目录

[MindSpore Serving服务接口说明（r2.1） 3](#_Toc151482131)

[1 模型及组batch配置 3](#_Toc151482132)

[1.1 prefill\_model\_path: 3](#_Toc151482133)

[1.2 decode\_model\_path: 3](#_Toc151482134)

[1.3 ctx\_path: 3](#_Toc151482135)

[1.4 inc\_path: 4](#_Toc151482136)

[1.5 topk\_model： 4](#_Toc151482137)

[1.6 argmax\_model: 4](#_Toc151482138)

[1.7 post\_model\_ini: 4](#_Toc151482139)

[1.8 device: 5](#_Toc151482140)

[1.9 tokenizer\_path: 5](#_Toc151482141)

[1.10 end\_token: 5](#_Toc151482142)

[1.11 vocab\_size: 5](#_Toc151482143)

[1.12 dyn\_batch\_size: 5](#_Toc151482144)

[1.13 batch\_size: 6](#_Toc151482145)

[1.14 prefill\_batch\_size: 6](#_Toc151482146)

[1.15 prefill\_batch\_size: 6](#_Toc151482147)

[1.16 tokenizer: 6](#_Toc151482148)

[1.17 Input\_function: 6](#_Toc151482149)

[1.18 Batching\_staeategy: 7](#_Toc151482150)

[1.19 decode\_batch\_waiting\_time: 7](#_Toc151482151)

[1.20 batch\_waiting\_time: 7](#_Toc151482152)

[1.21 model\_type: 7](#_Toc151482153)

[1.22 ModelName: 8](#_Toc151482154)

[2 Serving服务配置 8](#_Toc151482155)

[2.1 SERVER\_APP\_PORT 8](#_Toc151482156)

[2.2 SERVER\_APP\_HOST 9](#_Toc151482157)

[2.3 AgentIP 9](#_Toc151482158)

[2.4 AgentPorts 9](#_Toc151482159)

[2.5 getRouteByStartEndLink 9](#_Toc151482160)

[3 client请求入参配置 11](#_Toc151482161)

[3.1 非流式推理 12](#_Toc151482162)

[3.2 使能Sample推理 13](#_Toc151482163)

[3.3 关闭Sample推理 14](#_Toc151482164)

[4 Serving推理API接口 14](#_Toc151482165)

[4.1 generate\_answer(request\_id, \*\*params) 14](#_Toc151482166)

[5 Serving状态获取接口 16](#_Toc151482167)

[5.1 获取当前时刻的batch\_size大小： 16](#_Toc151482168)

[5.2 获取当前时刻请求队列里还在等待推理的请求数 16](#_Toc151482169)

[6 错误码 16](#_Toc151482170)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 版本 | 修订时间 | 修订人 | 修订类型 | 修订章节 | 修订内容 |
| A0.1 | 2023-11-21 | 舒驰 | A | 大纲 | 基本接口内容 |
| A0.2 | 2023-11-22 | 郭诗鹏 | A | 大纲 | 1. 增加接口入参取值说明 |

MindSpore Serving服务接口说明（r2.1）

# 模型及组batch配置

配置参数位置config/serving\_config.py：

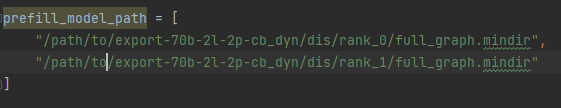
## prefill\_model\_path:

数据类型：List[str]

说明：全量模型的权重mindir路径，List长度为模型的切分数；

使用示例：切分数为2的2卡模型路径

类型：必要参数，用户必须传入



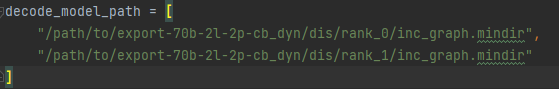
## decode\_model\_path:

数据类型：List[str]

