MindIE
1.0.RC3

MindIE Service 开发指南

文档版本 01

发布日期 2025-05-29

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

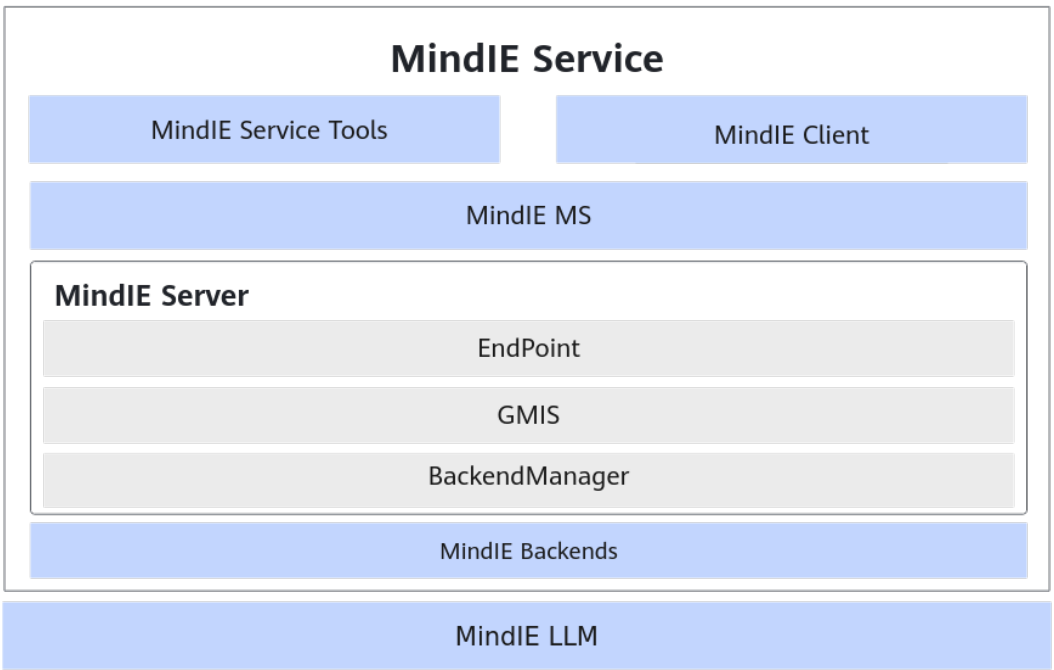
1 产品简介

产品介绍

MindIE Service是面向通用模型场景的推理服务化框架，通过开放、可扩展的推理服务化平台架构提供推理服务化能力，支持对接业界主流推理框架接口，满足大语言模型的高性能推理需求。

MindIE Service的组件包括MindIE Service Tools、MindIE Client、MindIE MS（MindIE Management Service）和MindIE Server，通过对接昇腾推理加速引擎带来大模型在昇腾环境中的性能提升，并逐渐以高性能和易用性牵引用户向全原生推理服务化框架迁移。其架构图如图1-1所示。

图 1-1 MindIE Service 架构图



- MindIE Service提供推理服务化部署和运维能力。
  - MindIE Service Tools：昇腾推理服务化工具；主要功能有大模型推理性能测试、精度测试和可视化能力，并且支持通过配置提升吞吐。

  - MindIE Client：昇腾推理服务化客户端；配套昇腾推理服务化MindIE Server提供完整的推理服务化能力，包括对接MindIE Server的通信协议、请求和返回的接口，提供给用户应用对接。
  - MindIE MS：服务策略管理，提供服务运维能力。主要功能包括模型Pod级和Pod内实例级管理、简化部署并提供服务质量监控、模型更新、故障重调度和自动扩缩负载均衡能力，不仅能够提升服务质量，同时也能提高推理硬件资源利用率。
  - MindIE Server：推理服务端；提供模型推理服务化能力，支持命令行部署RESTful服务。
    - EndPoint：提供RESTful接口；EndPoint面向推理服务开发者提供RESTful接口，推理服务化协议和接口封装，支持Triton/OpenAI/TGI/vLLM主流推理框架请求接口。
    - GMIS：模型推理调度器，提供多实例调度能力；实现从推理任务调度到任务执行的可扩展架构，适应各类推理方法。
    - BackendManager：模型执行后端，昇腾后端和自定义后端的管理模块；Backend管理模块面向不同推理引擎，不同模型，提供统一抽象接口，便于扩展，减少推理引擎、模型变化带来的修改。
  - MindIE Backends：支持昇腾MindIE LLM后端。
- MindIE LLM：提供大模型推理能力，同时提供多并发请求的调度功能。

支持的特性

须知

当前版本**PD分离部署**特性存在部分场景限制，开启前请仔细评估应用场景是否满足。

- 支持第三方框架接口：兼容Triton/OpenAI/TGI/vLLM第三方框架接口，详情请参见**4.2 EndPoint业务面RESTful接口**章节。
- 支持Transformer推理加速库（Ascend Transformer Boost）：支持基于Transformer推理加速库的模型接入，继承其加速能力，包括融合加速算子、量化等特性。
- 支持分布式多机推理：对于规格较大的大模型场景，单机可能无法完成推理，所以提供多台机器协同推理能力。当前仅Atlas 800I A2 推理产品支持分布式多机推理，需提前部署RoCE（RDMA over Converged Ethernet）网络，分布式多机推理部署方式请参见**3.5 分布式多机服务部署**章节。
- 支持智能运维管理：支持服务监控和NPU故障重调度，详情请参见**8.3.2.3 RESTful接口API**。
- 支持自动精度性能测试：支持测试模型的性能、精度，详情请参见**8.1.1 MindIE Benchmark**章节。
- 支持PD分离部署：Prefill和Decode分别部署在不同的实例上，减少互相干扰，降低时延，提升吞吐，其部署方式请参见**3.6 PD分离服务部署**章节。
- 支持多模态服务化：支持文本和图片多模态数据类型输入，功能详情请参见**7.4 多模态理解**章节。
- 支持Multi-Lora：使用Multi-Lora来执行基础模型和不同的LoRA权重进行推理，功能详情请参见**7.3 Multi-Lora**章节。

- 支持Function Call：支持Function Call函数调用，使大模型具备使用工具能力。功能详情请参见**7.5 Function Call**章节。
- 支持Prefix Cache：复用跨session的重复token序列对应的KV Cache，减少一部分前缀token的KV Cache计算时间，从而减少Prefill的时间。功能详情请参见**7.7 Prefix Cache**章节。
- 支持Splitfuse：将长提示词被分解成更小的块，并在多个推理轮次中进行调度，降低Prefill时延，功能详情请参见**7.6 Splitfuse**章节。

2 快速开始

2.1 环境准备

请参考《MindIE安装指南》中“**物理机部署MindIE**”章节进行环境的安装与部署，并参考**8.4.2 配置参数说明**根据用户需要配置参数。

2.2 启动服务

- MindIE Server可以部署兼容Triton/OpenAI/TGI/vLLM第三方框架接口的服务应用。推荐用户开启HTTPS通信，并按照《MindIE安装指南》中“配置MindIE > 配置MindIE Server > **单机推理**”章节，配置开启HTTPS通信所需服务证书、私钥等证书文件。
- MindIE Server启动的默认IP地址和端口号为https://127.0.0.1:1025，用户可修改config.json文件中的“ipAddress”和“port”参数来配置启动IP地址与端口号。
- MindIE Server可实现服务状态查询，模型信息查询，文本/流式推理等功能。

须知

HTTP缺乏必要的安全机制，容易受到数据泄露、数据篡改和中间人攻击的威胁，请谨慎使用。

步骤1 两种启动服务方法如下所示，以*{MindIE安装目录}*为例，用户可自行选择启动目录。

- （推荐）使用后台进程方式启动服务。后台进程方式启动服务后，关闭窗口时进程也会保留。


```
nohup ./bin/mindieservice_daemon > output.log 2>&1 &
```

在标准输出流捕获到的文件中，打印如下信息说明启动成功。

```
Daemon start success!
```

- 直接启动服务。

```
./bin/mindieservice_daemon
```

回显如下则说明启动成功。

```
Daemon start success!
```

须知

- bin目录按照安全要求，目录权限为550，没有写权限，但执行推理过程中，算子会在当前目录生成kernel_meta 文件夹，需要写权限，因此不能直接在bin启动mindieservice_daemon。
- Ascend-cann-toolkit工具会在执行服务启动的目录下生成kernel_meta_temp_xxxx目录，该目录为算子的cce文件保存目录。因此需要在当前用户拥有写权限目录下（例如Ascend-mindie-server_*{version}*_linux-*{arch}*目录，或者用户在Ascend-mindie-server_*{version}*_linux-*{arch}*目录下自行创建临时目录）启动推理服务。
- 如需切换用户，请在切换用户后执行**rm -f /dev/shm/***命令，删除由之前用户运行创建的共享文件。避免切换用户后，该用户没有之前用户创建的共享文件的读写权限，造成推理失败。
- 标准输出流捕获到的文件output.log支持用户自定义文件和路径。

步骤2 用户可使用HTTPS客户端（Linux curl命令，Postman工具等）发送HTTPS请求，此处以Linux curl命令为例进行说明。

重开一个窗口，使用以下命令发送请求。例如列出当前模型列表：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X GET https://127.0.0.1:1025/v1/models
```

说明

- --cacert：验签证书文件路径。
- ca.pem为MindIE Server服务端证书的验签证书/根证书。
- --cert： 客户端证书文件路径。
- client.pem为客户端证书。
- --key：客户端私钥文件路径。
- client.key.pem为客户端证书私钥（未加密，建议采用加密密钥）。

请用户根据实际情况对相应参数进行修改。

----结束

2.3 接口调用

2.3.1 使用兼容 TGI 0.9.4 版本的接口

- 文本推理接口：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X POST -d '{
  "inputs": "My name is Olivier and I",
```



```
 "parameters": {
  "decoder_input_details": true,
  "details": true,
  "do_sample": true,
  "max_new_tokens": 20,
  "repetition_penalty": 1.03,
  "return_full_text": false,
  "seed": null,
  "temperature": 0.5,
  "top_k": 10,
  "top_p": 0.95,
  "truncate": null,
  "typical_p": 0.5,
  "watermark": false
 }
}' https://127.0.0.1:1025/generate
```

- 流式推理接口：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X POST -d '{
 "inputs": "My name is Olivier and I",
 "parameters": {
  "details": true,
  "do_sample": true,
  "max_new_tokens": 20,
  "repetition_penalty": 1.03,
  "return_full_text": false,
  "seed": null,
  "temperature": 0.5,
  "top_k": 10,
  "top_p": 0.95,
  "truncate": null,
  "typical_p": 0.5,
  "watermark": false
 }
}' https://127.0.0.1:1025/generate_stream
```

其他接口请参见**兼容TGI 0.9.4版本接口**章节。

2.3.2 使用兼容 vLLM 0.2.6 版本接口

文本/流式推理接口，将请求体中的stream参数改为false即为文本推理，改为true即为流式推理：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X POST -d '{
    "prompt": "My name is Olivier and I",
    "max_tokens": 20,
    "repetition_penalty": 1.03,
    "presence_penalty": 1.2,
    "frequency_penalty": 1.2,
    "temperature": 0.5,
    "top_k": 10,
    "top_p": 0.95,
    "stream": false,
    "ignore_eos": false
}' https://127.0.0.1:1025/generate
```

其他接口请参见**兼容vLLM 0.2.6版本接口**章节。

2.3.3 使用兼容 OpenAI 接口

推理接口，将请求体中的stream参数改为false即为文本推理，改为true即为流式推理：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X POST -d '{
 "model": "gpt-3.5-turbo",
```



```
"messages": [{
 "role": "system",
 "content": "You are a helpful assistant."
}],
"max_tokens": 20,
"presence_penalty": 1.03,
"frequency_penalty": 1.0,
"seed": null,
"temperature": 0.5,
"top_p": 0.95,
"stream": false
}' https://127.0.0.1:1025/v1/chat/completions
```

其他接口请参见**兼容OpenAI接口**章节。

2.3.4 使用兼容 Triton 接口

- Token推理接口：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert
client.pem  --key client.key.pem -X POST -d '{

  "id": "42",
  "inputs": [{
      "name": "input0",
      "shape": [
          1,
          10
      ],
      "datatype": "UINT32",
      "data": [
          396, 319, 13996, 29877, 29901, 29907, 3333, 20718, 316, 23924
      ]
  }],
  "outputs": [{
      "name": "output0"
  }],
  "parameters": {
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "do_sample": true,
      "seed": null,
      "repetition_penalty": 1.03,
      "max_new_tokens": 20,
      "watermark": true
  }
}' https://127.0.0.1:1025/v2/models/llama_65b/infer
```

- 文本推理接口：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert
client.pem  --key client.key.pem -X POST -d '{
 "id":"a123",
 "text_input": "My name is Olivier and I",
 "parameters": {
  "details": true,
  "do_sample": true,
  "max_new_tokens":250,
  "repetition_penalty": 1.1,
  "seed": 123,
  "temperature": 1,
  "top_k": 10,
  "top_p": 0.99,
  "batch_size":100,
  "typical_p": 0.5,
  "watermark": false,
  "perf_stat": false
 }
}' https://127.0.0.1:1025/v2/models/llama_65b/generate
```


- 流式推理接口：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert
client.pem  --key client.key.pem -X POST -d '{
 "id":"a123",
 "text_input": "My name is Olivier and I",
 "parameters": {
  "details": true,
  "do_sample": true,
  "max_new_tokens":200,
  "repetition_penalty": 1.1,
  "seed": 123,
  "temperature": 1,
  "top_k": 10,
  "top_p": 0.99,
  "batch_size":100,
  "typical_p": 0.5,
  "watermark": false,
  "perf_stat": false
 }
}' https://127.0.0.1:1025/v2/models/llama_65b/generate_stream
```

其他接口请参见**兼容Triton接口**章节。

2.3.5 使用 MindIE 原生接口

文本/流式推理接口，将请求体中的stream参数改为false即为文本推理，改为true即为流式推理：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --
key client.key.pem -X POST -d '{
 "inputs": "My name is Olivier and I",
 "stream": true,
 "parameters": {
  "temperature": 0.5,
  "top_k": 10,
  "top_p": 0.95,
  "max_new_tokens": 20,
  "do_sample": true,
  "seed": null,
  "repetition_penalty": 1.03,
  "details": true,
  "typical_p": 0.5,
  "watermark": false
 }
}' https://127.0.0.1:1025/infer
```

其他接口请参见**4.2.6 MindIE原生接口**章节。

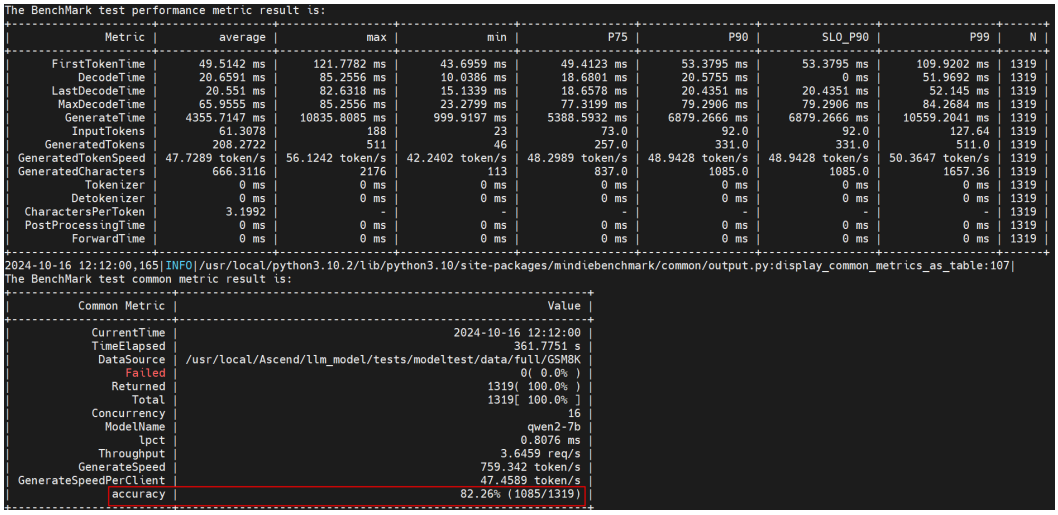
2.4 精度测试

精度测试样例如下所示，参数详细解释请参见**8.1.1.3.1 输入参数**。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型路径}/baichuan2-13b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--MaxOutputLen 512 \
--TestAccuracy True
```

其中--TestAccuracy True 参数是开启精度测试的开关。

测试结果如下图所示：



```
The BenchMark test performance metric result is:
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
|             Metric |        average |            max |            min |            P75 |            P90 |        SLO_P90 |            P99 |    N |
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
|      FirstTokenTime |      49.5142 ms |     121.7782 ms |      43.6959 ms |      49.4123 ms |      53.3795 ms |      53.3795 ms |     109.9202 ms | 1319 |
|         DecodeTime |      20.6591 ms |      85.2556 ms |      10.0386 ms |      18.6801 ms |      20.5755 ms |          0 ms |      51.9692 ms | 1319 |
|     LastDecodeTime |       20.551 ms |      82.6318 ms |      15.1339 ms |      18.6578 ms |      20.4351 ms |      20.4351 ms |       52.145 ms | 1319 |
|      MaxDecodeTime |      65.9555 ms |      85.2556 ms |      23.2799 ms |      77.3199 ms |      79.2906 ms |      79.2906 ms |      84.2684 ms | 1319 |
|       GenerateTime |    4355.7147 ms |   10835.8085 ms |     999.9197 ms |    5388.5932 ms |    6879.2666 ms |    6879.2666 ms |   10559.2041 ms | 1319 |
|        InputTokens |         61.3078 |             188 |              23 |            73.0 |            92.0 |            92.0 |          127.64 | 1319 |
|    GeneratedTokens |        208.2722 |             511 |              46 |           257.0 |           331.0 |           331.0 |           511.0 | 1319 |
| GeneratedTokenSpeed | 47.7289 token/s | 56.1242 token/s | 42.2402 token/s | 48.2989 token/s | 48.9428 token/s | 48.9428 token/s | 50.3647 token/s | 1319 |
| GeneratedCharacters |        666.3116 |            2176 |             113 |           837.0 |          1085.0 |          1085.0 |         1657.36 | 1319 |
|          Tokenizer |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|         Detokenizer |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|  CharactersPerToken |         3.1992 |              - |              - |              - |              - |              - |              - | 1319 |
|  PostProcessingTime |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|        ForwardTime |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
2024-10-16 12:12:00,165|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:107|
The BenchMark test common metric result is:
+----------------------+-----------------------------------------------------------+
|         Common Metric |                                                     Value |
+----------------------+-----------------------------------------------------------+
|           CurrentTime |                                       2024-10-16 12:12:00 |
|         TimeElapsed |                                                 361.7751 s |
|          DataSource | /usr/local/Ascend/llm_model/tests/modeltest/data/full/GSM8K |
|              Failed |                                                 0( 0.0% ) |
|            Returned |                                            1319( 100.0% ) |
|               Total |                                            1319[ 100.0% ] |
|         Concurrency |                                                        16 |
|           ModelName |                                                   qwen2-7b |
|                lpct |                                                0.8076 ms |
|          Throughput |                                              3.6459 req/s |
|       GenerateSpeed |                                             759.342 token/s |
| GenerateSpeedPerClient |                                             47.4589 token/s |
|            accuracy |                                        82.26% (1085/1319) |
+----------------------+-----------------------------------------------------------+
```

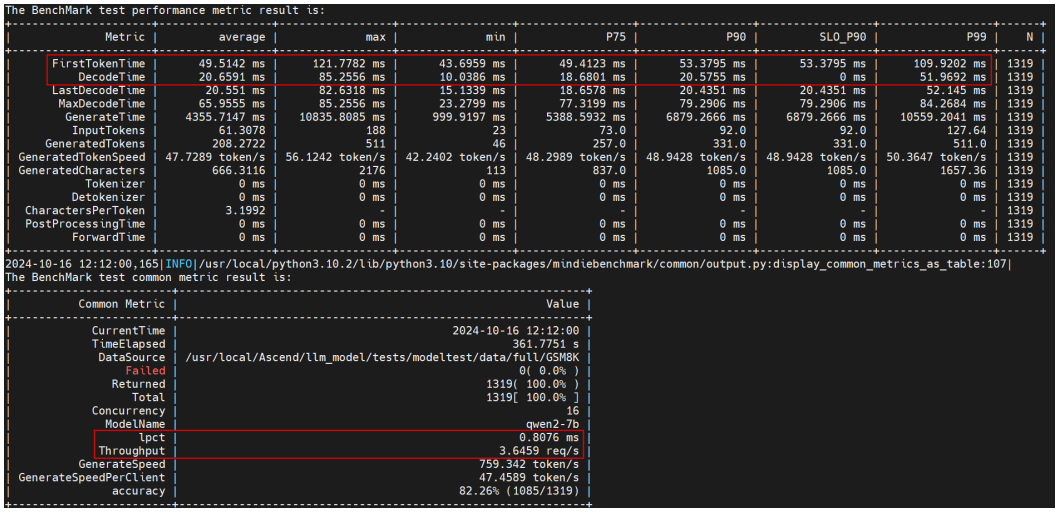
其中accuracy字段为精度测试结果。

2.5 性能测试

性能样例如下所示，参数详细解释请参见**8.1.1.3.1 输入参数**。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型路径}/baichuan2-13b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--MaxOutputLen 512 \
```

结果如下图所示：



```
The BenchMark test performance metric result is:
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
|             Metric |        average |            max |            min |            P75 |            P90 |        SLO_P90 |            P99 |    N |
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
|      FirstTokenTime |      49.5142 ms |     121.7782 ms |      43.6959 ms |      49.4123 ms |      53.3795 ms |      53.3795 ms |     109.9202 ms | 1319 |
|         DecodeTime |      20.6591 ms |      85.2556 ms |      10.0386 ms |      18.6801 ms |      20.5755 ms |          0 ms |      51.9692 ms | 1319 |
|     LastDecodeTime |       20.551 ms |      82.6318 ms |      15.1339 ms |      18.6578 ms |      20.4351 ms |      20.4351 ms |       52.145 ms | 1319 |
|      MaxDecodeTime |      65.9555 ms |      85.2556 ms |      23.2799 ms |      77.3199 ms |      79.2906 ms |      79.2906 ms |      84.2684 ms | 1319 |
|       GenerateTime |    4355.7147 ms |   10835.8085 ms |     999.9197 ms |    5388.5932 ms |    6879.2666 ms |    6879.2666 ms |   10559.2041 ms | 1319 |
|        InputTokens |         61.3078 |             188 |              23 |            73.0 |            92.0 |            92.0 |          127.64 | 1319 |
|    GeneratedTokens |        208.2722 |             511 |              46 |           257.0 |           331.0 |           331.0 |           511.0 | 1319 |
| GeneratedTokenSpeed | 47.7289 token/s | 56.1242 token/s | 42.2402 token/s | 48.2989 token/s | 48.9428 token/s | 48.9428 token/s | 50.3647 token/s | 1319 |
| GeneratedCharacters |        666.3116 |            2176 |             113 |           837.0 |          1085.0 |          1085.0 |         1657.36 | 1319 |
|          Tokenizer |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|         Detokenizer |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|  CharactersPerToken |         3.1992 |              - |              - |              - |              - |              - |              - | 1319 |
|  PostProcessingTime |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
|        ForwardTime |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 1319 |
+--------------------+----------------+----------------+----------------+----------------+----------------+----------------+----------------+------+
2024-10-16 12:12:00,165|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:107|
The BenchMark test common metric result is:
+----------------------+-----------------------------------------------------------+
|         Common Metric |                                                     Value |
+----------------------+-----------------------------------------------------------+
|           CurrentTime |                                       2024-10-16 12:12:00 |
|         TimeElapsed |                                                 361.7751 s |
|          DataSource | /usr/local/Ascend/llm_model/tests/modeltest/data/full/GSM8K |
|              Failed |                                                 0( 0.0% ) |
|            Returned |                                            1319( 100.0% ) |
|               Total |                                            1319[ 100.0% ] |
|         Concurrency |                                                        16 |
|           ModelName |                                                   qwen2-7b |
|                lpct |                                                0.8076 ms |
|          Throughput |                                              3.6459 req/s |
|       GenerateSpeed |                                             759.342 token/s |
| GenerateSpeedPerClient |                                             47.4589 token/s |
|            accuracy |                                        82.26% (1085/1319) |
+----------------------+-----------------------------------------------------------+
```

性能测试结果主要关注FirstTokenTime、DecodeTime等token生成时延的指标和lpct（latency per compelete token，prefill阶段平均每个token时延）、Throughput等测试吞吐量的指标。

2.6 停止服务

使用安装用户登录安装节点，两种停止MindIE Server服务方式如下所示。

- 方式一（推荐）：使用后台进程方式启动服务。
  - 使用kill命令停止进程。
    ```
    kill {mindieservice_daemon 主进程ID}
    ```

    说明

    Linux中查询mindieservice_daemon主进程ID：

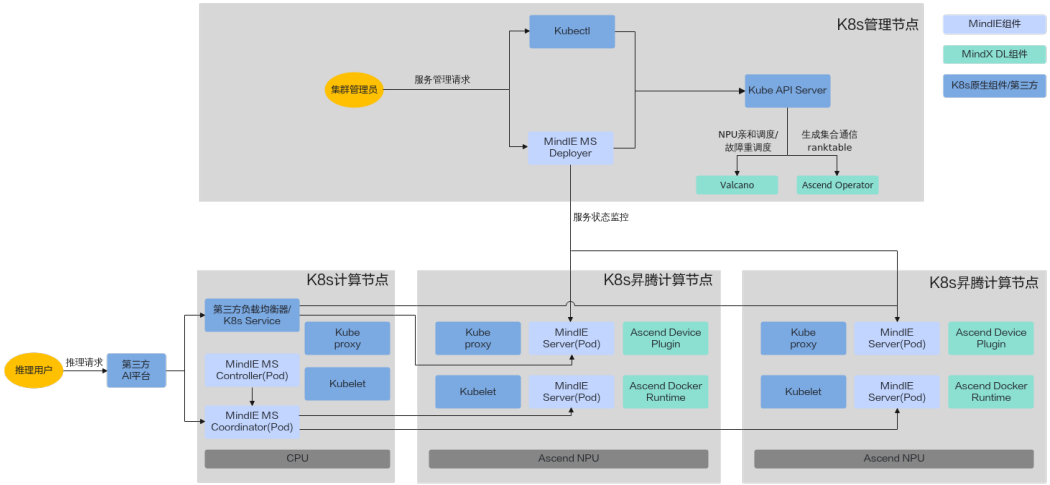
    1. 查看所有与mindieservice相关的进程列表。
       ```
       ps -ef | grep mindieservice
       ```
       回显示例如下：
       ```
       (base) [xxxx@localhost test]# ps -ef | grep 'mindieservice_daemon'
       xxxx     1595969  386706  5 11:37 pts/1    00:00:24 ./bin/mindieservice_daemon
       xxxx     1606004 1595969  0 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606005 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606006 1595969  0 11:42 pts/1    00:00:00 ./bin/mindieservice_daemon
       xxxx     1606007 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606009 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606010 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606012 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1606013 1595969  1 11:42 pts/1    00:00:01 ./bin/mindieservice_daemon
       xxxx     1616310  559909  0 11:44 pts/5    00:00:00 grep --color=auto
       mindieservice_daemon
       ```
    2. 在回显结果中找到PPID列，找出所有包含mindieservice_daemon且PPID相同的进程，这个相同的PPID指向的进程即为mindieservice_daemon主进程，它的PID即为主进程ID，回显示例中的主进程ID为：15955969。
  - 或使用pkill命令停止进程。
    ```
    pkill -9 mindieservice
    ```
- 方式二：以直接启动进程方式启动服务， 可以通过直接按ctrl+c组合键停止服务。

3 集群服务部署

3.1 总述

本章节只适用于基于K8s的集群服务部署，不包含其他场景，其部署示意图如图3-1所示。

图 3-1 K8s 集群整体部署视图



按照MindIE Server推理服务实例在集群计算节点（即推理服务器）上的分布和推理模式，可以分为以下三种部署形态。

表 3-1 部署形态

部署形态	含义
单机服务（非分布式）	单个MindIE Server可以独立作为推理服务实例对外提供推理服务，根据集群计算节点资源情况，整个集群可以支持一个或多个计算节点，单个计算节点可部署一个或多个MindIE Server，单机部署详情请参见**3.4 单机（非分布式）服务部署**。
分布式多机服务	多个MindIE Server跨多个计算节点联合部署，整体作为一个分布式多机推理服务实例对外提供推理服务，适用于模型权重规模较大的场景，分布式多机服务部署详情请参见**3.5 分布式多机服务部署**（当前已支持2个或4个MindIE Server跨机部署）。
PD分离服务	多个MindIE Server在一个或多个计算节点上联合部署，分为P实例（Prefill计算实例）和D实例（Decode计算实例），P实例与D实例分离部署，协同推理，整体作为一个Group对外提供推理服务，PD分离部署详情请参见**3.6 PD分离服务部署**。

3.2 环境准备

3.2.1 软件环境

集群容器化部署依赖Kuberntetes和MindCluster组件，具体部署场景请参考**表3-2**，Kuberntetes组件详细介绍请参见**Kubernetes安装工具**；MindCluster组件详细介绍请参见**MindCluster组件介绍**。

表 3-2 依赖列表

依赖包	软件说明	管理节点是否安装	计算节点是否安装
Kubernetes			
kubectl	Kubernetes的命令行工具。	Y	N
kubeadm	创建和管理Kubernetes集群工具。	Y	Y
kubelet	在集群中的每个节点上用来启动容器。	Y	Y
MindCluster			
Ascend Device Plugin	Kubernetes插件，用于管理和调度昇腾AI处理器设备，提供适合昇腾设备的资源发现和上报策略。需安装Ascend Device Plugin后方可使用。	N	Y

依赖包	软件说明	管理节点是否安装	计算节点是否安装
HCCL-Controller	创建ranktable文件，并按照configmap映射的方式挂载到容器，可以实现多个节点NPU设备之间的数据通信和任务协调，优化集合通信建链性能。	Y	Y
Volcano	基于开源Volcano调度插件机制，增加昇腾AI处理器的亲和性调度、故障重调度等特性，最大化发挥昇腾AI处理器计算性能。	Y	Y
Ascend Docker Runtime	提供docker或containerd的昇腾容器化支持，自动挂载所需文件和设备依赖。	N	Y

3.2.2 K8s 安装与配置

安装 Kubernetes

安装方式一（推荐）：

参考**Kubernetes官网**进行安装。

步骤1 安装Kubernetes的kubectl、kubeadm和kubelet工具。

说明

- 支持Kubernetes的版本为1.16.x~1.25.x，推荐使用1.19.x及以上版本。
- 所有节点都需要安装kubeadm和kubelet，管理节点只需安装kubectl。

步骤2 使用kubeadm工具创建Kubernetes集群，安装kubeadm和创建kubernetes集群请参见Kubernetes官网中的**使用Kubeadm创建集群**章节。

说明

集群初始化过程如遇到问题，请参见《 MindX DL 集群调度安装指南 》的“参考 > FAQ > 安装时出现的故障 > **初始化kubernetes失败**”章节。

----结束

安装方式二：

参考阿里镜像源安装Kubernetes，以下操作均在管理节点进行。当前MindIE仅支持ARM架构，请用户选择架构为ARM的软件版本。

步骤1 获取Kubernetes的kubectl、kubeadm和kubelet软件包，以v1.25版本为例。

1. 单击**阿里Kubernetes镜像官网**。
2. 在阿里Kubernetes镜像官网单击“简介”中“新版下载地址”后方的链接，然后进入/core/stable/v1.25/rpm/aarch64目录获取kubectl、kubeadm和kubelet的安装包。

步骤2 参考阿里Kubernetes镜像官网首页进行安装Kubernetes的kubectl、kubeadm和kubelet工具。

说明

- 如果出现“No match for argument: socat”或者“nothing provides socat needed by xxx”等回显信息，表示环境缺少socat库，解决方式如下所示。（其他库缺失也会有同样的回显，比如eatables、conntrak等）

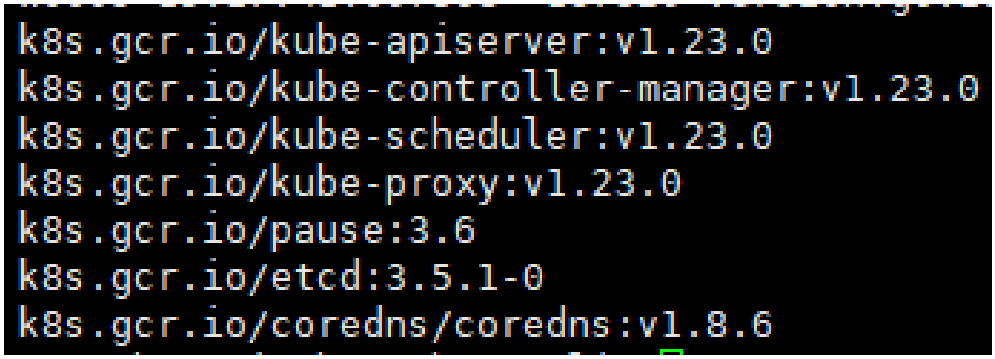
  使用以下命令自行安装缺失的库。

  ```
  #以CentOS系统为例
  yum install -y socat
  #以Ubuntu系统为例
  apt-get install -y socat
  ```

步骤3 执行以下命令查询部署Kubernetes需要的依赖和镜像，如图3-2所示。

```
kubeadm config images list
```

图 3-2 所需依赖及镜像查询结果



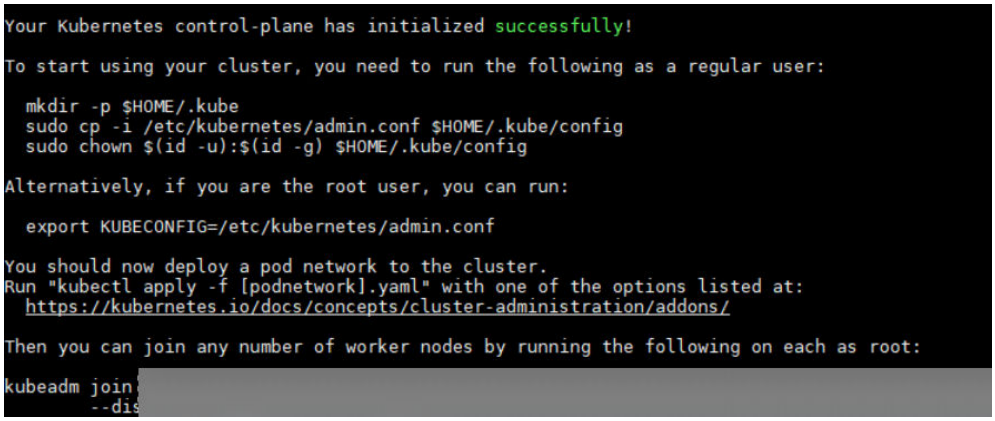
根据查询结果，用户需自行通过docker pull的方式依次进行安装，示例命令如下所示。

```
docker pull k8s.gcr.io/kube-apiserver:v1.23.0
```

步骤4 使用以下命令初始化Kubernetes集群，当出现如图3-3所示回显时，表示初始化成功。

```
kubeadm init
```

图 3-3 Kubernetes 集群初始化成功



然后执行图3-3中的内容，如下所示：

```
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
```

步骤5 执行以下命令查看当前默认启动项状态是否正常，如图3-4所示，状态为Running即为正常。

```
kubectl get pods -A
```

图 3-4 查看状态

```
kube-system   coredns-64897985d-pqbxx                1/1     Running   3 (29m ago)   3h40m
kube-system   coredns-64897985d-xqm4k                1/1     Running   3 (29m ago)   3h40m
kube-system   etcd-huawei                            1/1     Running   3 (29m ago)   3h40m
kube-system   kube-apiserver-huawei                  1/1     Running   5 (29m ago)   3h40m
kube-system   kube-controller-manager-huawei         1/1     Running   3 (29m ago)   3h40m
kube-system   kube-proxy-649lh                       1/1     Running   3 (29m ago)   3h40m
kube-system   kube-scheduler-huawei                  1/1     Running   3 (29m ago)   3h40m
```

----结束

重置 Kubernetes 设置

执行以下命令可以重置Kubernetes设置，回显如图3-5所示则表示重置成功。

```
kubeadm reset
```

图 3-5 重置成功

```
(base)                    $kubeadm reset
[reset] Reading configuration from the cluster...
[reset] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[reset] WARNING: Changes made to this host by 'kubeadm init' or 'kubeadm join' will be reverted.
[reset] Are you sure you want to proceed? [y/N]: y
[preflight] Running pre-flight checks
[reset] Removing info for node           from the ConfigMap "kubeadm-config" in the "kube-system" Namespace
[reset] Stopping the kubelet service
[reset] Unmounting mounted directories in "/var/lib/kubelet"
[reset] Deleting contents of config directories: [/etc/kubernetes/manifests /etc/kubernetes/pki]
[reset] Deleting files: [/etc/kubernetes/admin.conf /etc/kubernetes/kubelet.conf /etc/kubernetes/bootstrap-kube
[reset] Deleting contents of stateful directories: [/var/lib/etcd /var/lib/kubelet /var/lib/dockershim /var/run

The reset process does not clean CNI configuration. To do so, you must remove /etc/cni/net.d

The reset process does not reset or clean up iptables rules or IPVS tables.
If you wish to reset iptables, you must do so manually by using the "iptables" command.

If your cluster was setup to utilize IPVS, run ipvsadm --clear (or similar)
to reset your system's IPVS tables.

The reset process does not clean your kubeconfig files and you must remove them manually.
Please, check the contents of the $HOME/.kube/config file.
```

说明

重置成功后，用户需手动删除*{$HOME}*/.kube/config文件，确保Kubernetes的配置全部删除。

Kubernetes 集群增加计算节点

整个集群仅使用一台服务器的情况下，用户无需新增计算节点，可略过下面步骤

待新增节点需满足以下要求：

已安装Kubernetes基本软件kubeadm和kubelet。

步骤1 在管理节点上创建一个新增节点加入集群所需的token和ca-cert码。

token和ca-cert码的有效期为24小时，如果已过期，请使用以下命令创建。

- 创建token
  ```
  kubeadm token create
  ```


- 创建ca-cert码
  ```
  openssl x509 -pubkey -in /etc/kubernetes/pki/ca.crt | openssl rsa -pubin -outform der 2>/dev/null | openssl dgst -sha256 -hex | sed 's/^.* //'
  ```

  说明

  - 由于上述命令包含明文token，执行后可通过历史操作查询方式显示，导致敏感信息暴露，建议用户采用以下方式进行设置。
    - 在执行敏感命令前执行以下命令临时禁用历史操作查询功能。
      ```
      set +o history
      ```
    - 完成敏感命令操作后执行以下命令恢复历史查询功能。
      ```
      set -o history
      ```

步骤2 在新节点上执行以下命令加入集群。

```
kubeadm join ip:port --token {token} \
    -discovery-token-ca-cert-hash sha256:{ca-cert码}
```

参数解释：

- ip:port：管理节点上Kubernetes的IP和端口。
- --token：节点加入的令牌。
- -discovery-token-ca-cert-hash：加入集群的证书哈希值。

步骤3 在新节点上使用以下命令查询当前节点主机名称。

```
hostname
```

如节点主机名和集群中其他节点名称冲突，修改/etc/hostname文件更改节点的主机名。

步骤4 在管理节点上使用以下命令kubectl get nodes -A查看节点信息，如图3-6所示，localhost.localdomain即为新增节点。

图 3-6 新增节点

```
NaMe                  STATUS  ROLEs   AGE     VERSION
localhost.localdomain Ready   worker  78d     v1.19.16
server01              Ready   master  78d     v1.19.16
```

步骤5 在管理节点上使用以下命令根据实际的NPU设备类型为新增节点打上accelerator=huawei-Ascend910或者accelerator=huawei-Ascend310x标签。

```
#kubectl label nodes {节点名称} accelerator=huawei-Ascend910
kubectl label nodes localhost.localdomain accelerator=huawei-Ascend910
```

步骤6 在管理节点上使用以下命令查看为新增节点附的accelerator=huawei-Ascend910标签，有“accelerator=huawei-Ascend910”则表示成功。

```
kubectl get nodes --show-labels
```

----结束

3.2.3 MindCluster 组件安装

MindIE MS依赖MindCluster中的Ascend Docker Runtime、Ascend Device Plugin、Volcano和HCCL-Controller组件。其中Volcano和HCCL-Controller组件在管理节点安装，其他组件在计算节点上安装。

步骤1 请参考《 MindCluser 集群调度安装指南 》的“安装 > 组件安装 > 安装部署 > 手动安装 > **安装前准备**”章节完成创建节点标签、创建用户、创建日志目录和创建命名空间。

步骤2 请参考《MindX DL 集群调度安装指南》的“安装 > 组件安装 > 安装部署 > 手动安装 > **Ascend Docker Runtime**”章节中的“Containerd场景下安装Ascend Docker Runtime”安装Ascend Docker Runtime。

步骤3 请参考《MindX DL 集群调度安装指南》的“安装 > 组件安装 > 安装部署 > 手动安装 > **Ascend Device Plugin**”章节安装Ascend Device Plugin，使用device-plugin-*xxx*-v*{version}*.yaml文件进行安装。

说明

当Ascend Device Plugin启动时，*xxx*.yaml配置文件中参数useAscendDocker参数配置为true且用户已安装Ascend Docker Runtime并生效，会自动挂载在“/usr/local/Ascend”下驱动相关目录。

步骤4 请参考《MindX DL 集群调度安装指南》的“安装 > 组件安装 > 安装部署 > 手动安装 > **Volcano**”章节安装Volcano。

说明

- 请使用v1.7.0版本的volcano进行安装。
- 在单机场景下，参考《MindX DL 集群调度安装指南》的“安装 > 组件安装 > 安装部署 > 手动安装 > **Volcano**”章节安装Volcano时，在执行“Volcano”章节中的步骤9前，需要修改Volcano解压后生成的volcano-v1.7.0目录下的volcano-v1.7.0.yaml文件，搜索“useClusterInfoManager”字段并将该值改为“false”，如下图所示，修改完成后，再执行“Volcano”章节中的步骤9。

```
    enableNodeOrder: false
configurations:
  - name: init-params
    arguments: {"grace-over-time":"900","presetVirtualDevice":"true","nslb-version":"1.0","shared-tor-num":"2",
"useClusterInfoManager":"false","super-pod-size": "48", "reserve-nodes": "2"}
```

步骤5 请参考《MindCluser 集群调度安装指南》的“安装 > 组件安装 > 安装部署 > 手动安装 > **HCCL Controller**”章节安装HCCL-Controller。

说明

分布式多机部署和PD分离部署场景时该组件必装。

----结束

3.3 准备 MindIE Server 镜像

请参见《MindIE安装指南》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”制作MindIE Server镜像。

3.4 单机（非分布式）服务部署

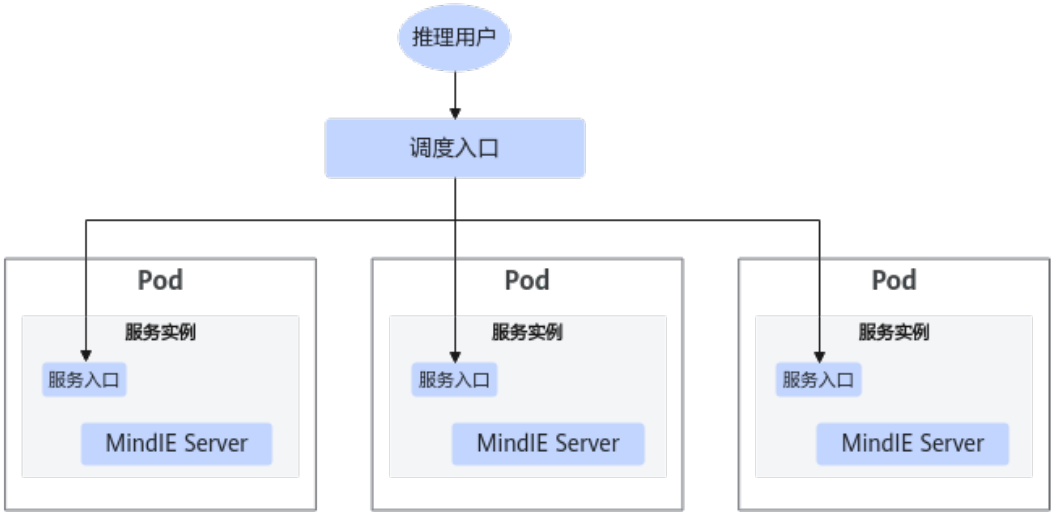
3.4.1 场景介绍

单机服务部署为非分布式实例部署的场景，即在一个计算节点内可部署一个完整独立的MindIE Server推理服务实例。根据设备资源情况，同一个计算节点可部署多个Server服务实例，也支持在多个计算节点上部署多个服务实例。 根据服务是否集成MindIE MS Coordinator负载均衡器， 可分为以下两种部署方案。

MindIE Server 作为对外服务入口

推理请求：通过第三方平台的调度入口（由用户部署平台而定，比如K8s的调度入口或MA的调度入口等），基于特定的调度算法，直接调度请求发送给各个单机版的MindIE Server实例。具体部署详情请参见**3.4.2.2 使用Deployer部署服务示例**章节。使用该部署方式部署单机（非分布式）服务时，其支持的接口请参见**4 服务化接口**。

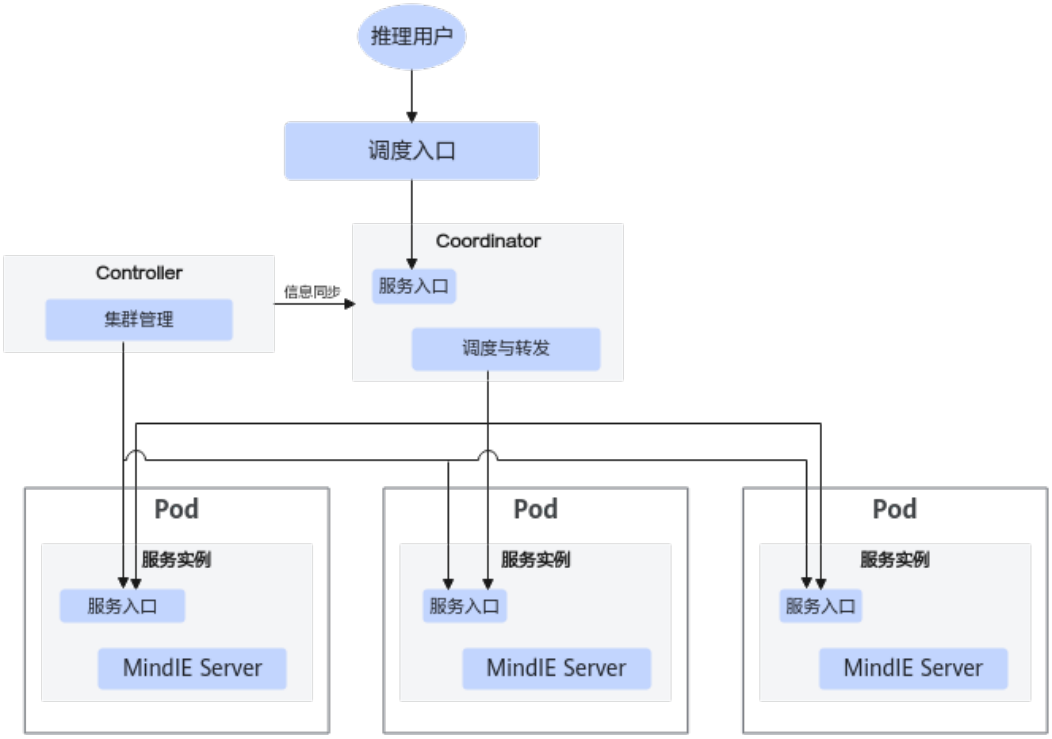
图 3-7 MindIE Server 作为对外服务入口



MindIE MS Coordinator 作为对外服务入口

推理请求：通过第三方平台的调度入口（由用户部署平台而定，比如k8s的调度入口或MA的调度入口等），将所有请求发送给Coordinator，Coordinator基于本身支持的负载调度算法，调度请求发送给各个单机版的MindIE Server实例。具体部署详情请参见**3.4.2.3 使用kubectl部署服务示例**章节。使用该部署方式部署单机（非分布式）服务时，其支持的接口请参见**8.3.4.6 RESTful接口API**。

图 3-8 MindIE MS Coordinator 作为对外服务入口



单机部署场景支持的调度算法如下所示：

调度算法	含义	部署建议
cache_affinity	Cache亲和调度算法：当前只支持OpenAI多轮会话场景的调度亲和。	OpenAI多轮会话场景，推荐配置。
round_robin	轮询调度算法。	非OpenAI多轮会话场景，推荐配置。

3.4.2 安装部署

3.4.2.1 前提条件

已参照**3.2.2 K8s安装与配置**、**3.2.3 MindCluster组件安装**和**3.3 准备MindIE Server镜像**完成K8s的安装配置、MindCluster组件的安装和MindIE Server镜像制作。

3.4.2.2 使用 Deployer 部署服务示例

3.4.2.2.1 部署 Deployer 服务端

实现说明

Deployer部署器提供MindIE Server推理服务在K8s集群场景下的部署管理能力，具体能力请参见**8.3.2.1 功能介绍**。 Deployer包含一个服务端组件ms_server和一个客户端组件msctl，以下内容将介绍Deployer服务端的部署流程。

准备 TLS 证书

MindIE MS当前业务定位是Kubernetes集群内组件，且仅集群管理员有使用权限，相关的TLS通信认证证书采用集群CA签发的证书。

证书分类

主要需要准备以下四套证书的CA证书、服务证书、私钥、KMC加密口令和KMC工作密钥。

- MindIE MS服务端证书
- MindIE MS服务端与kube API-Server通信的证书
- MindIE MS客户端msctl与MindIE MS服务端通信的证书
- MindIE MS服务端与MindIE Server通信的客户端证书

由于Kubernetes集群根CA证书（通常位于/etc/kubernetes/pki/ca.crt目录）不支持CRL吊销，为了保证MindIE MS相关组件在集群管理面的通信安全：

1. 首先使用kubernetes集群根CA签发且具备CRL吊销能力（包含CRL Sign签名）的**中间CA**；中间CA的签发方法请参见**签发中间CA**。
2. 使用该**中间CA**签发MindIE MS服务端证书、MindIE MS服务端与kube API-Server通信的客户端证书、MindIE MS客户端msctl与MindIE MS服务端通信证书；其签发方法请参见**签发其他证书**。
3. MindIE MS服务端与MindIE Server通信的客户端证书，由业务面的CA签发，需要用户自行准备。

说明

MindIE MS不支持证书有效期检查和更新机制，用户需要自行检测证书有效期，并对证书进行更新，避免业务中断。可参考**8.1.2 CertTools**章节，使用证书管理工具对证书进行有效期检查和更新。

签发中间 CA

当前要求签发中间CA的subject name中CN=mindiems, O=msgroup；这是MindIE MS服务端唯一认可的中间CA名，且该中间CA由集群根CA签发，管理员确保只用于MindIE MS组件的证书签发。

签发中间CA对于路径没有要求，以下步骤以**/home/*{用户名称}*/cas**目录为例进行操作。

步骤1 执行以下命令进入到**/home/*{用户名称}*/cas**目录。

```
cd /home/{用户名称}/cas
```

步骤2 执行以下命令创建PKI私钥。

```
export CERT_CN=mindiems
export CA_CERT=/etc/kubernetes/pki/ca.crt
export CA_KEY=/etc/kubernetes/pki/ca.key
mkdir "$CERT_CN"
openssl genrsa -aes256 -out  "$CERT_CN"/cert.key.pem 4096
```


参数解释：

- CERT_CN：MindIE MS服务端唯一认可的中间CA名，取值：mindiems；
- CA_CERT：Kubernetes集群根CA证书路径，取值：/etc/kubernetes/pki/ca.crt；
- CA_KEY：Kubernetes集群根CA私钥路径，取值：/etc/kubernetes/pki/ca.key。

根据回显输入私钥口令，然后按回车键。

```
Enter pass phrase for mindiems/cert.key.pem:
Verifying - Enter pass phrase for mindiems/cert.key.pem:
```

出于安全考虑，以及后续导入证书的要求，用户输入的私钥口令的复杂度必须符合以下要求：

- 口令长度至少8个字符；
- 口令必须包含如下至少两种字符的组合：
  - 至少一个小写字母；
  - 至少一个大写字母；
  - 至少一个数字；
  - 至少一个特殊字符。

步骤3 执行以下命令赋予cert.key.pem私钥文件可读权限。

```
chmod 400 "$CERT_CN"/cert.key.pem
```

执行以下命令检查"$CERT_CN"目录下是否存在cert.key.pem私钥文件，并查看私钥内容。

```
openssl rsa -in "$CERT_CN"/cert.key.pem
```

根据回显输入**步骤2**设置的私钥口令，然后按回车键，当显示私钥内容时表示cert.key.pem私钥文件生成成功。

步骤4 执行以下命令创建CSR文件，根据回显输入**步骤2**设置的私钥口令，然后按回车键。

```
openssl req -new -key  "$CERT_CN"/cert.key.pem -out "$CERT_CN"/cert.csr -subj "/CN="$CERT_CN"/
O=msgroup"
```

步骤5 执行以下命令赋予cert.csr文件可读可写权限。

```
chmod 600 "$CERT_CN"/cert.csr
```

执行以下命令检查"$CERT_CN"目录下是否存在cert.csr文件，当显示cert.csr文件内容表示cert.csr文件生成成功。

```
openssl req -in "$CERT_CN"/cert.csr -noout -text
```

步骤6 执行以下命令创建证书拓展配置文件"$CERT_CN"/cert.conf。

```
vi "$CERT_CN"/cert.conf
```

并在cert.conf文件中添加如下内容：

```
[ req ]
distinguished_name     = req_distinguished_name
prompt                 = no

[ v3_req ]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign
```

步骤7 执行以下命令签发证书（cert.pem），并赋予cert.pem文件可读权限。

```
openssl x509 -req -days 365 -sha256 -extensions v3_req -CA "$CA_CERT" -CAkey "$CA_KEY"  -
CAcreateserial -in "$CERT_CN"/cert.csr -out "$CERT_CN"/cert.pem -extfile "$CERT_CN"/cert.conf
chmod 400 "$CERT_CN"/cert.pem
```


执行以下命令检查“$CERT_CN”目录下是否存在cert.pem文件，当回显中有subject name: CN=mindiems, O=msgroup信息则证书签发成功。

```
openssl x509 -in "$CERT_CN"/cert.pem -noout -text
```

签发完成后，在**/home/*{用户名称}*/cas**目录下将生成一个/mindiems目录，/mindiems目录中包含的文件为：cert.conf、cert.csr、cert.key.pem和cert.pem。

----结束

签发其他证书

签发MindIE MS服务端证书、MindIE MS服务端与kube API-Server通信的客户端证书、MindIE MS客户端msctl与MindIE MS服务端通信证书对于路径没有要求，且签发方法相同。

以下步骤以**/home/*{用户名称}*/cas**目录为例签发MindIE MS服务端证书进行操作。

步骤1 执行以下命令进入到**/home/*{用户名称}*/cas**目录。

```
cd /home/{用户名称}/cas
```

步骤2 执行以下命令创建PKI私钥。

```
export CERT_CN=msserver
export CA_CERT=/home/{用户名称}/cas/mindiems/cert.pem
export CA_KEY=/home/{用户名称}/cas/mindiems/cert.key.pem
mkdir "$CERT_CN"
openssl genrsa -aes256 -out  "$CERT_CN"/cert.key.pem 4096
```

参数解释：

- CERT_CN：证书类型由该参数决定，取值如下：
  - MindIE MS服务端证书，取值为：msserver；
  - MindIE MS服务端与kube API-Server通信的客户端证书，取值为：kubeclient；
  - MindIE MS客户端msctl与MindIE MS服务端通信的证书，取值为：msclientuser。
- CA_CERT：**签发中间CA**的中间CA证书路径，取值：/home/*{用户名称}*/cas/mindiems/cert.pem；
- CA_KEY：**签发中间CA**的中间CA私钥路径，取值：/home/*{用户名称}*/cas/mindiems/cert.key.pem。

根据回显输入私钥口令，然后按回车键。

```
Enter pass phrase for msserver/cert.key.pem:
Verifying - Enter pass phrase for msserver/cert.key.pem:
```

出于安全考虑，以及后续导入证书的要求，用户输入的私钥口令的复杂度必须符合以下要求：

- 口令长度至少8个字符；
- 口令必须包含如下至少两种字符的组合：
  - 至少一个小写字母；
  - 至少一个大写字母；
  - 至少一个数字；
  - 至少一个特殊字符。

步骤3 执行以下命令赋予私钥文件（cert.key.pem）可读权限。

```
chmod 400 "$CERT_CN"/cert.key.pem
```

执行以下命令检查"$CERT_CN"目录下是否存在私钥文件（cert.key.pem），并查看私钥内容。

```
openssl rsa -in "$CERT_CN"/cert.key.pem
```

根据回显输入**步骤2**设置的私钥口令，然后按回车键，当显示私钥内容时表示私钥文件（cert.key.pem）生成成功。

步骤4 执行以下命令创建CSR文件，根据回显输入**步骤2**设置的私钥口令，然后按回车键。

```
openssl req -new -key  "$CERT_CN"/cert.key.pem -out "$CERT_CN"/cert.csr -subj "/CN="$CERT_CN"/
O=msgroup"
```

步骤5 执行以下命令赋予cert.csr文件可读可写权限。

```
chmod 600 "$CERT_CN"/cert.csr
```

执行以下命令检查"$CERT_CN"目录下是否存在cert.csr文件，当显示cert.csr文件内容表示cert.csr文件生成成功。

```
openssl req -in "$CERT_CN"/cert.csr -noout -text
```

步骤6 执行以下命令创建证书拓展配置文件"$CERT_CN"/cert.conf。

```
vi "$CERT_CN"/cert.conf
```

并在cert.conf文件中添加如下内容：

```
[ req ]
distinguished_name     = req_distinguished_name
prompt                 = no

[ v3_req ]
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:false
keyUsage = critical, digitalSignature, keyCertSign
```

步骤7 执行以下命令签发MindIE MS服务端证书（cert.pem），并赋予cert.pem文件可读权限。

```
openssl x509 -req -days 365 -sha256 -extensions v3_req -CA "$CA_CERT" -CAkey "$CA_KEY"  -
CAcreateserial -in "$CERT_CN"/cert.csr -out "$CERT_CN"/cert.pem -extfile "$CERT_CN"/cert.conf
chmod 400 "$CERT_CN"/cert.pem
```

执行以下命令检查“$CERT_CN”目录下是否存在cert.pem文件，当回显中有subject name: CN=msserver, O=msgroup信息则证书签发成功。

```
openssl x509 -in "$CERT_CN"/cert.pem -noout -text
```

签发完成后，在**/home/*{用户名称}*/cas**目录下将生成一个/msserver目录，/msserver目录中包含的文件为：cert.conf、cert.csr、cert.key.pem和cert.pem。

----结束

镜像制作

MindIE MS服务端可以采用镜像方式部署在Kubernetes的Deployment管理的容器中，以下操作步骤指导用户如何完成MindIE MS服务端镜像的制作。

须知

不建议用户直接在物理机上部署MindIE MS服务端，通过Kubernetes集群集成部署的方式可实现进程故障恢复，增强可靠性。

操作步骤

以下操作对路径没有要求，用户自行选择路径进行操作，这里以/home/*{用户名称}*/package路径为例。

步骤1 单击**链接**获取MindIE软件包（Ascend-mindie_*{version}*_linux-aarch64.run）并上传至/home/*{用户名称}*/package目录。

1. 使用以下命令进入/home/*{用户名称}*/package目录。
   ```
   cd /home/{用户名称}/package
   ```
2. 使用以下命令增加对MindIE软件包的可执行权限。
   ```
   chmod +x Ascend-mindie_{version}_linux-aarch64.run
   ```
3. 使用以下命令解压MindIE软件包到当前路径下的mindie目录，获取mindie-service的run安装包。
   ```
   ./Ascend-mindie_{version}_linux-aarch64.run --extract=mindie
   ```
4. 使用以下命令安装Ascend-mindie-service_*{version}*_linux-aarch64.run到当前目录。
   ```
   ./mindie/Ascend-mindie-service_{version}_linux-aarch64.run --install-path=$PWD
   ```

步骤2 配置ms_server.json文件，更多参数解释请参见**8.3.2.2 配置说明** 。

使用以下命令打开ms_server.json文件。

```
vi mindie-service/latest/conf/ms_server.json
```

配置以下参数：

k8s_apiserver_ip：配置为Kubernetes管理节点的物理机IP地址。

步骤3 使用以下命令拉取ubuntu基础镜像。

```
docker pull ubuntu:22.04
```

步骤4 执行以下命令检查ubuntu镜像，查询结果如**图3-9**所示。

```
docker images | grep ubuntu
```

图 3-9 镜像查询结果

步骤5 使用MindIE证书管理工具导入 **准备TLS证书**中所有的证书和私钥，并生成KMC加密口令文件。MindIE证书导入工具的详细用法请参见**8.1.2 CertTools**。

```
cd mindie-service/latest
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$PWD/lib
mkdir -p -m 700 security/msserver
mkdir -p -m 700 security/mindieclient
mkdir -p -m 700 ./security/kubeclient
cp -r ./bin ./security/kubeclient
mkdir -p -m 700 ./security/kubeclient/security/ca
mkdir -p -m 700 ./security/kubeclient/security/certs
mkdir -p -m 700 ./security/kubeclient/security/keys
mkdir -p -m 700 ./security/kubeclient/security/pass
cp -r ./security/kubeclient/*  ./security/msserver
cp -r ./security/kubeclient/*  ./security/mindieclient

# 可选，为保证通信安全，建议选择。导入MS Deployer服务端与MindIE Server通信的证书
python3 ./scripts/config_mindie_server_tls_cert.py ./security/mindieclient import_ca  {CA文件路径}
python3 ./scripts/config_mindie_server_tls_cert.py ./security/mindieclient import_cert  {证书文件路径} {加密私钥文件路径}

#必须，导入MS Deployer服务端证书
export KUBE_CA_CERT_PATH=/etc/kubernetes/pki/ca.crt       #Kubernetes根CA证书文件路径
cp $KUBE_CA_CERT_PATH ./security/msserver/security/ca/ca.pem
```



```
chmod 400 ./security/msserver/security/ca/ca.pem
mkdir -p msserver_chain
cat /home/{用户名称}/cas/msserver/cert.pem /home/{用户名称}/cas/mindiems/cert.pem > ./msserver_chain/
cert.pem     #/home/{用户名称}/cas/msserver/cert.pem：MindIE MS服务端证书文件路径；/home/{用户名
称}/cas/mindiems/cert.pem：中间CA证书文件路径
chmod 400 ./msserver_chain/cert.pem
python3 ./scripts/config_mindie_server_tls_cert.py ./security/msserver import_cert  ./msserver_chain/
cert.pem /home/{用户名称}/cas/msserver/cert.key.pem   #/home/{用户名称}/cas/msserver/cert.key.pem：
MindIE MS服务端私钥文件路径
rm -rf msserver_chain

# 可选，为保证通信安全，建议选择。导入MS Deployer服务端与kube API-Server通信的证书
cp $KUBE_CA_CERT_PATH ./security/kubeclient/security/ca/ca.pem
chmod 400 ./security/kubeclient/security/ca/ca.pem
mkdir -p kubeclient_chian
cat /home/{用户名称}/cas/kubeclient/cert.pem /home/{用户名称}/cas/mindiems/cert.pem > ./
kubeclient_chian/cert.pem   #/home/{用户名称}/cas/kubeclient/cert.pem：MindIE MS服务端与kube API-
Server通信的客户端证书文件路径；/home/{用户名称}/cas/mindiems/cert.pem：中间CA证书文件路径
chmod 400 ./kubeclient_chian/cert.pem
python3 ./scripts/config_mindie_server_tls_cert.py ./security/kubeclient import_cert  ./kubeclient_chian/
cert.pem /home/{用户名称}/cas/kubeclient/cert.key.pem   #/home/cas/kubeclient/cert.key.pem：MindIE MS
服务端与kube API-Server通信的客户端私钥文件路径
rm -rf kubeclient_chian
cd ../..
```

步骤6 在当前路径下编写Dockerfile文件，用于MindIE MS服务端镜像的制作，样例如下所示（格式必须和样例保持一致）。

说明

以下为制作非root用户权限的MindIE MS服务端镜像，出于安全考虑，不建议用户制作root用户权限的MindIE MS服务端镜像。

```
FROM ubuntu:22.04
ARG USER_GROUP={用户群组}
ARG USER={用户名}
ARG UID={用户ID}
ARG USER_HOME_DIR=/home/${USER}

RUN useradd -d ${USER_HOME_DIR} -u ${UID} -m -s /usr/sbin/nologin ${USER} &&\
    usermod root -s /usr/sbin/nologin

COPY ./mindie-service  /home/${USER}/mindie-service
RUN rm -rf /home/${USER}/mindie-service/latest/security
ARG MIES_INSTALL_PATH=/home/${USER}/mindie-service/latest

ENV LD_LIBRARY_PATH=${MIES_INSTALL_PATH}/lib:$LD_LIBRARY_PATH
ENV HSECEASY_PATH=${MIES_INSTALL_PATH}/lib

RUN chown -R ${USER}:${USER_GROUP} ${MIES_INSTALL_PATH}
USER ${USER}
```

参数解释：

- USER_GROUP：自定义MindIE MS的专用用户组名称，和USER保持一致；可使用cat /etc/passwd命令查看，回显中第四列即为该参数的值。
- USER：自定义MindIE MS的专用用户名称。
- UID：自定义用户UID，不要和已有用户ID冲突，且不能是rootID或者0；可使用cat /etc/passwd命令查看，回显中第三列即为该参数的值。
- USER_HOME_DIR：用户的home目录。
- chmod权限：500用于可执行文件，400用于so动态库，440用于json配置文件，750用于文件夹权限。

步骤7 修改证书相关文件属主为容器内用户群组，其中UID为上述Dockerfile中配置的UID

```
chown -R ${UID}:${UID} ./mindie-service/latest/security
```


步骤8 执行以下命令开始镜像制作，执行完成后回显如图3-10所示即制作成功。

```
docker build --no-cache -t {镜像名称}:{镜像版本} ./
```

说明

- 请确保宿主机上的{USER_GROUP}：{UID}用户是有权限使用kubernetes和MindIE MS的管理员用户，上述证书修改权限后将对该用户开放读取权限。
- *{镜像名称}:{镜像版本}*需要用户自定义。

图 3-10 镜像制作成功

----结束

K8s 集群部署 Deployer 服务端

本章节介绍如何通过kubectl创建Deployment， 实现Deployer服务端的部署。

操作步骤

步骤1 使用以下命令创建命名空间namespace。

```
kubectl create ns mindie
```

步骤2 创建ClusterRole和ClusterRoleBinding。

ClusterRole和ClusterRoleBinding用于授予Deployer服务端访问Kube API-Server的权限，用户需自行准备以下yaml文件（比如：ms-role.yaml）。

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: ms-role
rules:
- apiGroups: ["apps"]
  resources: ["deployments"]
  verbs: ["create", "delete", "get", "patch"]
- apiGroups: [""]
  resources: ["pods"]
  verbs: ["get", "list", "patch"]
- apiGroups: [""]
  resources: ["configmaps"]
  verbs: ["get", "create", "delete"]
- apiGroups: [""]
  resources: ["services"]
  verbs: ["get", "delete", "create"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: ms-role-bind
subjects:
- kind: User
  name: kubeclient # 集群用户名，该值为MindIE MS与Kube API-Server通信的客户端证书的CN字段值。
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: ms-role
  apiGroup: rbac.authorization.k8s.io
```


表 3-3 重要配置参数说明

参数	说明
rules	必填。 resources：Kubernetes资源，MindIE MS业务过程需要具备以下resources的操作权限。 • deployments：create、delete和get权限。 • pods：get、list和patch权限。 • configmaps：get、create和delete权限。 • services：get、create和delete权限。
ClusterRoleBinding中subjects参数的name	必填。 Kubernetes集群的用户名称，需要用户自定义。当创建该用户后，Kube API-Server将授予该用户rules参数中指定的权限。 Deployer服务端作为客户端向Kube API-Server发送HTTPS请求，Kube API-Server会检验客户端证书的CN字段，如果与该name一致，将授予上述操作权限。所以用户在使用Deployer服务端之前，需要先准备由集群中间CA签发的客户端证书，详情请参见**签发其他证书**。

使用以下命令创建Role资源。

```
kubectl apply -f ms-role.yaml
```

如显示以下回显则表示Role创建成功。

```
clusterrole.rbac.authorization.k8s.io/ms-role created
clusterrolebinding.rbac.authorization.k8s.io/ms-role-bind created
```

步骤3 创建Deployment。

1. 配置Deployment。

   用户需自行编写与Kubernetes交互的yaml文件（例：ms_server_deployment.yaml），样例如下所示（格式必须和样例保持一致），部分参数请参考**表3-4**自行修改。

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: ms-server
  namespace: mindie
  labels:
    app: ms-server
spec:
  replicas: 1
  selector:
    matchLabels:
      app: ms-server
  template:
    metadata:
      labels:
        app: ms-server
    spec:
      terminationGracePeriodSeconds: 5
      automountServiceAccountToken: false
      nodeSelector:
```



```
      masterselector: dls-master-node     #MindIE MS服务端需部署在Kubernetes管理节点上，节点标签必须选择为masterselector: dls-master-node
    containers:
      - image: {镜像名称}:{镜像版本}     #内容和用户自定义的镜像名称和镜像版本一致
        imagePullPolicy: IfNotPresent
        name: mindie-ms
        securityContext:
          allowPrivilegeEscalation: false
          readOnlyRootFilesystem: true
          runAsUser: {UID}     #容器用户ID，和Dockerfile的UID一致
          runAsGroup: {UID}    #容器所属组ID，和Dockerfile的UID一致。
          capabilities:
            drop: ["ALL"]
          seccompProfile:
            type: RuntimeDefault
        volumeMounts:
        - name: dir1
          mountPath: /var/log/mindie-ms
        - name: dir2                                    #容器内业务持久化数据文件路径
          mountPath: /var/lib/mindie-ms
        - name: security                                #容器内证书文件目录
          mountPath: /home/{容器内用户名}/mindie-service/latest/security
        command: ["bin/bash", "-c", "cd /home/{容器内用户名}/mindie-service/latest && ./bin/ms_server ./conf/ms_server.json"] #容器内用户名即为自定义MindIE MS的专用用户名称。
        env:
        - name: HSECEASY_PATH
          value: /home/{容器内用户名}/mindie-service/latest/lib #容器内用户名即为自定义MindIE MS的专用用户名称。
        - name: MINDIE_MS_SERVER_IP
          valueFrom:
            fieldRef:
              fieldPath: status.podIP
        resources:
          requests:
            memory: "512Mi"
            cpu: "4000m"
          limits:
            memory: "1024Mi"
            cpu: "4000m"
    volumes:
    - name: dir1
      hostPath:
        path: /var/log/mindie-ms
        type: Directory
    - name: dir2
      hostPath:
        path: /var/lib/mindie-ms
        type: Directory
    - name: security
      hostPath:
        path: /{用户工作目录}/mindie-service/latest/security
        type: Directory
```

表 3-4 Deployment 重要配置参数说明

参数	类型	说明
name	String	必填。 服务名称。
namespace	String	必填。 服务所属的命名空间。

参数	类型	说明
image	String	必填。 内容和用户自定义的镜像名称和镜像版本一致， **镜像制作**中Dockerfile已创建好的镜像。
terminationGracePeriodSeconds	Int	允许进程优雅退出的时间，单位秒，优雅退出过程中存在端口占用的可能，若用户删除deployment后退出过程重新部署，可能因端口占用启动失败。
runAsUser	Int	必填。 容器用户ID，和Dockerfile的UID一致。
runAsGroup	Int	必填。 容器所属组ID，和Dockerfile的UID一致。
volumeMounts	List	选填。 用户需要挂载的容器路径（mountPath），需要与volumes参数中的name一一对应进行挂载，需要被挂载的物理路径都可以配置name和mountPath参数对应到容器路径。 – dir1：日志路径；如需挂载，则容器路径必须为/var/log/mindie-ms/。 – dir2：容器内持久化保存业务数据status.json文件路径；路径建议与volumes参数中的dir2保持一致。 – security: 挂载到容器内的证书目录路径。 挂载容器路径时，请勿挂载到**步骤6**中的{USER_HOME_DIR}路径，防止覆盖。

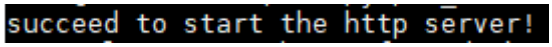
参数	类型	说明
volumes	List	选填。 用户需要挂载物理机路径（hostpath），需要与volumeMounts参数中的name一一对应进行挂载，需要保证挂载的物理路径容器内用户可读且可写。 用户自定义被挂载的物理路径： – dir1：服务端写入的日志目录；该路径需要保证可读取且真实存在，且文件属主需要与容器内用户保持一致。首次启动MindIE MS会自动创建日志文件，用户无需在该目录下创建文件，避免权限出现问题。 – dir2：持久化保存业务数据文件（status.json）的物理机路径，需和**ms_server.json配置文件**中的ms_status_file参数保持一致，用户不能创建文件，只能填写目录，目录要求真实存在且可读。 – security: 宿主机上的证书目录，{用户工作目录}是用户安装mindie-service的所在的目录。
command	List	必填。 启动容器的运行指令。根据用户实际使用场景，填入真实且可以正常启动MindIE MS服务的命令。 用户需保证命令的安全性。 – /home/*{容器内用户名}*：与**步骤6**中的{USER_HOME_DIR}一致。 – ms_server.json：启动服务配置文件，请参见**ms_server.json配置文件**进行配置。

2. 在宿主机上创建日志文件目录和数据文件目录。
```
mkdir -p /var/lib/mindie-ms/
mkdir -p /var/log/mindie-ms/run
mkdir -p /var/log/mindie-ms/operation
chown -R {容器内uid}:{容器内gid} /var/lib/mindie-ms/
chown -R {容器内uid}:{容器内gid} /var/log/mindie-ms/
chmod 750 /var/log/mindie-ms/run
chmod 750 /var/log/mindie-ms/operation
chmod 750 /var/lib/mindie-ms/
```
3. 使用以下命令创建Deployment。
```
kubectl apply -f {Deployment配置文件路径}
```
4. 使用以下命令查看Deployment的状态，如果为“Running”，则表示创建成功，如**图3-11**所示。
```
kubectl get pod -A
```

图 3-11 Deployment 状态

5. 使用以下命令查看Deployer服务端是否部署成功，如图3-12所示，则表示部署成功。
   kubectl logs *{NAME}* -n *{NAMESPACE}*
   - *{NAME}*：该参数为步骤3.4中查询到的NAME值。
   - *{NAMESPACE}*：该参数为步骤1中创建的服务所属的命名空间。

   图 3-12 Deployer 服务端部署成功

步骤4 卸载Deployer服务端。

如不再使用Deployer服务端，可使用kubectl命令卸载服务。
kubectl delete -f *{Deployment配置文件路径}*

回显如下所示则表示卸载成功：

deployment.apps *{Deployment配置文件中的name}* deleted

说明

- Deployer服务端部署成功后，可使用MindIE MS客户端和MindIE MS服务端进行交互，详细命令请参见 **使用msctl部署服务**。
- 启动时如报错打印“change file mode error 30”，是由于内部KMC组件尝试在read only的文件系统中修改相关密钥文件权限失败，按照操作指导，文件权限已设置正常，不影响功能，可忽略。
- 启动时如出现“[warn] [1] entropy may not be enough”，是由于环境熵不足，需要安装haveged组件补熵。详情请参考《**MindIE安装指南**》中“附录 > 启动haveged服务”章节。

----结束

3.4.2.2.2 部署 MindIE Server

部署推理服务前准备

在使用MindIE MS服务端API或者MindIE MS客户端命令行工具创建一个新的推理服务之前，需要完成以下的集群环境配置：

- MindIE MS部署一个服务时会挂载以下宿主机路径至Pod容器内，主机路径和容器挂载路径一致，并确保每个计算节点都包含以下文件。

  为了避免挂载软链接带来容器逃逸风险，用户需要保证以下宿主机路径不是软链接：

  - /mnt/mindie-service/ms/config/config.json：MindIE Server的config.json配置文件路径，文件权限设置为640，文件属主设置为MindIE镜像容器内用户群组。配置文件详情请参见**8.4.2 配置参数说明**。
  - /mnt/mindie-service/ms/model：模型文件目录，MindIE Server的config.json配置文件中的modelWeightPath参数需要配置为该目录下的权重路径。（例如：/mnt/mindie-service/ms/model/atb_testdata/weights/llama1-65b-safetensors）

    如果用户已在其他路径（如：/data）存放模型文件数据，可通过mount --bind /data/ /mnt/mindie-service/ms/model命令创建一个挂载点。重启服务器后会失效，需要重新创建挂载点。该目录权限需要设置为750，目录内文件权限设置为640，属主设置为MindIE镜像容器内用户群组。

  - /mnt/mindie-service/ms/writable-data：可写目录，容器内程序可将日志写入该目录下。目录权限需要设置为750，属主设置为MindIE镜像容器内用户群组。
  - /mnt/mindie-service/ms/run/run.sh：启动脚本文件路径，需要修改文件权限为容器内用户可执行的权限。启动Pod容器时会调用该脚本启动MindIE推理服务进程，单机样例参考**单机场景非root用户镜像启动脚本样例**。该文件权限需要设置为550，属主设置为MindIE镜像容器内用户群组。
- 当前服务部署的namespace以mindie为例，用户使用MindIE MS部署服务之前需保证已经通过kubectl创建该命名空间。
- 已在计算节点上参照**3.3 准备MindIE Server镜像**章节准备好MindIE Server镜像，并使用以下命令更改名称为mindie-server:RC3，当前部署的镜像名和镜像版本只能为mindie-server:RC3。
  ```
  docker tag mindie:{镜像名} mindie-server:RC3
  ```

脚本示例

单机场景非root用户镜像启动脚本样例如下所示：

```
LD_LIBRARY_PATH=/usr/local/python3.10.2/lib:/usr/local/Ascend/driver/lib64/common:/usr/local/Ascend/driver/lib64/driver:
PATH=/usr/local/python3.10.2/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:
export ASCEND_RT_VISIBLE_DEVICES=0,1,2,3,4,5,6,7                    #3-7行为运行所需环境变量
source /home/{用户名称}/Ascend/ascend-toolkit/set_env.sh
source /home/{用户名称}/Ascend/mindie/set_env.sh
source /home/{用户名称}/Ascend/nnal/atb/set_env.sh
source /home/{用户名称}/Ascend/llm_model/set_env.sh
cd /home/{用户名称}/Ascend/mindie/latest/mindie-service              #8-9行为运行启动命令/
./bin/mindieservice_daemon
```

说明

- {用户名称}为《MindIE安装指南》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”章节中非root用户镜像制作中的用户名称。
- 受限于kubernetes的能力，默认情况下configmap挂载到容器内文件的属主为root:root，且不可修改用户，只能设置群组。MindIE MS通过添加配置fsGroup 1001将挂载到容器内的confimap文件设置为1001群组，支持容器内非root用户读取。

配置自动生成证书

若用户需要开启MindIE Server的TLS认证功能（HTTPS或GRPC）时，通信客户端需要校验服务端证书的IP，由于podIP的动态性，需要在Pod启动时生成具有podIP别名的服务证书，以实现MindIE Server中master和slave间的通信，以及MindIE MS对MindIE Server的证书认证和校验。MindIE提供证书生成能力，具体操作步骤如下所示。

前提条件

在部署服务前准备MindIE Server服务端的CA证书和加密私钥。

操作步骤

步骤1 本地已通过MindIE证书管理工具import_cert接口导入CA证书和私钥，输入证书私钥口令、生成KMC加密口令文件和KMC密钥库文件。

步骤2 准备生成证书的输入输出配置文件（gen_cert.json）。

```
{
    "ca_cert": "./security/ca/ca.pem",
    "ca_key": "./security/ca/ca.key.pem",
```



```
    "ca_key_pwd": "./security/ca/ca_passwd.txt",
    "cert_config": "./cert_info.json",
    "output_path": "./gen_cert_output",
    "kmc_ksf_master": "./tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./tools/pmt/standby/ksfb"
}
```

server管理面证书输入输出配置文件（gen_management_cert.json）。

```
{
    "ca_cert": "./security/ca/management_ca.pem",
    "ca_key": "./security/ca/management_ca.key.pem",
    "ca_key_pwd": "./security/ca/management_ca_passwd.txt",
    "cert_config": "./cert_info.json",
    "output_path": "./gen_cert_output",
    "kmc_ksf_master": "./tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./tools/pmt/standby/ksfb"
}
```

步骤3 准备待生成证书的配置文件（cer_config.json）。

```
{
    "subject": "subject_name",
    "expired_time": 365,
    "serial_number": 123,
    "req_distinguished_name": {
        "C": "***",
        "ST": "***",
        "L": "***",
        "O": "***",
        "OU": "***",
        "CN": "***"
    },
    "alt_names": {
        "IP": [],
        "DNS": []
    }
}
```

步骤4 将**步骤1**~**步骤3**的文件放在《MindIE安装指南》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”章节中的Dockerfile所在目录，随其他文件一起复制至容器内的/tmp路径下，并在Dockfilefile文件中添加以下命令将这些文件移动至mindie-service目录下。

```
RUN chmod 400 ./ca* && \
    chmod 400 ./management_ca* &&\
    chmod 600 ./cert_info.json &&\
    chmod 600 ./gen_management_cert.json &&\
    chmod 600 ./gen_cert.json &&\
    chmod 600 ./tools/pmt/master/ksfa &&\
    chmod 600 ./tools/pmt/standby/ksfb &&\
    mv ./ca.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./ca.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca  && \
    mv ./ca_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./tools  /home/{用户名称}/Ascend/mindie/latest/mindie-service/tools && \
    mv ./gen_cert.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
    mv ./gen_management_cert.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
    mv ./cert_info.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
```

{用户名称}为容器内的用户账号。

步骤5 在**单机场景非root用户镜像启动脚本样例**中“./bin/mindieservice_daemon”所在行之前添加以下生成证书的命令。

```
export WORK_DIR=/home/{用户名称}/Ascend/mindie/latest/mindie-service/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$WORK_DIR/lib
export HSECEASY_PATH=$WORK_DIR/lib
```



```
cd $WORK_DIR
chmod 500 ./bin/gen_cert
mkdir gen_cert_output
python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_cert.json  --ip=$MIES_CONTAINER_IP,{host_ip}
chmod 400 ./gen_cert_output/*
cp ./gen_cert_output/cert.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/certs/server.pem
cp ./gen_cert_output/cert.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/keys/server.key.pem
cp ./gen_cert_output/cert_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/pass/mindie_server_key_pwd.txt
rm -rf ./gen_cert_output/*
python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_management_cert.json  --ip=$MIES_CONTAINER_IP,{host_ip}
chmod 400 ./gen_cert_output/*
cp ./gen_cert_output/cert.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/certs/management_server.pem
cp ./gen_cert_output/cert.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/keys/management_server.key.pem
cp ./gen_cert_output/cert_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/pass/management_mindie_server_key_pwd.txt
rm -rf ./gen_cert_output/*
```

{host_ip}：为提供推理API的物理机IP。

----结束

⚠ 注意

启动MindIE Server Pod调用生成证书接口如出现“failed to read random number from system.”报错，大概率是由于环境熵不足，需要在计算节点安装haveged组件补熵。详情请参考《**MindIE安装指南**》中“附录 > 启动haveged服务”章节，将熵补至4096。

准备 Deployer msctl 命令行客户端

工具准备

以当前所在路径为/home/*{用户名称}*/package为例进行以下操作。

步骤1 使用以下命令将Ascend-mindie-service_*{version}*_linux-aarch64.run安装后的msctl.json文件拷贝至*{$HOME}*/.mindie_ms/路径下， 且文件权限不高于640。

```
cp mindie-service/latest/conf/msctl.json ~/.mindie_ms/
```

步骤2 打开msctl.json文件并根据实际情况参考**表3-5**修改启动配置文件，配置样例如下所示（格式必须和样例保持一致）。

