

人工智能导论第四次作业

一、环境搭建

1. 基础环境配置

打开 terminal

输入

```
pip install \
torch==2.3.0+cpu \
torchvision==0.18.0+cpu \
--index-url https://download.pytorch.org/whl/cpu
```

安装基础依赖

```
pip install -U pip setuptools wheel
# 安装基础依赖（兼容 transformers 4.33.3 和 neuralchat）
pip install \
"intel-extension-for-transformers==1.4.2" \
"neural-compressor==2.5" \
"transformers==4.33.3" \
"modelscope==1.9.5" \
"pydantic==1.10.13" \
"sentencepiece" \
"tiktoken" \
"einops" \
"transformers_stream_generator" \
"uvicorn" \
"fastapi" \
"yacs" \
"setuptools_scm"
# 安装 fschat（需要启用 PEP517 构建）
pip install fschat --use-pep517
```

可选：安装 tqdm、huggingface-hub 等增强体验

```
pip install tqdm huggingface-hub
```

二、大模型实践

（一）chatglm3-6B 环境配置

1. 切换到数据目录

```
cd /mnt/data
```

2. 下载对应大模型

```
git clone https://www.modelscope.cn/hipuAI/chatglm3-6b.git
```

```
root@dsw-1136822-84d9544cfd-qwr21:/mnt/workspace# git clone https://www.modelscope.cn/ZhipuAI/chatglm3-6b.git
正克隆到 'chatglm3-6b'...
remote: Enumerating objects: 140, done.
remote: Total 140 (delta 0), reused 0 (delta 0), pack-reused 140
接收对象中: 100% (140/140), 47.10 KiB | 382.00 KiB/s, 完成.
处理 delta 中: 100% (64/64), 完成.
过滤内容: 100% (15/15), 23.26 GiB | 159.27 MiB/s, 完成.
root@dsw-1136822-84d9544cfd-qwr21:/mnt/workspace#
```

3. 构建实例

切换工作目录

```
cd /mnt/workspace
```

4. 实例代码:

编写推理脚本 run_zhipuai_cpu.py:

```
from transformers import TextStreamer, AutoTokenizer, AutoModelForCausalLM
model_name = "/mnt/data/chatglm3-6b" # 本地路径
prompt = "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"
tokenizer = AutoTokenizer.from_pretrained(
    model_name,
    trust_remote_code=True
)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    trust_remote_code=True,
    torch_dtype="auto" # 自动选择 float32/float16（根据模型配置）
).eval()
inputs = tokenizer(prompt, return_tensors="pt").input_ids
streamer = TextStreamer(tokenizer)
outputs = model.generate(inputs, streamer=streamer, max_new_tokens=300)
```

(二) Qwen-7B-Chat环境配置

1. 切换到数据目录

```
cd /mnt/data
```

2. 下载对应大模型

```
git clone https://www.modelscope.cn/qwen/Qwen-7B-Chat.git
```

```
(qwen_env) root@dsw-1135618-59c79b444b-vgxt4:/mnt/workspace# cd /mnt/data
(qwen_env) root@dsw-1135618-59c79b444b-vgxt4:/mnt/data# git clone https://www.modelscope.cn/qwen/Qwen-7B-Chat.git
正克隆到 'Qwen-7B-Chat'...
remote: Enumerating objects: 554, done.
remote: Counting objects: 100% (56/56), done.
remote: Compressing objects: 100% (30/30), done.
remote: Total 554 (delta 30), reused 49 (delta 26), pack-reused 498
接收对象中: 100% (554/554), 16.47 MiB | 595.00 KiB/s, 完成.
处理 delta 中: 100% (294/294), 完成.
正在更新文件: 100% (31/31), 完成.
过滤内容: 100% (8/8), 14.38 GiB | 156.91 MiB/s, 完成.
```

3. 构建实例

切换工作目录

```
cd /mnt/workspace
```

4. 实例代码：

编写推理脚本 run_qwen_cpu.py：

```
from transformers import TextStreamer, AutoTokenizer, AutoModelForCausalLM
model_name = "/mnt/data/Qwen-7B-Chat" # 本地路径
prompt = "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"
tokenizer = AutoTokenizer.from_pretrained(
    model_name,
    trust_remote_code=True
)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    trust_remote_code=True,
    torch_dtype="auto" # 自动选择 float32/float16（根据模型配置）
).eval()
inputs = tokenizer(prompt, return_tensors="pt").input_ids
streamer = TextStreamer(tokenizer)
outputs = model.generate(inputs, streamer=streamer, max_new_tokens=300)
```

