

# LMDeploy 量化部署 LLM 实践

## 1. LMDeploy环境部署

### 1.1 本地创建conda环境

```
1 conda create -n lmdeploy -y python=3.10
```

### 1.2 安装LMDeploy

```
1 conda activate lmdeploy
```

```
1 pip install lmdeploy[all]==0.3.0
```

## 2. LMDeploy模型对话

### 2.1 下载模型

由OpenXLab上下载模型

安装git-lfs组件：

```
1 curl -s https://packagecloud.io/install/repositories/github/git-lfs/script.deb.sh | sudo bash
2 sudo apt update
3 sudo apt install git-lfs
4 sudo git lfs install --system
```

安装好git-lfs组件后，由OpenXLab平台下载InternLM2-Chat-1.8B模型：

```
1 git clone https://code.openxlab.org.cn/OpenMLab/internlm2-chat-1.8b.git
```

```
(lmdeploy) PS D:\InternML> git clone https://code.openxlab.org.cn/OpenMLab/internlm2-chat-1.8b.git
Cloning into 'internlm2-chat-1.8b'...
remote: Enumerating objects: 21, done.
remote: Counting objects: 100% (21/21), done.
remote: Compressing objects: 100% (19/19), done.
remote: Total 21 (delta 1), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (21/21), 27.57 KiB | 6.89 MiB/s, done.
Resolving deltas: 100% (1/1), done.
Updating files: 100% (14/14), done.
```

更改命名

```
1 mv ./internlm2-chat-1.8b ./internlm2-chat-1_8b
```

## 2.2 使用Transformer库运行模型

新建 `pipeline_transformer.py`

```
1 touch ./pipeline_transformer.py
```

```
1 import torch
2 from transformers import AutoTokenizer, AutoModelForCausalLM
3
4 tokenizer = AutoTokenizer.from_pretrained("./internlm2-chat-1_8b",
5     trust_remote_code=True)
6
7 # Set `torch_dtype=torch.float16` to load model in float16, otherwise it will
8 # be loaded as float32 and cause OOM Error.
9
10 model = AutoModelForCausalLM.from_pretrained("./internlm2-chat-1_8b",
11     torch_dtype=torch.float16, trust_remote_code=True).cuda()
12 model = model.eval()
13
14 inp = "hello"
15 print("[INPUT]", inp)
16 response, history = model.chat(tokenizer, inp, history=[])
17 print("[OUTPUT]", response)
18
19 inp = "please provide three suggestions about time management"
20 print("[INPUT]", inp)
21 response, history = model.chat(tokenizer, inp, history=history)
22 print("[OUTPUT]", response)
```

```

import torch
from transformers import AutoTokenizer, AutoModelForCausalLM

tokenizer = AutoTokenizer.from_pretrained("./internlm2-chat-1_8b", trust_remote_code=True)

# Set `torch_dtype=torch.float16` to load model in float16, otherwise it will be loaded as float32 and cause OOM Error.
model = AutoModelForCausalLM.from_pretrained("./internlm2-chat-1_8b", torch_dtype=torch.float16, trust_remote_code=True).cuda()
model = model.eval()

inp = "hello"
print("[INPUT]", inp)
response, history = model.chat(tokenizer, inp, history=[])
print("[OUTPUT]", response)

inp = "please provide three suggestions about time management"
print("[INPUT]", inp)
response, history = model.chat(tokenizer, inp, history=history)
print("[OUTPUT]", response)

