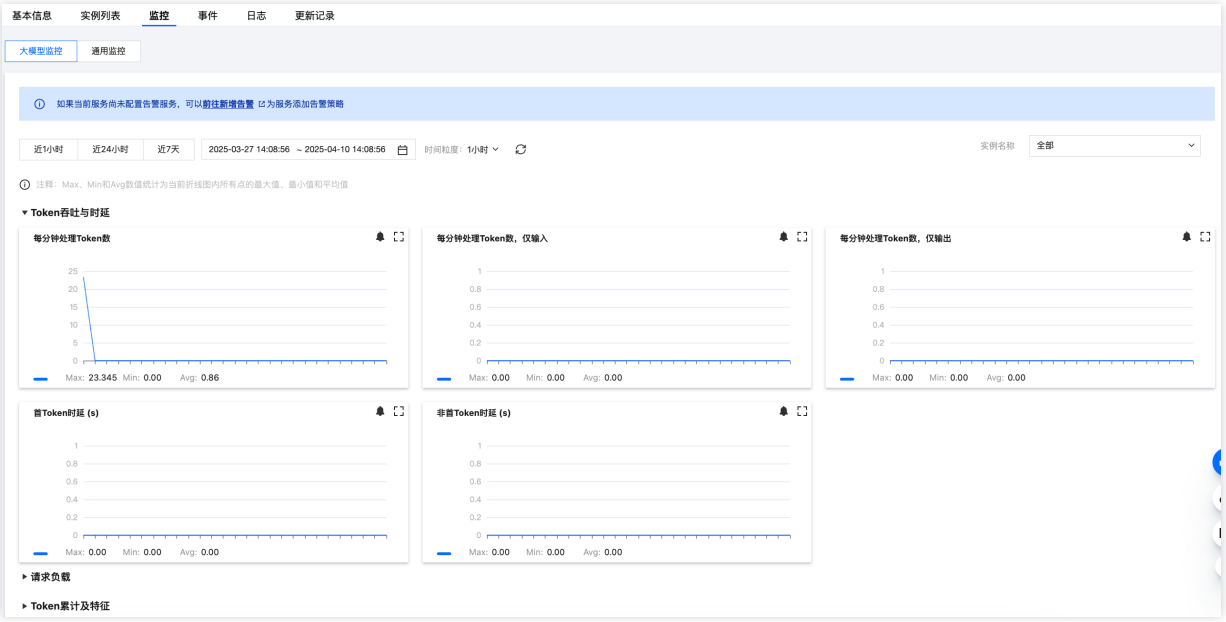


2. 在线服务监控页面，可跳转到 可观测平台的 告警策略 里，为服务添加告警策略。



3. 在线服务列表页面，单击服务名称进入版本列表页后，单击服务版本名称进入版本详情页面，可查看服务监控、事件监控、运行日志。其中服务监控分为大模型监控和通用监控：
    - 大模型监控：
        - Token 吞吐与时延：每分钟处理 Token 数（全口径/仅输入/仅输出）、首 Token 时延和非首 Token 时延。
        - 请求负载：处理中请求数和排队中请求数。
        - Token 累计与特征：已处理 Token 总量（全口径/仅输入/仅输出）、输入平均长度和输出平均长度。
    - 通用监控：
        - 流量信息：网络流量、QPS、QPS 限流和并发请求数。

- 资源信息：CPU 使用率、MEM 使用率、显存使用率和 GPU 使用率。
- 实例信息：实例数量和运行中实例数量。



服务更新

已部署的服务支持更新实例调节信息用于调整扩缩容策略，支持更新实例容器信息用于更新迭代模型，且在多实例的情况下更新服务时，后台会对多实例进行分批滚动更新，不影响生产业务对模型服务的调用。

1. 在线服务列表页面，单击服务名称进入版本列表页后，单击服务版本更新操作进入服务更新页面。
   更新支持编辑修改副本设置、请求限流、副本调节等服务参数。您也可通过更新升级基底模型版本。请注意：在单实例的情况下更新操作会导致服务重启，服务调用暂时不可用；在多实例的情况下更新的升级操作您可通过更新参数配置页面的滚动更新策略进行配置。

2. 若需进行服务扩缩容操作，可直接单击扩缩容进行快捷实例更新操作，当扩容的实例状态为运行中，流量分配至扩容实例。
3. 若需更新模型信息，可在实例容器模块修改模型文件或运行环境。
4. 配置信息确认无误后，单击启动服务完成服务参数配置的更新操作。
5. 在服务版本列表页单击服务版本名称，进入更新记录模块，可查看当前服务版本历史的更新记录信息。

   说明：

   当您更新文件存储系统里的文件内容时，可通过以下方式触发服务更新：

   - 重建实例，您可单击名称进入服务管理，再单击服务名称进入实例列表。

   - 更新服务时，填写一个环境变量，内容随意。

隔离异常实例

当服务的某个实例出现异常导致服务质量受影响时，平台支持用户在诊断期间手动隔离该实例的流量，而不需要重建实例 Pod。这样可以保留现场进行诊断，并支持在修复后手动恢复流量。
单击服务名称进入实例列表的详情页面，针对“实例状态”是“运行中”的实例，用户可单击“操作 > 隔离”按钮主动隔离指定实例，此后流量将不会再分配到被隔离的实例上。

针对已隔离的实例，用户可单击“操作 > 解除隔离”手动恢复该实例，解除隔离成功后，实例会重新恢复到“运行中”的状态。

使用自定义镜像发布在线服务开发指引

前言

本文档将向您介绍 TI-ONE 自定义镜像的2种方式，和需要遵循规范约束，再通过典型案例向您演示如何制作自定义镜像，发布在线服务。

基于平台 tiinfer 框架基础镜像制作自定义镜像

tiinfer 框架基础镜像说明

平台提供了内置 tiinfer 框架的基础推理镜像:
ccr.ccs.xxxxx.com/tione-public-images/ti-cloud-gpu-base-tiinfer:py38-cu111-1.0.0
基础镜像基于 centos 制作，其中包含的软件包有：

软件或包	版本
CUDA	11.1.1
python	3.9.13
cos-python-sdk-v5	1.9.14
coscmd	1.8.6.24
numpy	1.23.1
msgpack	1.0.5
opencv-python	4.6.0.66
opencv-contrib-python	4.6.0.66
pandas	1.4.3
Pillow	9.4.0
tiinfer	0.1.1
mosec-tiinfer	0.0.6

- 基础镜像的启动命令 /usr/local/service/ti-cloud-infer/entrypoint.sh
- entrypoint.sh 中的内容为：

```
#!/bin/bash
```

```
source /etc/profile
source /root/.bashrc
export LD_LIBRARY_PATH=/usr/local/python3/lib/python3.8/site-packages/torch/lib:/usr/local/openmpi/lib:/usr/local/nccl/lib:/usr/local/cuda/lib64:/usr/local/python3/lib:/usr/local/python3/lib64:/usr/local/openmpi/lib:/usr/local/gcc/lib:/usr/local/gcc/lib64


MODEL_DIR=/data/model

echo "================== code path ${MODEL_DIR}=========="
cd ${MODEL_DIR}

if [ -f "requirements.txt" ]; then
  echo "============== install python requirements ======================="
  echo "python3 -m pip install -r requirements.txt"
  python3 -m pip install -r requirements.txt
  echo "============== install python requirements done ================="
fi

echo "====================== start serving ============================="
echo "python3 -m tiinfer"
export TI_MODEL_DIR=${MODEL_DIR}
python3 -m tiinfer --timeout 30000
```