三、运行实例与结果

```
python run_qwen_cpu.py
```

Qwen-7B-Chat回复

- 问题一：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

这两句话的区别在于它们强调的是在不同季节中的穿衣习惯或策略。

"冬天：能穿多少穿多少"这句话强调的是在寒冷的冬季，人们应该尽可能地多穿衣服以保暖，因为天气冷。这意味着可能会选择穿更多的外套、毛衣和裤子等厚重衣物。

而"夏天：能穿多少穿多少"这句话则是在炎热的夏季，人们也应该尽量少穿衣服以保持凉爽，因为天气热。这意味着可能会选择穿更轻薄的衣服，如短袖衬衫、短裤和凉鞋等。

因此，这两句话的主要区别在于他们描述的是不同的气候条件，并建议相应地调整穿衣风格。

```
(qwen_env) root@dsw-1135618-59c79b444b-vgxt4:/mnt/data# cd /mnt/workspace
(qwen_env) root@dsw-1135618-59c79b444b-vgxt4:/mnt/workspace# python run_qwen_cpu.py
/opt/conda/envs/qwen_env/lib/python3.10/site-packages/transformers/utils/generic.py:311: UserWarning: torch.utils._pytree._register_pytree_node is deprecated. Please use torch.
utils._pytree.register_pytree_node instead.
  torch.utils._pytree._register_pytree_node(
Loading checkpoint shards: 100%
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
这两句话的区别在于它们强调的是在不同季节中的穿衣习惯或策略。
“冬天：能穿多少穿多少”这句话强调的是在寒冷的冬季，人们应该尽可能地多穿衣服以保暖，因为天气冷。这意味着可能会选择穿更多的外套、毛衣和裤子等厚重衣物。
而“夏天：能穿多少穿多少”这句话则是在炎热的夏季，人们也应该尽量少穿衣服以保持凉爽，因为天气热。这意味着可能会选择穿更轻薄的衣服，如短袖衬衫、短裤和凉鞋等。
因此，这两句话的主要区别在于他们描述的是不同的气候条件，并建议相应地调整穿衣风格。</endoftext>
```

- 问题二：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，二是谁都看不上自己

这两句话的区别在于：

“单身狗产生的原因有两个”是在客观地陈述一种现象或事实，而“谁都看不上，二是谁都看不上自己”则是主观地表达了一个人的态度和感受。第一个句子是中立的，第二个句子则带有个人情感色彩。

在语言表达上，第一个句子更注重逻辑性和准确性，而第二个句子则更加口语化和个性化。同时，第一个句子更适合用于描述事实或情况，而第二个句子则更适合用于表达个人观点或情绪。

```
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，二是谁都看不上自己
这两句话的区别在于：
“单身狗产生的原因有两个”是在客观地陈述一种现象或事实，而“谁都看不上，二是谁都看不上自己”则是主观地表达了一个人的态度和感受。第一个句子是中立的，第二个句子则带有个人情感色彩。
在语言表达上，第一个句子更注重逻辑性和准确性，而第二个句子则更加口语化和个性化。同时，第一个句子更适合用于描述事实或情况，而第二个句子则更适合用于表达个人观点或情绪。</endoftext>
```

- 问题三：他知道我知道你知道他不知道吗？这句话里，到底谁不知道谁知道谁？ 答案：

chatglm3-6B回复

- 问题一：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

这两句话的区别在于，第一句是关于冬天的，而第二句是关于夏天的。虽然两句话都在描述穿多少衣服，但是它们所针对的季节不同。

```
torch.utils._pytree._register_pytree_node(
Loading checkpoint shards: 100%
[gMASK]sop 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
这两句话的区别在于，第一句是关于冬天的，而第二句是关于夏天的。虽然两句话都在描述穿多少衣服，但是它们所针对的季节不同。
```

- 问题二：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，二是谁都看不上自己

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。

这两个句子在表达意思上有什么区别？

这两个句子在表达意思上有一定的区别。

第一句话“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。”中，第一个“谁”是指代“人”，表示单身狗的原因是单身狗自己看不上别人，同时别人也看不上单身狗。第二个“谁”是指代“人”，表示单身狗的原因是别人看不上单身狗，同时单身狗也看不上别人。