说明：增量模型的权重mindir路径，List长度为模型的切分数；

类型：必要参数，用户必须传入

使用示例：切分数为2的2卡模型路径，见下图：



## ctx\_path:

数据类型：str

说明：全量模型的配置文件路径，具体ini配置参考lite模型的配置说明；

类型：必要参数，用户必须传入

使用示例：见下图：



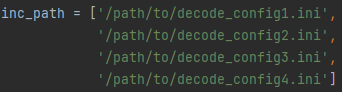
## inc\_path:

数据类型：list(str)

说明：增量模型的配置文件路径，具体ini配置参考lite模型的配置说明, 多个档位的配置用 ’ , ’分开；

类型：必要参数，用户必须传入；

使用示例：见下图：



## topk\_model：

数据类型：str

说明：topk模型的权重mindir路径；

类型：必要参数，用户需通过post\_sampling\_model.py脚本导出后处理图后，指导导出图的位置；

使用示例：见下图;



## argmax\_model:

数据类型：str

说明：argmax模型的权重mindir路径；

类型：必要参数，用户需通过post\_sampling\_model.py脚本导出后处理图后，指导导出图的位置；

使用示例：见下图：



## post\_model\_ini:

数据类型：str

说明：后处理模型（argmax, topk共用一个）的配置文件路径，具体ini配置参考lite模型的配置说明；

类型：必要参数，用户需通过post\_sampling\_model.py脚本导出后处理图后，指导导出图的位置；

使用示例：见下图;



## device:

数据类型：int

说明：模型加载的其实NPU卡号，如4卡模型，用4567的NPU做推理，device=4；

类型：必要参数，需用户查询空闲卡号后配置；

使用示例：见下图;



## tokenizer\_path:

数据类型：str

说明：分词器模型路径，用户自定义；

类型：必要参数，用户自定义，根据实际分词器模型路径；

使用示例：见下图;



## end\_token:

数据类型：int

说明：分词器中eos token（结束符）的id；

类型：必要参数，用户自定义，根据实际分词器词表情况填写；

使用示例：见下BaseConfig图示;

## vocab\_size:

数据类型：int

说明：分词器的词表长度；

类型：必要参数，用户自定义，根据实际分词器词表情况填写；

使用示例：见下BaseConfig图示;

## dyn\_batch\_size:

数据类型：List[int]

说明：decode动态分档的档位；

类型：必要参数，用户自定义，根据实际模型的配置情况填写；如果模型不支持动态分档，填写一个固定batch的值, e.g. 'dyn\_batch\_size': [8]

使用示例：见下BaseConfig图示;

## batch\_size:

数据类型：int

说明：decode静态模型配置，动态模型可忽略，保持默认值=8；

类型：必要参数，用户自定义，根据实际模型的配置情况填写；

使用示例：见下BaseConfig图示;

## prefill\_batch\_size:

数据类型：int

说明：prefill静态模型batch\_size配置，保持默认值=1；

类型：必要参数，用户自定义，根据实际模型的配置情况填写；

使用示例：见下BaseConfig图示;

## tokenizer:

数据类型：str

说明：tokenizer名称，用于加载分词器，参考MindFormers中支持的分词器name；

类型：必要参数，用户自定义，根据实际使用的模型情况填写；

使用示例：见下BaseConfig图示;

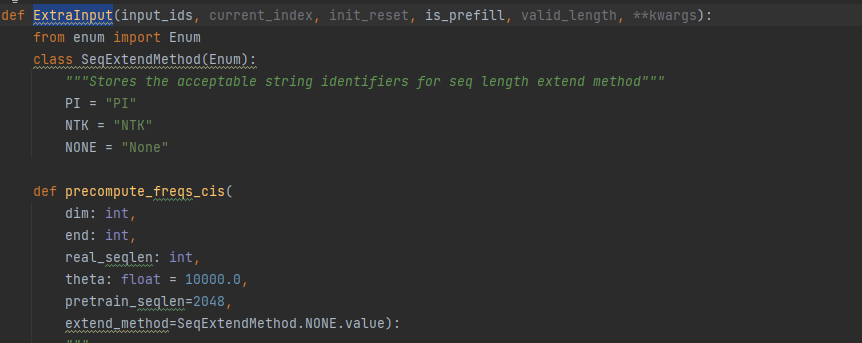
## Input\_function:

