chat

使用tigerbot模型进行chat completion

Example request

python

POST https://api.tigerbot.com/v1/chat/completions

单轮对话 single turn

```python
import requests
import json

API_KEY="{your api_key}"
url = "https://api.tigerbot.com/v1/chat/completions"

headers = {
    'Authorization': 'Bearer ' + API_KEY
}

payload = {
    "model": "tigerbot-70b-chat",
    "query": "中国的首都在哪里"
}

response = requests.post(url, headers=headers, json=payload)
print(json.dumps(response.json(), ensure_ascii=False, indent=4))
```

response

```
{
  "result": "北京。北京是中国的首都，中国政治、文化和国际交往的中心。",
  "input_tokens": 6,
  "total_tokens": 22
}
```

多轮对话 multi-turn

Request header

Authorization string required

用户唯一标识 Api_Key

Request Body

model string optional

使用的模型id，支持`tigerbot-13b-chat`,`tigerbot-70b-chat`

query string required

用来产生对话回复或补全的prompt输入

session array optional Default to []

多轮对话中的历史对话输入

- human string optional
  用户问句
- assistant string optional
  模型回答

max_input_tokens integer optional Default to 1024

输入的最大 tokens 数量，包含`query`和`session`中对话的长度，70b 最大`37888` tokens，13b 最大`30000` tokens

max_output_tokens integer optional Default to 1024

```python
import requests
import json

API_KEY="{your api_key}"
url = "https://api.tigerbot.com/v1/chat/completions"

headers = {
    'Authorization': 'Bearer ' + API_KEY
}

payload = {
    "query": "那英国的呢",
    "session": [{"human": "法国的首都在哪里", "assistant": "巴黎。"}]
}

response = requests.post(url, headers=headers, json=payload)
print(json.dumps(response.json(), ensure_ascii=False, indent=4))
```

response

```
{
  "result": "伦敦",
  "input_tokens": 13,
  "total_tokens": 15
}
```

使用 internet 插件

```python
import requests
import json

API_KEY="{your api_key}"
url = "https://api.tigerbot.com/v1/chat/completions"
```

模型输出的 tokens 数量，70b 最大8192 tokens，13b 最大8192 tokens

do_sample boolean optional

是否使用 sampling 方法生成

temperature number optional Default to 1

设置 sampling temperature，do_sample 必须设定 true。 temperature 的范围在0到2之间，高的temperature，比如 0.8，会使模型的生成的结果更加随机，而低的temperature，比如0.2，会使模型生成的结果更加稳定。

top_p number optional Default to 1

设置 nucleus sampling，do_sample 必须设定 true。模型生成的top_p probability mass tokens，top_p为0.1表示只生成概率超过10%的tokens，top_p范围在0-1之间。

stream boolean optional

是否选择流式输出，流式输出可参考示例代码

internet boolean optional

是否开启实时搜索

internet_config object optional

实时搜索设置，只有当internet为true时该配置才会生效

prompt_prefix string optional

输入模型的prompt拼接前缀

search_mode string optional Default to STRICT

搜索模式，当前可以用mode：STRICT

functions array optional

函数列表，供 function calling 功能使用

name string optional

```
headers = {
    'Authorization': 'Bearer ' + API_KEY
}

payload = {
    "model": "tigerbot-70b-chat",
    "query": "今天上海天气怎么样，多少度",
    "internet": True
}

response = requests.post(url, headers=headers, json=payload)
print(json.dumps(response.json(), ensure_ascii=False, indent=4))
```

response

```
{
  "result": "今天上海天气多云，最高气温34度，最低气温27度。",
  "input_tokens": 49,
  "total_tokens": 66
}
```

流式输出

使用流式输出请先安装`sseclient-py`， 安装命令 `pip install sseclient-py`

```
import json
import requests
import sseclient

API_KEY="{your api_key}"
url = 'https://api.tigerbot.com/v1/chat/completions'
headers = {
    'Accept': 'text/event-stream',
    'Content-Type': 'application/json',
    'Authorization': 'Bearer ' + API_KEY
```