- 启动逻辑为：
  i. 读取环境变量 ${MODEL_DIR} 目录下的 requirements.txt 文件，使用 pip 安装其中指定的依赖 python 包。
  ii. tiinfer 框架会读取环境变量 ${MODEL_DIR} 下的文件，加载模型后，启动一个 HTTP 服务并监听在环境变量 ${REST_PORT} 定义的端口。
  iii. tiinfer 框架启动时，会从 model_service.py 文件中加载模型。

自定义镜像规范

1. Dockerfile 文件中添加对基础镜像的引用，例如：
   FROM ccr.ccs.xxxx.com/tione-public-images/ti-cloud-gpu-base-tiinfer:py38-cu111-1.0.0
2. 自定义逻辑实现集中在 model_service.py 文件及 entrypoint.sh 文件的修改。
3. 使用 CFS、COS、或 GooseFS 作为模型来源时，平台默认将源路径下的模型文件（包括子目录），放在服务实例的 /data/model 目录下。因此自定义的代码及数据不能置于 /data/model目录，否则会被平台覆盖。

制作镜像

本案例介绍了基于 tiinfer 框架基础镜像，通过修改 model_service.py 及 entrypoint.sh 文件，实现一个简单的加法器。
注意：本案例不使用平台提供的模型仓库功能托管模型，而是将模型、推理代码直接封装到镜像中，所以需要避免将模型、代码放到 /data/model 目录。

编写代码

一共包含三个文件：

文件	作用
model_service.py	按照 tiinfer 的要求，编写加法器模型。
entrypoint.sh	启动脚本，可在此自行安装更多的依赖包。
Dockerfile	负责将前两个文件拷贝到镜像中。

1. model_service.py 的内容：

```
from typing import Dict
import tiinfer

class AdderModel(tiinfer.Model):
 def __init__(self, model_dir: str):
    super().__init__(model_dir)

 def load(self) -> bool:
    self.ready = True
    return self.ready

 def preprocess(self, request: Dict) -> Dict:
    return request

 def predict(self, request: Dict) -> Dict:
    return {'result': request['a'] + request['b']}

 def postprocess(self, result: Dict) -> Dict:
    return result
```

2. entrypoint.sh 的内容：

```
#!/bin/bash
source /etc/profile
source /root/.bashrc
export LD_LIBRARY_PATH=/usr/local/python3/lib/python3.8/site-packages/torch/lib:/usr/local/openmpi/lib:/usr/local/nccl/lib:/usr/local/cuda/lib64:/usr/local/python3/lib:/usr/local/python3/lib64:/usr/local/openmpi/lib:/usr/local/gcc/lib:/usr/local/gcc/lib64

MODEL_DIR=/opt/model

echo "================== code path ${MODEL_DIR}=========="
cd ${MODEL_DIR}

if [ -f "requirements.txt" ]; then
  echo "============== install python requirements ======================="
```

```
  echo "python3 -m pip install -r requirements.txt"
  python3 -m pip install -r requirements.txt
  echo "============== install python requirements done ================="
fi

echo "====================== start serving ============================="
echo "python3 -m tiinfer"
export TI_MODEL_DIR=${MODEL_DIR}
python3 -m tiinfer --timeout 30000
```

注意: 上述代码中的 MODEL_DIR=/opt/model 这一行，将启动目录由默认的 /data/model 改为 /opt/model ，避免被平台覆盖。

3. Dockerfile 的内容：

```
FROM ccr.ccs.xxxxx.com/tione-public-images/ti-cloud-gpu-base-tiinfer:py38-cu111-1.0.0

COPY model_service.py /opt/model/model_service.py
COPY entrypoint.sh ./entrypoint.sh
RUN chmod +x ./entrypoint.sh
```

需要注意的是，上述代码将 model_service.py 拷贝到 /opt/model 目录，而非默认的 /data/model 目录，避免被平台覆盖。

打包镜像

1. 整体步骤：
   - 本地配置 docker 环境，并开通容器镜像服务；
   - 创建命名空间及新建个人镜像仓库；
   - 编译自定义推理镜像，推送到个人镜像仓库；
   - 在启动模型服务时，实例容器栏选择不使用模型文件，选择运行环境进入个人镜像仓库列表，选择上一步推送的自定义镜像环境；
   - 配置好参数，启动服务。

2. 详细说明：
   执行如下命令来打包：

```
docker build . --tag ccr.ccs.xxxxxx.com/YOUR_NAMESPACE/YOUR_IMAGENAME
```

打包完成后，可以通过如下方式在本地检查服务运行是否正常：

- 执行 docker run -d --name myinfer ccr.ccs.xxxxxx.com/YOUR_NAMESPACE/YOUT_IMAGENAME 将服

务运行起来；
- 执行 docker exec -it myinfer bash 进入容器中；
- 在容器中执行 curl http://127.0.0.1:8501/v1/models/m:predict -d '{"a": 1, "b": 2}' 得到正确返回: {"result": 3}
- 退出容器，回到本地环境，上传镜像： docker push ccr.ccs.xxxxxx.com/YOUR_NAMESPACE/YOUR_IMAGENAME 。

基于其他推理框架制作自定义镜像

平台支持使用其它推理框架，通过自定义镜像的方式来部署模型在线服务。

自定义镜像规范

1. 服务必须以 HTTP 协议接收请求，并且只支持 POST 方法。
2. 使用 CFS 作为模型来源时，平台默认将源路径下的模型文件（包括子目录），放在服务实例的 /data/model 目录下。因此自定义的代码及数据不能置于 /data/model 目录，否则会被平台覆盖。
3. 镜像在本地经过验证，可以正常提供服务。