```
{
    "http":{
        "dstPort" : 9789,
        "dstIP" : "127.0.0.1",
        "ca_cert": "./security/ca/ca.pem",
        "tls_cert": "./security/certs/cert.pem",
        "tls_key": "./security/keys/cert.key.pem",
        "tls_crl": "",
        "timeout": 60
    },
    "log_level": "INFO"
}
```


表 3-5 启动配置参数

参数	类型	说明
dstPort	Int	必填。 对端端口（ MindIE MS服务端），与ms_server.json中的port保持一致。 取值范围：[1024，65535]，默认值9789。
dstIP	String	必填。 部署服务端IP地址，Deployment部署场景使用MindIE MS Server的Pod IP进行访问 。 可通过以下命令查询Pod IP： `kubectl get pods -n mindie -o wide`
ca_cert	String	必填。 默认使用HTTPS通信。 ca根证书文件。需MindIE MS服务端ca证书文件相互信任。
tls_cert	String	必填。 客户端tls证书文件。用户请参考下面步骤2准备证书。
tls_key	String	必填。 客户端tls私钥文件。出于安全的考虑，建议使用加密私钥，如使用，在执行命令时会提供用户输入加密口令。
tls_crl	String	必填。 校验通信对端服务端的证书吊销列表crl文件，如为空，则不进行吊销校验。
timeout	Int	必填。 MindIE MS客户端可设置的等待响应时长，取值范围[1,60]。
log_level	String	必填。 客户端日志级别。 ● DEBUG ● INFO ● WARNING ● ERROR ● CRITICAL

步骤3 进入到infer_server.json配置文件所在路径并打开。

```
cd /mindie-service/latest/conf
vi infer_server.json
```


步骤4 将infer_server.json配置文件的“server_type”参数改为“mindie_single_node”，infer_server.json配置文件如下所示，其参数解释请参见**表3-6**。

```
{
   "server_name": "mindie-server",
    "replicas": 1,
    "service_port": 31005,
    "cross_node_num":2,
    "service_type": "NodePort",
    "server_type":"mindie_single_node",
    "scheduler": "default",
    "init_delay": 180,
    "liveness_timeout": 10,
    "liveness_failure_threshold": 1,
    "readiness_timeout": 10,
    "readiness_failure_threshold": 1,
    "resource_requests": {
        "memory":   500000,
        "cpu_core" : 32000,
        "npu_type": "Ascend910",
        "npu_chip_num": 8
    },
    "mindie_server_config": {
        "mies_install_path": "/usr/local/Ascend/mindie/latest/mindie-service",
        "infer_port" : 1025,
        "management_port":1026,
        "enable_tls":true
    },
    "npu_fault_reschedule": true
    "termination_grace_period_seconds": 30,
    "max_unavailable": "50%",
    "max_surge": "50%"
}
```

表 3-6 infer_server.json 配置文件参数解释

参数	类型	说明
server_name	String	必填。 任务服务名称。在MindIE MS管理范围内具备唯一性的ID，当前只支持部署一个服务。
replicas	Int	必填。 部署的实例副本数量，其中实例指的是最小可提供推理服务的单元。 ● 分布式多机场景当前只支持部署一个实例。 ● 单机场景可部署多个实例，范围为[1, 256]，具体配置数量根据物理资源进行配置。
service_port	Int	必填。 部署服务对外可被用户访问的端口，NodePort取值范围[30000, 32767]。 访问MindIE推理请求时需要使用该接口。
cross_node_num	Int	分布式多机场景必填，**单机场景不支持该参数**。 一个实例跨机部署的节点个数，当前只支持取值为2或4。

参数	类型	说明
service_type	String	必填。 创建Kubernetes的Service资源类型，当前只支持NodePort类型。 NodePort：表示从该参数获取Service的资源类型。
server_type	String	必填。 服务部署的形态，支持mindie_cross_node为多机部署和mindie_single_node单机部署。
scheduler	String	必填。 可选调度器类型，分布式多机场景仅支持default类型，单机场景支持default和Volcano类型；如需启用NPU故障重调度功能，需要选择“volcano”调度器。 ● default：使用Kubernetes默认调度器。 ● volcano：使用MindCluster Volcano调度器。
init_delay	Int	必填。 允许推理实例启动的时间。 ● 分布式多机场景，从Pod启动后ranktable完成生成的时间开始计算，在这个时间内MindIE MS会周期性检测服务实例是否就绪，超过这个时间服务未就绪，实例将自动重启。 ● 单机场景，从pod启动的时间计算，超过这个时间，存活探针开始接管，若live检测失败（5秒触发一次），会触发重启。 取值范围为[10, 1800]， 单位为秒。
liveness_timeout	Int	必填。 存活探针超时时间设置，取值范围[1, 600]，单位秒。 注意在高并发场景下，liveness探针可能超时，此时认为MindIE Server状态异常，并对MindIE Server Pod进行重启。建议用户根据业务并发度配置合理的值，可以使用MindIE Benchmark工具设置并发量发送请求进行模拟测试，确保Pod在并发度条件下没有发生重启。 参考示例：llama2-7b在Atlas 800I A2 推理产品上单卡单实例部署，使用MMLU数据集将Benchmark并发度设置为1000，超时时间配置为10秒。

参数	类型	说明
liveness_failure_threshold	Int	必填。存活探针最大允许的失败次数，取值范围[1, 10]。 参考示例：llama2-7b在Atlas 800I A2 推理产品上单卡单实例部署，使用MMLU数据集将Benchmark并发度设置为1000，失败次数设置为2时Pod不重启。
readiness_timeout	Int	必填。就绪探针超时时间设置，取值范围[1, 600]，单位秒。 就绪探针超时时间配置不足且当请求并发量大时，可能导致就绪探针超时，超时后Service状态变成“未就绪”，无法正常发送推理请求。
readiness_failure_threshold	Int	必填。就绪探针最大允许的失败次数，建议配置为1。 取值范围[1, 10]。
memory	Int	必填。 服务所需最少内存资源大小，单位为M。 默认值256000，取值范围[1000, 256000]。
cpu_core	Int	必填。 服务所需最少CPU资源大小，取值范围[1000, 256000]，单位为M（1000M等于1核心）。
npu_type	String	必填。 服务所需NPU卡类型。 ● 分布式多机场景：支持输入"Ascend910"。 ● 单机场景：支持输入"Ascend910"和"Ascend310P"。
npu_chip_num	Int	必填。 单个节点所需NPU卡的数量。规格限制[1, 8]； 单机场景下，是一个实例占用的卡数； 分布式多机场景下，是一个实例在每个节点上占用的卡数，会分配匹配的卡资源并写入ranktable。且只支持全卡推理，当前仅支持配置为8卡的设备。

参数	类型	说明
mindie_server_config	Object	必填。 • infer_port：推理服务的推理端口。 • management_port：推理服务的管理端口，用于查询服务状态。 • enable_tls：是否开启https，使用服务端启动时，配置的证书发送tls请求需要与MindIE Server的**config.json配置文件**中httpsEnabled参数、**表8-16**中client_mindie_server_tls_enable参数的值保持一致。 • mies_install_path： MindIE Service在镜像内的安装路径。
infer_port	Int	必填。推理服务的推理端口。取值范围[1024, 65535]
management_port	Int	必填。推理服务的管理端口。取值范围[1024, 65535]
enable_tls	Bool	必填。是否开启https。
mies_install_path	String	必填。MindIE Service在镜像内的安装路径。
npu_fault_reschedule	Bool	NPU故障重调度（主动识别NPU设备故障并重调度Pod到正常NPU设备）配置，是否以优雅方式进行重调度。前提安装volcano组件，**该参数仅对单机场景生效**。 • true：表明使用优雅方式重调度，并在重调度过程优雅删除原Pod。 • false：表明使用非优雅方式重调度，在重调度过程中强制删除原Pod。
termination_grace_period_seconds	Int	Pod优雅退出时间，**单机场景必填，分布式多机场景不支持**。 设置Pod优雅退出最大的容忍时间，超过该时间K8s会强制杀掉服务。滚动更新过程，若设置时间不足，可能导致剩余请求未处理完中断。 参考配置：以llama2-7b模型实例数2为例，在Atlas 300I Duo 推理卡+Atlas 800 推理服务器（型号：3000）单卡设备使用gsm8k数据集并发度为1000的请求，需要配置Pod优雅退出时间为800秒，保证Pod退出过程业务不中断。 取值范围[0, 3600]；单位为秒。 在服务启动初始，未就绪时，此时服务无法正常处理优雅退出信号，删除服务将无法触发优雅退出，需等待程序自动退出或达到优雅退出的最大容忍时间。

参数	类型	说明
max_unavailable	String	滚动更新配置，**单机场景必填，分布式多机场景不支持**。 更新所需的可以处于不可用状态的最大 Pod 数量，指的是replicas * max_unavailable向下取整。 max_unavailable格式必须是“{int数字}%”，取值范围0%~100%。
max_surge	String	滚动更新配置，**单机场景必填，分布式多机场景不支持**。 更新所需的可以超出期望副本数的最大Pod数量，指replicas * max_surge向上取整。 max_surge格式必须是“{int数字}%，取值范围0%~100%。

步骤5 使用以下命令进入mindie-service的安装路径。

```
cd ./mindie-service/latest
```

步骤6 导入**签发其他证书**中MindIE MS客户端msctl与MindIE MS服务端通信证书（即CERT_CN取值为msclientuser签发的证书）。

在当前路径下创建证书目录，导入中间CA证书、MindIE MS客户端msctl与MindIE MS服务端通信的证书和加密私钥。

```
mkdir -m 700 ./security
mkdir -m 700 ./security/certs
mkdir -m 700 ./security/keys
mkdir -m 700 ./security/ca
mkdir -p msclientuser_chian
cat /home/{用户名称}/cas/msclientuser/cert.pem /home/{用户名称}/cas/mindiems/cert.pem > ./msclientuser_chian/cert.pem  #/home//{用户名称}cas/msclientuser/cert.pem为：MindIE MS客户端msctl与MindIE MS服务端通信的证书；/home/{用户名称}/cas/mindiems/cert.pem为：中间CA证书文件路径
chmod 400 ./msclientuser_chian/cert.pem
cp ./msclientuser_chian/cert.pem  ./security/certs
cp /etc/kubernetes/pki/ca.crt  ./security/ca/ca.pem          #/etc/kubernetes/pki/ca.crt为：Kubernetes根CA证书文件路径
cp /home/{用户名称}/cas/msclientuser/cert.key.pem  ./security/keys     #/home/{用户名称}/cas/msclientuser/cert.key.pem为：MindIE MS客户端msctl与MindIE MS服务端通信的私钥文件路径
rm -rf ./msclientuser_chian
```

步骤7 执行以下命令导入环境变量，将所依赖的so载入LD_LIBRARY_PATH。

```
export LD_LIBRARY_PATH=/path/to/mindie-service/latest/lib:$LD_LIBRARY_PATH
```

----结束

使用 msctl 部署服务

工具使用

MindIE MS Deployer提供命令行工具，可供用户使用，当前提供的对外命令为：

表 3-7 客户端命令

指令	类型	说明
./bin/msctl create infer_server -f ./conf/infer_server.json	create	客户端服务部署。 配置文件需要不大于640权限，否则启动失败。
./bin/msctl delete infer_server -n *{服务名称}*	delete	卸载服务，删除服务相关的Kubernetes资源，停止对外服务。
./bin/msctl get infer_server -n *{服务名称}*	get	获取服务状态，包括服务是否已就绪。
./bin/msctl -h/--help	-	命令使用帮助。
./bin/msctl update infer_server -f *./conf/update_server.json*	update	更新推理服务配置。

客户端部署命令

使用MindIE MS命令行客户端msctl发送部署请求命令，如下所示。

```
./bin/msctl create infer_server -f ./conf/infer_server.json
```

回显如下所示，则表示部署请求命令下发成功。

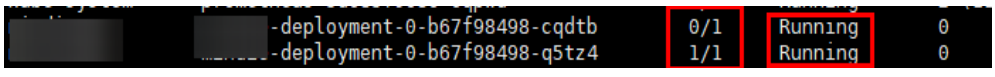
```
{
   "message": "creating the server!",
   "status": "0"
}
```

执行以下Kubernetes查询指令，查询是否部署成功。

```
kubectl get pod -A
```

如下图所示，在Kubernetes的Pod资源用户自定义的*{namespace}*中，存在*{server_name}*-deployment-0-*xxx*且状态为Running时，则表示部署成功。

*{server_name}*和*{namespace}*为**8.3.2.3 RESTful接口API**中用户自行编写的json配置文件中的“server_name”和“namespace”参数值。



说明

如果部署失败，请参见**表3-4**中的volumes参数，根据挂载的物理日志路径中的日志文件定位具体问题。

客户端卸载命令

使用msctl在MindIE MS客户端发送卸载请求命令，如下所示。

```
./bin/msctl delete infer_server -n {server_name}
```

回显如下所示，则表示卸载请求命令下发成功且服务删除成功。


```
{
   "message": "succeed to clear resources",
   "status": "0"
}
```

也可以使用Kubernetes命令查看该服务是否已删除或者处于非Running状态，如下图所示。

```
kubectl get pod -A
```

客户端查询命令

使用msctl在MindIE MS客户端发送查询请求命令，如下所示。

```
./bin/msctl get infer_server -n {server_name}
```

回显如下所示，则表示查询请求命令下发成功。

```
{
   "data": {
      "instances_status": [
         {
            "liveness": true,
            "pod_name": "mindie-server-zsm-586c8fb5f8-vtx2n",
            "readiness": true
         }
      ],
      "model_info": {
         "docker_label": null,
         "max_batch_total_tokens": 8192,
         "max_best_of": 1,
         "max_concurrent_requests": 200,
         "max_input_length": 2048,
         "max_stop_sequences": null,
         "max_waiting_tokens": null,
         "models": [
            {
               "max_total_tokens": 2560,
               "model_device_type": "npu",
               "model_dtype": "float16",
               "model_id": "llama2_7b",
               "model_pipeline_tag": "text-generation",
               "model_sha": null
            }
         ],
         "sha": null,
         "validation_workers": null,
         "version": "1.0.RC3",
         "waiting_served_ratio": null
      },
      "server_name": "mindie-server-zsm"
   },
   "message": "success",
   "status": "0"
}
```

重要参数解释：

- “liveness”：表示服务存活状态。
  - “true”：表示服务存活。
  - “false”：表示服务未存活。
- “readiness”：表示服务实例启动状态。

- “true”：表示服务实例已启动完成并进入就绪状态。
- “false”：表示服务实例未启动完成。

客户端滚动更新命令

使用msctl在MindIE MS客户端发送更新请求命令，如下所示。

```
./bin/msctl update infer_server -f ./conf/update_server.json
```

回显如下所示，则表示更新请求命令下发成功。

```
{
   "message": "update server success.",
   "status": "0"
}
```

3.4.2.3 使用 kubectl 部署服务示例

说明

- 该示例支持部署单机版的mindie_server、ms_coordiantor、ms_controller。
- Atlas 800I A2 推理产品需要配置卡IP并安装MindCluster中的HCCL-Controller组件；Atlas 300I Duo 推理卡+Atlas 800 推理服务器（型号：3000）无需配置和安装。
- 当前部署脚本不支持NPU故障重调度场景。

本小节使用mindie-service run包安装目录下（./mindie-service/latest/examples/kubernetes_deploy_scripts）中的脚本实现调用kubectl进行服务部署，脚本提供一键式部署和卸载单机形态集群功能，集群管理员用户可参考这些脚本线下使用K8s kubectl工具部署服务。

集群管理员用户只需在管理节点完成启动脚本编写、业务配置和K8s配置，再调用部署脚本，实现自动下发业务配置和启动脚本，自动生成包含节点设备信息的ranktable，并调度Pod到计算节点。

脚本文件所在目录结构如下所示：

```
├── boot_helper
│   ├── boot.sh
│   ├── get_group_id.py
│   └── update_mindie_server_config.py
├── chat.sh
├── conf
├── delete.sh
├── deployment
│   ├── mindie_ms_controller.yaml
│   ├── mindie_ms_coordinator.yaml
│   ├── mindie_server_heterogeneous.yaml
│   └── mindie_server.yaml
├── deploy.sh
├── generate_stream.sh
├── gen_ranktable_helper
│   ├── gen_global_ranktable.py
│   └── global_ranktable.json
└── log.sh
```

关键目录及文件解释如下所示：

- conf：MindIE MS和MindIE Server的主要业务配置文件，PD分离管理调度策略和模型相关配置。
- boot_helper：包含容器启动脚本boot.sh，获取group_id，刷新环境变量到配置文件，设置启动程序的环境变量等，用户可根据需要在这里调整日志等级等，环境变量详细介绍请参见**表3-12**。

- deployment: K8s部署任务定义，配置NPU资源使用量，实例数，镜像等。
- gen_ranktable_helper：生成global ranktable的工具，用户无需感知。
- chat.sh：使用curl发送HTTPS请求给推理服务的简单对话示例。
- generate_stream.sh：使用curl发送HTTPS请求给推理服务的流式响应示例。
- deploy.sh：部署入口脚本，一键拉起所有MindIE组件。
- delete.sh：卸载脚本，一键卸载所有MindIE组件。
- log.sh：查询Pod的打屏日志，可查询到部署的所有Pod的日志。

部署样例如下所示，以下操作均在部署脚本路径下完成：

步骤1 创建mindie命名空间。

```
kubectl create namespace mindie
```

步骤2 将mindie-service发布件中conf目录的config.json、http_client_ctl.json、ms_controller.json和ms_coordinator.json文件拷贝至脚本conf目录（./mindie-service/latest/examples/kubernetes_deploy_scripts/conf）。

步骤3 配置MindIE MS的Controller和Coordinator组件，将这两个组件的部署模式配置为单机（非分布式）服务部署模式，其必配参数如下所示。

- 配置ms_controller.json文件，参数详请请参见**8.3.3.3 配置说明**章节。

  将部署模式配置为单机（非分布式）服务部署模式，需配置以下参数。

  ```
  "deploy_mode"= "single_node"
  ```

- 配置ms_coordinator.json文件，参数详请请参见**8.3.4.3 配置说明**章节。
  - 配置单机（非分布式）服务部署模式，需配置以下参数。

    ```
    "deploy_mode"= "single_node"
    ```

  - 使能Prefix Cache特性，需配置以下参数。（Prefix Cache特性目前只支持单机部署场景）

    ```
    "scheduler_type": "default_scheduler",
    "algorithm_type": "cache_affinity",
    ```

步骤4 配置MindIE Server服务启动的config.json配置文件，单机（非分布式）服务部署模式需要配置以下参数，具体参数解释请参见**8.4.2 配置参数说明**。

- modelWeightPath：模型权重文件目录配置，默认情况下脚本会挂载物理机的/data目录，该参数需配置为/data路径下的模型权重路径，确保集群可调度计算节点在该路径下存在模型文件。
- worldSize：配置一个实例需要占用的卡数；例如配置为“2”，表示使用两张卡。
- npuDeviceIds：卡号配置成从0开始编号，总数与worldSize一致，如配置为[[0,1]]。
- 使能Prefix Cache场景：
  - 在ModelDeployConfig中的ModelConfig下添加以下信息：

    ```
    "plugin_params": "{\"plugin_type\":\"prefix_cache\"}"
    ```

  - 在ScheduleConfig中添加以下信息：

    ```
    "enablePrefixCache": true
    ```

步骤5 配置启动脚本boot.sh。

MindIE Server环境变量配置单机部署模式：

```
export PD_MODE=3
```


步骤6 配置kubernetes Deployment。

在部署脚本目录中的deployment目录下找到mindie_server.yaml、mindie_ms_coordinator.yaml和mindie_ms_controller.yaml文件。

说明

用户在使用kubetl部署Deployment时，需要修改脚本目录deployment中的.yaml配置文件，请避免使用危险配置，确保使用安全镜像（非root权限用户），配置Pod安全上下文，挂载安全路径（非软链接、系统危险路径或业务敏感路径）并设置了安全的文件目录权限。本脚本仅作为一个部署参考，Pod容器的安全性由用户自行保证，实际生产环境请针对镜像和Pod安全进行加固。

- mindie_server.yaml文件主要配置的字段如下所示：
  - replicas：配置总实例数。
  - huawei.com/Ascend910（或310P）：resources资源请求，配置一个实例占用的NPU卡数，与MindIE Serve的config.json配置文件中worldSize参数配置的卡数保持一致。
  - image：配置镜像名。
  - nodeSelector：节点选择，用户期望调度的节点，通过节点标签实现。
  - ring-controller.atlas：根据实际使用的设备型号配置为ascend-910b或ascend-310p。
  - livenessProbe：存活探针，当服务启动时间较长，需要延迟存活检测开始时间，需配置“initialDelaySeconds”参数为合理值。
- mindie_ms_coordinator.yaml和mindie_ms_controller.yaml文件主要配置的字段如下所示：
  - image：配置镜像名。
  - nodeSelector：节点选择，用户期望调度的节点，通过节点标签实现。

  说明

  另外，mindie_ms_coordinator.yaml文件还需特别关注livenessProbe（存活探针）参数，在高并发推理请求场景下，可能会因为探针超时，从而被K8s识别为Pod不存活，导致K8s重启Coordinator容器，建议用户谨慎开启livenessProbe（存活探针）。

步骤7 拉起集群。

首先配置容器内mindie安装的目录：根据制作镜像时实际的安装路径，修改MINDIE_USER_HOME_PATH的value值，如安装路径是/xxx/Ascend/mindie, 则配置为/xxx 。

```
export MINDIE_USER_HOME_PATH={镜像的安装路径}
```

使用以下命令拉起集群。

```
bash deploy.sh
```

执行后，会同步等待global ranktable生成完成，如长时间处于阻塞状态，请ctrl+c中断后查看集群Pod状态，进行下一步的调试定位。

说明

- 集群默认刷新挂载到容器内的configmap的频率是60s，如遇到容器内打印“status of ranktable is not completed”日志信息的时间偏久，可在每个待调度的计算节点修改kubelet同步configmap的周期，即修改/var/lib/kubelet/config.yaml中的syncFrequency参数，将周期减少到5s，注意此修改可能影响集群性能。
  ```
  syncFrequency: 5s
  ```
  然后使用以下命令重启kubelet：
  ```
  swapoff -a
  systemctl restart  kubelet.service
  systemctl status kubelet
  ```
- 确保Docker配置了标准输出流写入到文件的最大规格，防止磁盘占满导致Pod被驱逐。
  需在待部署服务的计算节点上修改Docker配置文件后重启Docker：
  1. 使用以下命令打开daemon.json文件。
     ```
     vim /etc/docker/daemon.json
     ```
     在daemon.json文件中添加日志选项“log-opts”，具体内容如下所示。
     ```
     "log-opts":{"max-size":"500m", "max-file":"3"}
     ```
     参数解释：
     max-size=500m：表示一个容器日志大小上限是500M。
     max-file=3：表示一个容器最多有三个日志，超过会自动滚动更新。
  2. 使用以下命令重启Docker.
     ```
     systemctl daemon-reload
     systemctl restart docker
     ```

步骤8 查看集群状态。

通过kubectl命令查看Pod状态：

```
kubectl get pods -n mindie
```

如启动4个MindIE Server 实例，回显如下所示：

```
NAME                                 READY   STATUS    RESTARTS   AGE    IP              NODE       NOMINATED
NODE   READINESS GATES
mindie-ms-controller-7845dcd697-h4gw7   1/1     Running   0          145m   xxx.xxx.xxx     ubuntu10
<none>         <none>
mindie-ms-coordinator-6bff995ff8-l6fwz   1/1     Running   0          145m   xxx.xxx.xxx     ubuntu10
<none>         <none>
mindie-server-7b795f8df9-2xvh4           1/1     Running   0          145m   xxx.xxx.xxx   ubuntu
<none>         <none>
mindie-server-7b795f8df9-j4z7d           1/1     Running   0          145m   xxx.xxx.xxx   ubuntu
<none>         <none>
mindie-server-7b795f8df9-v2tcz           1/1     Running   0          145m   xxx.xxx.xxx   ubuntu
<none>         <none>
mindie-server-7b795f8df9-vl9hv           1/1     Running   0          145m   xxx.xxx.xxx   ubuntu
<none>         <none>
```

- 以mindie-ms-controller开头的为MindIE MS的Controller控制器组件。
- 以mindie-ms-coordinator开头的为MindIE MS的Coordinator调度器组件。
- 以mindie-server开头的为MindIE Server推理服务。

如观察Pod进入Running状态，表示pod容器已成功被调度到节点并正常启动，但还需要进一步确认业务程序是否启动成功。

通过脚本示例提供的log.sh脚本可查询这些Pod的标准输出日志，查看是否出现异常：

```
bash log.sh
```

- 如需要查询具体某个Pod（如上面mindie-server-7b795f8df9-2xvh4）的日志，则执行以下命令：
  ```
  kubectl logs mindie-server-7b795f8df9-2xvh4 -n mindie
  ```


- 如需要进入容器查找更多定位信息，则执行以下命令：
  ```
  kubectl exec -it mindie-server-7b795f8df9-2xvh4 -n mindie -- bash
  ```

步骤9 通过脚本示例提供的chat.sh发起推理请求。

需修改chat.sh中的IP地址为当前节点的IP地址，其中role配置为user。

```
bash chat.sh
```

步骤10 卸载集群。

如需停止单机服务或者修改业务配置重新部署实例，需要调用以下命令卸载已部署的实例，重新部署请执行**步骤7**。

```
bash delete.sh
```

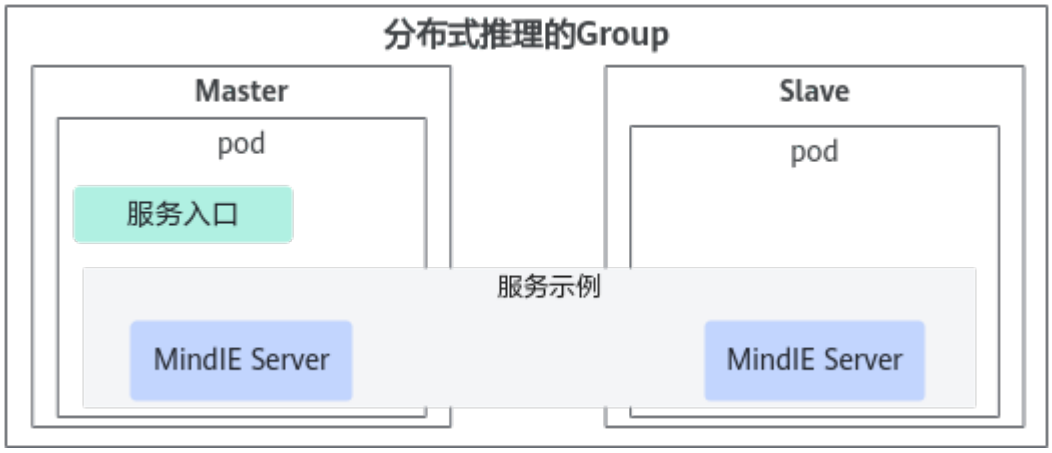
----结束

3.5 分布式多机服务部署

3.5.1 场景介绍

单个模型权重过大，单台推理机显存有限，无法容纳整个模型权重参数时，需要采用多个节点进行多机推理。分布式多机场景部署架构图如下**图3-13**所示。

图 3-13 分布式多机场景部署架构图



3.5.2 安装部署

3.5.2.1 前提条件

已参照**3.2.2 K8s安装与配置**、**3.2.3 MindCluster组件安装**和**3.3 准备MindIE Server镜像**完成K8s的安装配置、MindCluster组件的安装和MindIE Server镜像制作。

3.5.2.2 使用 MindIE MS Deployer 部署分布式多机服务示例

3.5.2.2.1 部署 MindIE MS Deployer

请参见**3.4.2.2.1 部署Deployer服务端**章节完成Deployer服务端的部署。

3.5.2.2.2 部署 MindIE Server

部署推理服务前准备

在使用MindIE MS服务端API或者MindIE MS客户端命令行工具创建一个新的推理服务之前，需要完成以下的集群环境配置：

- MindIE MS部署一个服务时会挂载以下宿主机路径至Pod容器内，主机路径和容器挂载路径一致，并确保每个计算节点都包含以下文件。

  为了避免挂载软链接带来容器逃逸风险，用户需要保证以下宿主机路径不是软链接：

  - /mnt/mindie-service/ms/config/config.json：MindIE Server的config.json配置文件路径，文件权限设置为640，文件属主设置为MindIE镜像容器内用户群组。配置文件详情请参见**8.4.2 配置参数说明**。
  - /mnt/mindie-service/ms/model：模型文件目录，MindIE Server的config.json配置文件中的modelWeightPath参数需要配置为该目录下的权重路径。（例如：/mnt/mindie-service/ms/model/atb_testdata/weights/llama1-65b-safetensors）

    如果用户已在其他路径（如：/data）存放模型文件数据，可通过mount --bind /data/ /mnt/mindie-service/ms/model命令创建一个挂载点。重启服务器后会失效，需要重新创建挂载点。该目录权限需要设置为750，目录内文件权限设置为640，属主设置为MindIE镜像容器内用户群组。
  - /mnt/mindie-service/ms/writable-data：可写目录，容器内程序可将日志写入该目录下。目录权限需要设置为750，属主设置为MindIE镜像容器内用户群组。
  - /mnt/mindie-service/ms/run/run.sh：启动脚本文件路径，需要修改文件权限为容器内用户可执行的权限。启动Pod容器时会调用该脚本启动MindIE推理服务进程，多机样例请参考**分布式多机场景非root用户镜像启动脚本样例**。该文件权限需要设置为550，属主设置为MindIE镜像容器内用户群组。
- 当前服务部署的namespace以mindie为例，用户使用MindIE MS部署服务之前需保证已经通过kubectl创建该命名空间。
- 当前部署的镜像名和镜像版本只能为mindie-server:RC3，用户需要在计算节点上准备MindIE Server镜像，并命名为mindie-server:RC3，镜像的制作请参见《MindIE安装指南》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”章节完成待部署任务节点MindIE Server镜像制作。
- 已在计算节点上参照**3.3 准备MindIE Server镜像**章节准备好MindIE Server镜像，并使用以下命令更改名称为mindie-server:RC3，当前部署的镜像名和镜像版本只能为mindie-server:RC3。

  ```
  docker tag mindie:{镜像名} mindie-server:RC3
  ```

脚本示例

分布式多机场景非root用户镜像启动脚本样例如下所示：

```
LD_LIBRARY_PATH=/usr/local/python3.10.2/lib:/usr/local/Ascend/driver/lib64/common:/usr/local/Ascend/driver/lib64/driver:
PATH=/usr/local/python3.10.2/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:
export ASCEND_RT_VISIBLE_DEVICES=0,1,2,3,4,5,6,7                #3-8行为运行所需环境变量
source /home/{用户名称}/Ascend/ascend-toolkit/set_env.sh
source /home/{用户名称}/Ascend/mindie/set_env.sh
source /home/{用户名称}/Ascend/nnal/atb/set_env.sh
source /home/{用户名称}/Ascend/llm_model/set_env.sh
export HCCL_OP_EXPANSION_MODE="AIV"
```



```
cd /home/{用户名称}/Ascend/mindie/latest/mindie-service          #9-10行为运行启动命令
./bin/mindieservice_daemon
```

说明

- 用户需确保容器内有网络可访问程序的HTTP端口。
- *{用户名称}*为《 MindIE安装指南 》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”章节中非root用户镜像制作中的用户名称。

配置自动生成证书

用户需要开启MindIE Server的TLS认证功能（ HTTPS或GRPC ）时，通信客户端需要校验服务端证书的IP，由于podIP的动态性，需要在Pod启动时生成具有podIP别名的服务证书，以实现MindIE Server中master和slave间的通信，以及MindIE MS对MindIE Server的证书认证和校验。MindIE提供证书生成能力，具体操作步骤如下所示。

前提条件

在部署服务前准备MindIE Server服务端的CA证书和加密私钥。

操作步骤

步骤1 本地已通过MindIE证书管理工具import_cert接口导入CA证书和私钥，输入证书私钥口令、生成KMC加密口令文件和KMC密钥库文件。

步骤2 准备生成证书的输入输出配置文件（ gen_cert.json ）。

```
{
    "ca_cert": "./security/ca/ca.pem",
    "ca_key": "./security/ca/ca.key.pem",
    "ca_key_pwd": "./security/ca/ca_passwd.txt",
    "cert_config": "./cert_info.json",
    "output_path": "./gen_cert_output",
    "kmc_ksf_master": "./tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./tools/pmt/standby/ksfb"
}
```

server管理面证书输入输出配置文件（ gen_management_cert.json ）。

```
{
    "ca_cert": "./security/ca/management_ca.pem",
    "ca_key": "./security/ca/management_ca.key.pem",
    "ca_key_pwd": "./security/ca/management_ca_passwd.txt",
    "cert_config": "./cert_info.json",
    "output_path": "./gen_cert_output",
    "kmc_ksf_master": "./tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./tools/pmt/standby/ksfb"
}
```

步骤3 准备待生成证书的配置文件（ cert_config.json ）。

```
{
    "subject": "subject_name",
    "expired_time": 365,
    "serial_number": 123,
    "req_distinguished_name": {
        "C": "***",
        "ST": "***",
        "L": "***",
        "O": "***",
        "OU": "***",
        "CN": "***"
    },
    "alt_names": {
        "IP": [],
```

```
    "DNS": []
  }
}
```

步骤4 将**步骤1**~**步骤3**的文件放在《 MindIE安装指南 》的“容器化部署和镜像制作 > **制作MindIE Server镜像**”章节中的Dockerfile所在目录，随其他文件一起复制至容器内的/tmp路径下，并在Dockfilefile文件中添加以下命令将这些文件移动至mindie-service目录下。

```
RUN chmod 400 ./ca* && \
    chmod 400 ./management_ca* &&\
    chmod 600 ./cert_info.json &&\
    chmod 600 ./gen_management_cert.json &&\
    chmod 600 ./gen_cert.json &&\
    chmod 600 ./tools/pmt/master/ksfa &&\
    chmod 600 ./tools/pmt/standby/ksfb &&\
    mv ./ca.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./ca.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca  && \
    mv ./ca_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./management_ca_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/ca && \
    mv ./tools  /home/{用户名称}/Ascend/mindie/latest/mindie-service/tools && \
    mv ./gen_cert.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
    mv ./gen_management_cert.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
    mv ./cert_info.json /home/{用户名称}/Ascend/mindie/latest/mindie-service/ && \
```

{用户名称}为容器内的用户账号。

步骤5 在**分布式多机场景非root用户镜像启动脚本样例**中“./bin/mindieservice_daemon”所在行之前添加以下生成证书的命令。

```
export WORK_DIR=/home/{用户名称}/Ascend/mindie/latest/mindie-service/
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$WORK_DIR/lib
export HSECEASY_PATH=$WORK_DIR/lib
cd $WORK_DIR
chmod 500 ./bin/gen_cert
mkdir gen_cert_output
python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_cert.json  --ip=$MIES_CONTAINER_IP,{host_ip}
chmod 400 ./gen_cert_output/*
cp ./gen_cert_output/cert.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/certs/server.pem
cp ./gen_cert_output/cert.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/keys/server.key.pem
cp ./gen_cert_output/cert_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/pass/mindie_server_key_pwd.txt
rm -rf ./gen_cert_output/*
python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_management_cert.json  --ip=$MIES_CONTAINER_IP,{host_ip}
chmod 400 ./gen_cert_output/*
cp ./gen_cert_output/cert.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/certs/management_server.pem
cp ./gen_cert_output/cert.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/keys/management_server.key.pem
cp ./gen_cert_output/cert_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/pass/management_mindie_server_key_pwd.txt
rm -rf ./gen_cert_output/*
```

{host_ip}：为提供推理API的物理机IP。

----结束

须知

启动MindIE Server Pod调用生成证书接口如出现“failed to read random number from system.”报错，大概率是由于环境熵不足，需要在计算节点安装haveged组件补熵。详情请参考《**MindIE安装指南**》中“附录 > 启动haveged服务”章节，将熵补至4096。

准备 Deployer msctl 命令行客户端

工具准备

以当前所在路径为/home/*{用户名称}*/package为例进行以下操作。

步骤1 使用以下命令将Ascend-mindie-service_*{version}*_linux-aarch64.run安装后的msctl.json文件拷贝至*{$HOME}*/.mindie_ms/路径下， 且文件权限不高于640。

```
cp mindie-service/latest/conf/msctl.json ~/.mindie_ms/
```

步骤2 打开msctl.json文件并根据实际情况参考**表3-8**修改启动配置文件，配置样例如下所示（格式必须和样例保持一致）。

```
vi ~/.mindie_ms/msctl.json
```

msctl.json文件信息如下所示：

```
{
    "http":{
        "dstPort" : 9789,
        "dstIP" : "127.0.0.1",
        "ca_cert": "./security/ca/ca.pem",
        "tls_cert": "./security/certs/cert.pem",
        "tls_key": "./security/keys/cert.key.pem",
        "tls_crl": "",
        "timeout": 60
    },
    "log_level": "INFO"
}
```

表 3-8 启动配置参数

参数	类型	说明
dstPort	Int	必填。 对端端口（ MindIE MS服务端），与ms_server.json中的port保持一致。 取值范围：[1024，65535]，默认值9789。
dstIP	String	必填。 部署服务端IP地址，Deployment部署场景使用MindIE MS Server的Pod IP进行访问 。 可通过以下命令查询Pod IP： `kubectl get pods -n mindie -o wide`
ca_cert	String	必填。 默认使用HTTPS通信。 ca根证书文件。需MindIE MS服务端ca证书文件相互信任。

参数	类型	说明
tls_cert	String	必填。 客户端tls证书文件。用户请参考下面步骤2准备证书。
tls_key	String	必填。 客户端tls私钥文件。出于安全的考虑，建议使用加密私钥，如使用，在执行命令时会提供用户输入加密口令。
tls_crl	String	必填。 校验通信对端服务端的证书吊销列表crl文件，如为空，则不进行吊销校验。
timeout	Int	必填。 MindIE MS客户端可设置的等待响应时长，取值范围[1,60]。
log_level	String	必填。 客户端日志级别。 • DEBUG • INFO • WARNING • ERROR • CRITICAL

步骤3 进入到infer_server.json配置文件所在路径并打开。

```
cd /mindie-service/latest/conf
vi infer_server.json
```

步骤4 将infer_server.json配置文件的“server_type”参数改为“mindie_cross_node”，infer_server.json配置文件如下所示，其参数解释请参见表3-9。

```
{
  "server_name": "mindie-server",
   "replicas": 1,
   "service_port": 31005,
   "cross_node_num":2,
   "service_type": "NodePort",
   "server_type":"mindie_cross_node",
   "scheduler": "default",
   "init_delay": 180,
   "liveness_timeout": 10,
   "liveness_failure_threshold": 1,
   "readiness_timeout": 10,
   "readiness_failure_threshold": 1,
   "resource_requests": {
       "memory":   500000,
       "cpu_core" : 32000,
       "npu_type": "Ascend910",
       "npu_chip_num": 8
   },
   "mindie_server_config": {
       "mies_install_path": "/usr/local/Ascend/mindie/latest/mindie-service",
       "infer_port" : 1025,
```



```
        "management_port":1026,
        "enable_tls":true
    },
    "npu_fault_reschedule": true
    "termination_grace_period_seconds": 30,
    "max_unavailable": "50%",
    "max_surge": "50%"
}
```

表 3-9 infer_server.json 配置文件参数解释

参数	类型	说明
server_name	String	必填。 任务服务名称。在MS管理范围内具备唯一性的ID，当前只支持部署一个服务。
replicas	Int	必填。 部署的实例副本数量。 ● 分布式多机场景当前只支持部署一个实例。 ● 单机场景可部署多个实例，范围为[1, 256]，具体配置数量根据物理资源进行配置。
service_port	Int	必填。 部署服务对外可被用户访问的端口，NodePort取值范围[30000, 32767]。 访问MindIE推理请求时需要使用该接口。
cross_node_num	Int	分布式多机场景必填，**单机场景不支持该参数**。 一个实例跨机部署的节点个数，当前只支持取值为2。
service_type	String	必填。 创建Kubernetes的Service资源类型，当前只支持NodePort类型。 NodePort：表示从该参数获取Service的资源类型。
server_type	String	必填。 服务部署的形态，支持mindie_cross_node为多机部署和mindie_single_node单机部署。
scheduler	String	必填。 可选调度器类型，分布式多机场景仅支持default类型，单机场景支持default和volcano类型；如需启用NPU故障重调度功能，需要选择“volcano”调度器。 ● default：使用Kubernetes默认调度器。 ● volcano：使用MindCluster Volcano调度器。

参数	类型	说明
init_delay	Int	必填。 允许推理实例启动的时间。 ● 分布式多机场景，从Pod启动后ranktable完成生成的时间开始计算，在这个时间内MindIE MS会周期性检测服务实例是否就绪，超过这个时间服务未就绪，实例将自动重启。 ● 单机场景，从pod启动的时间计算，超过这个时间，存活探针开始接管，若live检测失败（5秒触发一次），会触发重启。 取值范围为[10, 1800]， 单位为秒。
liveness_timeout	Int	必填。 存活探针超时时间设置，取值范围[1, 600]，单位秒。 注意在高并发场景下，liveness探针可能超时，此时认为MindIE Server状态异常，并对MindIE Server Pod进行重启。建议用户根据业务并发度配置合理的值，可以使用MindIE Benchmark工具设置并发量发送请求进行模拟测试，确保Pod在并发度条件下没有发生重启。 参考示例：llama2-7b在Atlas 800I A2 推理产品上单卡单实例部署，使用MMLU数据集将Benchmark并发度设置为1000，超时时间配置为10秒。
liveness_failure_threshold	Int	必填。存活探针最大允许的失败次数，取值范围[1, 10]。 参考示例：llama2-7b在Atlas 800I A2 推理产品上单卡单实例部署，使用MMLU数据集将Benchmark并发度设置为1000，失败次数设置为2时Pod不重启。
readiness_timeout	Int	必填。就绪探针超时时间设置，取值范围[1, 600]，单位秒。 就绪探针超时时间配置不足且当请求并发量大时，可能导致就绪探针超时，超时后Service状态变成“未就绪”，无法正常发送推理请求。
readiness_failure_threshold	Int	必填。就绪探针最大允许的失败次数，建议配置为1。 取值范围[1, 10]。
memory	Int	必填。 服务所需最少内存资源大小，单位为M。 默认值256000，取值范围[1000, 256000]。

参数	类型	说明
cpu_core	Int	必填。 服务所需最少CPU资源大小，取值范围[1000, 256000]，单位为M（1000M等于1核心）。
npu_type	String	必填。 服务所需NPU卡类型。 • 分布式多机场景：支持输入"Ascend910"。 • 单机场景：支持输入"Ascend910"和"Ascend310P"。
npu_chip_num	Int	必填。 单个节点所需NPU卡的数量。规格限制[1, 8]； 单机场景下，是一个实例占用的卡数； 分布式多机场景下，是一个实例在每个节点上占用的卡数，会分配匹配的卡资源并写入ranktable。且只支持全卡推理，当前仅支持配置为8卡的设备。
mindie_server_config	Object	必填。 • infer_port：推理服务的推理端口。 • management_port：推理服务的管理端口，用于查询服务状态。 • enable_tls：是否开启HTTPS，使用服务端启动时，使用服务端启动时，配置的证书发送tls请求需要与MindIE Server的**config.json配置文件**中httpsEnabled参数、**表8-16**中client_mindie_server_tls_enable参数的值保持一致。 • mies_install_path： MindIE Service在镜像内的安装路径。
npu_fault_reschedule	bool	scheduler参数为volcano时必填，**该参数仅对单机场景生效**。 • true：表明使用优雅方式重调度，并在过程中先优雅删除原Pod。 • false：表明使用非优雅方式重调度，配置任务采用强制删除模式，在过程中强制删除原Pod。

参数	类型	说明
termination_grace_period_seconds	Int	Pod优雅退出时间，**单机场景必填，分布式多机场景不支持**。 设置Pod优雅退出最大的容忍时间，超过该时间K8s会强制杀掉服务。滚动更新过程，若设置时间不足，可能导致剩余请求未处理完中断。 参考配置：以llama2-7b模型实例数2为例，在Atlas 300I Duo 推理卡+Atlas 800 推理服务器（型号：3000）单卡设备使用gsm8k数据集并发度为1000的请求，需要配置Pod优雅退出时间为800秒，保证Pod退出过程业务不中断。 取值范围[0, 3600]；单位为秒。 在服务启动初始，未就绪时，此时服务无法正常处理优雅退出信号，删除服务将无法触发优雅退出，需等待程序自动退出或达到优雅退出的最大容忍时间。
max_unavailable	String	滚动更新配置，**单机场景必填，分布式多机场景不支持**。 更新所需的可以处于不可用状态的最大 Pod 数量，指的是replicas * max_unavailable向下取整。 max_unavailable格式必须是“{int数字}%”，取值范围0%~100%。
max_surge	String	滚动更新配置，**单机场景必填，分布式多机场景不支持**。 更新所需的可以超出期望副本数的最大Pod数量，指replicas * max_surge向上取整。 max_surge格式必须是“{int数字}%，取值范围0%~100%。

步骤5 使用以下命令进入mindie-service的安装路径。

```
cd ./mindie-service/latest
```

步骤6 导入**签发其他证书**中MindIE MS客户端msctl与MindIE MS服务端通信证书（即CERT_CN取值为msclientuser签发的证书）。

在当前路径下创建证书目录，导入中间CA证书、MindIE MS客户端msctl与MindIE MS服务端通信的证书和加密私钥。

```
mkdir -m 700 ./security
mkdir -m 700 ./security/certs
mkdir -m 700 ./security/keys
mkdir -m 700 ./security/pass
python3 ./scripts/config_mindie_server_tls_cert.py ./ import_ca /etc/kubernetes/pki/ca.crt  #/etc/kubernetes/pki/ca.crt为：Kubernetes根CA证书文件路径
python3 ./scripts/config_mindie_server_tls_cert.py ./ import_cert /home/{用户名称}/cas/msclientuser/cert.pem /home/{用户名称}/cas/mindiems/cert.pem  #/home//{用户名称}cas/msclientuser/cert.pem为：MindIE MS客户端msctl与MindIE MS服务端通信的证书；/home/{用户名称}/cas/mindiems/cert.pem为：中间CA证书文件路径
```


步骤7 执行以下命令导入环境变量，将所依赖的so载入LD_LIBRARY_PATH。

```
export LD_LIBRARY_PATH=/path/to/mindie-service/latest/lib:$LD_LIBRARY_PATH
```

----结束

使用 msctl 部署服务

工具使用

MindIE MS deployer提供命令行工具，可供用户使用，当前提供的对外命令为：

表 3-10 客户端命令

指令	类型	说明
./bin/msctl create infer_server -f ./conf/infer_server.json	create	客户端服务部署。 配置文件需要不大于640权限，否则启动失败。
./bin/msctl delete infer_server -n *{服务名称}*	delete	卸载服务，删除服务相关的Kubernetes资源，停止对外服务。
./bin/msctl get infer_server -n *{服务名称}*	get	获取服务状态，包括服务是否已就绪。
./bin/msctl -h/--help	-	命令使用帮助。

客户端部署命令

使用msctl在MindIE MS客户端发送部署请求命令，如下所示。

```
./bin/msctl create infer_server -f ./conf/infer_server.json
```

infer_server.json：**步骤4**中用户配置的infer_server.json配置文件。

回显如下所示，则表示部署请求命令下发成功。

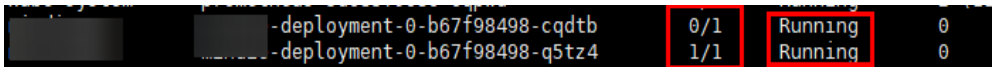
```
{
   "message": "creating the server!",
   "status": "0"
}
```

执行以下Kubernetes查询指令，查询是否部署成功。

```
kubectl get pod -A
```

如下图所示，在Kubernetes的Pod资源用户自定义的*{namespace}*中，存在*{server_name}*-deployment-0-*xxx*且状态为Running时，则表示部署成功。

*{server_name}*和*{namespace}*为**8.3.2.3 RESTful接口API**中用户自行编写的json配置文件中的“server_name”和“namespace”参数值。



说明

如果部署失败，请参见表3-4中的volumes参数，根据挂载的物理日志路径中的日志文件定位具体问题。

客户端卸载命令

使用msctl在MindIE MS客户端发送卸载请求命令，如下所示。

```
./bin/msctl delete infer_server -n {server_name}
```

回显如下所示，则表示卸载请求命令下发成功且服务删除成功。

```
{
   "message": "succeed to clear resources",
   "status": "0"
}
```

也可以使用Kubernetes命令查看该服务是否已删除或者处于非Running状态，如下图所示。

```
kubectl get pod -A
```

被删除的服务进入Terminating状态10秒后，再次查看部署服务状态，发现该服务已从列表中消失，即可开始部署新的分布式多机任务。

说明

- 卸载服务任务需等待服务彻底被删除后再进行分布式多机服务的部署。
- 系统出现异常不能通过MindIE MS客户端命令进行时，请参见**9.8 使用MindIE MS部署多机任务后系统出现异常，不能通过MindIE MS客户端卸载**进行处理。

客户端查询命令

使用msctl在MindIE MS客户端发送查询请求命令，如下所示。

```
./bin/msctl get infer_server -n {server_name}
```

回显如下所示，则表示查询请求命令下发成功。

```
{
   "data": {
      "instances_status": [
         {
            "instance_id": 0,
            "liveness": true,
            "readiness": true,
            "restore_state": "none"
         }
      ],
      "model_info": {
         "docker_label": null,
         "max_batch_total_tokens": 8192,
         "max_best_of": 1,
         "max_concurrent_requests": 200,
         "max_input_length": 2048,
         "max_stop_sequences": null,
         "max_waiting_tokens": null,
         "models": [
            {
               "max_total_tokens": 2560,
               "model_device_type": "npu",
```



```
                "model_dtype": "float16",
                "model_id": "llama_65b",
                "model_pipeline_tag": "text-generation",
                "model_sha": null
            }
        ],
        "sha": null,
        "validation_workers": null,
        "version": "1.0.RC3",
        "waiting_served_ratio": null
    },
    "server_name": "mindie-server",
    "server_status_msg": "created: succeed to create the server, and succeed to save server status to file"
  },
  "message": "success",
  "status": "0"
}
```


重要参数解释：

- “liveness”：表示服务存活状态。
  - “true”：表示服务存活。
  - “false”：表示服务未存活。
- “readiness”：表示服务实例启动状态。
  - “true”：表示服务实例已启动完成并进入就绪状态。
  - “false”：表示服务实例未启动完成。
- “server_status_msg”：表示服务状态。
  - “creating”：表示服务正在创建中。
  - “created”：表示服务已创建完成。
  - “failed”：表示服务创建失败。
  - “stopping”：表示服务正在停止中。

说明

后续用户使用MindIE Server推理能力输入**EndPoint RESTful接口使用说明**指令时，只需要将*{port}*设置为**表3-9**中service_port参数的值，*{ip}*设置为Kubernetes部署管理节点IP即可。

3.6 PD 分离服务部署

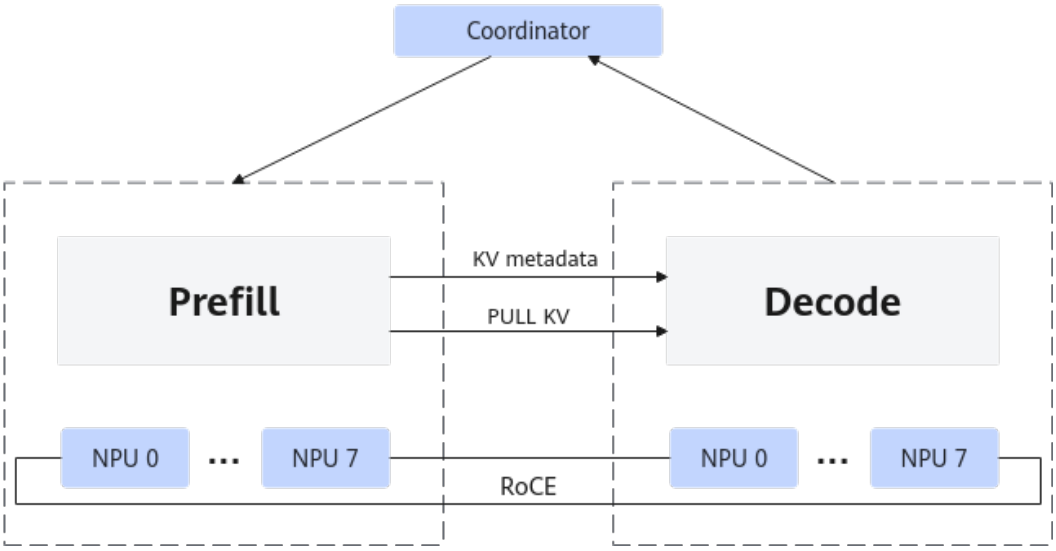
3.6.1 场景介绍

3.6.1.1 概述

PD 分离介绍

Prefill & Decode分离部署（简称：PD分离部署）是将Prefill（预填充）和Decode（解码）这两个推理阶段分开处理的技术，通常适用于对时延有严格要求的场景。PD分离部署可以提高NPU的利用率，尤其是大语言模型时，通过将Prefill实例和Decode实例分开部署，减少Prefill阶段和Decode阶段分时复用在时延上造成的互相干扰，实现同时延下吞吐提升。PD分离工作原理如**图3-14**所示。

图 3-14 PD 分离工作原理



大语言模型推理的阶段可以分为Prefill与Decode阶段：

- Prefill阶段：在生成式语言模型中，Prefill阶段涉及到模型对初始提示（Prompt）的处理，生成初始的隐藏状态（Hidden States）。这个阶段通常涉及对整个模型的一次前向传播，因此计算密集度较高。对于每个新的输入序列，都需要进行一次Prefill。
- Decode阶段：在Prefill阶段之后，模型基于初始隐藏状态逐步生成后续的文本。这一阶段的特点是计算相对较少，但需要反复进行计算，直到生成足够的文本或达到某个终止条件。在生成过程中，只计算最新的token激活值，并进行attention计算，计算最终的预测token。

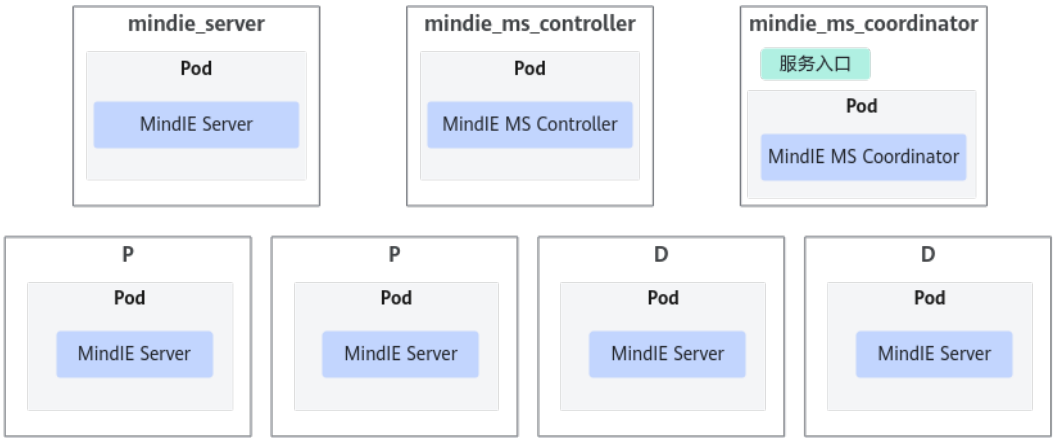
部署方案

PD分离部署在线下K8s集群的参考部署方案：

通过创建3个K8s的Deployment分别部署MindIE MS Controller（单Pod副本）、MindIE MS Coordinator（单Pod副本）以及MindIE Server（多Pod副本）。

通过K8s的Service为Coordinator Pod开放PD集群的推理入口。

图 3-15 PD 分离的部署形态



PD 分离优势

PD分离主要包括以下优势：

- 资源利用优化：由于Prefill阶段计算密集，而Decode阶段计算较为稀疏，将这两个阶段分离可以更好的利用NPU的计算资源。
- 提高吞吐量：分离后的Prefill和Decode可以同时处理不同的请求，这意味着在Prefill阶段处理新请求的同时，Decode阶段可以继续处理之前请求的解码任务，从而提高了整体的处理能力。
- 降低延迟：由于Prefill和Decode分别在不同的阶段进行，可以减少等待时间，特别是当有多个请求并发到达时。

限制与约束

- 仅Atlas 800I A2 推理产品支持此特性。
- P节点与D节点仅支持相同型号的机型。
- 不同P、D节点使用的NPU卡数量必须相同。
- NPU网口互联（带宽：200Gbps）。

说明

特性兼容性：MindIE RC3版本PD分离场景不支持停止词特性下不返回stop string功能，即不支持include_stop_str_in_output=false。

3.6.1.2 硬件环境

PD分离部署支持的硬件环境如表3-11所示。

表 3-11 PD 分离部署支持的硬件列表

类型	型号	内存
服务器	Atlas 800I A2 推理产品	• 32GB • 64GB

说明

集群必须具备参数面互联：即服务器NPU卡对应的端口处在同一个VLAN，可以通过RoCE互通。

3.6.2 配置说明

表 3-12 环境变量列表

环境变量	类型	说明
MINDIE_INFER_MODE	PD分离	推理模式，表示是否PD分离。 • standard：PD混部； • dmi：PD分离。

环境变量	类型	说明
MINDIE_DECODE_BATCH_SIZE	公共变量	最大Decode的batch大小。 取值范围：[1, 5000]
MINDIE_PREFILL_BATCH_SIZE	公共变量	最大Prefill的batch大小。 取值范围：[1, MINDIE_DECODE_BATCH_SIZE - 1]
MINDIE_MAX_SEQ_LEN	公共变量	最大序列长度。 整型数字，取值范围：(0, 4294967295]
MINDIE_MAX_ITER_TIMES	公共变量	最大输出长度。 整型数字，取值范围：[1, MINDIE_MAX_SEQ_LEN-1]
MINDIE_MODEL_NAME	公共变量	模型名。
MINDIE_MODEL_WEIGHT_PATH	公共变量	模型权重文件路径。
MINDIE_ENDPOINT_HTTPS_ENABLED	公共变量	是否在Prefill/Decode实例上启用HTTPS。 ● true：启用； ● false：禁用。
MINDIE_INTER_COMM_TLS_ENABLED	公共变量	推理实例间通信开启TLS。 ● true：启用； ● false：禁用。
MINDIE_LOG_TO_FILE	公共变量	日志是否打印到文件，默认值为1。 ● 0：不打印到文件； ● 1：打印到文件。
MINDIE_LOG_TO_STDOUT	公共变量	日志是否打印到标准输出，默认值为0。 ● 0：不打印到标准输出； ● 1：打印到标准输出。
MINDIE_MS_CONTROLLER_CONFIG_FILE_PATH	公共变量	MindIE MS Controller组件配置文件路径。
MINDIE_MS_COORDINATOR_CONFIG_FILE_PATH	公共变量	MindIE MS Coordinator组件配置文件路径。

表 3-13 关键配置文件

配置文件	参数	取值
ms_controller.json	deploy_mode	pd_separate：表示PD分离模式。详情请参见**8.3.3.3 配置说明**。
ms_coordinator.json	deploy_mode	pd_separate：表示PD分离模式；详情请参见**8.3.4.3 配置说明**。

3.6.3 安装部署

3.6.3.1 准备镜像

3.6.3.1.1 MindIE 与 ATB Models 配套使用场景

请参见**3.3 准备MindIE Server镜像**制作MindIE Server镜像。

3.6.3.1.2 MindIE 与 MindSpore 配套使用场景

1. 详情请参见《**MindIE LLM开发指南**》的“模型推理使用流程 > MindSpore Models使用”章节，制作支持MindSpore后端的MindIE容器。
2. 将**1**制作的容器通过以下命令保存为镜像。
   ```
   docker commit container_id image_id:tag
   ```

3.6.3.2 基于 Kubernetes 部署

3.6.3.2.1 软件环境准备

MindIE MS的安装依赖Kubernetes和MindCluster组件，Kubernetes的安装与配置和MindCluster组件安装详情请参见**3.2 环境准备**章节。

3.6.3.2.2 配置自动生成证书

若用户开启MindIE Server的TLS认证功能（HTTPS或GRPC）时，通信客户端需要校验服务端证书的IP，由于PodIP的动态性，需要在Pod启动时生成具有PodIP别名的服务证书，以实现MindIE Server中PD节点间的通信，以及MindIE MS对MindIE Server的证书认证和校验。MindIE提供证书生成能力，具体操作步骤如下所示。

须知

建议用户在运行环境中的各个计算节点准备和配置证书，提升服务安全性。

操作步骤

此方法只适用于使用自签名CA证书进行证书签发的场景。

需要按照以下方法准备MindIE Server、MindIE Controller和MindIE Coordinator三套证书。

步骤1 准备自签名CA证书和加密私钥及导入。

准备MindIE Server服务端数据面和管理面的CA证书和加密私钥

1. 执行以下命令生成配置文件。
   ```
   cat > ca.conf <<-EOF
   ```
   配置文件ca.conf示例如下，其中req_distinguished_name中的字段需要自行配置：
   ```
   [ req ]
   distinguished_name     = req_distinguished_name
   prompt                 = no

   [ req_distinguished_name ]
   C                      = CN
   ST                     = Sichuan
   L                      = Chengdu
   O                      = Huawei
   OU                     = Ascend
   CN                     = MindIE

   [ v3_ca ]
   subjectKeyIdentifier = hash
   authorityKeyIdentifier = keyid:always,issuer
   basicConstraints = critical, CA:true
   keyUsage = critical, digitalSignature, cRLSign, keyCertSign
   EOF
   ```
2. 执行以下命令创建格式为PKCS#1的PKI私钥ca.key.pem。
   ```
   openssl genrsa -aes256 -out ca.key.pem 4096
   ```
3. 根据回显输入私钥口令，然后按回车键。
   ```
   Enter pass phrase for ca.key.pem:
   Verifying - Enter pass phrase for ca.key.pem:
   ```
   出于安全考虑，以及后续导入证书的要求，用户输入的私钥口令的复杂度必须符合以下要求：
   - 口令长度至少8个字符；
   - 口令必须包含如下至少两种字符的组合：
     - 至少一个小写字母；
     - 至少一个大写字母；
     - 至少一个数字；
     - 至少一个特殊字符。
4. 执行以下命令赋予ca.key.pem私钥文件可读权限。
   ```
   chmod 400 ca.key.pem
   ```
5. 执行以下命令检查是否存在ca.key.pem私钥文件，并查看私钥内容。
   ```
   openssl rsa -in ca.key.pem
   ```
   根据回显输入**步骤1.3**设置的私钥口令，然后按回车键，当显示私钥内容时表示ca.key.pem私钥文件生成成功。
6. 执行以下命令创建CSR文件，根据回显输入**步骤1.3**设置的私钥口令，然后按回车键。
   ```
   openssl req -out ca.csr -key ca.key.pem -new -config ca.conf -batch
   ```
7. 执行以下命令赋予ca.csr文件可读可写权限。
   ```
   chmod 600 ca.csr
   ```


8. 执行以下命令检查是否存在ca.csr文件，当显示ca.csr文件内容表示ca.csr文件生成成功。
```
openssl req -in ca.csr -noout -text
```
9. 执行以下命令生成CA证书ca.pem。
```
openssl x509 -req -in ca.csr -out ca.pem -sha256 -days 7300 -extfile ca.conf -extensions v3_ca -signkey ca.key.pem
```
10. 执行以下命令检查是否存在ca.pem文件，当有回显内容时表示ca.pem生成成功。
```
openssl x509 -in ca.pem -noout -text
```
11. 执行以下命令赋予ca.pem文件可读权限。
```
chmod 400 ca.pem
```

导入自签名CA证书和加密私钥

1. 使用以下命令进入mindie-service安装目录。
```
cd /{MindIE安装目录}/latest/mindie-service/
```
2. 通过MindIE证书管理工具import_cert接口导入CA证书和私钥，输入证书私钥口令并生成KMC加密口令文件和KMC密钥库文件。MindIE证书管理工具详情请参见**8.1.2.1 config_mindie_server_tls_cert.py**。
```
python3 ./scripts/config_mindie_server_tls_cert.py ./security/ca import_cert {证书文件路径} {加密私钥文件路径}
```
参数解释：
   - *{证书文件路径}*：为CA证书的源路径。
   - *{加密私钥文件路径}*：为CA私钥的源路径。

   在回显时输入生成CA密钥时设置的口令：
```
Password for private key file:
Retype password for private key file:
```

步骤2 准备生成证书的输入输出配置文件。

- 生成用户证书的配置文件（gen_cert.json）：
```
{
    "ca_cert": "./security/ca/ca.pem",
    "ca_key": "./security/ca/ca.key.pem",
    "ca_key_pwd": "./security/ca/ca_passwd.txt",
    "cert_config": "./cert_info.json",
    "output_path": "./gen_cert_output",
    "kmc_ksf_master": "./tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./tools/pmt/standby/ksfb"
}
```
- 配置"cert_config"参数中的cert_info.json配置文件的待生成证书信息：
```
{
    "subject": "subject_name",
    "expired_time": 3650,
    "serial_number": 123,
    "req_distinguished_name": {
        "C": "***",
        "ST": "***",
        "L": "***",
        "O": "***",
        "OU": "***",
        "CN": "***"
    },
    "alt_names": {
        "IP": [],
        "DNS": []
    }
}
```

步骤3 在**脚本介绍**的mindie_server.yaml、mindie_ms_controller.yaml和mindie_ms_coordinator.yaml配置文件中挂载上述自签名CA证书文件和配置文件到容器内/mnt/security目录，并配置为只读权限。

步骤4 在**脚本介绍**的容器启动脚本boot.sh中适配添加证书生成命令，以生成MindIE Server的证书为例，在“if [$exit_code -eq 2]; then”分支内添加以下生成证书的命令。

```
cp /mnt/security/ca.pem $MIES_INSTALL_PATH/security/ca
cp /mnt/security/ca.key.pem $MIES_INSTALL_PATH/security/ca
cp /mnt/security/ca_passwd.txt $MIES_INSTALL_PATH/security/ca
cp /mnt/security/gen_cert.json $MIES_INSTALL_PATH
cp /mnt/security/cert_info.json $MIES_INSTALL_PATH
cp -r /mnt/security/tools $MIES_INSTALL_PATH/
chmod 500 ./bin/gen_cert
mkdir gen_cert_output
python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_cert.json  --ip=$MIES_CONTAINER_IP,{host_ip}
chmod 400 ./gen_cert_output/*
// 拷贝生成的证书到特定的路径
cp ./gen_cert_output/cert.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/certs/server.pem
cp ./gen_cert_output/cert.key.pem /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/keys/server.key.pem
cp ./gen_cert_output/cert_passwd.txt /home/{用户名称}/Ascend/mindie/latest/mindie-service/security/pass/mindie_server_key_pwd.txt
rm -rf ./gen_cert_output/*
// 下面使用其他证书配置(gen_cert_xxx.json, cert_info_xxx.json)重复上述步骤继续导入其他证书
// cp /mnt/security/gen_cert_xxx.json $MIES_INSTALL_PATH
// cp /mnt/security/cert_info_xxx.json $MIES_INSTALL_PATH
// python3 ./scripts/config_mindie_server_tls_cert.py  ./  gen_cert ./gen_cert_xxx.json  --ip=$MIES_CONTAINER_IP,{host_ip}
```

{host_ip}：仅MindIE MS调度器（Coordinator）需要配置，配置为提供推理API的物理机IP。

说明

- MindIE Server、MindIE MS控制器（Controller）和MindIE MS调度器（Coordinator）三套证书准备完成后，请参考**MindIE Server配置说明**、**MindIE MS控制器（Controller）配置说明**和**MindIE MS调度器（Coordinator）配置说明**将每个证书拷贝至指定的路径下。
- 启动MindIE Server Pod调用生成证书接口如出现“failed to read random number from system.”报错，大概率是由于环境熵不足，需要在计算节点安装haveged组件补熵。详情请参考《**MindIE安装指南**》中“附录 > 启动haveged服务”章节，将熵补至4096。

----结束

3.6.3.2.3 使用 kubectl 部署 PD 分离服务示例

脚本介绍

本节使用mindie-service安装目录（examples/kubernetes_deploy_scripts）中的脚本实现一键式部署和卸载MindIE PD分离集群功能，集群管理员用户可参考这些脚本文件线下使用K8s kubectl工具操作集群。

集群管理员用户只需在管理节点完成启动脚本编写、业务配置和kubernetes配置，然后调用部署脚本，实现自动下发业务配置和启动脚本，自动全局ranktable生成，以及自动调度Pod到计算节点。

脚本文件目录结构如下所示：

```
├── boot_helper
│   ├── boot.sh
│   ├── get_group_id.py
│   └── update_mindie_server_config.py
├── chat.sh
├── conf
├── delete.sh
```



```
├── deployment
│   ├── mindie_ms_controller.yaml
│   ├── mindie_ms_coordinator.yaml
│   ├── mindie_server_heterogeneous.yaml
│   └── mindie_server.yaml
├── deploy.sh
├── generate_stream.sh
├── gen_ranktable_helper
│   ├── gen_global_ranktable.py
│   └── global_ranktable.json
└── log.sh
```

关键目录及文件解释如下所示：

- conf：MindIE MS和MindIE Server的主要业务配置文件，PD分离管理调度策略和模型相关配置。
- boot_helper：包含容器启动脚本boot.sh，获取group_id，刷新环境变量到配置文件，设置启动程序的环境变量等，用户可根据需要在这里调整日志等级等，环境变量详细介绍请参见**表3-12**。
- deployment: K8s部署任务定义，配置NPU资源使用量，实例数，镜像等。
- gen_ranktable_helper：生成global ranktable的工具，用户无需感知。
- chat.sh：使用curl发送HTTP请求给推理服务的简单对话示例，**适用于Prefix Cache场景**。
- generate_stream.sh：使用curl发送HTTP请求给推理服务的流式响应示例。
- deploy.sh：部署入口脚本，一键拉起所有MindIE组件。
- delete.sh：卸载脚本，一键卸载所有MindIE组件。
- log.sh：查询Pod的打屏日志，可查询到部署的所有Pod的日志。

操作步骤

以llama3-8b模型为例，每个实例配置2张卡，配置4个实例。部署样例如下所示，以下操作均在部署脚本路径下完成：

步骤1 进入集群管理节点宿主机，首次部署用户需创建mindie命名空间。

```
kubectl create namespace mindie
```

步骤2 在宿主机上将mindie-service发布件run包安装到用户指定目录，取出其中conf目录的config.json、ms_controller.json、ms_coordinator.json和http_client_ctl.json文件拷贝到脚本conf目录。

步骤3 配置MindIE MS Controller组件的启动配置文件ms_controller.json，其配置文件详细说明请参见**8.3.3.3 配置说明**。

PD场景下需配置为PD分离部署模式：

```
deploy_mode="pd_separate"
```

说明

配置文件ms_controller.json中的“default_p_rate”和“default_d_rate”参数分别控制集群中P节点和D节点数量的比值，默认均为0，根据模型、硬件、服务信息等自动决策最佳配比，也可以根据场景分别设置为P和D节点的实际数量。

步骤4 配置MindIE MS Coordinator组件的启动配置文件ms_coordinator.json，其配置文件详细说明请参见**8.3.4.3 配置说明**。

PD场景下需配置为PD分离部署模式：


```
deploy_mode="pd_separate"
```

步骤5 配置MindIE Server服务启动的config.json配置文件，PD分离服务部署模式需要配置以下参数，具体参数解释请参见**8.4.2 配置参数说明**。

- modelName：模型名配置，关联模型权重文件的模型，如配置为llama3-8b。
- modelWeightPath：模型权重文件目录配置，默认情况下脚本会挂载物理机的/data目录，modelWeightPath需配置为/data路径下的模型权重路径，确保集群可调度Ascend计算节点在该路径下存在模型文件。
- worldSize：配置一个P/D实例占用的NPU卡数；例如配置为“2”，表示使用两张卡。
- npuDeviceIds：卡号配置成从0开始编号，总数与worldSize一致，如配置为[[0,1]]。
- inferMode：配置为dmi。

步骤6 配置http_client_ctl.json配置文件，该配置文件为集群启动、存活、就绪探针HTTP客户端工具的配置文件，具体参数解释请参见**表6-4**。

“tls_enable”参数为控制是否使用HTTPS的开关，若集群内MindIE组件使用了HTTPS接口，需设置“tls_enable”为“true”，并导入证书到容器内，配置相应的证书路径。如使用HTTP接口，则设置“tls_enable”为“false”，无需准备证书文件。

须知

建议用户打开tls_enable，确保通信安全；如果关闭则存在较高的网络安全风险。

步骤7 配置kubernetes Deployment。

在部署脚本目录中的deployment目录下找到mindie_server.yaml、mindie_ms_coordinator.yaml和mindie_ms_controller.yaml文件。

须知

用户在使用kubetl部署deployment时，可修改deployment的配置yaml文件，请避免使用危险配置，确保使用安全镜像(非root权限用户)，配置Pod安全上下文，挂载安全路径(非软链接，非系统危险路径，非业务敏感路径)并设置了安全的文件目录权限。本脚本仅作为一个部署参考，Pod容器的安全性由用户自行保证，实际生产环境请针对镜像和Pod安全进行加固。

- mindie_server.yaml主要配置的字段如下所示：
  - replicas： 配置P和D的总实例数，4个实例即配置为4。
  - huawei.com/Ascend910：配置一个P/D实例占用的NPU卡数，与MindIE Serve的config.json配置文件中worldSize参数配置的卡数保持一致。
  - image：配置镜像名。
    - MindIE与ATB Models配套使用场景：镜像名请参见**3.6.3.1.1 MindIE与ATB Models配套使用场景**。
    - MindIE与MindSpore配套使用场景：镜像名请参见**3.6.3.1.2 MindIE与MindSpore配套使用场景**。

  - livenessProbe：存活探针，当服务启动时间较长时，则需要延迟存活检测开始时间，配置initialDelaySeconds为合理值。
  - nodeSelector：节点选择，配置期望调度的节点，通过节点标签实现。
- mindie_ms_coordinator.yaml和mindie_ms_controller.yaml文件主要关注image镜像名的修改。
  - MindIE与ATB Models配套使用场景：镜像名请参见**3.6.3.1.1 MindIE与ATB Models配套使用场景**。
  - MindIE与MindSpore配套使用场景：镜像名请参见**3.6.3.1.2 MindIE与MindSpore配套使用场景**。

  说明

  另外，mindie_ms_coordinator.yaml文件还需特别关注livenessProbe（存活探针）参数，在高并发推理请求场景下，可能会因为探针超时，从而被K8s识别为Pod不存活，导致K8s重启Coordinator容器，建议用户谨慎开启livenessProbe（存活探针）。

步骤8 配置启动脚本boot.sh。

MindIE Server环境变量配置为PD部署模式：

```
export PD_MODE=0
```

步骤9 拉起PD集群。

配置容器内mindie安装的目录：根据制作镜像时实际的安装路径，修改MINDIE_USER_HOME_PATH的value值，如安装路径是/xxx/Ascend/mindie, 则配置为/xxx 。

```
export MINDIE_USER_HOME_PATH={镜像的安装路径}
```

使用以下命令拉起集群。

```
bash deploy.sh
```

执行命令后，会同步等待global_ranktable.json生成完成，如长时间处于阻塞状态，请ctrl+c中断后查看集群Pod状态，进行下一步的调试定位。

global_ranktable.json样例如下所示，样例中参数解释如**表3-14**所示。

```
{
  "version": "1.0",
  "status": "completed",
  "server_group_list": [
    {
      "group_id": "0",
      "server_count": "1",
      "server_list": [
        {
          "server_id": "coordinator",
          "server_ip": "xxx.xxx.xxx.1"
        }
      ]
    },
    {
      "group_id": "1",
      "server_count": "1",
      "server_list": [
        {
          "server_id": "controller",
          "server_ip": "xxx.xxx.xxx.1"
        }
      ]
    },
    {
```



```
    "group_id": "2",
    "server_count": "2",
    "server_list": [
     {
       "server_id": "server",
       "server_ip": "xxx.xxx.xxx.1",
       "device": [
        {
          "device_id": "0",
          "device_ip": "xxx.xxx.xxx.1"
        }
       ]
     },
     {
       "server_id": "server",
       "server_ip": "xxx.xxx.xxx.2",
       "device": [
        {
          "device_id": "1",
          "device_ip": "xxx.xxx.xxx.2"
        }
       ]
     }
    ]
  }
 ]
}
```

表 3-14 global_ranktable.json 文件参数解释

参数	类型	描述
version	string	HCCL-Controller的版本号。
status	string	集群信息表的状态。 ● completed：部署完成。 ● initializing：初始化中。
group_id	string	各组件的ID。 ● 0：存储Coordinator的部署信息。 ● 1：存储Controller的部署信息。 ● 2：存储Server的部署信息。
server_count	string	各组件的节点总数。
server_list	json对象数组	各组件的节点部署信息。 ● 最多包含1个Coordiantor实例，列表有效长度[0, 1] ● 最多包含96个Server实例，列表有效长度[0, 96]
server_id	string	组件节点的主机ID。
server_ip	string	组件节点的IP地址。
device	json对象数组	NPU设备信息，仅MindIE Server有此属性。列表有效长度[1, 128]。

参数	类型	描述
device_id	string	NPU的设备ID，即Server所在Pod内可见的卡设备的序列ID。
device_ip	string	NPU的IP地址。

说明

- 如果部署失败，则需要参见**步骤12**卸载集群后重新部署。
- 集群默认刷新挂载到容器内的configmap的频率是60s，如遇到容器内打印“status of ranktable is not completed”日志信息的时间偏久，可在每个待调度的计算节点修改kubelet同步configmap的周期，即修改/var/lib/kubelet/config.yaml中的syncFrequency参数，将周期减少到5s，注意此修改可能影响集群性能。
  ```
  syncFrequency: 5s
  ```
  然后使用以下命令重启kubelet：
  ```
  swapoff -a
  systemctl restart  kubelet.service
  systemctl status kubelet
  ```
- 确保Docker配置了标准输出流写入到文件的最大规格，防止磁盘占满导致Pod被驱逐。
  需在待部署服务的计算节点上修改Docker配置文件后重启Docker：
  1. 使用以下命令打开daemon.json文件。
     ```
     vim /etc/docker/daemon.json
     ```
     在daemon.json文件中添加日志选项“log-opts”，具体内容如下所示。
     ```
     "log-opts":{"max-size":"500m", "max-file":"3"}
     ```
     参数解释：
     max-size=500m：表示一个容器日志大小上限是500M。
     max-file=3：表示一个容器最多有三个日志，超过会自动滚动更新。
  2. 使用以下命令重启Docker.
     ```
     systemctl daemon-reload
     systemctl restart docker
     ```

步骤10 使用kubectl命令查看PD集群状态。

```
kubectl get pods -n mindie
```

如启动4个MindIE Server实例，回显如下所示：

```
NAME                                READY   STATUS    RESTARTS   AGE    IP             NODE       NOMINATED
NODE   READINESS GATES
mindie-ms-controller-7845dcd697-h4gw7   1/1     Running   0          145m   xx.xx.xx.xx    ubuntu10
<none>         <none>
mindie-ms-coordinator-6bff995ff8-l6fwz  1/1     Running   0          145m   xx.xx.xx.xx    ubuntu10
<none>         <none>
mindie-server-7b795f8df9-2xvh4          1/1     Running   0          145m   xx.xx.xx.xx    ubuntu
<none>         <none>
mindie-server-7b795f8df9-j4z7d          1/1     Running   0          145m   xx.xx.xx.xx    ubuntu
<none>         <none>
mindie-server-7b795f8df9-v2tcz          1/1     Running   0          145m   xx.xx.xx.xx    ubuntu
<none>         <none>
mindie-server-7b795f8df9-vl9hv          1/1     Running   0          145m   xx.xx.xx.xx    ubuntu
<none>         <none>
```

- 以mindie-ms-controller开头的为MindIE MS的Controller控制器组件。
- 以mindie-ms-coordinator开头的为MindIE MS的Coordinator调度器组件。
- 以mindie-server开头的为MindIE Server推理服务组件。

如观察Pod进入Running状态，表示Pod容器已成功被调度到节点并正常启动，但还需要进一步确认业务程序是否启动成功。

- 通过脚本示例提供的log.sh脚本可查询这些Pod的标准输出日志，查看程序是否出现异常：
  ```
  bash log.sh
  ```
- 如需要查询具体某个Pod（如上面mindie-server-7b795f8df9-vl9hv）的日志，则执行以下命令：
  ```
  kubectl logs mindie-server-7b795f8df9-vl9hv -n mindie
  ```
- 如需要进入容器查找更多定位信息，则执行以下命令：
  ```
  kubectl exec -it mindie-server-7b795f8df9-vl9hv -n mindie -- bash
  ```
- 如需确认的P，D对应的Pod，待MindIE MS Controller组件启动（如上面mindie-ms-controller-7845dcd697-h4gw7处于READY 1/1状态）后，执行以下命令：
  ```
  kubectl logs mindie-ms-controller-7845dcd697-h4gw7 -n mindie | grep UpdateServerInfo
  ```
  查询到P节点和D节点的Pod IP，并结合上面查询Pod状态命令回显的IP可找到对应的Pod。

步骤11 通过脚本示例提供的generate_stream.sh发起流式推理请求。

修改generate_stream.sh中的IP地址为当前节点的IP地址，如Coordinator组件启用了HTTPS，需要配置相关的证书；如使用HTTP，需修改脚本中的HTTPS为HTTP，并删除证书相关配置。

```
bash generate_stream.sh
```

步骤12 卸载PD集群。

如需停止PD服务或者修改业务配置重新部署实例，需要调用以下命令卸载已部署的实例，重新部署请执行**步骤9**。

```
bash delete.sh
```

----结束

4 服务化接口

4.1 说明

在当前版本中，用户可以在服务化配置中配置**ModelConfig参数说明**对应的后处理参数默认值。若用户没有指定相关配置，则后处理参数的默认值将会从模型权重目录下的配置文件中读取。不同后端的配置文件不同：

- atb（ATB Models）：配置文件为generation_config.json文件与config.json文件，其中generation_config.json的优先级更高。若用户与模型权重均未指定“top_k”参数，为平衡性能与推理效果，“top_k”参数将会被设定为1000。
- ms（MindSpore）：配置文件为“.yaml”结尾的文件。若用户与模型权重均未指定“top_k”参数，“top_k”参数将会被设定为0。

须知

- 上述规则可能会导致当前版本与1.0.RC2版本的推理结果不一致，但当前版本的结果会更加符合人类语言习惯。如需复现1.0.RC2版本的推理结果，需要指定后处理参数和上个版本Server提供的默认值一致，1.0.RC2版本的Server提供的默认值请参见**表4-1**。
- 当前版本暂不支持通过修改generation_config.json文件中的“do_sample”参数为True来控制服务端默认开启采样。
- 不可在generation_config.json文件中将“temperature”设为0，否则将导致推理结果异常。

表 4-1 1.0.RC2 版本的 Server 提供的默认值

参数	默认值
temperature	1.0

参数	默认值
top_k	0
top_p	1.0
do_sample	False
repetition_penalty	1.0
frequency_penalty	0.0
presence_penalty	0.0

采样规则

- vLLM接口和OpenAI接口不支持do_sample，传递的do_sample不会被校验。
- 若用户传递do_sample=true，则程序会开启采样。
- 若用户传递do_sample=false，则程序会关闭采样，进行greedy search。
- 若用户未传递do_sample，则程序会根据temperature、top_k、top_p参数进行自动判断。自动判断的逻辑如下：
  - temperature=0时，无视top_k、top_p取值，关闭后处理，进行greedy search。
  - 若传递了top_k、top_p、或非0的temperature中的任一参数，则开启后处理采样，未传递的参数将按上文规则读取默认值填充。
  - 若未传递任何参数，则进行greedy search。

4.2 EndPoint 业务面 RESTful 接口

4.2.1 兼容 TGI 0.9.4 版本接口

4.2.1.1 健康检查

接口功能

检查服务状态是否正常。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/health

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/health
```

