

# 大语言模型部署实验报告

2253893 苗君文

## 一、部署过程

1. 登录并使用魔搭平台，获取 CPU 云计算资源；



## 2. 通过 Jupyter Notebook 进行环境配置

在环境目录/opt/conda/envs 下新建文件夹 itrex

```
root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace# cd /opt/conda/envs
root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs# mkdir itrex
```

使用 `wget` 命令复制运行环境镜像文件并在云主机的 `itrex` 目录进行运行环境的恢复

```
root@k8s-405231-7cd5886796-jc7v21:~# curl -s -o /dev/null -w "%{httpcode} %{url_effective}" https://id-al.oss-cn-hangzhou.aliyuncs.com/LLM/trex.tar.gz
2024-05-29 21:24:58 - https://id-al.oss-cn-hangzhou.aliyuncs.com/LLM/trex.tar.gz
正在解所主机 id-al.oss-cn-hangzhou.aliyuncs.com (id-al.oss-cn-hangzhou.aliyuncs.com)... 118.31.219.212
正在连接 id-al.oss-cn-hangzhou.aliyuncs.com (id-al.oss-cn-hangzhou.aliyuncs.com)[118.31.219.212]:443... 已连接。
已发出 HTTP 请求，正在等待响应... 200 OK
长度: 3063101399 (2.9G) [application/gzip]
正在保存至: trex.tar.gz

trex.tar.gz      32          %[=====]
] 941          trex.tar.gz 32K 940.85M 10.0MB/s 剩余3          trex.tar.gz 32K 942.10M 10.4MB/s 剩余3          trex.tar.gz 32K 943.88M 10.8MB/s 剩余3          trex.tar.gz 32K 944.44M 10.6MB/s 剩余3          trex
trex.tar.gz 32K 950.77M 9.81MB/s 剩余3          trex.tar.gz 32K 950.77M 9.81MB/s 剩余3          trex.tar.gz 32K 952.71M 9.40MB/s 剩余3          trex.tar.gz 32K 954.88M 9.21MB/s 剩余3          trex
tar.gz 32K 954.25M 9.37MB/s 剩余3          trex.tar.gz 32K 955.70M 9.19MB/s 剩余3          trex.tar.gz 32K 957.70M 8.70MB/s 剩余3          trex.tar.gz 32K 957.05M 8.5
2MB/s 剩余3          trex.tar.gz 32K 962.57M 8.87MB/s 剩余3          trex.tar.gz 32K 968.61M 8.67MB/s 剩余3          trex.tar.gz 32K 965.34M 9.21MB/s 剩余3          trex.tar.gz 32K 960.69M 8.70MB/s 剩余3          trex
trex.tar.gz 32K 967.52M 9.43MB/s 剩余3          trex.tar.gz 32K 970.29M 9.81MB/s 剩余3          trex.tar.gz 32K 965.34M 9.21MB/s 剩余3          trex.tar.gz 32K 960.69M 8.70MB/s 剩余3          trex
tar.gz 32K 973.43M 10.2MB/s 剩余3          trex.tar.gz 32K 982.81M 11.1MB/s 剩余3          trex.tar.gz 32K 977.11M 10.7MB/s 剩余3          trex.tar.gz 32K 970.29M 9.81MB/s 剩余3          trex.tar.gz 32K 979.90M 10.8MB/s
剩余3          trex.tar.gz 32K 982.81M 11.1MB/s 剩余3          trex.tar.gz 32K 982.81M 11.1MB/s 剩余3          trex.tar.gz 32K 985.14M 11.3MB/s 剩余3          trex.tar.gz 32K 979.90M 10.8MB/s
trex.tar.gz 32K 968.98M 11.2MB/s 剩余3          trex.tar.gz 32K 968.98M 11.2MB/s 剩余3          trex.tar.gz 32K 988.31M 11.0MB/s 剩余3          trex.tar.gz 32K 985.14M 11.3MB/s 剩余3          trex.tar.gz 32K 996.93M 11.1MB/s
4K 992.21M 11.2MB/s 剩余3          trex.tar.gz 34K 994.92M 11.2MB/s 剩余3          trex.tar.gz 32K 991.67M 11.2MB/s 剩余3          trex.tar.gz 34K 996.93M 11.1MB/s 剩余3          trex.tar.gz 34K 1000M 12.0MB/s 剩余3          trex
剩余3          trex.tar.gz 34K 1001M 11.6MB/s 剩余3          trex.tar.gz 34K 1007M 12.3MB/s 剩余3          trex.tar.gz 100K[=====] 2.85G 19.5MB/s 用时3
n 39s

2024-05-29 21:28:37 (12.3 MB/s) - 已保存 trex.tar.gz [3063101399/3063101399]
```