上传自定义镜像并发布推理服务

上传自定义镜像

1. 登录容器镜像服务。
2. 在镜像仓库页面，单击新建。
3. 上传镜像。
   单击镜像仓库中后的快捷指令，查看操作命令，上传镜像。

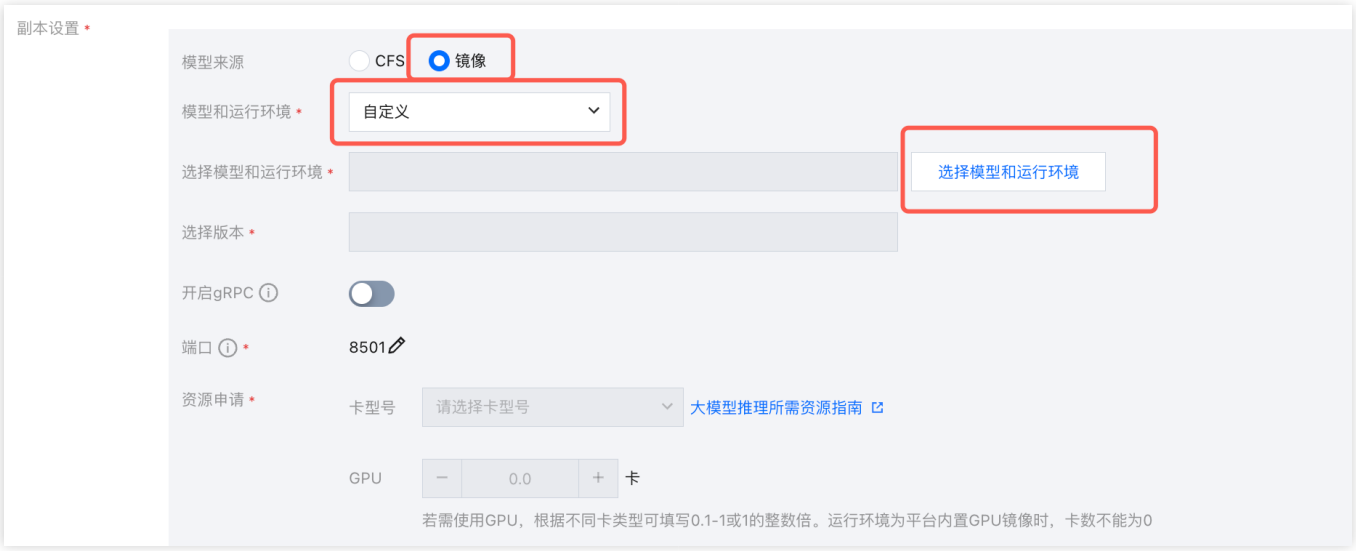
发布在线服务

在 TI-ONE【在线服务】页面，单击新建服务。

方式一：模型打包在镜像中使用

若您已将模型打包在镜像中，可以直接使用镜像发布服务。
【服务实例】详细配置：

- 模型来源选择镜像。
- 模型和运行环境选择自定义。
- 选择模型和运行环境选择之前上传的镜像。
- 端口号填写提供服务的端口号。



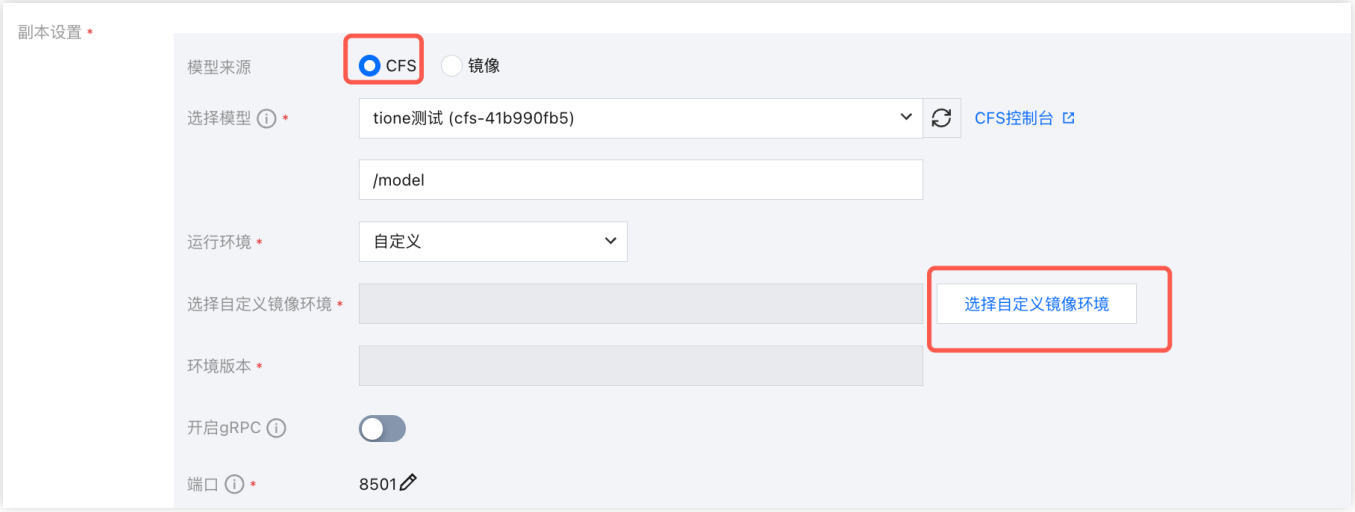
方式二：模型在 CFS，挂载到容器中使用

若您的镜像仅为服务运行环境，模型可以上传到CFS后，挂载到容器内部。模型会挂载到 /data/model 目录下，您的服务可以从此目录加载模型。

【服务实例】详细配置：

- 模型来源选择 CFS

- 选择模型选择模型存储的 CFS，并填写模型存储在 CFS 中的路径
- 模型和运行环境选择自定义
- 选择模型和运行环境选择之前上传的镜像
- 端口号填写提供服务的端口号



资源组管理
资源组简介

总览

TI-ONE 平台是以资源组的形式来管理用户已购买的 CVM 机器节点，并进行资源调度。其机器来源主要为“从 TI-ONE 平台购买”。
您可将同地域下的 CVM 添加至同一资源组进行管理，后续使用该资源组的节点资源去启动训练、推理任务。

新建资源组

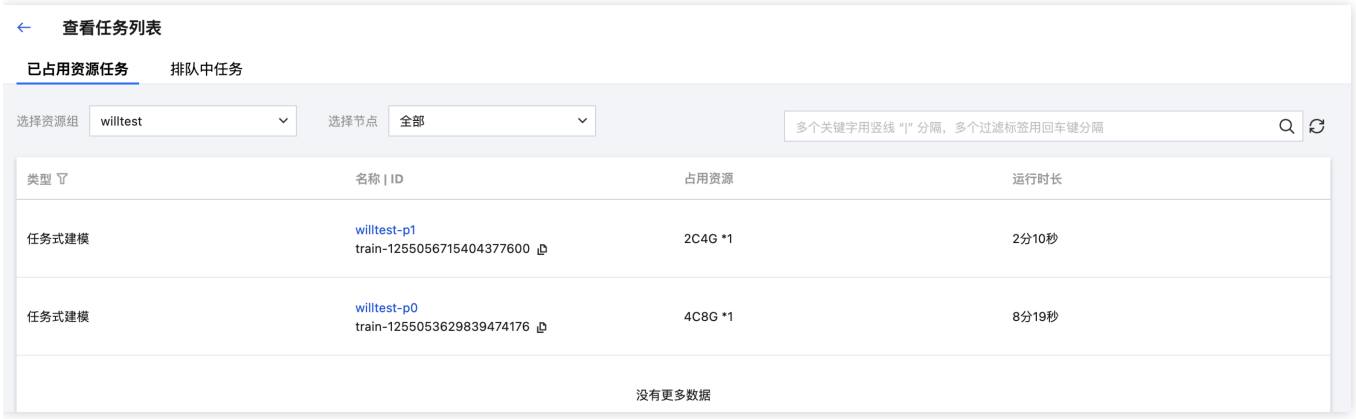
1. 注册账号，并完成登录认证。
2. 登录 TI-ONE 控制台，进入资源组管理页面。
3. 在页面最上方选择业务所在的地域。
4. 单击新建资源组，拉起资源组的新建弹窗。