数据类型：str

说明：用于构造模型的输入，支持用户自定义模型输入接口，默认值为’common’,用户自定义则使用’custom’；

类型：必要参数，用户自定义，根据实际使用的模型情况填写,如果自定义，需要自己定义ExtraInput方法，模板见serving\_config.py；

使用示例：见下BaseConfig图示;



## Batching\_staeategy:

数据类型：str

说明：组batch的策略，当前支持static batch和continuous batching两种模式，static batch该参数设置为’static’， continuous batching设置为’continuous’

类型：必要参数，用户自定义，根据模型的特性来定义该参数，需要检查模型是否支持continuous batching；

使用示例：见下BaseConfig图示；

## decode\_batch\_waiting\_time:

数据类型：float

说明：做调度的等待时间，当请求队列中没有请求时，等待一段时间再查询，等待时间由客户自己的业务逻辑决定，默认值为0.0，则查询一次后，直接做下一次decode推理，等下一次decode结束后，再查询；

类型：必要参数，用户自定义，用户根据自生业务逻辑决定；

使用示例：见下BaseConfig图示；

## batch\_waiting\_time:

数据类型：float

说明：做调度的等待时间，当请求队列中没有请求时，等待一段时间再查询，等待时间由客户自己的业务逻辑决定，默认值为0.0，则查询一次后，直接做下一次prefill推理，等下一次prefill结束后，再查询；

类型：必要参数，用户自定义，用户根据自生业务逻辑决定；

使用示例：见下BaseConfig图示；

## model\_type:

数据类型：int

说明：模型类型，用于构造输入使用，针对动态seq, 静态seq输入设置为1，动态设置为0；

类型：必要参数，用户自定义，根据实际部署的模型决定；

使用示例：见下BaseConfig图示；

## ModelName:

数据类型：str

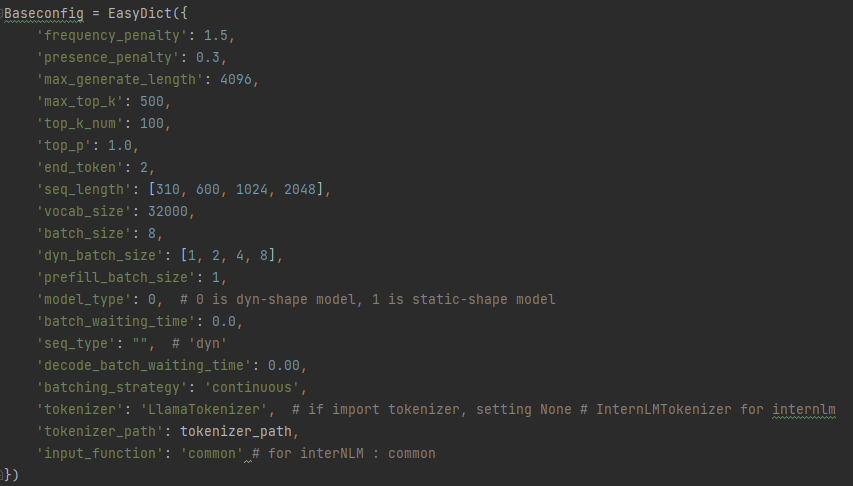
说明：模型名称，用于不通模型的构造输入，llama默认为：’llama\_\*’,

书生为：‘internlm\_\*’

类型：必要参数，用户自定义，根据实际部署的模型决定；

使用示例：见下图示；



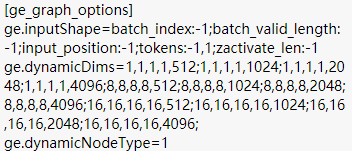


## zactivate\_len:

数据类型：list[int]