响应样例：

服务状态正常时无内容。

输出说明

- 状态码200，服务状态正常，消息体没有内容。
- 其他状态码，服务状态异常。

4.2.1.2 查询 TGI EndPoint 信息

说明

最大程度兼容TGI接口返回格式，对于MindIE Server不支持的返回字段，返回null。

接口功能

查询TGI EndPoint信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/info

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/info
```

响应样例：

```
{
    "docker_label": null,
    "max_batch_total_tokens": 8192,
    "max_best_of": 1,
    "max_concurrent_requests": 200,
    "max_stop_sequences": null,
    "max_waiting_tokens": null,
    "sha": null,
    "validation_workers": null,
    "version": "1.0.RC3",
    "waiting_served_ratio": null,
    "models": [
        {
            "model_device_type": "npu",
            "model_dtype": "float16",
            "model_id": "llama_65b",
            "model_pipeline_tag": "text-generation",
            "model_sha": null,
            "max_total_tokens": 2560
        }
    ],
    "max_input_length": 2048
}
```

响应状态码： 200

输出说明

参数	类型	说明
docker_label	string	暂不支持，默认返回null。
max_batch_total_tokens	int	取maxPrefillTokens。
max_best_of	int	暂不支持best_of参数，默认返回1，即每次只返回1个推理结果。
max_concurrent_requests	int	最大并发请求数，取maxBatchSize。
max_stop_sequences	int	暂不支持，默认返回null。
max_waiting_tokens	int	暂不支持，默认返回null。
sha	string	暂不支持，默认返回null。
validation_workers	int	暂不支持，默认返回null。
version	string	版本号。

参数	类型	说明
waiting_served_ratio	float	暂不支持，默认返回null。
models	list	模型配置。
model_device_type	string	模型运行设备类型，默认返回"npu"。
model_dtype	string	模型数据类型，读取权重配置文件目录config.json文件中的torch_dtype字段。
model_id	string	模型名称。
model_pipeline_tag	string	模型任务类型，默认返回"text-generation"。
model_sha	string	暂不支持，默认返回null。
max_total_tokens	int	最大推理token总数，读取maxSeqLen的值。
max_input_length	int	最大输入长度，读取maxInputTokenLen的值。

4.2.1.3 文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。如果“stream”=true，该参数不能为true。	bool类型，默认值false。

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“decoder_input_details”和“details”字段任意一个为true，即返回所有的details信息。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0, 2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true：添加。 ● false：不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段的值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 • true：带模型水印。 • false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 ● true：流式推理。 ● false：文本推理。

使用样例

请求样例：

```
POST https://{ip}:{port}/
```

请求消息体：

- 单模态文本模型：

```
{
    "inputs": "My name is Olivier and I",
    "parameters": {
        "decoder_input_details": true,
        "details": true,
        "do_sample": true,
        "max_new_tokens": 20,
        "repetition_penalty": 1.03,
        "return_full_text": false,
        "seed": null,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "truncate": null,
        "typical_p": 0.5,
        "watermark": false,
        "stop": ["stop1", "stop2"],
        "adapter_id": "None"
    },
    "stream": false
}
```


- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
          "type": "image_url",
          "image_url": "/xxxx/test.png"
      }
   ],
   "parameters": {
      "decoder_input_details": true,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": ["stop1", "stop2"],
      "adapter_id": "None"
   },
   "stream": false
}
```

响应样例：

- 文本推理（“stream”=false）：

```
[
   {
      "details": {
         "prompt_tokens": 6,
         "finish_reason": "length",
         "generated_tokens": 20,
         "prefill": [
            {
               "id": 5050,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 829,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 374,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 77018,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 323,
```



```
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 358,
            "logprob": null,
            "special": null,
            "text": null
        }
    ],
    "seed": 789070824,
    "tokens": [
        {
            "id": 2776,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 264,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 3162,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 23576,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 504,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 9625,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 13,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 358,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 2948,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 311,
```



```
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 1936,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 2513,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 11,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 5310,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 3482,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 8357,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 323,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 6371,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 10500,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 13,
            "logprob": null,
            "special": null,
            "text": null
        }
    ]
},
"generated_text": "'m a software engineer from France. I love to build things, especially web applications and mobile apps."
```



```
  }
]
```

- 流式推理：
  - 流式推理1（“stream”=true，“decoder_input_details”=false，使用sse格式返回）：

```
data: {"token":
{"id":29915,"text":"'","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29885,"text":"m","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":"
a","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":5176,"text":"
French","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1410,"text":"
gu","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29891,"text":"y","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1058,"text":"
who","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":338,"text":"
is","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3063,"text":"
looking","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":363,"text":"
for","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":"
a","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2058,"text":"
place","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":304,"text":"
to","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":5735,"text":"
live","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":297,"text":"
in","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29889,"text":".","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":306,"text":"
I","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29915,"text":"'","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29885,"text":"m","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":" a","logprob":null,"special":null},"generated_text":"'m a French
guy who is looking for a place to live in. I'm a","details":
{"prompt_tokens":8,"finish_reason":"length","generated_tokens":20,"seed":218884523}}
```

  - 流式推理2（“stream”=true，“decoder_input_details”=false，配置项“fullTextEnabled”=true，使用sse格式返回）：


```
data: {"token":{"id":11,"text":",","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":600,"text":",
i","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1184,"text":", i
need","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":498,"text":", i need
you","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":151645,"text":", i need you","logprob":null,"special":null},"generated_text":",
i need you","details":
{"prompt_tokens":1,"finish_reason":"eos_token","generated_tokens":5,"seed":1621189503}}
```

输出说明

表 4-2 文本推理结果说明

返回值	类型	说明
details	object	推理details结果，请求参数“decoder_input_details”和“details”任意一个字段为true，即返回details结果。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
prefill	List[token]	请求参数“decoder_input_details”=true，返回推理请求文本detokenizer之后的token，默认为空列表。

返回值	类型	说明
prefill.id	int	token ID。
prefill.logprob	float	概率对数，可以为空（第一个token概率值不能被计算获得）。当前不支持，默认返回null。
prefill.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
prefill.text	string	token对应文本。 当前不支持，默认返回null。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。
tokens	List[token]	返回推理结果的所有tokens。
tokens.id	int	token ID。
tokens.logprob	概率对数	概率对数。当前不支持，默认返回null。
tokens.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
tokens.text	string	token对应文本。 当前不支持，默认返回null。
generated_text	string	推理返回结果。

表 4-3 流式推理结果说明

返回值	类型	说明
data	object	一次推理返回的结果。
token	object	每一次推理的token。
id	int	token ID。
text	string	token对应文本。
logprob	概率对数	当前不支持，默认返回null。
special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
generated_text	string	推理文本返回结果，只在最后一次推理结果才返回。

返回值	类型	说明
details	object	推理details结果，只在最后一次推理结果返回，并且请求参数“details”=true才返回details结果。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。

4.2.1.4 文本推理接口

接口功能

提供文本推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于1MB、maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。	bool类型，默认值false。

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“decoder_input_details”和“details”字段任意一个为true，即返回所有的details信息。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true表示添加。 ● false表示不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate
```

请求消息体：

- 单模态文本模型：

```
{
   "inputs": "My name is Olivier and I",
   "parameters": {
      "decoder_input_details": true,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": null,
      "adapter_id": "None"
   }
}
```

- 多模态模式：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
```



```
            {"type": "text", "text": "My name is Olivier and I"},
            {
                "type": "image_url",
                "image_url": "/xxxx/test.png"
            }
        ],
        "parameters": {
            "decoder_input_details": true,
            "details": true,
            "do_sample": true,
            "max_new_tokens": 20,
            "repetition_penalty": 1.03,
            "return_full_text": false,
            "seed": null,
            "temperature": 0.5,
            "top_k": 10,
            "top_p": 0.95,
            "truncate": null,
            "typical_p": 0.5,
            "watermark": false,
            "stop": null,
            "adapter_id": "None"
        }
    }
```

响应样例：

```
{
   "details": {
      "finish_reason": "length",
      "generated_tokens": 1,
      "prefill": [{
         "id": 0,
         "logprob":null,
         "special": null,
         "text": "test"
      }],
      "prompt_tokens": 74,
      "seed": 42,
      "tokens": [{
         "id": 0,
         "logprob": null,
         "special": null,
         "text": "test"
      }]
   },
   "generated_text": "am a Frenchman living in the UK. I have been working as an IT consultant for "
}
```

输出说明

返回值	类型	说明
details	object	推理details结果，请求参数“decoder_input_details”和“details”任意一个字段为true，即返回details结果。

返回值	类型	说明
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
prefill	List[token]	请求参数“decoder_input_details”=true，返回推理请求文本detokenizer之后的token，默认为空列表。
prefill.id	int	token ID。
prefill.logprob	float	概率对数，可以为空（第一个token概率值不能被计算获得）。当前不支持，默认返回null。
prefill.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
prefill.text	string	token对应文本。 当前不支持，默认返回null。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。
tokens	List[token]	返回推理结果的所有tokens。
tokens.id	int	token ID。

返回值	类型	说明
tokens.logprob	概率对数	概率对数。当前不支持，默认返回null。
tokens.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
tokens.text	string	token对应文本。 当前不支持，默认返回null。
generated_text	string	推理返回结果。

4.2.1.5 流式推理接口

接口功能

提供流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate_stream

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。	bool类型，默认值false。 对于流式接口，该参数只能为false。

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“details”=true，即返回所有的details信息。	bool类型参数，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true表示添加。 ● false表示不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0，18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json或者padded_vocab_size的文件中读取的vocab_size值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认值使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate_stream
```

请求消息体：

- 单模态文本模型：

```
{
   "inputs": "My name is Olivier and I",
   "parameters": {
      "decoder_input_details": false,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": null,
      "adapter_id": "None"
   }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
```



```
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "parameters": {
        "decoder_input_details": false,
        "details": true,
        "do_sample": true,
        "max_new_tokens": 20,
        "repetition_penalty": 1.03,
        "return_full_text": false,
        "seed": null,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "truncate": null,
        "typical_p": 0.5,
        "watermark": false,
        "stop": null,
        "adapter_id": "None"
    }
}
```

响应样例：

- 响应样例1（使用sse格式返回）：

```
data: {"token":{"id":13,"text":"\n","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":26626,"text":"Jan","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":300,"text":"et","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3732,"text":"
makes","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":395,"text":" $","logprob":null,"special":null},"generated_text":null,"details":null}

......

data: {"token":{"id":395,"text":" $","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29896,"text":"1","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29896,"text":"1","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29947,"text":"8","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29889,"text":".","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2,"text":"","logprob":null,"special":null},"generated_text":"\nJanet makes $104 a
day at the farmers' market.\nThe number of eggs that Janet sells at the farmers' market each day is
16 - 3 - 4 = 7.\nShe makes $2 for each egg that she sells.\nThe total amount that she makes is $2 * 7
= $14.\nThe total amount that she makes is $14 + $104 = $118.\nThe total amount that she makes is
$118.","details":
{"prompt_tokens":74,"finish_reason":"eos_token","generated_tokens":116,"seed":8756727004129248931}}
```

- 响应样例2（配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data: {"token":{"id":198,"text":"\n","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":9707,"text":"\nHello","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":0,"text":"\nHello!","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2585,"text":"\nHello!
How","logprob":null,"special":null},"generated_text":null,"details":null}
```



```
data: {"token":{"id":646,"text":"\nHello! How
can","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":358,"text":"\nHello! How can
I","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":7789,"text":"\nHello! How can I
assist","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":498,"text":"\nHello! How can I assist
you","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3351,"text":"\nHello! How can I assist you
today","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":30,"text":"\nHello! How can I assist you
today?","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":151645,"text":"\nHello! How can I assist you
today?","logprob":null,"special":null},"generated_text":"\nHello! How can I assist you today?","details":
{"prompt_tokens":1,"finish_reason":"eos_token","generated_tokens":11,"seed":2296576927}}
```

输出说明

返回值	类型	说明
data	object	一次推理返回的结果。
token	object	每一次推理的token。
id	int	token ID。
text	string	token对应文本。
logprob	概率对数	当前不支持，默认返回null。
special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
generated_text	string	推理文本返回结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，并且请求参数“details”=true才返回details结果。
prompt_tokens	int	用户输入的prompt文本对应的token长度。

返回值	类型	说明
finish_reason	string	结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。

4.2.2 兼容 vLLM 0.2.6 版本接口

4.2.2.1 健康检查

接口功能

检查服务状态是否正常。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/health

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/health
```

响应样例：

服务状态正常时无内容。

输出说明

- 状态码200，服务状态正常，消息体没有内容。
- 其他状态码，服务状态异常。

4.2.2.2 文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
prompt	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。

参数	是否必选	说明	取值要求
max_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_tokens)。	int类型，取值范围(0，2147483647]，默认值为MindIE Server配置文件中的maxIterTimes。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，取值范围(0.0, 2.0]，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。
presence_penalty	可选	存在惩罚介于-2.0和2.0之间，它影响模型如何根据到目前为止是否出现在文本中来惩罚新token。正值将通过惩罚已经使用的词，增加模型谈论新主题的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
frequency_penalty	可选	频率惩罚介于-2.0和2.0之间，它影响模型如何根据文本中词汇的现有频率惩罚新词汇。正值将通过惩罚已经频繁使用的词来降低模型一行中重复用词的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。1.0表示不进行计算，大于1.0表示输出随机性提高。temperature=0.0，即采用greedy sampling。	float类型，取值大于或等于0.0。 取0.0时将忽略其他后处理参数做greedysearch。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。建议最大值取2.0，同时视模型而定。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。-1表示不进行top k计算。	uint32_t类型，取值范围-1或者(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 若传-1，-1会变为0传递给MindIE LLM后端，MindIE LLM后端会当做词表大小来处理。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 • true：流式推理。 • false：文本推理。
stop	可选	停止推理的文本。输出结果中默认不包含停止词列表文本。	List[string]类型或者string类型，默认为null。 • List[string]类型，每个元素字符长度大于或等于1，列表元素总长度不超过32768（32*1024）。列表为空时相当于null。 • string类型长度范围为1~32768。

参数	是否必选	说明	取值要求
stop_token_ids	可选	停止推理的token ID列表。输出结果中默认不包含停止推理列表中的token ID。	List[int32]类型，超出int32的元素将会被忽略。 默认为null。
model	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。
include_stop_str_in_output	可选	决定是否在生成的推理文本中包含停止字符串。	bool类型，默认值为false。**PD场景暂不支持此参数**。 ● true：包含停止字符串。 ● false：不包含停止字符串。 不传入stop或stop_token_ids时，此字段会被忽略。
skip_special_tokens	可选	指定在推理生成的文本中是否跳过特殊tokens。	bool类型，默认值为true。 ● true：跳过特殊tokens。 ● false：保留特殊tokens。
ignore_eos	可选	指定在推理文本生成过程中是否忽略eos_token结束符	bool类型，默认值为false。 ● true：忽略eos_token结束符。 ● false：不忽略eos_token结束符。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate
```

请求消息体：

- 单模态文本模型：

```
{
    "prompt": "My name is Olivier and I",
    "max_tokens": 20,
    "repetition_penalty": 1.03,
    "presence_penalty": 1.2,
    "frequency_penalty": 1.2,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": 10,
    "seed": null,
    "stream": false,
    "stop": null,
    "stop_token_ids": null,
    "model": "None",
    "include_stop_str_in_output": false,
    "skip_special_tokens": true,
```



```
    "ignore_eos": false
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
    "prompt": [
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "max_tokens": 20,
    "repetition_penalty": 1.03,
    "presence_penalty": 1.2,
    "frequency_penalty": 1.2,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": 10,
    "seed": null,
    "stream": false,
    "stop": null,
    "stop_token_ids": null,
    "model": "None",
    "include_stop_str_in_output": false,
    "skip_special_tokens": true,
    "ignore_eos": false
}
```

响应样例：

- 文本推理（“stream”=false）：

```
{"text":["My name is Olivier and I am a Frenchman living in the UK. I am a keen photographer and"]}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

```
{"text":["am"]}{"text":[" a"]}{"text":[" French"]}{"text":["man"]}{"text":[" living"]}{"text":["
in"]}"text":[" the"]}{"text":[" UK"]}{"text":["."]}{"text":[" I"]} {"text":[" am"]}{"text":[" a"]}"text":
[" keen"]}{"text":[" photograph"]}{"text":["er"]}{"text":[" and"]}
```

  - 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
{"text":[" to"]}{"text":[" to travel"]}{"text":[" to travel."]}{"text":[" to travel. I"]}{"text":[" to
travel. I'm"]}
```

输出说明

返回值	类型	说明
text	string	推理返回结果。

须知

vLLM流式返回结果，每个token的返回结果添加'\0'字符做分割。使用**curl**命令发送vLLM流式推理请求的样例如下：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert /home/runs/static_conf/ca/ca.pem  --cert /home/runs/static_conf/cert/client.pem  --key /home/runs/static_conf/cert/client.key.pem -X POST -d '{
"prompt": "My name is Olivier and I",
"stream": true,
"repetition_penalty": 1.0,
"top_p": 1.0,
"top_k": 10,
"max_tokens": 16,
"temperature": 1.0
}' https://{ip}:{port}/generate | cat
```

4.2.3 兼容 OpenAI 接口

4.2.3.1 列举当前模型列表

接口功能

列举当前模型列表信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v1/models

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/models
```

响应样例：

```
{
   "object": "list",
   "data": [
      {
         "id": "model-id-0",
         "object": "model",
         "created": 1686935002,
         "owned_by": "MindIE Server"
      }
   ]
}
```


响应状态码：200

输出说明

参数	类型	说明
object	string	返回结果类型，默认"list"。
data	list	模型列表。
data.id	string	模型名称。
data.object	string	数据结果类型，默认"model"。
data.created	int	模型加载时间戳，即服务启动时间戳，单位：秒。
data.owned_by	string	模型的拥有者，默认返回"MindIE Server"。

4.2.3.2 查询单个模型信息

接口功能

查询指定模型信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v1/models/*{model}*

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

URL链接中{model}字段为模型名称。

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/models/model-id-0
```

响应样例：

```
{
    "id": "model-id-0",
    "object": "model",
    "created": 1686935002,
    "owned_by": "MindIE Server"
}
```

响应状态码：200

输出说明

参数	类型	说明
id	string	模型名称。
object	string	数据结果类型，默认"model"。
created	int	模型加载时间戳，即服务启动时间戳，单位：秒。
owned_by	string	模型的拥有者，默认返回"MindIE Server"。

4.2.3.3 推理接口

须知

- 运行环境的transformers版本不可低于4.34.1，低版本tokenizer不支持"chat_template"方法。
- 推理模型权重路径下的tokenizer_config.json需要包含"chat_template"字段及其实现。
- function call功能的相关参数tool_call_id、tool_calls、tools和tool_choice当前仅支持部分模型，使用其他模型可能会报错。目前支持的模型只有ChatGLM3-6B。

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v1/chat/completions

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
model	必选	模型名。	与MindIE Server配置文件中modelName的取值保持一致。

参数	是否必选	说明	取值要求
messages	必选	推理请求消息结构。	list类型，0KB<messages内容包含的字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
role	必选	推理请求消息角色。	字符串类型，可取角色有： ● system：系统角色 ● user：用户角色 ● assistant：助手角色 ● tool：工具角色
content	必选	推理请求文本。	字符串类型。 ● 当role为assistant，且tool_calls非空时，content可以不传，其余角色非空。 ● 其余情况content非空。
tool_calls	可选	模型生成的工具调用。	类型为List[dict]，当role为assistant时，表示模型对工具的调用。
tool_calls.function	必选	表示模型调用的工具。	dict类型。 ● arguments，必选，使用JSON格式的字符串，表示调用函数的参数。 ● name，必选，字符串，调用的函数名。
tool_calls.id	必选	表示模型某次工具调用的ID。	字符串。
tool_calls.type	必选	调用的工具类型。	字符串，仅支持"function"。
tool_call_id	当role为tool时必选，否则可选	关联模型某次调用工具时的ID。	字符串。

参数	是否必选	说明	取值要求
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型参数，默认值false。 ● true：流式推理。 ● false：文本推理。
presence_penalty	可选	存在惩罚介于-2.0和2.0之间，它影响模型如何根据到目前为止是否出现在文本中来惩罚新token。正值将通过惩罚已经使用的词，增加模型谈论新主题的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
frequency_penalty	可选	频率惩罚介于-2.0和2.0之间，它影响模型如何根据文本中词汇的现有频率惩罚新词汇。正值将通过惩罚已经频繁使用的词来降低模型一行中重复用词的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
repetition_penalty	可选	重复惩罚是一种技术，用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，取值范围(0.0, 2.0]，默认值1.0。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，取值范围[0.0, 2.0]，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围[0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型，默认值null。 • List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 • string类型长度范围为1~1024。
stop_token_ids	可选	停止推理的token ID列表。输出结果默认不包含停止推理列表中的token ID。	List[int32]类型，超出int32的元素将会被忽略，默认值null。

参数	是否必选	说明	取值要求
include_stop_str_in_output	可选	决定是否在生成的推理文本中包含停止字符串。	bool类型，默认值false。**PD场景暂不支持此参数**。 ● true：包含停止字符串。 ● false：不包含停止字符串。 不传入stop或stop_token_ids时，此字段会被忽略。
skip_special_tokens	可选	指定在推理生成的文本中是否跳过特殊tokens。	bool类型，默认值true。 ● true：跳过特殊tokens。 ● false：保留特殊tokens。
ignore_eos	可选	指定在推理文本生成过程中是否忽略eos_token结束符。	bool类型，默认值false。 ● true：忽略eos_token结束符。 ● false：不忽略eos_token结束符。
max_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_tokens)。	int类型，取值范围(0，2147483647]，默认值maxIterTimes。
tools	可选	可能会使用的工具列表。	List[dict]类型。
tools.type	必选	说明工具类型。	仅支持字符串"function"。
tools.function	必选	函数描述。	dict类型。
function.name	必选	函数名称。	字符串。
function.strict	可选	表示生成tool calls是否严格遵循schema格式。	bool类型，默认false。
function.description	可选	描述函数功能和使用。	字符串。
function.parameters	可选	表示函数接受的参数。	JSON schema格式。
parameters.type	必选	表示函数参数属性的类型。	字符串，仅支持object。

参数	是否必选	说明	取值要求
parameters.properties	必选	函数参数的属性。每一个key表示一个参数名，由用户自定义。value为dict类型，表示参数描述，包含type和description两个参数。	dict类型。
function.required	必选	表示函数必填参数列表。	List[string]类型。
function.additionalProperties	可选	是否允许使用未提及的额外参数。	bool类型，默认值false。 ● true：允许使用未提及的额外参数。 ● false：不允许使用未提及的额外参数。
tool_choice	可选	控制模型调用工具。	string类型或者dict类型，可以为null，默认值"auto"。 ● "none"：表示模型不会调用任何工具，而是生成一条消息。 ● "auto"：表示模型可以生成消息或调用一个或多个工具。 ● "required"：表示模型必须调用一个或多个工具。 通过{"type": "function", "function": {"name": "my_function"}}指定特定的工具，将强制模型调用该工具。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/chat/completions
```

请求消息体：

- 单轮对话：

```
{
    "model": "gpt-3.5-turbo",
    "messages": [{
        "role": "user",
        "content": "You are a helpful assistant."
    }],
    "stream": false,
    "presence_penalty": 1.03,
```



```
    "frequency_penalty": 1.0,
    "repetition_penalty": 1.0,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": 0,
    "seed": null,
    "stop": ["stop1", "stop2"],
    "stop_token_ids": [2, 13],
    "include_stop_str_in_output": false,
    "skip_special_tokens": true,
    "ignore_eos": false,
    "max_tokens": 20
}
```

- 多轮对话：
  - 请求样例1：

```
{
    "model": "chatglm3-6b",
    "messages": [{
        "role": "system",
        "content": "You are a helpful customer support assistant. Use the supplied tools to assist the user."
        },
        {
        "role": "user",
        "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
        }
    ],
    "stream": false,
    "presence_penalty": 1.03,
    "frequency_penalty": 1.0,
    "repetition_penalty": 1.0,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": 0,
    "seed": null,
    "stop": ["stop1", "stop2"],
    "stop_token_ids": [2],
    "ignore_eos": false,
    "max_tokens": 1024,
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "strict": true,
                "description": "Get the delivery date for a customer's order. Call this whenever you need to know the delivery date, for example when a customer asks 'Where is my package'",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer's order ID."
                        }
                    },
                    "required": [
                        "order_id"
                    ],
                    "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "auto"
}
```

  - 请求样例2：

```
{
    "model": "chatglm3-6b",
```



```
    "messages": [
        {
            "role": "system",
            "content": "You are a helpful customer support assistant. Use the supplied tools to
assist the user."
        },
        {
            "role": "user",
            "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
        },
        {
            "role": "assistant",
            "tool_calls": [
                {
                    "function": {
                        "arguments": "{\"order_id\": \"12345\"}",
                        "name": "get_delivery_date"
                    },
                    "id": "tool_call_8p2Nk",
                    "type": "function"
                }
            ]
        },
        {
            "role": "tool",
            "content": "the delivery date is 2024.09.10.",
            "tool_call_id": "tool_call_8p2Nk"
        }
    ],
    "stream": false,
    "repetition_penalty": 1.1,
    "temperature": 0.9,
    "top_p": 1,
    "max_tokens": 1024,
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "strict": true,
                "description": "Get the delivery date for a customer's order. Call this whenever you
need to know the delivery date, for example when a customer asks 'Where is my package'",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer's order ID."
                        }
                    },
                    "required": [
                        "order_id"
                    ],
                    "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "auto"
}
```

响应样例：

- 文本推理（“stream”=false）：
  - 单轮对话：

```
{
    "id": "chatcmpl-123",
    "object": "chat.completion",
    "created": 1677652288,
```



```
  "model": "gpt-3.5-turbo-0613",
  "choices": [
     {
        "index": 0,
        "message": {
           "role": "assistant",
           "content": "\n\nHello there, how may I assist you today?"
        },
        "finish_reason": "stop"
     }
  ],
  "usage": {
     "prompt_tokens": 9,
     "completion_tokens": 12,
     "total_tokens": 21
  }
}
```

- 多轮对话：
  - 响应样例1：

```
{
   "id": "chatcmpl-123",
   "object": "chat.completion",
   "created": 1677652288,
   "model": "chatglm3-6b",
   "choices": [
      {
         "index": 0,
         "message": {
            "role": "assistant",
            "content": "",
            "tool_calls": [
               {
                  "function": {
                     "arguments": "{\"order_id\": \"12345\"}",
                     "name": "get_delivery_date"
                  },
                  "id": "call_JwmTNF3O",
                  "type": "function"
               }
            ]
         },
         "finish_reason": "tool_calls"
      }
   ],
   "usage": {
      "prompt_tokens": 226,
      "completion_tokens": 122,
      "total_tokens": 348
   }
}
```

  - 响应样例2：

```
{
   "id": "endpoint_common_25",
   "object": "chat.completion",
   "created": 1728959154,
   "model": "chatglm3-6b",
   "choices": [
      {
         "index": 0,
         "message": {
            "role": "assistant",
            "content": "\n Your order with ID 12345 is scheduled for delivery on September 10th, 2024.",
            "tool_calls": null
         },
         "finish_reason": "stop"
      }
```



```
    ],
    "usage": {
        "prompt_tokens": 265,
        "completion_tokens": 30,
        "total_tokens": 295
    }
}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：


```
data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"\t"},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"\t"},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}
```



```
data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"lla
ma_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"lla
ma_65b","usage":{"prompt_tokens":54,"completion_tokens":17,"total_tokens":71},"choices":
[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":"stop"}]}

data: [DONE]
```

- 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello!"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello!
How"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How
can"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can
I"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I
assist"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist
you"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you
today"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you
today?"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","full_text":"Hello! How can I assist you today?","usage":
{"prompt_tokens":31,"completion_tokens":10,"total_tokens":41},"choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello! How can I assist you today?"},"finish_reason":"length"}]}

data: [DONE]
```


输出说明

表 4-4 文本推理结果说明

参数名	类型	说明
id	string	请求ID。
object	string	返回结果类型目前都返回"chat.completion"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
choices	list	推理结果列表。
index	integer	choices消息index，当前只能为0。
message	object	推理消息。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。
tool_calls	list	模型工具调用输出。
function	dict	函数调用说明。
arguments	string	调用函数的参数，JSON格式的字符串。
name	string	调用的函数名。
tool_calls.id	string	模型调用工具的ID。
type	string	工具的类型，目前仅支持function。
finish_reason	string	结束原因。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● tool_calls：表示模型调用了工具。
usage	object	推理结果统计数据。

参数名	类型	说明
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。

表 4-5 流式推理结果说明

参数名	类型	说明
data	object	一次推理返回的结果。
id	string	请求ID。
object	string	目前都返回"chat.completion.chunk"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
full_text	string	全量文本结果，配置项“fullTextEnabled”=true时才有此返回值。
usage	object	推理结果统计数据。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。
choices	list	流式推理结果。
index	integer	choices消息index，当前只能为0。
delta	object	推理返回结果，最后一个响应为空。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。

参数名	类型	说明
finish_reason	string	结束原因，只在最后一次推理结果返回。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。

4.2.4 vLLM 兼容 OpenAI 接口

4.2.4.1 推理接口

须知

- 运行环境的transformers版本不可低于4.34.1，低版本tokenizer不支持"chat_template"方法。
- 推理模型权重路径下的tokenizer_config.json需要包含"chat_template"字段及其实现。
- function call功能的相关参数tool_call_id、tool_calls、tools和tool_choice当前仅支持部分模型，使用其他模型可能会报错。目前支持的模型只有ChatGLM3-6B。

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v1/chat/completions

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	是否必选	说明	取值要求
model	必选	模型名。	与MindIE Server配置文件中modelName的取值保持一致。
messages	必选	推理请求消息结构。	list类型，0KB<messages内容包含的字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取）。
role	必选	推理请求消息角色。	字符串类型，可取角色有： ● system：系统角色 ● user：用户角色 ● assistant：助手角色 ● tool：工具角色。使用该角色时，必须要传入tool_call_id。
content	必选	推理请求文本。	字符串类型。 ● 当role为assistant，且tool_calls非空时，content可以不传，其余角色非空。 ● 其余情况content非空。
tool_calls	可选	模型生成的工具调用。	类型为List[dict]，当role为assistant时，表示模型对工具的调用。
tool_calls.function	必选	表示模型调用的工具。	dict类型。 ● arguments，必选，使用JSON格式的字符串，表示调用函数的参数。 ● name，必选，字符串，调用的函数名。
tool_calls.id	必选	表示模型某次工具调用的ID。	字符串。

参数	是否必选	说明	取值要求
tool_calls.type	必选	调用的工具类型。	字符串，仅支持"function"。
tool_call_id	当role为tool时必选，否则可选	关联模型某次调用工具时的ID。	字符串。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型参数，可以为null，默认值false。 ● true：流式推理。 ● false：文本推理。
presence_penalty	可选	存在惩罚介于-2.0和2.0之间，它影响模型如何根据到目前为止是否出现在文本中来惩罚新token。正值将通过惩罚已经使用的词，增加模型谈论新主题的可能性。 不建议同时修改该值与repetition_penalty或frequency_penalty。	float类型，取值范围[-2.0, 2.0]，默认值0.0，可以为null。
frequency_penalty	可选	频率惩罚介于-2.0和2.0之间，它影响模型如何根据文本中词汇的现有频率惩罚新词汇。正值将通过惩罚已经频繁使用的词来降低模型一行中重复用词的可能性。 不建议同时修改该值与repetition_penalty或presence_penalty。	float类型，取值范围[-2.0, 2.0]，默认值0.0，可以为null。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 不建议同时修改该值与presence_penalty或frequency_penalty。	float类型，取值范围(0.0, 2.0]，默认值1.0，可以为null。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。 不建议同时修改该值与top_p。	float类型，取值范围大于或等于0.0，默认值1.0，可以为null。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 不建议同时修改该值与temperature。	float类型，取值范围(0.0, 1.0]，默认值1.0，可以为null。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32类型，取值范围-1或（0, 2147483647]，可以为null。字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 若传-1，-1会变为0传递给MindIE LLM后端，MindIE LLM后端会当做词表大小vocabSize来处理。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。

参数	是否必选	说明	取值要求
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint64类型，取值范围[0, 18446744073709551615]，可以为null。不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型，默认值null。 ● List[string]类型，每个元素字符长度大于或等于1，列表元素总长度不超过32768（32*1024）。列表为空时相当于null。 ● string类型长度范围为1~32768。
stop_token_ids	可选	停止推理的token ID列表。输出结果不包含停止推理列表中的token ID。	List[int32]类型，默认值null。若该字段值非null，列表中元素不能为null，超出int32的元素将会被忽略。
include_stop_str_in_output	可选	决定是否在生成的推理文本中包含停止字符串。	bool类型，可以为null，默认值false。**PD场景暂不支持此参数**。 ● true：包含停止字符串。 ● false：不包含停止字符串。 不传入stop或stop_token_ids时，此字段会被忽略。
skip_special_tokens	可选	指定在推理生成的文本中是否跳过特殊tokens。	bool类型，可以为null，默认值true。 ● true：跳过特殊tokens。 ● false：保留特殊tokens。

参数	是否必选	说明	取值要求
ignore_eos	可选	指定在推理文本生成过程中是否忽略eos_token结束符。	bool类型，可以为null，默认值false。 ● true：忽略eos_token结束符。 ● false：不忽略eos_token结束符。
max_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_tokens)。	整型，取值范围(0，2147483647]，可以为null，默认值为MindIE Server配置文件中的maxIterTimes。
tools	可选	可能会使用的工具列表。	List[dict]类型，默认值null。
tools.type	必选	说明工具类型。	仅支持字符串"function"。
tools.function	必选	函数描述。	dict类型。
function.name	必选	函数名称。	字符串。
function.strict	可选	表示生成tool calls是否严格遵循schema格式。	bool类型，默认false。
function.description	可选	描述函数功能和使用。	字符串。
function.parameters	可选	表示函数接受的参数。	JSON schema格式。
parameters.type	必选	表示函数参数属性的类型。	字符串，仅支持object。
parameters.properties	必选	函数参数的属性。每一个key表示一个参数名，由用户自定义。value为dict类型，表示参数描述，包含type和description两个参数。	dict类型。
function.required	必选	表示函数必填参数列表。	List[string]类型。

参数	是否必选	说明	取值要求
function.additionalProperties	可选	是否允许使用未提及的额外参数。	bool类型，默认值false。 ● true：允许使用未提及的额外参数。 ● false：不允许使用未提及的额外参数。
tool_choice	可选	控制模型调用工具。	string类型或者dict类型，可以为null，默认值"auto"。 ● "none"：表示模型不会调用任何工具，而是生成一条消息。 ● "auto"：表示模型可以生成消息或调用一个或多个工具。 ● "required"：表示模型必须调用一个或多个工具。 通过{"type": "function", "function": {"name": "my_function"}}指定特定的工具，将强制模型调用该工具。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/chat/completions
```

请求消息体：

- 单轮对话

```
{
    "model": "gpt-3.5-turbo",
    "messages": [{
        "role": "user",
        "content": "You are a helpful assistant."
    }],
    "stream": false,
    "presence_penalty": 1.03,
    "frequency_penalty": 1.0,
    "repetition_penalty": 1.0,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": -1,
    "seed": null,
    "stop": ["stop1", "stop2"],
    "stop_token_ids": [2, 13],
    "include_stop_str_in_output": false,
    "skip_special_tokens": true,
    "ignore_eos": false,
```



```
    "max_tokens": 20
}
```

- 多轮对话：
  - 请求样例1：

```
{
    "model": "gpt-3.5-turbo",
    "messages": [{
        "role": "system",
        "content": "You are a student who is good at math."
        },
        {
        "role": "user",
        "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
        }
    ],
    "stream": false,
    "presence_penalty": 1.03,
    "frequency_penalty": 1.0,
    "repetition_penalty": 1.0,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": -1,
    "seed": null,
    "stop": ["stop1", "stop2"],
    "stop_token_ids": [2, 13],
    "ignore_eos": false,
    "max_tokens": 1024,
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "strict": true,
                "description": "Get the delivery date for a customer's order. Call this whenever you need to know the delivery date, for example when a customer asks 'Where is my package'",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer's order ID."
                        }
                    },
                "required": [
                    "order_id"
                ],
                "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "auto"
}
```

  - 请求样例2：

```
{
    "model": "llama_65b",
    "messages": [
        {
            "role": "system",
            "content": "You are a helpful customer support assistant. Use the supplied tools to assist the user."
        },
        {
            "role": "user",
            "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
        },
        {
            "role": "assistant",
```



```
            "tool_calls": [
                {
                    "function": {
                        "arguments": "{\"order_id\": \"12345\"}",
                        "name": "get_delivery_date"
                    },
                    "id": "tool_call_8p2Nk",
                    "type": "function"
                }
            ]
        },
        {
            "role": "tool",
            "content": "the delivery date is 2024.09.10.",
            "tool_call_id": "tool_call_8p2Nk"
        }
    ],
    "stream": false,
    "repetition_penalty": 1.1,
    "temperature": 0.9,
    "top_p": 1,
    "max_tokens": 1024,
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "strict": true,
                "description": "Get the delivery date for a customer's order. Call this whenever you need to know the delivery date, for example when a customer asks 'Where is my package'",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer's order ID."
                        }
                    },
                    "required": [
                        "order_id"
                    ],
                    "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "auto"
}
```

响应样例：

- 文本推理（“stream”=false）：
  - 单轮对话：

```
{
    "id": "chatcmpl-123",
    "object": "chat.completion",
    "created": 1677652288,
    "model": "gpt-3.5-turbo-0613",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "\n\nHello there, how may I assist you today?"
            },
            "finish_reason": "stop"
        }
    ],
    "usage": {
```



```
        "prompt_tokens": 9,
        "completion_tokens": 12,
        "total_tokens": 21
    }
}
```

- 多轮对话：
  - 请求样例1：

```
{
    "id": "chatcmpl-123",
    "object": "chat.completion",
    "created": 1677652288,
    "model": "gpt-3.5-turbo-0613",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "",
                "tool_calls": [
                    {
                        "function": {
                            "arguments": "{\"order_id\": \"12345\"}",
                            "name": "get_delivery_date"
                        },
                        "id": "call_JwmTNF3O",
                        "type": "function"
                    }
                ]
            },
            "finish_reason": "tool_calls"
        }
    ],
    "usage": {
        "prompt_tokens": 226,
        "completion_tokens": 122,
        "total_tokens": 348
    }
}
```

  - 请求样例2：

```
{
    "id": "endpoint_common_25",
    "object": "chat.completion",
    "created": 1728959154,
    "model": "llama_65b",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "\n Your order with ID 12345 is scheduled for delivery on September 10th, 2024.",
                "tool_calls": null
            },
            "finish_reason": "stop"
        }
    ],
    "usage": {
        "prompt_tokens": 265,
        "completion_tokens": 30,
        "total_tokens": 295
    }
}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

```
data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_7","object":"chat.completion.chunk","created":1729614349,"model":"llama_65b","usage":{"prompt_tokens":54,"completion_tokens":13,"total_tokens":67},"choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":"stop"}]}

data: [DONE]
```

– 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello!"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello!
```

How"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you today"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you today?"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"llama_65b","full_text":"Hello! How can I assist you today?","usage":{"prompt_tokens":31,"completion_tokens":10,"total_tokens":41},"choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you today?"},"finish_reason":"length"}]}

data: [DONE]

输出说明

表 4-6 文本推理结果说明

参数名	类型	说明
id	string	请求ID。
object	string	返回结果类型目前都返回"chat.completion"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
choices	list	推理结果列表。
index	integer	choices消息index，当前只能为0。
message	object	推理消息。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。

参数名	类型	说明
tool_calls	list	模型工具调用输出。
function	dict	函数调用说明。
arguments	string	调用函数的参数，JSON格式的字符串。
name	string	调用的函数名。
tool_calls.id	string	模型调用工具的ID。
type	string	工具的类型，目前仅支持function。
finish_reason	string	结束原因。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● tool_calls：表示模型调用了工具。
usage	object	推理结果统计数据。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。

表 4-7 流式推理结果说明

参数名	类型	说明
data	object	一次推理返回的结果。
id	string	请求ID。

参数名	类型	说明
object	string	目前都返回"chat.completion.chunk"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
full_text	string	全量文本结果，配置项“fullTextEnabled”=true时才有此返回值。
usage	object	推理结果统计数据。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。
choices	list	流式推理结果。
finish_reason	string	结束原因，只在最后一次推理结果返回。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。
index	integer	choices消息index，当前只能为0。
delta	object	推理返回结果。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。

4.2.5 兼容 Triton 接口

4.2.5.1 健康检查

接口功能

检查服务状态是否正常。服务正常时，返回状态码200，消息体没有内容。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v2/health/live

URL：https://*{ip}*:*{port}*/v2/health/ready

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/ready

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

无

使用样例

请求样例一：

```
GET https://{ip}:{port}/v2/health/live
GET https://{ip}:{port}/v2/models/llama_65b/ready
```

响应样例一：

- 服务状态正常

  无内容
- 从节点异常

  ```
  {
     "message": no contact node detected,
     "no_contact_node": ["node(10.10.10.10) is no contact"]
  }
  ```

请求样例二：

```
GET https://{ip}:{port}/v2/health/ready
```

响应样例二：

- 服务状态正常

  无内容

- 服务状态异常
  其他状态码

输出说明

- 状态码200，服务状态正常，消息体没有内容。
- 其他状态码，服务状态异常。

4.2.5.2 查询服务元数据

接口功能

查询服务元数据信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v2

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2
```

响应样例：

```
{
   "name": "MindIE Server",
   "version": "{version}",
   "extensions": {
      "max_iter_times": 512,
      "prefill_policy_type": 0,
      "decode_policy_type": 0,
      "max_prefill_batch_size": 50,
      "max_prefill_tokens": 8192
   }
}
```

响应状态码：200

输出说明

参数	类型	说明
name	string	服务名称，暂定"MindIE Server"。
version	string	服务版本。

参数	类型	说明
extensions	object	扩展字段。
max_iter_times	int	最大可进行的Decode次数。
prefill_policy_type	int	Prefill阶段的调度策略，详情请参见ScheduleConfig参数说明中的prefillPolicyType参数。
decode_policy_type	int	Decode阶段的调度策略，详情请请见ScheduleConfig参数说明中的decodePolicyType参数。
max_prefill_batch_size	int	最大prefill batch size。
max_prefill_tokens	int	最大prefill token数量。

4.2.5.3 查询模型元数据

接口功能

查询模型元数据信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2/models/llama_65b
```

响应样例：

```
{
    "name": "llama_65b",
```



```
  "platform": "MindIE Server",
  "inputs": [
    {
      "name": "input0",
      "shape": [
        -1
      ],
      "datatype": "UINT32"
    }
  ],
  "outputs": [
    {
      "name": "output0",
      "shape": [
        -1
      ],
      "datatype": "UINT32"
    }
  ]
}
```

响应状态码：200

输出说明

参数	类型	说明
name	string	模型名称。
platform	string	固定返回“MindIE Server”。
inputs	json object	表示该模型infer接口接受的inputs参数格式，目前固定返回以下内容。 `"inputs": [ { "name": "input0", "shape": [ -1 ], "datatype": "UINT32" } ]`
outputs	json object	表示该模型infer接口接受的outputs参数格式，目前固定返回以下内容。 `"outputs": [ { "name": "output0", "shape": [ -1 ], "datatype": "UINT32" } ]`

4.2.5.4 查询模型配置数据

接口功能

查询模型配置数据信息。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/config

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2/models/llama_65b/config
```

响应样例：

```
{
    "model_name": "llama_65b",
    "input_datatype": "INT64",
    "output_datatype": "INT64",
    "max_seq_len": 2560,
    "npu_mem_size": 8,
    "cpu_mem_size": 5,
    "world_size": 8,
    "model_weight_path": "llama1-65b-safetensors",
    "model_instance_type": "Standard"
}
```

响应状态码：200

输出说明

参数	类型	说明
model_name	string	推理选取的模型名字。
input_datatype	string	输入的数据类型。
output_datatype	string	输出的数据类型。
max_seq_len	int	最大序列长度。
npu_mem_size	int	NPU中可以用来申请KV Cache的size上限。
cpu_mem_size	int	CPU中可以用来申请KV Cache的size上限。

参数	类型	说明
world_size	int	使用几张卡进行推理。
model_weight_path	string	模型权重文件路径的最后一层级目录名称。
model_instance_type	string	模型类型。

4.2.5.5 Token 推理接口

接口功能

提供Token推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/infer

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

参数	是否必选	说明	取值范围
id	可选	推理请求ID。	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
inputs	必选	只有一个元素的数组。	长度为1。
inputs.name	必选	输入名称，固定"input0"。	字符串长度小于或等于256。

参数	是否必选	说明	取值范围
inputs.shape	必选	参数维度，一维时代表data长度，二维时代表1行n列，n为data长度。	data长度范围为(0, min(1024*1024, maxInputTokenLen, maxSeqLen-1, max_position_embeddings)]。其中max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
inputs.datatype	必选	data数据类型，目前场景仅支持UINT32，传递tokenid。	“UINT32”。
inputs.data	必选	数组，代表输入的tokenId值。	data长度和shape中传入的data长度一致。 tokenId的值需要在模型词表范围内。
outputs	必选	推理结果输出结构。	outputs长度需要和inputs的长度保持一致。
outputs.name	必选	推理结果输出名。	字符串。
parameters	可选	模型推理后处理相关参数。	-
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值范围
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围，(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值范围
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。
details	可选	是否返回推理详细输出结果。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	bool类型，默认值false。
perf_stat	可选	是否打开性能统计。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	bool类型，默认值false。 ● true：打开性能统计。 ● false：不打开性能统计。
batch_size	可选	推理请求batch_size。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	int类型，取值范围(0，2147483647]，默认值1。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1，5]，默认值5。 值越低优先级越高，最高优先级为1。

参数	是否必选	说明	取值范围
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/infer
```

请求消息体：

```
{
    "id": "42",
    "inputs": [{
        "name": "input0",
        "shape": [
            1,
            10
        ],
        "datatype": "UINT32",
        "data": [
            396, 319, 13996, 29877, 29901, 29907, 3333, 20718, 316, 23924
        ]
    }],
    "outputs": [{
        "name": "output0"
    }],
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "max_new_tokens": 20,
        "watermark": true,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

```
{
    "id": "42",
    "outputs": [
        {
            "name": "output0",
            "shape": [
                1,
                20
            ],
            "datatype": "UINT32",
            "data": [
                1,
                396,
                319,
                13996,
                29877,
                29901,
```



```
            29907,
            3333,
            20718,
            316,
            23924,
            562,
            2142,
            1702,
            425,
            14015,
            16060,
            316,
            383,
            19498
        ]
    }
  ]
}
```

输出说明

返回值	类型	说明
id	string	请求ID。
outputs	list	推理结果列表。
name	string	默认"output0"。
shape	list	结构为[1, n]，1表示1维数组，n表示data字段中token结果长度。
datatype	string	"UINT32"。
data	list	推理后生成的token ID集合。

4.2.5.6 文本推理接口

接口功能

提供文本推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/generate

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

参数	是否必选	说明	取值要求
id	可选	请求ID。	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
text_input	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0，18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
batch_size	可选	推理请求batch_size。	int类型，取值范围(0，2147483647]，默认值1。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，字段未设置时，默认使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 • true：带模型水印。 • false：不带模型水印。
perf_stat	可选	是否打开性能统计。	bool类型，默认值false。 • true：打开性能统计。 • false：关闭性能统计。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。

参数	是否必选	说明	取值要求
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/generate
```

请求消息体：

- 单模态文本模型：

```
{
   "id":"a123",
   "text_input": "My name is Olivier and I",
   "parameters": {
      "details": true,
      "do_sample": true,
      "max_new_tokens":20,
      "repetition_penalty": 1.1,
      "seed": 123,
      "temperature": 1,
      "top_k": 10,
      "top_p": 0.99,
      "batch_size":100,
      "typical_p": 0.5,
      "watermark": false,
      "perf_stat": false,
      "priority": 5,
      "timeout": 10
   }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "id":"a123",
   "text_input": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
         "type": "image_url",
         "image_url": "/xxxx/test.png"
      }
   ],
   "parameters": {
      "details": true,
      "do_sample": true,
      "max_new_tokens":20,
      "repetition_penalty": 1.1,
      "seed": 123,
      "temperature": 1,
      "top_k": 10,
      "top_p": 0.99,
      "batch_size":100,
      "typical_p": 0.5,
      "watermark": false,
      "perf_stat": false,
```



```
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

```
{
    "id": "a123",
    "model_name": "llama_65b",
    "model_version": null,
    "text_output": "am living in South of France.\nI have been addicted to Jurassic Park since very young. I played some video game versions but especially the great first pinball model from William which reminds me a lot of JPOG1 by song (deluxe). Unfortunately, it stopped working and has been unprofitable for a long time before being exchanged for another game. Fortunately there was the computer version. Nevertheless, it came out only on PC in 2003 when mine was too weak... It's just been a couple of months that the game came out on Mac (a whole 15 years late) with the Version 0.91JAMS ! I know this may be a little antique with the realistic animations and versions today, but the memories are very deep-seated . So thank you all rebuilders for keeping alive wonderful games like this one.\nSince then, I try to keep me updated about this game and test if possible later Alpha. Thank you so much for your work!</s>",
    "details": {
        "finish_reason": "eos_token",
        "generated_tokens": 221,
        "first_token_cost": null,
        "decode_cost": null
    }
}
```

输出说明

返回值	类型	说明
id	string	请求ID。
model_name	string	模型名称。
model_version	string	模型版本。当前未统计该数据，返回null。
text_output	string	推理返回结果。
details	object	推理details结果。

返回值	类型	说明
finish_reason	string	推理结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
first_token_cost	List[token]	文本推理返回，首token产生时间，单位：ms，当前未统计该数据，返回null。
decode_cost	int	decode时间，单位：ms，当前未统计该数据，返回null。

4.2.5.7 流式推理接口

接口功能

提供流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}:{port}***/v2/models/***${MODEL_NAME}[/versions/${MODEL_VERSION}]***/generate_stream**

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

参数	是否必选	说明	取值要求
id	可选	请求ID	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
text_input	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。

参数	是否必选	说明	取值要求
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
batch_size	可选	推理请求batch_size	int类型，取值范围(0，2147483647]，默认值1。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，字段未设置时，默认使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 • true：带模型水印。 • false：不带模型水印。

参数	是否必选	说明	取值要求
perf_stat	可选	是否打开性能统计。	bool类型，默认值false。 ● true：打开性能统计。 ● false：不打开性能统计。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/generate_stream
```

请求消息体：

- 单模态文本模型：

```
{
   "id":"a123",
   "text_input": "My name is Olivier and I",
   "parameters": {
      "details": true,
      "do_sample": true,
      "max_new_tokens":20,
      "repetition_penalty": 1.1,
      "seed": 123,
      "temperature": 1,
      "top_k": 10,
      "top_p": 0.99,
      "batch_size":100,
      "typical_p": 0.5,
      "watermark": false,
      "perf_stat": false,
      "priority": 5,
      "timeout": 10
   }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "id":"a123",
   "text_input": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
         "type": "image_url",
         "image_url": "/xxxx/test.png"
      }
```



```
    ],
    "parameters": {
        "details": true,
        "do_sample": true,
        "max_new_tokens":20,
        "repetition_penalty": 1.1,
        "seed": 123,
        "temperature": 1,
        "top_k": 10,
        "top_p": 0.99,
        "batch_size":100,
        "typical_p": 0.5,
        "watermark": false,
        "perf_stat": false,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

- 响应样例1：

```
data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"am","details":{"generated_tokens":1,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"prefill_time":26.96,"decode_time":null}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" passion","details":{"generated_tokens":2,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":11},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"ate","details":{"generated_tokens":3,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":9},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" about","details":{"generated_tokens":4,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":11},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" music","details":{"generated_tokens":5,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":".","details":{"generated_tokens":6,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":12},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"\n","details":{"generated_tokens":7,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"T","details":{"generated_tokens":8,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":19},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"od","details":{"generated_tokens":9,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":12},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"ay","details":{"generated_tokens":10,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":30},"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"</s>","details":{"finish_reason":"eos_token","generated_tokens":424,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":5067},"prefill_time":null,"decode_time":16.80}
```

- 响应样例2（配置项“fullTextEnabled”=true）：

```
data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-Chat","model_version":null,"text_output":"'m","details":{"generated_tokens":1,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":5092
```



```
},"prefill_time":41.68000030517578,"decode_time":null}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from","details":
{"generated_tokens":2,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":43},"
prefill_time":null,"decode_time":20.440000534057617}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from France","details":
{"generated_tokens":3,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":27},"
prefill_time":null,"decode_time":12.175999641418457}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from France.","details":
{"generated_tokens":4,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":26},"
prefill_time":null,"decode_time":12.128000259399414}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from France. I","details":
{"finish_reason":"length","generated_tokens":5,"first_token_cost":null,"decode_cost":null,"batch_size":1,
"queue_wait_time":26},"prefill_time":null,"decode_time":12.458000183105469}
```

输出说明

返回值	类型	说明
data	object	一次推理返回的结果。
id	string	请求ID。
model_name	string	模型名称。
model_version	string	模型版本。
text_output	string	推理返回结果。
finish_reason	string	推理结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
details	object	推理details结果。

返回值	类型	说明
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
first_token_cost	List[token]	文本推理返回，首token产生时间，单位：ms，当前未统计该数据，返回null。
decode_cost	int	decode时间，单位：ms，当前未统计该数据，返回null。
batch_size	int	流式推理batch size。
queue_wait_time	int	队列等待时间，单位：us。
prefill_time	float	首token时延，单位：ms。
decode_time	float	非首token的token时延，单位：ms。

4.2.6 MindIE 原生接口

4.2.6.1 Slot 统计接口

接口功能

参考Triton格式，自定义的slot统计信息查询接口。

接口格式

操作类型：**GET**

URL：https://*{ip}:{port}***/v2/models/***${MODEL_NAME}[/versions/${MODEL_VERSION}]***/getSlotCount**

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2/models/llama_65b/getSlotCount
```

响应样例：

```
{
    "total_slots": 50,
    "free_slots": 50,
    "available_tokens_length": 30720
}
```

响应状态码：200

输出说明

返回值	类型	说明
total_slots	int	推理服务支持的最大batch_size，取值为配置文件中maxBatchSize字段。
free_slots	int	当前剩余slots字段，通过调度模块管理的参数获取。
available_tokens_length	int	KV Cache剩余的可放token数。

4.2.6.2 提前终止请求接口

说明

目前该接口只支持提前终止Triton推理请求。

接口功能

参考Triton接口定义，提供提前终止请求接口。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/stopInfer

说明

- *{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

请求参数

参数	是否必选	说明	取值要求
id	必选	推理请求ID。	长度不超过256的非空字符串。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/stopInfer
```

请求消息体：

```
{
   "id":"a123"
}
```

响应样例：

```
{
   "id":"a123"
}
```

响应状态码：200

输出说明

返回值	类型	说明
id	string	成功停止推理请求ID。

4.2.6.3 文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/infer

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数名	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 • true：流式推理。 • false：文本推理。
parameters	可选	模型推理后处理相关参数。	-

参数名	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 • true：做sampling。 • false：不做sampling。

参数名	是否必选	说明	取值要求
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/infer
```

请求消息体：

- 单模态文本模型：

```
{
    "inputs": "My name is Olivier and I",
    "stream": false,
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "max_new_tokens": 20,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "details": true,
        "typical_p": 0.5,
        "watermark": false,
        "priority": 5,
        "timeout": 10
    }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
    "inputs": [
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "stream": false,
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "max_new_tokens": 20,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "details": true,
        "typical_p": 0.5,
        "watermark": false,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

- 文本推理（“stream”=false）：

```
{
    "generated_text": "am a french native speaker. I am looking for a job in the hospitality industry. I",
    "details": {
        "finish_reason": "length",
        "generated_tokens": 20,
        "seed": 846930886
    }
}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

```
data: {"prefill_time":45.54,"decode_time":null,"token":{"id":626,"text":"am"}}

data: {"prefill_time":null,"decode_time":128.32,"token":{"id":263,"text":" a"}}
```



```
data: {"prefill_time":null,"decode_time":18.17,"token":{"id":5176,"text":" French"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":17739,"text":" photograph"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":261,"text":"er"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":2729,"text":" based"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":297,"text":" in"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":3681,"text":" Paris"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29889,"text":"."}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":13,"text":"\n"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29902,"text":"I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":505,"text":" have"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1063,"text":" been"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":27904,"text":" shooting"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1951,"text":" since"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":306,"text":" I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":471,"text":" was"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29871,"text":" "}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29896,"text":"1"}}

data: {"prefill_time":null,"decode_time":16.80,"generated_text":"am a French photographer based in Paris.\nI have been shooting since I was 15","details":{"finish_reason":"length","generated_tokens":20,"seed":846930886},"token":{"id":29945,"text":null}}
```

- 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data: {"prefill_time":33.55400085449219,"decode_time":null,"token":{"id":5122,"text":"：  "}}

data: {"prefill_time":null,"decode_time":19.922000885009766,"token":{"id":7552,"text":"：  ）"}}

data: {"prefill_time":null,"decode_time":12.803999900817871,"token":{"id":40,"text":"：  ）I"}}

data: {"prefill_time":null,"decode_time":11.869000434875488,"token":{"id":2776,"text":"：  ）I'm"}}

data: {"prefill_time":null,"decode_time":12.008000373840332,"generated_text":"：  ）I'm ","details":null,"token":{"id":4102,"text":null}}
```

输出说明

表 4-8 文本推理结果说明

返回值	类型	说明
generated_text	string	推理返回结果。
details	object	推理details结果。目前定义以下字段，支持扩展。

返回值	类型	说明
finish_reason	string	推理结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回该seed值。

表 4-9 流式推理结果说明

返回值	类型	说明
data	object	一次推理返回的结果。
prefill_time	float	流式推理下首token时延，单位：ms。
decode_time	float	流式推理下非首token的token时延，单位：ms。
generated_text	string	推理文本结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，支持扩展。

返回值	类型	说明
finish_reason	string	推理结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回改seed值。
token	List[token]	每一次推理的token。
id	int	token ID。
text	string	token对应文本。

4.2.6.4 Token 推理接口

接口功能

实现token输入的文本/流式推理。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/infer_token

说明

*{ip}*和*{port}*请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数名	是否必选	说明	取值要求
input_id	必选	数组，代表输入的token ID值。	tokenId的值需要在模型词表范围内。token ID取值范围为(1, 1024*1024]。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 ● true：流式推理。 ● false：文本推理。
parameters	可选	模型推理后处理相关参数。	-
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。

参数名	是否必选	说明	取值要求
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数名	是否必选	说明	取值要求
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/infer_token
```

请求消息体：

```
{
    "input_id": [5618, 19678, 701, 9072, 13],
    "stream": false,
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "max_new_tokens": 20,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "details": true,
        "typical_p": 0.5,
        "watermark": false,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

- 文本推理（“stream”=false）：

  ```
  {
      "generated_text": "am a french native speaker. I am looking for a job in the hospitality industry. I",
      "details": {
          "finish_reason": "length",
          "generated_tokens": 20,
          "seed": 846930886
      }
  }
  ```

- 流式推理（“stream”=true，使用sse格式返回）：

  ```
  data: {"prefill_time":45.54,"decode_time":null,"token":{"id":626,"text":"am"}}

  data: {"prefill_time":null,"decode_time":128.32,"token":{"id":263,"text":" a"}}

  data: {"prefill_time":null,"decode_time":18.17,"token":{"id":5176,"text":" French"}}

  data: {"prefill_time":null,"decode_time":16.80,"token":{"id":17739,"text":" photograph"}}

  data: {"prefill_time":null,"decode_time":16.80,"token":{"id":261,"text":"er"}}

  data: {"prefill_time":null,"decode_time":16.80,"token":{"id":2729,"text":" based"}}
  ```



```
data: {"prefill_time":null,"decode_time":16.80,"token":{"id":297,"text":" in"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":3681,"text":" Paris"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29889,"text":"."}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":13,"text":"\n"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29902,"text":"I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":505,"text":" have"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1063,"text":" been"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":27904,"text":" shooting"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1951,"text":" since"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":306,"text":" I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":471,"text":" was"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29871,"text":" "}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29896,"text":"1"}}

data: {"prefill_time":null,"decode_time":16.80,"generated_text":"am a French photographer based in Paris.\nI have been shooting since I was 15","details":
{"finish_reason":"length","generated_tokens":20,"seed":846930886},"token":{"id":29945,"text":null}}
```

输出说明

表 4-10 文本推理结果说明

返回值	类型	说明
generated_text	string	推理返回结果。
details	object	推理details结果。目前定义以下字段，支持扩展。

返回值	类型	说明
finish_reason	string	推理结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回该seed值。

表 4-11 流式推理结果说明

返回值	类型	说明
data	object	一次推理返回的结果。
prefill_time	float	流式推理下首token时延，单位：ms。
decode_time	float	流式推理下非首token的token时延，单位：ms。
generated_text	string	推理文本结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，支持扩展。

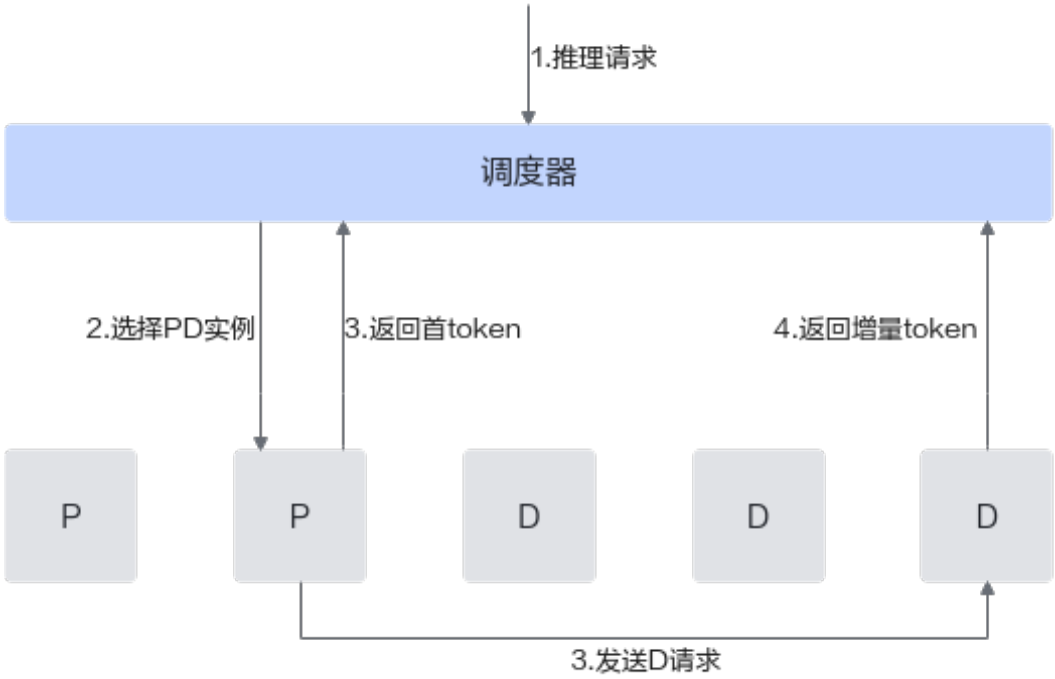
返回值	类型	说明
finish_reason	string	推理结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回改seed值。
token	List[token]	每一次推理的token。
id	int	token ID。
text	string	token对应文本。

4.2.7 PD 分离相关接口

4.2.7.1 PD 分离架构说明

该架构将推理的2个阶段（Prefill和Decode）分别部署到不同的实例上进行计算，从而减少Prefill和Decode之间互相干扰，提升推理性能。其基本流程如下：

图 4-1 PD 分离架构基本流程



PD分离是集群化架构，集群中，存在三种必要的元素，如下所示：

- 调度器
- Prefill实例（P）
- Decode实例（D）

其中，调度器负责对外发布推理接口，P、D负责各自推理阶段的计算。调度器（Coordinator）是由MindIE MS实现；P和D实例是由MindIE Server实现。

基于该架构，MindIE Server不再对终端用户直接开放接口，本章节主要描述MindIE Server对MindIE MS开放的接口。

4.2.7.2 推理接口

说明

- 推理接口属于业务面，使用业务面的IP地址和端口号。
- 推理接口主要分为请求和响应两部分。

请求

当调度器接收到外部请求后，会基于集群状态，计算用于服务该请求的PD实例。然后将请求发送给对应的P实例。

由于标准推理（PD混部）对外提供了五类推理接口兼容（分别是TGI、vLLM、OpenAI、Triton和MindIE原生接口），因而，在PD分离场景下，调度器需要兼容这5类接口。当调度器收到外部请求（RESTful请求）后，会向P实例转发本次请求，请求的URL和Body与PD混部保持一致，但需要在HTTP HEADER中加入部分控制信息，控制信息请参见**表4-12**。

表 4-12 P 请求 HTTP HEADER 字段说明

字段名	是否必选	字段描述	取值类型
req-id	必选	全局唯一的RequestID。	数字类型的字符串，要求全局唯一。
d-target	必选	目标D实例的IP地址。	IP地址。
req-type	必选	请求类型。	固定值：prefill。
is-recompute	可选（重计算场景下必选）	是否重计算。	• true：重计算 • false：非重计算

请求样例：

```
curl -i -H "req-id: 12345" -H "req-type: prefill" -H "d-target: 172.17.0.9" -H "Accept: application/json" -H "Content-type: application/json" -X POST -d '
{
    "model": "llama2_7b",
    "messages": [{
        "role": "system",
        "content": "You are a helpful assistant."
    }],
    "max_tokens": 20,
    "presence_penalty": 1.03,
    "frequency_penalty": 1.0,
    "seed": null,
    "temperature": 0.5,
    "top_p": 0.95,
    "stream": true
}' https://172.17.0.10:1025/v1/chat/completions
```

响应

一个请求的响应会分别发生在P实例和D实例中，且两实例的响应内容不同。

- **P实例响应：**

  响应体如下所示：

  ```
  {
      "reqId": "12345",
      "isStream": true,
      "output": "data:{...}",
      "isLastResp": false
  }
  ```

  响应体各字段说明请参见表4-13。

  表 4-13 P 实例响应字段说明

字段名	是否必选	字段描述	取值类型
reqId	必选	针对RequestID的响应。	数字类型的字符串。

字段名	是否必选	字段描述	取值类型
isStream	必选	本次请求是否流式请求。	bool类型 ● true：流式请求。 ● false：非流式请求。
output	可选	一次请求的响应内容。	字符串。
isLastResp	流式请求和不经过D节点的非流式请求必选	本次响应是否为最后一次响应。	bool类型 ● true：本次响应为最后一次响应。 ● false：本次响应为不是最后一次响应。

由于调度器不解析外部请求的内容（仅转发注入HTTP HEADER之后的请求），对于部分请求，无法确定是否流式（部分请求的流式/非流式在URL中能体现，但部分请求的流式/非流式只有Body中才有，目前的调度器实现中，并没有解析Body）。而针对请求是否流式，调度器对外返回的形式会有区别，因而，在P实例的响应中，直接告知是否流式。

调度器需要将推理结果响应给外部请求者，但响应的内容来自推理实例。在P响应中，响应的内容在output字段中体现。调度器不需要解析output的内容，仅需要使用合适的方式（流式、非流式，首token、唯一token、不同的接口类型等因素，会造成调度器的不同行为）返回给请求者即可。

- 针对流式请求，P实例的output中会体现出一次token（首token）的返回内容，以Trition的响应为例，output的内容如下：
  ```
  data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"am","details":{"generated_tokens":1,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"prefill_time":26.96,"decode_time":null}
  ```
- 针对非流式请求，P实例不会返回首token，一般没有output字段。
- 存在一种特殊情况，即整个推理只返回一个token。此时，请求会在P实例终结。isLastResp就是用于标识这种特殊情况。当isLastResp为true时，非流式请求的output会有本次推理的完整结果。

- **D实例响应**：

  调度器与D实例间，存在一个长连接，D实例每推理出一个结果，就通过该长连接响应给调度器。

  操作类型：**GET**

  URL：**https://*{ip}*:*{port}*/dresult**

  请求样例：

  ```
  GET https://{ip}:{port}/dresult
  ```

  以chunk类型响应，HTTP头中包含下面2个字段：

  ```
  Content-Type: text/event-stream
  Transfer-Encoding: chunked
  ```

  单次的返回结果由2部分组成，分别是size和data。

  其中，size是用16进制表示的字符串（不需要带0x），表示data的长度。data就是实际传输的内容。size和data都以\r\n结尾。

  ```
  18\r\n{"aaa":"bbb","ccc":"ddd"}\r\n
  ```


说明

- 18是24的16进制表达。
- 上面对HTTP chunk的原理说明，开发阶段不需要关注，HTTP库已经对其进行封装。

下面结合实例请求进行描述。

D实例向调度器返回的数据就是指chunk data部分，分为数据类响应和控制类响应。

- **数据类响应**
  - **普通token**

    普通token的响应包括request ID和token，二者以\0分隔，并且以\0结尾。其结构如下：

    ```
    reqId:<request id>\0data:<token>\0
    ```

    <request id>：即Coordinator为请求生成的request ID。

    <token>：是一次计算的返回值，下面是个例子：

    ```
    data:
    {"id":"a123","model_name":"llama2_7b","model_version":null,"text_output":"","details":{"finish_reason":"eos_token","generated_tokens":80,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":195}}
    ```

    完整示例如下：

    ```
    reqId:123456678\0data:data:
    {"id":"a123","model_name":"llama2_7b","model_version":null,"text_output":"","details":{"finish_reason":"eos_token","generated_tokens":80,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":195}}\0
    ```

    说明

    示例中有2个**data:**，第一个**data:**是报文格式中的关键字，第二个**data:**是<token>的内容。

    由于这里的例子是Triton，所以<token>的内容中有**data:**，在vLLM接口中，没有第二个**data:**。

  - **last token**

    last token的主体形式与普通token基本一致，区别在于token部分的前缀为lastData。完整示例如下：

    ```
    reqId:123456678\0lastData:data:
    {"id":"a123","model_name":"llama2_7b","model_version":null,"text_output":"","details":{"finish_reason":"eos_token","generated_tokens":80,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":195}}\0
    ```

    在Coordinator侧，如果收到普通token，直接把token部分返回给用户即可，如果收到last token，则需要中断与用户之间的chunk连接。

    说明

    流式请求只有一个响应体，属于last token。

  - **推理错误**

    当D实例的推理过程出现错误时（例如pull KV Cache失败），需要及时返回错误信息，以便Coordinator终止请求。错误消息的格式如下：

    ```
    reqId:<request id>\0error:<错误原因>\0
    ```

    整个消息体与普通token的返回类似，包含2部分：

    - 第一部分是request ID，其格式与普通token一样。
    - 第二部分是错误信息，以***error:***为前缀。***error:***后描述错误原因。

示例如下：

```
reqId:12345678\0error:pull kv cache failed.\0
```

- **控制类响应**

  - **保活响应**

    由于是长连接，如果D实例发生故障长期没有响应，Coordinator无法区分是否真的没有响应。因而，这里设计一个保活响应，如果一分钟都没有发送普通响应，就发送一个保活响应，保活响应的格式如下：

    ```
    ka:\0
    ```

  - **重计算响应**

    当D实例的负载过高，造成资源死锁后，需要打断一个请求，并使其进行重计算。重计算时，需要通知Coordinator重新调度，将请求发送到负载相对较低的实例。重计算的响应格式如下：

    ```
    reqId:<request id>\0<响应体>\0
    ```

    此结构与token的响应类似，也分为2部分，第一部分是reqId，第二部分是重计算的*<响应体>*。为与token响应区分，这里的*<响应体>*以***retry:***开头，示例如下：

    ```
    retry:{"id":"a123","text_input":"1,1619,1024,338,19802,631,322,306,11","parameters":{"details":true,"do_sample":true,"max_new_tokens":200,"repetition_penalty":1.100000023841858,"seed":123,"temperature":1.0,"top_k":2147,"top_p":0.9900000095367432,"batch_size":100}}
    ```

    retry:后面的部分是一个完整的推理请求body。Coordinator拿到这个body后，重新进行调度，并发给新的P、D实例。

    说明

    此body需要根据不同的请求类型进行组装，此处以是Triton文本接口的body为例。

    其中，text_input部分是tokenId的序列，包含了当前推理出来的所有结果。在发送这个响应前，需要保证这个tokenId的序列中的所有token都已经返回给Coordinator。

4.2.7.3 指定实例身份接口

接口功能

指定实例身份。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v1/role/*{role}*

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

表 4-14 URL 字段说明

参数	是否必选	类型	描述
role	必选	字符串	指定实例的身份。可选值： • prefill • decode

表 4-15 Body 字段说明

参数	是否必选	类型	描述
local	必选	结构体	表示本实例自己的信息。
local.server_ip	必选	字符串	本节点的IP地址。
local.device	必选	结构体数组	本节点NPU卡相关信息，可能存在多卡。
local.device_id	必选	整型字符串	NPU卡ID。
local.device_ip	必选	字符串	NPU卡IP地址。
peers	可选	结构体数组	表示需要建连的对象列表。每个实例只看到与自己有关的实例。 当目标是D时，对应的是其需要连接的P。 当目标是P时，对应的是会跟其连接的D。
peers.server_ip	可选	字符串	对端节点的IP地址。
peers.device	可选	结构体数组	对端节点NPU卡相关信息，可能存在多卡。卡数量必需与本节点的卡数量相同。
peers.device_id	可选	整型字符串	NPU卡ID。
peers.device_ip	可选	字符串	NPU卡IP地址。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/role/prefill
```

请求消息体：

```
{
  "local": {
    "server_ip": "1.2.3.4",
```



```
   "device": [{
       "device_id": "0",
       "device_ip": "xxx.xxx.xxx.xxx"
     },
     {
       "device_id": "1",
       "device_ip": "xxx.xxx.xxx.xxx"
     }
   ]
 },
 "peers": [{
   "server_ip": "5.6.7.8",
   "device": [{
       "device_id": "0",
       "device_ip": "xxx.xxx.xxx.xxx"
     },
     {
       "device_id": "1",
       "device_ip": "xxx.xxx.xxx.xxx"
     }
   ]
 }]
}
```

响应样例：

- 成功：
  ```
  {
      "result": "OK",
  }
  ```
- 失败：
  ```
  {
  "error": "xxx",
  "error_type": "xxxxx"
  }
  ```

响应状态码：

- 200：成功。
- 400：错误请求。
- 422：请求体非法。

输出说明

参数	类型	描述
result	string	成功。
error	string	错误描述。
error_type	string	错误类型。

4.2.7.4 静态配置采集接口

接口功能

采集静态配置。

接口格式

操作类型：**GET**

URL：**https://***{ip}*:*{port}*/**v1/config**

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/config
```

响应样例：

```
{
    "modelName": "llama_65b",
    "maxSeqLen": 2560,
    "npuMemSize": 8,
    "cpuMemSize": 5,
    "worldSize": 8,
    "maxOutputLen": 512,
    "cacheBlockSize": 128
}
```

输出说明

参数	类型	说明
modelName	string	推理选取的模型名字。
maxSeqLen	uint32_t	最大序列长度。
npuMemSize	uint32_t	单个NPU中可以用来申请KV Cache的size上限。
cpuMemSize	uint32_t	CPU中可以用来申请KV Cache的size上限。
worldSize	uint32_t	使用几张卡进行推理。
maxOutputLen	uint32_t	最大输出长度。
cacheBlockSize	uint32_t	KV Cache block的size大小。

4.2.7.5 动态状态采集接口

接口功能

采集动态状态。

接口格式

操作类型：**GET**

URL：**https://*{ip}*:*{port}*/v1/status**

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/status
```

响应样例：

```
{
   "service": {
      "roleStatus": "RoleUnknown",
      "currentRole": "prefill"
   },
   "resource": {
      "availSlotsNum": 200,
      "availBlockNum": 1024
   },
   "peers": ["1.2.3.4", "5.6.7.8"]
}
```

输出说明

参数	是否必选	类型	描述
service	必选	结构体	service相关属性。
resource	必选	结构体	当前剩余资源情况。

参数	是否必选	类型	描述
peers	可选	字符串数组	数组成员为IP地址，表示已经成功建连的P实例。 ● PD混部场景，或PD分离场景下的P实例不需要返回该字段。 ● PD分离场景下的D实例需要返回该字段。 ● D实例每成功连接一个P后，就向这里增加一个成员。在目标P没有发生变化的情况下，如果跟某个P发生了pull kv失败且不可恢复的情况，就把该P从这里删掉。
service.roleStatus	必选	字符串	表示当前服务的身份切换状态，可选值包括： ● RoleUnknown ● RoleSwitching ● RoleReady 如果是PD混部场景，该状态永远是RoleReady。
service.currentRole	必选	字符串	当前实例的身份，可选值包括： ● prefill ● decode ● none 如果是PD混部场景，该值永远是none。 在PD分离场景下，如果roleStatus进入RoleReady状态，则该值为prefill或decode，如果roleStatus不是RoleReady状态，该值为none。
resource.availSlotsNum	必选	整型	可用的slot数量。
resource.availBlockNum	必选	整型	可用的block数量。

4.2.7.6 计算 token 数量接口

接口功能

计算token数量。

接口格式

操作类型：**POST**

URL：**https://***{ip}*:*{port}*/**v1/tokenizer**

说明

{ip}和{port}请使用业务面的IP地址和端口号，即“ipAddress”和“port”。

请求参数

参数	类型	说明
inputs	string	待计算token数量的输入字符串。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/tokenizer
```

请求消息体：

```
{
    "inputs": "xxxxxx"
}
```

响应样例：

```
{
    "token_number": 1234,
    "tokens": ["abc", "xyz", ...]
}
```

输出说明

参数	是否必选	类型	描述
token_number	必选	int	token的数量。
tokens	必选	字符串数组	解析出来的token列表，如果一个中文由多个token组成，这里会将多个token合并成一个中文展示。因此，这个数组的数量可能比token_number的值小。

4.3 EndPoint 管理面接口

4.3.1 服务指标接口（JSON 格式）

接口功能

获取推理服务过程中请求的TTFT、TBT的动态平均值（默认近1000个请求的平均值），正在执行请求数、正在等待请求数量、剩余NPUblock数量。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/metrics-json

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/metrics-json
```

响应样例：

```
{
   "resultType": "vector",
   "result": [
      {
         "metric": [
            {
               "__name__": "TTFT",
               "job": "node",
               "instance": "127.0.0.2:1026"
            }
         ],
         "value": "0"
      },
      {
         "metric": [
            {
               "__name__": "TBT",
               "job": "node",
               "instance": "127.0.0.2:1026"
            }
         ],
         "value": "0"
      },
      {
         "metric": [
            {
               "__name__": "waitingInferRequestNum",
               "job": "node",
               "instance": "127.0.0.2:1026"
            }
         ],
         "value": "0"
      },
      {
         "metric": [
            {
               "__name__": "processingInferRequestNum",
               "job": "node",
               "instance": "127.0.0.2:1026"
            }
         ],
         "value": "0"
      },
```



```
    {
        "metric": [
            {
                "__name__": "remainBlocks",
                "job": "node",
                "instance": "127.0.0.2:1026"
            }
        ],
        "value": "1024"
    }
  ]
}
```

响应状态码：200

输出说明

参数	类型	说明
resultType	string	结果类型，默认为vector，包含查询结果的数组。
result	list	结果列表。
metric	list	五种服务指标结果。 ● TTFT：首token时延。 ● TBT：生成连续两个token之间的时间。 ● processingInferRequestNum：TBT正在执行请求数。 ● waitingInferRequestNum：正在等待请求数量。 ● remainBlocks：剩余NPUblock数量。
__name__	string	服务指标的名称。
job	string	服务指标作业名称。
instance	string	服务指标实例地址。
value	string	服务指标结果具体值。

4.3.2 优雅退出接口

接口功能

实现整个服务的优雅退出。调用该接口时，会等待服务中正在执行和等待的所有请求完成，并关闭服务，等待时所有推理接口将不可用。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/stopService

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/stopService
```

响应状态码：200

输出说明

无

4.3.3 健康探针接口

接口功能

检查推理流程是否正常。

接口格式

操作类型：**GET**

URL：https://*{ip}*:*{port}*/health/timed-*${TIMEOUT}***

说明

- {ip}字段优先读取环境变量值MIES_CONTAINER_MANAGEMENT_IP；如果没有该环境变量，则取配置文件的“managementIpAddress”参数；如果配置文件中没有“managementIpAddress”参数，则取配置文件的“ipAddress”参数。
- {port}字段优先读取配置文件的“managementPort”参数；如果配置文件中没有“managementPort”参数，则取配置文件的“port”参数。
- {TIMEOUT}需指定探针的超时时间，范围为[1, 600]的整数，单位：秒。如果不填，则默认值为10。

请求参数

无

使用样例

请求样例1：

```
GET https://{ip}:{port}/health/timed
```


请求样例2：

```
GET https://{ip}:{port}/health/timed-200
```

响应样例：

```
{
   "status": "healthy"
}
```

输出说明

- 状态码200，服务状态正常，消息体没有内容。
- 其他状态码，服务状态异常。

4.3.4 服务监控指标查询接口（普罗格式）

说明

- 当使用HTTP协议方式使用该接口的时候，请保护该接口不被终端用户直接调用，并且在局域网内使用。
- 如需使用该接口，请确保在启动服务前开启服务化监控开关。开启服务化监控功能的命令如下：
  ```
  export MIES_SERVICE_MONITOR_MODE=1
  ```

接口功能

查询推理服务化的相关服务监控指标。

接口格式

操作类型：**GET**

URL：http://*{ip}*:*{port}*/metrics

说明

*{ip}*和*{port}*请使用管理面的IP地址和指标端口号，即“managementIpAddress”和“metricsPort”。

请求参数

无

使用样例

请求样例：

```
GET http://{ip}:{port}/metrics
```

响应样例：

```
# HELP exposer_transferred_bytes_total Transferred bytes to metrics services
# TYPE exposer_transferred_bytes_total counter
exposer_transferred_bytes_total 7901
# HELP exposer_scrapes_total Number of times metrics were scraped
# TYPE exposer_scrapes_total counter
exposer_scrapes_total 1
# HELP exposer_request_latencies Latencies of serving scrape requests, in microseconds
```



```
# TYPE exposer_request_latencies summary
exposer_request_latencies_count 1
exposer_request_latencies_sum 282
exposer_request_latencies{quantile="0.5"} Nan
exposer_request_latencies{quantile="0.9"} Nan
exposer_request_latencies{quantile="0.99"} Nan
# HELP request_received_total Number of requests received so far.
# TYPE request_received_total counter
request_received_total{model_name="llama2-7b"} 3188
# HELP request_success_total Number of responses sent so far
# TYPE request_success_total counter
request_success_total{model_name="llama2-7b"} 2267
# HELP request_failed_total Number of responses failed so far
# TYPE request_failed_total counter
request_failed_total{model_name="llama2-7b"} 0
# HELP prompt_tokens_total Number of prefill tokens processed.
# TYPE prompt_tokens_total counter
prompt_tokens_total{model_name="llama2-7b"} 9564
# HELP generation_tokens_total Number of generation tokens processed.
# TYPE generation_tokens_total counter
generation_tokens_total{model_name="llama2-7b"} 84425
# HELP avg_prompt_throughput_toks_per_s Average prefill throughput in tokens/s.
# TYPE avg_prompt_throughput_toks_per_s gauge
avg_prompt_throughput_toks_per_s{model_name="llama2-7b"} 0.586739718914032
# HELP avg_generation_throughput_toks_per_s Average generation throughput in tokens/s.
# TYPE avg_generation_throughput_toks_per_s gauge
avg_generation_throughput_toks_per_s{model_name="llama2-7b"} 2.375296831130981
# HELP failed_request_perc Requests failure rate. 1 means 100 percent usage.
# TYPE failed_request_perc gauge
failed_request_perc{model_name="llama2-7b"} 0
# HELP npu_cache_usage_perc NPU KV-cache usage. 1 means 100 percent usage.
# TYPE npu_cache_usage_perc gauge
npu_cache_usage_perc{model_name="llama2-7b"} 1
# HELP cpu_cache_usage_perc CPU KV-cache usage. 1 means 100 percent usage.
# TYPE cpu_cache_usage_perc gauge
cpu_cache_usage_perc{model_name="llama2-7b"} 0
# HELP time_to_first_token_seconds Histogram of time to first token in seconds.
# TYPE time_to_first_token_seconds histogram
time_to_first_token_seconds_count{model_name="llama2-7b"} 2523
time_to_first_token_seconds_sum{model_name="llama2-7b"} 9740.00200343132
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.001"} 0
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.005"} 0
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.01"} 0
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.02"} 0
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.04"} 0
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.06"} 10
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.08"} 54
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.1"} 104
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.25"} 256
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.5"} 256
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="0.75"} 276
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="1"} 321
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="2.5"} 628
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="5"} 1148
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="7.5"} 2523
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="10"} 2523
time_to_first_token_seconds_bucket{model_name="llama2-7b",le="+Inf"} 2523
# HELP time_per_output_token_seconds Histogram of time per output token in seconds.
# TYPE time_per_output_token_seconds histogram
time_per_output_token_seconds_count{model_name="llama2-7b"} 85800
time_per_output_token_seconds_sum{model_name="llama2-7b"} 4445.857012826018
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.01"} 0
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.025"} 0
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.05"} 3
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.075"} 12
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.1"} 40283
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.15"} 83145
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.2"} 83339
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.3"} 83339
```



```
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.4"} 83539
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.5"} 85139
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="0.75"} 85740
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="1"} 85800
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="2.5"} 85800
time_per_output_token_seconds_bucket{model_name="llama2-7b",le="+Inf"} 85800
# HELP e2e_request_latency_seconds Histogram of end to end request latency in seconds.
# TYPE e2e_request_latency_seconds histogram
e2e_request_latency_seconds_count{model_name="llama2-7b"} 2267
e2e_request_latency_seconds_sum{model_name="llama2-7b"} 12684.5319980979
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="1"} 27
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="2.5"} 268
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="5"} 712
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="10"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="15"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="20"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="30"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="40"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="50"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="60"} 2267
e2e_request_latency_seconds_bucket{model_name="llama2-7b",le="+Inf"} 2267
# HELP request_prompt_tokens Number of prefill tokens processed.
# TYPE request_prompt_tokens histogram
request_prompt_tokens_count{model_name="llama2-7b"} 3188
request_prompt_tokens_sum{model_name="llama2-7b"} 9564
request_prompt_tokens_bucket{model_name="llama2-7b",le="10"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="50"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="100"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="200"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="500"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="1000"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="2000"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="5000"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="10000"} 3188
request_prompt_tokens_bucket{model_name="llama2-7b",le="+Inf"} 3188
# HELP request_generation_tokens Number of generation tokens processed.
# TYPE request_generation_tokens histogram
request_generation_tokens_count{model_name="llama2-7b"} 2267
request_generation_tokens_sum{model_name="llama2-7b"} 84425
request_generation_tokens_bucket{model_name="llama2-7b",le="10"} 0
request_generation_tokens_bucket{model_name="llama2-7b",le="50"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="100"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="200"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="500"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="1000"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="2000"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="5000"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="10000"} 2267
request_generation_tokens_bucket{model_name="llama2-7b",le="+Inf"} 2267
```