5. 填写资源组名称，勾选风险提示，单击确认完成创建。

查看任务列表

单击资源组列表页查看任务列表，可以查看该资源组下占用资源的任务和排队中的任务。

进入已占用资源任务Tab页，可查看该资源组下所有正在占用资源的任务，同时您可以在该页面切换资源组，选择具体某一个节点进行查看。



进入排队中任务Tab页，可查看该资源组下正在排队的训练任务。请注意，当资源组的排队策略为严格先进先出的时候，该序号即为任务的调度顺序；当排队策略为非严格先进先出的时候，该序号不代表实际调度顺序，详细规则请见 调度策略说明。

增加节点

1. 单击资源组名称后进入详情页，单击增加节点，进入购买页。

2. 在购买页，根据提示配置以下信息。

字段	描述
计费模式	仅支持按量计费模式
地域	默认展示该资源组所在地域
实例规格	该地域下，选择需要从 TI-ONE 平台购买并使用的 CVM 机器 说明： 实例规格中可选择的机型范围及对应价格，由平台运营端维护。
节点数量	表示购买实例规格的数量

3. 核对购买信息后提交订单，选择支付方式并完成支付。

4. 完成购买后，进入控制台查收节点。
   节点有如下9个状态：

节点状态	状态说明
部署中	CVM 资源在 TI-ONE 平台初始化中
部署失败	CVM 资源在 TI-ONE 平台部署失败
运行中	CVM 资源在 TI-ONE 平台部署成功后投入使用
释放中	CVM 资源在 TI-ONE 平台使用时间已到期自动释放，或被用户主动释放

节点状态	状态说明
已释放	CVM 资源在 TI-ONE 平台使用时间到期自动释放中，或被用户主动释放中
异常	CVM 资源在 TI-ONE 平台使用异常
已不使用	CVM 资源已不在 TI-ONE 平台使用
挂载中	CVM 资源将其挂载好的数据盘管理进 TI-ONE 平台里使用
维修中	CVM资源出现故障，自动维修中

资源组操作

批量释放

进入资源组详情页，单击批量释放可以对多个选中节点进行释放，释放后将停止计费，同时 CVM 机器不再支持在 TI-ONE 平台使用。

删除

进入资源组详情页，单击删除，可以删除整个资源组，删除后不可恢复。

> 注意：
>
> 仅在资源组内没有使用节点时可进行删除，使用节点状态包括“运行中”、“部署中”、“异常”、“释放中”、“维修中”、“挂载中”。

节点操作

释放

进入资源组详情页，单击释放可将 “运行中”或“异常”状态的节点将停止计费，同时 CVM 机器不再支持在 TI-ONE 平台使用。

历史节点记录

在进入某一个资源组后，单击顶部 历史节点记录 可查看“部署失败”和“已不使用”状态的节点列表。

调度策略说明

整体说明

TI-ONE 资源组在调度训练任务和在线服务时，支持“排队策略”及“优先级调度策略”。

- 排队策略：当资源不足时，默认排队策略是按照任务/服务提交时间的先后顺序先进先出。用户也可以在资源组配置遍历策略，优先调度队列中满足资源要求的任务/服务。详见 排队策略配置说明。
- 优先级调度策略：支持根据标签设置任务/服务的优先级（P0最高-P9最低），高优先级任务/服务会默认抢占低优先级任务，低优先级任务被抢占后会重新进入排队队列，详细使用说明请见 优先级调度说明。
  - 备注：通过标签设置任务优先级的功能仅支持“任务式建模”及“在线服务”模块。
  - 服务优先原则：高优的在线服务可以抢占低优任务式建模的资源，但高优任务式建模不会抢占低优在线服务的资源。且同时，高优的在线服务也不会抢占低优的在线服务。

    > 说明：
    >
    > - 针对 CPU 任务，默认会按照负载均衡的策略均分到资源组不同节点。
    > - 针对 GPU 任务，默认会按照最小化碎片的方式，优先调度到同一台节点。
    > - 不同卡类型的任务不会互相阻塞，例如T4卡的任务不会影响A100卡的任务。

排队策略配置说明

DefaultQueue 表示资源组内训练任务的默认排队策略，仅支持“任务式建模”及“在线服务”模块。

- 其中默认值为0，设置为0代表 严格先进先出 ：不管当前空闲多少资源，在等待队列中取到最早提交的任务，如果资源足够则调度执行，如果不够则等待；
- 设置为1代表 资源尽量利用的先进先出 ：根据当前空闲的资源情况，从队列中按照时间顺序找到第一个当前资源满足的任务，调度执行。

```
{
  "Version": "1.0",
  "ResourceRule": {
    "DefaultPriority": 0,
    "DefaultQueue": 0,
    "Preempted": 1
  },
  "TaskRules": []
}
```

参数说明；

- DefaultPriority：默认优先级，优先级为0-9，P0最高，默认值为0。
- DefaultQueue：默认排队策略，可选值0、1、2，默认值为0。

- 0代表严格先进先出：不管当前空闲多少资源，在等待队列中取到最早提交的任务，如果资源足够则调度执行，如果不够则等待；
- 1表示在优先级范围内的遍历策略，也就是说同一优先级的任务可以不严格按照任务提交时间按顺序出队，可以看资源空闲情况插队；
- 2表示全部队列的遍历策略，也就是说整个排队队列都按照遍历策略，只要资源满足，就出队。

- Preempted：默认优先级抢占策略，可选值0和1，默认值为1。
  - 0表示队列中的高优任务不会抢占已经在运行中的低优任务；
  - 1表示队列中的高优任务会抢占已经在运行中的低优任务。

优先级调度说明

策略语法示例

在资源组维度，支持配置调度策略的描述文件。其中，DefaultPriority 表示默认优先级，TaskRules 中可以按照以下示例通过任务标签设置优先级，其中 ValueType 默认为 Tag，后续可能会根据实际需求不断补充ValueType。
下面是一个示例语句，实现的场景为：该资源组下的训练任务默认优先级为P0，如果任务的标签符合任务类型-语音任务，则该任务的优先级为P1（注意：通过标签设置任务优先级的功能仅支持任务式建模及在线服务）。

- 低优任务会被高优任务或服务抢占，但低优的服务只会被高优服务抢占。被抢占后的任务/服务将进入排队队列，继续等待。
- 服务优先原则：高优的在线服务可以抢占低优任务式建模的资源，但高优任务式建模不会抢占低优在线服务的资源。且同时，高优的在线服务也不会抢占低优的在线服务。