函数名。

description string optional

函数描述。

parameters object optional

函数参数格式。使用 JSON Schema 定义。

function_call string optional

控制函数的使用。`none` 表示不使用 function calling 功能，`auto` 表示由模型自动选择函数。`functions` 为空时默认为 `none`，functions不为空时默认为auto`。

Response

result string

模型chat生成结果

input_tokens integer

输入的tokens数量

total_tokens integer

总消耗tokens数量

finished boolean

如果`stream`为`true`， `finished` 代表流式输出是否结束

new_text string

如果`stream`为`true`， `new_text` 代表流式输出当前chunk的输出

function_call object

name string

```
}
payload = {
    "query": "旅游行业有哪些创新机会，写一篇500字左右的研究报告。",
    "model": "tigerbot-70b-chat",
    "stream": True,
}

response = requests.post(url, stream=True, headers=headers, json=payload)
response.raise_for_status()
client = sseclient.SSEClient(response)
for event in client.events():
    print(json.loads(event.data))
```

response

```
{'finished': False, 'new_text': '旅游'}
{'finished': False, 'new_text': '行业'}
{'finished': False, 'new_text': '是一个'}
{'finished': False, 'new_text': '充满'}
{'finished': False, 'new_text': '活'}
{'finished': False, 'new_text': '力和'}
{'finished': False, 'new_text': '机会'}
...
{'finished': True, 'input_tokens': 16, 'result': '旅游行业是一个充满活力和机会的领域，随着全球经济的不断发展，旅游业也在不断壮大。在这样的背景下，旅游行业也面临着许多创新机会，这些机会可以为旅游行业带来新的发展动力，提高旅游行业的服务质量和竞争力。以下是一些旅游行业创新机会的研究报告。\n1. 旅游社交媒体\n随着社交媒体的不断发展，人们越来越喜欢在社交媒体上分享自己的旅游经历和经验。因此，旅游公司可以开发旅游社交媒体平台，为旅游者提供旅游信息和旅游建议，同时也可以为旅游者提供在线预订和支付服务。这种平台可以为旅游公司提供更多的销售机会，同时也可以为旅游者提供更好的旅游体验。\n2. 旅游虚拟现实\n虚拟现实技术在游戏和娱乐行业已经得到广泛应用，但是它也可以在旅游行业中发挥重要作用。旅游公司可以开发旅游虚拟现实平台，为旅游者提供身临其境的旅游体验。这种平台可以为旅游者提供更真实、更生动的旅游体验，同时也可以为旅游公司提供更多的销售机会。\n3. 旅游智能化\n随着人工智能和物联网技术的发展，旅游行业也可以实现智能化。旅游公司可以开发旅游智能化系统，为旅游者提供更智能、更个性化的旅游服务。这种系统可以为旅游者提供更准确、更及时的旅游建议和预订服务，同时也可以为旅游公司提供更准确的销售预测和市场分析。\n4. 旅游共享经济\n共享经济已经成为现代社会的一种趋势，旅游行业也可以利用这种趋势。旅游公司可以开发旅游共享经济平台，为旅游者提供共享旅游服务。这种平台可以为旅游者提供更经济、更环保的旅游方式，同时也可以为旅游公司提供更广泛的销售机会。\n5. 旅游区块链\n区块链技术在金融和支付领域已经得到广泛应用，但是它也可以在旅游行业中发挥重要作用。旅游公司可以开发旅游区块链平台，为旅游者提供更安全、更透明的旅游服务。这种平台可以为旅游者提供更安全、更透明的旅游服务，同时也可以为旅游公司提供更准确的销售预测和市场分析。\n综上所述，旅游行业是一个充满活力和机会的领域，旅游公司可以开发旅游社交媒体、旅游虚拟现实、旅游智能化、旅游共享经济和旅游区块链等创新机会，为旅游行业带来新的发展动力，提高旅游行业的服务质量和竞争力。', 'total_tokens': 521}
```