```

```

pipeline_transformer.py x
1 import torch
2 from transformers import AutoTokenizer, AutoModelForCausalLM
3
4 tokenizer = AutoTokenizer.from_pretrained(pretrained_model_name_or_path="Shanghai_AI_Laboratory/internlm2-chat-1_8b", trust_remote_code=True)
5
6 # Set `torch_dtype=torch.float16` to load model in float16, otherwise it will be loaded as float32 and cause OOM Error.
7 model = AutoModelForCausalLM.from_pretrained(pretrained_model_name_or_path="Shanghai_AI_Laboratory/internlm2-chat-1_8b", torch_dtype=torch.float16, trust_remote_code=True).cuda()
8 model = model.eval()
9
10 inp = "hello"
11 print("[INPUT]", inp)
12 response, history = model.chat(tokenizer, inp, history=[])
13 print("[OUTPUT]", response)
14
15 inp = "please provide three suggestions about time management"
16 print("[INPUT]", inp)
17 response, history = model.chat(tokenizer, inp, history=history)
18 print("[OUTPUT]", response)
19

```

运行python代码：

```
1 python pipeline_transformer.py
```

```
Loading checkpoint shards: 100%|██████████| 2/2 [00:01<00:00, 1.52it/s]
```

```
[INPUT] hello
```

```
[OUTPUT] 你好，我是书生·浦语。很高兴为您提供帮助。请问您需要什么帮助呢？
```

```
[INPUT] please provide three suggestions about time management
```

```
[OUTPUT] 当涉及到时间管理时，以下是一些建议：
```

1. 制定计划：在每天或每周开始之前，制定一个计划，列出要完成的任务和目标。这有助于您保持组织，并确保您有足够的时间来完成任务。
2. 优先级排序：将任务按照优先级排序，将最重要的任务放在最前面。这有助于您确保您的时间和精力用在最重要的事情上。
3. 避免拖延：尽可能避免拖延。如果您发现自己总是推迟任务，尝试找到一些方法来克服这种习惯，例如将任务分解为更小的部分，或者将任务分配给其他人。

```
pipeline_transformer.py X
pipeline_transformer.py > ...
2 from transformers import AutoTokenizer, AutoModelForCausalLM
3
4 tokenizer = AutoTokenizer.from_pretrained("/root/internlm2-chat-1_8b", trust_remote_code=True)
5
6 # Set `torch_dtype=torch.float16` to load model in float16, otherwise it will be loaded as float32 and cause OOM Error.
7 model = AutoModelForCausalLM.from_pretrained("/root/internlm2-chat-1_8b", torch_dtype=torch.float16, trust_remote_code=True).cuda()
8 model = model.eval()
9
10 inp = "hello"
11 print("[INPUT]", inp)
12 response, history = model.chat(tokenizer, inp, history=[])
13 print("[OUTPUT]", response)
14
15 inp = "please provide three suggestions about time management"
16 print("[INPUT]", inp)
17 response, history = model.chat(tokenizer, inp, history=history)
18 print("[OUTPUT]", response)
19
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1

Loading checkpoint shards: 100% | 2/2 [00:38<00:00, 19.05s/it]

[INPUT] hello  
[OUTPUT] 你好，我可以帮助你。请告诉我你需要什么帮助。  
[INPUT] please provide three suggestions about time management  
[OUTPUT] 当涉及到时间管理时，以下是一些有用的建议：

1. 制定计划和目标：在开始一天或一周之前，制定一个清晰的计划和目标清单，以便您知道自己要完成什么任务并设置优先级。
2. 设置时间限制：为每个任务设置一个时间限制，以确保您在规定的时间内完成任务。这可以帮助您更好地管理时间，并避免拖延。
3. 避免多任务处理：尽可能避免同时处理多项任务，因为这会分散您的注意力，导致效率低下。相反，集中精力完成一项任务，然后再转移到下一个任务。
4. 学会说“不”：如果您已经有很多任务要完成，不要害怕拒绝新的请求或任务。学会说“不”，以便您可以更好地管理时间，并确保您有足够的时间来完成您已经承诺的任务。
5. 利用技术：利用各种时间管理工具和技术，例如日历、提醒、待办事项列表和应用程序等，以帮助您更好地管理时间。

希望这些建议能够帮助您更好地管理时间。

## 2.3 使用LMDeploy与模型对话

执行如下命令运行下载的1.8B模型：