## 解压文件

```
lib/python3.10/site-packages/rapidfuzz/distance/Jaro_py.py
lib/python3.10/site-packages/langchain_community/document_loaders/blob_loaders/schema.py
lib/python3.10/site-packages/torchaudio/prototype/models/__pycache__/__init__.cpython-310.pyc
lib/python3.10/site-packages/langchain/vectorstores/__pycache__/timescalevector.cpython-310.pyc
lib/python3.10/site-packages/sympy/utilities/__pycache__/decorator.cpython-310.pyc
lib/python3.10/site-packages/sklearn/tests/test_naive_bayes.py
lib/python3.10/site-packages/pandas/tests/io/parser/test_index_col.py
lib/python3.10/site-packages/transformers/models/unispeech_sat/__pycache__/configuration_unispeech_sat.cpython-310.pyc
lib/python3.10/site-packages/kubernetes/client/__pycache__/api_client.cpython-310.pyc
lib/python3.10/site-packages/mpl_toolkits/axes_grid1/axes_size.py
lib/python3.10/site-packages/torchaudio/prototype/functional/__pycache__/__init__.cpython-310.pyc
lib/python3.10/site-packages/transformers/utils/__pycache__/dummy_flax_objects.cpython-310.pyc
lib/python3.10/site-packages/optimum/commands/neural_compressor/__pycache__/quantize.cpython-310.pyc
lib/python3.10/site-packages/typepy/checker/__pycache__/realnumber.cpython-310.pyc
lib/python3.10/site-packages/flatbuffers-24.3.25.dist-info/METADATA
bin/activate
bin/deactivate
bin/conda-unpack
root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs#
```

## 激活环境

```
root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs# conda activate itrex
(itrex) root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs#
```

## 安装对应的 kernel

```
(itrex) root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs# python -m ipykernel install --name itrex
Installed kernelspec itrex in /usr/local/share/jupyter/kernels/itrex
```

## 3. LLM 下载

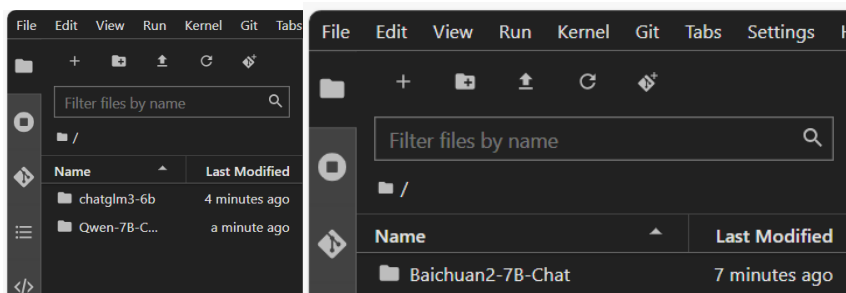
### 切换至工作目录

```
(itrex) root@dsw-485231-7cc5886795-jc7v2:/opt/conda/envs# cd /mnt/workspace
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace#
```

### git clone 下载中文大模型至本地

```
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace# git clone https://www.modelscope.cn/ZhipuAI/chatglm3-6b.git
git clone https://www.modelscope.cn/qwen/Qwen-7B-Chat.git
正克隆到 'chatglm3-6b'...
remote: Enumerating objects: 140, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 140 (delta 8), reused 1 (delta 0), pack-reused 122
接收对象中: 100% (140/140), 61.16 KiB | 7.64 MiB/s, 完成.
处理 delta 中: 100% (60/60), 完成.
过滤内容: 100% (15/15), 23.26 GiB | 404.57 MiB/s, 完成.
正克隆到 'Qwen-7B-Chat'...
remote: Enumerating objects: 551, done.
remote: Counting objects: 100% (53/53), done.
remote: Compressing objects: 100% (27/27), done.
接收对象中: 100% (551/551), 16.47 MiB | 175.00 KiB/s, 完成.
remote: Total 551 (delta 28), reused 50 (delta 26), pack-reused 498
处理 delta 中: 100% (292/292), 完成.
过滤内容: 100% (8/8), 14.38 GiB | 375.96 MiB/s, 完成.
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace#
```

```
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace# git clone https://www.modelscope.cn/baichuan-inc/Baichuan2-7B-Chat.git
正克隆到 'Baichuan2-7B-Chat'...
remote: Enumerating objects: 115, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 115 (delta 3), reused 0 (delta 0), pack-reused 100
接收对象中: 100% (115/115), 467.93 KiB | 22.28 MiB/s, 完成.
处理 delta 中: 100% (48/48), 完成.
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace#
```



