Deepseek企业级Agent项目开发实战

Part 5. Deepseek V3 和 R1 模型在线API调用

除了本地部署`DeepSeek`模型外，我们还可以通过`DeepSeek`提供的在线`API`接口进行调用。这是一种更加轻量级、灵活的使用方式。本节内容主要介绍如何使用`Deepseek V3`和`R1`模型进行在线`API`调用满血版`DeepSeek v3 & r1`模型。

一种最简单的理解方法是：前几节我们通过`ollama`在本地部署的了`DeepSeek R1`模型，最终的目的是能够提供一个类似于`http:localhost:11434/v1/chat/completions`的接口，然后我们就可以像调用`OpenAI`的接口一样调用`DeepSeek`的接口了。这个过程需要我们有本地的`GPU`资源，然后通过`ollama`来启动和管理模型。`DeepSeek`的在线API接口，则不需要我们自己用本地的`GPU`资源去部署，而是由服务商部署好模型，然后我们通过注册账号，获取`API Key`，然后就也可以像调用`OpenAI`的接口一样调用`DeepSeek`的接口。

1. 注册deepseek账号

如果想访问`DeepSeek`的在线`API`接口，首先我们需要注册一个`deepseek`的账号，然后去获取到一个有效的`API Key`。官方的`DeepSeek`的`API`服务地址是：https://platform.deepseek.com/usage

然后充值，按照如下方式获取`API Key`即可，非常简单。`DeepSeek`的`API`接口是按照`token`来收费的，不过现阶段因为服务器资源紧张，`DeepSeek`官方暂时停止了充值服务，大家可以等待服务恢复。

2. DeepSeek v3 调用指南

`DeepSeek API`使用与`OpenAI`兼容的`API`格式，如下是`OpenAI`的`API`调用格式：

```
    from openai import OpenAI
    client = OpenAI()

    completion = client.chat.completions.create(
    model="gpt-4o",
    messages=[
        {"role": "developer", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Hello!"}
    ]
    )

    print(completion.choices[0].message)
```

因此，需要在`OpenAI`的`API`调用格式的基础上，将`OpenAI`的`base_url`替换为`DeepSeek`的`endpoint`,以及将`model`替换为`DeepSeek`的`model`。其中`DeepSeek v3`的`model`名称是:`deepseek-chat`，即:

- **非流式输出**

```
from openai import OpenAI

client = OpenAI(api_key="", base_url="")

response = client.chat.completions.create(
    model="deepseek-ai/DeepSeek-V3",    # 注意：这里是因为我购买的 deepseek 服务上要求提供的模型名称是 DeepSeek-V3，大家根据自己的情况进行替换
    messages=[
        {"role": "system", "content": "你是一位乐于助人的AI助手"},
        {"role": "user", "content": "请问什么是大模型？"},
    ],
    stream=False
)

print(response.choices[0].message.content)
```

```
大模型（Large Model）通常指的是参数规模庞大、计算能力强大的机器学习模型，尤其是深度学习模型。这类模型在处理复杂任务时表现出色，尤其是在自然语言处理（NLP）、计算机视觉、语音识别等领域。

### 大模型的特点：
1. **参数规模大**：大模型的参数量通常在数亿到数千亿之间，甚至更多。参数越多，模型的表达能力越强，能够捕捉更复杂的模式和特征。

2. **计算资源需求高**：训练和运行大模型需要大量的计算资源，包括高性能GPU、TPU等硬件设备，以及大量的存储和内存。

3. **数据需求大**：大模型通常需要海量的训练数据来优化模型参数，避免过拟合并提升泛化能力。

4. **任务泛化能力强**：大模型经过预训练后，可以通过微调（Fine-tuning）或提示（Prompting）等方式适应多种下游任务，表现出较强的泛化能力。

5. **多模态能力**：一些大模型不仅限于单一模态（如文本），还可以处理多模态数据（如文本、图像、音频等），实现跨模态的理解和生成。

### 典型的大模型：
- **自然语言处理（NLP）**：
```