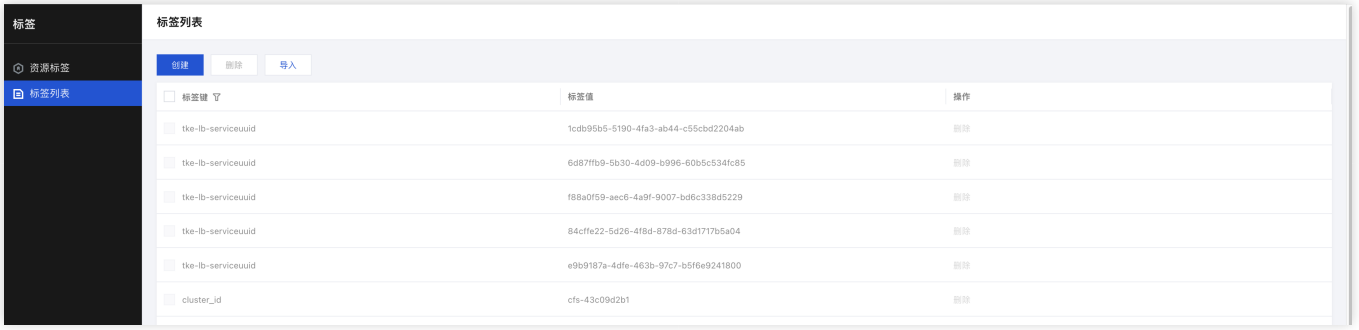
> 注意：
>
> 下述策略语句中的#注释部分不符合 json 语法，仅做文档展示，粘贴到平台需要清除。

```
{
 "Version":"1.0", // json格式的版本
 "ResourceRule":{
  // 资源组策略
  "DefaultPriority": 0, // 默认优先级，优先级0-9
  "DefaultQueue": 1,
  "Preempted": 1
 },
 "TaskRules":[
  // 任务调度策略，是规则匹配列表
  {
   // 每个规则是属性和调度的匹配关系
   "AttrRules":[{
    "ValueType":"Tag",
    "Key":"任务类型",
    "Value":"语音任务"
   }],
   "ScheduleRule":{
    "Priority": 1 // 设置优先级，优先级0-9
```

```
      }
    }
  ]
}
```

具体操作流程（以任务式建模为例）

1. 登录 标签控制台，进入标签列表。



单击创建标签，输入标签键任务类型，标签值语音任务。

2. 进入 TI-ONE 资源组管理页面，选择具体的资源组，单击编辑，在调度策略一栏输入上述的策略描述语句。

3. 新建训练任务的时候，如果需要当前这个任务的优先级为低优的话，需要在新建任务的时候给这个任务打上标签任务类型 > 语音任务，那么这个任务在调度的时候就是低优任务，会被抢占，自动进入排队队列。


TI-ONE 训练平台
大模型广场
训练工坊
任务式建模
开发机
Git存储库
模型服务
资源组管理
基本信息
任务名称 *
请输入任务名称
0/256
请输入不超过256个字符，仅支持中英文、数字、下划线"_"、短横"-"，只能以中英文、数字开头
地域 *
重庆
训练镜像 *
内置通用镜像 / PyTorch / tilearn-llm0.9-torch2.3-py3.10-cuda12.4-gpu
训练模式 *
DDP
MPI
Ray
资源组 *
请选择资源组
资源申请 *
卡型号
请选择卡型号
单节点GPU
0.0
卡
若需使用GPU，根据不同卡类型可填写0.1-1或1的整数倍。运行环境为平台内置GPU镜像时，卡数不能为0
单节点CPU *
1.0
核
单节点内存 *
1.00
G
节点数 *
1
个
标签
任务类型
语音任务
+ 添加

GPU 虚拟化

概述

TI-ONE 平台提供 GPU 虚拟化功能，可将同一张 GPU 卡的算力分配给不同训练任务和推理服务使用，提升资源分配灵活性和使用效率。

支持的 GPU 型号

平台的 GPU 虚拟化功能已支持的 GPU 型号如下：
T4、V100、A100、A10、PNV5b、A800、HCCA100、HCCPNV6、PNV6。

功能使用说明

前置条件

您需要在资源组管理页面提前创建资源组，并将您的 CVM 以节点形式添加至资源组。

使用方式

在新建开发机实例、任务式建模训练任务或在线服务时，当选择您的资源组后，如果其中包括已支持虚拟化的 GPU 卡型号，则可以在配置资源时，选择 GPU 卡数为0.1至1之间的数值。

注意事项

由于市面上的 GPU 卡型号在不断增多，相关驱动也在持续更新，因此平台的 GPU 虚拟化功能对于 GPU 型号的支持是逐步扩展的。平台新支持虚拟化的 GPU 可能无法在存量资源组中使用虚拟化功能，请您在使用时关注平台的相关提示。

- 如果在创建资源组时添加了平台尚未支持虚拟化的 GPU 卡型号的节点，则在平台后续支持该卡型号虚拟化后，仍无法使用该资源组内上述 GPU 的虚拟化功能。
- 在上述情况下，您也无法上述节点直接移动至其他资源组；但可以将节点从资源组中移除，并重新添加至新创建的资源组，从而可以使用虚拟化功能。

相关协议
开源软件信息

开源模型声明如下：
The following datasets and/or models are provided under their respective licenses or terms. Credits are given to their authors, and you should comply with these licenses and terms accordingly.

Model Licensed under the DEEPSEEK LICENSE AGREEMENT:

1. DeepSeek-V3
   Copyright (c) 2023 DeepSeek
   Terms of the DEEPSEEK LICENSE AGREEMENT:

DEEPSEEK LICENSE AGREEMENT

Version 1.0, 23 October 2023

Copyright (c) 2023 DeepSeek

Section I: PREAMBLE

Large generative models are being widely adopted and used, and have the potential to transform the way individuals conceive and benefit from AI or ML technologies.

Notwithstanding the current and potential benefits that these artifacts can bring to society at large, there are also concerns about potential misuses of them, either due to their technical limitations or ethical considerations.

In short, this license strives for both the open and responsible downstream use of the accompanying model. When it comes to the open character, we took inspiration from open source permissive licenses regarding the grant of IP rights. Referring to the downstream responsible use, we added use-based restrictions not permitting the use of the model in very specific scenarios, in order for the licensor to be able to enforce the license in case potential misuses of the Model may occur. At the same time, we strive to promote open and responsible research on generative models for content generation.

Even though downstream derivative versions of the model could be released under different licensing terms, the latter will always have to include - at minimum - the same use-based restrictions as the ones in the original license (this license). We believe in the intersection between open and responsible AI development; thus, this agreement aims to strike a balance between both in order to enable responsible open-science in the field of AI.

This License governs the use of the model (and its derivatives) and is informed by the model card associated with the model.

NOW THEREFORE, You and DeepSeek agree as follows:

1. Definitions
   "License" means the terms and conditions for use, reproduction, and Distribution as defined in this document.
   "Data" means a collection of information and/or content extracted from the dataset used with the Model, including to train, pretrain, or otherwise evaluate the Model. The Data is not licensed under this License.
   "Output" means the results of operating a Model as embodied in informational content resulting therefrom.
   "Model" means any accompanying machine-learning based assemblies (including checkpoints), consisting of learnt weights, parameters (including optimizer states), corresponding to the model architecture as embodied in the Complementary Material, that have been trained or tuned, in whole or in part on the Data, using the Complementary Material.
   "Derivatives of the Model" means all modifications to the Model, works based on the Model, or any other model which is created or initialized by transfer of patterns of the weights, parameters, activations or output of the Model, to the other model, in order to cause the other model to perform similarly to the Model, including - but not limited to - distillation methods entailing the use of intermediate data representations or methods based on the generation of synthetic data by the Model for training the other model.
   "Complementary Material" means the accompanying source code and scripts used to define, run, load, benchmark or evaluate the Model, and used to prepare data for training or evaluation, if any. This includes any accompanying documentation, tutorials, examples, etc, if any.
   "Distribution" means any transmission, reproduction, publication or other sharing of the Model or Derivatives of the Model to a third party, including providing the Model as a hosted service made available by electronic or other remote means - e.g. API-based or web access.
   "DeepSeek" (or "we") means Beijing DeepSeek Artificial Intelligence Fundamental Technology Research Co., Ltd., Hangzhou DeepSeek Artificial Intelligence Fundamental Technology Research Co., Ltd. and/or any of their affiliates.
   "You" (or "Your") means an individual or Legal Entity exercising permissions granted by this License and/or making use of the Model for whichever purpose and in any field of use, including usage of the Model in an end-use application - e.g. chatbot, translator, etc.
   "Third Parties" means individuals or legal entities that are not under common control with DeepSeek or You.