## 二、构建 chatbot 并进行问答测试

### 1. 基于 chatglm3-6b 模型

量化并构建 chatbot

```
root@dsw-485231-7cc58867 X test.ipynb X +
+  ✂  📄  ▶  ⏮  ⏭  Code  ⏱  git

# Build chatbot with INT4 weight-only quantization, computations in AMX INT8
from intel_extension_for_transformers.transformers import RtnConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import PipelineConfig

config = PipelineConfig(model_name_or_path="./chatglm3-6b",
                        optimization_config=
                        RtnConfig(bits=4, compute_dtype="int8",
                                weight_dtype="int4_fullrange"))

chatbot = build_chatbot(config)
```

使用 chatbot 得到回答

1) 上海有哪些景点

```
[6] # Perform inference/generate a response
response = chatbot.predict(query="上海有哪些景点")
print(response)

/opt/conda/envs/itrex/lib/python3.10/site-packages/torch/amp/autocast_mode.py:267: UserWarning: In CPU autocast, but the target dtype is not supported. Disabling autocast.
CPU Autocast only supports dtype of torch.bfloat16, torch.float16 currently.
warnings.warn(error_message)

上海是中国著名的旅游胜地，拥有许多著名景点。以下是一些值得一游的景点：东方明珠塔、外滩、南京路步行街、人民广场、城隍庙、上海博物馆、东方艺术中心、上海迪士尼乐园等。
```

2) 请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少 2、夏天：能穿多少穿多少

```
# Perform inference/generate a response
response = chatbot.predict(query="请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少 2、夏天：能穿多少穿多少")
print(response)

这两句话的意思是相同的，都是在询问在寒冷的冬天和炎热的夏天，人们应该穿多少衣服。但是，冬天和夏天的气候条件不同，所以人们在选择穿什么衣服上有不同的考虑。冬天通常很冷，所以需要穿厚实的外套、保暖内衣和舒适的鞋子等；而夏天则很热，人们需要穿轻便的衣服、短裤和凉鞋等。因此，虽然这两句话的意思相同，但它们所反映的气候条件和人们对穿着的需求不同。
```

3) 请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上

```
# Perform inference/generate a response
response = chatbot.predict(query="请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上")
print(response)

这两句话的语义重复。第一句“单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上”表达的意思是单身的原因有两个，一个是看不上别人，另一个是别人看不上自己。第二句“单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上”与第一句意思相同，只是表述方式略有不同。因此，这两句话在语义上是重复的。
```

4) 他知道我知道你知道他不知道吗？ 这句话里，到底谁不知道

```
# Perform inference/generate a response
response = chatbot.predict(query="他知道我知道你知道他不知道吗？ 这句话里，到底谁不知道")
print(response)

在这个问题中，“他”是指一个人。根据问题的表述，无法确定“他”具体是谁。可能是一个朋友、家人、同事等。在这个情境下，我们无法判断“他”是否知道“我”和“你知道”的事情。这个问题存在一定的模糊性，因此无法给出一个明确的答案。
```

5) 明明明明白白白喜欢他，可她就是不说话。 这句话里，明明和白白谁喜欢谁？

```
# Perform inference/generate a response
response = chatbot.predict(query="明明明明白白白喜欢他，可她就是不说话。 这句话里，明明和白白谁喜欢谁？")
print(response)

根据句子中的信息，无法确定是明明还是白白主动表达了自己的感情。可能是明明主动告诉白白自己的感情，也可能是白白自己告诉明明自己的感情。
```