```
1 lmdeploy chat /root/internlm2-chat-1_8b
```

输入“请给我讲一个小故事吧”，然后按两下回车键：

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 1

```
- InternLM (书生 侑语) is a conversational language model that is developed by Shanghai AI Laboratory (上海人工智能实验室). It is designed to be helpful, honest, and harm less.
- InternLM (书生 侑语) can understand and communicate fluently in the language chosen by the user such as English and 中文.
<|im_end|>
<|im_start|>user
请给我讲一个小故事吧<|im_end|>
<|im_start|>assistant
2024-05-30 10:21:48,774 - lmdeploy - WARNING - kwargs ignore_eos is deprecated for inference, use GenerationConfig instead.
2024-05-30 10:21:48,774 - lmdeploy - WARNING - kwargs random_seed is deprecated for inference, use GenerationConfig instead.
当然，我可以给您讲一个有趣的故事。比如，我曾经听过一个故事，叫做《小蝌蚪找妈妈》。
```

从前，有一只小蝌蚪，它游呀游，一直找不到它的妈妈。有一天，它决定去寻找妈妈，于是它开始了一段寻找之旅。

小蝌蚪在水里游呀游，游了很长时间，看到了很多美丽的青蛙。它找到了一只年轻的青蛙，问它：“妈妈在哪里？”

年轻的青蛙告诉小蝌蚪：“妈妈在池塘的深处，你只有跳进池塘才能找到她。”

小蝌蚪听了，立刻张开四肢，开始跳进池塘。它在水里跳来跳去，看到了很多美丽的鱼儿。它决定去找其他的鱼儿：“妈妈在哪里？”

鱼儿告诉小蝌蚪：“妈妈在鱼的肚子里。”

小蝌蚪听了，立刻跳进了鱼的肚子里，看到了一头大象。大象告诉小蝌蚪：“妈妈在象的肚子里。”

小蝌蚪听了，立刻跳进了象的肚子里，看到了一头狮子。狮子告诉小蝌蚪：“妈妈在狮子的肚子里。”

小蝌蚪听了，最终找到了妈妈。但是，它发现妈妈生病了，需要到池塘边才能治好。小蝌蚪立刻找到青蛙，告诉它自己的经历。青蛙告诉小蝌蚪：“你的妈妈需要休息，你需要在池塘边等待。”

小蝌蚪听了，立刻跳到了池塘边。它看到了很多水草和漂浮的树叶，于是决定等待。

速度明显比原生Transformer快

输入“exit”并按两下回车，可以退出对话：

这个故事告诉我们，只要我们勇敢地面对困难，充满希望，就能找到自己的方向，找到自己的答案。

```
double enter to end input >>> exit
```

```
(lmdeploy) root@intern-studio-161126: ~#
```

## 3. LMDeploy模型量化(lite)

### 3.1 设置最大KV Cache缓存大小

首先保持不加该参数（默认0.8），运行1.8B模型。

```
1 lmdeploy chat /root/internlm2-chat-1_8b
```

```
CPU 5.47% GPU-1: 10% Nvidia A100 3%
内存 3.54 / 24 GB 14.75% 显存 7816 / 8182 MiB 95.53%
```

下面，改变 `--cache-max-entry-count` 参数，设为0.5。

```
1 lmdeploy chat /root/internlm2-chat-1_8b --cache-max-entry-count 0.5
```

```
CPU 46.07% GPU-1: 10% Nvidia A100 0%
内存 2.93 / 24 GB 12.19% 显存 6600 / 8182 MiB 80.66%
```

把 `--cache-max-entry-count` 参数设置为0.01，约等于禁止KV Cache占用显存。

```
1 lmdeploy chat /root/internlm2-chat-1_8b --cache-max-entry-count 0.01
```

```
CPU 5.01% GPU-1: 10% Nvidia A100 0%
内存 3.06 / 24 GB 12.76% 显存 4552 / 8182 MiB 55.63%
```

### 3.2 使用W4A16量化

安装一个依赖库