说明：适配act len优化

类型：必要参数，用户自定义，根据实际部署的模型决定；

使用示例： [512, 1024, 2048, 4096] 配合ini使用，decode.ini增加

# Serving服务配置

## SERVER\_APP\_PORT

数据类型：int

说明：serving服务的端口号设置，通过该端口发送请求或获取serving状态；

类型：必要参数，需用户指定端口号；

使用示例：见下图;



## SERVER\_APP\_HOST

数据类型：int

说明：serving服务的ip号设置，通过该ip发送请求或获取serving状态；

类型：必要参数，需用户指定IP，默认为’localhost’；

使用示例：见下图；



## AgentIP

数据类型：str

说明：agent服务的ip号设置，通过该ip号与serving通信；

类型：必要参数，需用户指定ip号；

使用示例：见下图;



## AgentPorts

数据类型List[int]

说明：agent服务的端口号设置，通过该端口号与serving通信，list长度为模型的并行卡数（切分数）；

类型：必要参数，需用户指定端口号；

使用示例：见下图;

## getRouteByStartEndLink

函数名

|  |  |
| --- | --- |
| **String** | **getRouteByStartEndLink** (**SDKRequestBean** requestBean) |

入参同

|  |  |
| --- | --- |
| 入参 | [GuideService请求RouteService接口](#_GuideService请求RouteService接口) |
| 出参 | String |

# client请求入参配置

## 流式推理

每次推理请求结果按照单个单词依次返回，最后支持返回所有生成单词。

URI

* URI格式

POST /models/{mode\_type}/generate\_stream

* 参数描述

| 参数 | 是否必选 | 类型 | 描述 |
| --- | --- | --- | --- |
| inputs | 是 | String | 用户输入提示 |
| parameters | 是 | Json | 生成参数 |

请求

* 请求样例

POST https://endpoint/models/{mode\_type}/generate\_stream

{

"inputs":"what is Monetary Policy?",

"parameters":{

"max\_new\_tokens":20,

"temperature":1,

"top\_k":1,

"top\_p":1,

"do\_sample":"False",

"repetition\_penalty":1,

"return\_full\_text":"True"

}

* 参数说明

| 参数 | 是否必选 | 类型 | 描述 | 取值 |
| --- | --- | --- | --- | --- |
| do\_sample | 否 | bool | 后处理是否采样 | 默认值为False |
| max\_new\_tokens | 否 | int | 最大生成数量 | 默认值为300 |
| repetition\_penalty | 否 | float | 重复单子惩罚参数 | 默认值为1.0 |
| temperature | 否 | float | 后处理随机性参数 | 默认值为1.0 |
| top\_k | 否 | int | 后处理top\_k参数 | 默认值为3，如果请求给的top\_k < 0，调整为0 |
| top\_p | 否 | float | 后处理top\_p参数 | 默认值为1.0，如果请求给的top\_p < 0.01，调整为0.01,top\_p > 1.0 调整为1.0 |
| return\_full\_text | 否 | bool | 最后是否返回所有生成 | 默认值为True |
| return\_protocol | 否 | str | 默认使用sse协议返回,其他情况均使用fastapi的流式返回（StreamingResponse） | 默认值为’sse’ |

响应

* 流式中间响应样例

data:{

"token": {"text": "xx"}

}

1. 流式最后响应样例

data:{

"generated\_text": "xxxxxxxxxxxxxxxxxxxxxxxx"

}

## 非流式推理

功能介绍

每次推理请求结果一次返回所有生成单词。

URI

* URI格式

POST /models/{mode\_type}/generate

* 参数描述

| 参数 | 是否必选 | 类型 | 描述 |
| --- | --- | --- | --- |
| inputs | 是 | String | 用户输入提示 |
| parameters | 是 | Json | 生成参数 |