响应状态码：200

输出说明

参数	类型	说明
exposer_transferred_bytes_total	Counter	服务监控指标查询接口返回的总字节数。 **说明** 后续版本将会删除，不建议使用。
exposer_scrapes_total	Counter	服务监控指标查询接口被抓取的总次数。 **说明** 后续版本将会删除，不建议使用。

参数	类型	说明
exposer_request_latencies	Summary	服务监控指标查询接口抓取请求的时延，单位为微秒。 ● exposer_request_latencies_count：抓取请求的总次数。 ● exposer_request_latencies_sum：抓取请求的总延迟时间，单位为微秒。 ● exposer_request_latencies：抓取请求时延的摘要数据。 quantile：分位数。 **说明** 后续版本将会删除，不建议使用。
request_received_total	Counter	服务端到目前为止接收到的推理请求个数。 model_name：使用的模型名称。string类型，如果有多个模型，请使用"&"进行拼接。 **说明** 响应样例中的所有model_name都为此含义。
request_success_total	Counter	服务端到目前为止推理成功的请求个数。
request_failed_total	Counter	服务端到目前为止推理失败的请求个数。
prompt_tokens_total	Counter	已经处理的prefill tokens数量。
generation_tokens_total	Counter	已经处理的generation tokens数量。
avg_prompt_throughput_toks_per_s	Gauge	截止上一个请求完成，最新的平均prefill吞吐，单位为tokens/s。
avg_generation_throughput_toks_per_s	Gauge	截止上一个token生成，最新的平均generation吞吐，单位为tokens/s。
failed_request_perc	Gauge	服务端到目前为止执行失败的请求率。 失败的请求率=执行失败的请求个数/接收到的请求个数，1代表100%。
npu_cache_usage_perc	Gauge	当前KV Cache的NPU显存利用率，1代表100%。
cpu_cache_usage_perc	Gauge	当前KV Cache的NPU显存利用率，1代表100%。

参数	类型	说明
time_to_first_token_seconds	Histogram	首token时延，代表请求生成首个推理token消耗的时间，单位为秒。 ● time_to_first_token_seconds_count：截止目前，完成并统计首token时延的请求个数。 ● time_to_first_token_seconds_sum：截止目前，完成并统计首token时延的所有请求的首token时延的加和。 ● time_to_first_token_seconds_bucket：截止目前，直方图分桶统计的请求的首token时延数据。 le：less than or equal to，是直方图分桶的界限。 **说明** 响应样例中的所有le都为此含义。
time_per_output_token_seconds	Histogram	token生成时延，代表连续两个token生成之间的时间间隔，单位为秒。 ● time_per_output_token_seconds_count：截止目前为止，完成并统计token生成时延的token个数。 ● time_per_output_token_seconds_sum：截止目前为止，完成并统计token生成时延的所有token的token生成时延的加和。 ● time_per_output_token_seconds_bucket：截止目前为止，直方图分桶统计的token生成时延数据。
e2e_request_latency_seconds	Histogram	端到端时延，代表请求从接收到执行完成消耗的时间，单位为秒。 ● e2e_request_latency_seconds_count：截止目前为止，完成并统计端到端时延的请求个数。 ● e2e_request_latency_seconds_sum：截止目前为止，完成并统计端到端时延的所有请求的端到端时延的加和。 ● e2e_request_latency_seconds_bucket：截止目前为止，直方图分桶统计的请求的端到端时延数据。

参数	类型	说明
request_prompt_tokens	Histogram	请求输入的token数量，代表请求输入的prompt经过tokenzier之后得到的token个数。 ● request_prompt_tokens_count：截止目前为止，完成并统计当前指标的请求个数。 ● request_prompt_tokens_sum：截止目前为止，完成并统计当前指标的所有请求的输入的token数量的加和。 ● request_prompt_tokens_bucket：截止目前为止，直方图分桶统计的请求的输入的token数量数据。
request_generation_tokens	Histogram	请求输出的token数量，代表请求经过模型推理之后得到的token个数。 ● request_generation_tokens_count：截止目前为止，完成并统计当前指标的请求个数。 ● request_generation_tokens_sum：截止目前为止，完成并统计当前指标的所有请求的输出的token数量的加和。 ● request_generation_tokens_bucket：截止目前为止，直方图分桶统计的请求的输出的token数量数据。

5 性能调优

5.1 性能调优流程

通过参数调优，使吞吐率（TPS）达到时延约束条件下的最大值。

CPU 性能优化

在裸机中执行以下命令开启CPU高性能模式和透明大页，开启后可提升性能，建议开启。

- 开启CPU高性能模式，在相同时延约束下，TPS会有~3%的提升。
  ```
  cpupower -c all frequency-set -g performance
  ```
- 开启透明大页，多次实验的吞吐率结果会更稳定。
  ```
  echo always > /sys/kernel/mm/transparent_hugepage/enabled
  ```

性能调优环境变量配置

性能调优环境变量配置如**表5-1**所示。

表 5-1 性能调优环境变量

参数名称	默认值	说明	推荐值
MIES_USE_MB_SWAPPER	0	开启高性能Swap（不设置时默认关闭）；Atlas 推理系列产品上需要关闭该参数。 ● 0：关闭。 ● 1：开启。	maxPreemptCount >0 时，表示使用Swap功能，此时建议开启高性能Swap。 export MIES_USE_MB_SWAPPER=1

参数名称	默认值	说明	推荐值
MIES_RECOMPUTE_THRESHOLD	0.5	表示当前可下发的请求block数占总block数的比例（也就是block资源利用率）。 以0.5为例，当前可下发的请求资源利用率小于0.5时，就会触发重计算，释放少量请求的block，来保证其他请求资源使用。阈值范围只能是[0,1)，值越大越容易触发重计算。0表示所有请求都无法下发。	取值建议在0.5上下浮动调整。 export MIES_RECOMPUTE_THRESHOLD=0.5

最优性能参数配置

最优性能配置各参数说明及取值如表5-2所示。

表 5-2 最优性能参数配置

配置类型	配置项	配置介绍	推荐配置
调度配置	maxPrefillBatchSize	Prefill阶段一个batch中包含请求个数的上限。	小于等于maxBatchSize的值，建议设置为：maxBatchSize/2 ，若显存溢出可适当调小。
	maxPrefillTokens	Prefill阶段一个batch中包含input token总数的上限。	maxPrefillBatchSize * 数据集token ID平均输入长度。 不建议设置过大，若显存溢出可适当调小。
	maxBatchSize	Decode阶段一次推理包含请求的最大个数。	1. 根据ScheduleConfig参数说明 maxBatchSize参数中的公式计算出最大值。 2. 如果需要限制Decode时延，可适当调整maxBatchSize大小，一般情况maxBatchSize越小，Decode时延越小。

配置类型	配置项	配置介绍	推荐配置
	supportSelectBatch	• false：关闭，表示优先执行Prefill。 • true：开启，优化stage执行优先级；根据prefillTimeMsPerReq和decodeMsPerReq数值动态优化，prefillTimeMsPerReq设置越高，Prefill被优先执行的概率越低，也就是Prefill会等到多轮Decode后再执行。	• 吞吐优先时，建议设置为：true。 • 首token时延优先时，建议设置为：false。
	prefillTimeMsPerReq	平均每个请求Prefill时间。 “supportSelectBatch”设置为“true”时生效。	建议值：600，单位ms；若需要降低首token时延可适当调小。
	decodeTimeMsPerReq	平均每个请求Decode时间。 “supportSelectBatch”设置为“true”时生效。	建议值：50，单位为ms。
	maxQueueDelayMicroseconds	Prefill组Batch时，最大等待时长。	建议值：500，单位为us。
	maxPreemptCount	每一批次最大可抢占请求的上限，即限制一轮调度最多抢占请求的数量，最大上限为maxBatchSize，取值大于0则表示开启可抢占功能。	[0, maxBatchSize]，当取值大于0时，cpuMemSize取值不可为0。 建议值：0（关闭）。
模型配置	worldSize	节点可以使用的NPU卡数。	根据用户实际环境情况启用NPU卡数量。
	npuDeviceIds	推理使用的一组NPU卡号。	[0, 1, 2, ..., worldSize-1]

配置类型	配置项	配置介绍	推荐配置
	npuMemSize	单个NPU中可以用来申请KV Cache的size上限，单位GB。	npuMemSize=(单卡总空闲-权重/NPU卡数-后处理占用)*系数，其中系数取0.8。 通常情况下，大模型推理主要是显存bound，因此该值配置的越大，KV Cache可用的显存越多，BatchSize就越大，吞吐量将会更优。 **说明** 在一些小模型场景下，显存充足，主要是计算bound，调大显存效果并不明显。
	cpuMemSize	单个CPU中可以用来申请KV Cache的size上限。单位：GB。 开启Swap时生效，如何开启请参考**性能调优环境变量配置**中的MIES_USE_MB_SWAPPER环境变量。	上限根据显存和用户需求来决定。只有当maxPreemptCount为0时，才可以取值为0。 建议值：5。
	cacheBlockSize	表示一个block块的大小；NPU显存会被分成一个一个的block。 例如配置128，表示一个block实际大小为128*sizof（cache数据类型）字节。如果相同的显存，设置的block size越小，那么block num越多。	根据请求平均输入输出大小确定，一般默认为128，如果平均输入较小可以适当调小。

配置类型	配置项	配置介绍	推荐配置
其他配置	logLevel	设置日志级别。 • "Verbose"：打印Verbose、Info、Warning和Error级别的日志。 • "Info"：打印Info、Warning和Error级别的日志。 • "Warning"：打印Warning和Error级别的日志。 • "Error"：打印Error级别的日志。 • "None"：不打印日志。	建议值："Error"，打印Error级别的日志。

操作步骤

以下操作步骤以LLaMa3-8B双卡，数据类型bfloat16为例，进行最优性能的配置，环境信息举例如下：

本机配置8张显存大小为32G的卡，每张卡空闲状态下已占用3G，以占用2张卡进行操作。

步骤1 计算模型配置参数，请参考表5-2中的“npuMemSize”参数计算出“npuMemSize”的值并根据计算结果调整配置中的该值，计算过程如下所示。

1. 计算模型的权重大小，进入模型权重文件所在目录，使用“du -h”命令查看模型的权重大小。如图5-1所示，LLaMa3-8B模型权重大小为15G。

   图 5-1 查看权重大小

   ```
   root      :/home/data/acltransformer_testdata/weights/LLaMA3-8B# du -h
   15G     .
   ```

2. 计算“npuMemSize”的值，计算公式为：Floor[(单卡显存-空闲占用-权重/NPU卡数)* 系数]，系数取值为0.8。

   npuMemSize = Floor[(32 - 3 - 15/2)] * 0.8 = 17G。

   说明

   “Floor”表示计算结果向下取整。

根据计算结果，配置示例如下所示：

```
"ModelDeployConfig":
  {
     "maxSeqLen" : 2560,
     "maxInputTokenLen" : 2048,
```



```
    "truncation" : false,
    "ModelConfig" : [
        {
            "modelInstanceType": "Standard",
            "modelName" : "LLaMa3-8B",
            "modelWeightPath" : "/home/data/acltransformer_testdata/weights/LLaMa3-8B",
            "worldSize" : 2,
            "cpuMemSize" : 5,
            "npuMemSize" : 17,
            "backendType": "atb"
        }
    ]
},
```

说明

- “worldSize”、“npuDeviceIds”、“cacheBlockSize”、“npuMemSize”和“cpuMemSize”参数请参见**表5-2**取建议值。
- 其他参数取值请参考**8.4.2 配置参数说明**取默认值。

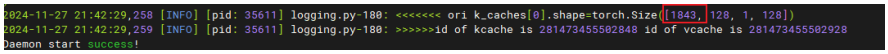
步骤2 计算调度配置参数，请参考**ScheduleConfig参数说明**中的“maxBatchSize”参数计算出“maxBatchSize”的值并根据计算结果调整配置中的该值，计算过程如下所示。

根据计算公式：maxBatchSize = Total Block Num/Block Num，需要先计算出“Total Block Num”和“Block Num”的值。

1. 计算“Total Block Num”的值。
   - 方式一（实测获取）：
     - “Total Block Num”的值可以通过跑一次性能后在Python日志中的“npuBlockNum”获取。

     - “Total Block Num”的值也可以直接从info级打屏日志k_caches[0].shape=torch.Size([npuBlockNum, -, -, -])中torch.Size的第一个值获取。

     说明

     Python日志路径：/usr/local/Ascend/mindie/latest/mindie-llm/logs/pythonlog.log
   - 方式二（公式计算）：

     Total Block Num = Floor[NPU显存/(模型网络层数*cacheBlockSize*模型注意力头数*注意力头大小*Cache类型字节数*Cache数)]，公式中各参数的取值信息如**表5-3**所示。

表 5-3 Total Block Num 公式中的参数值

参数	取值
NPU显存	**步骤1**中“npuMemSize”的值：17。
模型网络层数	模型网络层数是模型权重文件config.json中num_hidden_layers参数的值，LLaMa3-8B的模型网络层数取值为：32。

参数	取值
cacheBlockSize	默认值：128。该参数的值与**ScheduleConfig参数说明**中cacheBlockSize参数的值保持一致。
模型注意力头数	模型注意力头数是模型权重文件config.json中num_attention_heads参数的值。
注意力头大小	“模型注意力头数*注意力头大小”的值为模型权重文件config.json中hidden_size参数的值，即注意力头大小可以根据模型注意力头数计算获得。 **说明** 对于GQA类模型（分组查询注意力类模型，例如LLaMa3-8B），需使用模型权重文件config.json中num_key_value_heads参数的值而不是num_attention_heads参数的值作为“模型注意力头数”的值参与计算，即对于两卡LLaMa3-8B，每张卡上的“模型注意力头数*注意力头大小”的值应该为8*128/2=512。
Cache类型字节数	由模型config.json文件中的torch_dtype决定，一般为float16类型，取值为：2。
Cache数	Key+Value的值，各自代表1个Cache数。默认值：2。

将以上参数值代入公式，得到Total Block Num = Floor[17*1024*1024*1024/(32 * 128 * 8*128/2*2*2)] = 2176。

注：以上算式中128/2是由于LLaMa3-8B双卡，所以注意头数需均分在2张卡上。

2. 计算每个请求所需“Block Num”的值，公式中各参数的取值信息如**表5-4**所示。

   根据计算公式：

   - 所需最大Block Num = Ceil(输入Token数/cacheBlockSize)+Ceil(最大输出Token数/cacheBlockSize)
   - 所需最小Block Num = Ceil(输入Token数/cacheBlockSize)
   - 所需平均Block Num = Ceil(输入Token数/cacheBlockSize)+Ceil(平均输出Token数/cacheBlockSize)

表 5-4 Block Num 公式中的参数值

参数	取值
输入Token数	从Benchmark输出的InputTokens参数的平均值获取，取值示例：186。
最大输出Token数	从config.json文件中的maxIterTimes参数获取，示例取值：512。
平均输出Token数	实际运行测试数据集后统计GeneratedTokens参数平均输出长度，示例取值：346。

参数	取值
cacheBlockSize	默认值：128。该参数的值与**ScheduleConfig参数说明**中cacheBlockSize的值保持一致。

将以上参数值代入公式：

- 所需最小Block Num = Ceil(186/128) = 2
- 所需最大Block Num = Ceil(186/128)+Ceil(512/128) = 6
- 所需平均Block Num = Ceil(186/128)+Ceil(346/128) = 5

说明

“Ceil”表示计算结果向上取整。

3. 计算“maxBatchSize”的取值范围。

   根据**步骤2.1**和**步骤2.2**计算出的“Total Block Num”和“Block Num”值，然后使用公式maxBatchSize=Floor[Total Block Num/Block Num]计算“maxBatchSize”的取值范围。

   - 最小maxBatchSize = Floor[Total Block Num/所需最大Block Num] = 362
   - 最大maxBatchSize = Floor[Total Block Num/所需最小Block Num] = 1088
   - 平均maxBatchSize = Floor[Total Block Num/所需平均Block Num] = 435

   由以上公式得到“maxBatchSize”的取值范围为[362，1088]，设置初始值为**435**，然后根据吞吐量或者时延要求进行调整，具体场景请参见**最佳实践**。

步骤3 计算“maxPrefillBatchSize”和“maxPrefillTokens”的值。

- “maxPrefillBatchSize”的计算方式请参见**表5-2**中“maxPrefillBatchSize”参数，建议设置为：“maxBatchSize”值的一半。

  maxPrefillBatchSize = Floor[maxBatchSize/2] = 435/2 = 217

- “maxPrefillTokens”的值一般不超过8192，其计算方式请参见**表5-2**中“maxPrefillTokens”参数。

  maxPrefillTokens = maxPrefillBatchSize * 数据集token ID平均输入长度 = 217*186 = 40362

  根据公式计算出的值大于8192，所以“maxPrefillTokens”的取值为8192。

说明

- “maxPrefillBatchSize”和“maxPrefillTokens”的值一般根据“maxBatchSize”的值进行调整，其值不建议过大。
- “maxPrefillTokens”一般不用超过8192，若显存溢出可进一步适当调小“maxPrefillBatchSize”和“maxPrefillTokens”的值。

步骤4 通过配置“maxPreemptCount”和“cpuMemSize”参数确认Swap抢占，请参见**表5-2**配置其建议值。

说明

如果是显存受限场景，可开启“maxPreemptCount”（即设置为1或2），“cpuMemSize”可从5调整至40。

步骤5 通过配置“supportSelectBatch”、“prefillTimeMsPerReq”和“decodeTimeMsPerReq”参数确认Prefill/Decode切换调度策略。

- 当严格要求首token时延时：

  “supportSelectBatch”设置为“false”。

- 当严格要求吞吐量时（首token时延要求适中）：

  “supportSelectBatch”设置为“true”。

  “prefillTimeMsPerReq”和“decodeTimeMsPerReq”按照模型实际平均首token时延和Decode时延设置，也可参见表5-2使用推荐配置，然后根据下列场景进行调优。

  - 场景一：若希望降低首token时延：

    可调小“prefillTimeMsPerReq”，并调大“decodeTimeMsPerReq”，使Prefill优先执行。

  - 场景二：若希望提升吞吐量（首token时延要求较小）：

    可适当调大“prefillTimeMsPerReq”，并调小“decodeTimeMsPerReq”，使Decode优先执行。

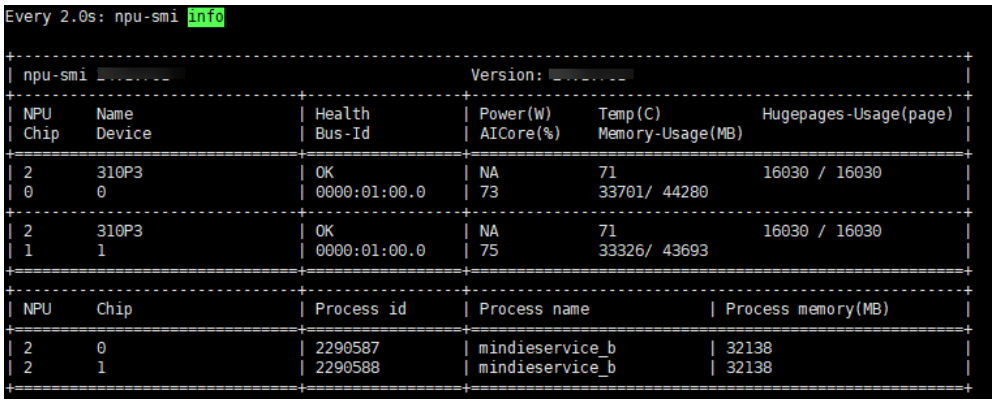
步骤6 实际运行时，若测试场景是显存bound，可进一步调整“npuMemSize”的值。

在调试过程中，重开一个窗口使用以下命令查看占卡情况，如果剩余空间还很大，可以调大npuMemSize的值，重复步骤2~步骤5后再次进行调试。

```
watch npu-smi info
```

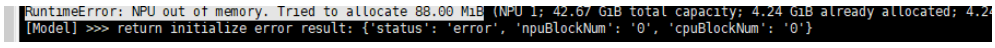
- 当npuMemSize的值不够大时，可以继续调大空闲值，如图5-2。

  图 5-2 npuMemSize 值不够大

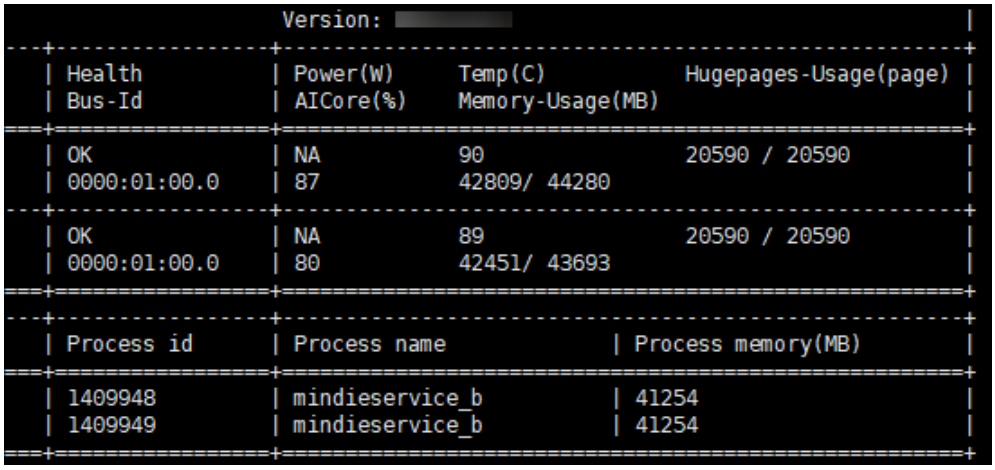
- 当npuMemSize的值过大时，则会报“Npu out of memory”错误，如图5-3所示。

  图 5-3 npuMemSize 过大的情况

- 根据“Npu out of memory”报错信息，将npuMemSize的值调小（比如在原来的值上减2，避免卡死），则可以达到最优npuMemSize的值。如图5-4，大概是占据95%卡的状态。

图 5-4 npuMemSize 最优值



----结束

须知

测试性能的数据集问题条数尽量超过1000条以上（可参考MindIE Benchmark支持的开源数据集CEval、MMLU，均在1000条以上），否则平均性能会有较大波动。

5.2 最佳实践

5.2.1 不考虑时延的极限吞吐

不考虑时延的极限吞吐的调试方式如下所示。

- 服务端：
  - “maxBatchSize”尽量调大到显存限制，一般情况下“maxBatchSize”值越大，则吞吐越大；若“maxBatchSize”增大时吞吐量不增反降，则停止调整。
  - 设置supportSelectBatch为true，“prefillTimeMsPerReq”和“decodeTimeMsPerReq”按照模型实际平均首token时延和Decode时延进行设置。
- 客户端：
  - 按并发数发送请求：客户端Concurrency的值通常配置为maxBatchSize的值减1。
  - 按频率发送请求：则Concurrency可设置为1000，请求发送频率根据实际业务场景或按模型实际QPS设置。

操作步骤

步骤1 使用以下命令启动服务，以当前所在Ascend-mindie-service_*{version}*_linux-*{arch}*目录为例。

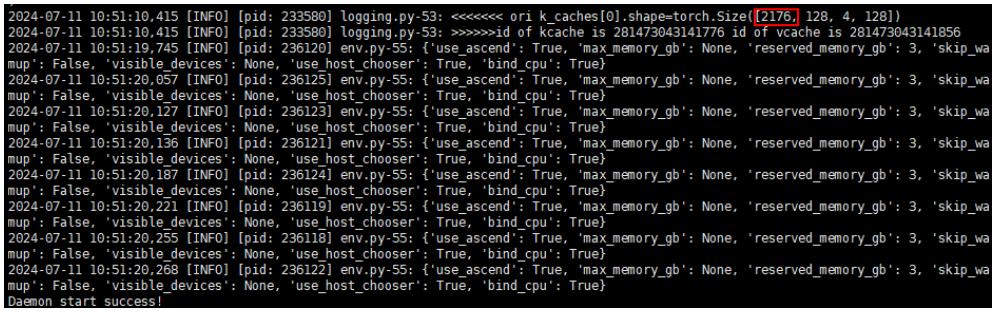
```
./bin/mindieservice_daemon
```

回显如下则说明启动成功。

```
Daemon start success!
```

服务启动后，可通过info级打屏日志k_caches[0].shape=torch.Size([npuBlockNum, x, x, x])中torch.Size的第一个值获取npuBlockNum的值，如图5-5所示，与步骤2.1中计算出来的值一致。

图 5-5 启动成功



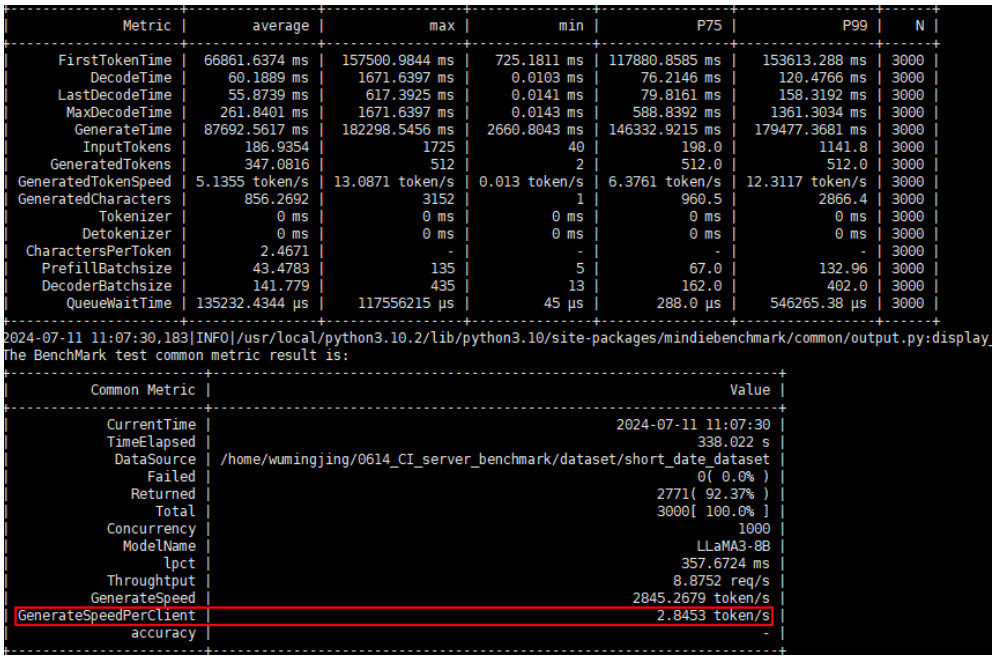
步骤2 根据步骤2.3计算出“maxBatchSize”的取值范围为[362，1088]，设置初始值为**435**；“maxPrefillBatchSize”参数的值设置为“maxBatchSize”值的一半，取值为**217**。

步骤3 配置完成后，用户可使用HTTPS客户端（Linux curl命令，Postman工具等）发送HTTPS请求，此处以Linux curl命令为例进行说明。

重开一个窗口，使用以下命令发送请求，获取当前的吞吐量（GenerateSpeedPerClient），如图5-6所示，此时吞吐量为2.8453 token/s。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \          //数据集路径
--DatasetType "gsm8k" \                                  //数据集类型
--ModelName LLaMa3-8B \                                  //模型名称
--ModelPath "/{模型路径}/LLaMa3-8B" \                    //模型路径
--TestType client \                                      //模式选择
--Http https://{ipAddress}:{port} \                      //请求url
--ManagementHttp https://{managementIpAddress}:{managementPort} \          //管理面url
--Concurrency 1000 \                                     //并发数
--TaskKind stream \                                      //Client不同推理模式，此处为文本流式推理
--Tokenizer True \                                       //分词向量化标识
--MaxOutputLen 512                                       //最大输出长度
```

图 5-6 maxBatchSize 取值为 435 的吞吐量



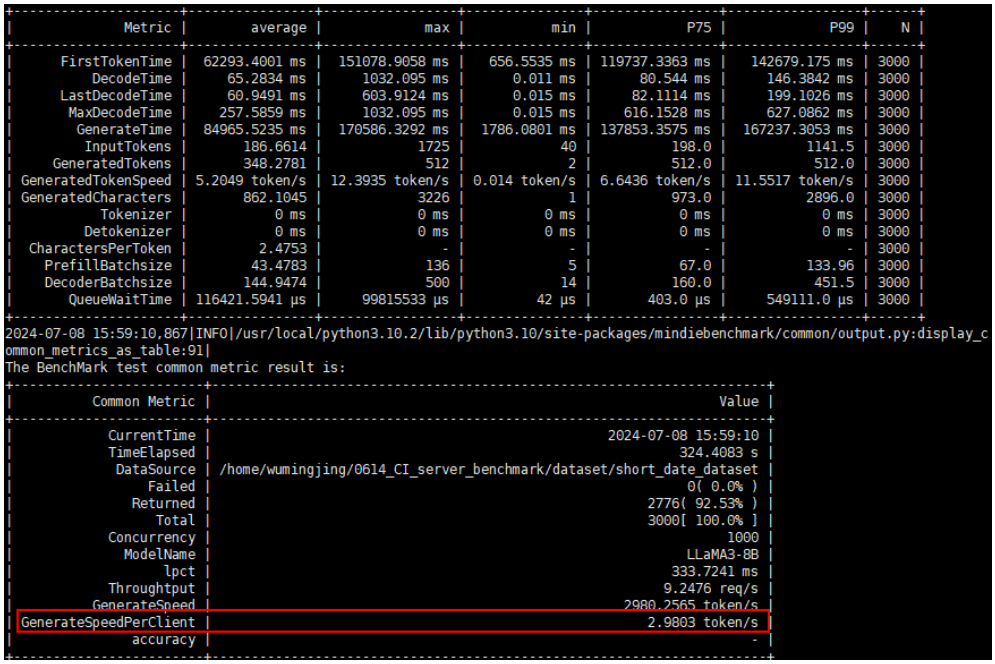
Metric	average	max	min	P75	P99	N
FirstTokenTime	66861.6374 ms	157500.9844 ms	725.1811 ms	117880.8585 ms	153613.288 ms	3000
DecodeTime	60.1889 ms	1671.6397 ms	0.0103 ms	76.2146 ms	120.4766 ms	3000
LastDecodeTime	55.8739 ms	617.3925 ms	0.0141 ms	79.8161 ms	158.3192 ms	3000
MaxDecodeTime	261.8401 ms	1671.6397 ms	0.0143 ms	588.8392 ms	1361.3034 ms	3000
GenerateTime	87692.5617 ms	182298.5456 ms	2660.8043 ms	146332.9215 ms	179477.3681 ms	3000
InputTokens	186.9354	1725	40	198.0	1141.8	3000
GeneratedTokens	347.0816	512	2	512.0	512.0	3000
GeneratedTokenSpeed	5.1355 token/s	13.0871 token/s	0.013 token/s	6.3761 token/s	12.3117 token/s	3000
GeneratedCharacters	856.2692	3152	1	960.5	2866.4	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4671	-	-	-	-	3000
PrefillBatchsize	43.4783	135	5	67.0	132.96	3000
DecoderBatchsize	141.779	435	13	162.0	402.0	3000
QueueWaitTime	135232.4344 μs	117556215 μs	45 μs	288.0 μs	546265.38 μs	3000

2024-07-11 11:07:30,183|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-11 11:07:30
TimeElapsed	338.022 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2771(92.37%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	357.6724 ms
Throughtput	8.8752 req/s
GenerateSpeed	2845.2679 token/s
GenerateSpeedPerClient	2.8453 token/s
accuracy	-

步骤4 以“100”为单位且取整向上调试“maxBatchSize”的值，所以设置“maxBatchSize”的值为**500**，“maxPrefillBatchSize”参数的值设置为**250**。然后执行**步骤3**，继续观察不考虑时延的极限吞吐，执行结果如**图5-7**所示，此时吞吐量为2.9803 token/s。

图 5-7 maxBatchSize 取值为 500 的吞吐量



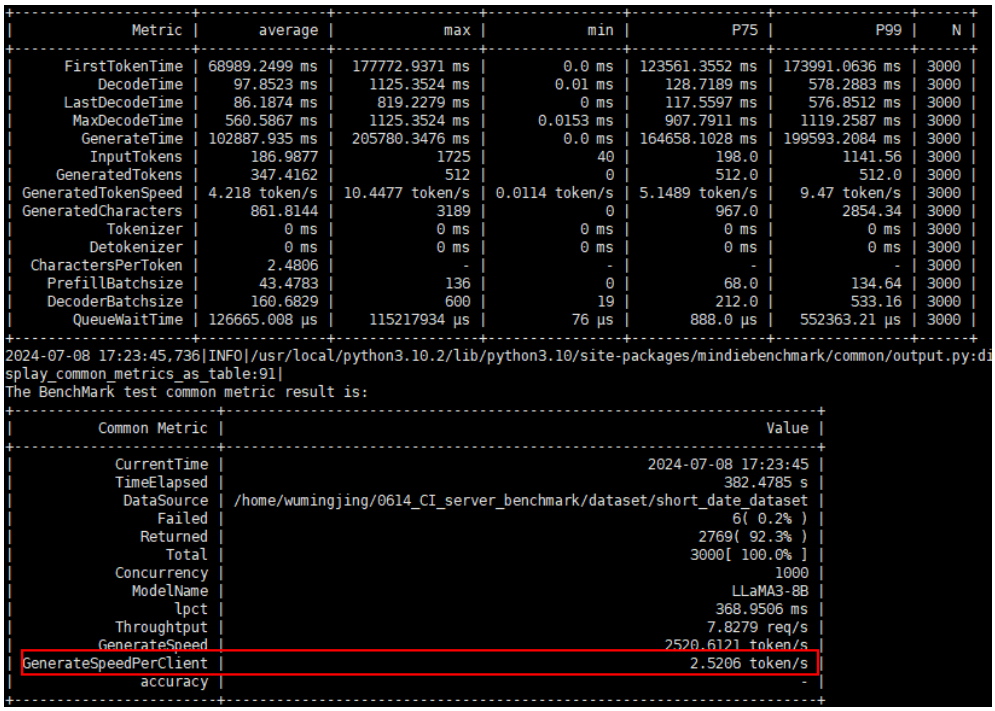
Metric	average	max	min	P75	P99	N
FirstTokenTime	62293.4001 ms	151078.9058 ms	656.5535 ms	119737.3363 ms	142679.175 ms	3000
DecodeTime	65.2834 ms	1032.095 ms	0.011 ms	80.544 ms	146.3842 ms	3000
LastDecodeTime	60.9491 ms	603.9124 ms	0.015 ms	82.1114 ms	199.1026 ms	3000
MaxDecodeTime	257.5859 ms	1032.095 ms	0.015 ms	616.1528 ms	627.0862 ms	3000
GenerateTime	84965.5235 ms	170586.3292 ms	1786.0801 ms	137853.3575 ms	167237.3053 ms	3000
InputTokens	186.6614	1725	40	198.0	1141.5	3000
GeneratedTokens	348.2781	512	2	512.0	512.0	3000
GeneratedTokenSpeed	5.2049 token/s	12.3935 token/s	0.014 token/s	6.6436 token/s	11.5517 token/s	3000
GeneratedCharacters	862.1045	3226	1	973.0	2896.0	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4753	-	-	-	-	3000
PrefillBatchsize	43.4783	136	5	67.0	133.96	3000
DecoderBatchsize	144.9474	500	14	160.0	451.5	3000
QueueWaitTime	116421.5941 μs	99815533 μs	42 μs	403.0 μs	549111.0 μs	3000

2024-07-08 15:59:10,867|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_c
ommon_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-08 15:59:10
TimeElapsed	324.4083 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2776(92.53%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	333.7241 ms
Throughtput	9.2476 req/s
GenerateSpeed	2980.2565 token/s
GenerateSpeedPerClient	2.9803 token/s
accuracy	-

步骤5 由于“maxBatchSize”的值为**500**的吞吐量优于“maxBatchSize”的值为**435**，所以继续设置“maxBatchSize”的值为**600**，“maxPrefillBatchSize”参数的值为**300**。然后执行**步骤3**，观察其吞吐量，执行结果如**图5-8**所示，此时吞吐量为2.5206 token/s。

图 5-8 maxBatchSize 取值为 600 的吞吐量



```
+--------------------+--------------+----------------+---------------+---------------+---------------+------+
|             Metric |      average |            max |           min |           P75 |           P99 |    N |
+--------------------+--------------+----------------+---------------+---------------+---------------+------+
|      FirstTokenTime | 68989.2499 ms |  177772.9371 ms |        0.0 ms | 123561.3552 ms | 173991.0636 ms | 3000 |
|          DecodeTime |    97.8523 ms |    1125.3524 ms |       0.01 ms |    128.7189 ms |    578.2883 ms | 3000 |
|      LastDecodeTime |    86.1874 ms |     819.2279 ms |          0 ms |    117.5597 ms |    576.8512 ms | 3000 |
|       MaxDecodeTime |   560.5867 ms |    1125.3524 ms |     0.0153 ms |    907.7911 ms |   1119.2587 ms | 3000 |
|        GenerateTime | 102887.935 ms |  205780.3476 ms |        0.0 ms | 164658.1028 ms | 199593.2084 ms | 3000 |
|         InputTokens |      186.9877 |           1725 |            40 |          198.0 |        1141.56 | 3000 |
|     GeneratedTokens |      347.4162 |            512 |             0 |          512.0 |          512.0 | 3000 |
| GeneratedTokenSpeed | 4.218 token/s | 10.4477 token/s | 0.0114 token/s | 5.1489 token/s |   9.47 token/s | 3000 |
| GeneratedCharacters |      861.8144 |           3189 |             0 |          967.0 |        2854.34 | 3000 |
|           Tokenizer |         0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 3000 |
|         Detokenizer |         0 ms |          0 ms |          0 ms |          0 ms |          0 ms | 3000 |
|  CharactersPerToken |       2.4806 |              - |             - |             - |             - | 3000 |
|    PrefillBatchsize |      43.4783 |            136 |             0 |          68.0 |         134.64 | 3000 |
|    DecoderBatchsize |      160.6829 |            600 |            19 |         212.0 |         533.16 | 3000 |
|       QueueWaitTime | 126665.008 µs |   115217934 µs |        76 µs |      888.0 µs |   552363.21 µs | 3000 |
+--------------------+--------------+----------------+---------------+---------------+---------------+------+
2024-07-08 17:23:45,736|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:di
splay_common_metrics_as_table:91|
The BenchMark test common metric result is:
+----------------------+-----------------------------------------------------------------------+
|         Common Metric |                                                                 Value |
+----------------------+-----------------------------------------------------------------------+
|           CurrentTime |                                                   2024-07-08 17:23:45 |
|           TimeElapsed |                                                             382.4785 s |
|            DataSource | /home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset |
|                Failed |                                                          6( 0.2% ) |
|              Returned |                                                       2769( 92.3% ) |
|                 Total |                                                       3000[ 100.0% ] |
|           Concurrency |                                                                1000 |
|             ModelName |                                                            LLaMA3-8B |
|                  lpct |                                                          368.9506 ms |
|           Throughtput |                                                           7.8279 req/s |
|          GenerateSpeed |                                                      2520.6121 token/s |
| GenerateSpeedPerClient |                                                         2.5206 token/s |
|              accuracy |                                                                   - |
+----------------------+-----------------------------------------------------------------------+
```

当“maxBatchSize”的值为**600**时，此时的吞吐量明显下降，停止调高“maxBatchSize”的值。

综上所述，当“maxBatchSize”的值在500左右时，可达到极限吞吐。如需获取达到极限吞吐时更精准的“maxBatchSize”值，请根据以上操作步骤继续调试。

----结束

5.2.2 限制非首 token 时延的极限吞吐

以Decode平均时延限制50ms以内为目标，限制非首token时延的极限吞吐的调试方式如下所示。

- 服务端：
  - “maxBatchSize”调小到卡对应的时延，一般情况下“maxBatchSize”越小，则Decode时延越小。
  - 设置supportSelectBatch为true，“prefillTimeMsPerReq”和“decodeTimeMsPerReq”按照模型实际平均首token时延和Decode时延进行设置。
- 客户端：
  - 按并发数发送请求：客户端Concurrency通常配置为maxBatchSize-1。
  - 按频率发送请求：则Concurrency可设置为1000，请求发送频率根据实际业务场景或按模型实际QPS设置。

操作步骤

步骤1 使用以下命令启动服务，以当前所在Ascend-mindie-service_*{version}*_linux-*{arch}*目录为例。

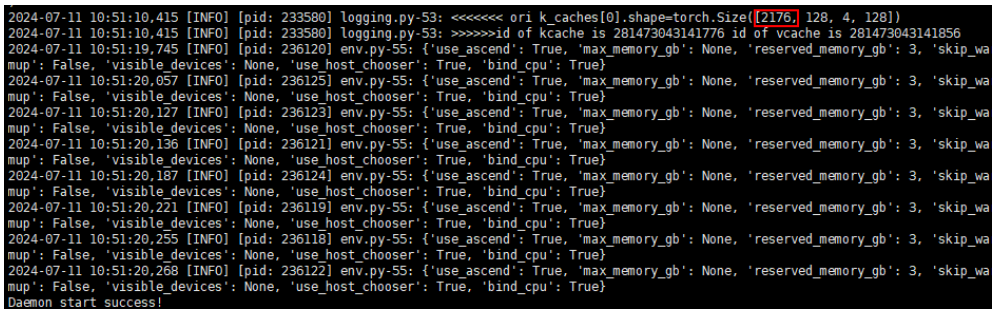
```
./bin/mindieservice_daemon
```


回显如下则说明启动成功。

```
Daemon start success!
```

服务启动后，可通过info级打屏日志k_caches[0].shape=torch.Size([npuBlockNum, x, x, x])中torch.Size的第一个值获取npuBlockNum的值，如图5-9所示，与步骤2.1中计算出来的值一致。

图 5-9 启动成功



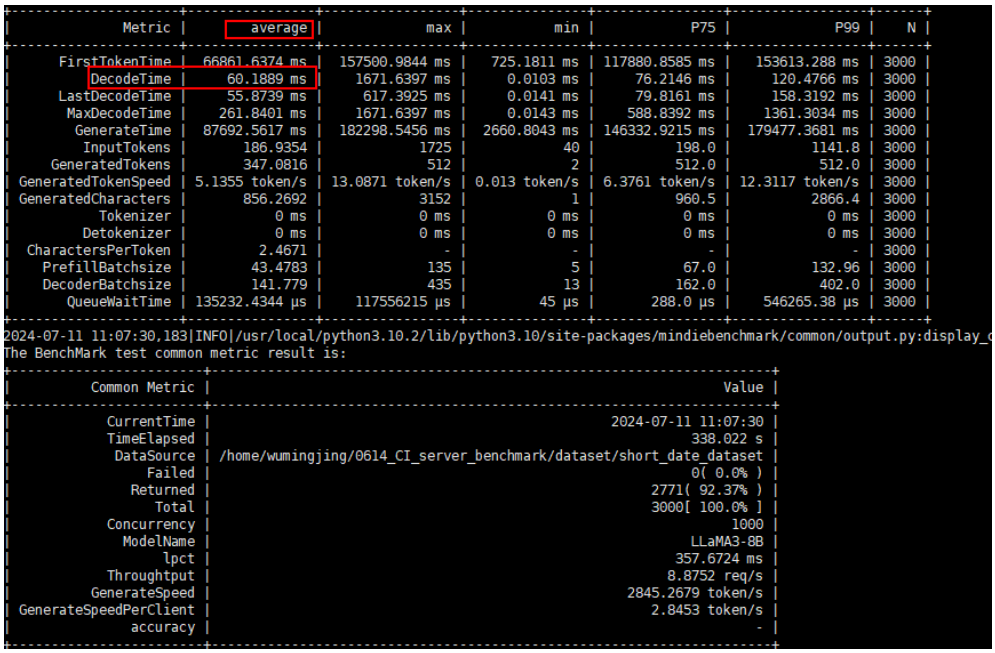
步骤2 根据步骤2.3计算出“maxBatchSize”的取值范围为[362，1088]，设置初始值为**435**；“maxPrefillBatchSize”参数的值设置为“maxBatchSize”值的一半，取值为**217**。

步骤3 配置完成后，用户可使用HTTPS客户端（Linux curl命令，Postman工具等）发送HTTPS请求，此处以Linux curl命令为例进行说明。

重开一个窗口，使用以下命令发送请求，获取当前DecodeTime的平均值（Average），如图5-10所示，此时Decode平均时延为60.1889ms。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \          //数据集路径
--DatasetType "gsm8k" \                                  //数据集类型
--ModelName LLaMa3-8B \                                  //模型名称
--ModelPath "/{模型路径}/LLaMa3-8B" \                    //模型路径
--TestType client \                                      //模式选择
--Http https://{ipAddress}:{port} \                      //请求url
--ManagementHttp https://{managementIpAddress}:{managementPort} \          //管理面url
--Concurrency 1000 \                                    //并发数
--TaskKind stream \                                      //Client不同推理模式，此处为文本流式推理
--Tokenizer True \                                       //分词向量化标识
--MaxOutputLen 512                                       //最大输出长度
```


图 5-10 maxBatchSize 取值为 435 时 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	66861.6374 ms	157500.9844 ms	725.1811 ms	117880.8585 ms	153613.288 ms	3000
DecodeTime	60.1889 ms	1671.6397 ms	0.0103 ms	76.2146 ms	120.4766 ms	3000
LastDecodeTime	55.8739 ms	617.3925 ms	0.0141 ms	79.8161 ms	158.3192 ms	3000
MaxDecodeTime	261.8401 ms	1671.6397 ms	0.0143 ms	588.8392 ms	1361.3034 ms	3000
GenerateTime	87692.5617 ms	182298.5456 ms	2660.8043 ms	146332.9215 ms	179477.3681 ms	3000
InputTokens	186.9354	1725	40	198.0	1141.8	3000
GeneratedTokens	347.0816	512	2	512.0	512.0	3000
GeneratedTokenSpeed	5.1355 token/s	13.0871 token/s	0.013 token/s	6.3761 token/s	12.3117 token/s	3000
GeneratedCharacters	856.2692	3152	1	960.5	2866.4	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4671	-	-	-	-	3000
PrefillBatchsize	43.4783	135	5	67.0	132.96	3000
DecoderBatchsize	141.779	435	13	162.0	402.0	3000
QueueWaitTime	135232.4344 μs	117556215 μs	45 μs	288.0 μs	546265.38 μs	3000

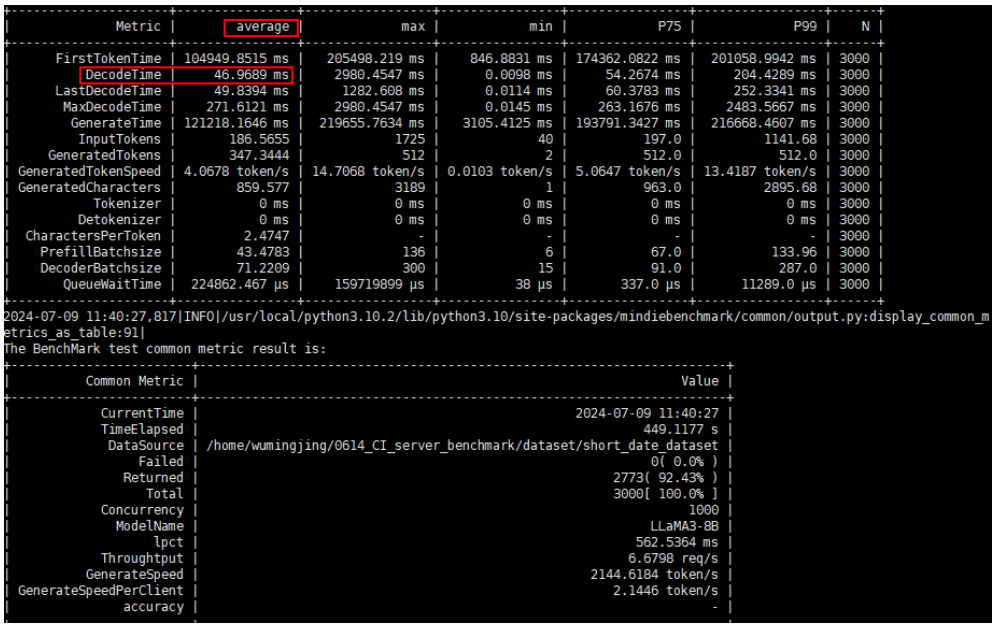
```
2024-07-11 11:07:30,183|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_c
The BenchMark test common metric result is:
```

Common Metric	Value
CurrentTime	2024-07-11 11:07:30
TimeElapsed	338.022 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2771(92.37%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	357.6724 ms
Throughtput	8.8752 req/s
GenerateSpeed	2845.2679 token/s
GenerateSpeedPerClient	2.8453 token/s
accuracy	-

以上结果超过了Decode平均时延为50ms的限制，所以需要调小“maxBatchSize”的值继续调试。

步骤4 设置“maxBatchSize”的值为**300**，“maxPrefillBatchSize”参数的值设置为**150**。然后执行**步骤3**，继续观察Decode平均时延，执行结果如**图5-11**所示，此时decode平均时延为46.9689ms。

图 5-11 maxBatchSize 取值为 300 时 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	104949.8515 ms	205498.219 ms	846.8831 ms	174362.0822 ms	201058.9942 ms	3000
DecodeTime	46.9689 ms	2980.4547 ms	0.0098 ms	54.2674 ms	204.4289 ms	3000
LastDecodeTime	49.8394 ms	1282.608 ms	0.0114 ms	60.3783 ms	252.3341 ms	3000
MaxDecodeTime	271.6121 ms	2980.4547 ms	0.0145 ms	263.1676 ms	2483.5667 ms	3000
GenerateTime	121218.1646 ms	219655.7634 ms	3105.4125 ms	193791.3427 ms	216668.4607 ms	3000
InputTokens	186.5655	1725	40	197.0	1141.68	3000
GeneratedTokens	347.3444	512	2	512.0	512.0	3000
GeneratedTokenSpeed	4.0678 token/s	14.7068 token/s	0.0103 token/s	5.0647 token/s	13.4187 token/s	3000
GeneratedCharacters	859.577	3189	1	963.0	2895.68	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4747	-	-	-	-	3000
PrefillBatchsize	43.4783	136	6	67.0	133.96	3000
DecoderBatchsize	71.2209	300	15	91.0	287.0	3000
QueueWaitTime	224862.467 μs	159719899 μs	38 μs	337.0 μs	11289.0 μs	3000

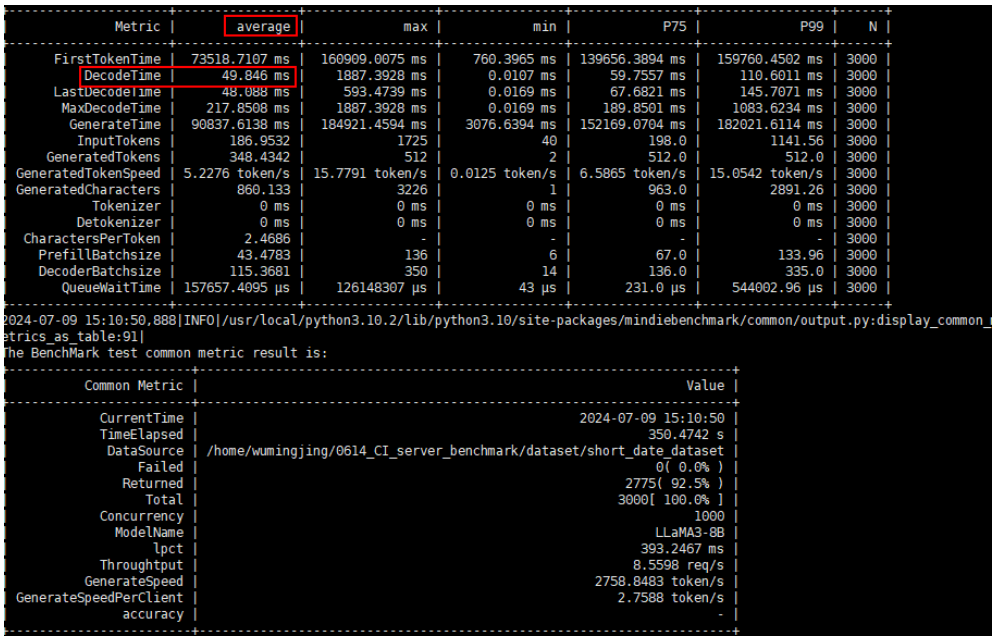
```
2024-07-09 11:40:27,817|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_m
etrics_as_table:91|
The BenchMark test common metric result is:
```

Common Metric	Value
CurrentTime	2024-07-09 11:40:27
TimeElapsed	449.1177 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2773(92.43%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	562.5364 ms
Throughtput	6.6798 req/s
GenerateSpeed	2144.6184 token/s
GenerateSpeedPerClient	2.1446 token/s
accuracy	-

以上结果可以看到Decode平均时延满足50ms以内的限制，但是还未接近50ms，所以需要调大“maxBatchSize”的值继续进行调试。

步骤5 设置“maxBatchSize”的值为**350**，“maxPrefillBatchSize”参数的值设置为**175**。然后执行**步骤3**，继续观察Decode平均时延，执行结果如**图5-12**所示，此时decode平均时延为49.846ms。

图 5-12 maxBatchSize 取值为 350 时 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	73518.7107 ms	160909.0075 ms	760.3965 ms	139656.3894 ms	159760.4502 ms	3000
DecodeTime	49.846 ms	1887.3928 ms	0.0107 ms	59.7557 ms	110.6011 ms	3000
LastDecodeTime	48.088 ms	593.4739 ms	0.0169 ms	67.6821 ms	145.7071 ms	3000
MaxDecodeTime	217.8508 ms	1887.3928 ms	0.0169 ms	189.8501 ms	1083.6234 ms	3000
GenerateTime	90837.6138 ms	184921.4594 ms	3076.6394 ms	152169.0704 ms	182021.6114 ms	3000
InputTokens	186.9532	1725	40	198.0	1141.56	3000
GeneratedTokens	348.4342	512	2	512.0	512.0	3000
GeneratedTokenSpeed	5.2276 token/s	15.7791 token/s	0.0125 token/s	6.5865 token/s	15.0542 token/s	3000
GeneratedCharacters	860.133	3226	1	963.0	2891.26	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4686	-	-	-	-	3000
PrefillBatchsize	43.4783	136	6	67.0	133.96	3000
DecoderBatchsize	115.3681	350	14	136.0	335.0	3000
QueueWaitTime	157657.4095 µs	126148307 µs	43 µs	231.0 µs	544002.96 µs	3000

2024-07-09 15:10:50,888|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-09 15:10:50
TimeElapsed	350.4742 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2775(92.5%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	393.2467 ms
Throughtput	8.5598 req/s
GenerateSpeed	2758.8483 token/s
GenerateSpeedPerClient	2.7588 token/s
accuracy	-

以上结果可以看到Decode平均时延已经很接近50ms，此时几乎已达到限制Decode时延下的最大吞吐量。如需获取Decode平均时延更接近50ms时的“maxBatchSize”值，请根据以上操作步骤继续调试。

----结束

5.2.3 首 token 时延限制严格，非首 token 时延也有限制

以首token平均时延限制在8000ms以内、Decode平均时延限制50ms以内为目标，首token时延限制严格，且非首token时延也有限制的调试方式如下所示。

- 服务端：
  - “maxBatchSize”调小到卡对应的时延，一般情况下“maxBatchSize”越小，则Decode时延越小。
  - 设置supportSelectBatch为false。
- 客户端：
  - 按并发数发送请求：客户端Concurrency通常配置为maxBatchSize-1。
  - 按频率发送请求：则Concurrency可设置为1000，请求发送频率根据实际业务场景或按模型实际QPS设置。

步骤1 使用以下命令启动服务，以当前所在Ascend-mindie-service_*{version}*_linux-*{arch}*目录为例。

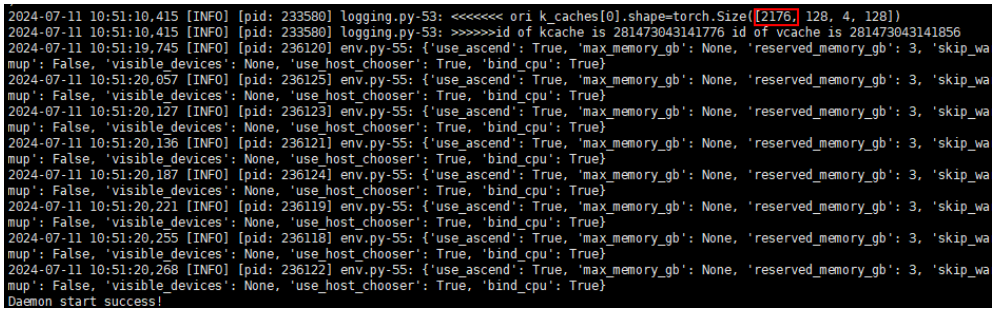
```
./bin/mindieservice_daemon
```

回显如下则说明启动成功。

```
Daemon start success!
```


服务启动后，可通过info级打屏日志k_caches[0].shape=torch.Size([npuBlockNum, x, x, x])中torch.Size的第一个值获取npuBlockNum的值，如图5-13所示，与步骤2.1中计算出来的值一致。

图 5-13 启动成功



步骤2 根据步骤2.3计算出“maxBatchSize”的取值范围为[362，1088]，设置初始值为**435**；“maxPrefillBatchSize”参数的值设置为“maxBatchSize”值的一半，取值为**217**。

说明

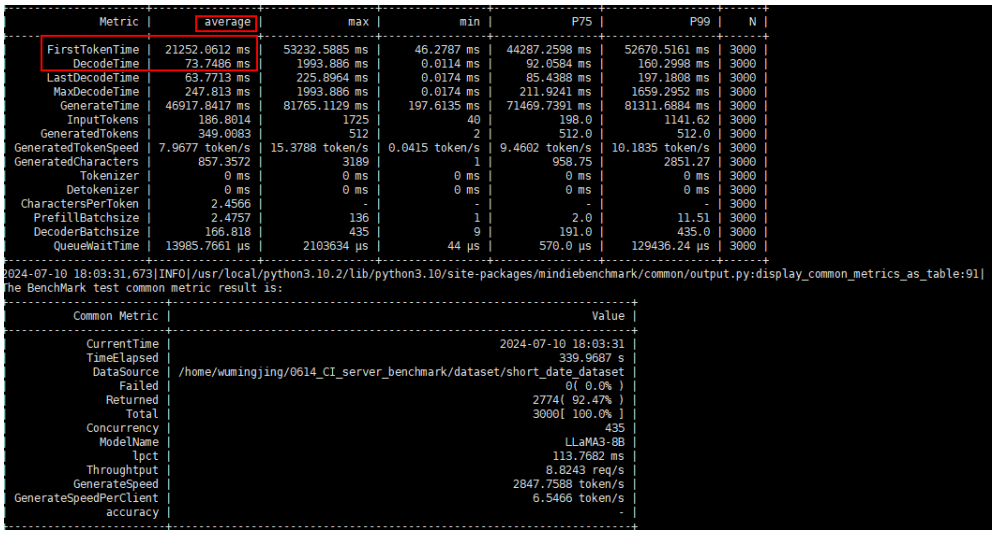
需要将“supportSelectBatch”参数设置为false，以获取更低的首token时延。

步骤3 配置完成后，用户可使用HTTPS客户端（Linux curl命令，Postman工具等）发送HTTPS请求，此处以Linux curl命令为例进行说明。

重开一个窗口，使用以下命令发送请求，获取当前首token平均值（Average）和DecodeTime的平均值（Average），如图5-14所示，此时首token平均时延为21252.0612ms，decode平均时延为73.7486ms。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \          //数据集路径
--DatasetType "gsm8k" \                                  //数据集类型
--ModelName LLaMa3-8B \                                  //模型名称
--ModelPath "/{模型路径}/LLaMa3-8B" \                    //模型路径
--TestType client \                                      //模式选择
--Http https://{ipAddress}:{port} \                      //请求url
--ManagementHttp https://{managementIpAddress}:{managementPort} \          //管理面url
--Concurrency 1000 \                                    //并发数
--TaskKind stream \                                      //Client不同推理模式，此处为文本流式推理
--Tokenizer True \                                       //分词向量化标识
--MaxOutputLen 512                                       //最大输出长度
```

图 5-14 maxBatchSize 取值为 435 时首 token 平均时延（Average）和 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	21252.0612 ms	53232.5885 ms	46.2787 ms	44287.2598 ms	52670.5161 ms	3000
DecodeTime	73.7486 ms	1993.886 ms	0.0114 ms	92.0584 ms	160.2998 ms	3000
LastDecodeTime	63.7713 ms	225.8964 ms	0.0174 ms	85.4388 ms	197.1808 ms	3000
MaxDecodeTime	247.813 ms	1993.886 ms	0.0174 ms	211.9241 ms	1659.2952 ms	3000
GenerateTime	46917.8417 ms	81765.1129 ms	197.6135 ms	71469.7391 ms	81311.6884 ms	3000
InputTokens	186.8014	1725	40	198.0	1141.62	3000
GeneratedTokens	349.0083	512	2	512.0	512.0	3000
GeneratedTokenSpeed	7.9677 token/s	15.3788 token/s	0.0415 token/s	9.4602 token/s	10.1835 token/s	3000
GeneratedCharacters	857.3572	3189	1	958.75	2851.27	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4566	-	-	-	-	3000
PrefillBatchsize	2.4757	136	1	2.0	11.51	3000
DecoderBatchsize	166.818	435	9	191.0	435.0	3000
QueueWaitTime	13985.7661 μs	2103634 μs	44 μs	570.0 μs	129436.24 μs	3000

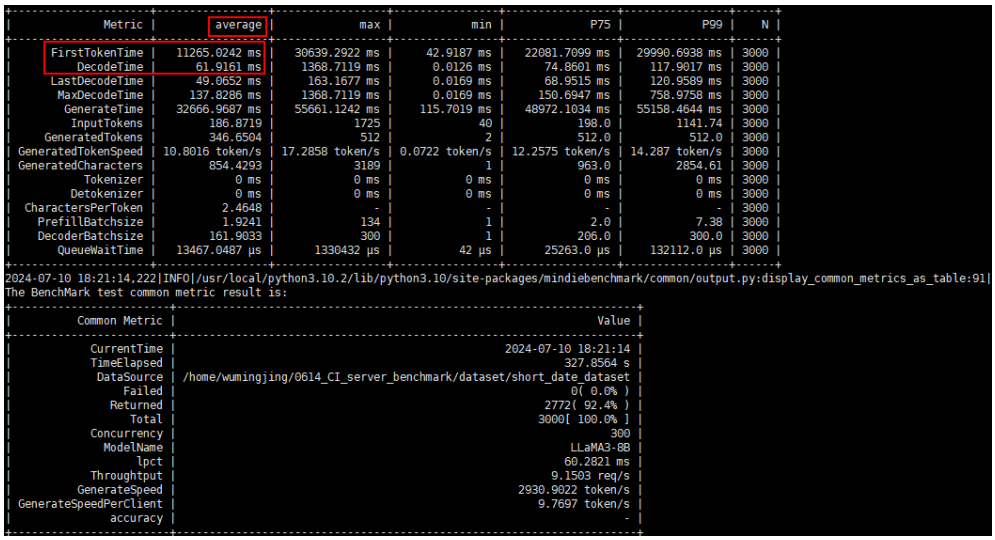
2024-07-10 18:03:31,673|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-10 18:03:31
TimeElapsed	339.9687 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2774(92.47%)
Total	3000[100.0%]
Concurrency	435
ModelName	LLaMA3-8B
lpct	113.7682 ms
Throughtput	8.8243 req/s
GenerateSpeed	2847.7588 token/s
GenerateSpeedPerClient	6.5466 token/s
accuracy	-

以上结果超过了首token平均时延为8000ms和Decode平均时延为50ms的限制，所以需要调小“maxBatchSize”的值继续调试。

步骤4 设置“maxBatchSize”的值为**300**，“maxPrefillBatchSize”参数的值设置为**150**。然后执行**步骤3**，继续观察首token平均时延和Decode平均时延，执行结果如**图5-15**所示，此时首token平均时延为11265.0242ms，Decode平均时延为61.9161ms。

图 5-15 maxBatchSize 取值为 300 时首 token 平均时延（Average）和 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	11265.0242 ms	30639.2922 ms	42.9187 ms	22081.7099 ms	29990.6938 ms	3000
DecodeTime	61.9161 ms	1368.7119 ms	0.0126 ms	74.8601 ms	117.9017 ms	3000
LastDecodeTime	49.0652 ms	163.1677 ms	0.0169 ms	68.9515 ms	120.9589 ms	3000
MaxDecodeTime	137.8286 ms	1368.7119 ms	0.0169 ms	150.6947 ms	758.9758 ms	3000
GenerateTime	32666.9687 ms	55661.1242 ms	115.7019 ms	48972.1034 ms	55158.4644 ms	3000
InputTokens	186.8719	1725	40	198.0	1141.74	3000
GeneratedTokens	346.6504	512	2	512.0	512.0	3000
GeneratedTokenSpeed	10.8016 token/s	17.2858 token/s	0.0722 token/s	12.2575 token/s	14.287 token/s	3000
GeneratedCharacters	854.4293	3189	1	963.0	2854.61	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4648	-	-	-	-	3000
PrefillBatchsize	1.9241	134	1	2.0	7.38	3000
DecoderBatchsize	161.9033	300	1	206.0	300.0	3000
QueueWaitTime	13467.0487 μs	1330432 μs	42 μs	25263.0 μs	132112.0 μs	3000

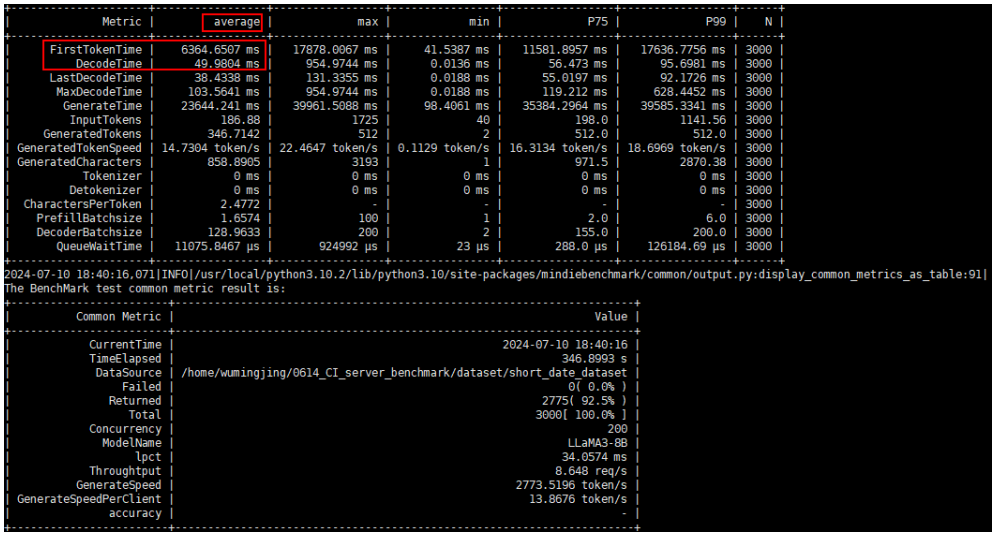
2024-07-10 18:21:14,222|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-10 18:21:14
TimeElapsed	327.8564 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2772(92.4%)
Total	3000[100.0%]
Concurrency	300
ModelName	LLaMA3-8B
lpct	60.2821 ms
Throughtput	9.1503 req/s
GenerateSpeed	2930.9022 token/s
GenerateSpeedPerClient	9.7697 token/s
accuracy	-

以上结果同样超过了首token平均时延为8000ms和Decode平均时延为50ms的限制，所以需要调小“maxBatchSize”的值继续调试。

步骤5 设置“maxBatchSize”的值为**200**，“maxPrefillBatchSize”参数的值设置为**100**。然后执行**步骤3**，继续观察首token平均时延和Decode平均时延，执行结果如**图5-16**所示，此时首token平均时延为6364.6507ms，Decode平均时延为49.9804ms。

图 5-16 maxBatchSize 取值为 200 时首 token 平均时延（Average）和 Decode 的平均时延（Average）



Metric	average	max	min	P75	P99	N
FirstTokenTime	6364.6507 ms	17878.0067 ms	41.5387 ms	11581.8957 ms	17636.7756 ms	3000
DecodeTime	49.9804 ms	954.9744 ms	0.0136 ms	56.473 ms	95.6981 ms	3000
LastDecodeTime	38.4338 ms	131.3355 ms	0.0188 ms	55.0197 ms	92.1726 ms	3000
MaxDecodeTime	103.5641 ms	954.9744 ms	0.0188 ms	119.212 ms	628.4452 ms	3000
GenerateTime	23644.241 ms	39961.5088 ms	98.4061 ms	35384.2964 ms	39585.3341 ms	3000
InputTokens	186.88	1725	40	198.0	1141.56	3000
GeneratedTokens	346.7142	512	2	512.0	512.0	3000
GeneratedTokenSpeed	14.7304 token/s	22.4647 token/s	0.1129 token/s	16.3134 token/s	18.6969 token/s	3000
GeneratedCharacters	858.8905	3193	1	971.5	2870.38	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4772	-	-	-	-	3000
PrefillBatchsize	1.6574	100	1	2.0	6.0	3000
DecoderBatchsize	128.9633	200	2	155.0	200.0	3000
QueueWaitTime	11075.8467 µs	924992 µs	23 µs	288.0 µs	126184.69 µs	3000

2024-07-10 18:40:16,071|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-07-10 18:40:16
TimeElapsed	346.8993 s
DataSource	/home/wumingjing/0614_CI_server_benchmark/dataset/short_date_dataset
Failed	0(0.0%)
Returned	2775(92.5%)
Total	3000[100.0%]
Concurrency	200
ModelName	LLaMA3-8B
lpct	34.0574 ms
Throughtput	8.648 req/s
GenerateSpeed	2773.5196 token/s
GenerateSpeedPerClient	13.8676 token/s
accuracy	-

以上结果可以看到首token平均时延和Decode平均时延已经在限制的8000ms和50ms以内，且Decode平均时延已经很接近50ms，此时几乎已达到限制首token时延和Decode时延下的最大吞吐量。如需获取Decode平均时延更接近50ms时的“maxBatchSize”值，请根据以上操作步骤继续调试。

----结束

6 配套工具

6.1 性能/精度测试工具

MindIE Service提供性能和精度测试工具MindIE Benchmark，具体使用方法详情请参见**8.1.1 MindIE Benchmark**。

6.2 证书管理工具

MindIE Service提供证书管理工具CertTools，具体使用方法详情请参见**8.1.2 CertTools**。

6.3 MindIE 探针工具

MindIE Service提供探针脚本，使能kubernetes探针检测功能，支持启动，存活，就绪三种探针。

脚本适用于以下部署场景：

- 集成MindIE MS Controller、MindIE MS Coordinator和MindIE Server的部署场景。
- 仅集成MindIE Server PD混合部署场景。

在MindIE Service安装路径下，可找到探针入口脚本：$MIES_INSTALL_PATH/scripts/http_client_ctl/probe.sh。

探针的使用命令如**表6-1**所示。

表 6-1 探针的使用

指令	类型	说明
bash probe.sh startup	启动探针	用于判断程序是否已启动，探针触发后设置了60s的超时时间。
bash probe.sh liveness	存活探针	用于发现程序进程状态是否健康，探针触发后内部设置了60s的请求超时时间。
bash probe.sh readiness	就绪探针	用于判断程序是否就绪接收流量，探针触发后设置了60s的超时时间。

使用到的环境变量如表6-2所示。

表 6-2 环境变量

环境变量	说明
POD_IP	容器Pod IP。
MIES_INSTALL_PATH	MindIE-Service安装路径。
MINDIE_SERVER_PROBE_ONLY	仅针对MindIE Server进行状态探测的标志，设置为1时生效，适用于仅集成部署MindIE Server的PD混合场景。
GLOBAL_RANK_TABLE_FILE_PATH	全局ranktable文件路径，适用于集成MindIE MS Controller、MindIE MS Coordinator和MindIE Server多组件部署的场景。
MINDIE_UTILS_HTTP_CLIENT_CTL_CONFIG_FILE_PATH	探针配置文件的读取路径。
MINDIE_USE_HTTPS	是否启用HTTPS安全通信，取值为true或false, 设置后优先使用该配置替代http_client_ctl.json。建议用户打开，确保通信安全。如果关闭则存在较高的网络安全风险。

probe.sh脚本依赖 http_client_ctl组件发送HTTP请求，请求指令如表6-3所示。

表 6-3 http_client_ctl 命令介绍

指令	说明
./http_client_ctl [ip] [port] [url] [timeout] [retrytime]	• [ip]：目标IPv4格式的IP地址。 • [port]：目标端口。取值范围[1024, 65535]。 • [url]：HTTP请求的URL。 • [timeout]：请求超时时间，单位秒。取值范围[1, 600]。 • [retrytime]：重试次数。取值范围[0, 30]。
./http_client_ctl -h/--help	命令使用帮助。

另外，http_client_ctl还需配置http_client_ctl.json文件，其字段解释如表6-4所示。

```
{
    "tls_enable" : true,
    "cert": {
        "ca_cert" : "./security/http_client/ca/ca.pem",
        "tls_cert": "./security/http_client/certs/cert.pem",
        "tls_key": "./security/http_client/keys/cert.key.pem",
        "tls_passwd": "./security/http_client/pass/key_pwd.txt",
        "kmc_ksf_master": "./tools/pmt/master/ksfa",
        "kmc_ksf_standby": "./tools/pmt/standby/ksfb",
        "tls_crl": ""
    },
    "log_info": {
        "log_level": "INFO",
        "to_file": false,
        "run_log_path": "/var/log/mindie-ms/run/log.txt",
        "operation_log_path": "/var/log/mindie-ms/operation/log.txt"
    }
}
```

表 6-4 http_client_ctl.json 配置说明

配置类型	配置项	配置介绍
证书配置	tls_enable	必填。是否开启HTTPS通信，默认为true。 • true：表示开启； • false：表示关闭。 建议用户开启，确保通信安全。如果关闭则存在较高的网络安全风险。 如设置环境变量MINDIE_USE_HTTPS，则优先读取环境变量的值。
	ca_cert	必填。 客户端ca根证书文件路径，该路径真实存在且可读。

配置类型	配置项	配置介绍
	tls_cert	必填。 客户端tls证书文件路径，该路径真实存在且可读。
	tls_key	必填。 客户端tls私钥文件路径，该路径真实存在且可读。
	tls_passwd	必填。 KMC加密的私钥口令的文件路径。
	kmcKsfMaster	必填。 加密口令的KMC密钥库文件。
	kmcKsfStandby	必填。 加密口令的KMC standby密钥库备份文件。
	tls_crl	必填。 证书吊销列表crl文件路径，该路径存在且可读。如为空，则不进行吊销校验。
日志配置	log_level	日志级别。默认值为INFO。 ● DEBUG ● INFO ● WARNING ● ERROR ● CRITICAL 如设置环境变量MINDIEMS_LOG_LEVEL或者MINDIE_LOG_LEVEL，则优先读取环境变量的值。
	to_file	是否写入到文件。默认值为false。 ● true：输出到文件。 ● false：不输出到文件。 如设置环境变量MINDIE_LOG_TO_FILE，则优先读取环境变量的值。
	run_log_path	运行日志文件路径。
	operation_log_path	操作日志文件路径。

表 6-5 环境变量

环境变量名称	含义
MINDIE_UTILS_HTTP_CLIENT_CTL_CONFIG_FILE_PATH	http_client_ctl.json的文件路径

若用户开启了HTTPS安全通信，需要设置上述tls_enable为true，并准备相关证书（CA证书，客户端证书，私钥文件），使用证书管理工具导入后生成KMC加密口令文件，详情请参见**8.1.2.1 config_mindie_server_tls_cert.py**。

将证书（包括security和tools目录）导入容器，两种方式如下所示：

- 方式一：在制作镜像的过程中拷贝相关的证书和KMC文件到容器内。
- 方式二：在启动容器时将相关文件通过宿主机挂载方式导入。

最后需将MindIE Service安装路径下的conf/http_client_ctl.json文件"cert"字段中的相关文件路径配置为导入容器后的绝对路径。

7 关键特性

7.1 特性概述

表 7-1 特性介绍

特性	说明
Multi-Lora	使用Multi-Lora来执行基础模型和不同的LoRA权重进行推理，其特性介绍详情请参见**7.3 Multi-Lora**。
多模态理解	多模态理解模型是指能够处理和理解包括多种模态数据的深度学习模型。其特性介绍详情请参见**7.4 多模态理解**。
Function Call	支持Function Call函数调用，使大模型具备使用工具能力。其特性介绍详情请参见**7.5 Function Call**。
Splitfuse	将长提示词分解成更小的块，并在多个forward step中进行调度，降低Prefill时延。其特性介绍详情请参见**7.6 Splitfuse**。

特性	说明
Prefix Cache	复用跨session的重复token序列对应的KV Cache，减少一部分前缀token的KV Cache计算时间，从而减少Prefill的时间。其特性介绍详情请参见**7.7 Prefix Cache**。
分布式多机部署	对于超大模型，单机推理无法容纳整个模型权重参数，因此需要多台推理机协同工作，共同完成整个模型的推理，其特性介绍详情请参见**7.8 分布式多机部署**。
PD分离部署	模型推理的Prefill阶段和Decode阶段分别实例化部署在不同的机器资源上同时进行推理，提升推理性能，其特性介绍详情请参见**7.9 PD分离部署**。
模型量化	其特性介绍详情请参见**7.10 模型量化**。

7.2 支持的模型

MindIE Service 支持的模型详情请参见《 MindIE是什么 》的“MindIE支持模型列表 > **大语言模型列表**”章节。

7.3 Multi-Lora

7.3.1 特性介绍

LoRA（ Low-Rank Adaptation ）是一种高效的参数微调方法。将大模型的权重矩阵分解为原始权重矩阵和两个低秩矩阵的乘积，即W' = W + BA。由于B和A矩阵参与训练的参数量远小于原始权重，其乘积结果又能合入线性层并向下传递，从而达到大模型轻量级微调的目的。

Multi-LoRA指基于一个基础模型，使用多个不同的LoRA权重进行推理。每个请求带有指定的LoRA ID，推理时动态匹配对应的LoRA权重。部署服务时，LoRA权重和基础模型权重预先加载至显存中。一个推理请求至多使用一个LoRA权重，兼容推理请求不使用Lora权重的情况。对于大参数量的模型，若模型参数量过大，无法单卡加载时，可进行Tensor Parallel并行。

LoRA权重中需包含"adapter_config.json"和"adapter_model.safetensors"文件，文件描述如**表7-2**所示。

表 7-2 文件说明

文件名称	文件描述	示例
adapter_config.json	包含LoRA权重的超参。	lora_rank（ LoRA 微调中的秩大小），lora_alpha（ LoRA低秩矩阵的缩放系数）等。

文件名称	文件描述	示例
adapter_model.safetensors	包含权重，权重以键值对的形式保存，其中LoRA权重的键名在基础模型的键名前后增加"base_model.model"前缀和"lora_A.weight"、"lora_B.weight"后缀。	当基础模型中的键名为"model.layers.9.self_attn.v_proj.weight"时，则LoRA权重中对应的键名应为"base_model.model.model.layers.9.self_attn.v_proj.lora_A.weight"和"base_model.model.model.layers.9.self_attn.v_proj.lora_B.weight"。

7.3.2 使用样例

限制与约束

- 仅Atlas 800I A2 推理产品支持此特性。
- LoRA权重个数上限受硬件显存限制，建议数量为小于等于10个。
- 不支持LoRA权重热加载。
- 仅Qwen1.5-14B、Qwen1.5-72B、LLaMa3.1-70B和Qwen2-72B支持Multi-LoRA特性。

操作步骤

本章节以LLaMa3.1 70B模型为例，简单介绍Multi-Lora如何使用。

步骤1 下载基础模型和Lora权重后，在基础模型路径下新增lora_adapter.json文件配置预加载的Lora权重。

以使用两个lora权重为例：

```
cd ${基础模型权重}
vi lora_adapter.json
# 在此文件中配置adapter名称及路径，示例：
{"adapter1": "${Lora权重1路径}", "adapter2": "${Lora权重2路径}"}
```

步骤2 配置服务化参数并启动服务化，服务化参数说明请参见**8.4.2 配置参数说明**章节。

```
cd ${mindie-service安装路径}
vi conf/config.json
./bin/mindieservice_daemon
```

步骤3 使用以下指令发送请求。

其中"model"参数可以设置为模型权重路径或lora_adapter.json中配置的adapter名称。当"model"参数为模型权重路径时，不使用Lora权重进行推理。当"model"参数为lora_adapter.json中配置的adapter名称时，启用基础模型权重和指定的Lora权重进行推理。

```
curl https://127.0.0.1:1025/generate \
-H "Content-Type: application/json" \
--cacert ca.pem --cert client.pem  --key client.key.pem \
-X POST \
-d '{
"model": "${基础模型权重}",
"prompt": "Taxation in Puerto Rico -- The Commonwealth government has its own tax laws and Puerto
```



```
Ricans are also required to pay some US federal taxes, although most residents do not have to pay the federal personal income tax. In 2009, Puerto Rico paid $3.742 billion into the US Treasury. Residents of Puerto Rico pay into Social Security, and are thus eligible for Social Security benefits upon retirement. However, they are excluded from the Supplemental Security Income.\nQuestion: is federal income tax the same as social security?\nAnswer:",
"max_tokens": 20,
"temperature": 0
}'

curl https://127.0.0.1:1025/generate \
-H "Content-Type: application/json" \
--cacert ca.pem --cert client.pem  --key client.key.pem \
-X POST \
-d '{
"model": "adapter1",
"prompt": "Taxation in Puerto Rico -- The Commonwealth government has its own tax laws and Puerto Ricans are also required to pay some US federal taxes, although most residents do not have to pay the federal personal income tax. In 2009, Puerto Rico paid $3.742 billion into the US Treasury. Residents of Puerto Rico pay into Social Security, and are thus eligible for Social Security benefits upon retirement. However, they are excluded from the Supplemental Security Income.\nQuestion: is federal income tax the same as social security?\nAnswer:",
"max_tokens": 20,
"temperature": 0
}'
```

----结束

7.4 多模态理解

7.4.1 特性介绍

模态理解模型是基于大语言模型的深度学习类模型，能够处理并理解多种不同的数据类型。当前多模态理解模型主要是针对文本、图片、视频、音频等数据类型进行处理，并提取整合其特征，最终由大语言基座模型进行理解并产生对应的内容。

- 主要特点和优势：拥有更完善的跨模态信息抽取能力和更准确的多模态数据理解能力。
- 应用场景：图像问答、情感分析、自然语言对话、视频分析、自动驾驶等领域。

其数据量多且大的特点，使得数据表征对齐和更高的计算资源要求等成为新的挑战。总的来说，多模态模型将文本、图像、音频或视频等至少两种模态的数据作为输入，从输入的多模态数据中提取特征并进行融合，从而使得多模态模型能够实现更全面、更准确的理解和推理能力。

7.4.2 使用样例

限制与约束

- Atlas 800I A2 推理产品和Atlas 300I Duo 推理卡支持此特性。
- 模型特性矩阵及相关文档请参考《 MindIE是什么 》的“MindIE支持模型列表 > **多模态理解模型列表**”章节。

操作步骤

本章节以qwen-vl模型为例，简单介绍多模态如何使用。

说明

更多多模态的使用详情请参见4.2 EndPoint业务面RESTful接口章节。

步骤1 安装多模态模型依赖。