```
1 pip install einops==0.7.0
```

```
(lmdeploy) root@intern-studio-161126: ~# pip install einops==0.7.0
Looking in indexes: https://pypi.tuna.tsinghua.edu.cn/simple
Requirement already satisfied: einops==0.7.0 in ./conda/envs/lmdeploy/lib/python3.10/site-packages (0.7.0)
```

执行命令完成模型量化工作

```
1 lmdeploy lite auto_awq \  
2   /root/internlm2-chat-1_8b \  
3   --calib-dataset 'ptb' \  
4   --calib-samples 128 \  
5   --calib-seqlen 1024 \  
6   --w-bits 4 \  
7   --w-group-size 128 \  
8   --work-dir /root/internlm2-chat-1_8b-4bit
```

量化工作结束后，新的HF模型被保存到 `internlm2-chat-1_8b-4bit` 目录。

```
• (lmdeploy) root@intern-studio-161126:~# ls  
internlm2-chat-1_8b internlm2-chat-1_8b-4bit pipeline_transformer.py share
```

使用Chat功能运行W4A16量化后的模型。

```
1 lmdeploy chat /root/internlm2-chat-1_8b-4bit --model-format awq
```

```
CPU 21.1% GPU-1: 10% Nvidia A100 0%  
内存 3.59 / 24 GB 14.96% 显存 7396 / 8182 MiB 90.39%
```

为了更加明显体会到W4A16的作用，将KV Cache比例再次调为0.01，查看显存占用情况。

```
1 lmdeploy chat /root/internlm2-chat-1_8b-4bit --model-format awq --cache-max-  
  entry-count 0.01
```

```
CPU 73.23% GPU-1: 10% Nvidia A100 3%  
内存 3.60 / 24 GB 15.02% 显存 2472 / 8182 MiB 30.21%
```

## 4. LMDeploy服务(serve)

### 4.1 启动API服务器

通过以下命令启动API服务器，推理 `internlm2-chat-1_8b` 模型：

```
1 lmdeploy serve api_server \  
2   /root/internlm2-chat-1_8b \  
3   --model-format hf \  
4   --quant-policy 0 \  

```

```
5 --server-name 0.0.0.0 \  
6 --server-port 23333 \  
7 --tp 1
```

其中，model-format、quant-policy这些参数是与第三章中量化推理模型一致的；server-name和server-port表示API服务器的服务IP与服务端口；tp参数表示并行数量（GPU数量）。

## 4.2 命令行客户端连接API服务器

新建一个命令行客户端去连接API服务器

激活conda环境

运行命令行客户端：

```
1 lmdeploy serve api_client http://localhost:23333
```

```
• (base) root@intern-studio-161126:~# conda activate lmdeploy  
◦ (lmdeploy) root@intern-studio-161126:~# lmdeploy serve api_client http://localhost:23333  
  
double enter to end input >>> 请讲一个小故事  
  
当然，我可以讲一个关于友谊的小故事。  
  
从前，有一只小兔子，它非常害怕黑暗。有一天，它遇到了一只小狐狸，小狐狸告诉小兔子，它知道如何帮助小兔子克服黑暗的恐惧。小兔子非常感激小狐狸的帮助，从此以后，每当小兔子感到害怕时，它就会想起小狐狸的话，它知道只要勇敢地面对黑暗，它就能够克服恐惧。  
  