6) 领导：你这是什么意思？ 小明：没什么意思。意思意思。 领导：你这就不够意思了。  
小明：小意思，小意思。领导：你这人真有意思。 小明：其实也没有别的意思。 领导：那我就不好意思了。 小明：是我不好意思。请问：以上“意思”分别是什么意思。

```
53 秒 # Perform inference/generate a response
response = chatbot.predict(query="""领导：你这是什么意思？ 小明：没什么意思。意思意思。_
领导：你这就不够意思了。小明：小意思，小意思。
领导：你这人真有意思。小明：其实也没有别的意思。
领导：那我就不好意思了。小明：是我不好意思。
请问：以上“意思”分别是什么意思。""")
print(response)
```

1. “意思”第一层意思是“含义”，领导问小明的话的意思是询问小明话语的含义。  
2. “意思”第二层意思是“意味”，领导的意思是询问小明话语的意味，即是否有其他深层次的意义或含义。  
3. “意思”第三层意思是“用意”，领导的意思是询问小明话语背后的用意，即他为什么要说这样的话。

## 2. 基于 Qwen-7B-Chat 模型

量化并构建 chatbot

```
8分 9 秒 # Build chatbot with INT4 weight-only quantization, computations in AMX INT8
from intel_extension_for_transformers.transformers import RtnConfig
from intel_extension_for_transformers.neural_chat import build_chatbot
from intel_extension_for_transformers.neural_chat import PipelineConfig

config = PipelineConfig(model_name_or_path="./Qwen-7B-Chat",
                        optimization_config=
                            RtnConfig(bits=4, compute_dtype="int8",
                                      weight_dtype="int4_fullrange"))

chatbot = build_chatbot(config)
```

使用 chatbot 得到回答

1) 上海有哪些景点

```
44 秒 # Perform inference/generate a response
response = chatbot.predict(query="上海有哪些景点")
print(response)
```

/opt/conda/envs/itrex/lib/python3.10/site-packages/torch/amp/autocast\_mode.py:267: UserWarning: In CPU autocast, but the target dtype is not supported. Disabling autocast.  
CPU Autocast only supports dtype of torch.bfloat16, torch.float16 currently.  
warnings.warn(error\_message)

上海有许多著名的景点，例如外滩、东方明珠电视塔、南京路步行街、豫园、田子坊等。此外，还有许多其他有趣的景点，如上海野生动物园、上海科技馆、上海海洋水族馆、上海欢乐谷等。

2) 请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少 2、夏天：能穿多少穿多少

```
# Perform inference/generate a response
response = chatbot.predict(query="请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少；2、夏天：能穿多少穿多少")
print(response)
```

这两句话的区别在于它们描述的季节不同。第一句话“冬天：能穿多少穿多少”暗示着在寒冷的冬季，人们需要尽可能多地穿上衣服来保暖。而第二句话“夏天：能穿多少穿多少”则表示在炎热的夏季，人们也需要尽可能地减少衣物以保持凉爽。因此，这两句话所表达的意思是相反的。

3) 请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上

```
# Perform inference/generate a response
response = chatbot.predict(query="请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上")
print(response)
```

这两句话的区别在于它们的语境和含义不同。第一句话是在描述一种现象，即有些人因为找不到合适的伴侣而被称为“单身狗”，并解释了产生这种现象的原因是“谁都不喜欢”。而第二句话则是在表达一个人的观点或态度，即认为自己之所以成为“单身狗”是因为“谁都不喜欢”，并且这种观点可能带有消极情绪。因此，虽然这两句话都提到了“单身狗”和“谁都不喜欢”，但它们的语气、情感色彩和语境都有所不同。

4) 他知道我知道你知道他不知道吗？这句话里，到底谁不知道

```
# Perform inference/generate a response
response = chatbot.predict(query="他知道我知道你知道他不知道吗？这句话里，到底谁不知道")
print(response)
```

这句话的含义是：这个人知道我知道你知道他不知道。因此，这个人的状态是知道我们知道他知道他不知道。

5) 明明明明明白白白喜欢他，可她就是不说话。这句话里，明明和白白谁喜欢谁？

```
# Perform inference/generate a response
response = chatbot.predict(query="明明明明明白白白喜欢他，可她就是不说话。这句话里，明明和白白谁喜欢谁？")
print(response)
```