Section II: INTELLECTUAL PROPERTY RIGHTS

Both copyright and patent grants apply to the Model, Derivatives of the Model and Complementary

Material. The Model and Derivatives of the Model are subject to additional terms as described in Section III.

2. Grant of Copyright License. Subject to the terms and conditions of this License, DeepSeek hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare, publicly display, publicly perform, sublicense, and distribute the Complementary Material, the Model, and Derivatives of the Model.

3. Grant of Patent License. Subject to the terms and conditions of this License and where and as applicable, DeepSeek hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this paragraph) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Model and the Complementary Material, where such license applies only to those patent claims licensable by DeepSeek that are necessarily infringed by its contribution(s). If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Model and/or Complementary Material constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for the Model and/or works shall terminate as of the date such litigation is asserted or filed.

Section III: CONDITIONS OF USAGE, DISTRIBUTION AND REDISTRIBUTION

4. Distribution and Redistribution. You may host for Third Party remote access purposes (e.g. software-as-a-service), reproduce and distribute copies of the Model or Derivatives of the Model thereof in any medium, with or without modifications, provided that You meet the following conditions:
   a. Use-based restrictions as referenced in paragraph 5 MUST be included as an enforceable provision by You in any type of legal agreement (e.g. a license) governing the use and/or distribution of the Model or Derivatives of the Model, and You shall give notice to subsequent users You Distribute to, that the Model or Derivatives of the Model are subject to paragraph 5. This provision does not apply to the use of Complementary Material.
   b. You must give any Third Party recipients of the Model or Derivatives of the Model a copy of this License;
   c. You must cause any modified files to carry prominent notices stating that You changed the files;
   d. You must retain all copyright, patent, trademark, and attribution notices excluding those notices that do not pertain to any part of the Model, Derivatives of the Model.
   e. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions - respecting paragraph 4.a. – for use, reproduction, or Distribution of Your modifications, or for any such Derivatives of the Model as a whole, provided Your use, reproduction, and Distribution of the Model otherwise complies with the conditions stated in this License.

5. Use-based restrictions. The restrictions set forth in Attachment A are considered Use-based

restrictions. Therefore You cannot use the Model and the Derivatives of the Model for the specified restricted uses. You may use the Model subject to this License, including only for lawful purposes and in accordance with the License. Use may include creating any content with, finetuning, updating, running, training, evaluating and/or reparametrizing the Model. You shall require all of Your users who use the Model or a Derivative of the Model to comply with the terms of this paragraph (paragraph 5).

6. The Output You Generate. Except as set forth herein, DeepSeek claims no rights in the Output You generate using the Model. You are accountable for the Output you generate and its subsequent uses. No use of the output can contravene any provision as stated in the License.

Section IV: OTHER PROVISIONS

7. Updates and Runtime Restrictions. To the maximum extent permitted by law, DeepSeek reserves the right to restrict (remotely or otherwise) usage of the Model in violation of this License.

8. Trademarks and related. Nothing in this License permits You to make use of DeepSeek' trademarks, trade names, logos or to otherwise suggest endorsement or misrepresent the relationship between the parties; and any rights not expressly granted herein are reserved by DeepSeek.

9. Personal information, IP rights and related. This Model may contain personal information and works with IP rights. You commit to complying with applicable laws and regulations in the handling of personal information and the use of such works. Please note that DeepSeek's license granted to you to use the Model does not imply that you have obtained a legitimate basis for processing the related information or works. As an independent personal information processor and IP rights user, you need to ensure full compliance with relevant legal and regulatory requirements when handling personal information and works with IP rights that may be contained in the Model, and are willing to assume solely any risks and consequences that may arise from that.

10. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, DeepSeek provides the Model and the Complementary Material on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Model, Derivatives of the Model, and the Complementary Material and assume any risks associated with Your exercise of permissions under this License.

11. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent

acts) or agreed to in writing, shall DeepSeek be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Model and the Complementary Material (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if DeepSeek has been advised of the possibility of such damages.

12. Accepting Warranty or Additional Liability. While redistributing the Model, Derivatives of the Model and the Complementary Material thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of DeepSeek, and only if You agree to indemnify, defend, and hold DeepSeek harmless for any liability incurred by, or claims asserted against, DeepSeek by reason of your accepting any such warranty or additional liability.

13. If any provision of this License is held to be invalid, illegal or unenforceable, the remaining provisions shall be unaffected thereby and remain valid as if such provision had not been set forth herein.

14. Governing Law and Jurisdiction. This agreement will be governed and construed under PRC laws without regard to choice of law principles, and the UN Convention on Contracts for the International Sale of Goods does not apply to this agreement. The courts located in the domicile of Hangzhou DeepSeek Artificial Intelligence Fundamental Technology Research Co., Ltd. shall have exclusive jurisdiction of any dispute arising out of this agreement.
END OF TERMS AND CONDITIONS
Attachment A
Use Restrictions
You agree not to use the Model or Derivatives of the Model:

- In any way that violates any applicable national or international law or regulation or infringes upon the lawful rights and interests of any third party;
- For military use in any way;
- For the purpose of exploiting, harming or attempting to exploit or harm minors in any way;
- To generate or disseminate verifiably false information and/or content with the purpose of harming others;
- To generate or disseminate inappropriate content subject to applicable regulatory requirements;
- To generate or disseminate personal identifiable information without due authorization or for unreasonable use;
- To defame, disparage or otherwise harass others;
- For fully automated decision making that adversely impacts an individual’s legal rights or otherwise

creates or modifies a binding, enforceable obligation;
- For any use intended to or which has the effect of discriminating against or harming individuals or groups based on online or offline social behavior or known or predicted personal or personality characteristics;
- To exploit any of the vulnerabilities of a specific group of persons based on their age, social, physical or mental characteristics, in order to materially distort the behavior of a person pertaining to that group in a manner that causes or is likely to cause that person or another person physical or psychological harm;
- For any use intended to or which has the effect of discriminating against individuals or groups based on legally protected characteristics or categories.