这个故事告诉我们，真正的友谊可以带给我们力量和勇气，只要我们相信自己，勇敢地面对困难，我们就能克服一切困难。  
double enter to end input >>>
```

## 4.3 网页客户端连接API服务器

使用Gradio作为前端，启动网页客户端。

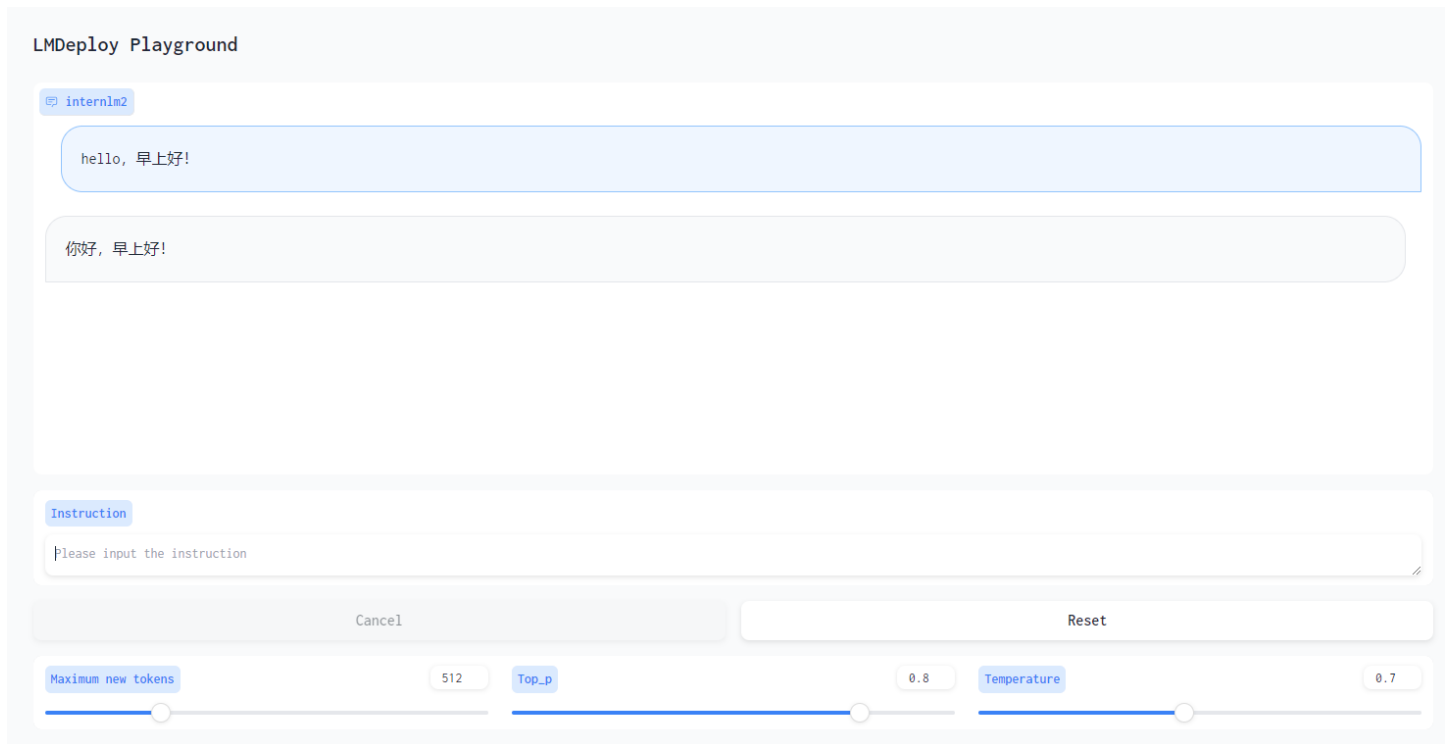
```
1 lmdeploy serve gradio http://localhost:23333 \  
2 --server-name 0.0.0.0 \  
3 --server-port 6006
```

```
◦ (lmdeploy) root@intern-studio-161126:~# lmdeploy serve gradio http://localhost:23333 \  
> --server-name 0.0.0.0 \  
> --server-port 6006  
  
server is gonna mount on: http://0.0.0.0:6006  
Running on local URL: http://0.0.0.0:6006  
IMPORTANT: You are using gradio version 3.50.2, however version 4.29.0 is available, please upgrade.  
-----  
  
Could not create share link. Missing file: /root/.conda/envs/lmdeploy/lib/python3.10/site-packages/gradio/frpc_linux_amd64_v0.2.  
  