  - **GPT系列**（如GPT-3、GPT-4）：由OpenAI开发的生成式预训练变换模型，擅长文本生成、问答、翻译等任务。
  - **BERT**：由Google开发的双向编码器表示模型，擅长文本分类、问答等任务。
  - **T5**：由Google开发的文本到文本转换模型，适用于多种NLP任务。

- **计算机视觉**：
  - **ResNet**：深度残差网络，用于图像分类、目标检测等任务。
  - **ViT**（Vision Transformer）：基于Transformer架构的图像处理模型。

- **多模态模型**：
  - **CLIP**：由OpenAI开发的多模态模型，能够理解图像和文本之间的关系。
  - **DALL·E**：由OpenAI开发的图像生成模型，能够根据文本描述生成图像。

大模型的应用场景：
- **智能助手**：如ChatGPT、Alexa等，能够进行自然语言对话、问答、任务执行等。
- **内容生成**：如自动生成文章、新闻、代码、图像等。
- **机器翻译**：如Google Translate等，能够实现高质量的跨语言翻译。
- **医疗诊断**：通过分析医学图像或文本，辅助医生进行诊断。
- **自动驾驶**：通过处理多模态数据（如摄像头、雷达等），实现车辆的自主导航和决策。

大模型的挑战：
1. **计算成本高**：训练和部署大模型需要大量的计算资源和能源消耗。
2. **数据隐私问题**：大模型通常需要大量数据，可能涉及用户隐私和数据安全问题。
3. **模型解释性差**：大模型的决策过程通常较为复杂，难以解释其内部机制。
4. **伦理问题**：如模型可能生成有害或偏见内容，需要谨慎处理。

总的来说，大模型是当前人工智能领域的重要进展，尽管面临诸多挑战，但其强大的能力正在推动多个行业的发展与变革。

- **流式输出**

```
from openai import OpenAI
import json

client = OpenAI(api_key="", base_url="")


# 调用聊天接口，启用流式输出
response = client.chat.completions.create(
    model="deepseek-ai/DeepSeek-V3",  # 根据实际情况替换模型名称
    messages=[
        {"role": "system", "content": "你是一位乐于助人的AI助手"},
        {"role": "user", "content": "请问什么是大模型？"},
    ],
    temperature=1.0,
    stream=True  # 启用流式输出

)

try:
    # 处理流式响应
    for chunk in response:
        if chunk.choices and chunk.choices[0].delta.content:
            print(chunk.choices[0].delta.content, end='', flush=True)

except Exception as e:
    print("发生错误:", e)
```

大模型（Large Model）通常指的是具有**大量参数**的机器学习模型，尤其是在自然语言处理（NLP）和计算机视觉（CV）领域中表现出色的模型。这些模型的参数量可以达到**数十亿甚至数千亿**，能够处理极其复杂的任务。

大模型的几个关键特点：
1. **参数量巨大**：大模型的参数量通常在数亿到数千亿之间，这使得它们能够捕捉和学习非常复杂的模式和关系。

2. **训练数据量大**：大模型通常需要**海量数据**进行训练，这些数据可以来自互联网、书籍、文章等多种来源。

3. **计算资源需求高**：训练和部署大模型需要大量的**计算资源**，包括高性能的GPU、TPU以及大规模的计算集群。

4. **多功能性**：大模型通常具有**通用性**，可以应用于多种任务，如文本生成、翻译、问答、图像识别等。

5. **涌现能力（Emergent Abilities）**：一些大模型显示出在训练数据中未明确训练的**新能力**，例如解决数学问题或生成连贯的长文。

典型的大模型示例：
1. **GPT（Generative Pre-trained Transformer）系列**：由OpenAI开发，参数量从GPT-3的1750亿到GPT-4的更大规模。
2. **BERT（Bidirectional Encoder Representations from Transformers）**：由Google开发，用于理解文本的双向表示。
3. **PaLM（Pathways Language Model）**：由Google开发，参数量达到5400亿。
4. **LLaMA（Large Language Model Meta AI）**：由Meta开发，参数量从70亿到650亿不等。