Models and software Licensed under the MIT:

1. DeepSeek-R1
   Copyright (c) 2023 DeepSeek

2. DeepSeek-V3-0324
   Copyright (c) 2023 DeepSeek

3. DeepSeek-R1-Distill-Qwen-1.5B
   Copyright (c) 2023 DeepSeek

4. DeepSeek-R1-Distill-Qwen-7B
   Copyright (c) 2023 DeepSeek

5. DeepSeek-R1-Distill-Qwen-14B
   Copyright (c) 2023 DeepSeek

6. DeepSeek-R1-Distill-Qwen-32B
   Copyright (c) 2023 DeepSeek

7.DeepSeek-V3 (Code)
Copyright (c) 2023 DeepSeek

Terms of the MIT:

MIT License

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Models Licensed under the TENCENT HUNYUAN COMMUNITY LICENSE AGREEMENT:

1. Hunyuan-A52B-Instruct(Hunyuan-Large-Instruct)
   Copyright © 2024 Tencent. All Rights Reserved.

2. Hunyuan-A52B-Pretrain
   Copyright © 2024 Tencent. All Rights Reserved.

3. Hunyuan-A52B-Instruct-FP8
   Copyright © 2024 Tencent. All Rights Reserved.

Tencent Hunyuan is licensed under the Tencent Hunyuan Community License Agreement, Copyright © 2024 Tencent. All Rights Reserved. The trademark rights of “Tencent Hunyuan” are owned by Tencent or its affiliate.

Terms of the TENCENT HUNYUAN COMMUNITY LICENSE AGREEMENT:

TENCENT HUNYUAN COMMUNITY LICENSE AGREEMENT

Tencent-Hunyuan-Large Release Date: November 5, 2024
THIS LICENSE AGREEMENT DOES NOT APPLY IN THE EUROPEAN UNION AND IS EXPRESSLY LIMITED TO THE TERRITORY, AS DEFINED BELOW.
By clicking to agree or by using, reproducing, modifying, distributing, performing or displaying any portion or element of the Tencent Hunyuan Works, including via any Hosted Service, You will be deemed to have recognized and accepted the content of this Agreement, which is effective immediately.

1. DEFINITIONS.
   a. “Acceptable Use Policy” shall mean the policy made available by Tencent as set forth in the Exhibit A.
   b. “Agreement” shall mean the terms and conditions for use, reproduction, distribution, modification, performance and displaying of Tencent Hunyuan Works or any portion or element thereof set forth herein.
   c. “Documentation” shall mean the specifications, manuals and documentation for Tencent Hunyuan made publicly available by Tencent.
   d. “Hosted Service” shall mean a hosted service offered via an application programming interface (API), web access, or any other electronic or remote means.
   e. “Licensee,” “You” or “Your” shall mean a natural person or legal entity exercising the rights granted by this Agreement and/or using the Tencent Hunyuan Works for any purpose and in any field of use.
   f. “Materials” shall mean, collectively, Tencent’s proprietary Tencent Hunyuan and Documentation (and any portion thereof) as made available by Tencent under this Agreement.
   g. “Model Derivatives” shall mean all: (i) modifications to Tencent Hunyuan or any Model Derivative of Tencent Hunyuan; (ii) works based on Tencent Hunyuan or any Model Derivative of Tencent Hunyuan; or (iii) any other machine learning model which is created by transfer of patterns of the weights, parameters, operations, or Output of Tencent Hunyuan or any Model Derivative of Tencent Hunyuan, to that model in order to cause that model to perform similarly to Tencent Hunyuan or a Model Derivative of Tencent Hunyuan, including distillation methods, methods that use intermediate data representations, or methods based on the generation of synthetic data Outputs by Tencent Hunyuan or a Model Derivative of Tencent Hunyuan for training that model. For clarity, Outputs by themselves are not deemed Model Derivatives.
   h. “Output” shall mean the information and/or content output of Tencent Hunyuan or a Model Derivative that results from operating or otherwise using Tencent Hunyuan or a Model Derivative, including via a Hosted Service.
   i. “Tencent,” “We” or “Us” shall mean THL A29 Limited.
   j. “Tencent Hunyuan” shall mean the large language models, text/image/video/audio/3D generation models, and multimodal large language models and their software and algorithms, including trained model weights, parameters (including optimizer states), machine-learning model code, inference-enabling code, training-enabling code, fine-tuning enabling code and other elements of the foregoing made publicly available by Us, including, without limitation to, Tencent-Hunyuan-Large

released at https://github.com/Tencent/Tencent-Hunyuan-Large, https://huggingface.co/tencent/Tencent-Hunyuan-Large.

k. “Tencent Hunyuan Works” shall mean: (i) the Materials; (ii) Model Derivatives; and (iii) all derivative works thereof.

l. “Territory” shall mean the worldwide territory, excluding the territory of the European Union.

m. “Third Party” or “Third Parties” shall mean individuals or legal entities that are not under common control with Us or You.

n. “including” shall mean including but not limited to.

2. GRANT OF RIGHTS.

We grant You, for the Territory only, a non-exclusive, non-transferable and royalty-free limited license under Tencent’s intellectual property or other rights owned by Us embodied in or utilized by the Materials to use, reproduce, distribute, create derivative works of (including Model Derivatives), and make modifications to the Materials, only in accordance with the terms of this Agreement and the Acceptable Use Policy, and You must not violate (or encourage or permit anyone else to violate) any term of this Agreement or the Acceptable Use Policy.

3. DISTRIBUTION.

You may, subject to Your compliance with this Agreement, distribute or make available to Third Parties the Tencent Hunyuan Works, exclusively in the Territory, provided that You meet all of the following conditions:

a. You must provide all such Third Party recipients of the Tencent Hunyuan Works or products or services using them a copy of this Agreement;

b. You must cause any modified files to carry prominent notices stating that You changed the files;

c. You are encouraged to: (i) publish at least one technology introduction blogpost or one public statement expressing Your experience of using the Tencent Hunyuan Works; and (ii) mark the products or services developed by using the Tencent Hunyuan Works to indicate that the product/service is “Powered by Tencent Hunyuan”; and

d. All distributions to Third Parties (other than through a Hosted Service) must be accompanied by a “Notice” text file that contains the following notice: “Tencent Hunyuan is licensed under the Tencent Hunyuan Community License Agreement, Copyright © 2024 Tencent. All Rights Reserved. The trademark rights of “Tencent Hunyuan” are owned by Tencent or its affiliate.”

You may add Your own copyright statement to Your modifications and, except as set forth in this Section and in Section 5, may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Model Derivatives as a whole, provided Your use, reproduction, modification, distribution, performance and display of the work otherwise complies with the terms and conditions of this Agreement (including as regards the Territory). If You receive Tencent Hunyuan Works from a Licensee as part of an integrated end user product, then this Section 3 of this Agreement will not apply to You.

4. ADDITIONAL COMMERCIAL TERMS.

If, on the Tencent Hunyuan version release date, the monthly active users of all products or services

made available by or for Licensee is greater than 100 million monthly active users in the preceding calendar month, You must request a license from Tencent, which Tencent may grant to You in its sole discretion, and You are not authorized to exercise any of the rights under this Agreement unless or until Tencent otherwise expressly grants You such rights.

5. RULES OF USE.

a. Your use of the Tencent Hunyuan Works must comply with applicable laws and regulations (including trade compliance laws and regulations) and adhere to the Acceptable Use Policy for the Tencent Hunyuan Works, which is hereby incorporated by reference into this Agreement. You must include the use restrictions referenced in these Sections 5(a) and 5(b) as an enforceable provision in any agreement (e.g., license agreement, terms of use, etc.) governing the use and/or distribution of Tencent Hunyuan Works and You must provide notice to subsequent users to whom You distribute that Tencent Hunyuan Works are subject to the use restrictions in these Sections 5(a) and 5(b).

b. You must not use the Tencent Hunyuan Works or any Output or results of the Tencent Hunyuan Works to improve any other large language model (other than Tencent Hunyuan or Model Derivatives thereof).

c. You must not use, reproduce, modify, distribute, or display the Tencent Hunyuan Works, Output or results of the Tencent Hunyuan Works outside the Territory. Any such use outside the Territory is unlicensed and unauthorized under this Agreement.