Please check your internet connection. This can happen if your antivirus software blocks the download of this file. You can install manually by following these steps:  
  
1. Download this file: https://cdn-media.huggingface.co/frpc-gradio-0.2/frpc_linux_amd64  
2. Rename the downloaded file to: frpc_linux_amd64_v0.2  
3. Move the file to this location: /root/.conda/envs/lmdeploy/lib/python3.10/site-packages/gradio
```

运行命令后，网页客户端启动。在电脑本地新建一个cmd终端，新开一个转发端口：

```
1 ssh -CNg -L 6006:127.0.0.1:6006 root@ssh.intern-ai.org.cn -p 33820
```

打开浏览器，访问地址 `http://127.0.0.1:6006`



LMDeploy Playground

internlm2

hello, 早上好!

你好, 早上好!

Instruction

Please input the instruction

Cancel Reset

Maximum new tokens 512 Top\_p 0.8 Temperature 0.7

## 5. Python代码集成

### 5.1 Python代码集成运行1.8B模型

新建Python源代码文件 `pipeline.py`。

```
1 touch /root/pipeline.py
```

打开 `pipeline.py`，填入以下内容。

```
1 from lmdeploy import pipeline
2
3 pipe = pipeline('/root/internlm2-chat-1_8b')
4 response = pipe(['Hi, pls intro yourself', '上海是'])
5 print(response)
```



```
pipeline_transformer.py  pipeline.py X
pipeline.py > ...
1  from lmdeploy import pipeline
2
3  pipe = pipeline('/root/internlm2-chat-1_8b')
4  response = pipe(['Hi, pls intro yourself', '上海是'])
5  print(response)
```

代码解读：

- 第1行，引入lmdeploy的pipeline模块
- 第3行，从目录“./internlm2-chat-1\_8b”加载HF模型
- 第4行，运行pipeline，这里采用了批处理的方式，用一个列表包含两个输入，lmdeploy同时推理两个输入，产生两个输出结果，结果返回给response
- 第5行，输出response

保存后运行代码文件：

```
1 python /root/pipeline.py
```

```
(base) root@intern-studio-161126:~# conda activate lmdeploy
(lmdeploy) root@intern-studio-161126:~# python /root/pipeline.py
[WARNING] gemm_config.in is not found; using default GEMM algo
[Response(text='Hello! I'm InternLM, a conversational language model developed by Shanghai AI Laboratory. I'm here to help and provide information in English and Chinese. I'm designed to be helpful, honest, and harmless. Let me know if there's anything specific you'd like to know or discuss!', generate_token_len=60, input_token_len=108, session_id=0, finish_reason='stop'), Response(text='上海是中国东部的一个大都市，是中国的经济、文化和交通中心之一。它位于长江和黄浦江之间，有着悠久的历史和丰富的文化遗产。上海也是一个现代化的城市，拥有许多现代化的建筑和科技设施。上海还是中国最受欢迎的旅游目的地之一，因其美丽的自然风光、丰富的历史遗迹和文化活动而备受游客喜爱。', generate_token_len=69, input_token_len=104, session_id=1, finish_reason='stop')]
```

## 5.2 向TurboMind后端传递参数

以设置KV Cache占用比例为例，新建python文件 `pipeline_kv.py`。

```
1 touch /root/pipeline_kv.py
```

打开 `pipeline_kv.py`，填入如下内容：

```
1 from lmdeploy import pipeline, TurbomindEngineConfig
2
3 # 调低 k/v cache内存占比调整为总显存的 20%
4 backend_config = TurbomindEngineConfig(cache_max_entry_count=0.2)
5
6 pipe = pipeline('/root/internlm2-chat-1_8b',
7               backend_config=backend_config)
8 response = pipe(['Hi, pls intro yourself', '上海是'])
```