```
cd /usr/local/Ascend/llm_model/requirements/models  # mindIE安装包默认路径
pip install -r requirements_qwen_vl.txt
```

步骤2 下载基础模型权重后，配置服务化参数config.json文件，然后启动服务化，服务化参数说明请参见8.4.2 配置参数说明章节。

```
cd ${mindie-service安装路径}
vi conf/config.json  # 配置模型参数"modelName"、"modelWeightPath"等
./bin/mindieservice_daemon
```

步骤3 使用以下指令发送请求，参数说明见4.2.1.3 文本/流式推理接口章节。

```
curl https://127.0.0.11:1025/generate -H "Content-Type: application/json" --cacert ca.pem --cert client.pem
--key client.key.pem -X POST -d '{
    "prompt": [
        {"type": "text", "text": "What is in this image?"},
    { "type": "image_url", "image_url": "/xxx/test.png"}
    ],
    "max_tokens": 100,
    "repetition_penalty": 1.03,
    "presence_penalty": 1.2,
    "frequency_penalty": 1.2,
    "temperature": 0.5,
    "top_k": 10,
    "top_p": 0.95
}'
```

----结束

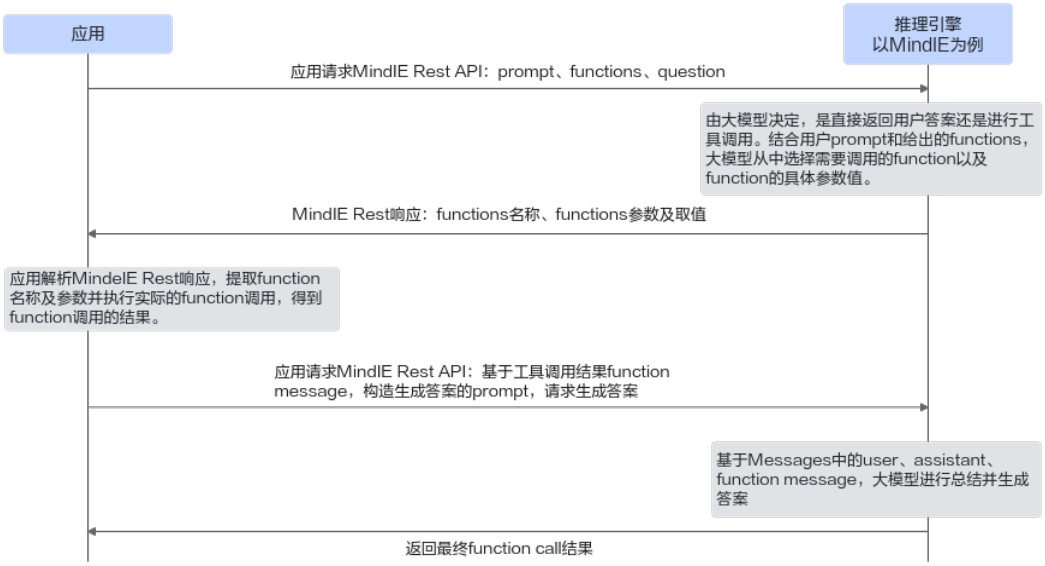
7.5 Function Call

7.5.1 特性介绍

模型的函数调用（Function Call）能力，又称工具调用能力（tool use），是指大模型能够调用外部工具以扩展其应用范围。函数调用功能允许模型直接调用外部函数或API，从而获得执行特定任务、获取实时数据或增强决策的能力。这一特性不仅扩展了模型的应用范围，使其能处理更复杂、更具体的问题，提升了模型的实用性和交互性，实现了大模型与外部世界的高效连接，为用户提供更丰富、更个性化的服务。

以下统一使用“工具调用“（tool use）来介绍Function Call特性。

图 7-1 大模型工具调用的流程图



流程步骤

1. 上层应用将系统prompt和用户输入内容给到大模型，同时由上层应用负责给出大模型本次执行可用的工具集合。
2. 大模型根据系统promp和用户输入内容，决定是直接返回答案还是从应用给出的工具集合中选取一个或多个函数。如果选择使用工具，则将本次选择的工具名称和工具参数取值信息返回给上层应用。
3. 上层应用解析来自推理引擎的响应，提取模型选择的工具信息，执行模型选择出的函数，得到本次工具调用的结果。
4. 上层应用使用工具调用的结果，构造生成答案的prompt，再次发送给大模型，请求生成最终答案。
5. 大模型根据工具调用的结果，总结信息，生成答案并返回。

7.5.2 使用样例

本章节以使用ChatGLM3-6B为例，介绍Function Call如何使用，使用Function Call时无需配置额外参数。

步骤1 配置服务化参数config.json，启动服务化。

```
cd ${mindie-service安装路径}
vi conf/config.json
./bin/mindieservice_daemon
```

步骤2 向服务发送请求，参数说明见**4.2.3.3 推理接口**章节。

请求样例：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --key client.key.pem -X POST -d '{

  "model": "chatglm3-6b",
  "messages": [
    {
        "role": "system",
        "content": "You are a helpful customer support assistant. Use the supplied tools to assist the user."
    },
    {
        "role": "user",
```



```
            "content": "Hi, can you tell me the delivery date for my order?  my order number is 999888"
        }
    ],
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "description": "Get the delivery date for a customer\u0027s order. Call this whenever you need to know the delivery date, for example when a customer asks \u0027Where is my package\u0027",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer\u0027s order ID."
                        }
                    },
                    "required": [
                        "order_id"
                    ],
                    "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "required",
    "stream": false
}'  https://127.0.0.1:1025/v1/chat/completions
```

响应样例：

```
{
    "id": "chatcmpl-123",
    "object": "chat.completion",
    "created": 1677652288,
    "model": "chatglm3-6b",
    "choices": [
        {
            "index": 0,
            "message": {
                "role": "assistant",
                "content": "",
                "tool_calls": [
                    {
                        "function": {
                            "arguments": "{\"order_id\": \"999888\"}",
                            "name": "get_delivery_date"
                        },
                        "id": "call_JwmTNF3O",
                        "type": "function"
                    }
                ]
            },
            "finish_reason": "tool_calls"
        }
    ],
    "usage": {
        "prompt_tokens": 226,
        "completion_tokens": 122,
        "total_tokens": 348
    },
    "prefill_time": 200,
    "decode_time_arr": [56, 28, 28]
}
```

步骤3 根据模型返回的tool_calls调用相关的本地工具，使用assitant角色关联**步骤2**中接口返回的tool_calls和id，并使用tool角色关联工具执行的结果和**步骤2**中接口返回的id，向大模型发送请求。


```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert ca.pem --cert client.pem  --
key client.key.pem -X POST -d '{
   "model": "chatglm3-6b",
   "messages": [

      {
         "role": "user",
         "content": "Hi, can you tell me the delivery date for my order?  my order number is 999888"
      },
      {
         "role": "assistant",
         "tool_calls": [
         {
            "function": {
               "arguments": "{\"order_id\": \"999888\"}",
               "name": "get_delivery_date"
            },
            "id":"call_JwmTNF3O",
            "type": "function"
          }
         ]
      },
      {
         "role": "tool",
         "content": "the delivery date is 2024.09.10.",
         "tool_call_id": "call_JwmTNF3O"
      }
   ],
   "stream": false,
   "max_tokens": 4096

}'  https://127.0.0.1:1025/v1/chat/completions
```

----结束

7.6 Splitfuse

7.6.1 特性介绍

SplitFuse特性的目的是将长提示词分解成更小的块，并在多个forward step（即通过多轮更短的Prefill代替原本的单次Prefill）中进行调度，只有最后一块的forward完成后才开始这个提示词的生成。将短提示词组合以精确填充step的空隙，每个step的计算量基本相等，达到所有请求平均延迟更稳定的目的。

其优势主要包括以下几点：

- **提高响应速度**：减少长提示词处理延迟，提升用户体验。
- **提升效率**：通过合理组合短提示词，保持模型高吞吐量运行。
- **增强一致性**：统一前向传递大小，降低延迟波动，使生成频率更稳定。

7.6.2 使用样例

限制与约束

- 仅Atlas 800I A2 推理产品支持此特性。
- LLaMa3.1-70b浮点模型 和Qwen2-72B 浮点模型支持对接此特性。
- 暂不支持与LoRA特性配合。
- 该特性不能和PD分离特性同时使用。

操作步骤

本章节简单介绍如何使用Splitfuse功能。

步骤1 配置MindIE Server的config.json配置文件，参数解释请参见配置说明。

```
cd ${mindie-service安装路径}
vi conf/config.json
```

SplitFuse特性必须额外配置的参数如下：

- 在ModelDeployConfig中的“ModelConfig”下添加：
```
"plugin_params": "{\"plugin_type\":\"splitfuse\"}"
```
- 在ScheduleConfig中修改“templateType”参数为Mix：
```
"templateType": "Mix",
```
- 在ScheduleConfig中设置“policyType”参数，详细策略请参见配置说明。
```
"policyType": 0,
```
- 在ScheduleConfig中添加SplitFuse相关的关键参数：
```
"enableSplit": true,
"splitType": false,
"splitStartType": false,
"splitChunkTokens": 512,
"splitStartBatchSize": 16
```

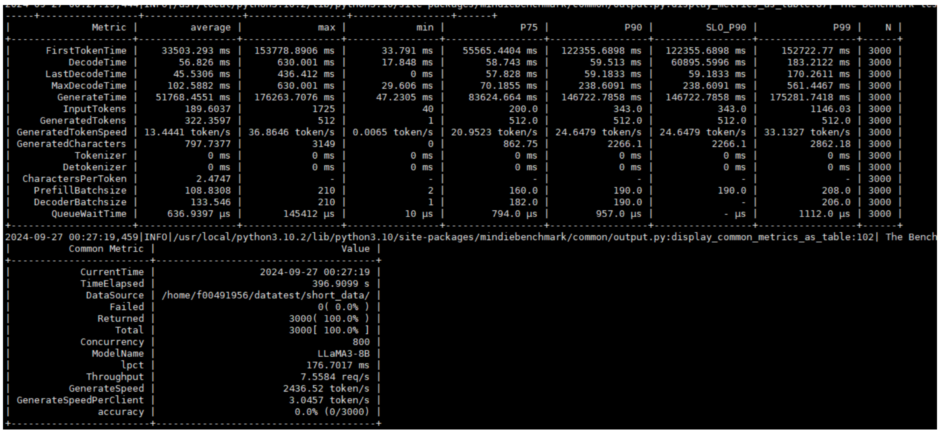
保存修改后的配置并启动服务化：

```
./bin/mindieservice_daemon
```

步骤2 使用以下样例启动Benchmark。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型路径}/baichuan2-13b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--MaxOutputLen 512
```

Benchmark输出结果如下所示：



Metric	average	max	min	P75	P90	SLO_P90	P99	N
FirstTokenTime	33503.293 ms	153778.8906 ms	33.791 ms	55565.4404 ms	122355.6898 ms	122355.6898 ms	152722.77 ms	3000
DecodeTime	56.826 ms	630.001 ms	17.848 ms	58.743 ms	59.513 ms	60895.5996 ms	183.2122 ms	3000
LastDecodeTime	45.5306 ms	436.412 ms	0 ms	57.828 ms	59.1833 ms	59.1833 ms	170.2611 ms	3000
MaxDecodeTime	102.5882 ms	630.001 ms	29.606 ms	70.1855 ms	238.6091 ms	238.6091 ms	561.4467 ms	3000
GenerateTime	51768.4551 ms	176263.7076 ms	47.2305 ms	83624.664 ms	146722.7858 ms	146722.7858 ms	175281.7418 ms	3000
InputTokens	189.6037	1725	40	200.0	343.0	343.0	1146.03	3000
GeneratedTokens	322.3597	512	1	512.0	512.0	512.0	512.0	3000
GeneratedTokenSpeed	13.4441 token/s	36.8646 token/s	0.0065 token/s	20.9523 token/s	24.6479 token/s	24.6479 token/s	33.1327 token/s	3000
GeneratedCharacters	797.7377	3149	0	862.75	2266.1	2266.1	2862.18	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.4747	-	-	-	-	-	-	3000
PrefillBatchsize	108.8308	210	2	160.0	190.0	190.0	208.0	3000
DecoderBatchsize	133.546	210	1	182.0	190.0	-	206.0	3000
QueueWaitTime	636.9397 μs	145412 μs	10 μs	794.0 μs	957.0 μs	- μs	1112.0 μs	3000

2024-09-27 00:27:19,459|INFO|/usr/local/python3.10.2/lib/python3.10/site-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:102| The Bench

Common Metric	Value
CurrentTime	2024-09-27 00:27:19
TimeElapsed	396.9099 s
DataSource	/home/f00491956/datatest/short_data/
Failed	0(0.0%)
Returned	3000(100.0%)
Total	3000[100.0%]
Concurrency	800
ModelName	LLaMA3-8B
lpct	176.7017 ms
Throughput	7.5584 req/s
GenerateSpeed	2436.52 token/s
GenerateSpeedPerClient	3.0457 token/s
accuracy	0.0% (0/3000)

----结束

7.6.3 SplitFuse 性能调优

概述

MindIE支持SplitFuse特性，当MindIE在默认情况下使用PD竞争策略，Prefill和Decode阶段请求不会同时被组合成一个batch。打开SplitFuse特性后，MindIE会在优先处理Decode请求的基础上，且batch小于maxBatchSize的情况下在同一批次中加入Prefill请求。

当该次处理的feedforward大于splitchunk tokens时，SplitFuse会对其进行切分，解释如下所示：

- 每一推理轮次中：$token_{forward} = token_{prefill} + token_{deocde}$，其中：

  $token_{prefill} = \Sigma输入token$

- Prefill阶段的tokens为输入token数量，Decode阶段每个请求的为1token：

  $token_{decode} = batchsize_{decode} * 1\ token$

该特性可以通过*{MindIE安装目录}*/latest/mindie-service/conf/config.json配置文件中通过以下参数打开。

- plugin_params：需配置为"{\"plugin_type\":\"splitfuse\"}"；
- templateType：需配置为"Mix"；
- enableSplit：需配置为"true"。

优势

- 高响应性：由于Prefill阶段和Decode阶段的请求可以同时进行批处理，减少了空泡。
- 高效率：该特性优先处理Decode请求，提供了更大的系统吞吐和更好的Decode时延。
- 低的波动和高一致性：由于feedforward（前向传播）的大小一致，因此相比PD混部，可以提供更稳定的生成频率。

约束

- 仅Atlas 800I A2 推理产品支持SplitFuse性能调优。
- 支持对接SplitFuse性能调优的模型为llama3.1-70b。

配置说明

开启Splitfuse特性需要在config.json配置文件中补充的参数如表7-3所示。

表 7-3 Splitfuse 补充参数

配置项	取值类型	说明
ModelDeployConfig		

配置项	取值类型	说明
plugin_params	std::string	必选； ● "{\"plugin_type\":\"splitfuse\"}"：表示执行splitfuse。 ● 不添加该字段或将该字段设置为""：表示默认不生效任何插件功能。
ScheduleConfig		
templateType	std::string	必选； ● Mix：混部推理场景；Prefill和Decode可同时进行批处理。 ● Standard：基线场景；表示Prefill和Decode各自分别组batch。
policyType	Enum	可选； ● 0：先入先出策略（FCFS）。 ● 4：短任务优先（SJF）。 ● 5：长任务优先（LJF）。 ● 6：多级反馈队列（Skip-Join MLFQ）。 ● 7：短任务优先多级反馈队列（SJF-MLFQ）。
enableSplit	bool	必选； ● true：开启对prompt进行动态切分。 ● false：关闭对prompt进行动态切分。
splitType	bool	可选； ● true：优化切分长度，仅对超出切分长度2倍的prompt进行切分。 ● false：按照配置的切分长度切分。 建议值：false；默认值：false
splitStartType	bool	可选； ● true：每次组batch时重置切分状态，重新判断当前是否满足splitStartBatchSize ● false：首次满足splitStartBatchSize条件后，不再重置切分状态 建议值：false；默认值：false
splitChunkTokens	uint32_t	可选； 当前参与组batch的总token个数。 取值范围：[1,8192]；建议值：16的整数倍；默认值：512

配置项	取值类型	说明
splitStartBatchSize	uint32_t	可选； 当batch数达到该值时开启切分。 取值范围：[0,maxBatchSize]；建议值：16；默认值：16

原理

大模型推理过程中，推理性能主要受访存速度限制，在此种条件下可见模型推理耗时主要与**Forward Tokens**的数量呈线性关系，如图7-2所示。因此，打开**SplitFuse**时，当需要增量推理时延满足**特定控制服务级别目标**（SLO），如**90%轮次的增量推理时延**（SLO_P90 decode时延）小于50ms时，可以通过控制**SplitFuse**特性的**ChunkSize**进行控制。需要说明的是减少ChunkSize在约束SLO指标的同时，可能会增加推理轮次，从而产生额外的开销导致总体吞吐的下降。

图 7-2 模型推理耗时与 Forward Tokens 的拟合关系

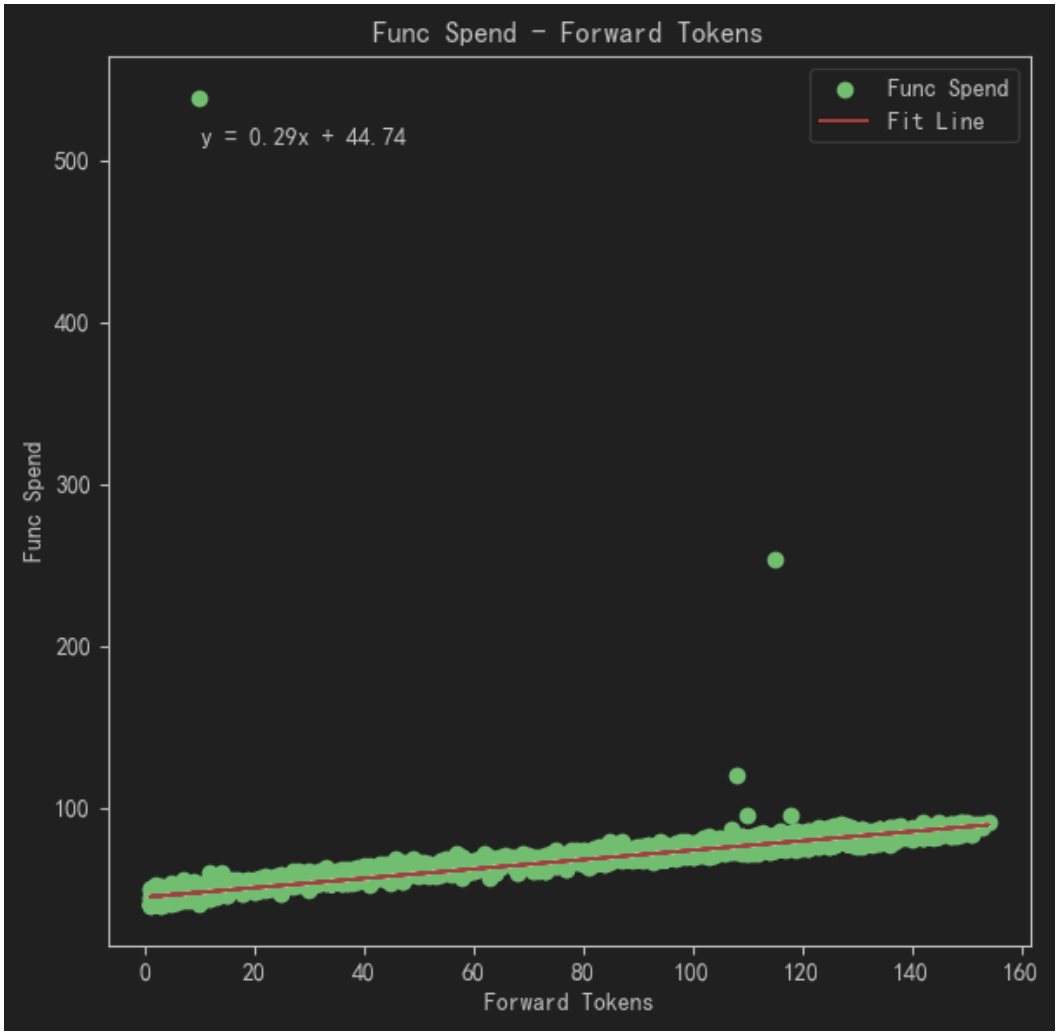
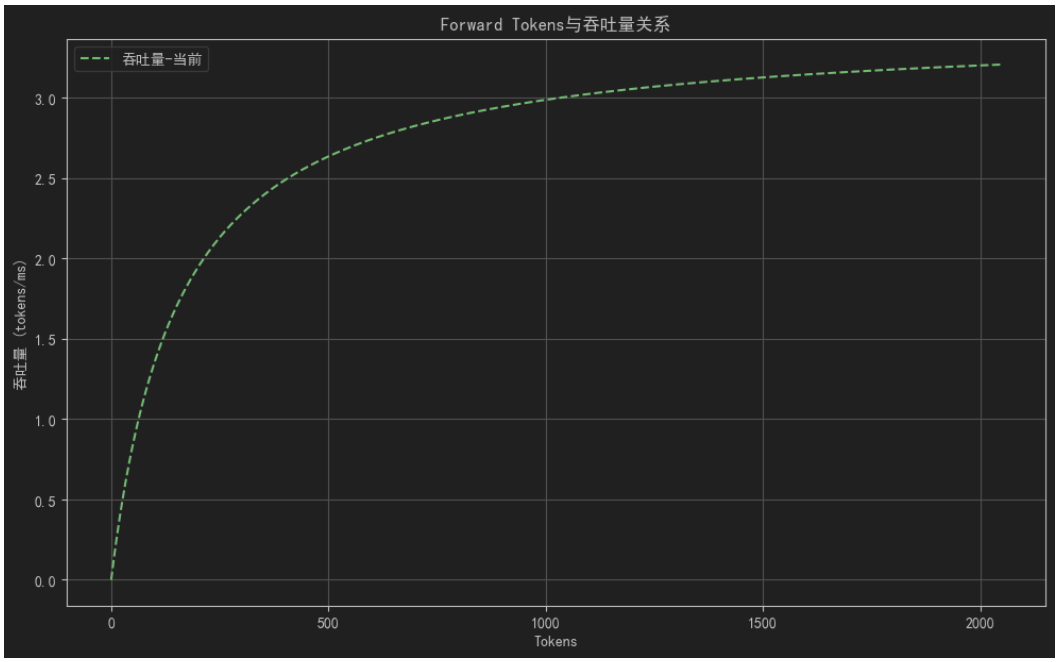


图 7-3 模型推理耗时与系统吞吐



操作步骤

步骤1 访问MindIE Service配置文件config.json，路径如下所示。

{MindIE安装目录}/latest/mindie-service/conf/config.json

步骤2 请参见表7-3中“plugin_params”、“templateType”和“enableSplit”参数开启Splitfuse特性。对于性能调优，需要编辑config.json配置文件中的**ScheduleConfig**部分，建议将“maxBatchSize”与“splitChunkTokens”参数配置为相同大小，并通过调整这两个参数的值来控制SLO的Decode时延指标。

请参见以下配置样例，SplitFuse的相关配置已加粗显示，参数具体解释请参见表7-3。

```
    "ModelDeployConfig":
    {
        "maxSeqLen" : 65536,
        "maxInputTokenLen" : 65536,
        "truncation" : false,
        "ModelConfig" : [
            {
                "modelInstanceType": "Standard",
                "modelName" : "LLaMA3-70B",
                "modelWeightPath" : "/home/models/LLaMA3-70B/",
                "worldSize" : 8,
                "cpuMemSize" : 5,
                "npuMemSize" : -1,
                "backendType": "atb",
                "plugin_params": "{\"plugin_type\":\"splitfuse\"}"
            }
        ]
    },
    "ScheduleConfig":
    {
        "templateType": "Mix",
        "templateName" : "Standard_LLM",
        "cacheBlockSize" : 128,

        "maxPrefillBatchSize" : 40,
        "maxPrefillTokens" : 65536,
        "prefillTimeMsPerReq" : 600,
```



```
        "prefillPolicyType" : 0,

        "decodeTimeMsPerReq" : 50,
        "decodePolicyType" : 0,
        "policyType": 0,

        "maxBatchSize" : 256,
        "maxIterTimes" : 512,
        "maxPreemptCount" : 0,
        "supportSelectBatch" : false,
        "maxQueueDelayMicroseconds" : 5000,

        "enableSplit": true,
        "splitType": false,
        "splitStartType": false,
        "splitChunkTokens": 256,
        "splitStartBatchSize": 16
    }
```

步骤3 本样例以MindIE Benchmark方式展示调优方式。config.json配置完成后，执行如下MindIE Benchmark启动命令。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName LLaMa3-70B \
--ModelPath "/{模型权重路径}/LLaMa3-70B" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 1000 \
--RequestRate 5 \
--TaskKind stream \
--Tokenizer True \
--MaxOutputLen 512
```

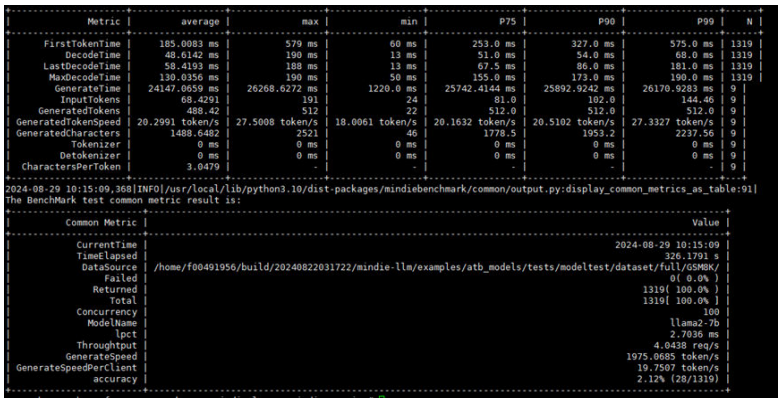
步骤4 根据首Token时延和Decode时延的实际数据调整参数。

- 首Token时延和Decode时延（均值，P90）都满足约束阈值，则加大“RequestRate”的值。
- Decode时延均值位于约束阈值以内，而首Token时延均值大于约束阈值。则“RequestRate”已大于系统吞吐GenerateSpeed，为满足约束需降低“RequestRate”的值。
- 当首Token时延均值和Decode时延均值满足阈值约束，而Decodes时延P90不满足均值时，则考虑降低ChunkSize减小切分，但该操作可能影响吞吐。
- 在输入问题长度不一的场景下，SplitFuse相对于基线SelectBatch场景通常拥有更好的首Token时延。

使用MindIE Benchmark对GSM8K数据集进行测试的结果如下：

- **打开SplitFuse**时：

  测试数据集：GSM8K；并发数：100；相对基线吞吐：101.59%；相对基线首Token时延：22.42%；



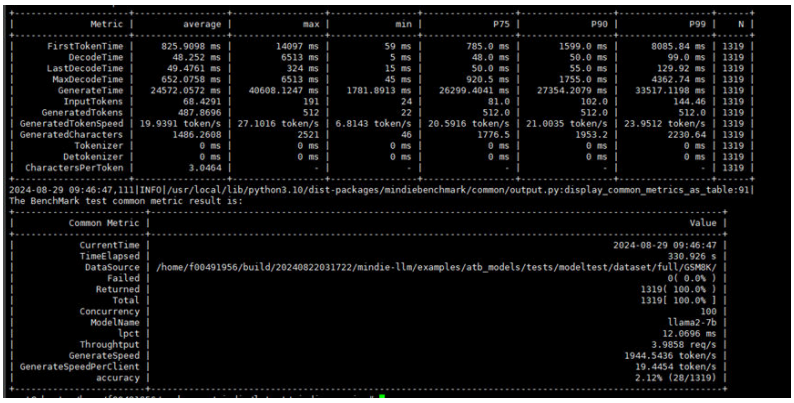
Metric	average	max	min	P75	P90	P99	N
FirstTokenTime	185.0083 ms	579 ms	60 ms	253.0 ms	327.0 ms	575.0 ms	1319
DecodeTime	48.6142 ms	190 ms	13 ms	51.0 ms	54.0 ms	68.0 ms	1319
LastDecodeTime	58.4193 ms	188 ms	13 ms	67.5 ms	86.0 ms	181.0 ms	1319
MaxDecodeTime	130.0356 ms	190 ms	50 ms	155.0 ms	173.0 ms	190.0 ms	1319
GenerateTime	24147.0659 ms	26268.6272 ms	1220.0 ms	25742.4144 ms	25892.9242 ms	26170.9283 ms	9
InputTokens	68.4291	191	24	81.0	102.0	144.46	9
GeneratedTokens	488.42	512	22	512.0	512.0	512.0	9
GeneratedTokenSpeed	20.2991 token/s	27.5008 token/s	18.0061 token/s	20.1632 token/s	20.5102 token/s	27.3327 token/s	9
GeneratedCharacters	1488.6482	2521	46	1778.5	1953.2	2237.56	9
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	9
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	9
CharactersPerToken	3.0479	-	-	-	-	-	9

2024-08-29 10:15:09,368|INFO|/usr/local/lib/python3.10/dist-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-08-29 10:15:09
TimeElapsed	326.1791 s
DataSource	/home/f00491956/build/20240822031722/mindie-llm/examples/atb_models/tests/modeltest/dataset/full/GSM8K/
Failed	0(0.0%)
Returned	1319(100.0%)
Total	1319[100.0%]
Concurrency	100
ModelName	llama2-7b
lpct	2.7036 ms
Throughtput	4.0438 req/s
GenerateSpeed	1975.0685 token/s
GenerateSpeedPerClient	19.7507 token/s
accuracy	2.12% (28/1319)

- **关闭SplitFuse，打开SelectBatch时：**

  测试集：GSM8K；并发数：100。



Metric	average	max	min	P75	P90	P99	N
FirstTokenTime	825.9098 ms	14097 ms	59 ms	785.0 ms	1599.0 ms	8085.84 ms	1319
DecodeTime	48.252 ms	6513 ms	5 ms	48.0 ms	50.0 ms	99.0 ms	1319
LastDecodeTime	49.4761 ms	324 ms	15 ms	50.0 ms	55.0 ms	129.92 ms	1319
MaxDecodeTime	652.0758 ms	6513 ms	45 ms	920.5 ms	1755.0 ms	4362.74 ms	1319
GenerateTime	24572.0572 ms	40608.1247 ms	1781.8913 ms	26299.4041 ms	27354.2079 ms	33517.1198 ms	1319
InputTokens	68.4291	191	24	81.0	102.0	144.46	1319
GeneratedTokens	487.8696	512	22	512.0	512.0	512.0	1319
GeneratedTokenSpeed	19.9391 token/s	27.1016 token/s	6.8143 token/s	20.5916 token/s	21.0035 token/s	23.9512 token/s	1319
GeneratedCharacters	1486.2608	2521	46	1776.5	1953.2	2230.64	1319
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	1319
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	1319
CharactersPerToken	3.0464	-	-	-	-	-	1319

2024-08-29 09:46:47,111|INFO|/usr/local/lib/python3.10/dist-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:91|
The BenchMark test common metric result is:

Common Metric	Value
CurrentTime	2024-08-29 09:46:47
TimeElapsed	330.926 s
DataSource	/home/f00491956/build/20240822031722/mindie-llm/examples/atb_models/tests/modeltest/dataset/full/GSM8K/
Failed	0(0.0%)
Returned	1319(100.0%)
Total	1319[100.0%]
Concurrency	100
ModelName	llama2-7b
lpct	12.0696 ms
Throughtput	3.9858 req/s
GenerateSpeed	1944.5436 token/s
GenerateSpeedPerClient	19.4454 token/s
accuracy	2.12% (28/1319)

----结束

7.7 Prefix Cache

7.7.1 特性介绍

当前大语言模型推理系统普遍采用KV Cache缓存机制，但该机制存在以下两个问题：

1. 随着LLM支持的序列长度不断增长，KV Cache所需要的显存资源也急剧增加。
2. KV Cache只对当前session有效，如果跨session存在重复token序列的情况下无法实现复用。

Prefix Cache通过RadixTree保留session结束后的KV Cache，新的session请求在RadixTree中查找是否存在相同的Token序列，即可复用之前计算好的KV Cache，从而实现跨session的KV Cache复用。

其优势主要包括：

- **更短的prefill时间**：由于跨session的重复token序列对应的KV Cache可以复用，那么就可以减少一部分前缀token的KV Cache计算时间，从而减少prefill的时间。
- **更高效的显存使用**：当正在处理的sessions相互之间存在公共前缀时，公共前缀部分的KV Cache可以共用，不必重复占用多份显存。

7.7.2 使用样例

限制与约束

- Atlas 800I A2 推理产品和Atlas 300I Duo 推理卡支持此特性。
- LLaMa系列、Qwen1、Qwen1.5和Qwen2系列模型支持对接此特性。
- 当跨session公共前缀token数大于等于Page Attention中的block size，才会进行公共前缀token的KV Cache复用。
- 该特性只支持单机（非分布式）服务部署场景。

操作步骤

本章节以多轮对话为例，简单介绍Prefix Cache如何使用。

步骤1 配置服务化参数，服务化参数说明请参见**8.4.2 配置参数说明**章节。

```
cd ${mindie-service安装路径}
vi conf/config.json
```

Prefix Cache特性需要额外配置的参数：

- 在ModelDeployConfig中的ModelConfig下添加以下参数：
  ```
  "plugin_params": "{\"plugin_type\":\"prefix_cache\"}"
  ```
- 在ScheduleConfig中添加以下参数：
  ```
  "enablePrefixCache": true
  ```

保存修改后的配置后启动服务化：

```
./bin/mindieservice_daemon
```

步骤2 第一次使用以下指令发送请求，prompt为第一轮问题。

如需使用到Prefix Cache特性，第二次请求的prompt需要与第一次的prompt有一定长度的公共前缀，常见使用场景有多轮对话和few-shot学习等。

```
curl https://127.0.0.1:1025/generate \
-H "Content-Type: application/json" \
--cacert ca.pem --cert client.pem  --key client.key.pem \
-X POST \
-d '{
"inputs": "Question: Parents have complained to the principal about bullying during recess. The principal wants to quickly resolve this, instructing recess aides to be vigilant. Which situation should the aides report to the principal?\na) An unengaged girl is sitting alone on a bench, engrossed in a book and showing no interaction with her peers.\nb) Two boys engaged in a one-on-one basketball game are involved in a heated argument regarding the last scored basket.\nc) A group of four girls has surrounded another girl and appears to have taken possession of her backpack.\nd) Three boys are huddled over a handheld video game, which is against the rules and not permitted on school grounds.\nAnswer:",
"parameters": {"max_new_tokens":512}
}'
```

步骤3 第二次发送请求，prompt为：第一轮问题+第一轮答案+第二轮问题，此时第一轮问题为可复用的公共前缀（实际复用部分可能不是第一轮问题的完整prompt；由于cache实现以block为单位，Prefix Cache以blocksize的倍数储存，如第一轮问题prompt的token数量为164，当blocksize为128时，实际复用部分只有前128token）。

```
curl https://127.0.0.1:1025/generate \
-H "Content-Type: application/json" \
--cacert ca.pem --cert client.pem  --key client.key.pem \
-X POST \
-d '{
"inputs": "Question: Parents have complained to the principal about bullying during recess. The principal wants to quickly resolve this, instructing recess aides to be vigilant. Which situation should the aides report
```



```
to the principal?\na) An unengaged girl is sitting alone on a bench, engrossed in a book and showing no interaction with her peers.\nb) Two boys engaged in a one-on-one basketball game are involved in a heated argument regarding the last scored basket.\nc) A group of four girls has surrounded another girl and appears to have taken possession of her backpack.\nd) Three boys are huddled over a handheld video game, which is against the rules and not permitted on school grounds.\nAnswer:c) A group of four girls has surrounded another girl and appears to have taken possession of her backpack.\nExplanation: The principal wants to quickly resolve this, instructing recess aides to be vigilant. The principal is concerned about bullying during recess. The principal wants the aides to report any bullying behavior to him. The principal is not concerned about the other situations.\nQuestion: If the aides confront the group of girls from situation (c) and they deny bullying, stating that they were merely playing a game, what specific evidence should the aides look for to determine if this is a likely truth or a cover-up for bullying?\nAnswer:",
"parameters": {"max_new_tokens":512}
}'
```

----结束

7.8 分布式多机部署

7.8.1 特性介绍

在深度学习领域，大语言模型LLM的权重参数快速增长，参数量早已迈向千亿、甚至万亿级别，占用显存量巨大。例如，1.4万亿参数的模型，仅参数（FP16）本身就会占用将近2.8TB的显存，单台推理机（通常8个NPU，每个NPU卡不到100GB显存）显存有限，无法容纳整个模型权重参数，因此需要多台推理机协同工作，共同完成整个模型的推理，即分布式多机推理。

7.8.2 环境部署

分布式多机部署环境准备如下所示：

- K8s的安装配置请参见**3.2.2 K8s安装与配置**；
- MindCluster组件的安装**3.2.3 MindCluster组件安装**；
- MindIE Server镜像制作请参见**3.3 准备MindIE Server镜像**。

7.8.3 使用样例

分布式多机部署使用样例如下所示：

基于MindIE MS调用K8s进行分布式多机部署详情请参见**3.5.2.2 使用MindIE MS Deployer部署分布式多机服务示例**。

7.9 PD 分离部署

7.9.1 特性介绍

Transformer大模型推理PD分离部署特性，主要是指模型推理的Prefill阶段和Decode阶段分别实例化部署在不同的机器资源上同时进行推理，其结合Prefill阶段的计算密集型特性，以及Decode阶段的访存密集型特性，通过调节PD节点数量配比来提升Decode节点的batch size来充分发挥NPU卡的算力，进而提升集群整体吞吐。此外，在Decode平均低时延约束场景，PD分离相比PD混合部署，更加能够发挥性能优势。

7.9.2 环境部署

PD分离部署环境准备如下所示：

- K8s的安装配置请参见**3.2.2 K8s安装与配置**；
- MindCluster组件的安装**3.2.3 MindCluster组件安装**；
- MindIE Server镜像制作请参见**3.3 准备MindIE Server镜像**。

7.9.3 使用样例

PD分离部署使用样例如下所示：

基于基于kubectl调用K8s进行PD分离部署详情请参见**3.6.3.2.3 使用kubectl部署PD分离服务示例**。

7.9.4 性能调优

7.9.4.1 PD 配比调优

7.9.4.1.1 配比调优理论原理

PD分离部署场景下，一般实例都加载相同模型。如何分配实例的初始属性，并根据实际需求动态调整实例属性。不合理的实例配比将造成Prefill实例等待空闲或Decode实例等待空闲，造成资源浪费，最终在MFU和端到端吞吐性能上产生劣化，无法发挥PD分离调度架构的优势。

将整个PD分离系统看做**生产消费模型，P实例生产KV，PD之间传输KV，D实例消费KV**，三者组成Pipeline完成大模型推理。当三者中任一速率低并成为瓶颈，就会产生请求堆积，进而影响整体的吞吐量和时延。请求增加输入长度降低Prefill生产速率（计算量增加），同时也降低Decode消费速率（KV Cache访存增加），增加P和D实例可以提高生产和消费速率。

因此，PD分离系统良好运行的关键在于满足时延SLO约束下，那么面对不同的请求分布，尽可能提高这三种速率。PD配比寻优保持一个原则：**使Prefill速率、Decode速率、传输速率三者互不为短板。**

Prefill和Decode实例最佳配比搜索（假设Prefill实例为M，Decode实例数为N。）：

- Prefill实例按最大速率处理Prefill batch，那么每个Prefill实例最大的处理速率为：S_p
- 若请求能被均匀调度到各个Decode实例，那每个Decode实例到达请求速率可以估计为：$S_d = S_p * \frac{M}{N}$
- Decode实例最大Batch数为B_{dmax}
- 在稳态情况下，若Decode实例每次都能以最大Batch数调度Decode，那每次调度Decode完成时间为：$T_d^{(B_{dmax})}$

$$E_{decode} = \frac{1}{T_d^{(B_{dmax})}}$$

- Decode实例上，每秒平均Decode调度epoch为：
- 单个Decode实例总吞吐为：$V^{(1)} = E_{decode} * B_{dmax}$
- 所有Decode实例总吞吐为：$V^{(N)} = N * E_{decode} * B_{dmax}$
- 系统平均每卡吞吐为：

$$V^{(avg)} = \frac{V^{(N)}}{M+N}$$

$$V^{(avg)} = \frac{V^{(N)}}{M+N} = \frac{NE_{decode}B_{dmax}}{M+N} = \frac{NB_{dmax}}{(M+N)T_d^{(B_{dmax})}}$$

对所有可能的M:N配比进行搜索，求得最大的平均每卡吞吐即为最佳配比。

7.9.4.1.2 手动配比调优

手动配比调优操作步骤如下所示。

步骤1 修改MindIE Server的config.json文件的部分参数如下所示配置，更多参数的配置请参见5.1 性能调优流程。

- 对于短输入数据集：“maxPrefillBatchSize”可以设置大一些，比如设置为10；
- 对于中长输入数据集：“maxPrefillBatchSize”可以设置为1，以降低首token时延；“maxBatchSize”可以设置的大一些，比如设置为50。

```
"ModelDeployConfig":
{
    "maxSeqLen" : 9728,
    "maxInputTokenLen" : 8704,
    "truncation" : false,
    "ModelConfig" : [
        {
            "modelInstanceType": "Standard",
            "modelName" : "llama3_8b",
            "modelWeightPath" : "/home/data/llama3-8B",
            "worldSize" : 1,
            "cpuMemSize" : 40,
            "npuMemSize" : 11,
            "backendType": "atb"
        }
    ]
},

"ScheduleConfig":
{
    "templateType": "Standard",
    "templateName" : "Standard_LLM",
    "cacheBlockSize" : 128,

    "maxPrefillBatchSize" : 1,
    "maxPrefillTokens" : 9728,
    "prefillTimeMsPerReq" : 1000,
    "prefillPolicyType" : 0,

    "decodeTimeMsPerReq" : 50,
    "decodePolicyType" : 0,

    "maxBatchSize" :50,
    "maxIterTimes" : 1024,
```

```
        "maxPreemptCount" : 0,
        "supportSelectBatch" : false,
        "maxQueueDelayMicroseconds" : 5000
    }
```

步骤2 配置完成后，用户可使用Linux curl命令、Postman工具、Benchmark工具等方式发送推理请求，此处以MindIE Benchmark命令为例进行说明。

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \          //数据集路径
--DatasetType "gsm8k" \                                  //数据集类型
--ModelName LLaMa3-8B \                                    //模型名称
--ModelPath "/{模型路径}/LLaMa3-8B" \                                //模型路径
--TestType client \                                  //模式选择
--Http https://{ipAddress}:{port} \                      //请求url
--ManagementHttp https://{managementIpAddress}:{managementPort} \                    //管理面url
--Concurrency 1000 \                                 //并发数
--RequestRate 5                                      //请求频率
--TaskKind stream \                                    //Client不同推理模式，此处为文本流式推理
--Tokenizer True \                                   //分词向量化标识
--MaxOutputLen 512                                     //最大输出长度
```

步骤3 首先使用PD配比为1:1进行测试，根据测试的Prefill和Decode时延结果，不断调整PD配比达到最优配比即可。

手动调整PD配比需要配置Controller配置文件中的两个参数“default_p_rate”和“default_d_rate”，如果要配置PD配比为2:3，那么相应的“default_p_rate”和“default_d_rate”参数配置如下所示。

```
{
    "deploy_mode": "pd_separate",
    "process_manager_file_path": "./log/controller_process_status.json",
    "cluster_status_output_file_path": "./log/cluster_status_output.json",
    "global_rank_table_file_path": "./conf/global_rank_table_file.json",
    "cluster_synchronization_seconds": 1,
    "rank_table_detecting_seconds": 60,
    "disappeared_server_waiting_seconds": 120,
    "default_p_rate": 2,
    "default_d_rate": 3,
    "is_heterogeneous": false,
```

根据首token时延和Decode时延的实际数据，最多出现下面四种情况，根据实际结果调整参数：

- 首token时延和Decode时延都没超过约束阈值，则加大“RequestRate”的值即可。
- 首token时延和Decode时延都超过约束阈值，则减小RequestRate即可。
- 首token时延超过约束阈值，说明Prefill节点的数量匹配不上Decode节点的数量，可以适当增加Prefill节点的数量以降低首token时延。
- Decode时延超过约束阈值，说明Decode节点的数量匹配不上Prefill节点的数量，可以适当增加Decode节点的数量以降低Decode时延。

步骤4 确定好最佳PD配比后，Prefill和Decode时延理论上可以同时接近时延约束，因此不断调整“requestRate”参数的值获得满足时延约束下的最大吞吐。

首token时延约束1000ms，Decode时延约束50ms的情况下，调优后两者比较理想的数值如下图所示，更多手动PD配比调优示例如**表7-4**所示。

Metric	average	max	min	P75	P90
FirstTokenTime	465.1421 ms	2008.104 ms	137.806 ms	588.644 ms	969.8731 ms
DecodeTime	44.7129 ms	134.83 ms	8.776 ms	46.501 ms	47.725 ms
LastDecodeTime	42.1503 ms	54.524 ms	10.495 ms	46.1658 ms	46.9208 ms
MaxDecodeTime	95.8202 ms	134.83 ms	46.908 ms	107.945 ms	114.2465 ms
GenerateTime	18277.4516 ms	25788.5656 ms	655.8142 ms	24087.8224 ms	24434.6123 ms
InputTokens	1948.5741	2047	1396	2035.0	2047.0
GeneratedTokens	398.486	513	15	513.0	513.0
GeneratedTokenSpeed	21.3899 token/s	32.6087 token/s	12.981 token/s	21.9528 token/s	23.8007 token/s
GeneratedCharacters	1697.3126	3436	42	2417.0	2536.0
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms
CharactersPerToken	4.2594	-	-	-	-
PrefillBatchsize	1.0	1	1	1.0	1.0
DecoderBatchsize	25.2847	35	1	30.0	31.0
QueueWaitTime	942.9332 μs	1817512 μs	6 μs	237.0 μs	299.0 μs

表 7-4 PD 配比调优示例

模型	数据类型	平均输入长度	平均输出长度	首token时延约束	Decode时延约束	初始配比	示例调优后配比
llama3-8b	short	258	59	500	50	1:1	2:3
	medium1	1995	369	1000	50	1:1	2:3
	medium2	957	114	500	50	1:1	1:3
	long	6813	932	3000	50	1:1	1:5
llama3-70b	short	258	59	500	50	1:1	1:2
	medium1	1995	369	1000	50	1:1	1:2
	medium2	957	114	500	50	1:1	1:2
	long	6813	932	3000	50	1:1	1:2

----结束

7.9.4.1.3 自动配比调优

使用自动调整配比的算法，需要先配置Controller配置文件中的“default_p_rate”和“default_d_rate”两个参数，把这两个参数的值设置为0时，将会触发最佳PD比例计算，自动计算出最佳PD比例。Controller配置文件中其他关键配置参数如表7-5所示。

表 7-5 Controller 配置文件关键参数

参数名称	说明	配置值
digs_request_summary_input_length	推理请求的平均输入长度。 取值范围：[0,65535]	默认值：3000

参数名称	说明	配置值
digs_request_summary_output_length	推理请求的平均输出长度。 取值范围：[0,65535]	默认值：200
digs_model_config_path	身份决策算法需要使用的模型参数信息。	/usr/local/Ascend/mindie/latest/mindie-service/conf/model_config/llama2-70B.json
digs_machine_config_path	身份决策算法需要使用的机器参数信息。	/usr/local/Ascend/mindie/latest/mindie-service/conf/machine_config/ascend910*xx*.json
digs_prefill_slo	首token时延约束。 取值范围：[0,65535]	默认值：1000
digs_decode_slo	Decode时延约束。 取值范围：[0,65535]	默认值：50
model_type	模型名称。	默认值：llama-70B
transfer_type	传输类型。	默认值：D2DTransfer
digs_pp/usr/local/Ascend/llm_model/	任务并行数。 取值范围：[0,65535]	默认值：1

7.10 模型量化

模型量化介绍详情请参见《 MindIE LLM开发指南 》的“特性介绍 > **量化特性介绍**”章节。

8 MindIE Service 组件

8.1 MindIE Service Tools

8.1.1 MindIE Benchmark

8.1.1.1 功能介绍

服务化MindIE Benchmark工具是通过部署昇腾服务化配套包后，以调用终端命令的方式测试大语言模型在不同配置参数下的推理性能和精度，并通过表格的形式展示模型在各个阶段的推理耗时（例如FirstTokenTime、DecodeTime等），以及对应时延的平均值、最小值、最大值、75分位（P75）、90分位（P90、SLO_P90）和99分位（P99）概率统计值，最后将计算结果保存到本地csv文件中。

须知

Benchmark当前运行路径的属主和属组必须和当前用户所在组对应，可以通过 "ls -l <路径>" 查看指定路径的属主和属组，通过 "chown <属主>:<属组> <路径>" 的方式更改指定路径的属主和属组。日志文件、权重配置文件和用户配置文件等通常涉及文件或目录属主/属组检验。

应用场景

- 支持Client和Engine两种不同的推理模式：
  - Client模式：MindIE Benchmark支持调用MindIE Client接口的方式，与Server-Endpoint进行通信并完成测试。

- 文本模式：此模式输入和接收的数据均为文本形式；该模式下支持全量文本生成及流式文本生成两种，调用MindIE Client的**.generate()**和**.generate_stream()**接口，对应MindIE Server的**兼容Triton的文本推理接口**和**兼容Triton的流式推理接口**。

  须知

  MindIE Client与Server-Endpoint之间的通信会因网络波动影响最终统计的性能结果。

- Engine模式：MindIE Benchmark支持通过直接调用MindIE LLM中LLM Manger提供的Python接口进行全量推理，接口详情请参见《MindIE LLM开发指南》的“API接口说明 > **LLM Manger提供的Python接口**”。
  - 支持token ID到token ID异步推理，数据集转换为token ID的具体转换方法请参考**10.3 数据集使用**。
  - 支持文本到文本的异步推理。

- 支持精度测试的数据集，包括CEval 5-shot、GSM8K和MMLU 5-shot，Engine模式和Client模式都支持测试，请参见**10.3 数据集使用**获取数据集。
- 支持性能测试的数据集，包括Gsm8k、OA、CEval 5-shot、MMLU 5-shot、BoolQ、HumanEval、mtbench和cocotest，主要使用GSM8K和OA两个数据集来测试模型的性能，请参见**10.3 数据集使用**获取数据集。

说明

- Client模式适用于模拟多用户并发场景，兼容TGI、Triton和vLLM等多种接口，主要用于测量服务化性能。Client测量的吞吐量为用户真实感知的吞吐量，其计入包括网络请求和数据处理等消耗的时间。
- Engine模式直接调用底层API，并将NPU推理返回的结果暂存，当所有推理完成后再由CPU处理暂存的数据，其测量的吞吐量更接近NPU卡的真实性能。
- Engine模式中使用到的ModelName（模型名称）和ModelPath（模型权重路径）需要和MindIE Server的config.json配置文件中modelName（模型名称）和modelWeightPath（模型权重路径）参数保持一致，而npuDeviceIds（NPU卡编号）和maxBatchSize（最大decode batch size）参数的值是由MindIE Server中的config.json配置文件决定，详情请参见**8.4.2 配置参数说明**。

8.1.1.2 配置参数

前提条件

- 已参考《**MindIE安装指南**》中“物理机部署MindIE”章节进行环境的安装与部署，MindIE Benchmark会随着MindIE的安装自动安装。MindIE Benchmark的Wheel包所在路径/*{mindie-install-path}*/mindie/{version}/mindie-service/bin/mindiebenchmark-*{version}*-py3-none-any.whl。
- 使用镜像部署MindIE时，已参考《MindIE安装指南》中“安装开发环境 > **安装依赖**”章节完成MindIE所需依赖的安装。
- tritonclient使用grpc通信时，需要依赖grpcio等模块，使用以下命令进行安装。
  ```
  pip install tritonclient[all]
  ```
- 可视化功能需要依赖matplotlib模块，使用以下命令进行安装。
  ```
  pip install matplotlib
  ```


- 使用Client推理模式需要提前启动MindIE Server 服务，Engine推理模式不需要提前启动。

操作步骤

步骤1 根据用户需要设置配置参数。

进入MindIE Benchmark的安装路径*{$HOME}/{python版本}*/site-packages/mindiebenchmark/config并打开“config.json”文件。

```
vim config.json
```

说明

使用以下命令可查看MindIE Benchmark的安装路径。

```
pip show mindiebenchmark
```

config.json文件如下所示：

```
{
  "INSTANCE_PATH": "./instance",            # Benchmark结果保存路径
  "LOG_PATH": "./instance",                 # 日志保存路径
  "LOG_LEVEL": "DEBUG",                     # 日志等级
  "CA_CERT": "/path/to/cacert.pem",         # 验签证书文件路径
  "KEY_FILE": "/path/to/cacert.pem.key",    # 客户端私钥文件路径
  "CERT_FILE": "/path/to/client.pem",       # 客户端证书文件路径
  "CRL_FILE": "/path/to/crl.pem",           # 客户端吊销列表路径
  "ENABLE_MANAGEMENT": false,               # 管理端口使能
  "MAX_LINK_NUM": 1000                      # 服务端最大连接数（当前最大支持1000）
}
```

表 8-1 MindIE Benchmark 配置参数说明

参数	参数类型	说明
INSTANCE_PATH	String	MindIE Benchmark结果保存路径，默认保存在./instance。
LOG_PATH	String	MindIE Benchmark日志保存路径，默认保存在./instance。
LOG_LEVEL	String	日志等级，默认为DEBUG。 其他等级为"INFO"、"WARNING"、"ERROR"和"CRITICAL"。
CA_CERT	String	验签证书文件路径，为MindIE Server服务端证书的验签证书/根证书。 当使用MindIE Benchmark的Client推理模式且MindIE Server的配置参数httpsEnabled为true时，必须配置。
KEY_FILE	String	客户端私钥文件路径。 当使用MindIE Benchmark的Client推理模式且MindIE Server的配置参数httpsEnabled为true时，必须配置。

参数	参数类型	说明
CERT_FILE	String	客户端证书文件路径。 当使用MindIE Benchmark的Client推理模式且MindIE Server的配置参数httpsEnabled为true时，必须配置。
CRL_FILE	String	客户端吊销列表证书文件路径。 可选配置，当使用MindIE Benchmark的Client推理模式且MindIE Server的配置参数httpsEnabled为true时，会校验验签证书是否在吊销列表内。
ENABLE_MANAGEMENT	bool	MindIE Benchmark管理端口使能。 MindIE Benchmark是否通过管理端口查询服务端健康状态。 默认值：false
MAX_LINK_NUM	Int	服务端最大连接数，默认为1000。当前支持最大连接数为1000，不可配置超过1000。

步骤2 （可选）如果需要使用合成数据（synthetic），指定输入输出长度进行性能测试，则需要指定合成数据长度的采样方式。

合成数据自动生成，指定输入token数量后，将生成内容全为 "A" 的数据作为prompt，中间用空格连接（例如：token数量为5，则对应的合成数据为 "A A A A A"）。输出数据的token数量也可以指定，此时模型不会因为结束符eos而停止输出。

进入MindIE Benchmark的安装路径*{$HOME}|{python版本}*/site-packages/mindiebenchmark/config并打开“synthetic_config.json”文件。

```
vim synthetic_config.json
```

“synthetic_config.json”文件配置样例如下所示：

```
{
    "Input":{
        "Method": "uniform",
        "Params": {"MinValue": 1, "MaxValue": 200}
    },
    "Output": {
        "Method": "gaussian",
        "Params": {"Mean": 100, "Var": 200, "MinValue": 1, "MaxValue": 100}
    },
    "RequestCount": 100
}
```

其中，采样的的可选采样方法和对应的参数请参见**表8-2**。

表 8-2 合成数据可选采样方法和对应参数

名称	采样方法（Method）	参数（Params）
均匀分布	uniform	{"MinValue": *最小值*, "MaxValue": *最大值*}
高斯分布	gaussian	{"Mean": *均值*, "Var": *方差*, "MinValue": *最小值*, "MaxValue": *最大值*}

名称	采样方法（Method）	参数（Params）
zipf分布	zipf	{"Alpha": *Zipf参数*, "MinValue": *最小值*, "MaxValue": *最大值*}

其中，使用各种采样方法采样后的数字将再通过MinValue和MaxValue进行截断，确保采样的数值在[MinValue, MaxValue]范围内。MinValue和MaxValue为对应的token数量，具体的取值范围请参见**表8-3**，计数时只计算增量Token的数量。

表 8-3 合成数据参数配置说明

参数	含义	取值范围
Input	输入配置	-
Output	输出配置	-
RequestCount	请求次数，即样本数量	[1,1048576]
Method	采样方法	取值如下： • uniform • gaussian • zipf
Params	采样方法中对应的采样参数	取值详情请参见**表8-2**。
"Input" 中的 "MinValue"	token数量最小值	[1,1048576]
"Input" 中的 "MaxValue"	token数量最大值	[1,1048576]
"Output" 中的 "MinValue"	token数量最小值	[1,1048576]
"Output" 中的 "MaxValue"	token数量最大值	[1,1048576]
"gaussian" 中的 "Mean"	高斯分布均值	[-3.0 x 10^38, 3.0 x 10^38]
"gaussian" 中的 "Var"	高斯分布方差	[0, 3.0 x 10^38]
"zipf" 中的 "Alpha"	zipf分布Alpha系数	(1.0,10.0]
注：1048576 = 2^20 = 1 M。		

----结束

8.1.1.3 API 接口

8.1.1.3.1 输入参数

MindIE Benchmark的Engine推理模式和Client推理模式输入参数说明如表8-4所示。

表 8-4 MindIE Benchmark 输入参数

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--DatasetPath	String	必选。 数据集文件所在的目录，根据数据集路径获取数据，解析后根据批次和分词标识进行处理。 • Tokenizer为True时：采用**文本推理模式**，该参数应该传入原始数据集文件夹路径； • Tokenizer为False时：采用**tokenids推理模式**，该参数传入已经将数据集处理为tokenids的csv文件的路径。（--Tokenizer为False时，数据集转换方式参考**10.3 数据集使用**）	必选。 数据集文件所在的目录；根据数据集路径获取数据，解析后根据批次和分词标识进行处理。 • Tokenizer为True时，采用**文本推理模式**，该参数传入原始数据集文件夹路径。 • Tokenizer为False时：采用**token推理样例**，该参数传入已经将数据集处理为tokenids的csv文件的路径。（--Tokenizer为False时，数据集转换方式参考**10.3 数据集使用**）
--DatasetType	Enum	必选。 数据集类型，枚举值：ceval、gsm8k、oasst1、boolq、truthfulqa、humaneval、mmlu、mtbench、cocotest和synthetic。	必选。 数据集类型，枚举值：ceval、gsm8k、oasst1、boolq、truthfulqa、humaneval、mmlu、mtbench、cocotest和synthetic。
--ModelName	String	必选。 模型名称。与MindIE Server**ModelConfig参数说明**中的modelName参数保持一致。	必选。 模型名称；与MindIE Server**ModelConfig参数说明**中的modelName参数保持一致，用于client-sdk拼接得到endpoint对应的uri：/v2/models/*{model-name}*/*{infer,generate,generate_stream}*。
--ModelPath	String	必选。 模型路径；传递模型初始化配置。与MindIE Server**ModelConfig参数说明**中的modelWeightPath参数保持一致。	必选。 模型路径；传递模型初始化配置。与MindIE Server**ModelConfig参数说明**中的modelWeightPath参数保持一致。

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--ModelVersion	String	不涉及。	可选。 模型版本；在Endpoint部署同一模型多个版本时需要提供，client-sdk对应拼接的uri地址为：/v2/models/*{model-name}*/versions/*{ModelVerson}*/*{generate,generate_stream}*
--TestType	String	必选。 模式选择；取值为engine。	必选。 模式选择；取值为client、vllm_client、tgi_client和triton_client。其中vllm_client、tgi_client或triton_client只能基于对应的三方框架上才能运行。
--Http	String	不涉及。	必选。 请求url，默认为"https://127.0.0.1:1025"，实际请根据MindIE Server**ServerConfig参数说明**中的ipAddress和port参数进行配置。
--Concurrency	Int	可选。 并发数，限制同时发起的连接数。 取值范围：[1，1000]，默认值：128。	可选。 并发数，限制同时发起的连接数。Client模式下，不超过endpoint所支持的最大连接数。取值范围：[1，1000]，默认值：128。
--TaskKind	String	不涉及。	可选。 判断使用Client的哪种模式进行推理，默认值为stream。 • stream：流式推理接口； • text：非流式推理接口； • stream_token：token流式推理接口。
--DoSampling	Bool	可选。 是否进行推理结果后处理，取值True或者False，默认值为False。为True时，需要与SamplingParams参数一起使用。	可选。 是否进行推理结果后处理，取值True或者False，默认值为False。为True时，需要与SamplingParams参数一起使用。

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--TestAccuracy	Bool	可选。 是否测试精度标识，取值True或者False，默认值为False。 MindIE Benchmark目前支持精度测试数据集：ceval、gsm8k和mmlu。 仅在Tokenizer=True，非token_id的输入场景下才能配置该参数。	可选。 测试精度标识，取值True或者False，默认值为False。 MindIE Benchmark目前支持精度测试数据集：ceval、gsm8k和mmlu。 仅在Tokenizer=True，非token_id的输入场景下才能配置该参数。
--SavePath	String	可选。 输出结果保存路径*{SavePath}*，默认保存在${INSTANCE_PATH}路径下。	可选。 输出结果保存路径*{SavePath}*，默认保存在${INSTANCE_PATH}路径下，INSTANCE_PATH为benchmark配置参数，详情请参表8-1。
--ConfigPath	String	可选。 MindIE Benchmark配置文件config.json，默认为安装路径：*{$HOME}*/python3.10/site-packages/mindiebenchmark/config/config.json。	可选。 MindIE Benchmark配置文件config.json，默认安装路径为：*{$HOME}*/python3.10/site-packages/mindiebenchmark/config/config.json。
--Tokenizer	Bool	可选。 分词向量化标识，取值True或者False，默认值为True。 ● True：输入结果为文本； ● False：表示输入结果为tokens。	可选。 分词向量化标识，取值True或者False，默认值为True。 ● True：输入结果为文本； ● False：表示输入结果为tokens。
--MaxInputLen	Int	可选。 最大输入tokenids长度，超过会截断，取值范围：[0，4096]；默认不开启，取值为0。	不涉及。

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--MaxOutputLen	Int	可选。 最大输出长度，取值范围：[1，4096]，默认值：20。 • 该参数与**ScheduleConfig参数说明**中的“maxlterTimes”参数取小作为最大输出长度。 • 当“DatasetType”取值为“synthetic”时，该参数无效，通过配置文件synthetic_config.json中的“Output”参数控制最大输出长度，详情请参见**步骤2**。	可选。 最大输出长度，取值范围：[1，4096]，默认值：20。 • 当TaskKind=triton_client时，该参数无效，通过设置“maxlterTimes”参数控制最大输出长度，参数解释请见**ScheduleConfig参数说明**。 • 当“DatasetType”取值为“synthetic”时，该参数无效，通过配置文件synthetic_config.json中的“Output”参数控制最大输出长度，详情请参见**步骤2**。
--SamplingParams	/	可选。 采样参数，采样标识DoSampling为True时有效。 SamplingParams参数取值说明请参见《MindIE LLM开发指南》的“API接口说明 > Infer Engine API参考（C++） > 结构体和枚举说明 > **SamplingParams结构体**”章节。	可选。 输出结果采样标识DoSampling为True时有效。 SamplingParams参数取值说明如下所示： • stream（文本流式推理接口）请参见**10.4.1 benchmark的client模式的文本流式推理后处理参数**。 • text（文本非流式推理接口）请参见**10.4.2 benchmark的client模式的文本非流式推理后处理参数**。 • stream_token（token流式推理接口）请参见**10.4.3 benchmark的client模式的token推理后处理参数**。
--SaveTokensPath	String	可选。 保存推理输入输出的tokenids到本地路径，以.csv结尾。 例如：./output.csv。	可选。 保存推理输入输出的tokenids到本地路径，以.csv结尾。 例如：./output.csv。

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--SaveHumanEvalOutputFile	String	可选。 保存推理HumanEval数据集时输出文本结果路径，以.jsonl结尾。默认值为humaneval_output.jsonl，需要同时开启精度--TestAccuracy True。	可选。 保存推理HumanEval数据集时输出文本结果路径，以.jsonl结尾。默认值为humaneval_output.jsonl，需要同时开启精度--TestAccuracy True。
--RequestRate	String	不涉及。	可选。 指定一组发送频率，按照Distribution参数设置的模式进行发送，以每个频率完成一次推理，频率的取值范围：（0，10000]。 例如：--RequestRate "2,4,6,8"。 当推荐concurrency参数配置为1000时，可以达到高性能、低时延。
--Distribution	String	不涉及。	可选。 请求发送模式，仅在设置了RequestRate参数才生效，默认值为uniform。 ● uniform：请求发送为均一分布； ● poisson：请求发送为泊松分布。
--WarmupSize	Int	可选。 warm up的条数，默认值：10。取值范围[0, 1000]。 当取值为0时，不进行warm up。	可选。 warm up的条数，默认值：10。取值范围[0, 1000]。 当取值为0时，不进行warm up。
--ManagementHttp	String	不涉及。	可选。 管理端口的url，默认为"https://127.0.0.2:1026"，与server config.json的managementIpAddress和managementPort保持一致。 实际使用协议要和Http参数设置保持一致。 当MindIE Benchmark的config.json中ENABLE_MANAGEMENT=false时，该参数无效。

参数	参数类型	说明（Engine推理模式）	说明（Client推理模式）
--SyntheticConfigPath	String	可选。 合成数据配置路径，其为json文件。仅当“--DatasetType”为“synthetic”且“--TestType”为“engine”或“vllm_client”有效。 默认值：*{$HOME}/{Python版本}*/site-packages/mindiebenchmark/config/synthetic_config.json。 文件配置内容请参见表8-2。	可选。 合成数据配置路径，其为json文件。仅当“--DatasetType”为“synthetic”且“--TestType”为“engine”或“vllm_client”有效。 默认值：*{$HOME}/{Python版本}*/site-packages/mindiebenchmark/config/synthetic_config.json。 文件配置内容请参见表8-2。
注： 1. 对于所有的数据集性能测试默认开启， 精度测试开启需要添加参数--TestAccuracy True，且仅支持ceval、gsm8k和mmlu数据集。 2. 输入数据token长度加上最大输出长度--MaxOutputLen不能超过ModelDeployConfig参数说明中的maxSeqLen长度，建议使用--MaxInputLen做输入截断。 3. 当“--DatasetType”配置为“synthetic”，使用--SyntheticConfigPath的配置确定输入和输出token数量时，此时“--MaxOutputLen”参数配置的值将不会生效，其中输入的token全为大写字母“A“。			

说明

- 对于所有的数据集性能测试默认开启， 精度测试开启需要添加参数“--TestAccuracy”参数配置为“True”时，且仅支持ceval、gsm8k和mmlu数据集。
- 输入数据token长度加上最大输出长度“--MaxOutputLen”不能超过ModelDeployConfig参数说明中的“maxSeqLen”长度，建议使用“--MaxInputLen”做输入截断。
- 当“--DatasetType”配置为“synthetic”，使用“--SyntheticConfigPath”的配置确定输入和输出token数量时，此时“--MaxOutputLen”参数配置的值将不会生效，其中输入的token全为大写字母“A“。
- 当 “--DatasetType”参数取值为“synthetic”时，仅支持“--TestType”参数只能配置为“vllm_client”或“engine”。
- 性能测试推荐用OA数据集或gsm8k数据集，精度测试推荐ceval数据集，数据集获取请参见10.3 数据集使用。
- warm up通过模拟推理负载和触发硬件和软件系统的初始化，减少推理的延迟，保证推理的稳定性。在NPU或其他深度学习加速卡上，使用warm up能够避免首次推理时的性能瓶颈。

8.1.1.3.2 输出参数

输出参数的统计指标如表8-5和表8-6所示，部分统计指标解释如下所示：

- P75：统计的颗粒度以token为单位，以DecodeTime为例，所有请求的DecodeTime的75分位。
- P90：统计的颗粒度以token为单位，以DecodeTime为例，所有请求的DecodeTime的90分位。

- P99：统计的颗粒度以token为单位，以DecodeTime为例，所有请求的DecodeTime的99分位。
- SLO_P90：统计的颗粒度以请求为单位，以DecodeTime为例，首先每个请求的所有DecodeTime取平均值，然后对所有请求的平均DecodeTime取90分位。

表 8-5 单个推理请求性能输出结果

参数	说明（average）	说明（max）	说明（min）	说明（P75）	说明（P90）	说明（SLO_P90）	说明（P99）	说明（N）
FirstTokenTime	首个token平均时延，单位（ms）	首个token最大时延，单位（ms）	首个token最小时延，单位（ms）	首个token75分位时延，单位（ms）	首个token90分位时延，单位（ms）	首个token90分位时延，单位（ms）	首个token99分位时延，单位（ms）	测试数据量，来源于输入参数
DecodeTime	Decode阶段平均时延，单位（ms）	最大Decode阶段时延，单位（ms）	最小Decode阶段时延，单位（ms）	75分位Decode阶段时延，单位（ms）	90分位Decode阶段时延，单位（ms）	90分位每条请求Decode阶段平均时延，单位（ms）	99分位Decode阶段时延，单位（ms）	测试数据量，来源于输入参数
LastDecodeTime	最后一个token平均时延，单位（ms）	最后一个token最大时延，单位（ms）	最后一个token最小时延，单位（ms）	最后一个token75分位时延，单位（ms）	最后一个token90分位时延，单位（ms）	最后一个token90分位时延，单位（ms）	最后一个token99分位时延，单位（ms）	测试数据量，来源于输入参数
MaxDecodeTime	所有请求最大Decode阶段平均时延，单位（ms）	所有请求最大Decode阶段时延，单位（ms）	所有请求最小Decode阶段时延，单位（ms）	所有请求75分位Decode阶段时延，单位（ms）	所有请求90分位Decode阶段时延，单位（ms）	所有请求90分位Decode阶段时延，单位（ms）	所有请求99分位Decode阶段时延，单位（ms）	测试数据量，来源于输入参数

参数	说明（average）	说明（max）	说明（min）	说明（P75）	说明（P90）	说明（SLO_P90）	说明（P99）	说明（N）
GenerateTime	请求推理平均时延，单位（ms）	最大请求推理时延，单位（ms）	最小请求推理时延，单位（ms）	75分位请求推理时延，单位（ms）	90分位请求推理时延，单位（ms）	90分位请求推理时延，单位（ms）	99分位请求推理时延，单位（ms）	测试数据量，来源于输入参数
InputTokens	输入token平均长度	最大输入token长度	最小输入token长度	75分位输入token长度	90分位输入token长度	90分位输入token长度	99分位输入token长度	测试数据量，来源于输入参数
GeneratedTokens	生成token平均长度	最大生成token长度	最小生成token长度	75分位生成token长度	90分位生成token长度	90分位生成token长度	99分位生成token长度	测试数据量，来源于输入参数
GeneratedTokenSpeed	生成token平均速度，单位（token/s）	最大生成token速度，单位（token/s）	最小生成token速度，单位（token/s）	75分位生成token速度，单位（token/s）	90分位生成token速度，单位（token/s）	90分位生成token速度，单位（token/s）	99分位生成token速度，单位（token/s）	测试数据量，来源于输入参数
GeneratedCharacters	生成字符平均长度	最大生成字符长度	最小生成字符长度	75分位生成字符长度	90分位生成字符长度	90分位生成字符长度	99分位生成字符长度	测试数据量，来源于输入参数
Tokenizer	tokenizer的平均时间，单位（ms）	最大tokenizer时间，单位（ms）	最小tokenizer时间，单位（ms）	75分位tokenizer处理时间，单位（ms）	90分位tokenizer处理时间，单位（ms）	90分位tokenizer处理时间，单位（ms）	99分位tokenizer时间，单位（ms）	测试数据量，来源于输入参数

参数	说明（average）	说明（max）	说明（min）	说明（P75）	说明（P90）	说明（SLO_P90）	说明（P99）	说明（N）
Detokenizer	detokenizer的平均时间，单位（ms）	最大detokenizer时间，单位（ms）	最小detokenizer时间，单位（ms）	75分位detokenizer处理时间，单位（ms）	90分位detokenizer处理时间，单位（ms）	90分位detokenizer处理时间，单位（ms）	99分位detokenizer时间，单位（ms）	测试数据量，来源于输入参数
CharactersPerToken	每个token平均生成的字符数	-	-	-	-	-	-	测试数据量，来源于输入参数
PostProcessingTime	所有token平均后处理时间，单位（ms）	所有token最大后处理时间，单位（ms）	所有token最小后处理时间，单位（ms）	所有token 75分位后处理时间，单位（ms）	所有token 90分位后处理时间，单位（ms）	所有token 90分位后处理时间，单位（ms）	所有token 99分位后处理时间，单位（ms）	测试数据量，来源于输入参数
ForwardTime	所有token平均模型推理时间，单位（ms）	所有token最大模型推理时间，单位（ms）	所有token最小模型推理时间，单位（ms）	所有token 75分位模型推理时间，单位（ms）	所有token 90分位模型推理时间，单位（ms）	所有token 90分位模型推理时间，单位（ms）	所有token 99分位模型推理时间，单位（ms）	测试数据量，来源于输入参数
PrefillBatchsize	Prefill阶段batchsize平均值	最大Prefill阶段batchsize	最小Prefill阶段batchsize	75分位Prefill阶段batchsize	90分位Prefill阶段batchsize	90分位Prefill阶段batchsize	99分位Prefill阶段batchsize	测试数据量，来源于输入参数

参数	说明（average）	说明（max）	说明（min）	说明（P75）	说明（P90）	说明（SLO_P90）	说明（P99）	说明（N）
DecoderBatchsize	Decode阶段batchsize平均值	最大Decode阶段batchsize	最小Decode阶段batchsize	75分位Decode阶段batchsize	90分位Decode阶段batchsize	-	99分位Decode阶段batchsize	测试数据量，来源于输入参数
QueueWaitTime	队列等待时间平均值，单位（μs）	最大队列等待时间，单位（μs）	最小队列等待时间，单位（μs）	75分位队列等待时间，单位（μs）	90分位队列等待时间，单位（μs）	-	99分位队列等待时间，单位（μs）	测试数据量，来源于输入参数
注：**只有Client文本流式推理模式才能获取到PrefillBatchsize、DecoderBatchsize和QueueWaitTime参数。**								

表 8-6 端到端性能输出结果

参数	说明
CurrentTime	输出结果的当前时间点。
TimeElapsed	测试总耗时。单位（s）。
DataSource	测试数据集路径。
Failed	失败请求数据量（包含空和未返回数据的响应）。
Returned	返回请求总数据量（包含非空和空）。
Total	测试数据量。
Concurrency	测试并发数
ModelName	模型名称。
lpct	首token总时延/输入总token数。单位（ms）。
Throughput	整体测试过程的每秒请求数，吞吐量指标。单位（req/s）。
GenerateSpeed	整体测试并发下token的生成速度。单位（token/s）。
GenerateSpeedPerClient	整体的token生成速度/测试并发数。单位（token/s）。
accuracy	精度。

8.1.1.4 样例代码

8.1.1.4.1 Engine 推理模式

文本推理样例

说明

- 如果执行文本推理时报“cannot allocate memory in static TLS block”错误，处理方法请参见**9.10 使用第三方库transformers跑模型推理时，报错“cannot allocate memory in static TLS block”**章节。
- 执行样例前请使用以下命令配置环境变量。

```
source /usr/local/Ascend/ascend-toolkit/set_env.sh          # CANN
source /usr/local/Ascend/nnal/atb/set_env.sh                # ATB
source /usr/local/Ascend/llm_model/set_env.sh               # ATB Models
source /usr/local/Ascend/mindie/set_env.sh                  # MindIE
```

带后处理性能测试样例

```
# Engine模式 文本推理
SMPL_PARAM='{\"temperature\":0.5,\"top_k\":10,\"top_p\":0.9,\"typical_p\":0.9,\"seed
\":1234,\"repetition_penalty\":1,\"watermark\":true}'
benchmark \
--DatasetPath "/{数据集路径}/GSM8K/" \
--DatasetType "gsm8k" \
--ModelName llama2-7b \
--ModelPath "/{模型权重路径}/LLaMA3-8B/" \
--TestType engine \
--Tokenizer True \
--Concurrency 128 \
--MaxOutputLen 512 \
--DoSampling True \
--SamplingParams=$SMPL_PARAM
```

Engine文本推理模式输出结果如**图8-1**所示：

图 8-1 Engine 文本推理模式

```
2024-10-17 05:29:37,280|INFO|/usr/local/lib/python3.10/dist-packages/mindiebenchmark/common/output.py:display_metrics_as_table:89| The BenchMark test perfor
+--------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+
|             Metric |         average |             max |             min |             P75 |             P90 |         SLO_P90 |          P99 |    N |
+--------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+
|      FirstTokenTime |      929.2383 ms |    5396.5735 ms |     75.2986 ms |    1276.9199 ms |    1777.2201 ms |    1777.2201 ms |    3638.2515 ms | 3000 |
|         DecodeTime |       59.1548 ms |    5447.4268 ms |      0.2825 ms |      61.3225 ms |      68.1764 ms |           0 ms |      78.8115 ms | 3000 |
|     LastDecodeTime |       41.8001 ms |    2578.1655 ms |           0 ms |      54.8365 ms |      62.9192 ms |      62.9192 ms |      83.6755 ms | 3000 |
|      MaxDecodeTime |      621.606 ms |    5447.4268 ms |      0.2849 ms |     834.6292 ms |    1024.5859 ms |    1024.5859 ms |    3523.4852 ms | 3000 |
|       GenerateTime |   25496.3406 ms |   37075.6511 ms |     76.7969 ms |   32346.3979 ms |   34146.6119 ms |   34146.6119 ms |   35755.5617 ms | 3000 |
|        InputTokens |         256.604 |            2047 |              63 |           232.0 |           669.0 |           669.0 |          1506.0 | 3000 |
|    GeneratedTokens |         416.302 |             512 |               1 |           512.0 |           512.0 |           512.0 |           512.0 | 3000 |
| GeneratedTokenSpeed | 15.2606 token/s | 25.0746 token/s | 0.2748 token/s | 16.9581 token/s | 18.2633 token/s | 18.2633 token/s | 22.7495 token/s | 3000 |
| GeneratedCharacters |        935.5593 |            3217 |              15 |         1059.75 |          2310.0 |          2310.0 |          2797.0 | 3000 |
|          Tokenizer |       1.1733 ms |      15.7864 ms |      0.1879 ms |      1.2116 ms |      2.8121 ms |      2.8121 ms |      5.2326 ms | 3000 |
|        Detokenizer |           0 ms |           0 ms |           0 ms |           0 ms |           0 ms |           0 ms |           0 ms | 3000 |
|  CharactersPerToken |          2.2473 |               - |               - |               - |               - |               - |               - | 3000 |
+--------------------+-----------------+-----------------+-----------------+-----------------+-----------------+-----------------+------+
2024-10-17 05:29:37,290|INFO|/usr/local/lib/python3.10/dist-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:104| The BenchMark test
+---------------------+-----------------------------------------------+
|       Common Metric |                                         Value |
+---------------------+-----------------------------------------------+
|         CurrentTime |                           2024-10-17 05:29:37 |
|         TimeElapsed |                                    396.1505 s |
|          DataSource | /home/dataset/service_perf_test/short_input_64_2k |
|              Failed |                                      0( 0.0% ) |
|            Returned |                                3000( 100.0% ) |
|               Total |                                3000[ 100.0% ] |
|         Concurrency |                                        1000 |
|           ModelName |                                    LLaMA3-8B |
|                lpct |                                    3.6213 ms |
|          Throughput |                                  7.5729 req/s |
|       GenerateSpeed |                              3152.605 token/s |
| GenerateSpeedPerClient |                                3.1526 token/s |
|            accuracy |                                           - |
+---------------------+-----------------------------------------------+
```

不带后处理性能测试样例

```
# Engine模式 文本推理
benchmark \
```

```
--DatasetPath "/{数据集路径}/CEval" \
--DatasetType "ceval" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型权重路径}/baichuan2-13b" \
--TestType engine \
--Concurrency 50 \
--MaxOutputLen 512 \
--Tokenizer True
```

精度测试样例

须知

- 使用CEval或MMLU数据集计算精度时，MaxOutputLen的值不应设置太大，比如设置为20即可。
- MindIE Server的config.json配置文件中maxSeqLen参数的值一般根据数据序列长度设置。
  - 对于CEval数据集，maxSeqLen的参考值为3072；
  - 对于MMLU数据集，maxSeqLen的参考值为3600，该数据集中有约为1.4w条数据，推理耗时会比较长。

样例如下所示：

```
benchmark \
--DatasetPath "/{数据集路径}/CEval" \
--DatasetType "ceval" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型权重路径}/baichuan2-13b" \
--TestType engine \
--Concurrency 1 \
--MaxOutputLen 20 \
--Tokenizer True \
--TestAccuracy True
```

tokenids 推理样例

说明

- 如果执行文本推理时报“cannot allocate memory in static TLS block”错误，处理方法请参见**9.10 使用第三方库transformers跑模型推理时，报错“cannot allocate memory in static TLS block”**章节。
- 执行样例前请使用以下命令配置环境变量。

```
source /usr/local/Ascend/ascend-toolkit/set_env.sh          # CANN
source /usr/local/Ascend/nnal/atb/set_env.sh                # ATB
source /usr/local/Ascend/llm_model/set_env.sh               # ATB Models
source /usr/local/Ascend/mindie/set_env.sh                  # MindIE
```

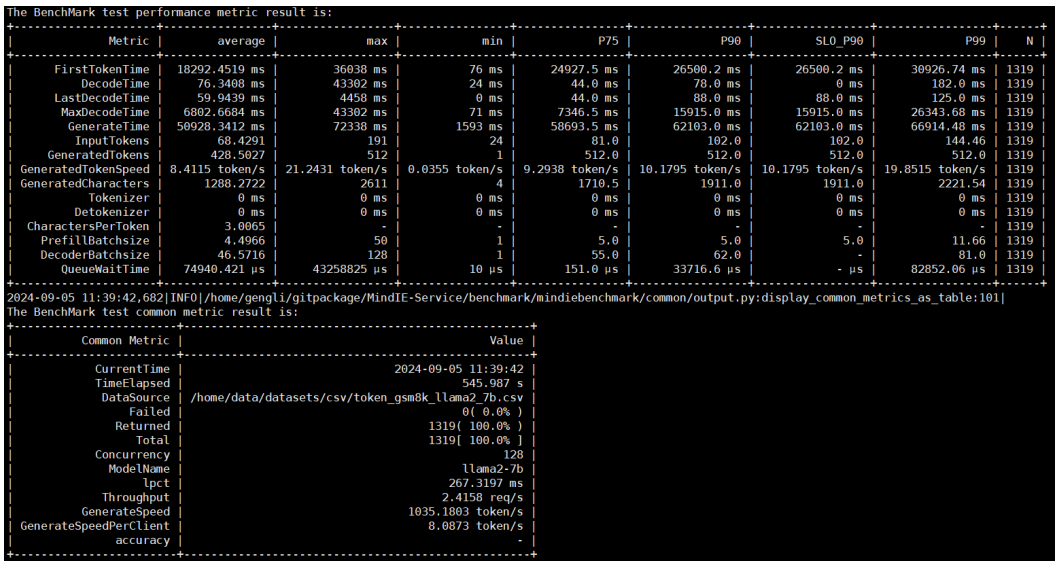
带后处理性能测试样例

