

# 02-调用第三方平台的模型

1、魔塔社区的模型

2、HuggingFace 服务器上的大模型

3、官方服务器的模型

## 1、魔塔社区的模型

官网地址: <https://community.modelscope.cn/>

The screenshot shows the ModelScope community platform interface. The main navigation bar includes '首页', '模型库', '数据集', '创空间', 'AIGC专区', '文档', '社区', 'MCP广场', and 'GitHub'. A search bar at the top right allows users to search for '感兴趣的内容'.

In the center, there is a modal window titled '推理 API-Inference beta'. It contains a code editor with Python code for interacting with the DeepSeek-R1 model. The code imports OpenAI and defines a client with a base URL and API key. It then creates a completion message with a user role and content. A red box highlights the '替代驼峰例' (Replace CamelCase Example) button in the bottom right of the code editor.

The background shows a sidebar with sections like '1. 介绍' (Introduction), '请注意, 图像和链接中的文本没有被' (Please note, the text in images and links has not been processed), and '我们介绍了我们的第一代推理模型 DeepSeek-R1, 它结合了大模型推理和' (We introduced our first generation inference model DeepSeek-R1, which integrates large model inference and). At the bottom of the sidebar, it says 'DeepSeek-R1 基于 Llama 和 Qwen 等推出的六个密集模型。DeepSeek-R1-Distill-Qwen-32B 在各种基准测试中超过了 OpenAI-o1-mini, 达到了密集模型的新'.

The right side of the interface shows a sidebar with 'Notebook快速开发' (Quick Notebook Development), '部署' (Deployment), and a list of 26 models. A footer at the bottom right indicates the page is from 'API-Inference in LLMs via ModelScope'.

```
1  from openai import OpenAI
2
3  client = OpenAI(
4      base_url='https://api-inference.modelscope.cn/v1/',
5      api_key='0206b945-8fa0-4bf2-bd17-ea3926ddbe97', # ModelScope Token
6  )
7
8  response = client.chat.completions.create(
9      model='deepseek-ai/DeepSeek-R1', # ModelScope Model-Id
10     messages=[
11         {
12             'role': 'user',
13             'content': '你好'
14         }
15     ],
16     stream=True
17 )
18 done_reasoning = False
19 for chunk in response:
20     reasoning_chunk = chunk.choices[0].delta.reasoning_content
21     answer_chunk = chunk.choices[0].delta.content
22     if reasoning_chunk != '':
23         print(reasoning_chunk, end='', flush=True)
24     elif answer_chunk != '':
25         if not done_reasoning:
26             print('\n\n === Final Answer ===\n')
27             done_reasoning = True
28         print(answer_chunk, end='', flush=True)
```

## 2、HuggingFace 服务器上的大模型

官网: <https://huggingface.co/>

The screenshot shows the Hugging Face platform's model card for 'deepseek-ai/DeepSeek-V3'. The main panel displays code examples for Python, JavaScript, and cURL. The Python example uses the 'requests' library to interact with the model. The code is as follows:

```
import os
from openai import OpenAI

client = OpenAI(
    base_url="https://router.huggingface.co/novita/v3/openai",
    api_key=os.environ["HF_TOKEN"],
)

completion = client.chat.completions.create(
    model="deepseek/deepseek-v3-turbo",
    messages=[
        {
            "role": "user",
            "content": "What is the capital of France?"
        }
    ],
)
print(completion.choices[0].message)
```

```
1 import os
2 from openai import OpenAI
3
4 client = OpenAI(
5     base_url ="https://router.huggingface.co/novita/v3/openai",
6     api_key  = 'hf_IdMIOwEExeTnSHjKT0VeeZXMENPmlydsizz',
7 )
8
9 stream = client.chat.completions.create(
10    model="deepseek/deepseek-v3-turbo",
11    messages=[[
12        {
13            "role": "user",
14            "content": "你好，能给我讲个笑话吗？"
15        }
16    ],
17    stream=True,
18 )
19
20 for chunk in stream:
21     print(chunk.choices[0].delta.content, end="")
```