```
9 print(response)
```

```
pipeline_transformer.py  pipeline.py  pipeline_kv.py X
pipeline_kv.py > ...
1  from lmdeploy import pipeline, TurbomindEngineConfig
2
3  # 调低 k/v cache内存占比调整为总显存的 20%
4  backend_config = TurbomindEngineConfig(cache_max_entry_count=0.2)
5
6  pipe = pipeline('/root/internlm2-chat-1_8b',
7                backend_config=backend_config)
8  response = pipe(['Hi, pls intro yourself', '上海是'])
9  print(response)
```

保存后运行python代码：

```
1 python /root/pipeline_kv.py
```

```
(lmdeploy) root@intern-studio-161126:~# touch /root/pipeline_kv.py
(lmdeploy) root@intern-studio-161126:~# python /root/pipeline_kv.py
[WARNING] gemm_config.in is not found; using default GEMM algo
[Response(text='Hello! I'm InternLM, a conversational language model developed by Shanghai AI Laboratory. I'm here to help you with any questions or tasks you have in English or Chinese. I'm designed to be helpful, honest, and harmless. Let's get started!', generate_token_len=54, input_token_len=108, session_id=0, finish_reason='stop'), Response(text='上海是中国东部的一个特大城市，也是中国最大的城市之一。它位于长江三角洲的核心地带，是中国的经济、文化和交通中心。上海有许多著名的景点，如东方明珠塔、外滩、上海博物馆等，还有着丰富的历史文化遗产和美食文化。', generate_token_len=55, input_token_len=104, session_id=1, finish_reason='stop')]
```

## 6.拓展部分

### 6.1 使用LMDeploy运行视觉多模态大模型llava

安装llava依赖库。

```
1 pip install git+https://github.com/haotian-liu/LLaVA.git@4e2277a060da264c4f21b364c867cc622c945874
```

新建一个python文件，比如 `pipeline_llava.py`。

```
1 touch /root/pipeline_llava.py
```

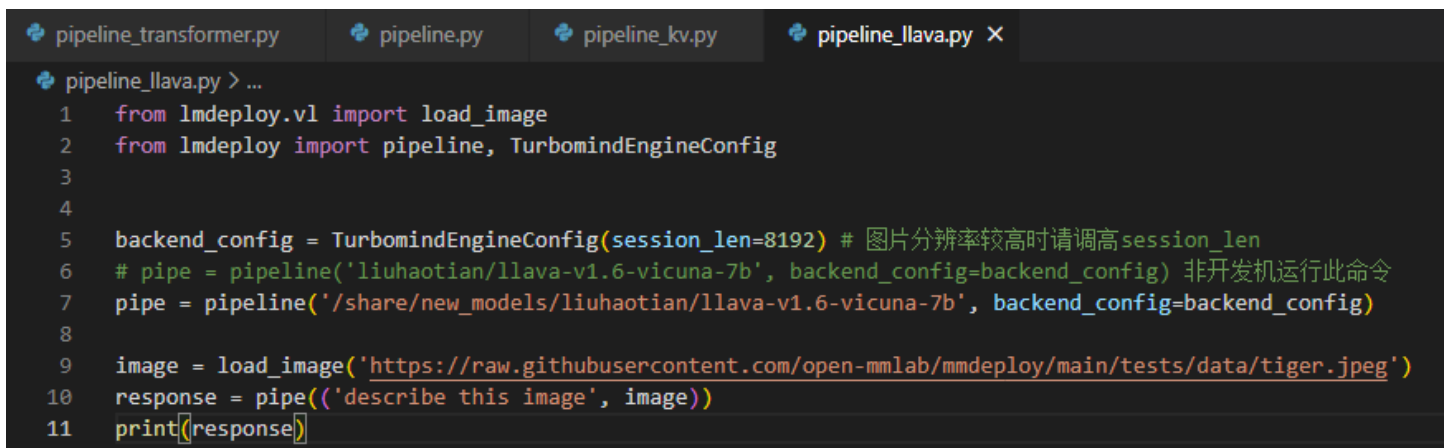
打开 `pipeline_llava.py`，填入内容如下：