根据句子的描述，“明明”喜欢“他”，而“白白”没有明确表示自己喜欢谁。因此，可以得出结论：明明喜欢他。

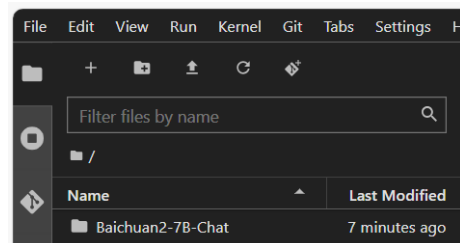
6) 领导：你这是什么意思？ 小明：没什么意思。意思意思。 领导：你这就不够意思了。 小明：小意思，小意思。领导：你这人真有意思。 小明：其实也没有别的意思。 领导：那我就不好意思了。 小明：是我不好意思。 请问：以上“意思”分别是什么意思。



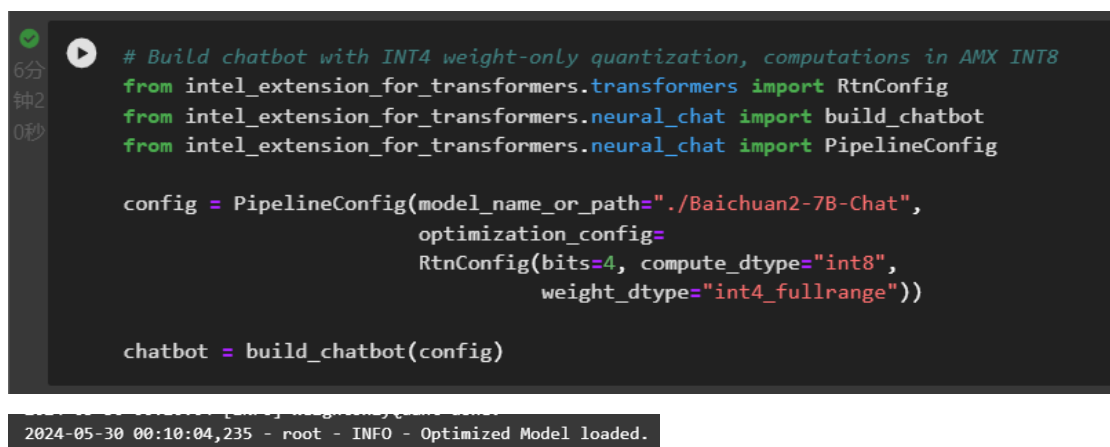
### 3. 基于 Baichuan2-7B-Chat 模型

Git clone

```
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace# git clone https://www.modelscope.cn/baichuan-inc/Baichuan2-7B-Chat.git
正克隆到 'Baichuan2-7B-Chat'...
remote: Enumerating objects: 115, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (15/15), done.
remote: Total 115 (delta 3), reused 0 (delta 0), pack-reused 100
接收对象中: 100% (115/115), 467.93 KiB | 22.28 MiB/s, 完成。
处理 delta 中: 100% (48/48), 完成。
(itrex) root@dsw-485231-7cc5886795-jc7v2:/mnt/workspace#
```