### 3、官方服务器的模型

官网（阿里百炼平台）：<https://www.deepseek.com/>

```
1 import os
2 from openai import OpenAI
3
4 client = OpenAI(
5     # 若没有配置环境变量, 请用阿里云百炼API Key将下行替换为: api_key="sk-xxx",
6     api_key=os.getenv("DASHSCOPE_API_KEY"),
7     base_url="https://dashscope.aliyuncs.com/compatible-mode/v1"
8 )
9
10 completion = client.chat.completions.create(
11     model="qwen-plus", # 此处以qwen-plus为例, 您可按需更换模型名称。模型列表: https://help.aliyun.com/zh/model-studio/getting-started/models
12     messages=[
13         {"role": "system", "content": "You are a helpful assistant."},
14         {"role": "user", "content": "你是谁? "}
15     ],
16     stream=True,
17     # Qwen3模型通过enable_thinking参数控制思考过程 (开源版默认True, 商业版默认False)
18     # 使用Qwen3开源版模型时, 请将下行取消注释, 否则会报错
19     # extra_body={"enable_thinking": False},
20 )
21
22 full_content = ""
23 print("流式输出内容为: ")
24 for chunk in completion:
25     # 如果stream_options.include_usage为True, 则最后一个chunk的choices字段为空
26     # 列表, 需要跳过 (可以通过chunk.usage获取 Token 使用量)
27     if chunk.choices:
28         full_content += chunk.choices[0].delta.content
29         print(chunk.choices[0].delta.content)
30 print(f"完整内容为: {full_content}")
```

deepseek 官网: <https://www.deepseek.com/>

```
1 # Please install OpenAI SDK first: `pip3 install openai`  
2  
3 from openai import OpenAI  
4  
5 client = OpenAI(  
6     api_key='<DeepSeek API Key>'  
7     base_url="https://api.deepseek.com"  
8 )  
9  
10 response = client.chat.completions.create(  
11     model="deepseek-chat",  
12     messages=[  
13         {"role": "system", "content": "You are a helpful assistant"},  
14         {"role": "user", "content": "你好,给我讲个笑话吧"},  
15     ],  
16     stream=False  
17 )  
18  
19 print(response.choices[0].message.content)
```

流式输出，聊天交互模型

```
1 import sys
2 from openai import OpenAI
3
4 client = OpenAI(api_key="sk-899fd42834ad447882e596f1787fadd7", base_url="https://api.deepseek.com")
5
6 def stream_response(messages):
7     response = client.chat.completions.create(
8         model="deepseek-reasoner",
9         messages=messages,
10        stream=True
11    )
12
13    full_content = ""
14    for chunk in response:
15        if hasattr(chunk.choices[0].delta, 'content') and chunk.choices[0].delta.content:
16            content = chunk.choices[0].delta.content
17            sys.stdout.write(content) # 逐字输出
18            sys.stdout.flush()
19            full_content += content
20
21    print() # 换行
22    return full_content
23
24 # 初始化对话
25 messages = [{"role": "system", "content": "你是一个幽默的助手，喜欢讲笑话和有趣的对话。"}]
26
27 while True:
28     try:
29         # 用户输入
30         user_input = input("\n用户：")
31         if user_input.lower() in ['退出', 'exit', 'quit']:
32             print("对话结束，再见！")
33             break
34
35         messages.append({"role": "user", "content": user_input})
36
37         # 获取AI回复并流式输出
38         print("AI: ", end="")
39         assistant_content = stream_response(messages)
40         messages.append({"role": "assistant", "content": assistant_content})
41     
```

```
42     except KeyboardInterrupt:  
43         print("\n对话结束, 再见! ")  
44         break  
45     except Exception as e:  
46         print(f"\n发生错误: {e}")  
47         break
```