```
1 from lmdeploy.vl import load_image
2 from lmdeploy import pipeline, TurbomindEngineConfig
```

```

3
4
5 backend_config = TurbomindEngineConfig(session_len=8192) # 图片分辨率较高时请调高
  session_len
6 # pipe = pipeline('liuhaotian/llava-v1.6-vicuna-7b',
  backend_config=backend_config) 非开发机运行此命令
7 pipe = pipeline('/share/new_models/liuhaotian/llava-v1.6-vicuna-7b',
  backend_config=backend_config)
8
9 image = load_image('https://raw.githubusercontent.com/open-
  mmlab/mmdploy/main/tests/data/tiger.jpeg')
10 response = pipe(('describe this image', image))
11 print(response)

```



```

pipeline_transformer.py  pipeline.py  pipeline_kv.py  pipeline_llava.py X
pipeline_llava.py > ...
1  from lmdeploy.vl import load_image
2  from lmdeploy import pipeline, TurbomindEngineConfig
3
4
5  backend_config = TurbomindEngineConfig(session_len=8192) # 图片分辨率较高时请调高session_len
6  # pipe = pipeline('liuhaotian/llava-v1.6-vicuna-7b', backend_config=backend_config) 非开发机运行此命令
7  pipe = pipeline('/share/new_models/liuhaotian/llava-v1.6-vicuna-7b', backend_config=backend_config)
8
9  image = load_image('https://raw.githubusercontent.com/open-mmlab/mmdploy/main/tests/data/tiger.jpeg')
10 response = pipe(('describe this image', image))
11 print(response)

```

保存后运行pipeline。

```
1 python /root/pipeline_llava.py
```

我们也可以通过Gradio来运行llava模型。新建python文件 `gradio_llava.py`。

```
1 touch /root/gradio_llava.py
```

打开文件，填入以下内容：

```

1 import gradio as gr
2 from lmdeploy import pipeline, TurbomindEngineConfig
3
4
5 backend_config = TurbomindEngineConfig(session_len=8192) # 图片分辨率较高时请调高
  session_len

```

```

6 # pipe = pipeline('liuhaotian/llava-v1.6-vicuna-7b',
  backend_config=backend_config) 非开发机运行此命令
7 pipe = pipeline('/share/new_models/liuhaotian/llava-v1.6-vicuna-7b',
  backend_config=backend_config)
8
9 def model(image, text):
10     if image is None:
11         return [(text, "请上传一张图片。")]
12     else:
13         response = pipe((text, image)).text
14         return [(text, response)]
15
16 demo = gr.Interface(fn=model, inputs=[gr.Image(type="pil"), gr.Textbox()],
  outputs=gr.Chatbot())
17 demo.launch()

```

```

pipeline_transformer.py pipeline.py pipeline_kv.py pipeline_llava.py gradio_llava.py X
gradio_llava.py > ...
1 import gradio as gr
2 from lmdeploy import pipeline, TurbomindEngineConfig
3
4
5 backend_config = TurbomindEngineConfig(session_len=8192) # 图片分辨率较高时请调高session_len
6 # pipe = pipeline('liuhaotian/llava-v1.6-vicuna-7b', backend_config=backend_config) 非开发机运行此命令
7 pipe = pipeline('/share/new_models/liuhaotian/llava-v1.6-vicuna-7b', backend_config=backend_config)
8
9 def model(image, text):
10     if image is None:
11         return [(text, "请上传一张图片。")]
12     else:
13         response = pipe((text, image)).text
14         return [(text, response)]
15
16 demo = gr.Interface(fn=model, inputs=[gr.Image(type="pil"), gr.Textbox()], outputs=gr.Chatbot())
17 demo.launch()

```

运行python程序。

```
1 python /root/gradio_llava.py
```

通过ssh转发一下7860端口。

```
1 ssh -CNg -L 7860:127.0.0.1:7860 root@ssh.intern-ai.org.cn -p <你的ssh端口>
```

通过浏览器访问 <http://127.0.0.1:7860>。