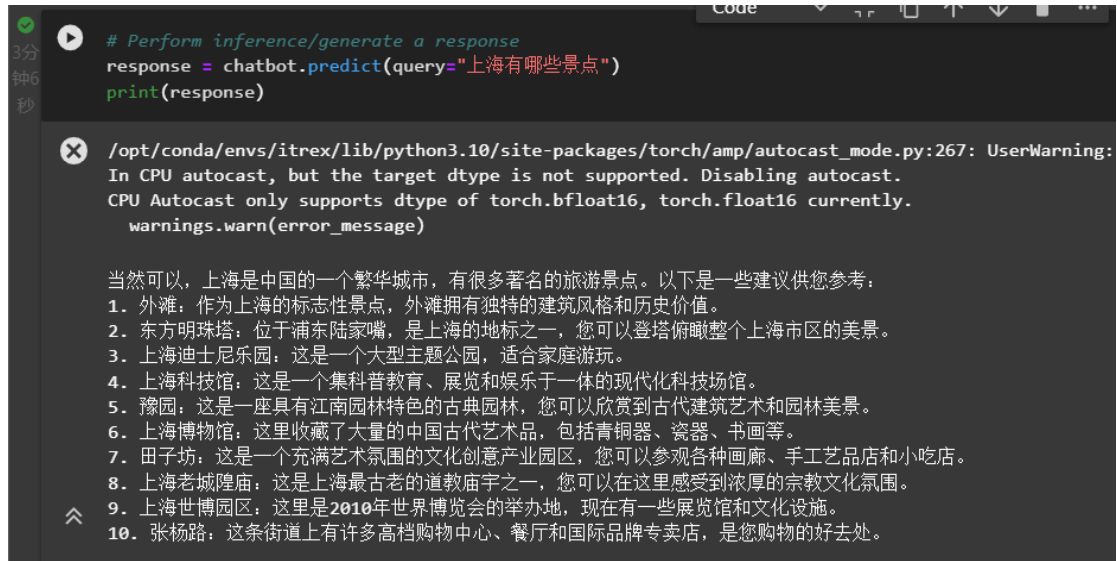


量化并构建 chatbot



使用 chatbot 得到回答

1) 上海有哪些景点



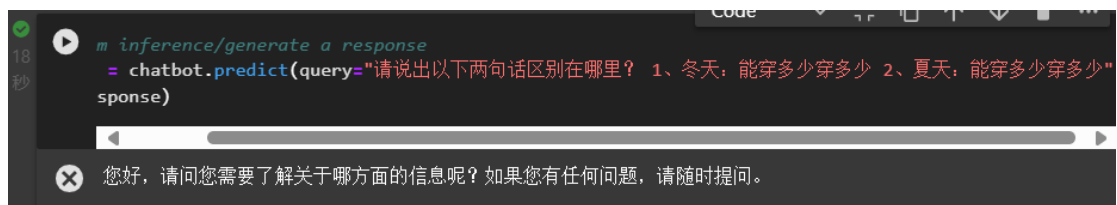
```
# Perform inference/generate a response
response = chatbot.predict(query="上海有哪些景点")
print(response)
```

/opt/conda/envs/itrex/lib/python3.10/site-packages/torch/amp/autocast\_mode.py:267: UserWarning: In CPU autocast, but the target dtype is not supported. Disabling autocast. CPU Autocast only supports dtype of torch.bfloat16, torch.float16 currently. warnings.warn(error\_message)

当然可以，上海是中国的一个繁华城市，有很多著名的旅游景点。以下是一些建议供您参考：

1. 外滩：作为上海的标志性景点，外滩拥有独特的建筑风格和历史价值。
2. 东方明珠塔：位于浦东陆家嘴，是上海的地标之一，您可以登塔俯瞰整个上海市区的美景。
3. 上海迪士尼乐园：这是一个大型主题公园，适合家庭游玩。
4. 上海科技馆：这是一个集科普教育、展览和娱乐于一体的现代化科技场馆。
5. 豫园：这是一座具有江南园林特色的古典园林，您可以欣赏到古代建筑艺术和园林美景。
6. 上海博物馆：这里收藏了大量的中国古代艺术品，包括青铜器、瓷器、书画等。
7. 田子坊：这是一个充满艺术氛围的文化创意产业园区，您可以参观各种画廊、手工艺品店和小吃店。
8. 上海老城隍庙：这是上海最古老的道教庙宇之一，您可以在这里感受到浓厚的宗教文化氛围。
9. 上海世博园区：这里是2010年世界博览会的举办地，现在有一些展览馆和文化设施。
10. 张杨路：这条街道上有许多高档购物中心、餐厅和国际品牌专卖店，是您购物的好去处。

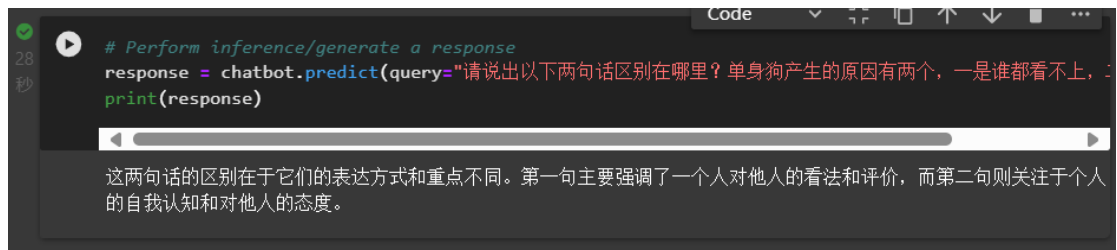
2) 请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少 2、夏天：能穿多少穿多少



```
m inference/generate a response
= chatbot.predict(query="请说出以下两句话区别在哪里？ 1、冬天：能穿多少穿多少 2、夏天：能穿多少穿多少")
sponse)
```