大模型的应用场景：
- **自然语言处理**：文本生成、机器翻译、情感分析、问答系统等。
- **计算机视觉**：图像生成、物体检测、图像分类等。
- **多模态任务**：结合文本和图像的处理，例如生成图像描述或从文本生成图像。
- **科学研究**：辅助药物研发、蛋白质结构预测等。

大模型是人工智能领域的一个重要发展方向，但也面临**数据隐私、能源消耗、训练成本**等挑战。

3. DeepSeek R1 调用指南

`DeepSeek R1` 的 `API` 调用与 `DeepSeek V3` 的 `API` 调用类似，只需要将 `model` 替换为 `DeepSeek R1` 的 `model` 即可。其中 `DeepSeek R1` 的 `model` 名称是：`deepseek-reasoner`，即：

- **非流式输出**

```
from openai import OpenAI

client = OpenAI(api_key="", base_url="")

response = client.chat.completions.create(
    model="deepseek-ai/DeepSeek-R1",    # 注意：这里是因为我购买的 deepseek 服务上要求提供的模型名称是 DeepSeek-R1，大家根据自己的情况进行替换
```

```
    messages=[
        {"role": "system", "content": "你是一位乐于助人的AI助手"},
        {"role": "user", "content": "请问什么是大模型？"},
    ],
    stream=False
)

print(response.choices[0].message.content)
```

```
<think>
嗯，用户问什么是大模型。首先，我需要明确大模型的定义，可能涉及参数规模、训练数据、应用场景等。大模型通常指参数数量巨大的深度学习模型，比如GPT-3、BERT这些。参数多意味着模型更复杂，能处理更复杂的任务。然后要说明大模型的特点，比如需要大量计算资源和数据，以及它们在NLP、图像识别等领域的应用。还要提到大模型的优势，比如强大的泛化能力，但也可能涉及计算成本高、部署困难的问题。需要确保解释通俗易懂，适合不同知识背景的读者。同时，可能要区分大模型与传统模型的区别，强调规模和性能的提升。最后，检查是否有遗漏点，比如预训练和微调的概念，或者举例说明常见的大模型，帮助用户更好理解。
</think>

大模型（Large Model）通常指参数规模巨大、计算能力要求高的深度学习模型。这类模型通常具备以下特点：

### 1. **参数规模庞大**
   - 参数数量可达**数十亿甚至上万亿**，例如：
     - **GPT-3**（1750亿参数）
     - **PaLM**（谷歌，5400亿参数）
     - **GPT-4**（具体参数未公开，推测在万亿级）。

### 2. **训练数据海量**
   - 使用互联网级别的文本、图像等多模态数据进行训练，例如：
     - GPT-3 使用了45TB的文本数据；
     - 某些多模态模型（如DALL·E）融合了文本-图像对数据。

### 3. **核心能力**
   - **涌现能力**：当模型规模超过临界值（如100亿参数）时，会突然获得小模型不具备的能力（如复杂推理、上下文学习）。
   - **多任务统一处理**：单一模型可同时完成翻译、问答、代码生成等任务。

### 4. **技术实现**
   - **Transformer架构**：基于自注意力机制，支持并行计算；
   - **混合专家系统**（MoE）：如GPT-4可能采用的稀疏激活架构，降低计算成本；
   - **分布式训练**：需数千张GPU（如使用NVIDIA A100集群）进行并行训练。

### 5. **应用场景**
   - **AIGC**：生成高质量文本（ChatGPT）、图像（Stable Diffusion）、视频（Sora）；
   - **科学计算**：AlphaFold2预测蛋白质结构，DeepMind天气预测模型；
   - **企业服务**：客服自动化、智能文档处理、代码补全（GitHub Copilot）。

### 6. **挑战**
   - **训练成本**：GPT-3训练成本约460万美元；
   - **能耗问题**：单次训练碳足迹相当于5辆汽车生命周期排放；
   - **伦理风险**：生成虚假信息、深度伪造内容等。

### 演进趋势
- **稀疏化**：通过MoE等架构提升计算效率
- **多模态融合**：如GPT-4V整合视觉-语言能力
- **小型化**：模型压缩技术（如LoRA微调）推动端侧部署
```