```
# Engine模式 tokenids推理
SMPL_PARAM='{\"temperature\":0.5,\"top_k\":10,\"top_p\":0.9,\"typical_p\":0.9,\"seed
\":1234,\"repetition_penalty\":1,\"watermark\":true}'
benchmark \
--DatasetPath "/{数据集路径}/gsm8k/xx.csv" \
--DatasetType "gsm8k" \
--ModelName llama2-7b \
--ModelPath "/{模型权重路径}/LLaMA3-8B/" \
--TestType engine \
```



```
--Tokenizer False \
--Concurrency 128 \
--MaxOutputLen 512 \
--DoSampling True \
--SamplingParams=$SMPL_PARAM
```

图 8-2 Engine 模式 tokenids 推理



MindIE Benchmark的Engine推理模式输出参数说明请参见**表8-5**和**表8-6**。

不带后处理性能测试样例

```
# Engine模式 tokenids推理
benchmark \
--DatasetPath "/{数据集路径}/gsm8k/xx.csv" \
--DatasetType "gsm8k" \
--ModelName "baichuan2_13b" \
--ModelPath "/{模型权重路径}/baichuan2-13b" \
--TestType engine \
--Tokenizer False \
--Concurrency 50
```

8.1.1.4.2 Client 推理模式

总体说明

Client推理模式在运行前，需要启动MindIE Server的服务。

使用以下命令启动MindIE Server服务。

```
cd $MIES_INSTALL_PATH/
./bin/mindieservice_daemon
```

显示如下则说明启动成功。

```
Daemon start success!
```

Client推理模式运行时，会调用MindIE Client相应的接口向MindIE Server Endpoint发送推理请求并进行打点统计。

支持开启或关闭--DoSampling两种模式，开启DoSampling时，需要配合--SamplingParams传入采样参数，否则使用如下默认采样参数：


```
DEFAULT_SMPL_PARAM = {
    "temperature": 0.5,
    "top_k": 10,
    "top_p": 0.9,
    "typical_p": 0.9,
    "seed": 1,
    "repetition_penalty": 1,
    "watermark": True,
    "truncate": 5
}
```

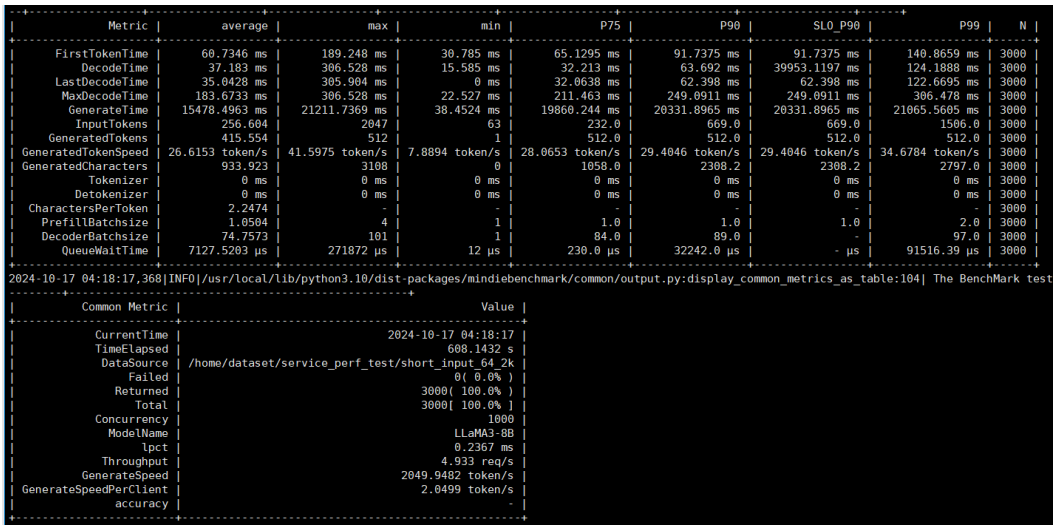
client模式下，并发数为影响性能（吞吐量及平均Decode时间）的关键指标，根据用户的不同需求，初步选取标准如下：

- 提高系统吞吐量（QPS）：适当提高并发数可以提高。
- 提高系统的平均非首token时间：随着并发数增大，非首token的Decode时间会增加，卡非首token时间时可以设定一个较小的并发数，baichuan2-13b并发数参考设定为**64**或更小的数值（卡50ms平均Decode时间），llama-65b及类似大小模型的参考并发数为**16/32**。

此外，Endpoint启动前还需要根据模型权重计算合适的npuMemsize以获得最佳性能，具体配置方法请参见**5.1 性能调优流程**。

Client推理模式输出结果如**图8-3**所示：

图 8-3 Client 文本流式推理模式



Metric	average	max	min	P75	P90	SLO_P90	P99	N
FirstTokenTime	60.7346 ms	189.248 ms	30.785 ms	65.1295 ms	91.7375 ms	91.7375 ms	140.8659 ms	3000
DecodeTime	37.183 ms	306.528 ms	15.585 ms	32.213 ms	63.692 ms	39953.1197 ms	124.1888 ms	3000
LastDecodeTime	35.0428 ms	305.904 ms	0 ms	32.0638 ms	62.398 ms	62.398 ms	122.6695 ms	3000
MaxDecodeTime	183.6733 ms	306.528 ms	22.527 ms	211.463 ms	249.0911 ms	249.0911 ms	306.478 ms	3000
GenerateTime	15478.4963 ms	21211.7369 ms	38.4524 ms	19860.244 ms	20331.8965 ms	20331.8965 ms	21065.5605 ms	3000
InputTokens	256.604	2047	63	232.0	669.0	669.0	1506.0	3000
GeneratedTokens	415.554	512	1	512.0	512.0	512.0	512.0	3000
GeneratedTokenSpeed	26.6153 token/s	41.5975 token/s	7.8894 token/s	28.0653 token/s	29.4046 token/s	29.4046 token/s	34.6784 token/s	3000
GeneratedCharacters	933.923	3108	0	1058.0	2308.2	2308.2	2797.0	3000
Tokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	3000
Detokenizer	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	0 ms	3000
CharactersPerToken	2.2474	-	-	-	-	-	-	3000
PrefillBatchsize	1.0504	4	1	1.0	1.0	1.0	2.0	3000
DecoderBatchsize	74.7573	101	1	84.0	89.0	-	97.0	3000
QueueWaitTime	7127.5203 μs	271872 μs	12 μs	230.0 μs	32242.0 μs	- μs	91516.39 μs	3000

2024-10-17 04:18:17,368|INFO|/usr/local/lib/python3.10/dist-packages/mindiebenchmark/common/output.py:display_common_metrics_as_table:104| The BenchMark test

Common Metric	Value
CurrentTime	2024-10-17 04:18:17
TimeElapsed	608.1432 s
DataSource	/home/dataset/service_perf_test/short_input_64_2k
Failed	0(0.0%)
Returned	3000(100.0%)
Total	3000[100.0%]
Concurrency	1000
ModelName	LLaMA3-8B
lpct	0.2367 ms
Throughput	4.933 req/s
GenerateSpeed	2049.9482 token/s
GenerateSpeedPerClient	2.0499 token/s
accuracy	-

MindIE Benchmark的Client推理模式输出参数说明请参见**表8-5**和**表8-6**。

文本推理样例

说明

执行样例前请使用以下命令配置环境变量。

```
source /usr/local/Ascend/ascend-toolkit/set_env.sh          # CANN
source /usr/local/Ascend/nnal/atb/set_env.sh                # ATB
source /usr/local/Ascend/llm_model/set_env.sh                  # ATB Models
source /usr/local/Ascend/mindie/set_env.sh                  # MindIE
```

后处理性能测试样例

```
SMPL_PARAM="{\"temperature\":0.5,\"top_k\":10,\"top_p\":0.9,\"typical_p\":0.9,\"seed
\":1234,\"repetition_penalty\":1,\"watermark\":true,\"truncate\":10}"
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--TaskKind stream \
--Tokenizer True \
--MaxOutputLen 512 \
--DoSampling True \
--SamplingParams $SMPL_PARAM
```

说明

Client模式下仅当TaskKind=stream_token时，--Tokenizer为False。

不带后处理性能测试样例

通过--TaskKind参数区分不同Client推理模式。

- 文本非流式推理：

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--TaskKind text \
--Tokenizer True \
--MaxOutputLen 512
```

- 文本流式推理：

```
benchmark \
--DatasetPath "/{数据集路径}/GSM8K" \
--DatasetType "gsm8k" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--TaskKind stream \
--Tokenizer True \
--MaxOutputLen 512
```

使用合成数据（synthetic）进行性能测试样例

在使用合成数据（synthetic）进行性能测试时，不需要指定 --DatasetPath，需要指定--DatasetType 为"synthetic"。另外需要指定配置文件--SyntheticConfigPath，文件配置内容见**7.3 配置MindIE Benchmark**。其他参数和其他性能测试保持一致。

```
benchmark \
--DatasetType "synthetic" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType vllm_client \
```



```
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--MaxOutputLen 20 \
--TaskKind stream \
--Tokenizer True \
--SyntheticConfigPath /{配置文件路径}/synthetic_config.json
```

精度测试样例

须知

- 需要开启确定性计算环境变量。
  ```
  export LCCL_DETERMINISTIC=1
  export HCCL_DETERMINISTIC=true
  export ATB_MATMUL_SHUFFLE_K_ENABLE=0
  ```
- 并发数需设置为1，确保模型推理时是1batch输入，这样才可以和纯模型比对精度。
- 使用CEval比对精度时，MaxOutputLen应该设为20，MindIE Server的config.json文件中maxSeqLen需要设置为3072。
- 使用MMLU比对精度时，MaxOutputLen应该设为20，MindIE Server的config.json文件中maxSeqLen需要设置为3600，该数据集中有约为1.4w条数据，推理耗时会比较长。

```
benchmark \
--DatasetPath "/{数据集路径}/CEval" \
--DatasetType "ceval" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 1 \
--MaxOutputLen 20 \
--TaskKind stream \
--Tokenizer True \
--TestAccuracy True
```

token 推理样例

说明

执行样例前请使用以下命令配置环境变量。

```
source /usr/local/Ascend/ascend-toolkit/set_env.sh          # CANN
source /usr/local/Ascend/nnal/atb/set_env.sh                # ATB
source /usr/local/Ascend/llm_model/set_env.sh                  # ATB Models
source /usr/local/Ascend/mindie/set_env.sh                  # MindIE
```

后处理性能测试样例

```
SMPL_PARAM="{\"temperature\":0.5,\"top_k\":10,\"top_p\":0.9,\"typical_p\":0.9,\"seed
\":1234,\"repetition_penalty\":1,\"watermark\":true,\"truncate\":10}"
benchmark
--DatasetPath "/{数据集路径}/token_gsm8k_model.csv" \
--DatasetType "gsm8k" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
```



```
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--TaskKind stream_token \
--Tokenizer False \
--MaxOutputLen 512 \
--DoSampling True \
--SamplingParams $SMPL_PARAM
```

不带后处理性能测试样例

通过--TaskKind参数设定stream_token指定模式为token流式推理。

```
benchmark \
--DatasetPath "/{数据集路径}/token_gsm8k_model.csv" \
--DatasetType "gsm8k" \
--ModelName llama_7b \
--ModelPath "/{模型权重路径}/llama_7b" \
--TestType client \
--Http https://{ipAddress}:{port} \
--ManagementHttp https://{managementIpAddress}:{managementPort} \
--Concurrency 128 \
--TaskKind stream_token \
--Tokenizer False \
--MaxOutputLen 512
```

8.1.2 CertTools

8.1.2.1 config_mindie_server_tls_cert.py

脚本功能

EndPoint开启HTTPS时，使用该脚本对证书进行管理，主要功能如下：

- 对CA证书进行导入/更新、删除。
- 对服务证书，私钥进行导入/更新、删除。
- 对吊销列表进行导入/更新。
- 生成CA证书。
- 查询已导入的CA证书，服务证书，吊销列表详情信息。

参数说明

参数类型	参数名	参数说明
位置参数	<project_path>	软件包安装路径。
位置参数	<sub_command>	子命令类型，包括[gen_cert, import_ca, delete_ca, import_cert, delete_cert, import_crl, query]，分别对应生成证书，导入CA证书，删除CA证书，导入服务证书和私钥，删除服务证书和私钥，导入更新吊销列表，查询已导入证书信息。各个子命令的使用方法请参见**使用指南**。

参数类型	参数名	参数说明
选项参数	--business	需要证书管理业务类型，默认为HTTPS证书，可以设置为[management, grpc]，分别对应HTTPS management证书和GRPC证书。
选项参数	--ip	证书的IP，如果不设置证书，则使用None代替。

证书校验规则

- 验签证书CA校验规则如下：

```
#1. 是否属于X509v3证书，不是则直接退出。
#2. 证书是否过期，过期则直接退出。
#3. 签名算法是否为sha256WithRSAEncryption和sha512WithRSAEncryption，不是则有告警提示。
#4. 是否为签名证书（存在CA Flag Digital Signature Certificate cRLSign字段），不是则有告警提示。
#5. RSA密钥算法是否长度 >= 3072，不是则直接退出。
```

- 验证服务证书Cert校验规则如下：

```
#1. 格式是否为X509v3，不是则直接退出。
#2. 签名算法是否是sha256WithRSAEncryption和sha512WithRSAEncryption，不是则有告警提示。
#3. RSA密钥算法长度是否 >= 3072，不是则直接退出。
#4. 是否包含Certificate Signature和cRLSign字段，不是则有告警提示。
#5. 证书是否过期。
#6. 证书和私钥是否匹配。
```

- 验证吊销列表校验规则如下：

```
#1. crl列表是否为空，为空则直接退出。
#2. 吊销列表是否过期，过期则直接退出。
```

前置配置

```
# 缺失的包
pip3 install pyOpenSSL

# 开启日志
export MIES_CERTS_LOG_TO_FILE=1
export MIES_CERTS_LOG_TO_STDOUT=1

# 日志配置
export MIES_CERTS_LOG_LEVEL=INFO
export MIES_CERTS_LOG_PATH=/workspace/log/certs.log   # 文件当前需要通过umask为0077指定为0600
```

使用指南

- 生成证书。

```
cd /{MindIE安装目录}/latest/mindie-service/ #进入mindie-service安装目录
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 gen_cert CA证书配置文件路径 --ip 证书ip
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} gen_cert /home/Ascend-mindie-service_{version}_linux-{arch}/conf/gen_cert.json --ip=1.1.1.1,2.2.2.2
```

- 对CA证书进行导入/更新。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 import_ca CA文件列表（不超过5个）
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} import_ca /home/ca.pem /home/ca2.pem
```

- 对已导入的CA文件进行删除。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 delete_ca CA文件名称列表（不超过5个）
```



```
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} delete_ca ca.pem ca2.pem
```

- 对服务证书，私钥进行导入/更新（会在弹窗中进行私钥口令输入）。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 import_cert 服务证书路径 服务私钥路径
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} import_cert /home/server.pem /home/server.key.pem
```

- 对服务证书，私钥进行删除。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 delete_cert
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} delete_cert --cert_file=server.pem --key_file=server.key.pem
```

参数解释：

--cert_file：客户端证书文件路径；

--key_file：客户端私钥文件路径。

- 对吊销列表进行导入/更新，crl配套CA文件列表中的第一个CA文件。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 import_crl 吊销列表文件路径
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} import_crl /home/server_crl.pem
```

- 生成CA证书。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 gen_cert 配置文件的路径 --ip=IP地址
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} gen_cert /home/config.json --ip=1.1.1.1
```

- 查询导入的CA证书，服务证书，吊销列表文件详情。

```
python3 scripts/config_mindie_server_tls_cert.py 软件包安装目录 query
# 样例
python3 scripts/config_mindie_server_tls_cert.py /home/Ascend-mindie-service_{version}_linux-{arch} query --cert_file=server.pem --crl_file=server_crl.pem
```

参数解释：

--crl_file：客户端吊销列表路径。

说明

- HTTPS使用三面隔离时，https的业务面和管理面不建议使用同一套安全证书，使用同一套安全证书会导致存在较高的网络安全风险。
- HTTPS和GRPC不建议使用同一套安全证书，使用同一套安全证书会导致存在较高的网络安全风险。
- 导入证书时，对于用户导入的CA文件证书工具要求的权限为600，服务证书文件证书工具要求的权限为600，私钥文件证书工具要求的权限要求为400，吊销列表证书工具要求的权限为600。

8.1.2.2 seceasy_encrypt

8.1.2.2.1 encrypt

命令功能

EndPoint开启https时，对服务端密钥口令加密。交互输入明文字符串，输出密文字符串。

命令格式

```
./bin/seceasy_encrypt --encrypt domainId domainCount
```

说明

该命名必须在*{MindIE安装目录}*/latest/mindie-service/bin目录下执行。

参数说明

参数	说明
domainId	域ID，取值不能大于domainCount，当前业务要求取值为1。
domainCount	域的数量，取值范围为[2,1023]，当前业务要求取值为2。

8.1.2.2.2 activeKey

命令功能

更新主密钥。

命令格式

```
./bin/seceasy_encrypt --activeKey domainId domainCount
```

说明

该命名必须在*{MindIE安装目录}*/latest/mindie-service/bin目录下执行。

参数说明

参数	说明
domainId	域ID，取值不能大于domainCount，当前业务要求取值为1。
domainCount	域的数量，取值范围为[2,1023]，当前业务要求取值为2。

8.1.2.2.3 secureEraseAllKeystore

命令功能

删除密钥。

须知

- 执行该指令后会删除密钥，会导致功能异常。
- 仅在卸载MindIE Server时需执行该命令。

命令格式

```
./bin/seceasy_encrypt --secureEraseAllKeystore domainCount
```

说明

该命名必须在*{MindIE安装目录}*/latest/mindie-service/bin目录下执行。

参数说明

参数	说明
domainCount	域的数量，取值范围为[2,1023]，当前业务要求取值为2。

8.1.2.3 gen_cert

命令功能

根据配置文件，离线生成证书、密钥和口令文件。

命令格式

```
./bin/gen_cert configFile ip
```

参数说明

参数	说明
configFile	gen_cert.json配置文件的路径。
ip	证书的IP，如果不设置证书，则使用None代替。

gen_cert.json文件如下所示：

```
{
    "ca_cert": "/path/to/ca.pem",                       # CA证书的路径
    "ca_key": "/path/to/ca.key.pem",                    # CA证书的密钥文件路径
    "ca_key_pwd": "/path/to/ca_passwd.txt",             # CA证书的密钥口令文件路径
    "cert_config": "/path/to/cert_config.json",         # 证书配置文件路径
    "output_path": "/path/to/output_dir",               # 存放生成文件的目录
    "kmc_ksf_master": "/path/to/tools/pmt/master/ksfa",   # 加解密使用的kmc master文件
    "kmc_ksf_standby": "/path/to/tools/pmt/standby/ksfb"  # 加解密使用的kmc standby文件
}
```

cert_config.json文件如下所示：


```
{
    "subject": "subject_name",      # 证书主体名称
    "expired_time": 365,            # 证书有效时间(天)
    "serial_number": 123,           # 证书序列号
    "req_distinguished_name": {     # 证书所属组织
        "C": "CN",                  # 国家
        "ST": "Sichuan",            # 省份
        "L": "Chengdu",             # 城市
        "O": "Huawei",              # 公司
        "OU": "Ascend",             # 部门
        "CN": "MindIE"              # 组织
    },
    "alt_names": {                  # 别名
        "IP": [],                   # ip地址
        "DNS": []                   # 域名
    }
}
```

8.2 MindIE Client

8.2.1 功能介绍

MindIE Client是MindIE Service的模块之一，在启动MindIE Server Endpoint服务后，MindIE Client可以使用HTTP/HTTPS协议对它发送请求。MindIE Client提供了多种功能的接口，包括模型推理、请求管理和服务状态查询，用户调用接口即可实现与MindIE Server通信。

应用场景

- MindIE Client支持以下模型推理接口：
  - 同步推理（token_id to token_id）：

    同步推理模式下，客户端发送一个请求给服务端（如：MindIE Server），然后等待服务端完成推理处理并返回结果后，才继续执行下一步操作，在此期间，客户端处于等待状态，其对应的接口请参见 **def infer(self, model_name, inputs, model_version, outputs, request_id, parameters)**。
  - 异步推理（token_id to token_id）：**异步推理（token_id to token_id）**：

    异步推理模式下，客户端发送请求给服务端后，不需要等待响应，可以继续执行其他任务。服务端完成推理后，通过回调、通知或者其他方式告知客户端推理结果，其对应的接口请参见 **def async_infer(self, model_name, inputs, model_version, outputs, request_id, parameters)**。
  - 全量文本推理（text to text）：

    全量文本推理模式下，输入的是prompt（文本），推理结果输出的内容格式也是文本，一次性返回全部响应内容，其对应的接口请参见 **def generate(self, model_name, prompt, model_version, request_id, parameters)**。
  - 流式文本推理（text to text）：

    流式文本推理模式下，输入的是prompt（文本），推理结果按流式返回，每次返回的是单个字符，其对应的接口请参见 **def generate_stream(self, model_name, prompt, model_version, request_id, parameters)**。
- MindIE Client支持以下请求管理接口：
  - 提前终止推理请求，其对应的接口请参见 **def cancel(self, model_name, request_id, model_version)**。

  - 统计slot数量，其对应的接口请参见 **def get_slot_count(self, model_name, model_version)**。
- MindIE Client支持以下服务状态查询接口：
  - 查询Server和Model的状态和元数据，其对应的接口请参见 **def get_server_metadata(self)**和 **def get_model_metadata(self, model_name, model_version)**。
  - 查询Model配置，其对应的接口请参见 **def get_model_config(self, model_name, model_version)**。

8.2.2 API 参考（Python）

8.2.2.1 类参考

8.2.2.1.1 class AsyncRequest

类说明

异步请求类，保存了异步推理的结果。

def __init__(self, greenlet, verbose)

函数功能

类构造函数。

函数原型

```
def __init__(self, greenlet, verbose)
```

参数说明

参数名	参数类型	输入/输出	说明
greenlet	greenlet对象	输入	当前请求对应的协程，用于获取结果。
verbose	bool	输入	是否打印详细日志，默认为False。

def get_result(self, timeout)

函数功能

获取异步推理的结果。

函数原型

```
def get_result(self, timeout)
```


约束说明

1. timeout大于0或者为None;

2. 当timeout不满足约束1或者获取结果超时会抛异常。

参数说明

参数名	参数类型	输入/输出	说明
timeout	float	输入	表示当前协程获取结果的时间限制，默认为None。

返回值

Result对象，表示推理结果。

8.2.2.1.2 class MindIEHTTPClient

类说明

MindIEHTTPClient类封装了HTTP请求接口，具体接口如下所示。

- 模型推理接口：infer、async_infer、generate、generate_stream、generate_stream_with_details、generate_stream_self。
- 请求管理接口：cancel和get_slot_count。
- 服务状态查询接口：is_server_live、is_server_ready、is_model_ready、get_server_metadata、get_model_metadata、get_model_config。

成员变量

成员名称	描述
valid_url	URL对象，通过str(valid_url)可以得到string类型的url，如https://127.0.0.1:1025。
manage_url	string类型的管理url，如https://127.0.0.2:1026。

def __init__(self, url, greenlet_size, enable_ssl, ssl_options, network_timeout, connection_timeout, concurrency, verbose, management_url)

函数功能

HttpClient对象初始化。

函数原型

```
def __init__(self, url, greenlet_size, enable_ssl, ssl_options, network_timeout, connection_timeout, concurrency, verbose, management_url)
```


约束说明

- url的scheme只支持HTTPS和HTTP。
- network_timeout、connection_timeout的取值为大于0。
- management_url的scheme必须和url保持一致。
- 当开启HTTPS认证，用户需通过ssl_options提供ca_certs、certfile、keyfile和crlfile，这四个证书所在目录权限需为700，且证书权限不高于600。

参数说明

参数名	参数类型	输入/输出	说明
url	str	输入	格式为http://IP:Port 或 https://IP:Port。
greenlet_size	int	输入	协程数量，取值范围：(0, 1e9]，默认为None。
enable_ssl	bool	输入	是否开启ssl认证，默认为True。
ssl_options	dict	输入	认证鉴权设置，默认为None。当开启HTTPS认证，用户需提供ca_certs、certfile、keyfile和crlfile（可选），详情请参见**表8-7**。该参数的使用方法详情请参见**8.2.3.1 创建客户端**。
network_timeout	float	输入	网络时间限制，默认为600.0秒。
connection_timeout	float	输入	连接时间限制，默认为600.0秒。
concurrency	int	输入	并发连接数，取值范围：(0, 1000]，默认为1。
verbose	bool	输入	是否输出详细日志，默认为False。
management_url	str	输入	管理端口url，用于is_server_live、is_server_ready、is_model_ready与get_slot_count接口，默认为"https://127.0.0.2:1026"。

表 8-7 ssl_options 参数对

参数名	参数类型	输入/输出	说明
ca_certs	str	输入	CA证书的路径。
certfile	str	输入	客户端证书路径。

参数名	参数类型	输入/输出	说明
keyfile	str	输入	客户端证书密钥路径。
crlfile	str	输入	吊销列表路径。

说明

当参数不满足约束说明时，创建Client会失败并抛异常。

返回值

无

def __del__(self)

函数功能

MindIEHTTPClient的析构函数。

函数原型

```
def __del__(self)
```

返回值

无

def close(self)

函数功能

关闭HTTPClient，在MindIEHTTPClient析构函数中调用。

函数原型

```
def close(self)
```

返回值

无

def is_server_live(self)

函数功能

判断服务状态是否正常。

函数原型

```
def is_server_live(self)
```


返回值

bool对象，表示服务状态是否正常，如果返回True，则表示正常。

def is_server_ready(self)

函数功能

判断server的准备状态。

函数原型

```
def is_server_ready(self)
```

返回值

bool对象，表示server是否准备好。如果返回True，则表示server已经准备好。

def is_model_ready(self, model_name, model_version)

函数功能

判断model的准备状态。

函数原型

```
def is_model_ready(self, model_name, model_version)
```

约束说明

model_name支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数说明

参数名	参数类型	输入/输出	说明
model_name	String	输入	模型名称。
model_version	String	输入	模型版本，默认为""。该字段暂不支持，不传递。

返回值

bool对象，表示model是否准备好。如果返回True，则表示model已经准备好。

def get_server_metadata(self)

函数功能

查询Server元数据。

函数原型

```
def get_server_metadata(self)
```

返回值

返回json字典形式的Server元数据。

说明

- 当response返回状态码不等于200时会抛异常。
- 返回值详情请参见**4.2.5.2 查询服务元数据**章节。

def get_model_metadata(self, model_name, model_version)

函数功能

查询Model的元数据。

函数原型

```
def get_model_metadata(self, model_name, model_version)
```

约束说明

model_name支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数说明

参数名	参数类型	输入/输出	说明
model_name	String	输入	模型名称。
model_version	String	输入	模型版本，默认为""。该字段暂不支持，不传递。

说明

当参数名不满足字符组成限制或者response返回状态码不等于200时会抛异常。

返回值

返回json字典形式的Model元数据。

说明

返回值详情请参见**4.2.5.3 查询模型元数据**章节。

def get_model_config(self, model_name, model_version)

函数功能

查询Model配置。

函数原型

```
def get_model_config(self, model_name, model_version)
```

约束说明

model_name支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数说明

参数名	参数类型	输入/输出	说明
model_name	String	输入	模型名称。
model_version	String	输入	模型版本，默认为""。该字段暂不支持，不传递。

说明

当参数不满足字符组成限制或者response返回状态码不等于200时会抛异常。

返回值

返回json字典形式的model配置。

说明

返回值详情请参见**4.2.5.4 查询模型配置数据**。

def infer(self, model_name, inputs, model_version, outputs, request_id, parameters)

函数功能

实现同步推理。

函数原型

```
def infer(self, model_name, inputs, model_version, outputs, request_id, parameters)
```


参数说明

参数名	参数类型	输入/输出	说明
model_name	str	输入	模型名称。 模型名称支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。
inputs	list	输入	模型输入。 目前只支持输入1个Input实例，该实例中的参数取值如下所示： ● name：支持自定义，默认为"input0"。 ● shape：取值为[1,n]（n <= 16000）。 ● dataType：只支持UINT32。
model_version	str	输入	模型版本，默认为""。该字段暂不支持，不传递。
outputs	list	输入	指定需要返回的模型输出，默认值为None。如果为None则全部返回。目前只支持指定一个输出，name支持自定义，默认为“output0”。
request_id	str	输入	请求ID，默认为""；为空时会随机初始化一个值。如果为非空，长度需满足[1,1000]。
parameters	dict	输入	额外的请求参数，包括seed、temperature、top_k、top_p、do_sample、repetition_penalty和max_new_tokens，默认None，详情请参见**表8-8**。

表 8-8 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。

参数名	参数类型	输入/输出	说明
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。

参数名	参数类型	输入/输出	说明
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见**ScheduleConfig参数说明**中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

Result对象表示同步推理结果。

def async_infer(self, model_name, inputs, model_version, outputs, request_id, parameters)

函数功能

实现异步推理。

函数原型

```
def async_infer(self, model_name, inputs, model_version, outputs, request_id, parameters)
```

参数说明

参数名	参数类型	输入/输出	说明
model_name	str	输入	模型名称。 模型名称支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数名	参数类型	输入/输出	说明
inputs	list	输入	模型输入。 目前只支持输入1个Input实例，该实例中的参数取值如下所示： • name：支持自定义，默认为"input0"。 • shape：取值为[1,n]（n <= 16000）。 • dataType：只支持UINT32。
model_version	str	输入	模型版本，默认为""。该字段暂不支持，不传递。
outputs	list	输入	指定需要返回的模型输出，默认值为None。如果为None则全部返回。目前只支持指定一个输出，name支持自定义，默认为“output0”。
request_id	str	输入	请求ID，默认为""；如果request_id为空，则会随机初始化一个值。如果为非空，长度需满足[1,1000]。
parameters	dict	输入	额外的请求参数。包括：seed、temperature、top_k、top_p、do_sample、repetition_penalty和max_new_tokens，默认None，详情请参见表8-9。

表 8-9 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数名	参数类型	输入/输出	说明
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见**ScheduleConfig参数说明**中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

AsyncRequest对象表示异步推理结果。

def generate(self, model_name, prompt, model_version, request_id, parameters)

函数功能

实现全量文本生成。

函数原型

```
def generate(self, model_name, prompt, model_version, request_id, parameters)
```

参数说明

参数名	参数类型	输入/输出	说明
model_name	str	输入	模型名称。 模型名称支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。
prompt	str	输入	模型输入字符串，长度取值范围为[1, 512000] 。
model_version	str	输入	模型版本，默认为""。该字段暂不支持，不传递。
request_id	str	输入	请求ID，默认为""；如果request_id为空，则会随机初始化一个值。如果为非空，长度需满足[1,1000]。
parameters	dict	输入	额外的请求参数，默认为None；包括seed、temperature、top_k、top_p、do_sample、repetition_penalty、typical_p、batch_size、details和max_new_tokens，详情请参见**表 8-10**。

表 8-10 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值范围为(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。

参数名	参数类型	输入/输出	说明
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。
typical_p	float	输入	解码输出概率分布指数，当前后处理不支持。 取值范围为(0.0,1.0]，默认值1.0。
batch_size	int	输入	推理请求batch_size。 取值大于0，默认值为1。

参数名	参数类型	输入/输出	说明
details	bool	输入	是否返回推理详细输出结果，默认值false。
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见ScheduleConfig参数说明中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

Result对象表示全量文本推理结果。

def generate_stream(self, model_name, prompt, model_version, request_id, parameters)

函数功能

实现流式文本生成。

函数原型

```
def generate_stream(self, model_name, prompt, model_version, request_id, parameters)
```

参数说明

参数名	参数类型	输入/输出	说明
model_name	str	输入	模型名称。 模型名称支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。
prompt	str	输入	模型输入字符串，长度取值范围为[1, 512000]。
model_version	str	输入	模型版本，默认为""。该字段暂不支持，不传递。

参数名	参数类型	输入/输出	说明
request_id	str	输入	请求ID，默认为""；如果request_id为空，则会随机初始化一个值。如果为非空，长度需满足[1,1000]。
parameters	dict	输入	额外的请求参数，默认为None；包括seed、temperature、top_k、top_p、do_sample、repetition_penalty、typical_p、batch_size、details和max_new_tokens，详情请参见**表8-11**。

表 8-11 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值范围为(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。

参数名	参数类型	输入/输出	说明
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。
typical_p	float	输入	解码输出概率分布指数，当前后处理不支持。 取值范围为(0.0,1.0]，默认值1.0。
batch_size	int	输入	推理请求batch_size。 取值大于0，默认值为1。
details	bool	输入	是否返回推理详细输出结果，默认值false。
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见ScheduleConfig参数说明中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

每次推理返回当前生成的文本。

def generate_stream_with_details(self, model_name, prompt, model_version, request_id, parameters)

函数功能

实现流式文本生成。

函数原型

```
def generate_stream_with_details(self, model_name, prompt, model_version, request_id, parameters)
```

参数说明

参数名	参数类型	输入/输出	说明
model_name	str	输入	模型名称。 模型名称支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。
prompt	str	输入	模型输入字符串，长度取值范围为[1, 512000]。
model_version	str	输入	模型版本，默认为""。该字段暂不支持，不传递。
request_id	str	输入	请求ID，默认为""；如果request_id为空，则会随机初始化一个值。如果为非空，长度需满足[1,1000]。
parameters	dict	输入	额外的请求参数，默认为None；包括seed、temperature、top_k、top_p、do_sample、repetition_penalty、typical_p、batch_size、details和max_new_tokens，详情请参见**表 8-11**。

表 8-12 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值范围为(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。

参数名	参数类型	输入/输出	说明
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。
typical_p	float	输入	解码输出概率分布指数，当前后处理不支持。 取值范围为(0.0,1.0]，默认值1.0。
batch_size	int	输入	推理请求batch_size。 取值大于0，默认值为1。

参数名	参数类型	输入/输出	说明
details	bool	输入	是否返回推理详细输出结果，默认值false。
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见ScheduleConfig参数说明中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

- details=True时，每次推理返回一个字典（当前生成文本,batchsize, 队列等待时间）。
- details=False时，每次推理返回一个字典（当前生成文本）。

def generate_stream_self(self, token, request_id, parameters)

函数功能

实现token流式推理生成。

函数原型

```
def generate_stream_self(self, token, request_id, parameters)
```

参数说明

参数名	参数类型	输入/输出	说明
token	np.ndarray	输入	token格式的输入。
request_id	String	输入	请求ID，默认为""；如果request_id为空，则会随机初始化一个值。如果为非空，长度需满足[1,1000]。

参数名	参数类型	输入/输出	说明
parameters	dict	输入	额外的请求参数，默认为None；包括seed、temperature、top_k、top_p、do_sample、repetition_penalty、typical_p、batch_size、details和max_new_tokens，详情请参见表8-11。

表 8-13 parameters 参数说明

参数名	参数类型	输入/输出	说明
seed	int64	输入	随机种子数。 取值范围为(0,18446744073709551615]，不传递该参数，系统会产生一个随机seed值。
temperature	float	输入	控制生成的随机性，较高的值会产生更多样化的输出。 取值范围大于0，默认值为1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	int32	输入	控制模型生成过程中考虑的词汇范围，只从概率最高的 k 个候选词中选择，0表示不做top_k。 取值范围[0, 2147483647]&&[0, vocabSize)，默认值为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	float	输入	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。 取值范围：(0.0,1.0]，默认值为1.0。

参数名	参数类型	输入/输出	说明
do_sample	bool	输入	是否做sampling，默认值为false。
repetition_penalty	float	输入	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。 取值范围大于0，默认值为1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2，同时视模型而定。
typical_p	float	输入	解码输出概率分布指数，当前后处理不支持。 取值范围为(0.0,1.0]，默认值1.0。
batch_size	int	输入	推理请求batch_size。 取值大于0，默认值为1。
details	bool	输入	是否返回推理详细输出结果，默认值false。
max_new_tokens	int	输入	允许推理生成的最大token个数。该字段受到配置文件maxIterTimes参数影响，推理token个数<=maxIterTimes。 取值范围(0, maxIterTimes]，默认值为20。 maxIterTimes是MindIE Server配置文件config.json中的参数，详情请参见**ScheduleConfig参数说明**中的maxIterTimes。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

每次推理返回当前生成的文本。

def cancel(self, model_name, request_id, model_version)

函数功能

提前中止请求。

函数原型

```
def cancel(self, model_name, request_id, model_version)
```

约束说明

model_name支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数说明

参数名	参数类型	输入/输出	说明
model_name	String	输入	模型名称。
request_id	String	输入	请求ID。
model_version	String	输入	模型版本，默认为""。该字段暂不支持，不传递。

说明

当参数不满足限制条件时会抛异常。

返回值

bool类型，True表示请求中止成功；False表示请求中止失败。

def get_slot_count(self, model_name, model_version)

函数功能

获取slot统计结果。

函数原型

```
def get_slot_count(self, model_name, model_version)
```

约束说明

model_name支持大小写字母、数字、点号、中横线和下划线中一种或任意几种字符组成，最大长度为1000。

参数说明

参数名	参数类型	输入/输出	说明
model_name	String	输入	模型名称。
model_version	String	输入	模型版本，默认为""。该字段暂不支持，不传递。

说明

当参数不满足限制条件或者response返回状态码不等于200时会抛异常。

返回值

Result对象表示slot统计结果。

8.2.2.1.3 class Input

类说明

Client输入类。

def __init__(self, name, shape, datatype)

函数功能

Input对象初始化。

函数原型

```
def __init__(self, name, shape, datatype)
```

参数说明

参数名	参数类型	输入/输出	说明
name	str	输入	输入名称，长度范围[1, 1000]，name输入为空时，默认设为input0。
shape	array	输入	输入形状。 只支持[1,n]，n为tokenid数量，且n<=16000。
datatype	str	输入	输入名称数据类型。 只支持UINT32。

返回值

无

def get_input_name(self)

函数功能

获取输入名称。

函数原型

```
def get_input_name(self)
```


返回值

返回输入名称。

def get_input_shape(self)

函数功能

获取输入形状。

函数原型

```
def get_input_shape(self)
```

返回值

返回输入形状。

def set_input_shape(self, shape)

函数功能

设置输入形状。

函数原型

```
def set_input_shape(self, shape)
```

参数说明

参数名	参数类型	输入/输出	说明
shape	array	输入	输入形状。 只支持[1,n]，n为tokenid数量，且n<=16000。

返回值

无

def get_input_data_type(self)

函数功能

获取输入数据类型。

函数原型

```
def get_input_data_type(self)
```

返回值

返回输入数据类型。

def initialize_data(self, input_tensor)

函数功能

初始化输入的tensor数据。

函数原型

```
def initialize_data(self, input_tensor)
```

参数说明

参数名	参数类型	输入/输出	说明
input_tensor	numpy array	输入	输入tensor。 input_tensor需要符合Input类要求的数据类型和shape格式。

说明

当参数不满足限制条件时会抛异常。

返回值

无

def get_input_tensor(self)

函数功能

获取输入数据tensor。

函数原型

```
def get_input_tensor(self)
```

返回值

返回输入数据tensor。

说明

当input_data为空（未调用initialize_data初始化input_data）时会抛异常。

8.2.2.1.4 class RequestedOutput

类说明

Client输出类，表示请求返回哪些输入。

def __init__(self, name)

函数功能

RequestedOutput对象初始化。

函数原型

```
def __init__(self, name)
```

参数说明

参数名	参数类型	输入/输出	说明
name	String	输入	输入名称，长度范围[1, 1000]，name输入为空时，默认取值为output0。

返回值

无

def get_output_name(self)

函数功能

获取需要输出的名称。

函数原型

```
def get_output_name(self)
```

返回值

返回需要输出的名称。

def get_output_tensor(self)

函数功能

获取需要输出的tensor。

函数原型

```
def get_output_tensor(self)
```

返回值

返回需要输出的tensor（输出name的字典）。

8.2.2.1.5 class Result

类说明

Result请求结果类。

def __init__(self, response, verbose)

函数功能

Result对象初始化。

函数原型

```
def __init__(self, response, verbose)
```

参数说明

参数名	参数类型	输入/输出	说明
response	JSON对象	输入	编码后的JSON字符串，JSON对象字符串的大小不能超过10M。
verbose	bool	输入	是否打印详细日志。

返回值

无

def retrieve_output_index_to_numpy(self, index)

函数功能

将指定索引的输出转化成numpy形式。

函数原型

```
def retrieve_output_index_to_numpy(self, index)
```

参数说明

参数名	参数类型	输入/输出	说明
index	Int	输入	表示输出索引。 index值大于或等于0且小于输出个数。

返回值

返回numpy格式的特定索引的输出。

说明

当返回值不包含输出字段“outputs”，或index值大于或等于输出个数时会抛异常。

def retrieve_output_name_to_numpy(self, name)

函数功能

将特定名称的输出转化成numpy形式。

函数原型

```
def retrieve_output_name_to_numpy(self, name)
```

参数说明

参数名	参数类型	输入/输出	说明
name	String	输入	表示输出的名称。

返回值

返回numpy格式的特定名称的输出。

说明

当返回值不包含输出字段“outputs”或“name”与输出名称不一致时会抛异常。

def get_response(self)

函数功能

获取推理结果。

函数原型

```
def get_response(self)
```

返回值

返回推理结果。

8.2.2.1.6 class MindIEClientException(Exception)

类说明

异常类，用于表示Client的异常信息。

def __init__(self, message)

函数功能

MindIEClientException对象初始化。

函数原型

```
def __init__(self, message)
```

参数说明

参数名	参数类型	输入/输出	说明
message	String	输入	异常信息，长度取值范围在[1, 512000]。

返回值

无

def get_message(self)

函数功能

获取异常的信息。

函数原型

```
def get_message(self)
```

返回值

获取异常的信息。

8.2.3 样例代码

8.2.3.1 创建客户端

首先可以创建create_client函数，将函数所在文件命名为utils.py，后续可持续使用该方法；建议url和management_url的默认值为MindIE Server配置的IP和端口。

```
import argparse
from mindieclient.python.httpclient import MindIEHTTPClient
def create_client(request_count=1):
    # get argument
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "-u",
        "--url",
        required=False,
        default="https://127.0.0.1:1025",
        help="MindIE-Server URL.",
    )
```



```
parser.add_argument(
    "-mu",
    "--management_url",
    required=False,
    default="https://127.0.0.2:1026",
    help="MindIE-Server management URL.",
)
parser.add_argument(
    "-v",
    "--verbose",
    action="store_true",
    required=False,
    default=True,
    help="Enable detailed information output.",
)
parser.add_argument(
    "-s",
    "--ssl",
    action="store_true",
    required=False,
    default=False,
    help="Enable encrypted link with https",
)
parser.add_argument(
    "-ca",
    "--ca_certs",
    required=False,
    default="ca.pem",
    help="Provide https ca certificate.",
)
parser.add_argument(
    "-key",
    "--key_file",
    required=False,
    default="client.key.pem",
    help="Provide https client certificate.",
)
parser.add_argument(
    "-cert",
    "--cert_file",
    required=False,
    default="client.pem",
    help="Provide https client keyfile.",
)
parser.add_argument(
    "-crl",
    "--crl_file",
    required=False,
    default="crl.pem",
    help="Provide https crlfile.",
)
args = parser.parse_args()
# create client
try:
    if args.ssl:
        ssl_options = {}
        if args.ca_certs is not None:
            ssl_options["ca_certs"] = args.ca_certs
        if args.key_file is not None:
            ssl_options["keyfile"] = args.key_file
        if args.cert_file is not None:
            ssl_options["certfile"] = args.cert_file
        if args.crl_file is not None:
            ssl_options["crlfile"] = args.crl_file
        mindie_client = MindIEHTTPClient(
            url=args.url,
            verbose=args.verbose,
            enable_ssl=True,
            ssl_options=ssl_options,
```



```
            concurrency=request_count,
            management_url=args.management_url
        )
    else:
        mindie_client = MindIEHTTPClient(
            url=args.url, verbose=args.verbose,
            enable_ssl=False,concurrency=request_count,
            management_url=args.management_url
        )
except Exception as e:
    raise e
return mindie_client
```

8.2.3.2 同步推理

如创建样例代码test_infer.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_infer.py命令运行该样例。

```
import sys
import numpy as np
from mindieclient.python.httpclient import Input, RequestedOutput
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    try:
        mindie_client = create_client()
    except Exception as e:
        print("Client Creation falied!")
        sys.exit(1)
    # create input and requested output
    inputs = []
    outputs = []
    input_data = np.arange(start=0, stop=16, dtype=np.uint32)
    input_data = np.expand_dims(input_data, axis=0)
    inputs.append(Input("INPUT0", [1, 16], "UINT32"))
    inputs[0].initialize_data(input_data)
    outputs.append(RequestedOutput("OUTPUT0"))
    # apply model inference
    model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
    results = mindie_client.infer(
        model_name,
        inputs,
        outputs=outputs,
    )
    print(results.get_response())
    output_data = results.retrieve_output_name_to_numpy("OUTPUT0")
    print("input_data: %s" % np.array2string(input_data))
    print("output_data: %s" % np.array2string(output_data))
```

8.2.3.3 异步推理

如创建样例代码test_async_infer.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_async_infer.py命令运行该样例。

```
import sys
import numpy as np
from mindieclient.python.httpclient import Input, RequestedOutput
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    request_count = 2
    try:
```



```
    mindie_client = create_client(request_count)
except Exception as e:
    print("Client Creation falied!")
    sys.exit(1)
# create input and requested output
inputs = []
outputs = []
input_data = np.arange(start=0, stop=16, dtype=np.uint32)
input_data = np.expand_dims(input_data, axis=0)
inputs.append(Input("INPUT0", [1, 16], "UINT32"))
inputs[0].initialize_data(input_data)
outputs.append(RequestedOutput("OUTPUT0"))
# apply async inference
model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
async_requests = []
for _ in range(request_count):
    async_requests.append(
        mindie_client.async_infer(
            model_name,
            inputs,
            outputs=outputs,
        )
    )
# get_result
for async_request in async_requests:
    result = async_request.get_result()
    print(result.get_response())
    output_data = result.retrieve_output_name_to_numpy("OUTPUT0")
    print("output_data: %s" % output_data)
```

8.2.3.4 全量文本生成

如创建样例代码 test_generate.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_generate.py命令运行该样例。

```
import sys
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    try:
        mindie_client = create_client()
    except Exception as e:
        print("Client Creation falied!")
        sys.exit(1)
    # create input
    prompt = "My name is Olivier and I"
    model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
    parameters = {
        "do_sample": True,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.9,
        "truncate": 5,
        "typical_p": 0.9,
        "seed": 1,
        "repetition_penalty": 1,
        "watermark": True,
        "details": True,
    }
    # apply model inference
    result = mindie_client.generate(
        model_name,
        prompt,
        request_id="1",
        parameters=parameters,
```



```
)
print(result.get_response())
```

8.2.3.5 流式文本生成

如创建样例代码 test_generate_stream.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_generate_stream.py命令运行该样例。

```
import sys
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    try:
        mindie_client = create_client()
    except Exception as e:
        print("Client Creation falied!")
        sys.exit(1)
    # create input
    prompt = "My name is Olivier and I"
    model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
    parameters = {
        "do_sample": True,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.9,
        "truncate": 5,
        "typical_p": 0.9,
        "seed": 1,
        "repetition_penalty": 1,
        "watermark": True,
        "details": True,
    }
    # apply model inference
    results = mindie_client.generate_stream(
        model_name,
        prompt,
        request_id="1",
        parameters=parameters,
    )
    generated_text = ""
    for cur_res in results:
        if cur_res == "</s>":
            break
        generated_text += cur_res
        print("cur text: %s" % cur_res)
    print("final generated text: %s" % generated_text)
```

8.2.3.6 查询服务状态

如创建样例代码 test_query.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_query.py命令运行该样例。

```
import sys
import json
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    try:
        mindie_client = create_client()
    except Exception as e:
        print("Client Creation falied!")
        sys.exit(1)
```



```
model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
if mindie_client.is_server_live():
    print("The server is alive!")
else:
    print("The server is not alive!")
    sys.exit(1)
if mindie_client.is_server_ready():
    print("The server is ready!")
else:
    print("The server is not ready!")
    sys.exit(1)
if mindie_client.is_model_ready(model_name):
    print("The model is ready!")
else:
    print("The model is not ready!")
    sys.exit(1)
server_metadata_dict = mindie_client.get_server_metadata()
print("get_server_metadata: %s" % json.dumps(server_metadata_dict))
model_metadata_dict = mindie_client.get_model_metadata(model_name)
print("get_model_metadata: %s" % json.dumps(model_metadata_dict))
model_config_dict = mindie_client.get_model_config(model_name)
print("get_model_config: %s" % json.dumps(model_config_dict))
# get slots
result = mindie_client.get_slot_count(model_name)
print("gets_slot_count: %s" % result.get_response())
```

8.2.3.7 提前中止请求

如创建样例代码 test_cancel.py（需要和**8.2.3.1 创建客户端**中创建客户端的代码样例utils.py在同一个目录下），并根据实际情况修改model_name，然后执行python test_cancel.py命令运行该样例。

```
import sys
from utils import create_client

if __name__ == "__main__":
    # get argument and create client
    try:
        mindie_client = create_client()
    except Exception as e:
        print("Client Creation falied!")
        sys.exit(1)
    # create input
    prompt = "My name is Olivier and I"
    model_name = "llama_65b" # 需要和服务端配置的modelName保持一致
    parameters = {
        "do_sample": True,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.9,
        "truncate": 5,
        "typical_p": 0.9,
        "seed": 1,
        "repetition_penalty": 1,
        "watermark": True,
        "details": True,
    }
    # apply model inference
    results = mindie_client.generate_stream(
        model_name,
        prompt,
        request_id="1",
        parameters=parameters,
    )
    generated_text = ""
    index = 0
    for cur_res in results:
        index += 1
```



```
    if index == 10:
        flag = mindie_client.cancel(model_name, "1")
        if flag:
            print("Test cancel api succeed!")
            sys.exit(0)
        else:
            print("Test cancel api failed!")
            sys.exit(1)
    print("current result: %s" % cur_res)
```

8.3 MindIE MS

8.3.1 概述

MindIE MS（MindIE management service）作为MindIE的集群管理组件，主要服务于集群场景，根据应用场景，分为以下三个组件。

表 8-14 组件分类

组件名称	应用场景	能力描述
Deployer	集群部署时的简化部署场景。	部署器，底层集成Kubernetes（简称K8s）生态，主要支持对MindIE Server服务集群的一键式部署管理，详情请参见**8.3.2 部署器（Deployer）**。
Controller	集群运行时的运维管理场景。	控制器，完成集群内所有MindIE Server的业务状态监控、PD身份管理与决策、资源管理决策等，是整个集群的状态监控器和决策大脑，详情请参见**8.3.3 控制器（Controller）**。
Coordinator	集群运行时的推理请求入口场景。	调度器，是用户推理请求的入口，接收高并发的推理请求，进行请求调度、请求管理、请求转发等，是整个集群的数据请求管理入口，详情请参见**8.3.4 调度器（Coordinator）**。

特性支持度

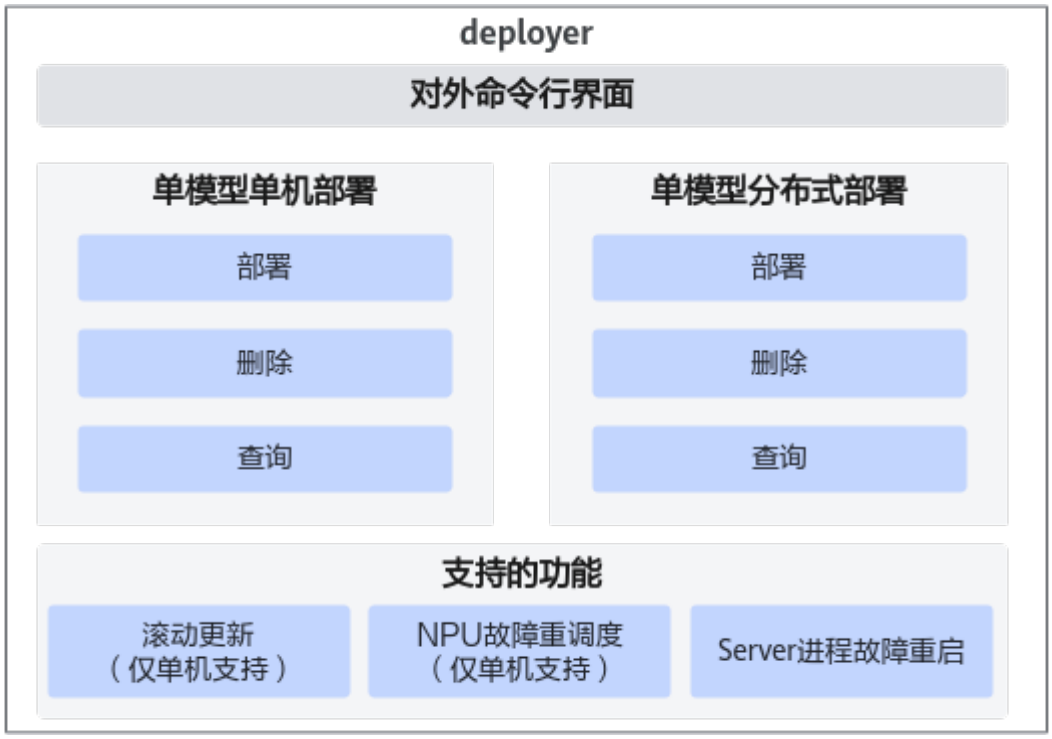
特性	Atlas 300I Duo 推理卡+Atlas 800 推理服务器（型号：3000）	Atlas 800I A2 推理产品
单机服务部署	支持	支持
Prefix Cache	支持	支持
分布式多机服务部署	不支持	支持
PD分离服务部署	不支持	支持

8.3.2 部署器（Deployer）

8.3.2.1 功能介绍

基于Kubernetes（简称K8s）平台，整合Kubernetes生态组件（包括昇腾MindCluster组件），提供对MindIE Server推理服务的一键式自动部署能力，用户可以通过deployer提供的命令行工具，实现一键自动化部署和运维管理，简化用户部署MindIE Server推理服务集群的流程，以下是MindIE MS部署面支持的特性框架图。

图 8-4 特性框架图



支持的特性如下所示：

- 支持命令行操作界面；
- 提供单模型单机的部署能力；
- 支持单模型分布式部署的部署能力；
- 支持滚动更新（仅单机支持）、NPU故障重调度（仅单机支持）、Server进程故障重启等能力。

8.3.2.2 配置说明

MindIE MS Deployer启动时，需要启动的核心组件为ms_server，相关命令解释如**表8-15**所示。

表 8-15 MindIE MS 服务端相关命令

指令	说明
./ms_server ms_server.json	启动MindIE MS服务端命令。ms_server是MindIE Service中的MindIE MS服务端组件，命令中ms_server.json配置文件的具体配置和参数说明请参见**ms_server.json配置文件**。 ms_server.json需要不大于640权限，否则启动失败。
./ms_server -h/--help	查询命令使用方法。
进程退出信号	./ms_server进程注册了信号处理函数捕获SIGINT和SIGTERM信号，用户启动ms_server后，通过发送SIGINT，SIGTERM信号可正常停止该进程。

ms_server.json 配置文件

```
{
    "ip": "127.0.0.1",
    "port": 9789,
    "k8s_apiserver_ip": "127.0.0.1",
    "k8s_apiserver_port": 6443,
    "log_info": {
        "log_level": "INFO",
        "run_log_path": "/var/log/mindie-ms/run/log.txt",
        "operation_log_path": "/var/log/mindie-ms/operation/log.txt"
    },
    "ms_status_file": "/var/lib/mindie-ms/status.json",
    "server_tls_items": {
        "ca_cert" : "./security/msserver/security/certs/ca.pem",
        "tls_cert": "./security/msserver/security/certs/cert.pem",
        "tls_key": "./security/msserver/security/keys/cert.key.pem",
        "tls_passwd": "./security/msserver/security/pass/key_pwd.txt",
        "kmcKsfMaster": "./security/msserver/tools/pmt/master/ksfa",
        "kmcKsfStandby": "./security/msserver/tools/pmt/standby/ksfb",
        "tls_crl": ""
    },
    "client_k8s_tls_enable": true,
    "client_k8s_tls_items": {
        "ca_cert" : "./security/kubeclient/security/certs/ca.pem",
        "tls_cert": "./security/kubeclient/security/certs/cert.pem",
        "tls_key": "./security/kubeclient/security/keys/cert.key.pem",
        "tls_passwd": "./security/kubeclient/security/pass/key_pwd.txt",
        "kmcKsfMaster": "./security/kubeclient/tools/pmt/master/ksfa",
        "kmcKsfStandby": "./security/kubeclient/tools/pmt/standby/ksfb",
        "tls_crl": ""
    },
    "client_mindie_server_tls_enable": true,
    "client_mindie_tls_items": {
        "ca_cert" : "./security/mindieclient/security/certs/ca.pem",
        "tls_cert": "./security/mindieclient/security/certs/cert.pem",
        "tls_key": "./security/mindieclient/security/keys/cert.key.pem",
        "tls_passwd": "./security/mindieclient/security/pass/key_pwd.txt",
        "kmcKsfMaster": "./security/mindieclient/tools/pmt/master/ksfa",
        "kmcKsfStandby": "./security/mindieclient/tools/pmt/standby/ksfb",
        "tls_crl": ""
    }
}
```


表 8-16 ms_server.json 参数解释

参数	类型	说明
ip	String	必填。 MindIE MS服务端IP；仅支持IPv4，如配置了环境变量MINDIE_MS_SERVER_IP，则优先从环境变量读取。 考虑到MindIE MS运行业务的安全问题，建议不对外提供服务，配置成127.0.0.1；使用其他IP对外暴露接口将有安全风险，请用户需要慎重使用。
port	Int	必填。 MindIE MS服务端对外端口，取值范围：[1024，65535]，默认值9789。
k8s_apiserver_ip	String	必填。 Kubernetes管理节点的物理机IP地址。
k8s_apiserver_port	Int	必填。 Kubernetes对外访问的HTTPS端口，取值范围：[1024，65535]，默认值：6443（该端口为K8s的HTTPS端口）。
log_info	Object	MindIE MS服务端日志配置信息，包括log_level、run_log_path和operation_log_pat子参数，详情请参见表8-18。
ms_status_file	String	必填。 用于服务状态保存的文件路径。**该参数仅分布式多机场景支持**。 用户需要提前创建该文件所在的目录，第一次启动ms_server时会自动创建该文件，用户不需要自己创建，也不能手动修改。 后续部署的服务状态信息将由ms_server自动写入该文件，ms_server重启将读取该文件恢复状态。该文件的格式如下表8-17
server_tls_items	Object	MindIE MS服务端tls配置，详情请参见表8-19。
client_k8s_tls_enable	Bool	必填。 MindIE MS与Kubernetes对接的客户端是否需要开启tls安全通信，建议用户打开，确保与kubernetes通信安全。 • client_k8s_tls_enable为true，访问Kubernetes的HTTPS接口。 • client_k8s_tls_enable为false，访问Kubernetes的HTTP接口。
client_k8s_tls_items	Object	Kubernetes客户端tls配置，当client_k8s_tls_enable为true时，该参数中的key值为必填项，详情请参见表8-20。

参数	类型	说明
client_mindie_server_tls_enable	Bool	必填。 MindIE MS对接MindIE Server接口是否开启tls安全通信。建议用户打开，确保管理节点MS与计算节点MindIE Server间通信安全。
client_mindie_tls_items	Object	MindIE Server客户端tls配置，当client_mindie_server_tls_enable为true时，该参数中的key值为必填项，详情请参见表8-21。

表 8-17 status 持久化业务状态文件解释

参数	类型	说明
server_list	Array	表示MS管理的推理服务列表清单。 未部署服务时为空。
namespace	String	表示服务的命名空间。
replicas	Int	推理实例数。
server_name	Sting	推理服务名。
server_type	String	服务类型，当前只支持多机推理服务，取值mindie_cross_node。
use_service	Bool	是否使用kubernetes的Service暴露服务端口。

表 8-18 log_info 子参数解释

参数	类型	说明
log_level	String	必填。 支持等级如下： • DEBUG • INFO • WARNING • ERROR • CRITICAL
run_log_path	String	MindIE MS服务端运行日志保存文件路径。 可访问文件，必填。 Kubernetes容器化部署MindIE MS服务端时，必须为/var/log/mindie-ms/run/log.txt。

参数	类型	说明
operation_log_path	String	MindIE MS服务端操作保存文件路径。 可访问文件，必填。 Kubernetes容器化部署MindIE MS服务端时，必须为/var/log/mindie-ms/operation/log.txt。

表 8-19 server_tls_items 子参数解释

参数	类型	说明
ca_cert	String	必填。 MindIE MS HTTP服务端的CA证书，使用kubernetes集群CA证书。MindIE MS服务端会校验客户端证书信息的CN为msclientuser，O为msgroup，请确保客户端的证书是kubernetes CA所信任的，且包含这些信息。 MindIE MS服务端ca根证书文件路径，该路径真实存在且可读。
tls_cert	String	必填。 使用kubernetes集群CA证书签发的服务证书。 MindIE MS服务端tls证书文件路径，该路径真实存在且可读。
tls_key	String	必填。 使用kubernetes集群CA证书签发的服务证书私钥。 MindIE MS服务端tls私钥文件路径，该路径真实存在且可读。
tls_passwd	String	必填。 MindIE MS HTTP服务端的KMC加密的私钥口令的文件路径。
kmcKsfMaster	String	必填。 MindIE MS HTTP服务端加密口令的KMC密钥库文件。
kmcKsfStandby	String	必填。 MindIE MS HTTP服务端加密口令的KMC standby密钥库备份文件。
tls_crl	String	必填。 MindIE MS服务端校验客户端的证书吊销列表crl文件路径，该路径存在且可读。如为空，则不进行吊销校验。

表 8-20 client_k8s_tls_items 子参数解释

参数	类型	说明
ca_cert	String	必填。 使用kubernetes集群CA证书。 MindIE MS和Kubernetes对接的客户端的ca根证书文件路径，该路径真实存在且可读。
tls_cert	String	必填。 使用kubernetes集群CA证书签发的客户端证书，证书的CN与通过kubernetes RBAC机制创建的用户名一致。 MindIE MS和Kubernetes对接的客户端的tls证书文件路径，该路径真实存在且可读。
tls_key	String	必填。 使用kubernetes集群CA证书签发的客户端证书私钥。 MindIE MS和Kubernetes对接的客户端的tls私钥文件路径，该路径真实存在且可读。
tls_passwd	String	必填。 kmc加密的私钥口令的文件路径。
kmcKsfMaster	String	必填。 MS Server作为kube api-server的客户端加密口令的KMC密钥库文件。
kmcKsfStandby	String	必填。 MS Server作为kube api-server的客户端加密口令的KMC密钥库备份文件。
tls_crl	String	必填。 MindIE MS服务端校验Kubernetes API Server的证书吊销列表crl文件路径，如为空，则不进行吊销校验，文件存在路径存在并可读的。

表 8-21 client_mindie_tls_items 子参数解释

参数	类型	说明
ca_cert	String	必填。 MindIE MS和MindIE Server对接的客户端的ca根证书文件路径，该路径真实存在且可读。
tls_cert	String	必填。 MindIE MS和MindIE Server对接的客户端的tls证书文件路径，该路径真实存在且可读。

参数	类型	说明
tls_key	String	必填。 MindIE MS和MindIE Server对接的客户端的tls私钥文件路径，该路径真实存在且可读。
tls_passwd	String	必填。 kmc加密的私钥口令的文件路径。
kmcKsfMaster	String	必填。 MS Server作为kube api-server的客户端加密口令的KMC密钥库文件。
kmcKsfStandby	String	必填。 MS Server作为kube api-server的客户端加密口令的KMC密钥库备份文件。
tls_crl	String	必填。 MindIE MS服务端校验MindIE Server的证书吊销列表crl文件路径，如为空，则不进行吊销校验，文件存在路径存在并可读的。

8.3.2.3 RESTful 接口 API

8.3.2.3.1 推理服务部署接口

接口功能

推理服务部署，发起部署请求，异步接口，部署配置信息请参见**表3-6**。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/servers

须知

- 滚动更新配置max_unavailable和max_surge不能全为0，另外若配置max_unavaiable > 0, max_surge = 0，但是replicas * max_unavaiable 向下取整为0，那么此时将触发强制更新。
- 当前滚动更新过程使用kubernetes的原生Service进行负载均衡，若用户使用http长链接发送请求，在更新过程中Service不会主动中断请求客户端与正在退出的旧推理服务实例的长链接，新的请求进入旧实例将被拒绝，导致请求失败。建议用户在遇到中断后主动重新建链或使用短链接（一次请求响应后断链）发送请求。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/servers
```

响应样例：

```
{
   "message": "creating the server!",
   "status": "0"
}
```

输出说明

表 8-22 请求响应状态码

code	说明
200	ok：请求成功。
400	bad_request：请求失败，非法请求。
404	not_found：请求失败，找不到资源。
500	internal_server_error：请求失败，内部出现错误。

8.3.2.3.2 推理服务查询接口

接口功能

查询服务的部署状态，包括部署阶段、就绪状态、模型信息等。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/servers/*{server_name}*

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/servers/{server_name}
```

响应样例：

```
{
   "data": {
      "instances_status": [
         {
            "liveness": true,
```



```
            "pod_name": "mindie-server-zsm-586c8fb5f8-vtx2n",
            "readiness": true
        }
    ],
    "model_info": {
        "docker_label": null,
        "max_batch_total_tokens": 8192,
        "max_best_of": 1,
        "max_concurrent_requests": 200,
        "max_input_length": 2048,
        "max_stop_sequences": null,
        "max_waiting_tokens": null,
        "models": [
            {
                "max_total_tokens": 2560,
                "model_device_type": "npu",
                "model_dtype": "float16",
                "model_id": "llama2_7b",
                "model_pipeline_tag": "text-generation",
                "model_sha": null
            }
        ],
        "sha": null,
        "validation_workers": null,
        "version": "1.0.RC3",
        "waiting_served_ratio": null
    },
    "server_name": "mindie-server"
  },
  "message": "success",
  "status": "0"
}
```


重要参数解释：

liveness：表示服务存活状态，取值如下：

- true：表示服务存活。
- false：表示服务未存活。

readiness：表示服务实例启动状态，取值如下：

- true：表示服务实例已启动完成并进入就绪状态。
- false：表示服务实例未启动完成。

输出说明

表 8-23 请求响应状态码

code	说明
200	ok：请求成功。
400	bad_request：请求失败，非法请求。
404	not_found：请求失败，找不到资源。
500	internal_server_error：请求失败，内部出现错误。

8.3.2.3.3 推理服务卸载接口

接口功能

卸载服务，删除相关已创建资源。

接口格式

操作类型：**DELETE**

URL：https://{ip}:{port}/v1/servers/*{server_name}*

请求参数

无

使用样例

请求样例：

```
DELETE https://{ip}:{port}/v1/servers/{server_name}
```

响应样例：

```
{
    "message": "succeed to clear resources",
    "status": "0"
}
```

输出说明

表 8-24 请求响应状态码

code	说明
200	ok：请求成功。
400	bad_request：请求失败，非法请求。
404	not_found：请求失败，找不到资源。
500	internal_server_error：请求失败，内部出现错误。

8.3.2.3.4 推理服务更新接口

接口功能

用于强制滚动更新服务，用户更新镜像后可通过调用该接口更新服务，更新的策略在**8.3.2.3.1 推理服务部署接口**服务部署时指定。

说明

当前只支持单机（非分布式）部署场景的更新。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/servers/*{server_name}*

请求参数

参数	类型	说明
server_name	string	必填。 待更新的服务名称，需与服务部署配置指定的server_name一致。

使用样例

准备服务更新update_server.json配置文件，如下所示：

```
{
    "server_name": "mindie-server"
}
```

请求样例：

```
POST https://{ip}:{port}/v1/servers/{server_name}
```

响应样例：

```
{
    "message": "update server success.",
    "status": "0"
}
```

输出说明

表 8-25 请求响应状态码

code	说明
200	ok：请求成功。
400	bad_request：请求失败，非法请求。
404	not_found：请求失败，找不到资源。
500	internal_server_error：请求失败，内部出现错误。

8.3.2.4 错误码说明

当前只会返回两种错误码，如下所示。

具体错误客户端会在屏幕打印异常信息；服务端的错误信息请参见**表8-18**中"run_log_path"参数配置的文件路径。

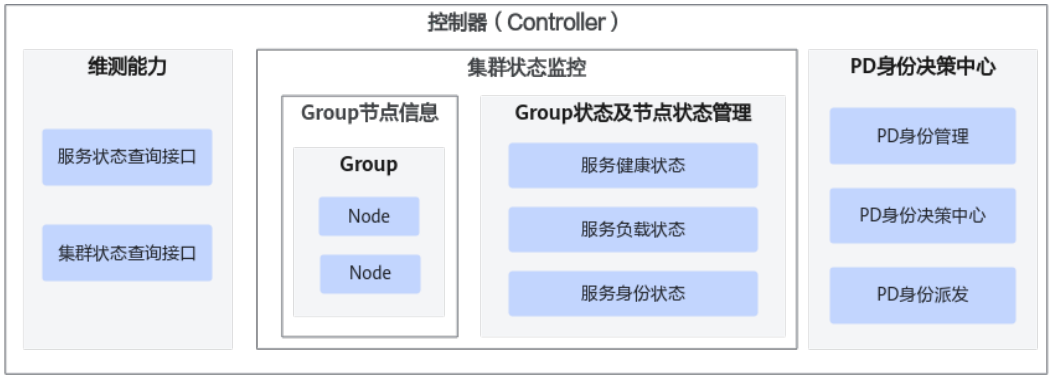
- 0：表示服务正常。
- 1：表示异常情况。

8.3.3 控制器（Controller）

8.3.3.1 功能介绍

MindIE MS控制器（Controller）作为整个MindIE Server集群的身份决策大脑及状态监控中心，主要功能包括：集群节点状态监控、PD身份决策与下发等，其架构图如图8-5所示。

图 8-5 控制器（Controller）架构图



8.3.3.2 安装部署

使用MindIE MS控制器（Controller）之前，需要完成以下环境准备。

1. 已参见3.3 准备MindIE Server镜像完成MindIE Server镜像制作。
2. 参见 准备TLS证书提前准备好证书，将证书路径配置到启动配置文件（ms_controller.json）对应证书路径。
3. 参见3.6.3.2.3 使用kubectl部署PD分离服务示例完成部署。

8.3.3.3 配置说明

启动配置文件（ms_controller.json）

ms_controller.json配置文件样例如下所示，参数解释请参见ms_controller.json配置文件参数解释，用户可根据具体场景进行配置。

```
{
  "deploy_mode": "pd_separate",
  "process_manager" : {
    "to_file": true,
    "file_path": "./logs/controller_process_status.json"
  },
  "cluster_status" : {
    "to_file": true,
    "file_path": "./logs/cluster_status_output.json"
  },
  "global_rank_table_file_path": "./conf/global_rank_table_file.json",
  "cluster_synchronization_seconds": 1,
  "rank_table_detecting_seconds": 60,
  "disappeared_server_waiting_seconds": 120,
```



```
"default_p_rate": 0,
"default_d_rate": 0,
"is_heterogeneous": false,
"server_online_attempt_times": 60,
"server_online_wait_seconds": 5,
"init_role_attempt_times": 2,
"check_role_attempt_times": 60,
"check_role_wait_seconds": 5,
"mindie_server_control_port": 1026,
"mindie_server_port": 1025,
"mindie_ms_coordinator_port": 1026,
"http_timeout_seconds": 20,
"http_retries": 3,
"role_decision_methods": "digs",
"digs_request_summary_input_length": 3000,
"digs_request_summary_output_length": 200,
"digs_model_config_path":"./conf/model_config/llama2-70B.json",
"digs_machine_config_path":"./conf/machine_config/800IA2.json",
"digs_prefill_slo": 1000,
"digs_decode_slo": 50,
"model_type": "llama2-70B",
"transfer_type": "D2DTransfer",
"digs_pp": 1,
"http_server": {
  "ip" : "127.0.0.1",
  "port": 1026
},
"tls_config": {
  "request_coordinator_tls_enable": true,
  "request_coordinator_tls_items": {
    "ca_cert" : "./security/request_coordinator/security/certs/ca.pem",
    "tls_cert": "./security/request_coordinator/security/certs/cert.pem",
    "tls_key": "./security/request_coordinator/security/keys/cert.key.pem",
    "tls_passwd": "./security/request_coordinator/security/pass/key_pwd.txt",
    "kmc_ksf_master": "./security/request_coordinator/tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./security/request_coordinator/tools/pmt/standby/ksfb",
    "tls_crl": ""
  },
  "request_server_tls_enable": true,
  "request_server_tls_items": {
    "ca_cert" : "./security/request_server/security/certs/ca.pem",
    "tls_cert": "./security/request_server/security/certs/cert.pem",
    "tls_key": "./security/request_server/security/keys/cert.key.pem",
    "tls_passwd": "./security/request_server/security/pass/key_pwd.txt",
    "kmc_ksf_master": "./security/request_server/tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./security/request_server/tools/pmt/standby/ksfb",
    "tls_crl": ""
  },
  "http_server_tls_enable": true,
  "http_server_tls_items": {
    "ca_cert" : "./security/http_server/security/certs/ca.pem",
    "tls_cert": "./security/http_server/security/certs/cert.pem",
    "tls_key": "./security/http_server/security/keys/cert.key.pem",
    "tls_passwd": "./security/http_server/security/pass/key_pwd.txt",
    "kmc_ksf_master": "./security/http_server/tools/pmt/master/ksfa",
    "kmc_ksf_standby": "./security/http_server/tools/pmt/standby/ksfb",
    "tls_crl": ""
  }
},
"log_info": {
  "log_level": "INFO",
  "to_file": false,
  "to_stdout": true,
  "run_log_path": "./logs/ms_controller_run_log.txt",
  "operation_log_path": "./logs/ms_controller_operation_log.txt",
  "max_log_str_size": 4096,
  "max_log_file_size": 20,
  "max_log_file_num": 10
```



```
    }
}
```

ms_controller.json 配置文件参数解释

ms_controller.json配置文件中各个字段解释如表8-26所示，用户可根据具体场景进行配置。

表 8-26 参数说明

参数名称	取值范围	配置说明
deploy_mode	部署模式。 • pd_separate：PD分离部署； • single_node：无身份部署。	部署模式。 必填；默认值为pd_separate。
process_manager：进程状态备份功能		
to_file	• true：开启。 • false：关闭。	必填；默认值为true。 是否开启进程状态备份功能。 如果开启，会在程序启动时，自动加载进程状态备份文件。在程序运行的过程中，自动生成进程状态备份文件。备份文件详情请参见表8-27。
file_path	-	开启to_file开关时必填。 进程状态备份文件路径，用户不能创建文件，只能填写路径，要求该文件所在的路径为真实存在且可读的持久化路径，运行时会自动创建文件。如果进程文件保存在容器中的临时路径，在controller容器重启后，将发生文件丢失，需要重启整个集群才能恢复正常服务。 如果修改身份决策的相关配置，则需要删除进程状态备份文件，重启整个集群，否则将继承历史的身份决策结果。
cluster_status：集群服务状态导出功能		
to_file	• true：开启。 • false：关闭。	必填；默认值为true。 是否开启集群服务状态导出功能。 如果开启，在程序运行的过程中会输出集群状态至文件。集群服务状态输出文件详情请参见集群服务状态输出文件。

参数名称	取值范围	配置说明
file_path	-	开启to_file开关时必填。 集群状态导出文件路径，要求该文件所在的路径真实存在且可读，运行时会自动创建文件。
global_rank_table_file_path	-	必填。 全局集群信息表路径，要求该文件真实存在且可读。 如设置环境变量GLOBAL_RANK_TABLE_FILE_PATH，则优先读取环境变量的值。 **说明** 全局集群信息表是Controller监控集群状态、管理Server节点的信息源。需要保证文件的安全性和内容的正确性。
cluster_synchronization_seconds	[1,65535]，单位为秒。	必填；默认值为1秒。 同步集群信息的时间周期。
rank_table_detecting_seconds	[1,65535]，单位为秒。	必填；默认值为60秒。 读取集群信息表的周期。
disappeared_server_waiting_seconds	[1,65535]，单位为秒。	必填；默认值为120秒。 全局集群信息表中，删除Server的等待时间，到此时间时Server将会被移除。
default_p_rate	[0,15]	必填；默认值为0。 PD分离部署模式下，P所占的比例。 ● 0：表示自动决策最佳比例，default_d_rate需要同时为0； ● 非0：表示指定P的比例，default_d_rate需要同时非0，且default_p_rate和default_d_rate的和小于等于16。 如设置环境变量MINDIE_MS_P_RATE，则优先读取环境变量的值。
default_d_rate	[0,15]	必填；默认值为0。 PD分离部署模式下，D所占的比例。 ● 0：表示自动决策最佳比例，default_p_rate需要同时为0； ● 非0：表示指定D的比例，default_p_rate需要同时非0，且default_p_rate和default_d_rate的和小于等于16。 如设置环境变量MINDIE_MS_D_RATE，则优先读取环境变量的值。

参数名称	取值范围	配置说明
is_heterogeneous	• false：表示同构场景；即Server在不同硬件设备上随机指定PD身份。 • true：表示异构场景；即支持指定Atlas 800I A2 推理产品（32G）为P节点，指定Atlas 800I A2 推理产品（64G）为D节点。	必填；默认值为false。 是否为异构场景，异构场景需要同步修改集群信息表。 **说明** 目前暂不支持异构场景。
server_online_attempt_times	[1,65535]	必填；默认值为60。 集群初始化时，Server节点上线状态的检查次数
server_online_wait_seconds	[1,65535]，单位为秒。	必填；默认值为5。 集群初始化时，Server节点上线状态的检查间隔
init_role_attempt_times	[1,65535]	必填；默认值为2。 Server身份初始化时的重试次数，只有当检查到身份为Unkonwn才会触发重新下发身份的操作。
check_role_attempt_times	[1,65535]	必填；默认值为60。 Server身份初始化的检查次数。
check_role_wait_seconds	[1,65535]，单位为秒。	必填；默认值为5秒。 Server身份初始化的检查间隔。
mindie_server_control_port	[1024,65535]	必填；默认值为1026。 MindIE Server的管理端口。
mindie_server_port	[1024,65535]	必填；默认值为1025。 MindIE Server的数据端口。
mindie_ms_coordinator_port	[1024,65535]	必填；默认值为1026。 MindIE Coordinator的管理端口。
http_timeout_seconds	[1,65535]，单位为秒。	必填；默认值为20秒。 通信超时时间。
http_retries	[0,65535]	必填；默认值为3。 通信异常重试次数。

参数名称	取值范围	配置说明
role_decision_methods	digs	必填；默认值为digs。 PD身份决策算法；目前仅支持digs。
digs_request_summary_input_length	[1,65535]	必填；默认值为3000。 推理请求的平均输入长度。 确定模型及设备配置后，此参数和digs_request_summary_output_length是影响PD最佳比例计算的主要参数。 ● 推理请求为长输入短输出时，P实例的个数多于D实例的个数； ● 推理请求为短输入长输出时，D实例的个数多于P实例的个数。
digs_request_summary_output_length	[1,65535]	必填；默认值为200。 推理请求的平均输出长度。 确定模型及设备配置后，此参数和digs_request_summary_input_length是影响PD最佳比例计算的主要参数。 ● 推理请求为长输入短输出时，P实例的个数多于D实例的个数； ● 推理请求为短输入长输出时，D实例的个数多于P实例的个数。
digs_model_config_path	-	必填；默认值为llama2-70B模型的配置文件路径。 身份决策算法需要使用的模型参数信息，要求该文件真实存在且可读，格式参考模型配置文件描述。
digs_machine_config_path	-	必填；默认值为Atlas 800I A2 推理产品的硬件参数文件路径。 身份决策算法需要使用的机器参数信息，要求该文件真实存在且可读，格式参考硬件设备文件描述。
digs_prefill_slo	[1,65535]	必填；默认值为1000。 Prefill速率。
digs_decode_slo	[1,65535]	必填；默认值为50。 Decode速率。
model_type	-	必填；默认值为llama2-70B。 推理模型的名称。
transfer_type	D2DTransfer	必填；默认值为D2DTransfer。 传输类型。

参数名称	取值范围	配置说明
digs_pp	[1,65535]	必填；默认值为1。 任务并行数。
http_server：MindIE MS Controller HTTP服务端配置		
ip	-	必填；默认值为127.0.0.1。 MindIE MS服务端IP，用于健康检查。 如设置环境变量POD_IP，则优先读取环境变量的值。
port	[1024,65535]	必填；默认值为1026。 MindIE MS服务端端口，用于健康检查。
request_coordinator_tls_enable	• true：开启。 • false：关闭。	必填；默认值为true。 是否开启与MindIE MS Coordinator通信接口的tls安全认证。建议用户开启，确保与MindIE MS Coordinator的通信安全。如果关闭则存在较高的网络安全风险。
request_coordinator_tls_items：MindIE MS Coordinator HTTP客户端的证书相关配置		
ca_cert	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端ca根证书文件路径，要求该文件真实存在且可读。
tls_cert	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端tls证书文件路径，要求该文件真实存在且可读。
tls_key	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端tls私钥文件路径，要求该文件真实存在且可读。
tls_passwd	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端的KMC加密的私钥口令的文件路径，要求该文件真实存在且可读。
kmc_ksf_master	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端加密口令的KMC密钥库文件，要求该文件真实存在且可读。

参数名称	取值范围	配置说明
kmc_ksf_standby	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端加密口令的KMC standby密钥库备份文件，要求该文件真实存在且可读。
tls_crl	-	开启tls时必填。 MindIE MS Coordinator HTTP客户端校验证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。
request_server_tls_enable	• true：开启。 • false：关闭。	必填。默认值为true。 是否开启与MindIE Server 通信接口的tls安全认证。建议用户开启，确保与MindIE Server的通信安全。如果关闭则存在较高的网络安全风险。
request_server_tls_items：MindIE Server HTTP客户端的证书相关配置		
ca_cert	-	开启tls时必填。 MindIE Server HTTP客户端ca根证书文件路径，要求该文件真实存在且可读。
tls_cert	-	开启tls时必填。 MindIE Server HTTP客户端tls证书文件路径，要求该文件真实存在且可读。
tls_key	-	开启tls时必填。 MindIE Server HTTP客户端tls私钥文件路径，要求该文件真实存在且可读。
tls_passwd	-	开启tls时必填。 MindIE Server HTTP客户端的KMC加密的私钥口令的文件路径，要求该文件真实存在且可读。
kmc_ksf_master	-	开启tls时必填。 MindIE Server HTTP客户端加密口令的KMC密钥库文件，要求该文件真实存在且可读。
kmc_ksf_standby	-	开启tls时必填。 MindIE Server HTTP客户端加密口令的KMC standby密钥库备份文件，要求该文件真实存在且可读。
tls_crl	-	开启tls时必填。 MindIE Server HTTP客户端校验证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。

参数名称	取值范围	配置说明
http_server_tls_enable	• true：开启。 • false：关闭。	必填。默认值为true。 是否开启MindIE MS Controller HTTP服务端tls。建议用户开启，确保MindIE MS Controller与客户端之间的通信安全。如果关闭则存在较高的网络安全风险。
http_server_tls_items属性：MindIE MS Controller HTTP服务端的证书相关配置		
ca_cert	-	开启tls时必填。 MindIE MS Controller HTTP服务端ca根证书文件路径，要求该文件真实存在且可读。
tls_cert	-	开启tls时必填。 MindIE MS Controller HTTP 服务端tls证书文件路径，要求该文件真实存在且可读。
tls_key	-	开启tls时必填。 MindIE MS Controller HTTP 服务端tls私钥文件路径，要求该文件真实存在且可读。
tls_passwd	-	开启tls时必填。 MindIE MS Controller HTTP 服务端的KMC加密的私钥口令的文件路径，要求该文件真实存在且可读。
kmc_ksf_master	-	开启tls时必填。 MindIE MS Controller HTTP 服务端加密口令的KMC密钥库文件，要求该文件真实存在且可读。
kmc_ksf_standby	-	开启tls时必填。 MindIE MS Controller HTTP 服务端加密口令的KMC standby密钥库备份文件，要求该文件真实存在且可读。
tls_crl	-	开启tls时必填。 MindIE MS Controller HTTP 服务端校验客户端的证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。
log_info：日志功能		