* Parameters字段描述

请求

* 请求样例

POST https://endpoint/models/{mode\_type}/generate

{

"inputs":"what is Monetary Policy?",

"parameters":{

"max\_new\_tokens":20,

"temperature":1,

"top\_k":1,

"top\_p":1,

"do\_sample":"False",

"repetition\_penalty":1

}

* 参数说明

| 参数 | 是否必选 | 类型 | 描述 | 取值（同流式请求） |
| --- | --- | --- | --- | --- |
| do\_sample | 否 | bool | 后处理是否采样 | - |
| max\_new\_tokens | 否 | int | 最大生成数量 | - |
| repetition\_penalty | 否 | float | 重复单子惩罚参数 | - |
| temperature | 否 | float | 后处理随机性参数 | - |
| top\_k | 否 | int | 后处理top\_k参数 | - |
| top\_p | 否 | float | 后处理top\_p参数 | - |

响应

* 响应样例

{

"generated\_text": "xxxxxxxxxxxxxxxxxxxxxxxx"

}

## 使能Sample推理

参数设置

“do\_sample”设置为“True”

请求

* 请求样例

POST https://endpoint/models/{mode\_type}/generate

{

"inputs":"what is Monetary Policy?",

"parameters":{

"max\_new\_tokens":20,

"temperature":1,

"top\_k":1,

"top\_p":1,

"do\_sample":"True",

"repetition\_penalty":1

}

## 关闭Sample推理

参数设置

“do\_sample”设置为“False”

请求

* 请求样例

POST https://endpoint/models/{mode\_type}/generate

{

"inputs":"what is Monetary Policy?",

"parameters":{

"max\_new\_tokens":20,

"do\_sample":"False",

}

# Serving推理API接口

## generate\_answer(request\_id, \*\*params)

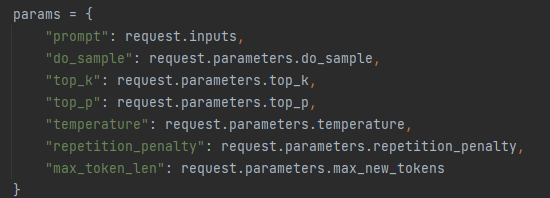
使用方法：

1、实例化llm\_server = LLMServer()

2、准备好请求request\_id及入参，params;

3、results = llm\_server.generate\_answer(request\_id, \*\*params)

4、params如下：



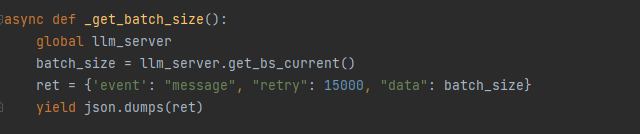
5、返回值为async\_generator对象；

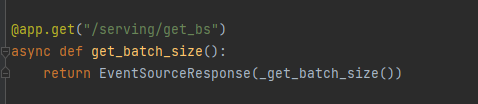
6、使用API方式调用的，可以将async\_generator对象解析成流式的或非流式的结果，参照代码clinet/server\_app\_post.py, 以下是以sse协议封装的用例。



# Serving状态获取接口

## 获取当前时刻的batch\_size大小：



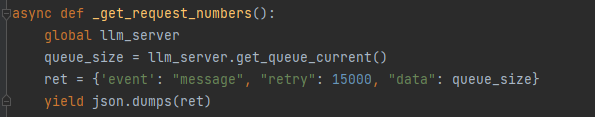


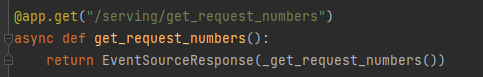
以sse的形式提供，同流式和非流式的接收方法：



## 获取当前时刻请求队列里还在等待推理的请求数

同5.1





以sse的形式提供，同流式和非流式的接收方法：



# 错误码

目前包含的错误码总结如下表：

|  |  |
| --- | --- |
| 错误码返回值 | 错误原因 |
| 202 | 分词长度校验，超过模型输入 |
| 203 | 解分词超过模型支持范围 |
| 204 | 入参超过边界值，返回警告和默认值 |
| 301 | 请求队列超过了设定阈值，返回警告 |