这类模型正在重塑AI技术栈，推动AGI（通用人工智能）的探索，同时也带来新的技术和伦理挑战。

- **流式输出**

```
from openai import OpenAI

client = OpenAI(api_key="", base_url="")

response = client.chat.completions.create(
    model="deepseek-ai/DeepSeek-R1",    # 注意：这里是因为我购买的 deepseek 服务上要求提供的模型名称是 DeepSeek-R1，大家根据自己的情况进行替换
    messages=[
        {"role": "system", "content": "你是一位乐于助人的AI助手"},
        {"role": "user", "content": "请问什么是大模型？"},
    ],
    stream=True,
    temperature=0.6,
)

try:
    # 处理流式响应
    for chunk in response:
        if chunk.choices and chunk.choices[0].delta.content:
            print(chunk.choices[0].delta.content, end='', flush=True)

except Exception as e:
    print("发生错误:", e)
```

```
<think>
嗯，用户问“什么是大模型？”，我需要先确定用户对这个概念的理解程度。大模型这个词在AI领域比较常见，但可能对于普通用户来说有点抽象。首先，我应该从基础开始解释，比如大模型指的是参数量非常大的机器学习模型，比如GPT-3、BERT这些。然后要说明参数量大带来的优势，比如更强的学习能力和处理复杂任务的能力，比如自然语言理解、生成、图像识别等。同时，也要提到大模型的挑战，比如训练需要大量计算资源和数据，能耗高，可能存在偏见等问题。用户可能还想知道大模型的应用场景，比如聊天机器人、内容生成、自动化客服等。另外，可能需要区分大模型和小模型的不同，比如参数量、适用场景的区别。还要注意用词通俗易懂，避免专业术语过多，必要时举例说明。最后，确认回答是否全面，是否需要补充其他相关信息，比如当前的发展趋势或者常见的大模型例子。
</think>

大模型（Large Model）通常指参数量巨大（数十亿甚至数千亿级别）的机器学习模型，尤其在自然语言处理（NLP）、计算机视觉（CV）等领域表现突出。以下是关键点：

1. **核心特征**
   - **参数量大**：例如GPT-3有1750亿参数，能捕捉复杂数据规律。
   - **训练数据多**：需海量文本、图像等数据训练。
   - **计算资源密集**：依赖高性能GPU/TPU集群，训练耗时数周。

2. **典型应用**
   - **自然语言处理**：对话系统（如ChatGPT）、文本生成、翻译
   - **多模态任务**：图文生成（如DALL-E）、视频理解
   - **决策支持**：金融预测、药物研发

3. **代表模型**
   - GPT系列（OpenAI）
   - BERT（Google）
```

```
   - PaLM（Google）
   - 文心一言（百度）
   - 通义千问（阿里）

4. **技术挑战**
   - 训练成本高（单次训练耗资数百万美元）
   - 部署困难（需模型压缩、蒸馏等技术）
   - 存在偏见/幻觉问题
   - 高能耗争议

5. **发展趋势**
   - 多模态融合
   - 小型化/专业化（大模型→垂直领域小模型）
   - 绿色AI（提升能效）
   - 开源与闭源并行发展

大模型推动AI能力边界，但也引发伦理、安全等方面的讨论，是当前人工智能领域的核心技术方向之一。
```

这里需要注意的是：`DeepSeek R1` 现在还不支持 `Function Calling` 和 `Json Output` 格式化输出，所以目前还无法直接接入 `Agent` 构建工作流。