您好，请问您需要了解关于哪方面的信息呢？如果您有任何问题，请随时提问。

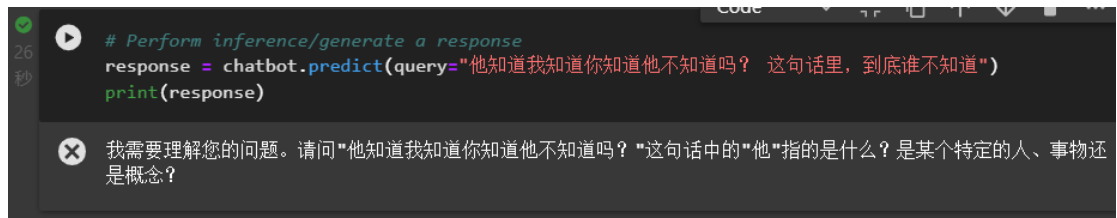
3) 请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上



```
# Perform inference/generate a response
response = chatbot.predict(query="请说出以下两句话区别在哪里？单身狗产生的原因有两个，一是谁都看不上，二是谁都看不上")
print(response)
```

这两句话的区别在于它们的表达方式和重点不同。第一句主要强调了一个人对他人的看法和评价，而第二句则关注于个人的自我认知和对他人的态度。

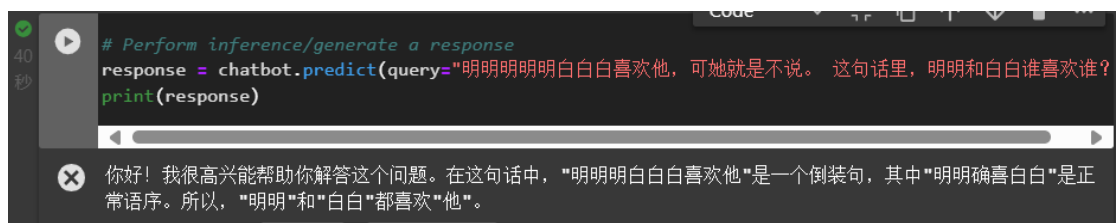
4) 他知道我知道你知道他不知道吗？ 这句话里，到底谁不知道



```
# Perform inference/generate a response
response = chatbot.predict(query="他知道我知道你知道他不知道吗？ 这句话里，到底谁不知道")
print(response)
```

我需要理解您的问题。请问“他知道我知道你知道他不知道吗？”这句话中的“他”指的是什么？是某个特定的人、事物还是概念？

5) 明明明明白白白喜欢他，可她就是不说话。 这句话里，明明和白白谁喜欢谁？



```
# Perform inference/generate a response
response = chatbot.predict(query="明明明明白白白喜欢他，可她就是不说话。 这句话里，明明和白白谁喜欢谁？")
print(response)
```

你好！我很高兴能帮助你解答这个问题。在这句话中，“明明明明白白白喜欢他”是一个倒装句，其中“明明明确白白白”是正常语序。所以，“明明”和“白白”都喜欢他。







### 3. 针对具体问答的对比分析

#### 1) 上海有哪些景点

ChatGLM3-6B: 回答准确, 涵盖了主要的旅游景点, 信息全面。

Qwen-7B-Chat: 回答详细, 提供了更多的景点, 覆盖面广, 包含了部分非主流但有趣的景点。

Baichuan2-7B-Chat: 回答详细且丰富, 包含了多个类型的景点, 回答内容全面且多样化, 简单介绍了每个景点值得游玩的理由。

#### 2) 冬天: 能穿多少穿多少 vs. 夏天: 能穿多少穿多少

ChatGLM3-6B: 解释准确但有点重复, 缺乏深入分析, 没有解释清楚“多少”和“多‘少’”在两种语境下的区别。

Qwen-7B-Chat: 描述了季节不同, 冬天需要尽可能多穿以保暖, 夏天需要尽可能少穿以保持凉爽。解释清晰, 准确地指出了反向意义, 且逻辑清晰、理由充分。