参数名称	取值范围	配置说明
log_level	• CRITICAL • ERROR • WARNING • INFO • DEBUG	必填；默认值为INFO。 设置日志级别。 如设置环境变量MINDIEMS_LOG_LEVEL，则优先读取环境变量的值。
to_file	• true：输出到文件。 • false：不输出到文件。	必填；默认值为false。 是否输出到文件。
to_stdout	• true：输出到标准输出流。 • false：不输出到标准输出流。	必填；默认值为true。 是否输出到标准输出流。
run_log_path	-	必填；当to_file为true时生效。 运行日志路径，要求该文件所在的路径真实存在且可读，运行时会自动创建文件。
operation_log_path	-	必填；当to_file为true时生效。 审计日志路径，要求该文件所在的路径真实存在且可读，运行时会自动创建文件。
max_log_str_size	[0,65535]	必填；默认值为4096 单条日志最大长度。
max_log_file_size	[1,100]，单位MB。	必填；默认值为20，当to_file为true时生效。 单个日志文件存储上限。
max_log_file_num	[2,64]	必填；默认值为10，当to_file为true时生效。 最大日志文件存储数量。

进程备份文件

支持持久化保存业务数据文件，通过ms_controller.json配置文件指定路径信息，进程备份文件的更新周期与同步集群状态的周期一致。进程备份文件包含Server是否为故障节点、Server的身份、所属组ID、资源总量和PD分离部署模式下Server节点是否已经完成身份下发等信息。

- 在集群初始化阶段，如果超过Server节点上线状态的检查总时间，Server节点的静态配置采集接口或者动态状态采集接口仍然返回非200，则认为Server节点上线失败，为故障节点。

- 在集群初始化阶段，如果Server节点指定身份接口请求失败，则认为身份下发失败。如果身份下发成功，但是Server节点在身份初始化检查总时间内一直切换失败，则认为Server节点为故障节点。可能有以下两种场景：
  - Server节点被指定为P节点，超过身份初始化的检查总时间，仍然为RoleUnknown或者RoleSwitching状态。
  - Server节点被指定为D节点，超过身份初始化的检查总时间，仍然为RoleUnknown状态或者该D节点和全部的P节点建链失败。

进程备份文件样例如下所示，参数解释请参见表8-27所示。

```
{
  "server": [
    {
      "delete_time": 0,
      "id": xx,
      "ip": "xx.xx.xx.1",
      "model_name": "llama3-8b",
      "is_faulty": false,
      "is_initialized": true,
      "peers": [
        xx,
      ],
      "static_info": {
        "group_id": 0,
        "label": 2,
        "role": 80,
        "total_block_num": 320,
        "total_slots_num": 200
      }
    },
    {
      "delete_time": 0,
      "id": xx,
      "ip": "xx.xx.xx.2",
      "model_name": "llama3-8b",
      "is_faulty": false,
      "is_initialized": true,
      "peers": [
        xx
      ],
      "static_info": {
        "group_id": 0,
        "label": 3,
        "role": 68,
        "total_block_num": 320,
        "total_slots_num": 200
      }
    }
  ]
}
```

表 8-27 进程备份文件参数解释

参数	类型	描述
delete_time	int64_t	Server节点在ranktable中的删除时间。
id	uint64_t	Server节点的IP地址转换得到的唯一ID。
ip	string	Server节点的IP地址。
model_name	string	推理使用的模型。
is_faulty	bool	Server节点是否为故障节点。

参数	类型	描述
is_initialized	bool	Server节点是否已经完成身份下发。 无身份部署模式下，此字段永远为false。
peers	size_t数组	Server节点需要连接的其他节点ID。 ● 当前节点为P节点，则表示需要与该P节点连接的D节点ID。 ● 当前节点为D节点，则表示需要与该D节点连接的P节点ID。
static_info属性		
group_id	uint64_t	Server节点所在的GroupID。
label	枚举类型	Server节点的标签类型： ● 0：表示倾向于作为Prefill实例。 ● 1：表示倾向于作为Decode实例。 ● 2：表示只允许作为Prefill实例。 ● 3：表示只允许作为Decode实例。 当前版本仅支持2和3标签。
role	枚举类型	Server节点的身份： ● 85：表示角色未知。 ● 80：表示该Server节点为P节点。 ● 68：表示该Server节点为D节点。
total_block_num	size_t	Server节点的block总量。
total_slots_num	size_t	Server节点的slot总量。

集群服务状态输出文件

集群状态输出文件主要包含以下内容：

- Coordinator的健康状态，Controller和Coordinator之间的通信接口请求成功，则认为Coordinator为健康状态。
- Server是否为故障节点，以及非故障节点的健康状态。
  - 在集群初始化阶段，如果超过Server节点上线状态的检查总时间后，Server节点的静态配置采集接口或者动态状态采集接口仍然返回非200，则认为Server节点上线失败，为故障节点。
  - 在集群初始化阶段，如果Server节点身份初始化失败，则认为Server节点上线失败，为故障节点。可能有以下两种场景：
    - Server节点被指定为P节点，超过身份初始化的检查总时间，仍然为RoleUnknown或者RoleSwitching状态。

    - Server节点被指定为D节点，超过身份初始化的检查总时间，仍然为RoleUnknown状态或者该D节点和全部的P节点建链失败。
  - 集群初始化完成后，Controller和非故障的Server之间的通信接口请求成功，则认为Server为健康状态。

集群服务状态输出文件样例如下所示，参数解释请参见**表8-28**所示。

```
{
 "coordinator": [
  {
   "ip": "xx.xx.xx.xx",
   "is_healthy": true
  }
 ],
 "server": [
  {
   "delete_time": 0,
   "dynamic_info": {
    "avail_block_num": 320,
    "avail_slots_num": 200
   },
   "ip": "xx.xx.xx.1",
   "is_faulty": false,
   "is_healthy": true,
   "model_name": "llama3-8b",
   "peers": [
    "xx.xx.xx.2",
   ],
   "static_info": {
    "block_size": 128,
    "group_id": 1,
    "label": 2,
    "max_output_len": 512,
    "max_seq_len": 2560,
    "role": 80,
    "total_block_num": 320,
    "total_slots_num": 200
   }
  },
  {
   "delete_time": 0,
   "dynamic_info": {
    "avail_block_num": 244,
    "avail_slots_num": 181
   },
   "ip": "xx.xx.xx.2",
   "is_faulty": false,
   "is_healthy": true,
   "model_name": "llama3-8b",
   "peers": [
    "xx.xx.xx.1"
   ],
   "static_info": {
    "block_size": 128,
    "group_id": 1,
    "label": 3,
    "max_output_len": 512,
    "max_seq_len": 2560,
    "role": 68,
    "total_block_num": 320,
    "total_slots_num": 200
   }
  },
  {
   "delete_time": 0,
   "dynamic_info": {
    "avail_block_num": 0,
    "avail_slots_num": 200
```



```
    },
    "ip": "xx.xx.xx.3",
    "is_faulty": true,
    "is_healthy": false,
    "model_name": "llama3-8b",
    "peers": [
      "xx.xx.xx.4"
    ],
    "static_info": {
      "block_size": 128,
      "group_id": 0,
      "label": 2,
      "max_output_len": 512,
      "max_seq_len": 2560,
      "role": 80,
      "total_block_num": 1024,
      "total_slots_num": 200
    }
  },
  {
    "delete_time": 0,
    "dynamic_info": {
      "avail_block_num": 0,
      "avail_slots_num": 200
    },
    "ip": "xx.xx.xx.4",
    "is_faulty": true,
    "is_healthy": true,
    "model_name": "llama3-8b",
    "peers": [
      "xx.xx.xx.3"
    ],
    "static_info": {
      "block_size": 128,
      "group_id": 0,
      "label": 3,
      "max_output_len": 512,
      "max_seq_len": 2560,
      "role": 68,
      "total_block_num": 1024,
      "total_slots_num": 200
    }
  }
 ]
}
```


表 8-28 集群服务状态输出文件参数解释

参数	类型	描述
Coordinator信息		
ip	string	Coordinator节点的IP地址。
is_healthy	bool	Coordinator节点的健康状态。 ● true：表示健康，Controller与该Coordinator通信正常。 ● false：表示非健康，Coordinator与该Coordinator通信正常。
Server信息		
delete_time	int64_t	Server节点在ranktable中的删除时间。

参数	类型	描述
dynamic_info属性		
avail_block_num	size_t	Server节点的可用block数量。
avail_slots_num	size_t	Server节点的可用slot数量。
ip	string	Server节点的IP地址。
is_faulty	bool	是否为故障节点。
is_healthy	bool	是否为健康节点。 当节点为故障节点时，此字段仅表示Controller与该Server的最后一次通信是否成功。
model_name	string	推理使用的模型。
peers	string数组	Server节点需要连接的其他节点。 ● 当前节点为P节点，则表示需要与该P节点连接的D节点。 ● 当前节点为D节点，则表示需要与该D节点连接的P节点。
static_info属性		
block_size	size_t	KV Cache block的size大小。
group_id	uint64_t	Server节点所在的GroupID。
label	枚举类型	Server节点的标签类型： ● 0，表示倾向于作为Prefill实例。 ● 1，表示倾向于作为Decode实例。 ● 2，表示只允许作为Prefill实例。 ● 3，表示只允许作为Decode实例。 当前版本仅支持2和3标签。
max_output_len	size_t	最大输出长度。
max_seq_len	size_t	最大序列长度。
role	枚举类型	Server节点的身份： ● 85，表示角色未知。 ● 80，表示该Server节点为P节点。 ● 68，表示该Server节点为D节点。
total_block_num	size_t	Server节点的block总量。

参数	类型	描述
total_slots_num	size_t	Server节点的slot总量。

环境变量

当前支持的环境变量如下所示。

表 8-29 Controller 支持的环境变量

环境变量名称	含义
MINDIE_MS_CONTROLLER_CONFIG_FILE_PATH	ms_controller配置文件的读取路径。
GLOBAL_RANK_TABLE_FILE_PATH	集群信息文件的读取路径。集群信息文件请参见**步骤9**中的global_ranktable.json文件。 环境变量的优先级高于ms_controller配置文件的global_rank_table_file_path属性。
POD_IP	ms_controller所在Pod的IP。 环境变量的优先级高于ms_controller配置文件的http_server.ip属性。
MINDIE_MS_P_RATE	PD分离部署模式下，P的比例。 ● 0：表示自动进行决策最佳比例，D的比例需要同时为0； ● 非0：表示指定P的比例，D的比例需要同时非0。 环境变量的优先级高于ms_controller配置文件的default_p_rate属性。
MINDIE_MS_D_RATE	PD分离部署模式下，D的比例。 ● 0：表示自动进行决策最佳比例，P的比例需要同时为0； ● 非0：表示指定D的比例，P的比例需要同时非0。 环境变量的优先级高于ms_controller配置文件的default_d_rate属性。

环境变量名称	含义
MINDIEMS_LOG_LEVEL	用户可动态设置Controller输出的日志等级。 环境变量的优先级高于中log_level参数。 用户可动态设置Controller输出的日志等级。 默认值为空，环境变量的优先级高于表8-26中log_level参数。日志级别如下所示： • CRITICAL • ERROR • WARNING • INFO • DEBUG

模型配置文件

模型配置文件信息如下所示。

表 8-30 模型配置文件

参数名称	含义	配置值
hidden_size	隐藏层的特征维度。	8129
initializer_range	权重参数初始化的范围。	0.02
intermediate_size	FFN隐藏层大小。	28672
max_position_embeddings	最大位置嵌入数量。	4096
num_attention_heads	attention头数量。	64
num_hidden_layers	隐藏层数。	80
num_key_value_heads	kv头数量。	8
torch_dtype	PyTorch所用数据类型。	float64

硬件设备信息文件

硬件设备信息文件信息如下所示。

表 8-31 硬件设备信息文件

参数名称	含义	配置值
BW_GB	节点内卡间通信带宽。	392
BW_RDMA_Gb	节点间通信带宽。	200
BWeff	节点内通信带宽效率。	0.5
BW_RDMAeff	节点间通信带宽效率。	0.5
TFLOPS	单卡算力。	246
TFLOPSeff	算力效率。	0.5
MBW_TB	NPU访存带宽。	0.8
MBW_TBeff	NPU访存带宽效率。	0.3
alpha	集合通信启动时延。	2
MEMCapacity	显存GB。	32
eta_OOM	显存OOM水线。	0.9
staticTransferDelay	节点间通信静态时延。	0.00001

说明

其他配置要求：

- ms_controller运行时需要依赖对应lib目录下的so，主要涉及：libboundscheck.so、libcrypto.so.3、libhse_cryption.so、libmie_role_manager.so和libssl.so.3。
- export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$MIES_INSTALL_PATH/lib（该目录需要替换为对应依赖的so的路径）
- 开启tls校验或者将日志写入日志文件时，需要导入以下KMC依赖的环境变量。
  ```
  export HSECEASY_PATH=$MIES_INSTALL_PATH/lib
  ```
- 设置输出的日志等级，例如将日志等级设置为INFO。
  ```
  export MINDIEMS_LOG_LEVEL=INFO
  ```

8.3.3.4 RESTful 接口 API

8.3.3.4.1 启动状态查询接口

接口功能

查询服务启动状态。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/startup

说明

- {ip}优先取环境变量POD_IP的值，如果没有配置该环境变量，则取配置文件ms_controller.json的“http_server”中的“ip”参数。
- {port}取配置文件ms_controller.json的“http_server”中的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/startup
```

响应样例：

```
{
   "message":"",
   "status":"0"
}
```

输出说明

- 状态码200，表示服务已启动。
- 无响应，表示服务未启动。

8.3.3.4.2 健康状态查询接口

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/health

说明

- {ip}优先取环境变量POD_IP的值；如果没有配置该环境变量，则取配置文件ms_controller.json的“http_server”中的“ip”参数。
- {port}取配置文件ms_controller.json的“http_server”中的“port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/health
```

响应样例：

```
{
   "message":"",
   "status":"0"
}
```


输出说明

- 状态码200，表示服务状态正常。
- 无响应，表示服务异常。

8.3.3.5 报错信息查询

ms_controller启动后，运行过程中可能出现的报错信息主要如下所示。

表 8-32 报错信息

报错类型	报错原因	报错信息
系统异常	自身节点通信异常。	与周边组件失联，报出错误日志，级别为错误，关键日志信息如下所示： [HttpClient] Failed to send http request.
	进程异常退出。	部署平台通过健康探针，识别该异常退出，实现进程重启，业务恢复。关键日志信息如下所示： [NodeScheduler]Run: end
节点异常	部分节点无法获得实时状态。	• 无法获得状态的节点，不会被纳入可调度节点。 • 节点异常，报出错误日志。关键日志信息如下所示： [ServerRequestHandler]GetNodeStatus: send request failed, node id xxx, ip xxx, port xxx, ret code xxx, request ret %d • 集群服务状态输出文件中，可观测到该节点异常。该节点的is_healthy字段为false。
	部分节点身份下发失败。	• 下发身份失败的节点，不会被纳入可调度节点。 • 节点异常，报出错误日志。关键日志信息如下所示： [NodeScheduler]PostSingleRole: request failed, node id xxx, ip xxx, port xxx, ret code xxx, request ret xxx • 集群服务状态中可观测到该节点异常，该节点的is_healthy字段为false。
	某个Group内可用节点少于1P1D。	Group不可用，报出错误日志，关键日志信息如下所示： [NodeScheduler]SendRole: all p node failed in group xxx [NodeScheduler]SendRole: all d node failed in group xxx

8.3.4 调度器（Coordinator）

8.3.4.1 功能介绍

MindIE MS调度器（Coordinator）作为集群的数据面入口，主要提供负载均衡的调度算法和Cache亲和的调度算法。MindIE MS调度器（Coordinator）作为用户的推理请求入口，基于当前集群的节点现状和配置的调度算法，实现最优节点的选择、请求的监听、转发等，提升集群场景的节点资源利用率，其架构图如图8-6所示。

图 8-6 调度器（Coordinator）架构图

8.3.4.2 安装部署

环境准备

使用MindIE MS调度器（Coordinator）之前，需要完成以下环境准备。

1. 已参见**3.3 准备MindIE Server镜像**完成MindIE Server镜像制作。
2. 参见 **准备TLS证书**提前准备好证书，将证书路径配置到**ms_coordinator.json启动配置文件**对应证书路径。
3. 参见**3.6.3.2.3 使用kubectl部署PD分离服务示例**完成部署。

8.3.4.3 配置说明

ms_coordinator.json 启动配置文件

ms_coordinator.json启动配置文件样例如下所示，参数解释请参见**ms_coordinator.json启动配置文件参数解释**。

```
{
    "http_config": {
        "predict_ip": "127.0.0.1",
        "predict_port": "1025",
        "manage_ip": "127.0.0.1",
        "manage_port": "1026",
        "server_thread_num": 1,
        "client_thread_num": 1,
        "http_timeout_seconds": 10,
        "keep_alive_seconds": 180,
        "server_name": "MindIE-MS",
        "user_agent": "Coordinator/1.0"
    },
    "request_limit": {
        "single_node_max_requests": 1000,
        "max_requests": 10000
    },
    "metrics_config": {
        "enable": false,
        "trigger_size": 100
    },
    "exception_config": {
        "max_retry": 5,
        "schedule_timeout": 60,
        "first_token_timeout": 60,
```



```
        "infer_timeout": 300,
        "tokenizer_timeout": 300
    },
    "log_info": {
        "log_level": "INFO",
        "to_file": false,
        "to_stdout": true,
        "run_log_path": "./log/run/log.txt",
        "operation_log_path": "./log/operation/log.txt",
        "max_log_str_size": 4096,
        "max_log_file_size": 20,
        "max_log_file_num": 10
    },
    "digs_scheduler_config": {
        "deploy_mode": "pd_separate",
        "scheduler_type": "digs_scheduler",
        "algorithm_type": "load_balance",
        "cache_size": "100",
        "slots_thresh": "0.05",
        "block_thresh": "0.05",
        "max_schedule_count": "10000",
        "reordering_type": "1",
        "max_res_num": "5000",
        "res_limit_rate": "1.1"
    },
    "tls_config": {
        "controller_server_tls_enable": true,
        "controller_server_tls_items": {
            "ca_cert" : "./security/controller/security/certs/ca.pem",
            "tls_cert": "./security/controller/security/certs/cert.pem",
            "tls_key": "./security/controller/security/keys/cert.key.pem",
            "tls_passwd": "./security/controller/security/pass/key_pwd.txt",
            "kmcKsfMaster": "./security/controller/tools/pmt/master/ksfa",
            "kmcKsfStandby": "./security/controller/tools/pmt/standby/ksfb",
            "tls_crl": ""
        },
        "request_server_tls_enable": true,
        "request_server_tls_items": {
            "ca_cert" : "./security/request/security/certs/ca.pem",
            "tls_cert": "./security/request/security/certs/cert.pem",
            "tls_key": "./security/request/security/keys/cert.key.pem",
            "tls_passwd": "./security/request/security/pass/key_pwd.txt",
            "kmcKsfMaster": "./security/request/tools/pmt/master/ksfa",
            "kmcKsfStandby": "./security/request/tools/pmt/standby/ksfb",
            "tls_crl": ""
        },
        "mindie_client_tls_enable": true,
        "mindie_client_tls_items": {
            "ca_cert" : "./security/mindie/security/certs/ca.pem",
            "tls_cert": "./security/mindie/security/certs/cert.pem",
            "tls_key": "./security/mindie/security/keys/cert.key.pem",
            "tls_passwd": "./security/mindie/security/pass/key_pwd.txt",
            "kmcKsfMaster": "./security/mindie/tools/pmt/master/ksfa",
            "kmcKsfStandby": "./security/mindie/tools/pmt/standby/ksfb",
            "tls_crl": ""
        }
    }
}
```


ms_coordinator.json 启动配置文件参数解释

ms_coordinator.json配置文件中各个字段解释如**表8-33**所示，用户可根据具体场景进行配置。

表 8-33 ms_coordinator.json 启动配置文件参数说明

参数名称	支持特性	取值范围	配置说明
http_config：通信配置			
predict_ip	PD分离 Prefix Cache（单机）	-	必填；默认值为"127.0.0.1" 推理IP。 用户侧接口的侦听IP。
predict_port	PD分离 Prefix Cache（单机）	[1024,65535]	必填；默认值为"1025"。 推理端口。 用户侧接口的侦听端口。
manage_ip	PD分离 Prefix Cache（单机）	-	必填；默认值为"127.0.0.1"。 管理IP 集群内通信接口的侦听IP。
manage_port	PD分离 Prefix Cache（单机）	[1024,65535]	必填；默认值为"1026"。 管理端口。 集群内通信接口的侦听端口。
server_thread_num	PD分离 Prefix Cache（单机）	[1,10000]	必填；默认值为1。 HTTP Server线程池数量；建议不超出系统最大线程数的1/4。
client_thread_num	PD分离 Prefix Cache（单机）	[1,10000]	必填；默认值为1。 HTTP Client线程池数量；建议不超出系统最大线程数的1/4。
http_timeout_seconds	PD分离 Prefix Cache（单机）	[0,600]，单位秒。	必填；默认值为10秒。 HTTP通信超时时间。
keep_alive_seconds	PD分离	[0,3600]，单位秒。	必填；默认值为180秒。 与D实例长链接的保活时间。
server_name	PD分离 Prefix Cache（单机）	-	必填；默认值为MindIE-MS。 服务器名称。
user_agent	PD分离 Prefix Cache（单机）	-	必填；默认值为Coordinator/1.0。 软件版本号。

参数名称	支持特性	取值范围	配置说明
request_limit：请求限制			
single_node_max_requests	PD分离 Prefix Cache（单机）	[1,2000]	必填；默认值为1000。 单个节点可处理的最大请求数量；该参数配置的值不能超过MindIE Server能支持的最大限制。 如设置环境变量MINDIE_MS_COORDINATOR_CONFIG_SINGLE_NODE_MAX_REQ，则优先读取环境变量的值。
max_requests	PD分离 Prefix Cache（单机）	[1,90000]	必填；默认值为10000。 可处理的最大请求数量。 ● PD分离场景：建议single_node_max_requests * P节点数量 + 1000左右余量。 ● PD混部场景：建议single_node_max_requests * 节点数量 + 1000左右余量。 如设置环境变量MINDIE_MS_COORDINATOR_CONFIG_MAX_REQ，则优先读取环境变量的值。
metrics_config：性能统计			
enable	PD分离 Prefix Cache（单机）	● true：开启。 ● false：关闭。	必填；默认值为false。 是否开启性能统计。 该参数的功能为辅助定位问题，打开此功能有可能影响业务性能，建议用户在正常业务场景下关闭此功能。
trigger_size	PD分离 Prefix Cache（单机）	[1,10000]	必填；当enable为true时有效。默认值为100。 触发性能统计的请求数。
exception_config：异常配置			
max_retry	PD分离 Prefix Cache（单机）	[0,10]	必填；默认值为5。 通信异常最大重试次数。
schedule_timeout	PD分离 Prefix Cache（单机）	[0,3600]，单位秒。	必填；默认值为60；0表示关闭调度的超时检查。 调度超时时间，请求在调度超时时间内没完成调度，将向用户返回错误。

参数名称	支持特性	取值范围	配置说明
first_token_timeout	PD分离 Prefix Cache（单机）	[0,3600]，单位秒。	必填；默认值为60；0表示关闭首token的超时检查。 首token超时时间，请求在首token的超时时间内没完成首token推理，将向用户返回错误。
infer_timeout	PD分离 Prefix Cache（单机）	[0,3600]，单位秒。	必填；默认值为300；0表示关闭推理的超时检查。 请求推理的超时时间，请求在推理的超时时间内没完成全部推理，将向用户返回错误。
tokenizer_timeout	PD分离	[0,3600]，单位秒。	必填；默认值为300；0表示关闭计算token的超时检查。 计算token的超时时间，请求在计算token的超时时间内没完成tokenizer任务，将向用户返回错误。
log_info：日志配置			
log_level	PD分离 Prefix Cache（单机）	• CRITICAL • ERROR • WARNING • INFO • DEBUG	必填；默认值为INFO。 设置日志级别。 如设置环境变量MINDIEMS_LOG_LEVEL，则优先读取环境变量的值。 **说明** 业务性能受参数影响较大，参数配置为ERROR时性能最佳，配置为DEBUG时性能最差，两者性能相差十倍左右。
to_file	PD分离 Prefix Cache（单机）	• true：输出到文件。 • false：不输出到文件。	必填；默认值为false。 是否输出到文件。
to_stdout	PD分离 Prefix Cache（单机）	• true：输出到标准输出流。 • false：不输出到标准输出流。	必填；默认值为true。 是否输出到标准输出流。
run_log_path	PD分离 Prefix Cache（单机）	-	必填，当to_file为true时生效。 运行日志路径，要求该文件所在的路径真实存在且可读，运行时会自动创建文件。

参数名称	支持特性	取值范围	配置说明
operation_log_path	PD分离 Prefix Cache（单机）	-	必填，当to_file为true时生效。 审计日志路径，要求该文件所在的路径真实存在且可读，运行时会自动创建文件。
max_log_str_size	PD分离 Prefix Cache（单机）	[128,4096]	必填；默认值为4096。 单条日志最大长度。
max_log_file_size	PD分离 Prefix Cache（单机）	[1,100]，单位MB。	必填；默认值为20。 单个日志文件存储上限。
max_log_file_num	PD分离 Prefix Cache（单机）	[2,64]	必填；默认值为10。 最大日志文件存储数量。
digs_scheduler_config：调度器配置			
deploy_mode	PD分离 Prefix Cache（单机）	• PD分离：pd_separate：PD分离模式部署； • Prefix Cache（单机）single_node：单机部署模式。	必填；默认值为"pd_separate"。 部署模式。
scheduler_type	PD分离 Prefix Cache（单机）	• PD分离：digs_scheduler：digs调度器； • Prefix Cache（单机）default_scheduler：默认调度器。	必填；默认值为"digs_scheduler"。 调度器类型。 • deploy_mode为"pd_separate"时，此值必须为"digs_scheduler"； • deploy_mode为"single_node"时，此值必须为"default_scheduler"。

参数名称	支持特性	取值范围	配置说明
algorithm_type	PD分离 Prefix Cache（单机）	• PD分离：load_balance：负载均衡；推理请求调度分配给资源较多的实例。 • Prefix Cache（单机）cache_affinity：Cache亲和算法；OpenAI多轮会话场景下，推理请求调度给处理过历史轮次会话的实例。	必填；默认值为"load_balance"。 调度算法。 • deploy_mode为"pd_separate"时，此值必须为"load_balance"； • deploy_mode为"single_node"时，此值必须为"cache_affinity"。
cache_size	Prefix Cache（单机）	["1", "10000"]	algorithm_type为"cache_affinity"时必填；默认值为"100"。 Cache缓存上限 。（仅支持Prefix Cache算法）
slots_thresh	Prefix Cache（单机）	["0.0", "1.0"]	algorithm_type为"cache_affinity"时必填；默认值为"0.05"。 可用slot占总slot的比例，slots资源预警线 。（仅支持Prefix Cache算法）
block_thresh	Prefix Cache（单机）	["0.0", "1.0"]	algorithm_type为"cache_affinity"时必填；默认值为"0.05"。 可用block占总block的比例，block资源预警线。（仅支持Prefix Cache算法）
max_schedule_count	PD分离	["1", "90000"]	scheduler_type为"digs_scheduler"时必填；默认值为"10000"。 可以同时调度的最大请求数量，建议与max_requests保持一致。（仅支持PD分离负载均衡算法）

参数名称	支持特性	取值范围	配置说明
reordering_type	PD分离	• "1"：fcfs，先到先调度； • "2"：sjf，短序列先调度； • "3"：ljf，长序列先调度。	scheduler_type为"digs_scheduler"时必填；默认值为"1"。
max_res_num	PD分离	["1", "10000"]	scheduler_type为"digs_scheduler"时必填；默认值为"5000"。 调度器可注册的最大节点。
res_limit_rate	PD分离	["1.0","100.0"]	scheduler_type为"digs_scheduler"时必填；默认值为"1.1"。 转换后的资源上限与转换前的比率。（仅支持PD分离负载均衡算法）
tls_config：证书配置			
调度器（Coordinator）的管理端口通信证书配置			
controller_server_tls_enable	PD分离 Prefix Cache（单机）	与MindIE MS Controller的通信，是否开启tls校验。 建议用户打开，确保控制器（Controller）或用户与管理端口通信安全。如果关闭则存在较高的网络安全风险。	必填；默认值为true。
ca_cert	PD分离 Prefix Cache（单机）	ca根证书路径。	开启tls校验时必填。
tls_cert	PD分离 Prefix Cache（单机）	tls证书路径。	开启tls校验时必填。

参数名称	支持特性	取值范围	配置说明
tls_key	PD分离 Prefix Cache（单机）	经口令加密的tls私钥证书路径。	开启tls校验时必填。
tls_passwd	PD分离 Prefix Cache（单机）	加密tls私钥证书的口令，经KMC加密后，落盘的密文路径。	开启tls校验时必填。
kmcKsfMaster	PD分离 Prefix Cache（单机）	KMC加密的根密钥路径。	开启tls校验时必填。
kmcKsfStandby	PD分离 Prefix Cache（单机）	KMC加密的工作密钥路径。	开启tls校验时必填。
tls_crl	PD分离 Prefix Cache（单机）	吊销的证书列表路径。	开启tls校验时必填。 证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。
调度器（Coordinator）的数据端口与用户的通信证书配置			
request_server_tls_enable	PD分离 Prefix Cache（单机）	接受推理请求的输入，通信是否开启tls校验。 建议用户打开，确保与推理用户的通信安全。如果关闭则存在较高的网络安全风险。	必填；默认值为true。
ca_cert	PD分离 Prefix Cache（单机）	ca根证书路径。	开启tls校验时必填。
tls_cert	PD分离 Prefix Cache（单机）	tls证书路径。	开启tls校验时必填。

参数名称	支持特性	取值范围	配置说明
tls_key	PD分离 Prefix Cache（单机）	经口令加密的tls私钥证书路径。	开启tls校验时必填。
tls_passwd	PD分离 Prefix Cache（单机）	加密tls私钥证书的口令，经KMC加密后，写入的密文路径。	开启tls校验时必填。
kmcKsfMaster	PD分离 Prefix Cache（单机）	KMC加密的根密钥路径。	开启tls校验时必填。
kmcKsfStandby	PD分离 Prefix Cache（单机）	KMC加密的工作密钥路径。	开启tls校验时必填。
tls_crl	PD分离 Prefix Cache（单机）	吊销的证书列表路径。	开启tls校验时必填。 证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。
调度器（Coordinator）与MindIE Server数据端口的通信证书配置			
mindie_client_tls_enable	PD分离 Prefix Cache（单机）	与MindIE Server的通信，是否开启tls校验。	必填；默认值为true。
ca_cert	PD分离 Prefix Cache（单机）	ca根证书路径。	开启tls校验时必填。
tls_cert	PD分离 Prefix Cache（单机）	tls证书路径。	开启tls校验时必填。
tls_key	PD分离 Prefix Cache（单机）	经口令加密的tls私钥证书路径。	开启tls校验时必填。

参数名称	支持特性	取值范围	配置说明
tls_passwd	PD分离 Prefix Cache（单机）	加密tls私钥证书的口令，经KMC加密后，写入的密文路径。	开启tls校验时必填。
kmcKsfMaster	PD分离 Prefix Cache（单机）	KMC加密的根密钥路径。	开启tls校验时必填。
kmcKsfStandby	PD分离 Prefix Cache（单机）	KMC加密的工作密钥路径。	开启tls校验时必填。
tls_crl	PD分离 Prefix Cache（单机）	吊销的证书列表路径。	开启tls校验时必填。 证书吊销列表crl文件路径，要求该文件真实存在且可读。如为空，则不进行吊销校验。

环境变量

当前**调度器（Coordinator）**支持的环境变量如下所示。

表 8-34 Coordinator 支持的环境变量

环境变量名称	含义
MINDIE_MS_COORDINATOR_CONFIG_FILE_PATH	ms_coordinator配置文件的读取路径。
MINDIE_MS_COORDINATOR_CONFIG_SINGLE_NODE_MAX_REQ	单个节点可处理的最大请求数量。
MINDIE_MS_COORDINATOR_CONFIG_MAX_REQ	可处理的最大请求数量。

环境变量名称	含义
MINDIEMS_LOG_LEVEL	用户可动态设置MindIE MS客户端输出的日志等级。 默认值为空，环境变量的优先级高于**表8-33**中log_level参数。日志级别如下所示： • DEBUG • INFO • WARNING • ERROR • CRITICAL

说明

额外配置要求：

- ms_coordinator运行时需要依赖对应lib目录下的so，主要涉及：libboundscheck.so、libcrypto.so、libhse_cryption.so、libssl.so和libmie_digs.so。
- export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$MIES_INSTALL_PATH/lib（该目录需要替换为对应依赖的so的路径）
- 开启tls校验或者将日志写入日志文件时，需要导入以下KMC依赖的环境变量。
  ```
  export HSECEASY_PATH=$MIES_INSTALL_PATH/lib
  ```
- 设置输出的日志等级，例如将日志等级设置为INFO。
  ```
  export MINDIEMS_LOG_LEVEL=INFO
  ```

8.3.4.4 启动调度器

调度器的执行程序为ms_coordinator，其存放路径为：*{MindIE安装目录}*/latest/mindie-service/bin/，启动ms_coordinator时需要读取**ms_coordinator.json启动配置文件**中的配置信息，且运行ms_coordinator时需要依赖*{MindIE安装目录}*/latest/mindie-service/lib/目录下的so文件，包括libboundscheck.so、libcrypto.so.3、libhse_cryption.so、libssl.so.3和libmie_digs.so文件。

前提条件

启动ms_coordinator需提前配置好以下环境变量。

- 设置链接库的路径：
  ```
  export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:${MindIE安装目录}/latest/mindie-service/lib
  ```
- 设置KMC解密工具的依赖库路径：
  ```
  export HSECEASY_PATH=${MindIE安装目录}/latest/mindie-service/lib
  ```
- 设置ms_coordinator.json配置文件的路径：
  ```
  export MINDIE_MS_COORDINATOR_CONFIG_FILE_PATH=${MindIE安装目录}/latest/mindie-service/conf/ms_coordinator.json
  ```

启动命令

ms_coordinator有两种启动方式，且命令必须在*{MindIE安装目录}*/latest/mindie-service/bin/目录中执行。

- 方式一：直接启动
  ```
  ./ms_coordinator
  ```
- 方式二：带参数启动
  ```
  ./ms_coordinator {predict_ip} {predict_port} {manage_ip} {manage_port}
  ```
  参数解释：
  - *{predict_ip}*：如果配置，则将取代ms_coordinator.json启动配置文件中的predict_ip参数；
  - *{predict_port}*：如果配置，则将取代ms_coordinator.json启动配置文件中的predict_port参数;
  - *{manage_ip}*：如果配置，则将取代ms_coordinator.json启动配置文件中的manage_ip参数；
  - *{manage_port}*：如果配置，则将取代ms_coordinator.json启动配置文件中的manage_port参数。

8.3.4.5 停止调度器

两种停止调度器的方式如下所示。

- 方式一（推荐）：使用后台进程方式启动服务。
  - 使用kill命令停止进程。
    ```
    kill {ms_coordinator 进程ID}
    ```
    说明

    Linux系统中查询ms_coordinator主进程ID：
    1. 查看所有与ms_coordinator相关的进程列表。
       ```
       ps -ef | grep ms_coordinator
       ```
    2. 在输出结果中找到PID列，PID即为ms_coordinator的主进程ID。
  - 或使用pkill命令停止进程。
    ```
    pkill -9 ms_coordinator
    ```
- 方式二：以直接启动进程方式启动服务， 可以通过直接按ctrl+c组合键停止服务。

8.3.4.6 RESTful 接口 API

8.3.4.6.1 说明

说明

- 调度器接收的单条请求体最大规格不得大于1MB。
- 配置ulimit值，建议配置为：3*最大并发请求数。
  1. 用户需要使用以下命令查看环境中ulimit值的上限：
     ```
     ulimit -n
     ```
  2. 使用以下命令配置ulimit值：

     例如：最大并发请求数为500，其值建议配置为3*500=1500。
     ```
     ulimit -n 1500
     ```

8.3.4.6.2 用户侧接口

TGI 流式推理接口

接口功能

提供流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate_stream

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片

参数	是否必选	说明	取值要求
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。	bool类型，默认值false。 对于流式接口，该参数只能为false。
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“details”=true，即返回所有的details信息。	bool类型参数，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。

参数	是否必选	说明	取值要求
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true表示添加。 ● false表示不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json或者padded_vocab_size的文件中读取的vocab_size值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认值使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate_stream
```

请求消息体：

- 单模态文本模型：

```
{
   "inputs": "My name is Olivier and I",
   "parameters": {
      "decoder_input_details": false,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": null,
      "adapter_id": "None"
   }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
```



```
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "parameters": {
        "decoder_input_details": false,
        "details": true,
        "do_sample": true,
        "max_new_tokens": 20,
        "repetition_penalty": 1.03,
        "return_full_text": false,
        "seed": null,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "truncate": null,
        "typical_p": 0.5,
        "watermark": false,
        "stop": null,
        "adapter_id": "None"
    }
}
```

响应样例：

- 响应样例1（使用sse格式返回）：

```
data: {"token":{"id":13,"text":"\n","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":26626,"text":"Jan","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":300,"text":"et","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3732,"text":"
makes","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":395,"text":" $","logprob":null,"special":null},"generated_text":null,"details":null}

......

data: {"token":{"id":395,"text":" $","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29896,"text":"1","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29896,"text":"1","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29947,"text":"8","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":29889,"text":".","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2,"text":"","logprob":null,"special":null},"generated_text":"\nJanet makes $104 a
day at the farmers' market.\nThe number of eggs that Janet sells at the farmers' market each day is
16 - 3 - 4 = 7.\nShe makes $2 for each egg that she sells.\nThe total amount that she makes is $2 * 7
= $14.\nThe total amount that she makes is $14 + $104 = $118.\nThe total amount that she makes is
$118.","details":
{"prompt_tokens":74,"finish_reason":"eos_token","generated_tokens":116,"seed":875672700412924893
1}}
```

- 响应样例2（配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data: {"token":{"id":198,"text":"\n","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":9707,"text":"\nHello","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":0,"text":"\nHello!","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2585,"text":"\nHello!
How","logprob":null,"special":null},"generated_text":null,"details":null}
```



```
data: {"token":{"id":646,"text":"\nHello! How
can","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":358,"text":"\nHello! How can
I","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":7789,"text":"\nHello! How can I
assist","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":498,"text":"\nHello! How can I assist
you","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3351,"text":"\nHello! How can I assist you
today","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":30,"text":"\nHello! How can I assist you
today?","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":151645,"text":"\nHello! How can I assist you
today?","logprob":null,"special":null},"generated_text":"\nHello! How can I assist you today?","details":
{"prompt_tokens":1,"finish_reason":"eos_token","generated_tokens":11,"seed":2296576927}}
```

输出说明

返回值	类型	说明
data	object	一次推理返回的结果。
token	object	每一次推理的token。
id	int	token ID。
text	string	token对应文本。
logprob	概率对数	当前不支持，默认返回null。
special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
generated_text	string	推理文本返回结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，并且请求参数“details”=true才返回details结果。
prompt_tokens	int	用户输入的prompt文本对应的token长度。

返回值	类型	说明
finish_reason	string	结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。

TGI 文本推理接口

接口功能

提供文本推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于1MB、maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。	bool类型，默认值false。

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“decoder_input_details”和“details”字段任意一个为true，即返回所有的details信息。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true表示添加。 ● false表示不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate
```

请求消息体：

- 单模态文本模型：

```
{
   "inputs": "My name is Olivier and I",
   "parameters": {
      "decoder_input_details": true,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": null,
      "adapter_id": "None"
   }
}
```

- 多模态模式：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
```



```
            {"type": "text", "text": "My name is Olivier and I"},
            {
                "type": "image_url",
                "image_url": "/xxxx/test.png"
            }
        ],
        "parameters": {
            "decoder_input_details": true,
            "details": true,
            "do_sample": true,
            "max_new_tokens": 20,
            "repetition_penalty": 1.03,
            "return_full_text": false,
            "seed": null,
            "temperature": 0.5,
            "top_k": 10,
            "top_p": 0.95,
            "truncate": null,
            "typical_p": 0.5,
            "watermark": false,
            "stop": null,
            "adapter_id": "None"
        }
    }
```

响应样例：

```
{
   "details": {
      "finish_reason": "length",
      "generated_tokens": 1,
      "prefill": [{
         "id": 0,
         "logprob":null,
         "special": null,
         "text": "test"
      }],
      "prompt_tokens": 74,
      "seed": 42,
      "tokens": [{
         "id": 0,
         "logprob": null,
         "special": null,
         "text": "test"
      }]
   },
   "generated_text": "am a Frenchman living in the UK. I have been working as an IT consultant for "
}
```

输出说明

返回值	类型	说明
details	object	推理details结果，请求参数“decoder_input_details”和“details”任意一个字段为true，即返回details结果。

返回值	类型	说明
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
prefill	List[token]	请求参数“decoder_input_details”=true，返回推理请求文本detokenizer之后的token，默认为空列表。
prefill.id	int	token ID。
prefill.logprob	float	概率对数，可以为空（第一个token概率值不能被计算获得）。当前不支持，默认返回null。
prefill.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
prefill.text	string	token对应文本。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。
tokens	List[token]	返回推理结果的所有tokens。
tokens.id	int	token ID。
tokens.logprob	概率对数	概率对数。当前不支持，默认返回null。

返回值	类型	说明
tokens.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
tokens.text	string	token对应文本。
generated_text	string	推理返回结果。

TGI 文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
decoder_input_details	可选	是否返回推理请求文本的token ID。如果“stream”=true，该参数不能为true。	bool类型，默认值false。

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。根据TGI 0.9.4接口行为，“decoder_input_details”和“details”字段任意一个为true，即返回所有的details信息。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0, 2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
return_full_text	可选	是否将推理请求文本（inputs参数）添加到推理结果前面。	bool类型，默认值false。 ● true：添加。 ● false：不添加。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
truncate	可选	输入文本做tokenizer之后，将token数量截断到该长度。读取截断后的n个token。若该字段值大于或等于token数量，则该字段无效。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认使用0来表示不进行该项处理，但是不可主动设置为0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型。 ● List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 ● string类型长度范围为1~1024。 默认值null。
adapter_id	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。由字母、数字、点、中划线、下划线和正斜杠组成，字符串长度小于或等于256。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 ● true：流式推理。 ● false：文本推理。

使用样例

请求样例：

```
POST https://{ip}:{port}/
```

请求消息体：

- 单模态文本模型：

```
{
    "inputs": "My name is Olivier and I",
    "parameters": {
        "decoder_input_details": true,
        "details": true,
        "do_sample": true,
        "max_new_tokens": 20,
        "repetition_penalty": 1.03,
        "return_full_text": false,
        "seed": null,
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "truncate": null,
        "typical_p": 0.5,
        "watermark": false,
        "stop": ["stop1", "stop2"],
        "adapter_id": "None"
    },
    "stream": false
}
```


- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "inputs": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
          "type": "image_url",
          "image_url": "/xxxx/test.png"
      }
   ],
   "parameters": {
      "decoder_input_details": true,
      "details": true,
      "do_sample": true,
      "max_new_tokens": 20,
      "repetition_penalty": 1.03,
      "return_full_text": false,
      "seed": null,
      "temperature": 0.5,
      "top_k": 10,
      "top_p": 0.95,
      "truncate": null,
      "typical_p": 0.5,
      "watermark": false,
      "stop": ["stop1", "stop2"],
      "adapter_id": "None"
   },
   "stream": false
}
```

响应样例：

- 文本推理（“stream”=false）：

```
[
   {
      "details": {
         "prompt_tokens": 6,
         "finish_reason": "length",
         "generated_tokens": 20,
         "prefill": [
            {
               "id": 5050,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 829,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 374,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 77018,
               "logprob": null,
               "special": null,
               "text": null
            },
            {
               "id": 323,
```



```
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 358,
            "logprob": null,
            "special": null,
            "text": null
        }
    ],
    "seed": 789070824,
    "tokens": [
        {
            "id": 2776,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 264,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 3162,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 23576,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 504,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 9625,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 13,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 358,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 2948,
            "logprob": null,
            "special": null,
            "text": null
        },
        {
            "id": 311,
```



```
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 1936,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 2513,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 11,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 5310,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 3482,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 8357,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 323,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 6371,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 10500,
            "logprob": null,
            "special": null,
            "text": null
         },
         {
            "id": 13,
            "logprob": null,
            "special": null,
            "text": null
         }
      ]
   },
   "generated_text": "'m a software engineer from France. I love to build things, especially web
applications and mobile apps."
```



```
  }
]
```

- 流式推理：
  - 流式推理1（“stream”=true，“decoder_input_details”=false，使用sse格式返回）：

```
data: {"token":
{"id":29915,"text":"'","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29885,"text":"m","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":"
a","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":5176,"text":"
French","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1410,"text":"
gu","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29891,"text":"y","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1058,"text":"
who","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":338,"text":"
is","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":3063,"text":"
looking","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":363,"text":"
for","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":"
a","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":2058,"text":"
place","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":304,"text":"
to","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":5735,"text":"
live","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":297,"text":"
in","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29889,"text":".","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":306,"text":"
I","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29915,"text":"'","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":
{"id":29885,"text":"m","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":263,"text":" a","logprob":null,"special":null},"generated_text":"'m a French
guy who is looking for a place to live in. I'm a","details":
{"prompt_tokens":8,"finish_reason":"length","generated_tokens":20,"seed":218884523}}
```

  - 流式推理2（“stream”=true，“decoder_input_details”=false，配置项“fullTextEnabled”=true，使用sse格式返回）：


```
data: {"token":{"id":11,"text":",","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":600,"text":",
i","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":1184,"text":", i
need","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":498,"text":", i need
you","logprob":null,"special":null},"generated_text":null,"details":null}

data: {"token":{"id":151645,"text":", i need you","logprob":null,"special":null},"generated_text":",
i need you","details":
{"prompt_tokens":1,"finish_reason":"eos_token","generated_tokens":5,"seed":1621189503}}
```

输出说明

表 8-35 文本推理结果说明

返回值	类型	说明
details	object	推理details结果，请求参数“decoder_input_details”和“details”任意一个字段为true，即返回details结果。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
prefill	List[token]	请求参数“decoder_input_details”=true，返回推理请求文本detokenizer之后的token，默认为空列表。

返回值	类型	说明
prefill.id	int	token ID。
prefill.logprob	float	概率对数，可以为空（第一个token概率值不能被计算获得）。当前不支持，默认返回null。
prefill.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
prefill.text	string	token对应文本。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。
tokens	List[token]	返回推理结果的所有tokens。
tokens.id	int	token ID。
tokens.logprob	概率对数	概率对数。当前不支持，默认返回null。
tokens.special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
tokens.text	string	token对应文本。
generated_text	string	推理返回结果。

表 8-36 流式推理结果说明

返回值	类型	说明
data	object	一次推理返回的结果。
token	object	每一次推理的token。
id	int	token ID。
text	string	token对应文本。
logprob	概率对数	当前不支持，默认返回null。
special	bool	表明该token是否是special，如果是“special”=true，该token在做连接的时候可以被忽略。当前不支持，默认返回null。
generated_text	string	推理文本返回结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，并且请求参数“details”=true才返回details结果。

返回值	类型	说明
prompt_tokens	int	用户输入的prompt文本对应的token长度。
finish_reason	string	结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	返回推理请求的seed值，如果请求参数没有指定seed参数，则返回系统随机生成的seed值。

vLLM 文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/generate

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
prompt	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。

参数	是否必选	说明	取值要求
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
max_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_tokens)。	int类型，取值范围(0，2147483647]，默认值为MindIE Server配置文件中的maxIterTimes。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，取值范围(0.0, 2.0]，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。
presence_penalty	可选	存在惩罚介于-2.0和2.0之间，它影响模型如何根据到目前为止是否出现在文本中来惩罚新token。正值将通过惩罚已经使用的词，增加模型谈论新主题的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
frequency_penalty	可选	频率惩罚介于-2.0和2.0之间，它影响模型如何根据文本中词汇的现有频率惩罚新词汇。正值将通过惩罚已经频繁使用的词来降低模型一行中重复用词的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。1.0表示不进行计算，大于1.0表示输出随机性提高。temperature=0.0，即采用greedy sampling。	float类型，取值大于或等于0.0。 取0.0时将忽略其他后处理参数做greedysearch。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。-1表示不进行top k计算。	uint32_t类型，取值范围-1或者(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 若传-1，-1会变为0传递给MindIE LLM后端，MindIE LLM后端会当做词表大小来处理。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 ● true：流式推理。 ● false：文本推理。

参数	是否必选	说明	取值要求
stop	可选	停止推理的文本。输出结果中默认不包含停止词列表文本。	List[string]类型或者string类型，默认为null。 ● List[string]类型，每个元素字符长度大于或等于1，列表元素总长度不超过32768（32*1024）。列表为空时相当于null。 ● string类型长度范围为1~32768。
stop_token_ids	可选	停止推理的token ID列表。输出结果中默认不包含停止推理列表中的token ID。	List[int32]类型，超出int32的元素将会被忽略。 默认为null。
model	可选	指定推理时使用的Lora权重，即loraId。	string类型，默认值"None"。
include_stop_str_in_output	可选	决定是否在生成的推理文本中包含停止字符串。	bool类型，默认值为false。**PD场景暂不支持此参数**。 ● true：包含停止字符串。 ● false：不包含停止字符串。 不传入stop或stop_token_ids时，此字段会被忽略。
skip_special_tokens	可选	指定在推理生成的文本中是否跳过特殊tokens。	bool类型，默认值为true。 ● true：跳过特殊tokens。 ● false：保留特殊tokens。
ignore_eos	可选	指定在推理文本生成过程中是否忽略eos_token结束符	bool类型，默认值为false。 ● true：忽略eos_token结束符。 ● false：不忽略eos_token结束符。

使用样例

请求样例：

```
POST https://{ip}:{port}/generate
```

请求消息体：

- 单模态文本模型：


```
{
   "prompt": "My name is Olivier and I",
   "max_tokens": 20,
   "repetition_penalty": 1.03,
   "presence_penalty": 1.2,
   "frequency_penalty": 1.2,
   "temperature": 0.5,
   "top_p": 0.95,
   "top_k": 10,
   "seed": null,
   "stream": false,
   "stop": null,
   "stop_token_ids": null,
   "model": "None",
   "include_stop_str_in_output": false,
   "skip_special_tokens": true,
   "ignore_eos": false
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "prompt": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
          "type": "image_url",
          "image_url": "/xxxx/test.png"
      }
   ],
   "max_tokens": 20,
   "repetition_penalty": 1.03,
   "presence_penalty": 1.2,
   "frequency_penalty": 1.2,
   "temperature": 0.5,
   "top_p": 0.95,
   "top_k": 10,
   "seed": null,
   "stream": false,
   "stop": null,
   "stop_token_ids": null,
   "model": "None",
   "include_stop_str_in_output": false,
   "skip_special_tokens": true,
   "ignore_eos": false
}
```

响应样例：

- 文本推理（“stream”=false）：

```
{"text":["My name is Olivier and I am a Frenchman living in the UK. I am a keen photographer and"]}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

    ```
    {"text":["am"]}{"text":[" a"]}{"text":[" French"]}{"text":["man"]}{"text":[" living"]}{"text":[" in"]}"text":[" the"]}{"text":[" UK"]}{"text":["."]}{"text":[" I"]} {"text":[" am"]}{"text":[" a"]}"text":[" keen"]}{"text":[" photograph"]}{"text":["er"]}{"text":[" and"]}
    ```

  - 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

    ```
    {"text":[" to"]}{"text":[" to travel"]}{"text":[" to travel."]}{"text":[" to travel. I"]}{"text":[" to travel. I'm"]}
    ```


输出说明

返回值	类型	说明
text	string	推理返回结果。

须知

vLLM流式返回结果，每个token的返回结果添加'\0'字符做分割。使用**curl**命令发送vLLM流式推理请求的样例如下：

```
curl -H "Accept: application/json" -H "Content-type: application/json" --cacert /home/runs/static_conf/ca/ca.pem  --cert /home/runs/static_conf/cert/client.pem  --key /home/runs/static_conf/cert/client.key.pem -X POST -d '{
"prompt": "My name is Olivier and I",
"stream": true,
"repetition_penalty": 1.0,
"top_p": 1.0,
"top_k": 10,
"max_tokens": 16,
"temperature": 1.0
}' https://{ip}:{port}/generate | cat
```

OpenAI 推理接口

须知

- 运行环境的transformers版本不可低于4.34.1，低版本tokenizer不支持"chat_template"方法。
- 推理模型权重路径下的tokenizer_config.json需要包含"chat_template"字段及其实现。
- function call功能的相关参数tool_call_id、tool_calls、tools和tool_choice当前仅支持部分模型，使用其他模型可能会报错。目前支持的模型只有ChatGLM3-6B。

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v1/chat/completions

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。

请求参数

参数	是否必选	说明	取值要求
model	必选	模型名。	与MindIE Server配置文件中modelName的取值保持一致。
messages	必选	推理请求消息结构。	list类型，0KB<messages内容包含的字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
role	必选	推理请求消息角色。	字符串类型，可取角色有： ● system：系统角色 ● user：用户角色 ● assistant：助手角色 ● tool：工具角色
content	必选	推理请求文本。	字符串类型。 ● 当role为assistant，且tool_calls非空时，content可以不传，其余角色非空。 ● 其余情况content非空。
tool_calls	可选	模型生成的工具调用。	类型为List[dict]，当role为assistant时，表示模型对工具的调用。
tool_calls.function	必选	表示模型调用的工具。	dict类型。 ● arguments，必选，使用JSON格式的字符串，表示调用函数的参数。 ● name，必选，字符串，调用的函数名。
tool_calls.id	必选	表示模型某次工具调用的ID。	字符串。
tool_calls.type	必选	调用的工具类型。	字符串，仅支持"function"。

参数	是否必选	说明	取值要求
tool_call_id	当role为tool时必选，否则可选	关联模型某次调用工具时的ID。	字符串。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型参数，默认值false。 ● true：流式推理。 ● false：文本推理。
presence_penalty	可选	存在惩罚介于-2.0和2.0之间，它影响模型如何根据到目前为止是否出现在文本中来惩罚新token。正值将通过惩罚已经使用的词，增加模型谈论新主题的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
frequency_penalty	可选	频率惩罚介于-2.0和2.0之间，它影响模型如何根据文本中词汇的现有频率惩罚新词汇。正值将通过惩罚已经频繁使用的词来降低模型一行中重复用词的可能性。	float类型，取值范围[-2.0, 2.0]，默认值0.0。
repetition_penalty	可选	重复惩罚是一种技术，用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，取值范围(0.0, 2.0]，默认值1.0。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，取值范围[0.0, 2.0]，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围[0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
stop	可选	停止推理的文本。输出结果默认不包含停止词列表文本。	List[string]类型或者string类型，默认值null。 • List[string]类型列表元素不超过1024个，每个元素的长度为1~1024，列表元素总长度不超过32768（256*128）。列表为空时相当于null。 • string类型长度范围为1~1024。
stop_token_ids	可选	停止推理的token ID列表。输出结果默认不包含停止推理列表中的token ID。	List[int32]类型，超出int32的元素将会被忽略，默认值null。

参数	是否必选	说明	取值要求
include_stop_str_in_output	可选	决定是否在生成的推理文本中包含停止字符串。	bool类型，默认值false。**PD场景暂不支持此参数**。 ● true：包含停止字符串。 ● false：不包含停止字符串。 不传入stop或stop_token_ids时，此字段会被忽略。
skip_special_tokens	可选	指定在推理生成的文本中是否跳过特殊tokens。	bool类型，默认值true。 ● true：跳过特殊tokens。 ● false：保留特殊tokens。
ignore_eos	可选	指定在推理文本生成过程中是否忽略eos_token结束符。	bool类型，默认值false。 ● true：忽略eos_token结束符。 ● false：不忽略eos_token结束符。
max_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_tokens)。	int类型，取值范围(0，2147483647]，默认值maxIterTimes。
tools	可选	可能会使用的工具列表。	List[dict]类型。
tools.type	必选	说明工具类型。	仅支持字符串"function"。
tools.function	必选	函数描述。	dict类型。
function.name	必选	函数名称。	字符串。
function.strict	可选	表示生成tool calls是否严格遵循schema格式。	bool类型，默认false。
function.description	可选	描述函数功能和使用。	字符串。
function.parameters	可选	表示函数接受的参数。	JSON schema格式。
parameters.type	必选	表示函数参数属性的类型。	字符串，仅支持object。

参数	是否必选	说明	取值要求
parameters.properties	必选	函数参数的属性。每一个key表示一个参数名，由用户自定义。value为dict类型，表示参数描述，包含type和description两个参数。	dict类型。
function.required	必选	表示函数必填参数列表。	List[string]类型。
function.additionalProperties	可选	是否允许使用未提及的额外参数。	bool类型，默认值false。 ● true：允许使用未提及的额外参数。 ● false：不允许使用未提及的额外参数。
tool_choice	可选	控制模型调用工具。	string类型或者dict类型，可以为null，默认值"auto"。 ● "none"：表示模型不会调用任何工具，而是生成一条消息。 ● "auto"：表示模型可以生成消息或调用一个或多个工具。 ● "required"：表示模型必须调用一个或多个工具。 通过{"type": "function", "function": {"name": "my_function"}}指定特定的工具，将强制模型调用该工具。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/chat/completions
```

请求消息体：

- 单轮对话：

```
{
    "model": "gpt-3.5-turbo",
    "messages": [{
        "role": "user",
        "content": "You are a helpful assistant."
    }],
    "stream": false,
    "presence_penalty": 1.03,
```



```
    "frequency_penalty": 1.0,
    "repetition_penalty": 1.0,
    "temperature": 0.5,
    "top_p": 0.95,
    "top_k": 0,
    "seed": null,
    "stop": ["stop1", "stop2"],
    "stop_token_ids": [2, 13],
    "include_stop_str_in_output": false,
    "skip_special_tokens": true,
    "ignore_eos": false,
    "max_tokens": 20
}
```

- 多轮对话：
  - 请求样例1：

    ```
    {
        "model": "chatglm3-6b",
        "messages": [{
            "role": "system",
            "content": "You are a helpful customer support assistant. Use the supplied tools to assist the user."
            },
            {
            "role": "user",
            "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
            }
        ],
        "stream": false,
        "presence_penalty": 1.03,
        "frequency_penalty": 1.0,
        "repetition_penalty": 1.0,
        "temperature": 0.5,
        "top_p": 0.95,
        "top_k": -1,
        "seed": null,
        "stop": ["stop1", "stop2"],
        "stop_token_ids": [2],
        "ignore_eos": false,
        "max_tokens": 1024,
        "tools": [
            {
                "type": "function",
                "function": {
                    "name": "get_delivery_date",
                    "strict": true,
                    "description": "Get the delivery date for a customer's order. Call this whenever you need to know the delivery date, for example when a customer asks 'Where is my package'",
                    "parameters": {
                        "type": "object",
                        "properties": {
                            "order_id": {
                                "type": "string",
                                "description": "The customer's order ID."
                            }
                        },
                        "required": [
                            "order_id"
                        ],
                        "additionalProperties": false
                    }
                }
            }
        ],
        "tool_choice": "auto"
    }
    ```

  - 请求样例2：

    ```
    {
        "model": "chatglm3-6b",
    ```



```
    "messages": [
        {
            "role": "system",
            "content": "You are a helpful customer support assistant. Use the supplied tools to
assist the user."
        },
        {
            "role": "user",
            "content": "Hi, can you tell me the delivery date for my order? my order id is 12345."
        },
        {
            "role": "assistant",
            "tool_calls": [
                {
                    "function": {
                        "arguments": "{\"order_id\": \"12345\"}",
                        "name": "get_delivery_date"
                    },
                    "id": "tool_call_8p2Nk",
                    "type": "function"
                }
            ]
        },
        {
            "role": "tool",
            "content": "the delivery date is 2024.09.10.",
            "tool_call_id": "tool_call_8p2Nk"
        }
    ],
    "stream": false,
    "repetition_penalty": 1.1,
    "temperature": 0.9,
    "top_p": 1,
    "max_tokens": 1024,
    "tools": [
        {
            "type": "function",
            "function": {
                "name": "get_delivery_date",
                "strict": true,
                "description": "Get the delivery date for a customer's order. Call this whenever you
need to know the delivery date, for example when a customer asks 'Where is my package'",
                "parameters": {
                    "type": "object",
                    "properties": {
                        "order_id": {
                            "type": "string",
                            "description": "The customer's order ID."
                        }
                    },
                    "required": [
                        "order_id"
                    ],
                    "additionalProperties": false
                }
            }
        }
    ],
    "tool_choice": "auto"
}
```

响应样例：

- 文本推理（“stream”=false）：
  - 单轮对话：

```
{
    "id": "chatcmpl-123",
    "object": "chat.completion",
    "created": 1677652288,
```



```
  "model": "gpt-3.5-turbo-0613",
  "choices": [
     {
        "index": 0,
        "message": {
           "role": "assistant",
           "content": "\n\nHello there, how may I assist you today?"
        },
        "finish_reason": "stop"
     }
  ],
  "usage": {
     "prompt_tokens": 9,
     "completion_tokens": 12,
     "total_tokens": 21
  }
}
```

- 多轮对话：
  - 响应样例1：

```
{
   "id": "chatcmpl-123",
   "object": "chat.completion",
   "created": 1677652288,
   "model": "chatglm3-6b",
   "choices": [
      {
         "index": 0,
         "message": {
            "role": "assistant",
            "content": "",
            "tool_calls": [
               {
                  "function": {
                     "arguments": "{\"order_id\": \"12345\"}",
                     "name": "get_delivery_date"
                  },
                  "id": "call_JwmTNF3O",
                  "type": "function"
               }
            ]
         },
         "finish_reason": "tool_calls"
      }
   ],
   "usage": {
      "prompt_tokens": 226,
      "completion_tokens": 122,
      "total_tokens": 348
   }
}
```

  - 响应样例2：

```
{
   "id": "endpoint_common_25",
   "object": "chat.completion",
   "created": 1728959154,
   "model": "chatglm3-6b",
   "choices": [
      {
         "index": 0,
         "message": {
            "role": "assistant",
            "content": "\n Your order with ID 12345 is scheduled for delivery on September 10th, 2024.",
            "tool_calls": null
         },
         "finish_reason": "stop"
      }
```



```
    ],
    "usage": {
        "prompt_tokens": 265,
        "completion_tokens": 30,
        "total_tokens": 295
    }
}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

```
data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"\t"},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"\t"},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"llama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}
```

```
data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"lla
ma_65b","choices":[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":null}]}

data:
{"id":"endpoint_common_8","object":"chat.completion.chunk","created":1729614610,"model":"lla
ma_65b","usage":{"prompt_tokens":54,"completion_tokens":17,"total_tokens":71},"choices":
[{"index":0,"delta":{"role":"assistant","content":""},"finish_reason":"stop"}]}

data: [DONE]
```

- 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello!"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello!
How"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How
can"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can
I"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I
assist"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist
you"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you
today"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","choices":[{"index":0,"delta":{"role":"assistant","content":"Hello! How can I assist you
today?"},"finish_reason":null}]}

data:
{"id":"endpoint_common_11","object":"chat.completion.chunk","created":1730184192,"model":"ll
ama_65b","full_text":"Hello! How can I assist you today?","usage":
{"prompt_tokens":31,"completion_tokens":10,"total_tokens":41},"choices":[{"index":0,"delta":
{"role":"assistant","content":"Hello! How can I assist you today?"},"finish_reason":"length"}]}

data: [DONE]
```


输出说明

表 8-37 文本推理结果说明

参数名	类型	说明
id	string	请求ID。
object	string	返回结果类型目前都返回"chat.completion"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
choices	list	推理结果列表。
index	integer	choices消息index，当前只能为0。
message	object	推理消息。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。
tool_calls	list	模型工具调用输出。
function	dict	函数调用说明。
arguments	string	调用函数的参数，JSON格式的字符串。
name	string	调用的函数名。
tool_calls.id	string	模型调用工具的ID。
type	string	工具的类型，目前仅支持function。
finish_reason	string	结束原因。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● tool_calls：表示模型调用了工具。
usage	object	推理结果统计数据。

参数名	类型	说明
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。

表 8-38 流式推理结果说明

参数名	类型	说明
data	object	一次推理返回的结果。
id	string	请求ID。
object	string	目前都返回"chat.completion.chunk"。
created	integer	推理请求时间戳，精确到秒。
model	string	使用的推理模型。
full_text	string	全量文本结果，配置项“fullTextEnabled”=true时才有此返回值。
usage	object	推理结果统计数据。
prompt_tokens	int	用户输入的prompt文本对应的token长度。
completion_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中completion_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
total_tokens	int	请求和推理的总token数。
choices	list	流式推理结果。
index	integer	choices消息index，当前只能为0。
delta	object	推理返回结果，最后一个响应为空。
role	string	角色，目前都返回"assistant"。
content	string	推理文本结果。

参数名	类型	说明
finish_reason	string	结束原因，只在最后一次推理结果返回。 ● stop： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 – 请求遇eos结束符正常结束。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。

Triton 流式推理接口

接口功能

提供流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/generate_stream

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
id	可选	请求ID	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
text_input	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-

参数	是否必选	说明	取值要求
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0, 2147483647]，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
batch_size	可选	推理请求batch_size	int类型，取值范围(0，2147483647]，默认值1。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，字段未设置时，默认使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 • true：带模型水印。 • false：不带模型水印。
perf_stat	可选	是否打开性能统计。	bool类型，默认值false。 • true：打开性能统计。 • false：不打开性能统计。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。

参数	是否必选	说明	取值要求
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/generate_stream
```

请求消息体：

- 单模态文本模型：

```
{
   "id":"a123",
   "text_input": "My name is Olivier and I",
   "parameters": {
      "details": true,
      "do_sample": true,
      "max_new_tokens":20,
      "repetition_penalty": 1.1,
      "seed": 123,
      "temperature": 1,
      "top_k": 10,
      "top_p": 0.99,
      "batch_size":100,
      "typical_p": 0.5,
      "watermark": false,
      "perf_stat": false,
      "priority": 5,
      "timeout": 10
   }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
   "id":"a123",
   "text_input": [
      {"type": "text", "text": "My name is Olivier and I"},
      {
         "type": "image_url",
         "image_url": "/xxxx/test.png"
      }
   ],
   "parameters": {
      "details": true,
      "do_sample": true,
      "max_new_tokens":20,
      "repetition_penalty": 1.1,
      "seed": 123,
      "temperature": 1,
      "top_k": 10,
      "top_p": 0.99,
      "batch_size":100,
      "typical_p": 0.5,
      "watermark": false,
      "perf_stat": false,
```



```
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

- 响应样例1：

```
data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"am","details":
{"generated_tokens":1,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"
prefill_time":26.96,"decode_time":null}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" passion","details":
{"generated_tokens":2,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":11},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"ate","details":
{"generated_tokens":3,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":9},"p
refill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" about","details":
{"generated_tokens":4,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":11},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":" music","details":
{"generated_tokens":5,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":".","details":
{"generated_tokens":6,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":12},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"\n","details":
{"generated_tokens":7,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":10},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"T","details":
{"generated_tokens":8,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":19},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"od","details":
{"generated_tokens":9,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":12},"
prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"ay","details":
{"generated_tokens":10,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":30},
"prefill_time":null,"decode_time":16.80}

data:{"id":"a123","model_name":"llama_65b","model_version":null,"text_output":"</s>","details":
{"finish_reason":"eos_token","generated_tokens":424,"first_token_cost":null,"decode_cost":null,"batch_s
ize":1,"queue_wait_time":5067},"prefill_time":null,"decode_time":16.80}
```

- 响应样例2（配置项“fullTextEnabled”=true）：

```
data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m","details":
{"generated_tokens":1,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":5092
},"prefill_time":41.68000030517578,"decode_time":null}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from","details":
{"generated_tokens":2,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":43},"
prefill_time":null,"decode_time":20.440000534057617}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from France","details":
{"generated_tokens":3,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":27},"
prefill_time":null,"decode_time":12.175999641418457}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-
Chat","model_version":null,"text_output":"'m from France.","details":
```


{"generated_tokens":4,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":26},"prefill_time":null,"decode_time":12.128000259399414}

data:{"id":"endpoint_common_20","model_name":"Qwen1.5-14B-Chat","model_version":null,"text_output":"'m from France. I","details":{"finish_reason":"length","generated_tokens":5,"first_token_cost":null,"decode_cost":null,"batch_size":1,"queue_wait_time":26},"prefill_time":null,"decode_time":12.458000183105469}

输出说明

返回值	类型	说明
data	object	一次推理返回的结果。
id	string	请求ID。
model_name	string	模型名称。
model_version	string	模型版本。
text_output	string	推理返回结果。
finish_reason	string	推理结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
details	object	推理details结果。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
first_token_cost	List[token]	文本推理返回，首token产生时间，单位：ms，当前未统计该数据，返回null。
decode_cost	int	decode时间，单位：ms，当前未统计该数据，返回null。

返回值	类型	说明
batch_size	int	流式推理batch size。
queue_wait_time	int	队列等待时间，单位：us。
prefill_time	float	首token时延，单位：ms。
decode_time	float	非首token的token时延，单位：ms。

Triton Token 推理接口

接口功能

提供Token推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/v2/models/*${MODEL_NAME}[/versions/${MODEL_VERSION}]*/infer

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值范围
id	可选	推理请求ID。	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
inputs	必选	只有一个元素的数组。	长度为1。
inputs.name	必选	输入名称，固定"input0"。	字符串长度小于或等于256。

参数	是否必选	说明	取值范围
inputs.shape	必选	参数维度，一维时代表data长度，二维时代表1行n列，n为data长度。	data长度范围为(0, min(1024*1024, maxInputTokenLen, maxSeqLen-1, max_position_embeddings)]。其中max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
inputs.datatype	必选	data数据类型，目前场景仅支持UINT32，传递tokenid。	“UINT32”。
inputs.data	必选	数组，代表输入的tokenId值。	data长度和shape中传入的data长度一致。 tokenId的值需要在模型词表范围内。
outputs	必选	推理结果输出结构。	outputs长度需要和inputs的长度保持一致。
outputs.name	必选	推理结果输出名。	字符串。
parameters	可选	模型推理后处理相关参数。	-
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值范围
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围，(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。

参数	是否必选	说明	取值范围
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。
details	可选	是否返回推理详细输出结果。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	bool类型，默认值false。
perf_stat	可选	是否打开性能统计。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	bool类型，默认值false。 ● true：打开性能统计。 ● false：不打开性能统计。
batch_size	可选	推理请求batch_size。 此配置项为Triton文本推理所使用参数，在Triton Token推理无效，不建议传输。	int类型，取值范围(0，2147483647]，默认值1。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1，5]，默认值5。 值越低优先级越高，最高优先级为1。

参数	是否必选	说明	取值范围
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/infer
```

请求消息体：

```
{
    "id": "42",
    "inputs": [{
        "name": "input0",
        "shape": [
            1,
            10
        ],
        "datatype": "UINT32",
        "data": [
            396, 319, 13996, 29877, 29901, 29907, 3333, 20718, 316, 23924
        ]
    }],
    "outputs": [{
        "name": "output0"
    }],
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "max_new_tokens": 20,
        "watermark": true,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

```
{
    "id": "42",
    "outputs": [
        {
            "name": "output0",
            "shape": [
                1,
                20
            ],
            "datatype": "UINT32",
            "data": [
                1,
                396,
                319,
                13996,
                29877,
                29901,
```



```
            29907,
            3333,
            20718,
            316,
            23924,
            562,
            2142,
            1702,
            425,
            14015,
            16060,
            316,
            383,
            19498
          ]
        }
    ]
}
```

输出说明

返回值	类型	说明
id	string	请求ID。
outputs	list	推理结果列表。
name	string	默认"output0"。
shape	list	结构为[1, n]，1表示1维数组，n表示data字段中token结果长度。
datatype	string	"UINT32"。
data	list	推理后生成的token ID集合。

Triton 文本推理接口

接口功能

提供文本推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/**v2/models/***${MODEL_NAME}*[/versions/*${MODEL_VERSION}*]/**generate**

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	是否必选	说明	取值要求
id	可选	请求ID。	长度不超过256的非空字符串。只允许包含下划线、中划线、大写英文字母、小写英文字母和数字。
text_input	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片

参数	是否必选	说明	取值要求
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
parameters	可选	模型推理后处理相关参数。	-
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。

参数	是否必选	说明	取值要求
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 • 小于1.0表示对重复进行奖励。 • 1.0表示不进行重复度惩罚。 • 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见**4.1 说明**。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。

参数	是否必选	说明	取值要求
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
batch_size	可选	推理请求batch_size。	int类型，取值范围(0，2147483647]，默认值1。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，字段未设置时，默认使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。
perf_stat	可选	是否打开性能统计。	bool类型，默认值false。 ● true：打开性能统计。 ● false：关闭性能统计。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/v2/models/llama_65b/generate
```

请求消息体：

- 单模态文本模型：

```
{
    "id":"a123",
    "text_input": "My name is Olivier and I",
    "parameters": {
        "details": true,
        "do_sample": true,
        "max_new_tokens":20,
```



```
        "repetition_penalty": 1.1,
        "seed": 123,
        "temperature": 1,
        "top_k": 10,
        "top_p": 0.99,
        "batch_size":100,
        "typical_p": 0.5,
        "watermark": false,
        "perf_stat": false,
        "priority": 5,
        "timeout": 10
    }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
    "id":"a123",
    "text_input": [
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "parameters": {
        "details": true,
        "do_sample": true,
        "max_new_tokens":20,
        "repetition_penalty": 1.1,
        "seed": 123,
        "temperature": 1,
        "top_k": 10,
        "top_p": 0.99,
        "batch_size":100,
        "typical_p": 0.5,
        "watermark": false,
        "perf_stat": false,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

```
{
    "id": "a123",
    "model_name": "llama_65b",
    "model_version": null,
    "text_output": "am living in South of France.\nI have been addicted to Jurassic Park since very young. I played some video game versions but especially the great first pinball model from William which reminds me a lot of JPOG1 by song (deluxe). Unfortunately, it stopped working and has been unprofitable for a long time before being exchanged for another game. Fortunately there was the computer version. Nevertheless, it came out only on PC in 2003 when mine was too weak... It's just been a couple of months that the game came out on Mac (a whole 15 years late) with the Version 0.91JAMS ! I know this may be a little antique with the realistic animations and versions today, but the memories are very deep-seated . So thank you all rebuilders for keeping alive wonderful games like this one.\nSince then, I try to keep me updated about this game and test if possible later Alpha. Thank you so much for your work!</s>",
    "details": {
        "finish_reason": "eos_token",
        "generated_tokens": 221,
        "first_token_cost": null,
        "decode_cost": null
    }
}
```


输出说明

返回值	类型	说明
id	string	请求ID。
model_name	string	模型名称。
model_version	string	模型版本。当前未统计该数据，返回null。
text_output	string	推理返回结果。
details	object	推理details结果。
finish_reason	string	推理结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
first_token_cost	List[token]	文本推理返回，首token产生时间，单位：ms，当前未统计该数据，返回null。
decode_cost	int	decode时间，单位：ms，当前未统计该数据，返回null。

MindIE 原生文本/流式推理接口

接口功能

提供文本/流式推理处理功能。

接口格式

操作类型：**POST**

URL：https://*{ip}*:*{port}*/infer

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数名	是否必选	说明	取值要求
inputs	必选	推理请求内容。单模态文本模型为string类型，多模态模型为list类型。	• string：非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。 • list：形如多模态模型inputs参数的示例格式即可。
type	可选	推理请求内容类型。	• text：文本 • image_url：图片

参数名	是否必选	说明	取值要求
text	可选	推理请求内容为文本。	非空，0KB<字符数<=4MB，支持中英文。tokenizer之后的token数量小于或等于maxInputTokenLen、maxSeqLen-1、max_position_embeddings和1MB之间的最小值。其中，max_position_embeddings从权重文件config.json中获取，其他相关参数从配置文件中获取。
image_url	可选	推理请求内容为图片。	支持本地图片传入，图片类型支持jpg、png、jpeg和base64编码的jpg图片，支持URL图片传入。支持http和https协议。当前支持传入的最大图片数为1。
stream	可选	指定返回结果是文本推理还是流式推理。	bool类型，默认值false。 ● true：流式推理。 ● false：文本推理。
parameters	可选	模型推理后处理相关参数。	-
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数名	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, 2147483647]，字段未设置时，默认值由后端模型确定，详情请参见4.1 说明。取值大于或等于vocabSize时，默认值为vocabSize。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，字段未设置时，默认使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int类型，取值范围(0，2147483647]，默认值20。
do_sample	可选	是否做sampling。	bool类型，不传递该参数时，将由其他后处理参数决定是否做sampling。 • true：做sampling。 • false：不做sampling。

参数名	是否必选	说明	取值要求
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
details	可选	是否返回推理详细输出结果。	bool类型，默认值false。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。
priority	可选	设置请求优先级。	uint64_t类型，取值范围[1, 5]，默认值5。 值越低优先级越高，最高优先级为1。
timeout	可选	设置等待时间，超时则断开请求。	uint64_t类型，取值范围(0, 3600]，默认值600，单位：秒。

使用样例

请求样例：

```
POST https://{ip}:{port}/infer
```

请求消息体：

- 单模态文本模型：

```
{
    "inputs": "My name is Olivier and I",
    "stream": false,
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "max_new_tokens": 20,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "details": true,
        "typical_p": 0.5,
        "watermark": false,
        "priority": 5,
        "timeout": 10
    }
}
```

- 多模态模型：

  说明

  "image_url"参数的取值请根据实际情况进行修改。

```
{
    "inputs": [
        {"type": "text", "text": "My name is Olivier and I"},
        {
            "type": "image_url",
            "image_url": "/xxxx/test.png"
        }
    ],
    "stream": false,
    "parameters": {
        "temperature": 0.5,
        "top_k": 10,
        "top_p": 0.95,
        "max_new_tokens": 20,
        "do_sample": true,
        "seed": null,
        "repetition_penalty": 1.03,
        "details": true,
        "typical_p": 0.5,
        "watermark": false,
        "priority": 5,
        "timeout": 10
    }
}
```

响应样例：

- 文本推理（“stream”=false）：

```
{
    "generated_text": "am a french native speaker. I am looking for a job in the hospitality industry. I",
    "details": {
        "finish_reason": "length",
        "generated_tokens": 20,
        "seed": 846930886
    }
}
```

- 流式推理：
  - 流式推理1（“stream”=true，使用sse格式返回）：

```
data: {"prefill_time":45.54,"decode_time":null,"token":{"id":626,"text":"am"}}

data: {"prefill_time":null,"decode_time":128.32,"token":{"id":263,"text":" a"}}
```



```
data: {"prefill_time":null,"decode_time":18.17,"token":{"id":5176,"text":" French"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":17739,"text":" photograph"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":261,"text":"er"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":2729,"text":" based"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":297,"text":" in"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":3681,"text":" Paris"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29889,"text":"."}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":13,"text":"\n"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29902,"text":"I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":505,"text":" have"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1063,"text":" been"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":27904,"text":" shooting"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":1951,"text":" since"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":306,"text":" I"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":471,"text":" was"}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29871,"text":" "}}

data: {"prefill_time":null,"decode_time":16.80,"token":{"id":29896,"text":"1"}}

data: {"prefill_time":null,"decode_time":16.80,"generated_text":"am a French photographer based in Paris.\nI have been shooting since I was 15","details":{"finish_reason":"length","generated_tokens":20,"seed":846930886},"token":{"id":29945,"text":null}}
```

- 流式推理2（“stream”=true，配置项“fullTextEnabled”=true，使用sse格式返回）：

```
data: {"prefill_time":33.55400085449219,"decode_time":null,"token":{"id":5122,"text":"：  "}}

data: {"prefill_time":null,"decode_time":19.922000885009766,"token":{"id":7552,"text":"：  ）"}}

data: {"prefill_time":null,"decode_time":12.803999900817871,"token":{"id":40,"text":"：  ）I"}}

data: {"prefill_time":null,"decode_time":11.869000434875488,"token":{"id":2776,"text":"：  ）I'm"}}

data: {"prefill_time":null,"decode_time":12.008000373840332,"generated_text":"：  ）I'm ","details":null,"token":{"id":4102,"text":null}}
```

输出说明

表 8-39 文本推理结果说明

返回值	类型	说明
generated_text	string	推理返回结果。
details	object	推理details结果。目前定义以下字段，支持扩展。

返回值	类型	说明
finish_reason	string	推理结束原因。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回该seed值。

表 8-40 流式推理结果说明

返回值	类型	说明
data	object	一次推理返回的结果。
prefill_time	float	流式推理下首token时延，单位：ms。
decode_time	float	流式推理下非首token的token时延，单位：ms。
generated_text	string	推理文本结果，只在最后一次推理结果才返回。
details	object	推理details结果，只在最后一次推理结果返回，支持扩展。

返回值	类型	说明
finish_reason	string	推理结束原因，只在最后一次推理结果返回。 ● eos_token：请求正常结束。 ● stop_sequence： – 请求被主动CANCEL或STOP，用户不感知，丢弃响应。 – 请求执行中出错，响应输出为空，err_msg非空。 – 请求输入校验异常，响应输出为空，err_msg非空。 ● length： – 请求因达到最大序列长度而结束，响应为最后一轮迭代输出。 – 请求因达到最大输出长度（包括请求和模型粒度）而结束，响应为最后一轮迭代输出。 ● invalid flag：无效标记。
generated_tokens	int	推理结果token数量。PD场景下统计P和D推理结果的总token数量。当一个请求的推理长度上限取maxIterTimes的值时，D节点响应中generated_tokens数量为maxIterTimes+1，即增加了P推理结果的首token数量。
seed	int	如果请求指定了sampling seed，返回改seed值。
token	List[token]	每一次推理的token。
id	int	token ID。
text	string	token对应文本。

Token 计算接口

接口功能

将文本转换为token。

端口类型

数据端口。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/tokenizer

说明

- {ip}优先取**启动命令**参数中的{predict_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_ip”参数。
- {port}优先取**启动命令**参数中的{predict_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“predict_port”参数。
- 当ms_coordinator.json的"algorithm_type"参数配置为"cache_affinity"时，该接口不支持使用。

请求参数

参数	取值要求	是否必选	说明
inputs	string	必选	输入文本

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/tokenizer
```

请求消息体：

```
{
    "inputs": "what is your name?"
}
```

响应样例：

```
{
    "token_number": 6,
    "tokens": ["what", "is", "your", "name", "?"]
}
```

输出说明

参数	类型	说明
token_number	size_t	文本转换为token的个数。
tokens	string[]	每个token的值。

8.3.4.6.3 集群内通信接口

启动状态查询接口

接口功能

查询服务启动状态。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/startup

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/startup
```

响应样例：

服务已启动时无内容。

输出说明

- 状态码200，表示服务已启动，消息体没有内容。
- 无响应，表示服务未启动。

健康状态查询接口

接口功能

查询服务状态是否正常。

端口类型

管理接口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/health

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/health
```

响应样例：

服务状态正常时无内容。

输出说明

- 状态码200，表示服务状态正常，消息体没有内容。
- 无响应，表示服务异常。

就绪状态查询接口

接口功能

查询服务就绪状态。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/readiness

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/readiness
```

响应样例：

服务准备就绪时无内容。

输出说明

- 状态码200，表示服务已就绪，消息体没有内容。
- 状态码503，表示服务未就绪，消息体没有内容。

Triton 健康检查接口

接口功能

检查Triton健康状态。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v2/health/live

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2/health/live
```

响应样例：

服务正常运行时无内容。

输出说明

- 状态码200，表示服务正常运行，消息体没有内容。
- 无响应，表示服务异常。

Triton 就绪状态检查接口

接口功能

检查Triton就绪状态。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v2/health/ready