使用函数调用（Function Calling）功能

函数调用（Function Calling）提供从用户查询中提取指定函数的指定参数的功能。该功能可以方便地与你提供的函数（如第三方 API）进行集成，让用户使用自然语言调用函数。

一般通过以下流程实现：

1. 获取用户的查询式（query），将查询式以及你的自定义函数信息发送请求至 Tigerbot Chat API。Tigerbot 将根据你提供的函数信息，从用户的查询式中提取出符合函数参数定义的值返回。

使用的函数名称。

arguments string

从 `query` 中提取出的，符合 `parameters` 所定义格式的函数参数。以 JSON 字符串的形式给出。

2. 你根据 Tigerbot 返回的函数参数，调用你的自定义函数，获得自定义函数的返回值。
3. 将返回值、查询式（query）、函数信息等作为内容发送请求至 Tigerbot Chat API，Tigerbot 将根据你提供的信息生成回答。

示例

假设我作为一个服务提供方，我编写了一个函数 `eval_math` 用于计算数学表达式的值。 这个函数有 `expression` 这个参数，表示数学表达式。 用户输入的是自然语言""计算 1+1"，没有直接给出函数参数的具体值，因此无法直接调用函数 `eval_math`。 幸好有 TigerBot ，此时可以利用 TigerBot 提供的 "函数调用" 功能从用户的输入中提取出指定函数所需参数的值，在此处为 `eval_math` 函数的 `expression` 参数。 "函数调用"功能将为我提取出正确的参数值，`{"expression": "1+1"}`，得到这些参数值后我立即可以调用 `eval_math` 并获取到结果。将 `eval_math` 的结果再次提交至 TigerBot，我将会得到自然的回答，"根据给出的数据，计算 1 + 1 的结果是 2。"。

1. 获取用户的输入，将用户输入以及你的自定义函数信息发送请求至 Tigerbot Chat API：

```
import requests

url = "https://api.tigerbot.com/v1/chat/completions"

headers = {
    'Authorization': 'Bearer ' + API_KEY
}

payload = {
    "model": "tigerbot-70b-chat",
    "query": "计算 1+1",
    "functions": [
        {
            "name": "eval_math",
            "description": "计算数学表达式的值",
            "parameters": {
                "type": "object",
                "properties": {
```

```
                    "expression": {
                        "type": "string",
                        "description": "数学表达式"
                    }
                },
                "required": [
                    "expression"
                ]
            }
        }
    ]
}

response = requests.post(url, headers=headers, json=payload)
print(response.json())
```

返回查询式中提取出符合函数参数定义的值：

```
{
  "function_call": {
    "name": "eval_math",
    "arguments": "{\"expression\": \"1+1\"}"
  },
  "input_tokens": 104,
  "total_tokens": 117
}
```

2. 根据 Tigerbot 返回的函数参数，调用你的自定义函数

   设此处的函数为（Python）：

```
def eval_math(expression: str):
    return {"result": eval(expression)}
```

调用 eval_math(expression="1+1")，将获得结果 {"result": 2}

将返回值、查询式（query）、函数信息等作为内容发送请求至 Tigerbot Chat API:

```
import requests

url = "https://api.tigerbot.com/v1/chat/completions"

headers = {
    'Authorization': 'Bearer ' + API_KEY
}

payload = {
    "model": "tigerbot-70b-chat",
    "query": "计算 1+1",
    "session": [
        {
            "function": "{\"result\": 2}"
        }
    ],
    "functions": [
        {
            "name": "eval",
            "description": "计算数学表达式的值",
            "parameters": {
                "type": "object",
                "properties": {
                    "expression": {
                        "type": "string",
                        "description": "数学表达式"
                    }
                },
                "required": [
                    "expression"
                ]
```

```
                }
            }
        ]
}

response = requests.post(url, headers=headers, json=payload)
print(response.json())
```

Tigerbot 将根据你提供的信息生成回答：

```
{
  "result": "根据给出的数据，计算 1 + 1 的结果是 2。",
  "input_tokens": 30,
  "total_tokens": 45
}
```