第二句话“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。”中，两个“谁”都指代“人”，表示单身狗的原因是单身狗自己看不上别人，同时别人也看不上单身狗。但是，第一句中的“谁”是主动的，自己看不上别人，而第二句中的“谁”是被动的，别人看不上自己。

```
[gMASK]sop 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。
这两个句子在表达意思上有什么区别？
这两个句子在表达意思上有一定的区别。

第一句话“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。”中，第一个“谁”是指代“人”，表示单身狗的原因是单身狗自己看不上别人，同时别人也看不上单身狗。第二个“谁”是指代“人”，表示单身狗的原因是别人看不上单身狗，同时单身狗也看不上别人。

第二句话“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。”中，两个“谁”都指代“人”，表示单身狗的原因是单身狗自己看不上别人，同时别人也看不上单身狗。但是，第一句中的“谁”是主动的，自己看不上别人，而第二句中的“谁”是被动的，别人看不上自己。
root@dsw-1136822-84d9544cfd-qwr21:/mnt/workspace#
```

- 问题三：他知道我知道你知道他不知道吗？这句话里，到底谁不知道谁知道谁？ 答案：

他知道我知道你知道他不知道吗？这句话里，到底谁不知道的事情是什么？

这句话是一个典型的谜语或者绕口令,它的含义是:一个人知道另外一个人知道某些事情,但是这个人自己并不知道这些事情。

在这个绕口令中,到底谁不知道的事情是“他不知道的事情”。因为这个人知道自己知道某些事情,但是并不知道不知道自己不知道某些事情。

```
[gMASK]sop 他知道我知道你知道他不知道吗？这句话里，到底谁不知道的事情是什么？
这句话是一个典型的谜语或者绕口令,它的含义是:一个人知道另外一个人知道某些事情,但是这个人自己并不知道这些事情。

在这个绕口令中,到底谁不知道的事情是“他不知道的事情”。因为这个人知道自己知道某些事情,但是并不知道不知道自己不知道某些事情。
```

- 问题四：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？

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？

这句话里,明明喜欢白白。明明明白白白喜欢他,可她就是不说话。

```
Loading checkpoint shards: 100%
[gMASK]sop 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？
这句话里,明明喜欢白白。明明明白白白喜欢他,可她就是不说话。
root@dsw-1136822-84d9544cfd-qwr21:/mnt/workspace#
```

- 问题五：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。

1. 领导：你这是什么意思？ - 意思：这里的意思是指含义、目的或作用。领导在问小明到底是什么意思。

2. 小明：没什么意思。意思意思。 - 意思：这里的意思是表示没有特别的含义、目的或作用。小明在回答领导的问题，表示自己的回答没有什么特别的意义。

3. 领导：你这就不够意思了。 - 意思：这里的意思是指不够充分、不够满足。领导认为小明的回答不够充分，表示不满。
4. 小明：小意思，小意思。 - 意思：这里的意思是表示事情不太大，没有太大意义。小明在回应领导的问题，表示自己的回答只是一个小事，没有太大的意义。
5. 领导：你这人真有意思。 - 意思：这里的意思是表示对某人的行为、性格或想法表示赞赏。领导认为小明很有趣，很有意思。
6. 小明：其实也没有别的意思。 - 意思：这里的意思是表示自己的回答确实没有什么特别的含义、目的或作用。小明在回应领导的问题，表示自己的回答并没有其他特别的意义。
7. 领导：我就不好意思了。 - 意思：这里的意思是表示对某事或某人的态度。领导在这里表示自己不好意思了，可能是因为对小明的回答感到意外或困惑。
8. 小明：是我不好意思。 -

```
Loading checkpoint shards: 100% | 7/7 [00:05<00:00, 1.36it/s]
[gMASK]sop 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。
1. 领导：你这是什么意思？ - 意思：这里的意思是指含义、目的或作用。领导在问小明到底是什么意思。
2. 小明：没什么意思。意思意思。 - 意思：这里的意思是表示没有特别的含义、目的或作用。小明在回答领导的问题，表示自己的回答没有什么特别的意义。
3. 领导：你这就不够意思了。 - 意思：这里的意思是指不够充分、不够满足。领导认为小明的回答不够充分，表示不满。
4. 小明：小意思，小意思。 - 意思：这里的意思是表示事情不太大，没有太大意义。小明在回应领导的问题，表示自己的回答只是一个小事，没有太大的意义。
5. 领导：你这人真有意思。 - 意思：这里的意思是表示对某人的行为、性格或想法表示赞赏。领导认为小明很有趣，很有意思。
6. 小明：其实也没有别的意思。 - 意思：这里的意思是表示自己的回答确实没有什么特别的含义、目的或作用。小明在回应领导的问题，表示自己的回答并没有其他特别的意义。
7. 领导：我就不好意思了。 - 意思：这里的意思是表示对某事或某人的态度。领导在这里表示自己不好意思了，可能是因为对小明的回答感到意外或困惑。
8. 小明：是我不好意思。 -
root@ds-1136822-84d9544cfd-qwr21:/mnt/workspace#
```