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/readiness
```

响应样例：

服务准备就绪时无内容。

输出说明

- 状态码200，表示服务已就绪，消息体没有内容。
- 状态码503，表示服务未就绪，消息体没有内容。

Triton 模型就绪状态检查接口

接口功能

检查Triton模型就绪状态。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v2/models/{MODEL_NAME}/versions/${MODEL_VERSION}/ready

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。
- ${MODEL_NAME}字段指定需要查询的模型名称。
- /versions/${MODEL_VERSION}字段暂不支持，不传递。

请求参数

无

使用样例

请求样例：

```
GET https://{ip}:{port}/v2/models/llama3-70b/ready
```

响应样例：

Triton模型已就绪时无内容。

输出说明

- 状态码200，表示Triton模型已就绪，消息体没有内容。
- 状态码503，表示Triton模型未就绪，消息体没有内容。

集群节点同步接口

接口功能

同步集群节点。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/instances/refresh

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

参数	类型	说明
ids	uint64_t[]	必填。 同步节点的ID列表。
instances	object[]	必填。 同步节点的具体信息。
id	uint64_t	必填。 节点ID。
ip	string	必填。 节点IP。
port	string	必填，取值范围：["1024", "65535"] 节点数据端口。
model_name	string	必填。 节点加载的模型名。
static_info	object	必填。 节点静态信息。
group_id	uint64_t	必填。 节点所属的组ID。
max_seq_len	uint32_t	必填。取值大于0。 节点最大队列长度。
max_output_len	uint32_t	必填，取值范围[1, max_seq_len - 1]。 节点最大推力输出长度。
total_slots_num	uint32_t	必填，取值范围[1, 5000]。 节点最大推理任务数。
total_block_num	uint32_t	必填，取值大于0。 节点最大内存块个数。
block_size	uint32_t	必填，取值范围[1, 128]。 内存块大小。
lable	uint32_t	必填。只在PD分离场景有用。 节点标签；取值如下所示： ● 2：Prefill节点 ● 3：Decode节点

参数	类型	说明
role	uint32_t	必填。 节点身份；取值如下所示： ● 80：Prefill节点 ● 68：Decode节点 ● 85：Undef节点
dynamic_info	object	必填。 节点动态信息。
avail_slots_num	uint32_t	必填。取值范围[0, total_slots_num]。 节点剩余可用推理任务数。
avail_block_num	uint32_t	必填。取值范围[0, total_block_num]。 节点剩余可用内存块个数。
peers	uint64_t[]	Decode节点连接的所有Prefill节点ID。 ● 当节点为Prefill时，不需要该字段； ● 当节点为Decode时，必填。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/instances/refresh
```

请求消息体：

```
{
   "ids": [
      0,1
   ],
   "instances": [
      {
         "id": 0,
         "ip": "0.0.0.0",
         "port": "1025",
         "model_name": "your_model_name",
         "static_info": {
            "group_id": 0,
            "max_seq_len": 2048,
            "max_output_len": 512,
            "total_slots_num": 200,
            "total_block_num": 1024,
            "block_size": 128,
            "label": 2,
            "role": 80
         },
         "dynamic_info": {
            "avail_slots_num": 200,
            "avail_block_num": 1024
         }
      },
      {
         "id": 1,
         "ip": "0.0.0.0",
         "port": "1025",
```



```
        "model_name": "your_model_name",
        "static_info": {
            "group_id": 0,
            "max_seq_len": 2048,
            "max_output_len": 512,
            "total_slots_num": 200,
            "total_block_num": 1024,
            "block_size": 128,
            "label": 3,
            "role": 68
        },
        "dynamic_info": {
            "avail_slots_num": 200,
            "avail_block_num": 1024,
            "peers": [
                0
            ]
        }
    }
  ]
}
```

响应样例：

正常时无响应消息体。

输出说明

当请求的消息体json格式和字段正确时，返回http200状态码；否则返回http400状态码。

集群节点停止服务接口

接口功能

停止集群节点服务。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/instances/offline

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

参数	类型	说明
ids	uint64_t[]	必填。 停止服务的节点ID。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/instances/offline
```

请求消息体：

```
{
   "ids": [
      0,1
   ]
}
```

响应样例：

正常时无响应消息体。

输出说明

- 请求的消息体json格式和字段正确时，返回http200状态码。
- 当Coordinator未就绪时，返回http503状态码。
- 当请求消息体格式错误时，返回http400状态码。

集群节点恢复服务接口

接口功能

恢复集群节点服务。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/instances/online

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

参数	类型	说明
ids	uint64_t[]	必填。 恢复服务的节点ID。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/instances/online
```

请求消息体：

```
{
   "ids": [
      0,1
   ]
}
```

响应样例：

正常时无响应消息体。

输出说明

- 请求的消息体json格式和字段正确时，返回http200状态码。
- 当Coordinator未就绪时，返回http503状态码。
- 当请求消息体格式错误时，返回http400状态码。

节点请求状态查询接口

接口功能

查询节点请求状态。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/instances/tasks?{key}={value}&{key}={value}

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。
- ${key}字段只支持"id"。
- ${value}字段为节点ID。
- 多个{key}={value}之间用&分隔。

请求参数

无。

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/instances/tasks?id=1&id=2&id=3
```

响应样例：

```
{
    "tasks": [0,0,0]
}
```

输出说明

表 8-41

参数	类型	说明
tasks	uint64_t[]	每个查询节点上的剩余任务数。

PD 节点之间任务状态查询接口

接口功能

查询PD节点之间的任务状态。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**POST**

URL：https://{ip}:{port}/v1/instances/query_tasks

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

参数	类型	说明
p_id	uint64_t	必填。 P节点ID。
d_id	uint64_t	必填。 D节点ID。

使用样例

请求样例：

```
POST https://{ip}:{port}/v1/instances/query_tasks
```

请求消息体：

```
{
    "p_id": 0,
    "d_id": 1
}
```

响应样例：

- PD之间没有任务
  ```
  {
      "is_end": true
  }
  ```
- PD之间有任务
  ```
  {
      "is_end": false
  }
  ```

输出说明

参数	类型	说明
is_end	bool	PD之间任务是否结束。

Coordinator 运行状态查询接口

接口功能

查询Coordinator运行状态。

说明

该接口不需要用户调用，属于Coordinator和Controllor之间的通信接口。

端口类型

管理端口。

接口格式

操作类型：**GET**

URL：https://{ip}:{port}/v1/coordinator_info

说明

- {ip}优先取**启动命令**参数中的{manage_ip}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_ip”参数。
- {port}优先取**启动命令**参数中的{manage_port}；如果没有配置该命令行参数，则取配置文件ms_coordinator.json的“manage_port”参数。

请求参数

无。

使用样例

请求样例：

```
GET https://{ip}:{port}/v1/coordinator_info
```

响应样例：

```
{
   "schedule_info": [
      {
         "id": 0,
         "allocated_slots": 10,
         "allocated_blocks": 10
      },
      {
         "id": 1,
         "allocated_slots": 10,
         "allocated_blocks": 10
      }
   ],
   "request_length_info": {
      "input_len": 100,
      "output_len": 100
   }
}
```


输出说明

参数	类型	说明
schedule_info	object[]	必填。 节点调度信息。
id	uint64_t	必填。 节点ID。
allocated_slots	size_t	必填。 已分配的slots数量。
allocated_blocks	size_t	必填。 已分配的block数量。
request_length_info	-	必填。 请求的平均输入输出长度。
input_len	size_t	必填。 请求的平均输入长度。
output_len	size_t	必填。 请求的平均输出长度。

8.3.4.7 报错信息查询

ms_coordinator启动后，运行过程中可能出现的报错信息主要如下所示。

异常类型	异常原因	异常表现

系统异常	自身节点通信异常。	与周边组件失联，报出错误日志，级别为致命，关键日志信息如下所示：
		● 获取请求信息失败：
		`Get request info failed`
		如果推理请求结果正确，请忽略。
		● 记录请求与用户链接的映射失败：
		`Connection record failed`
		如果推理请求结果正确，请忽略。
		● 获取链接失败：
		`Get connection failed`
		该报错信息可能是推理请求处理过程中，集群里部分节点存在故障，导致通信失败；只要集群内还有可用的节点，后续的推理请求会调用到可用的节点上进行处理，可能会影响推理性能，但不影响请求推理结果。

异常类型	异常原因	异常表现
	进程异常退出。	部署平台通过健康探针，识别该异常退出，实现进程重启，业务恢复。 部署平台会将故障异常退出的节点重新拉起，如果一直拉起失败，则需要检查是否硬件故障。
	集群中无可用调度节点。	拒绝推理服务请求，在返回的错误码中告知错误详情。 记录异常日志，级别为致命。关键日志信息如下所示： ● PD分离不够1P1D： `MindIE-MS Coordinator is not ready` ● PD混部不够1个节点： `MindIE-MS Coordinator is not ready` 该报错信息说明集群中可用的节点数量不足，有以下两种情况： ● 节点还没完成初始化，等待一段时间后再次查看。 ● 节点出现故障。

	集群中某些节点无法通信。	报出错误日志，关键日志信息如下所示：
		● 与MindIE Server连接失败：
		`Apply connection to node failed`
		该报错信息可能是推理请求处理过程中，集群里部分节点存在故障，导致通信失败；只要集群内还有可用的节点，后续的推理请求会调用到可用的节点上进行处理，可能会影响推理性能，但不影响请求推理结果。
		● P节点的返回消息格式不对：
		`P instance error`
		该报错信息为程序BUG，需进一步排查问题。
		● 收到不存在的请求推理结果：
		`DResultNormalToken: Get request id info failed`
		该报错信息为程序BUG，需进一步排查问题。
		● 接收P节点消息失败：
		`Receive messages from p instance failed`
		该报错信息为程序BUG，需进一步排查问题。
		● 发送消息给P节点失败：
		`Send messges to p instance failed`
		该报错信息可能是推理请求处理过程中，集群里部分节点存在故障，导致通信失败；只要集群内还有可用的节点，后续的推理请求会调用到可用的节点上进行处理，可能会影响推理性能，但不影响请求推理结果。

异常类型	异常原因	异常表现
请求异常	请求量过大，超出限流阈值。	拒绝推理服务请求，关键日志信息如下所示： `Too many requests` 该报错信息两种解决方式如下所示： ● 请用户自行降低发送请求的数量，直到没有出现Too many requests报错信息。 ● 修改ms_coordinator.json文件中的“max_requests”参数，按照实际需要发送的推理请求量调大此值。

	请求处理超时。	返回通知用户请求处理超时。包含三种类型： 调度超时，首token超时和推理整体超时。关键日志信息如下所示：
		● 调度超时：
		`Request schedule timeout`
		现该报错信息时，建议降低发送请求的数量，或者修改ms_coordinator.json文件中的“schedule_timeout”参数，按照用户能接受的等待实际调大此值。
		● 首token超时：
		`Request first token timeout`
		出现该报错信息时，建议降低发送请求的数量，或者修改ms_coordinator.json文件中的“first_token_timeout”参数，按照用户能接受的等待实际调大此值。
		● 推理整体超时：
		`Request inference timeout`
		出现该报错信息时，建议降低发送请求的数量，或者修改ms_coordinator.json文件中的“infer_timeout”参数，按照用户能接受的等待实际调大此值。
		● tokenizer超时：
		`Request tokenizer timeout`
		出现该报错信息时，建议降低发送请求的数量，或者修改ms_coordinator.json文件中的“tokenizer_timeout”参数，按照用户能接受的等待实际调大此值。

	推理请求处理计算失败。	返回通知用户请求处理失败，直接将MindIE Server的错误日志透传回用户，并删除本次请求。 如果推理请求结果正确，请忽略；如果推理不正确，说明为程序BUG，需进一步排查问题。

8.3.5 日志配置

- 客户端日志会根据*{$HOME}*/.mindie_ms/msctl.json配置的日志等级log_level参数进行过滤，将日志内容打印到客户端屏幕上。
- 服务端日志会根据服务端配置文件ms_server.json中的以下代码进行设置。

```
"log_info": {
    "log_level": "INFO",                    // 日志级别
```



```
    "run_log_path": "/var/log/mindie-ms/run/log.txt",  // 运行日志写入的文件路径
    "operation_log_path": "/var/log/mindie-ms/operation/log.txt" //操作日志写入的文件路径
}
```

说明

- 过滤等级后，将日志内容写入"log_path"路径中的日志文件，服务端报错可以通过日志进行定位；当前日志写入策略是循环写入，最多保存10个日志文件，每个日志文件最大为20M。
- MindIE MS客户端可通过MINDIEMS_LOG_LEVEL环境变量动态设置日志打印等级，如下所示：
  ```
  export MINDIEMS_LOG_LEVEL={日志打印等级}
  ```
  - DEBUG
  - INFO
  - WARNING
  - ERROR
  - CRITICAL
- 将日志写入日志文件时，需要导入以下KMC依赖的环境变量。
  export HSECEASY_PATH=$MIES_INSTALL_PATH/lib

8.4 MindIE Server

8.4.1 功能介绍

面向通用模型的推理服务化场景，实现开放、可扩展的推理服务化平台架构，支持对接业界主流推理框架接口，满足大语言模型、文生图等多类型模型的高性能推理需求。主要包括以下特性：

- 服务启动：Daemon负责推理服务启动，加载配置文件，初始化其他模块。
- 北向接口：ServerEndpoint面向推理服务开发者提供极简易用的API接口，支持Triton/OpenAI/TGI/vLLM第三方主流推理框架请求接口。
- 统一推理API：统一推理API提供模型初始化、推理请求处理、服务/模型信息查询接口，ServerEndpoint和其他三方框架调用统一推理API使能推理服务。
- 推理服务化平台：GMIS模块支持推理流程的工作流定义扩展，以工作流为驱动，实现从推理任务调度到任务执行的可扩展架构，适应各类推理方法如投机推理、LoRA推理、LLMA、Prompt增强等的快速落地。
- 南向接口：ModelWrapper模块面向不同推理引擎，不同模型，提供统一抽象接口，便于扩展，减少推理引擎、模型变化带来的修改。

8.4.2 配置参数说明

说明

配置项取值请参考《 MindIE安装指南 》中“配置MindIE > 配置MindIE Server > **单机推理**”章节的步骤3。

配置文件参数说明

配置项	取值类型	取值范围	配置说明
Version	std::string	"1.0.0"	标注配置文件版本，当前版本指定为1.0.0，不支持修改。
LogConfig	map	-	日志相关配置。详情请参见**LogConfig参数说明**。
ServerConfig	map	-	服务端相关配置，例如ip:port、网络请求、网络安全等。详情请参见**ServerConfig参数说明**。
BackendConfig	map	-	模型后端相关配置，包含调度、模型相关配置。详情请参见**BackendConfig参数说明**。

LogConfig 参数说明

配置项	取值类型	取值范围	配置说明
logLevel	std::string	• "Verbose" • "Info" • "Warning" • "Error" • "None"	• "Verbose"：打印Verbose、Info、Warning和Error级别的日志。 • "Info"：打印Info、Warning和Error级别的日志。 • "Warning"：打印Warning和Error级别的日志。 • "Error"：打印Error级别的日志。 • "None"：不打印日志。 必填，默认值："Info"。 此配置支持热更新。
logFileSize	uint32_t	[0, 500]	日志最大文件大小，单位MB。 选填，默认值：20。
logFileNum	uint32_t	[0, 64]	历史日志文件最多保存数量。 选填，默认值：20。

配置项	取值类型	取值范围	配置说明
logPath	std::string	日志文件路径，长度<=4096。 日志路径设置限制与操作系统有关，且会受到spdlog三方库约束。因此建议配置的最大长度为1024，且符合目录规范。	支持绝对路径和相对路径。如果配置为相对路径，则代码中会取安装目录，最后拼接而成。 例如，假设MindIE Service的安装路径为“/opt/Ascend-mindie-service_{version}_linux-x86_64/”，当logPath=“logs/mindservice.log”，则其实际日志生成路径为“/opt/Ascend-mindie-service_{version}_linux-x86_64/logs/mindservice.log”。 若配置路径不满足要求，则使用默认路径。 必填，默认值："logs/mindservice.log"。

ServerConfig 参数说明

配置项	取值类型	取值范围	配置说明
ipAddress	std::string	IPv4地址。	EndPoint提供的业务面RESTful接口绑定的IP地址。 ● 如果存在环境变量MIES_CONTAINER_IP，则优先取环境变量值作为业务面IP地址。 ● 如果不存在环境变量MIES_CONTAINER_IP，则取该配置值。 必填，默认值："127.0.0.1"。 **说明** 全零侦听会导致三面隔离失效，不满足安全配置要求，故默认禁止绑定IP地址为0.0.0.0。若仍需绑定IP地址为0.0.0.0，那么在保证安全前提下，需要将配置文件中的“allowAllZeroIpListening”设置为true。

配置项	取值类型	取值范围	配置说明
managementIpAddress	std::string	IPv4地址。	EndPoint提供的管理面RESTful接口绑定的IP地址。 ● 如果该环境变量MIES_CONTAINER_MANAGEMENT_IP存在，则直取环境变量值作为管理面IP地址。 ● 如果“managementIpAddress”字段存在，则取字段本身值；否则取“ipAddress”字段的值作为管理面IP地址。 ● 如果采用多IP地址的方案，对“ipAddress”和“managementAddress”的初始值都需要做相应的修改。 选填，默认值："127.0.0.2"。 **说明** 全零侦听会导致三面隔离失效，不满足安全配置要求，故默认禁止绑定IP地址为0.0.0.0。若仍需绑定IP地址为0.0.0.0，那么在保证安全前提下，需要将配置文件中的“allowAllZeroIpListening”设置为true。
port	int32_t	[1024, 65535]	EndPoint提供的业务面RESTful接口绑定的端口号。 如果采用物理机/宿主机IP地址通信，请自行保证端口号无冲突。 必填，默认值：1025。
managementPort	int32_t	[1024, 65535]	EndPoint提供的管理面（管理面接口请参见**表8-42**）接口绑定的端口号。 业务面与管理面可采用四种方案： ● 多IP地址多端口号（推荐） ● 多IP地址单端口号 ● 单IP地址多端口号 ● 单IP地址单端口号 选填，默认值：1026。
metricsPort	int32_t	[1024, 65535]	EndPoint提供的性能指标监控接口绑定的端口号。 如果采用物理机/宿主机IP地址通信，请自行保证端口号无冲突。 必填，默认值：1027。

配置项	取值类型	取值范围	配置说明
allowAllZeroIpListening	bool	• true • false	是否支持全零侦听IP配置。 • true：支持全零侦听IP配置。 • false：不支持全零侦听IP配置。 必填，默认值：false，建议值：false。取值为true时，会存在全零侦听风险，用户环境需要自行保证具备全零侦听的防护能力。
maxLinkNum	uint32_t	[1, 1000]	RESTful接口请求并发处理数，EndPoint支持的最大并发请求处理数。 表示有maxLinkNum个请求正在并发处理，此外有2*maxLinkNum个请求在队列中等待。因此第3*maxLinkNum+1个请求会被拒绝。 必填，默认值：1000。
httpsEnabled	bool	• true • false	是否开启HTTPS通信安全认证。 • true：开启HTTPS通信。 • false：关闭HTTPS通信。 必填，默认值：true，建议值：true，取值为false时，忽略后续HTTPS通信相关参数。
fullTextEnabled	bool	• true • false	是否开启流式接口全量返回历史结果。 • true：开启流式接口全量返回历史结果。 • false：关闭流式接口全量返回历史结果。 选填，默认值：false。
tlsCaPath	std::string	文件绝对路径长度<=4096。实际路径为工程路径+tlsCaPath，上限与操作系统有关，最小值为1。	根证书路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true生效，生效后必填，默认值："security/ca/"。
tlsCaFile	std::set<std::string>	文件绝对路径长度<=4096。不可为空，并且tlsCaPath+tlsCaFile路径长度上限与操作系统有关，最小值为1。	业务面根证书名称列表。 “httpsEnabled”=true生效，生效后必填，默认值：["ca.pem"]。

配置项	取值类型	取值范围	配置说明
tlsCert	std::string	文件绝对路径长度<=4096。实际路径为工程路径+tlsCert，上限限制与操作系统有关，最小值为1。	业务面服务证书文件路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true生效，生效后必填，默认值："security/certs/server.pem"。
tlsPk	std::string	文件绝对路径长度<=4096。实际路径为工程路径+tlsPk，上限与操作系统有关，最小值为1。	业务面服务证书私钥文件路径，只支持软件包安装路径下的相对路径，证书私钥的长度要求>=3072。 “httpsEnabled”=true生效，生效后必填，默认值："security/keys/server.key.pem"。
tlsPkPwd	std::string	文件绝对路径长度<=4096。支持为空；若非空，则实际路径为工程路径+tlsPkPwd，上限与操作系统有关，最小值为1。	业务面服务证书私钥加密密钥文件路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true生效，生效后选填，默认值："security/pass/key_pwd.txt"。 若私钥经过加密但是未提供此文件，系统启动时会要求用户在交互窗口输入私钥加密口令。
tlsCrl	std::string	文件绝对路径长度<=4096。支持为空；若非空，则实际路径为工程路径+tlsCrl，上限与操作系统有关，最小值为1。	业务面服务证书吊销列表文件路径，只支持软件包安装路径下的相对路径。 ● “httpsEnabled”=true生效，生效后必填，默认值："security/certs/server_crl.pem"。 ● “httpsEnabled”=false不启用吊销列表。 “tlsCrl”的值只能配套“tlsCaFile”文件列表中的第一个CA文件。
managementTlsCaFile	std::set<std::string>	建议tlsCaPath+managementTlsCaFile路径长度<=4096。不可为空，并且tlsCaPath+managementTlsCaFile路径长度上限与操作系统有关，最小值为1。	管理面根证书名称列表，当前管理面证书和业务面证书放在同一个路径（tlsCaPath）下。 “httpsEnabled”=true且“ipAddress”!=“managementIpAddress”生效，生效后必填，默认值：["management_ca.pem"]。

配置项	取值类型	取值范围	配置说明
managementTlsCert	std::string	建议文件路径长度<=4096。实际路径为工程路径+managementTlsCert，上限与操作系统有关，最小值为1。	管理面服务证书文件路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true且“ipAddress”!=“managementIpAddress”生效，生效后必填，默认值："security/certs/management/server.pem"。
managementTlsPk	std::string	建议文件路径长度<=4096。实际路径为工程路径+managementTlsPk，上限与操作系统有关，最小值为1。	管理面服务证书私钥文件路径，只支持软件包安装路径下的相对路径，证书私钥的长度要求>=3072。 “httpsEnabled”=true且“ipAddress”!=“managementIpAddress”生效，生效后必填，默认值："security/keys/management/server.key.pem"。
managementTlsPkPwd	std::string	文件路径长度<=4096。支持为空；若非空，则实际路径为工程路径+managementTlsPkPwd，上限与操作系统有关，最小值为1	管理面服务证书私钥加密密钥文件路径。 “httpsEnabled”=true且“ipAddress”!=“managementIpAddress”生效，生效后选填，默认值："security/pass/management/key_pwd.txt"。 若私钥经过加密但是未提供此文件，系统启动时会要求用户在交互窗口输入私钥加密口令。
managementTlsCrl	std::string	建议文件路径长度<=4096。支持为空；若非空，则实际路径为工程路径+managementTlsCrl，上限与操作系统有关，最小值为1。	管理面证书吊销列表文件路径，只支持软件包安装路径下的相对路径。 • “httpsEnabled”=true且“ipAddress”!=“managementIpAddress”生效，生效后必填，默认值："security/certs/management/server_crl.pem"。 • “httpsEnabled”=false不启用吊销列表。 “managementTlsCrl”的值只能配套“managementTlsCaFile”文件列表中的第一个CA文件。
kmcKsfMaster	std::string	建议文件路径长度<=4096。实际路径为工程路径+kmcKsMaster，上限与操作系统有关，最小值为1。	KMC密钥库文件路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true生效，生效后必填，默认值："tools/pmt/master/ksfa"。

配置项	取值类型	取值范围	配置说明
kmcKsfStandby	std::string	建议文件路径长度<=4096。实际路径为工程路径+kmcKsStandby，上限与操作系统有关，最小值为1。	KMC密钥库备份文件路径，只支持软件包安装路径下的相对路径。 “httpsEnabled”=true生效，生效后必填，默认值："tools/pmt/standby/ksfb"。
inferMode	std::string	• standard • dmi	标识是否PD分离。 • standard：表示PD混部模式； • dmi：表示PD分离模式。 默认值：standard。
interCommTLSEnabled	bool	• true • false	集群内部实例间的通信是否启用TLS。 • true：启用 • false：不启用 选填，默认值：true，需要配置证书相关内容。
interCommPort	uint16_t	[1024, 65535]	集群内部实例间的通信端口。 选填，默认值：1121。
interCommTlsCaFile	std::string	建议文件路径长度<=4096。实际路径为工程路径+interCommTlsCaFile，上限与操作系统有关，最小值为1。	集群内部实例间的通信如果启用TLS，则使用这里指定的文件作为CA。 选填，默认值："security/grpc/ca/ca.pem"。
interCommTlsCert	std::string	建议文件路径长度<=4096。实际路径为工程路径+interCommTlsCert，上限与操作系统有关，最小值为1。	集群内部实例间的通信如果启用TLS，则使用这里指定的文件作为证书。 选填，默认值："security/grpc/certs/server.pem"。
interCommPk	std::string	建议文件路径长度<=4096。实际路径为工程路径+interCommPk，上限与操作系统有关，最小值为1。	集群内部实例间的通信如果启用TLS，则使用这里指定的文件作为私钥。 选填，默认值："security/grpc/keys/server.key.pem"。

配置项	取值类型	取值范围	配置说明
interCommPkPwd	std::string	建议文件路径长度<=4096。实际路径为工程路径+interCommPkPwd，上限与操作系统有关，最小值为1。	集群内部实例间的通信如果启用TLS，则使用这里指定的文件作为私钥的密码。 选填，默认值："security/grpc/pass/key_pwd.txt"。
interCommTlsCrl	std::string	建议文件路径长度<=4096。实际路径为工程路径+interCommTlsCrl，上限与操作系统有关，最小值为1。	集群内部实例间的通信如果启用TLS，则使用这里指定的文件作为证书吊销列表。 选填，默认值："security/grpc/certs/server_crl.pem"。
openAiSupport	std::string	-	是否启用vLLM兼容的OpenAI。选填，默认值："vllm"。 ● 取值为"vllm"或者配置字段缺失时，代表**/v1/chat/completions**接口使用vLLM兼容的OpenAI接口版本。 ● 取值为其他字符时，代表**/v1/chat/completions**接口使用原生OpenAI接口版本。 此配置支持热更新。

说明

- 如果网络环境不安全，不开启HTTPS通信，即“httpsEnabled”=“false”时，会存在较高的网络安全风险。
- 如果推理服务所在的计算节点的网络为跨公网和局域网，绑定0.0.0.0的IP地址可能导致网络隔离失效，存在较大安全风险。故该场景下默认禁止EndPoint的IP地址绑定为0.0.0.0。若用户仍需要使用0.0.0.0，请在环境具备全零监听防护能力的前提下，通过设置配置项“allowAllZeroIpListening”=true手动打开允许配置0.0.0.0的IP地址开关，启用全零监听的安全风险由用户自行承担。
- 如果配置了相同的管理面和业务面的IP地址，会导致隔离失效。

BackendConfig 参数说明

配置项	取值类型	取值范围	配置说明
backendName	std::string	长度1~50，只支持小写字母和下划线。且不以下划线作为开头和结尾。	推理后端名称，可以通过该参数获得后端实例。 必填，目前只支持："mindieservice_llm_engine"。

配置项	取值类型	取值范围	配置说明
modelInstanceNumber	uint32_t	[1, 10]	模型实例个数。 必填，默认值：1。 单模型多机推理场景，该值需为1。
npuDeviceIds	std::vector<std::set<size_t>>	根据模型和环境的实际情况来决定。	表示启用哪几张卡。对于每个模型实例分配的npuIds。 多机推理场景下该值无效，每个节点上使用的npuDeviceIds根据ranktable计算获得。 必填，默认值：[[0,1,2,3]]。 **说明** 1. 执行下列命令，设置设备上加速库可见卡（**此操作需在启动MindIE Server之前执行**）。 `export ASCEND_RT_VISIBLE_DEVICES=0,1,2,3,4,5,6,7` 2. “npuDeviceIds”表示在上述可见卡中，按ASCEND_RT_VISIBLE_DEVICES设置的顺序，选取对应的卡。 例如： ● ASCEND_RT_VISIBLE_DEVICES=7,0,1,2,3,4,5,6 ● npuDeviceIds：[[0,1,3,4]] 则最终分配的卡为7,0,2,3。
tokenizerProcessNumber	uint32_t	[1, 32]	tokenizer进程数。必填，默认值：8。 在CPU核较多时，可以适当调大该值，tokenizer性能会更好。
multiNodesInferEnabled	bool	● true ● false	● false：单机推理 ● true：多机推理 选填，默认值：false。
multiNodesInferPort	int32_t	[1024, 65535]	跨机通信的端口号，多机推理场景使用。 选填，默认值：1120。
interNodeTLSEnabled	bool	● true ● false	多机推理时，跨机通信是否开启证书安全认证。 ● true：开启证书安全认证。 ● false：关闭证书安全认证。 选填，默认值：true。取值为false时，忽略后续参数。

配置项	取值类型	取值范围	配置说明
interNodeTlsCaFile	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeTlsCaFile，上限与操作系统有关，最小值为1。	根证书名称路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："security/grpc/ca/ca.pem"。
interNodeTlsCert	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeTlsCert，上限与操作系统有关，最小值为1。	服务证书文件路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："security/grpc/certs/server.pem"。
interNodeTlsPk	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeTlsPk，上限与操作系统有关，最小值为1。	服务证书私钥文件路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："security/grpc/keys/server.key.pem"。
interNodeTlsPkPwd	std::string	建议文件路径长度<=4096。支持为空；若非空，则实际路径为工程路径+interNodeTlsPkPwd，上限与操作系统有关，最小值为1。	服务证书私钥加密密钥文件路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："security/grpc/pass/mindie_server_key_pwd.txt"。
interNodeTlsCrl	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeTlsCrl，上限与操作系统有关，最小值为1。	服务证书吊销列表文件路径。 “interNodeTLSEnabled”=true生效，生效后选填，默认值："security/grpc/certs/server_crl.pem"。
interNodeKmcKsfMaster	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeKmcKsfMaster，上限与操作系统有关，最小值为1。	KMC密钥库文件路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："tools/pmt/master/ksfa"。

配置项	取值类型	取值范围	配置说明
interNodeKmcKsfStandby	std::string	建议文件路径长度<=4096。实际路径为工程路径+interNodeKmcKsfStandby，上限与操作系统有关，最小值为1。	KMC密钥库备份文件路径，只支持软件包安装路径下的相对路径。 “interNodeTLSEnabled”=true生效，生效后必填，默认值："tools/pmt/standby/ksfb"。
ModelDeployConfig	map	-	模型部署相关配置。详情请参见**ModelDeployConfig参数说明**。
ScheduleConfig	map	-	调度相关配置。详情请参见**ScheduleConfig参数说明**。

ModelDeployConfig 参数说明

配置项	取值类型	取值范围	配置说明
maxSeqLen	uint32_t	上限根据显存和用户需求来决定，最小值需大于0。	最大序列长度。请根据推理场景选择合适的maxSeqLen。 如果maxSeqLen大于模型支持的最大序列长度，可能会影响推理精度。 必填，默认值：2560。
maxInputTokenLen	uint32_t	(0, 4294967295]	输入token id最大长度。 必填，默认值：2048。 maxInputTokenLen = min(maxInputTokenLen, maxSeqLen -1) • 当truncation=true时，请求的输入长度inputLen会自动截断，请求的实际输入长度inputLen = min(inputLen, maxInputLen)。 • 当truncation=false时，若请求的输入长度inputLen > maxInputTokenLen，会返回Error。
truncation	bool	• true • false	是否进行参数合理化校验拦截。 • false：校验 • true：不校验 选填，默认值：false。
ModelConfig	map	-	模型相关配置，包括后处理参数。详情请参见**ModelConfig参数说明**。

ModelConfig 参数说明

配置项	取值类型	取值范围	配置说明
modelInstanceType	std::string	• "Standard" • "StandardMock"	模型类型。 • "Standard"：普通推理 • "StandardMock"：假模型（此模式下不加载模型，仅运行Server） 选填，默认值："Standard"。
modelName	string	由大写字母、小写字母、数字、中划线、点和下划线组成，且不以中划线、点和下划线作为开头和结尾，字符串长度小于或等于256。	模型名称。 必填，默认值："llama_65b"。
modelWeightPath	std::string	文件绝对路径长度的上限与操作系统有关，最小值为1。	模型权重路径。程序会读取该路径下的config.json中torch_dtype和vocab_size字段的值，需保证路径和相关字段存在。 必填，默认值："/data/atb_testdata/weights/llama1-65b-safetensors"。 该路径会进行安全校验，需要和执行用户的属组和权限保持一致。
worldSize	uint32_t	根据模型实际情况来决定。每一套模型参数中worldSize必须与使用的NPU数量相等。	启用几张卡推理。目前llama-65b至少启用四张NPU卡。 多机推理场景下该值无效，worldSize根据ranktable计算获得。 PD分离场景下需要与下发身份设置的卡数一致。 必填，默认值：4。
cpuMemSize	uint32_t	上限根据显存和用户需求来决定。只有当maxPreemptCount为0时，才可以取值为0。	单个CPU中可以用来申请KV Cache的size上限。 必填，默认值：5，建议值：5，单位：GB。

配置项	取值类型	取值范围	配置说明
npuMemSize	int32_t	• -1 • 整型数字，取值范围：(0, 2147483647]	单个NPU中可以用来申请KV Cache的size上限。 必填，默认值：-1，建议值：-1，单位：GB。 • 自动分配KV Cache：当配置值为-1时，kv_cache会根据可用显存自动进行分配。 KV Cache快速计算公式： npuMemSize=单卡总内存*内存分配比例-单卡权重占用内存-运行时相关变量占用内存-系统占用内存 – 单卡总内存：通过**npu-smi info**命令查看总显存。 – 内存分配比例：默认值0.8；可通过环境变量NPU_MEMORY_FRACTION控制；**当出现权重加载OOM情况时，可适当调高分配比例或使用更多显卡进行推理**。 – 单卡权重占用内存：约等于权重大小*类型大小（浮点为2，int8类型为1）/ 卡数；以实际加载权重为准。 – 运行时相关变量占用内存：模型输入变量、输出变量、中间变量等内存。 – 系统占用内存：通过**npu-smi info**命令查看静息状态下使用显存。 • 手动分配KV Cache：当配置值大于0时，根据设置值会固定分配KV Cache大小。 **说明** • 为快速确定最佳显存参数取值范围，提供了快速计算公式。该公式计算的结果仅作为取值参考，为了达到最佳性能，可以适当向上调整该值并进行性能压力测试。 • 如果设置的“npuMemSize”参数超过系统可分配最大显存值，会出现推理服务启动失败、推理服务启动卡死等异常现象，需要减小该值并重试。 • PD分离部署场景中，当“backendType”选择“atb”时该参数才可以设置为-1。

配置项	取值类型	取值范围	配置说明
backendType	std::string	• "atb" • "ms"	对接的后端类型。 • atb：推理引擎后端为加速库。 • ms：推理引擎后端为MindSpore。 必填，默认值："atb"。 **说明** 如果选择"ms"作为对接的推理引擎后端，需要提前安装MindSpore和MindFormers，以及修改MindIE启动配置信息，详情请参见**链接**。

ScheduleConfig 参数说明

配置项	取值类型	取值范围	配置说明
templateType	std::string	• "Standard" • "Mix"	推理类型。 • Standard：PD混部场景，Prefill请求和Decode请求各自组batch。 • Mix：Splitfuse特性相关参数，Prefill请求和Decode请求可以一起组batch。 PD分离场景下该字段配置不生效。 必填，默认值："Standard"。
templateName	std::string	当前取值只能为："Standard_LLM"	调度工作流名称。 必填，默认值："Standard_LLM"。
cacheBlockSize	uint32_t	[1, 128]	KV Cache block的size大小。 必填，默认值：128，建议值：128，其他值建议取为2的n次幂。
maxPrefillBatchSize	uint32_t	[1, maxBatchSize]	最大prefill batch size。maxPrefillBatchSize和maxPrefillTokens谁先达到各自的取值就完成本次组batch。 该参数主要是在明确需要限制Prefill阶段batch size的场景下使用，否则可以设置为0（此时引擎将默认取maxBatchSize值）或与maxBatchSize值相同。 必填，默认值：50。

配置项	取值类型	取值范围	配置说明
maxPrefillTokens	uint32_t	[5120,2097152]，且必须大于或等于maxInputTokenLen的取值。	每次Prefill时，当前batch中所有input token总数，不能超过maxPrefillTokens。maxPrefillTokens和maxPrefillBatchSize谁先达到各自的取值就完成本次组batch。 不建议设置过大，若显存溢出可适当调小。 必填，默认值：8192。
prefillTimeMsPerReq	uint32_t	[0, 1000]	与decodeTimeMsPerReq比较，计算当前应该选择Prefill还是decode。单位：ms，当“supportSelectBatch”设置为“true”时有效。其调度策略流程图请参见图8-7。 必填，默认值：150。
prefillPolicyType	uint32_t	• 0 • 1 • 2 • 3	Prefill阶段的调度策略。其调度策略流程图请参见图8-8。 • 0：FCFS，先来先服务。 • 1：STATE，Prefill阶段等同于FCFS策略。 • 2：PRIORITY，优先级队列。 • 3：MLFQ，多级反馈队列。 其中，3是0/1的组合。 必填，默认值：0。
decodeTimeMsPerReq	uint32_t	[0, 1000]	与prefillTimeMsPerReq比较，计算当前应该选择Prefill还是Decode。单位：ms，当“supportSelectBatch”设置为“true”时有效。其调度策略流程图请参见图8-7。 必填，默认值：50。
decodePolicyType	uint32_t	• 0 • 1 • 2 • 3	Decode阶段的调度策略。其调度策略流程图请参见图8-8。 • 0：FCFS，先来先服务。 • 1：STATE，Decode阶段优先执行未被抢占和换出的请求。 • 2：PRIORITY，优先级队列。 • 3：MLFQ，多级反馈队列。 其中，3是0/1的组合。 必填，默认值：0。

配置项	取值类型	取值范围	配置说明
maxBatchSize	uint32_t	[1, 5000]，且必须大于或等于maxPreemptCount参数的取值。	最大decode batch size。 1. 首先计算block_num：Total Block Num = Floor(NPU显存/(模型网络数*cacheBlockSize*模型注意力头数*注意力头大小*Cache类型字节数*Cache数))，其中，Cache数=2；在tensor并行的情况下，block_num*world_size为实际的分配block数。 如果是多卡，公式中的模型注意力头数*注意力大小的值需要均摊在每张卡上，即“模型注意力头数*注意力大小/卡数”。 公式中的Floor表示计算结果向下取整。 2. 为每个请求申请的block数量Block Num=Ceil(输入Token数/Block Size)+Ceil(最大输出Token数/Block Size)。输入Token数：输入（字符串）做完tokenizer后的tokenID个数；最大输出Token数：模型推理最大迭代次数和最大输出长度之间取较小值。 公式中的Ceil表示计算结果向上取整。 3. maxBatchSize=Total Block Num/Block Num。 必填，默认值：200。
maxIterTimes	uint32_t	[1, maxSeqLen-1]	模型全局最大输出长度。与请求级最大输出token个数max_tokens（或max_new_tokens）取较小值作为最大可生成长度。 必填，默认值：512。 请求的最大输出长度maxOutputLen=min(maxIterTimes, max_tokens， max_new_tokens) 请求的实际输出长度outputLen = min(maxSeqLen - inputLen, maxOutputLen)
maxPreemptCount	uint32_t	[0, maxBatchSize]，当取值大于0时，cpuMemSize取值不可为0。	每一批次最大可抢占请求的上限，即限制一轮调度最多抢占请求的数量，最大上限为maxBatchSize，取值大于0则表示开启可抢占功能。 必填，默认值：0。

配置项	取值类型	取值范围	配置说明
supportSelectBatch	bool	• true • false	batch选择策略。 PD分离场景下该字段不生效。 • false：表示每一轮调度时，优先调度和执行Prefill阶段的请求。 • true：表示每一轮调度时，根据当前Prefill与Decode请求的数量，自适应调整Prefill和Decode阶段请求调度和执行的先后顺序。 必填，默认值：false。
maxQueueDelayMicroseconds	uint32_t	[500, 1000000]	请求的队列等待时间，单位：us。 必填，默认值：5000。

图 8-7 调度策略和执行先后顺序流程图

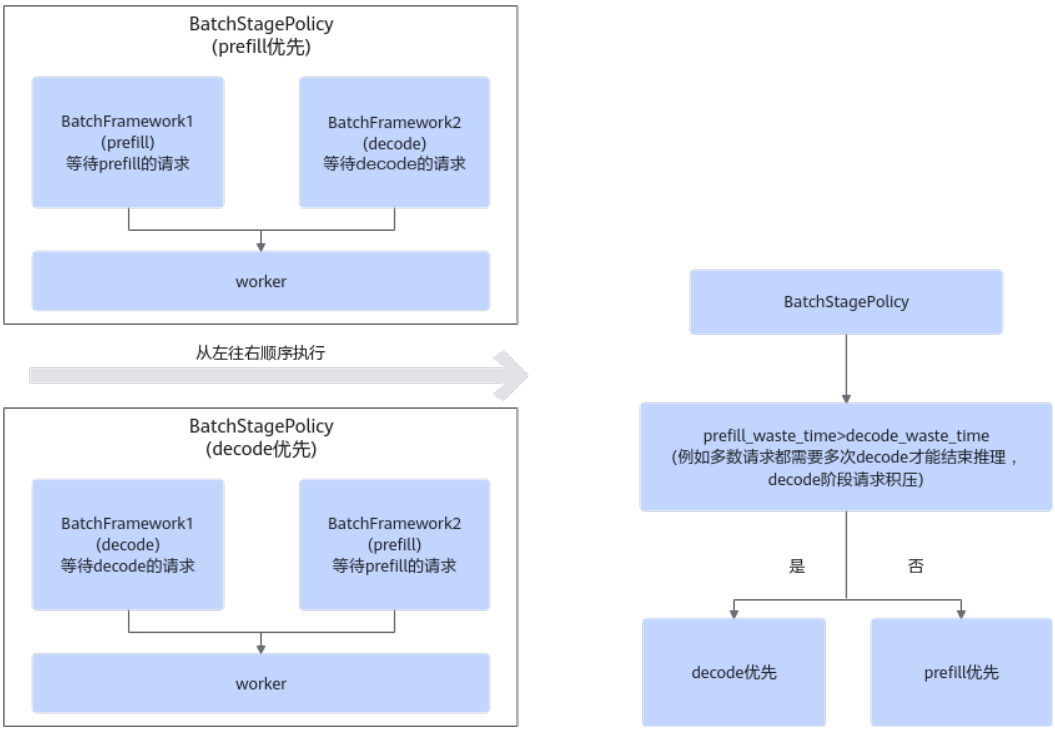
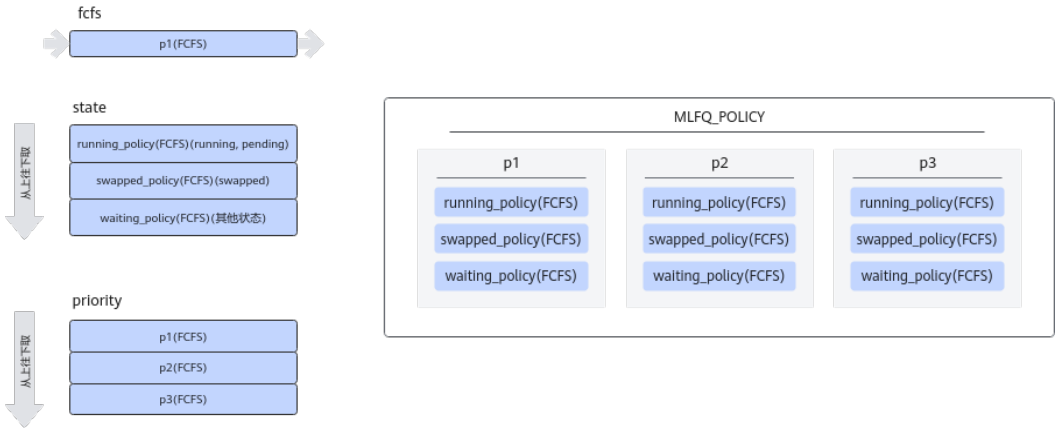


图 8-8 Prefill 和 Decode 阶段的调度策略流程图



8.4.3 使用指导

场景说明

MindIE Server提供EndPoint模块对推理服务化协议和接口封装，兼容Triton/OpenAI/TGI/vLLM等第三方框架接口。使用单节点安装模式安装MindIE Server之后，用户使用客户端（Linux curl命令，Postman工具等）发送HTTP/HTTPS请求，即可调用EndPoint提供的接口。

EndPoint RESTful 接口使用说明

HTTP/HTTPS请求的URL的IP地址和端口号在config.json中进行配置，详情请参见**ServerConfig参数说明**。

- 以Linux curl工具发送generate请求，URL请求格式如下：
  - 操作类型：**POST**
  - **URL：http*[s]*://*{ip}*:*{port}*/generate**
- 未开启HTTPS，发送推理请求：

```
curl -H "Accept: application/json" -H "Content-type: application/json" -X POST -d '{
  "inputs": "My name is Olivier and I",
  "parameters": {
    "details": true,
    "do_sample": true,
    "repetition_penalty": 1.1,
    "return_full_text": false,
    "seed": null,
    "temperature": 1,
    "top_p": 0.99
  },
  "stream": false
}' http://{ip}:{port}/generate
```

- HTTPS双向认证的请求方式示例：

```
curl --location --request POST 'https://{ip}:{port}/generate' \
--header 'Content-Type: application/json' \
--cacert /home/runs/static_conf/ca/ca.pem \
--cert /home/runs/static_conf/cert/client.pem \
--key /home/runs/static_conf/cert/client.key.pem \
--data-raw '{
    "inputs": "My name is Olivier and I",
    "parameters": {
        "best_of": 1,
```



```
        "decoder_input_details": false,
        "details": false,
        "do_sample": true,
        "max_new_tokens": 20,
        "repetition_penalty": 2,
        "return_full_text": false,
        "seed": 12,
        "temperature": 0.1,
        "top_k": 1,
        "top_p": 0.9,
        "truncate": 1024
    },
    "stream": true
}'
```

说明

- --cacert：验签证书文件路径。
- ca.pem为MindIE Server服务端证书的验签证书/根证书。
- --cert：客户端证书文件路径。
- client.pem为客户端证书。
- --key：客户端私钥文件路径。
- client.key.pem为客户端证书私钥（未加密，建议采用加密密钥）。

请用户根据实际情况对相应参数进行修改。

提供的RESTful API列表如下：

表 8-42 服务状态查询 API（管理面的查询类接口）

API	接口类型	URL	说明	支持框架
Server Live	GET	/v2/health/live	检查服务器是否在线。	Triton
Server Ready	GET	/v2/health/ready	检查服务器是否准备就绪。	Triton
Model Ready	GET	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/ready	检查模型是否准备就绪。	Triton
health	GET	/health	服务健康检查。	TGI/vLLM
查询TGI EndPoint信息	GET	/info	查询TGI EndPoint信息。	TGI
Slot统计	GET	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/getSlotCount	参考Triton格式，自定义的Slot统计信息查询接口。	原生

API	接口类型	URL	说明	支持框架
健康探针接口	GET	/health/timed[-${TIMEOUT}]	检查推理流程是否正常。	原生
优雅退出接口	GET	/stopService	实现整个服务的优雅退出。调用该接口时，会等待服务中正在执行和等待的所有请求完成，并关闭服务，等待时所有推理接口将不可用。	原生
静态配置采集接口	GET	/v1/config	采集静态配置。	原生
动态状态采集接口	GET	/v1/status	采集动态状态。	原生
指定实例身份接口	POST	/v1/role/${role}	指定实例身份。	原生
服务指标接口（JSON格式）	GET	/metrics-json	获取推理服务过程中请求的TTFT（Time To First Token）、TBT（Time Between Tokens）的动态平均值（默认近1000个请求的平均值），正在执行请求数、正在等待请求数量、剩余NPUblock数量。	原生
服务监控指标查询接口（普罗格式）	GET	/metrics	查询推理服务化的相关服务监控指标。	原生

表 8-43 模型/服务查询 API（业务面的查询接口）

API	接口类型	URL	说明	支持框架
models列表	GET	/v1/models	列举当前可用模型列表。	OpenAI
model详情	GET	/v1/models/{model}	查询模型信息。	OpenAI

API	接口类型	URL	说明	支持框架
服务元数据查询	GET	/v2	获取服务元数据。	Triton
模型元数据查询	GET	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]	查询模型元数据信息。	Triton
查询模型配置	GET	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/config	查询模型配置。	Triton

表 8-44 推理 API（业务面的业务接口）

API	接口类型	URL	说明	支持框架
推理任务	POST	/	TGI推理接口，stream==false返回文本推理结果，stream==true返回流式推理结果。	TGI
	POST	/generate	TGI和vLLM的推理接口，通过请求参数来区分是哪种服务的接口。	TGI/vLLM
	POST	/generate_stream	TGI流式推理接口，使用Server-Sent Events格式返回结果。	TGI
	POST	/v1/chat/completions	OpenAI文本推理接口。	OpenAI
	POST	/infer	原生推理接口，支持文本/流式返回结果。	原生
	POST	/infer_token	原生推理接口，实现token输入的文本/流式推理。	原生
	POST	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/infer	Triton的token推理接口。	Triton

API	接口类型	URL	说明	支持框架
	POST	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/stopInfer	参考Triton接口定义，提供提前终止请求接口。	原生
	POST	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/generate	Triton文本推理接口。	Triton
	POST	/v2/models/${MODEL_NAME}[/versions/${MODEL_VERSION}]/generate_stream	Triton流式推理接口。	Triton
	POST	/v1/tokenizer	计算token数量。	原生
	GET	/dresult	调度器与D实例间，存在一个长连接，D实例每推理出一个结果，就通过该长连接响应给调度器。	PD分离相关

说明

- ${MODEL_NAME}字段指定需要查询的模型名称。
- [/versions/${MODEL_VERSION}]字段暂不支持，不传递。

9 FAQ

9.1 发送请求返回乱码

问题描述

使用curl命令、MindIE Client或者Benchmark发起request请求，Server返回如下乱码。


```
root@ubuntu:~# curl  http://[redacted]/info
<!DOCTYPE html
        PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/[redacted]">
<html xmlns="http://www.w3.[redacted]">

<head>
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />
        <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
        <meta name="keywords" content="SWG,Proxy,NetentSec" />

        <title>HIS Proxy Notification</title>
        <link rel="icon" href="http://his.huawei.[redacted]o">
        <style type=text/css>
                body {
                        width: 100%;
                        float: none;
                        background-color: #FFFFFF;
                        font-size: 0.75em;
                        margin: 0 auto;
                }

                #header {
                        width: 100%;
                        min-width: 1342px;
                        height: 20px;
                        padding: 5px 0;
                        background-color: black;
                }

                #headerContainer {
                        width: 1200px;
                        height: 20px;
                        margin: auto;
                }
```

原因分析

设置代理后，代理服务器无法识别Service配置的内网IP，无法将请求转发给目标服务器，导致请求失败。

解决方案

使用以下命令即可解决。

```
unset http_proxy
unset https_proxy
```

9.2 部署 Service 服务时出现 LLMPythonModel initializes fail 的报错提示

问题描述

部署Service服务时，出现LLMPythonModel initializes fail的报错，如下图所示。

```
2024-03-27 16:42:59.836422 29577 Log started at [trace] level
2024-03-27 16:42:59.836427 29577 Log default format: yyyy-mm-dd hh:mm:ss.uuuuuu threadid loglevel msg

2024-03-27 16:42:59.843784 29581 Log started at [trace] level
2024-03-27 16:42:59.843816 29581 Log default format: yyyy-mm-dd hh:mm:ss.uuuuuu threadid loglevel msg

2024-03-27 16:42:59.843782 29580 Log started at [trace] level
2024-03-27 16:42:59.843817 29580 Log default format: yyyy-mm-dd hh:mm:ss.uuuuuu threadid loglevel msg

2024-03-27 16:42:59.843904 29583 Log started at [trace] level
2024-03-27 16:42:59.843935 29583 Log default format: yyyy-mm-dd hh:mm:ss.uuuuuu threadid loglevel msg

2024-03-27 16:42:59.843902 29582 Log started at [trace] level
2024-03-27 16:42:59.843932 29582 Log default format: yyyy-mm-dd hh:mm:ss.uuuuuu threadid loglevel msg
2024-03-27 16:43:02.626024 29595 error LLMModelExecutorPython.cpp:54] LLMPythonModel initializes fail: modelWrapper return status is error
2024-03-27 16:43:02.626116 29595 error slave_IPC_communicator.cpp:286] [SlaveIPCCommunicator::ProcessBroadcastRecvMessage]Executor failed to init model
2024-03-27 16:43:02.626134 29595 error slave_IPC_communicator.cpp:399] [HandleRcvMsg]Executor failed to process recv broadcastMessage
2024-03-27 16:43:02.626138 29595 error slave_IPC_communicator.cpp:407] [HandleRcvMsg]slave communicator notify connector to terminate
2024-03-27 16:43:02.629014 29593 error LLMModelExecutorPython.cpp:54] LLMPythonModel initializes fail: modelWrapper return status is error
2024-03-27 16:43:02.629093 29593 error slave_IPC_communicator.cpp:286] [SlaveIPCCommunicator::ProcessBroadcastRecvMessage]Executor failed to init model
2024-03-27 16:43:02.629112 29593 error slave_IPC_communicator.cpp:399] [HandleRcvMsg]Executor failed to process recv broadcastMessage
2024-03-27 16:43:02.629116 29593 error slave_IPC_communicator.cpp:407] [HandleRcvMsg]slave communicator notify connector to terminate
2024-03-27 16:43:02.641121 29594 error LLMModelExecutorPython.cpp:54] LLMPythonModel initializes fail: modelWrapper return status is error
2024-03-27 16:43:02.641225 29594 error slave_IPC_communicator.cpp:286] [SlaveIPCCommunicator::ProcessBroadcastRecvMessage]Executor failed to init model
2024-03-27 16:43:02.641246 29594 error slave_IPC_communicator.cpp:399] [HandleRcvMsg]Executor failed to process recv broadcastMessage
2024-03-27 16:43:02.641250 29594 error slave_IPC_communicator.cpp:407] [HandleRcvMsg]slave communicator notify connector to terminate
2024-03-27 16:43:02.724189 29592 error LLMModelExecutorPython.cpp:54] LLMPythonModel initializes fail: modelWrapper return status is error
2024-03-27 16:43:02.724262 29592 error slave_IPC_communicator.cpp:286] [SlaveIPCCommunicator::ProcessBroadcastRecvMessage]Executor failed to init model
2024-03-27 16:43:02.724279 29592 error slave_IPC_communicator.cpp:399] [HandleRcvMsg]Executor failed to process recv broadcastMessage
2024-03-27 16:43:02.724283 29592 error slave_IPC_communicator.cpp:407] [HandleRcvMsg]slave communicator notify connector to terminate
2024-03-27 16:43:04.486649 29577 warning llm_daemon.cpp:41] received exit signal[17]
(base) [root@localhost logs]#
```


原因分析

ibis缺少Python依赖。

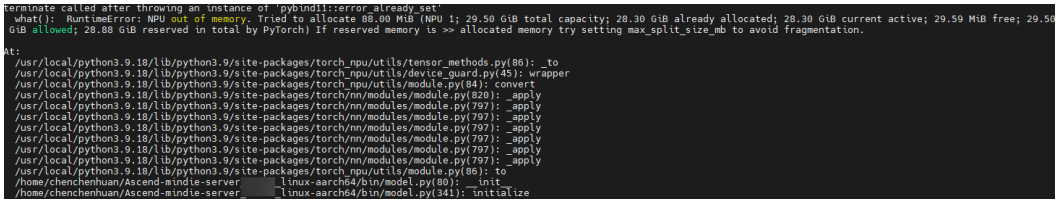
解决步骤

进入/*Service安装路径*/logs目录，打开Python日志，根据日志报错信息，安装需要的依赖。

9.3 加载模型时出现 out of memory 报错提示

问题描述

部署service服务，加载llama-65b模型时出现out of memory报错提示，如下图所示。



原因分析

权重太大，内存不足。

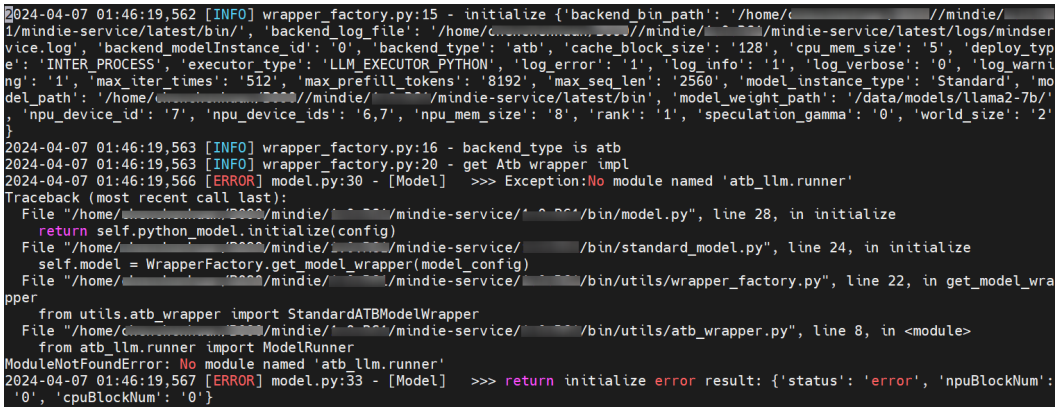
解决步骤

将config.json文件中ModelConfig的npuMemSize调小，比如调成8。

9.4 部署 Service 服务时，出现 atb_llm.runner 无法 import 报错

问题描述

部署Service服务时，出现atb_llm.runner无法import，如下图所示。



原因分析

由于Python版本不是配套版本3.10，或者pip对应的Python版本不是目标版本3.10，找不到对应的包。可以通过python和pip -V查看对应的Python版本进行确认。

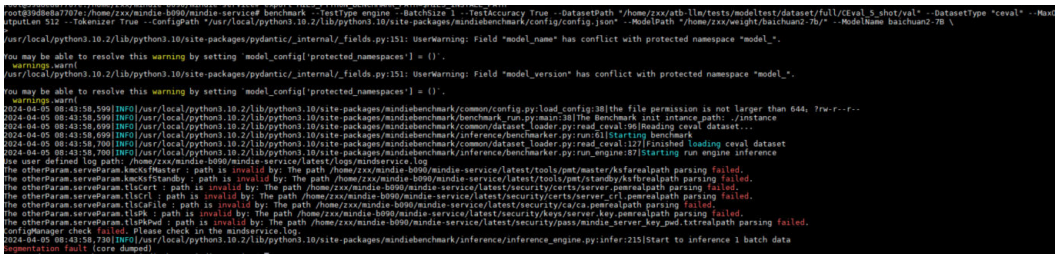
解决步骤

1. 使用以下命令打开bashrc文件。
   ```
   vim ~/.bashrc
   ```
2. 在bashrc文件内添加如下环境变量，保存并退出。
   ```
   ## 例如系统使用3.10.13版本，安装目录位于/usr/local/python3.10.13
   export LD_LIBRARY_PATH=/usr/local/python3.10.13/lib:$LD_LIBRARY_PATH
   export PATH=/usr/local/python3.10.13/bin:$PATH
   ```
3. 使用以下命令是环境变量生效。
   ```
   source ~/.bashrc
   ```
4. 使用以下命令建立软链接。
   ```
   ln -s /usr/local/python3.10.13/bin/python3.10 /usr/bin/python
   ln -s /usr/local/python3.10.13/bin/pip3.10 /usr/bin/pip
   ```

9.5 部署 Service 服务时，找不到 tlsCert 等路径

问题描述

部署Service服务时，找不到tlsCert等路径，如下图所示。



原因分析

开启https服务时，未将需要的证书放到对应的目录下。

解决步骤

将生成服务器证书、CA证书、和服务器私钥等认证需要的文件，放置在对应的目录。

9.6 MindIE Benchmark 使用 Client 推理模式运行时出现 HTTPS 连接错误

问题描述

curl命令正常发送，MindIE Benchmark使用Client推理模式运行时报HTTPS连接错误，如下图所示。

原因分析

config.json中配置的managementIpAddress和managementPort和benchmark命令中--ManagementHttp参数不匹配。

解决步骤

如果用户修改了config.json的managementIpAddress和managementPort配置，根据具体配置使用--ManagementHttp参数。

9.7 使用 MindIE MS 部署单机任务后系统出现异常，不能通过 MindIE MS 客户端卸载

问题描述

使用MindIE MS部署单机任务后系统出现异常，不能通过MindIE MS客户端卸载，需要手动执行kubectl命令删除相关资源。

解决步骤

执行kubectl命令手动删除相关资源，命令如下：

```
kubectl delete deployment {server_name} -n {namespace}
kubectl delete service {server_name} -n {namespace}
```

{server_name}：是用户服务配置中**表3-6**的server_name。

{namespace}：是用户服务配置中**表3-4**的namespace。

9.8 使用 MindIE MS 部署多机任务后系统出现异常，不能通过 MindIE MS 客户端卸载

问题描述

使用MindIE MS部署多机任务后系统出现异常，不能通过MindIE MS客户端卸载，需要手动执行kubectl命令删除相关资源。

解决步骤

执行kubectl命令手动删除相关资源，命令如下：

```
kubectl delete deployment {server_name}-deployment-0 -n {namespace}
kubectl delete cm rings-config-{server_name}-deployment-0 -n {namespace}
kubectl delete service {server_name}-service -n {namespace}
```

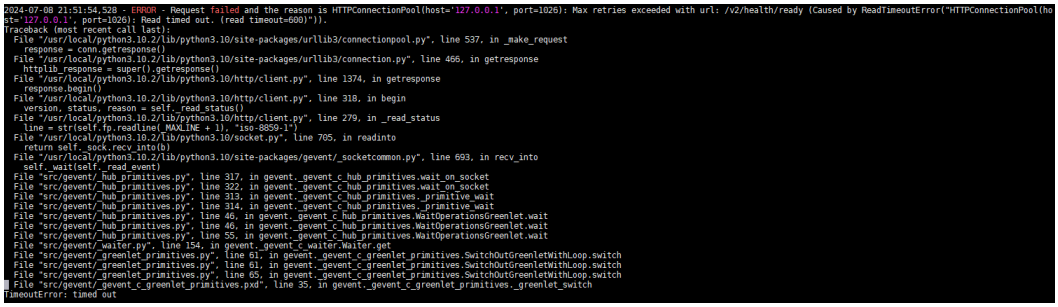
{server_name}：是用户服务配置中**表3-6**的server_name。

{namespace}：是用户服务配置中表3-4的namespace。

9.9 使用 MindIE Benchmark 或者脚本发送请求时出现超时提醒

问题描述

使用MindIE Benchmark或者脚本对MindIE Server发送请求时，部分请求出现超时且无返回的情况。



原因分析

发送请求速率超过服务化所能处理请求的能力，请求积压导致返回超时。

解决步骤

- 使用MindIE Benchmark对MindIE Server发送请求时

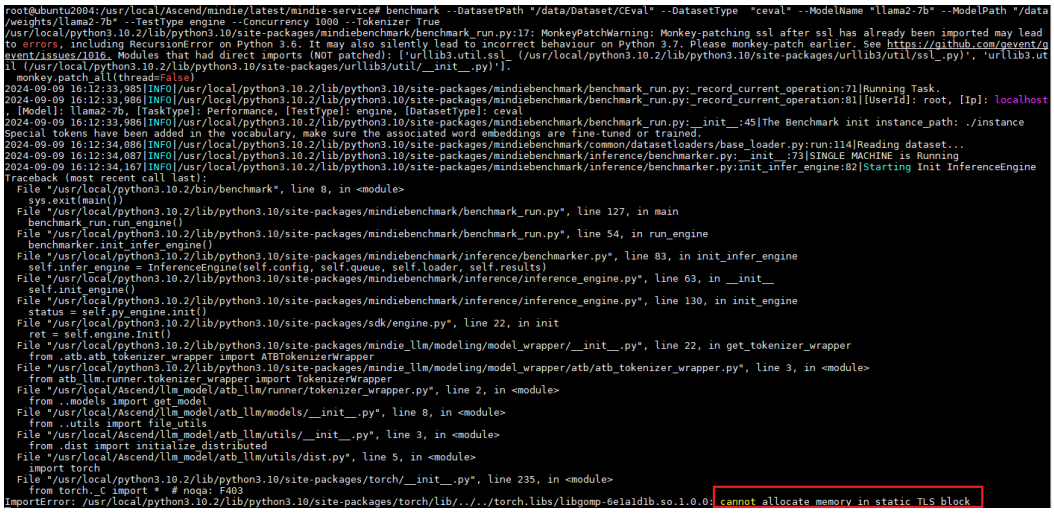
  降低并发数，即降低MindIE Benchmark输入参数--Concurrency的值，其理论值为：npuBlockNum*cacheBlockSize/(平均输入长度+平均输出长度)。
- 使用脚本对MindIE Server发送请求时

  可提升脚本中设置超时的时间限制。

9.10 使用第三方库 transformers 跑模型推理时，报错“cannot allocate memory in static TLS block”

问题描述

报错详细信息如下所示：



原因分析

glibc.so本身的bug。

解决步骤

执行以下命令：

```
export LD_PRELOAD=$LD_PRELOAD:/usr/local/python3.10.2/lib/python3.10/site-packages/torch/lib/../../torch.libs/libgomp-6e1a1d1b.so.1.0.0
```


10 附录

10.1 环境变量

- MindIE Server的环境变量如下所示：

参数名称	参数说明	取值范围	缺省值
MIES_INSTALL_PATH	MindIE Server的安装路径。	路径参数。	/usr/local/Ascend/mindie/latest/mindie-service
MIES_CONFIG_JSON_PATH	config.json文件的路径。 如果该环境变量存在，则读取该值； 如果不存在，则读取*${MIES_INSTALL_PATH}*/conf/config.json文件。	路径参数。	NA

参数名称	参数说明	取值范围	缺省值
MIES_CONTAINER_IP	容器IP地址，容器部署时配置。 EndPoint提供的业务面RESTful接口绑定的IP地址和多机推理场景GRPC通信采用的IP地址。多机推理时需要设置该环境变量。	IPv4地址。	NA
MIES_CONTAINER_MANAGEMENT_IP	EndPoint提供的管理面RESTful接口绑定的IP地址。	IPv4地址。	NA
MIES_SERVICE_MONITOR_MODE	推理服务化的在线指标监控开关。	• 0：关闭 • 1：开启	0
MIES_MEMORY_DETECTOR_MODE	内存状态打点使能开关。	• 0：关闭 • 1：开启	0
MIES_PROFILER_MODE	性能状态打点使能开关。	• 0：关闭 • 1：开启	0
HOST_IP	宿主机IP地址，宿主机部署时配置，当前商用仅支持容器场景，不建议设置。	IPv4地址。	NA
LD_LIBRARY_PATH	lib所在的路径。	路径参数。	${MIES_INSTALL_PATH}/lib:${LD_LIBRARY_PATH}
RANKTABLEFILE	ranktable json文件的绝对路径。 • 多机推理必须配置。 • 单机推理建议取消该环境变量（取消命令：**unset RANKTABLEFILE**）。如果设置该环境变量，文件内容必须正确有效（节点IP地址和device_ip必须正确），否则会导致模型初始化失败。	路径参数。	NA

参数名称	参数说明	取值范围	缺省值
ATB_OPERATION_EXECUTE_ASYNC	算子setup和execute异步执行开关。	• 0：关闭 • 1：开启	1
ASCEND_SLOG_PRINT_TO_STDOUT	CANNDEV日志打印控制开关。	• 1：打屏。 • 0：写入到“~/ascend”目录。	0
ASCEND_GLOBAL_LOG_LEVEL	CANNDEV日志级别。	• 0：debug • 1：info • 2：warn • 3：error	3
ASCEND_GLOBAL_EVENT_ENABLE	设置应用类日志是否开启Event日志。	• 0：关闭Event日志。 • 1：开启Event日志。	0
TASK_QUEUE_ENABLE	推理使能多stream。	• 0：表示推理使能单stream。 • 1：表示推理使能多stream。	1
HCCL_BUFFSIZE	控制两个NPU之间共享数据的缓存区大小。	大于或等于1，单位：MB。	120
ASDOPS_LOG_TO_FILE	算子库日志是否输出到文件。	• 0：输出到文件，默认输出路径为“~/atb/log”。 • 1：不输出。	0
ASDOPS_LOG_TO_STDOUT	算子库日志打印控制开关。	• 0：不打印日志。 • 1：打印日志。	0

参数名称	参数说明	取值范围	缺省值
ASDOPS_LOG_LEVEL	算子库日志级别。	• FATAL • ERROR • WARN • INFO • DEBUG	ERROR
ASDOPS_LOG_TO_FILE_FLUSH	是否刷新日志写文件。	• 0：关闭 • 1：开启	0
ATB_LOG_TO_FILE	加速库环境变量，加速库（ATB Models）日志是否输出到文件。	• 0：输出到文件，默认输出路径为“~/atb/log”。 • 1：不输出。	0
ATB_LOG_TO_STDOUT	加速库环境变量，加速库（ATB Models）日志打印控制开关。	• 0：不打印日志。 • 1：打印日志。	0
ATB_LOG_LEVEL	加速库环境变量，加速库（ATB Models）日志级别。	• TRACE • DEBUG • INFO • WARN • ERROR • FATAL	ERROR
ATB_LOG_TO_FILE_FLUSH	日志写文件是否刷新。	• 0：关闭 • 1：开启	0
EP_OPENSSL_PATH	EndPoint开启HTTPS认证后，通过该环境变量来指定openssl加载运行时so文件。该环境变量在EndPoint模块启动时自动设置，不需要用户手动设置。	路径参数。	${MIES_INSTALL_PATH}/lib

参数名称	参数说明	取值范围	缺省值
HSECEASY_PATH	EndPoint开启HTTPS认证后，使用HSECEASY工具对秘钥口令进行加密。该环境变量指定HSECEASY加载运行时so文件路径。	路径参数。	${MIES_INSTALL_PATH}/lib
OCK_LOG_LEVEL	后处理环境变量。	• TRACE • DEBUG • INFO • WARN • ERROR • FATAL	ERROR
OCK_LOG_TO_STDOUT	后处理环境变量，加速库日志打印控制开关。	• 0：不打印日志。 • 1：打印日志。	0
MIES_CERTS_LOG_TO_FILE	证书管理工具环境变量，日志是否输出到文件。	• 0：输出到文件。 • 1：不输出。	0
MIES_CERTS_LOG_TO_STDOUT	证书管理工具环境变量，日志打印控制开关。	• 0：不打印日志。 • 1：打印日志。	1
MIES_CERTS_LOG_LEVEL	证书管理工具环境变量，日志级别。	• DEBUG • INFO • WARNING • ERROR • FATAL	INFO
MIES_CERTS_LOG_PATH	证书管理工具环境变量，日志路径。	路径参数。	/workspace/log/certs.log
MINDIE_LLM_LOG_TO_FILE	MindIE LLM日志是否打印到文件	0：不打印 1：打印，默认输出路径为“~/atb/log”	1
MINDIE_LLM_LOG_TO_STDOUT	MindIE LLM日志是否打印到标准输出	0：不打印 1：打印	0

参数名称	参数说明	取值范围	缺省值
MINDIE_LLM_CONTINUOUS_BATCHING	是否支持batch内结束请求退出，未结束请求继续推理。建议开启。	• 0：关闭 • 1：开启	1
MINDIE_LOG_TO_STDOUT	MindIE Service日志是否打印到标准输出。	• 0：不打印 • 1：打印	0
DYNAMIC_AVERAGE_WINDOW_SIZE	/metrics-json接口中，动态统计指标平均值的动态窗口大小。	正数	1000
MIES_SERVICE_MONITOR_MODE	是否开启推理服务化的在线监控指标，开启时才可以正常请求/metrics接口。	• 0：关闭 • 1：开启	0
PD_MODE	PD分离模式。	• 0：OFF_SWITCH，关闭PD切换模式。 • 1：REQ_LEVEL_SWITCH，P节点请求级别PD切换模式。 • 2：ITER_LEVEL_SWITCH，迭代级别PD切换，从decode队列预取sequence到publish队列。 • 3：NON_SPLITWISE，标准推理Prefill。	0
MINDIE_LOG_TO_FILE	MindIE Service日志是否打印到文件。	• 0：不打印 • 1：打印	1

参数名称	参数说明	取值范围	缺省值
MINDIE_LLM_RECOMPUTE_THRESHOLD	触发重计算的阈值，阈值越大越不容易触发重计算。	[0, 1)	0.5
MINDIE_LLM_FRAMEWORK_BACKEND	模型框架类型。	• ATB：推理引擎后端为加速库。 • MS：推理引擎后端为MindSpore。	ATB
LOCAL_DEVICE_PORT	所有device监听的端口。	1024~65535	1234
MINDIE_LLM_PYTHON_LOG_MAXSIZE	日志最大大小。	正数	1073741824
MINDIE_LLM_PYTHON_LOG_MAXNUM	日志最大数量。	正数	10
LOCAL_IMAGE_CACHE_DIR	收到多模态请求后，通过该环境变量来指定图片的暂存路径。	路径参数。	${MIES_INSTALL_PATH}/bin/cache

- MindIE Benchmark的环境变量如下所示：

参数名称	参数说明	取值范围	缺省值
CONFIG_PATH	MindIE Benchmark接口路径参数。 MindIE Server的安装路径。	-	$MIES_INSTALL_PATH
MIES_PYTHON_BENCHMARK_PATH	Python接口路径参数。 MindIE Server的安装路径。	-	$MIES_INSTALL_PATH

- MindIE MS的环境变量如下所示：

参数名称	参数说明	取值范围	缺省值
MINDIEMS_LOG_LEVEL	用户可动态设置MindIE MS客户端输出的日志等级。	• DEBUG • INFO • WARNING • ERROR • CRITICAL	默认值为空，设置为**表8-18**中log_level参数的日志等级。
HOME	用户动态设置MindIE MS客户端msctl.json配置文件的路径。	存在可读取的*{$HOME}*/.mindie_ms/msctl.json文件，详情请参考**表3-5**。	• root用户：默认值为/root。 • 非root用户：默认值为/*{$HOME}*/*{非root用户名}*。
MINDIE_MS_SERVER_IP	MindIE MS服务端容器化部署时容器的Pod IP地址。	取值必须为部署的容器IP，需与**步骤3.1**样例中"- name: MINDIE_MS_SERVER_IP"部分的格式保持一致。	默认为MindIE MS服务端容器化部署时容器Pod IP的地址。

10.2 RESTful 响应状态码

错误码	错误说明	返回结果
200	success	OK

404	URL not found	{
		"error": *具体的错误信息字符串*
		}

408	Timeout error	{
		"error": *具体的错误信息字符串*
		}

422	Input validation error	{
		"error": *具体的错误信息字符串*
		}

424	Generation error	{
		"error": *具体的错误信息字符串*
		}

错误码	错误说明	返回结果

429	Model is overloaded	{
		"error": *具体的错误信息字符串*
		}

500	Incomplete generation	{
		"error": *具体的错误信息字符串*
		}

10.3 数据集使用

以下是两个用于性能测试的常见数据集，在此提供两个脚本用于自动化加载模型，将数据集转换为token ID。需要注意OA数据集的平均SequenceLen较长，总量超过三千条，在模型体量大（65B及以上）而服务化配置的MaxBatchSize较小时，跑完整个数据集耗时久，可能需要数个小时。

OA 数据集

1. 单击**链接**获取OA数据集。
2. 转换为token ID方式。

   使用tokenizer_model.encode进行加密。

   python脚本示例参考如下：

```
import csv
from pathlib import Path
import pyarrow.parquet as pq
import glob, os
from transformers import AutoTokenizer
def read_oa(dataset_path, tokenizer_model):
    out_list = []
    for file_path in glob.glob((Path(dataset_path) / "*.parquet").as_posix()):
        file_name = file_path.split("/")[-1].split("-")[0]
        data_dict = pq.read_table(file_path).to_pandas()
        data_dict = data_dict[data_dict['lang'] == 'zh']
        ques_list = data_dict['text'].to_list()
        for ques in ques_list:
            tokens = tokenizer_model.encode(ques)
            if len(tokens) <= 2048:
                out_list.append(tokens)
            else:
                out_list.append(tokens[0:2048])
    return out_list
def save_csv(file_path, out_tokens_list):
    with open(file_path, 'w', newline='') as csvfile:
        csv_writer = csv.writer(csvfile)
        for row in out_tokens_list:
            csv_writer.writerow(row)
if __name__ == '__main__':
    model_path = "/data/models/baichuan2-7b"
    oa_dir = "/home/xxx/oasst1"
    save_path = "oa_tokens.csv"
    tokenizer_model = AutoTokenizer.from_pretrained(model_path, trust_remote_code=True,
use_fast=True, local_files_only=True)
    tokens_lists = read_oa(oa_dir, tokenizer_model)
    save_csv(save_path, tokens_lists)
```


数据集获取

数据集的获取方式请参见/usr/local/Ascend/llm_model/tests/modeltest/README_NEW.md。

支持的数据集如下所示：

- BoolQ
- CEval
- CMMLU
- HumanEval
- HumanEval_X
- GSM8K
- LongBench
- MMLU
- NeedleBench
- TruthfulQA

GSM8K 数据集转 tokenids

使用pandas read_json后，然后使用tokenizer直接转换，再用numpy保存到csv中。

python脚本示例如下：

```
import numpy as np
import pandas as pd
from transformers import AutoTokenizer

MODEL_PATH = "/home/weight/llama2-70b"
OUT_FILE = "token_gsm8k.csv"
tokenizer = AutoTokenizer.from_pretrained(MODEL_PATH, trust_remote_code=True, use_fast=True, local_files_only=True)

def gen_requests_from_trace(trace_file):
    len = 0
    with open(OUT_FILE, "w") as f:
        df = pd.read_json(trace_file, lines=True)
        for i, row in df.iterrows():
            ques = row["question"]
            token = tokenizer([ques], return_tensors="np")
            token: np.ndarray = token["input_ids"].astype(np.int64)
            np.savetxt(f, token, fmt="%d", delimiter=",")
            len+=token.shape[-1]
    print(len / 1319)

if __name__ == '__main__':
    gen_requests_from_trace("./GSM8K.jsonl")
```

10.4 后处理参数说明

10.4.1 benchmark 的 client 模式的文本流式推理后处理参数

参数	是否必选	说明	取值要求
do_sample	可选	是否做sampling。	bool类型，默认值false。 ● true：做sampling。 ● false：不做sampling。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int32_t类型，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]&&[0, vocabSize)，默认值0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值-1.0。 字段未设置时，默认值使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

10.4.2 benchmark 的 client 模式的文本非流式推理后处理参数

参数	是否必选	说明	取值要求
do_sample	可选	是否做sampling。	-

参数	是否必选	说明	取值要求
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	int32_t类型，默认值20。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。

参数	是否必选	说明	取值要求
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	int32_t类型，取值范围[0, 2147483647]&&[0, vocabSize)，默认值0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0]，默认值1.0。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值-1.0。 字段未设置时，默认值使用-1.0来表示不进行该项处理，但是不可主动设置为-1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

10.4.3 benchmark 的 client 模式的 token 推理后处理参数

参数名	是否必选	说明	取值要求
temperature	可选	控制生成的随机性，较高的值会产生更多样化的输出。	float类型，大于0.0，默认值1.0。 取值越大，结果的随机性越大。推荐使用大于或等于0.001的值，小于0.001可能会导致文本质量不佳。 建议最大值取2.0，同时视模型而定。
top_k	可选	控制模型生成过程中考虑的词汇范围，只从概率最高的k个候选词中选择。	uint32_t类型，取值范围(0, vocabSize)&&(0, 2147483647]，默认值0。 字段未设置时，默认值使用0来表示不进行该项处理，但是不可主动设置为0。 vocabSize是从modelWeightPath路径下的config.json文件中读取的vocab_size或者padded_vocab_size的值，若不存在则vocabSize取默认值0。建议用户在config.json文件中添加vocab_size或者padded_vocab_size参数，否则可能导致推理失败。
top_p	可选	控制模型生成过程中考虑的词汇范围，使用累计概率选择候选词，直到累计概率超过给定的阈值。该参数也可以控制生成结果的多样性，它基于累积概率选择候选词，直到累计概率超过给定的阈值为止。	float类型，取值范围(0.0, 1.0)，默认值1.0。 字段未设置时，默认值使用1.0来表示不进行该项处理，但是不可主动设置为1.0。
max_new_tokens	可选	允许推理生成的最大token个数。实际产生的token数量同时受到配置文件maxIterTimes参数影响，推理token个数小于或等于Min(maxIterTimes, max_new_tokens)。	uint_64类型，默认值20。
do_sample	可选	是否做sampling。	-

参数名	是否必选	说明	取值要求
seed	可选	用于指定推理过程的随机种子，相同的seed值可以确保推理结果的可重现性，不同的seed值会提升推理结果的随机性。	uint_64类型，取值范围(0, 18446744073709551615]，不传递该参数，系统会产生一个随机seed值。 当seed取到临近最大值时，会有WARNING，但并不会影响使用。若想去掉WARNING，可以减小seed取值。
repetition_penalty	可选	重复惩罚用于减少在文本生成过程中出现重复片段的概率。它对之前已经生成的文本进行惩罚，使得模型更倾向于选择新的、不重复的内容。	float类型，大于0.0，默认值1.0。 ● 小于1.0表示对重复进行奖励。 ● 1.0表示不进行重复度惩罚。 ● 大于1.0表示对重复进行惩罚。 建议最大值取2.0，同时视模型而定。
typical_p	可选	解码输出概率分布指数。 当前后处理不支持。	float类型，取值范围(0.0, 1.0]，默认值1.0。
watermark	可选	是否带模型水印。 当前后处理不支持。	bool类型，默认值false。 ● true：带模型水印。 ● false：不带模型水印。

10.5 术语&缩略语

术语/缩略语	含义
LLM	Large Language Model，大语言模型。
TGI	Text Generation Inference，文本生成推理。是一个用于部署和服务大型语言模型的工具包。TGI为最流行的开源LLM提供高性能文本生成，包括Llama、Falcon、StarCoder、BLOOM、GPT-NeoX等。
vLLM	vLLM是一个开源的大模型推理加速框架。
Trition	Triton是一个开源的推理服务软件，全称为Triton Inference Server。通过Triton，您可以在基于GPU或CPU的各种基础架构（云、数据中心或边缘）上部署、运行和扩展来自任何框架的AI模型。

术语/缩略语	含义
CB	Continuous Batching，连续批处理。
PA	Paged Attention，PA是一种用于处理长序列数据的注意力机制。
RoCE	RDMA over Converged Ethernet，RoCE是一种网络协议，允许通过以太网使用远程直接内存访问（RDMA）。目前存在两个RoCE版本，分别是RoCE v1和v2。RoCE v1是数据链路层协议，允许在同一个以太网广播域内的任意两台主机之间通信。RoCE v2是网络层协议，其报文可以被路由。
GMIS	General Model Inference Scheduler，是一个用于模型推理的调度器。它在大型模型训练中起着关键作用，旨在减少计算资源的空闲时间，提高计算资源的利用率，从而加快模型训练和模型推理的进度模型推理调度器，提供各种模型调度能力。
Daemon	Daemon（守护进程）是运行在后台的一种特殊进程。它独立于控制终端并且周期性地执行某种任务或等待处理某些发生的事件。它不需要用户输入就能运行，同时提供某种服务，不仅对整个系统，还可以对某个用户程序提供服务。
EndPoint	推理服务化协议和接口封装，兼容Triton/OpenAI/TGI/vLLM等第三方框架接口。
KMC	Key Management Center，密钥管理系统。用于管理和保护加密算法中使用的密钥。它可以为企业或组织提供安全的密钥存储、密钥分发、密钥轮换、密钥备份和密钥恢复等功能。KMC秘钥库可以确保密钥的安全性和可靠性，防止密钥泄露、丢失或被篡改。同时，KMC秘钥库还可以支持多种加密算法和密钥长度，满足不同应用场景的需求。
GRPC	Google Remote Procedure Call，Google远程过程调用协议。
GCC	GNU Compiler Collection，GNU编译器集。
业务面	MindIE Server推理等业务接口所处的平面，在通信矩阵体现为数据面。
管理面	MindIE Server健康状态信息接口所处的平面。