6. INTELLECTUAL PROPERTY.

a. Subject to Tencent's ownership of Tencent Hunyuan Works made by or for Tencent and intellectual property rights therein, conditioned upon Your compliance with the terms and conditions of this Agreement, as between You and Tencent, You will be the owner of any derivative works and modifications of the Materials and any Model Derivatives that are made by or for You.

b. No trademark licenses are granted under this Agreement, and in connection with the Tencent Hunyuan Works, Licensee may not use any name or mark owned by or associated with Tencent or any of its affiliates, except as required for reasonable and customary use in describing and distributing the Tencent Hunyuan Works. Tencent hereby grants You a license to use "Tencent Hunyuan" (the "Mark") in the Territory solely as required to comply with the provisions of Section 3(c), provided that You comply with any applicable laws related to trademark protection. All goodwill arising out of Your use of the Mark will inure to the benefit of Tencent.

c. If You commence a lawsuit or other proceedings (including a cross-claim or counterclaim in a lawsuit) against Us or any person or entity alleging that the Materials or any Output, or any portion of any of the foregoing, infringe any intellectual property or other right owned or licensable by You, then all licenses granted to You under this Agreement shall terminate as of the date such lawsuit or other proceeding is filed. You will defend, indemnify and hold harmless Us from and against any claim by any Third Party arising out of or related to Your or the Third Party's use or distribution of the Tencent Hunyuan Works.

d. Tencent claims no rights in Outputs You generate. You and Your users are solely responsible for Outputs and their subsequent uses.

7. DISCLAIMERS OF WARRANTY AND LIMITATIONS OF LIABILITY.

a. We are not obligated to support, update, provide training for, or develop any further version of the Tencent Hunyuan Works or to grant any license thereto.

b. UNLESS AND ONLY TO THE EXTENT REQUIRED BY APPLICABLE LAW, THE TENCENT HUNYUAN WORKS AND ANY OUTPUT AND RESULTS THEREFROM ARE PROVIDED "AS IS" WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES OF ANY KIND INCLUDING ANY WARRANTIES OF TITLE, MERCHANTABILITY, NONINFRINGEMENT, COURSE OF DEALING, USAGE OF TRADE, OR FITNESS FOR A PARTICULAR PURPOSE. YOU ARE SOLELY RESPONSIBLE FOR DETERMINING THE APPROPRIATENESS OF USING, REPRODUCING, MODIFYING, PERFORMING, DISPLAYING OR DISTRIBUTING ANY OF THE TENCENT HUNYUAN WORKS OR OUTPUTS AND ASSUME ANY AND ALL RISKS ASSOCIATED WITH YOUR OR A THIRD PARTY'S USE OR DISTRIBUTION OF ANY OF THE TENCENT HUNYUAN WORKS OR OUTPUTS AND YOUR EXERCISE OF RIGHTS AND PERMISSIONS UNDER THIS AGREEMENT.

c. TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL TENCENT OR ITS AFFILIATES BE LIABLE UNDER ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, TORT, NEGLIGENCE, PRODUCTS LIABILITY, OR OTHERWISE, FOR ANY DAMAGES, INCLUDING ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, EXEMPLARY, CONSEQUENTIAL OR PUNITIVE DAMAGES, OR LOST PROFITS OF ANY KIND ARISING FROM THIS AGREEMENT OR RELATED TO ANY OF THE TENCENT HUNYUAN WORKS OR OUTPUTS, EVEN IF TENCENT OR ITS AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF ANY OF THE FOREGOING.

8. SURVIVAL AND TERMINATION.

a. The term of this Agreement shall commence upon Your acceptance of this Agreement or access to the Materials and will continue in full force and effect until terminated in accordance with the terms and conditions herein.

b. We may terminate this Agreement if You breach any of the terms or conditions of this Agreement. Upon termination of this Agreement, You must promptly delete and cease use of the Tencent Hunyuan Works. Sections 6(a), 6(c), 7 and 9 shall survive the termination of this Agreement.

9. GOVERNING LAW AND JURISDICTION.

a. This Agreement and any dispute arising out of or relating to it will be governed by the laws of the Hong Kong Special Administrative Region of the People's Republic of China, without regard to conflict of law principles, and the UN Convention on Contracts for the International Sale of Goods does not apply to this Agreement.

b. Exclusive jurisdiction and venue for any dispute arising out of or relating to this Agreement will be a court of competent jurisdiction in the Hong Kong Special Administrative Region of the People's Republic of China, and Tencent and Licensee consent to the exclusive jurisdiction of such court with respect to any such dispute.

EXHIBIT A

ACCEPTABLE USE POLICY

Tencent reserves the right to update this Acceptable Use Policy from time to time.
Last modified: November 5, 2024

Tencent endeavors to promote safe and fair use of its tools and features, including Tencent Hunyuan. You agree not to use Tencent Hunyuan or Model Derivatives:

1. Outside the Territory;
2. In any way that violates any applicable national, federal, state, local, international or any other law or regulation;
3. To harm Yourself or others;
4. To repurpose or distribute output from Tencent Hunyuan or any Model Derivatives to harm Yourself or others;
5. To override or circumvent the safety guardrails and safeguards We have put in place;
6. For the purpose of exploiting, harming or attempting to exploit or harm minors in any way;
7. To generate or disseminate verifiably false information and/or content with the purpose of harming others or influencing elections;
8. To generate or facilitate false online engagement, including fake reviews and other means of fake online engagement;
9. To intentionally defame, disparage or otherwise harass others;
10. To generate and/or disseminate malware (including ransomware) or any other content to be used for the purpose of harming electronic systems;
11. To generate or disseminate personal identifiable information with the purpose of harming others;
12. To generate or disseminate information (including images, code, posts, articles), and place the information in any public context (including –through the use of bot generated tweets), without expressly and conspicuously identifying that the information and/or content is machine generated;
13. To impersonate another individual without consent, authorization, or legal right;
14. To make high-stakes automated decisions in domains that affect an individual’s safety, rights or wellbeing (e.g., law enforcement, migration, medicine/health, management of critical infrastructure, safety components of products, essential services, credit, employment, housing, education, social scoring, or insurance);
15. In a manner that violates or disrespects the social ethics and moral standards of other countries or regions;
16. To perform, facilitate, threaten, incite, plan, promote or encourage violent extremism or terrorism;
17. For any use intended to discriminate against or harm individuals or groups based on protected characteristics or categories, online or offline social behavior or known or predicted personal or personality characteristics;
18. To intentionally exploit any of the vulnerabilities of a specific group of persons based on their age, social, physical or mental characteristics, in order to materially distort the behavior of a person pertaining to that group in a manner that causes or is likely to cause that person or another person physical or psychological harm;

19. For military purposes;
20. To engage in the unauthorized or unlicensed practice of any profession including, but not limited to, financial, legal, medical/health, or other professional practices.

==========

End of the Attribution Notice of this project.