Baichuan2-7B-Chat: 表现不佳, 未能理解问题, 未能提供有效回答, 未能识别问题的具体含义。

#### 3) 单身狗产生的原因有两个, 一是谁都看不上, 二是谁都看不上

ChatGLM3-6B: 解释准确, 指出了一个是不会被别人看上, 一个是看不上别人, 但是对句子结构的认识有重复, 认为出现了两句话。。

Qwen-7B-Chat: 解释细致, 不仅解释了含义不同及其背后原因, 而且涵盖了语气和情感色彩的不同。

Baichuan2-7B-Chat: 回答内容强调表达方式和重点不同, 但解释不够具体, 缺乏深入分析。

#### 4) 他知道我知道你知道他不知道吗?

ChatGLM3-6B: 对问题分析的侧重点不正确, 没有给出明确答案, 无法处理复杂的嵌套逻辑, 表现不佳。

Qwen-7B-Chat: 解释清楚, 指出这个人知道其他人知道他不知道。逻辑分析清晰, 能理解复杂嵌套。

Baichuan2-7B-Chat: 未能提供有效回答, 问题理解有误。无法处理复杂的嵌套逻辑, 表现不佳。

#### 5) 明明明明白白白喜欢他, 可她就是不说。

ChatGLM3-6B: 无法确定明明还是白白喜欢他。未能准确理解句子结构和逻辑。

Qwen-7B-Chat: 理解了问题的意思, 但是分析有误, 未能正确解析句子结构。

Baichuan2-7B-Chat: 认为句子语序有误, 分析角度错误, 理解有误, 未能正确解析句子。

#### 6) 领导和小明的对话中“意思”分别是什么意思

ChatGLM3-6B: 解释了不同语境下“意思”的含义。但是没有解释所有出现的“意思”, 仅解释了三个, 分析不够全面。

Qwen-7B-Chat: 解释了不同语境下“意思”的含义。解释详细且准确, 涵盖了各语境下的不同意思, 但依旧缺少了一些“意思”的解释。

Baichuan2-7B-Chat: 仅解释了“意思”可能表示的含义礼貌、客气、幽默等。解释泛泛而谈, 缺乏具体上下文的详细分析。

#### 4. 对比总结与评价

**ChatGLM3-6B:** 总体表现稳定，在信息完整性和解释准确性方面表现较好，对于具体的知识性问题和简单的语义问题回答较好，但在复杂逻辑和嵌套问题、复杂句子结构的处理上稍显不足。

**Qwen-7B-Chat:** 回答详细且准确，能够很好地理解并解释复杂语句和逻辑关系，是三者中表现最优的模型。尤其在复杂问题和逻辑分析方面展现了强大的能力，适合需要深入解释和逻辑分析的应用场景。

**Baichuan2-7B-Chat:** 在多样化和详细回答方面有一定优势，但在处理复杂逻辑和语义分析问题表现欠佳，存在理解偏差，更适合娱乐问答和需要详细回答的基础对话场景。

用户可以根据具体应用场景使用不同的大语言模型：

对于一般信息查询和问答任务，推荐使用 **Qwen-7B-Chat**，因其详细且准确的回答能力。

对于涉及复杂逻辑和多层嵌套的问答，首选 **Qwen-7B-Chat**，其强大的逻辑分析能力能够准确解答；其次可以选择 **ChatGLM3-6B** 作为备选。

对于需要丰富多样的回答内容，可以参考 **Baichuan2-7B-Chat**，但需注意其在复杂问题上的表现。

三款模型各有优势，适用于不同的应用场景。**Qwen-7B-Chat** 在多数测试场景中表现最佳，尤其适合需要深入分析和解释的场景；**ChatGLM3-6B** 在知识性问答和简单语义处理上表现出色，适合基础问答场景；**Baichuan2-7B-Chat** 则在提供多样化回答方面具有优势，更适用于娱乐和基础对话场景。根据具体需求选择合适的模型，可以最大化发挥其性能优势。

#### 5. 优化建议

**Qwen-7B-Chat:** 可以通过优化信息量管理，避免信息过载，同时进一步提升在非常复杂语境下的解析能力。

**ChatGLM3-6B:** 可以加强对复杂逻辑和嵌套语义的处理能力，提升其在深度分析问题上的表现。

**Baichuan2-7B-Chat:** 需要提升对复杂逻辑和语义结构的理解能力，同时保持其在多样化回答方面的优势。

通过进一步优化和增强这些方面，这三种模型都可以在各自的适用场景中提供更为优质的服务。