四、大模型横向对比

1.对话内容对比

问题一：

问题内容：请说出“冬天：能穿多少穿多少”与“夏天：能穿多少穿多少”的区别。

Qwen-7B-Chat：

- 结构上清晰对比冬天/夏天的穿衣逻辑。
- 明确指出冬天是“穿得越多越好”、夏天则是“穿得越少越好”，解释了语言的“反语”性质。

ChatGLM3-6B：

- 明确指出季节不同，但解释偏向表面，没有深入语用层面。
- 没有揭示两句话表面相同而意义相反的幽默反讽效果。

问题二：

问题内容：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，二是谁都看不上自己。

Qwen-7B-Chat：

- 解释两个成分所传达的情感与语气不同：一个是中性描述，另一个是富有主观色彩。

ChatGLM3-6B：

- 对句子进行结构解释，尝试分析“主动vs被动”的意义。

问题三：

问题内容：他知道我知道你知道他不知道吗？这句话里，到底谁不知道谁知道谁？

Qwen-7B-Chat：

- 尝试梳理“我-你-他”之间的认知关系，进行逻辑链条的层层分析。
- 表述复杂，仍保留一定模糊性，但体现推理过程。

ChatGLM3-6B：

- 直接指出“他不知道”的成分是中心问题，但没有逐层展开谁知道谁。
- 更像在泛化地解释这是个绕口令，语义推理不完全。

问题四：

问题内容：明明明明白白白喜欢他，可她就是不。这句话里，明明和白白谁喜欢谁？

Qwen-7B-Chat：

- 提供了明确答案：“白白喜欢明明”，说明正确识别了句法结构。
- 没有详细解释推理过程，但结论准确。

ChatGLM3-6B：

- 回答是“明明喜欢白白”，说明识别主谓关系出错。

问题五：

问题内容：分析“意思”一词在不同语境下的含义。

Qwen-7B-Chat：

- 每一句的“意思”都进行解释,分析贴合语境

ChatGLM3-6B：

- 同样对每句话进行了逐句解释，表达清晰。

2.模型架构对比

2.1模型基础架构与设计理念

Qwen-7B-Chat 是阿里达摩院推出的对话语言模型，基于经典的 Decoder-only Transformer 架构，整体设计非常接近 GPT-4 系列，采用了 Rotary Position Embedding (RoPE) 来增强长距离建模能力，并在激活函数上使用了当前主流的 SwiGLU，使得模型在处理复杂指令时更加稳定高效。而 ChatGLM3-6B 是由智谱 AI 开发的第三代中文对话模型，它延续了其特有的 GLM 架构思路，不是传统的 GPT 风格 Transformer，而是结合了 BERT 和 GPT 优势的 auto-regressive masked 模型，在保持自回归生成能力的同时，采用了更具效率的注意力机制和相对位置编码。

2.2 对话格式与多轮交互机制

在多轮对话建模方式上，两者也有明显差异。从截图中可以看出，Qwen-7B-Chat 采用类 GPT 的 system / user / assistant 格式，使用 `<|im_start|>` 和 `<|im_end|>` 明确标记对话轮次与角色，从而适配 ChatGPT 风格的接口和上层交互逻辑。而 ChatGLM3-6B 则延续了自己的一套特殊 prompt 控制机制，例如 `[gMASK]`、`[sMASK]` 等 token，用以管理生成位置、上下文窗口以及对话状态，虽然形式更特化，但也具备更高的灵活性，特别是在嵌入式部署和资源受限场景下更为高效。

2.3 预训练语料与指令微调策略

从语料与训练策略来看，Qwen-7B-Chat 的预训练数据量超过 3 万亿 token，涵盖了中文网页、代码、英文百科、小说等多领域内容，明显侧重中英双语能力和代码生成能力；在训练目标方面，它结合了传统的自回归语言建模和对话微调过程，最终通过 RLHF 等机制优化对齐人类意图。而 ChatGLM3-6B 同样在中文语料上做了大量优化，虽然具体语料未完全公开，但在中英文混合任务中有较强适应性，且针对 API 应答、问答摘要等典型场景进行了大量指令微调，其生成风格更偏任务指令导向。

2.4 预训练语料与指令微调策略

在模型推理与部署方面，ChatGLM3-6B 的最大优势之一就是轻量高效，它原生支持 INT4 和 INT8 量化，并通过多 Query Attention 机制极大减少显存占用，能够在低于 6GB 显存的环境中本地运行，部署门槛更低。Qwen-7B-Chat 则专注于高性能和大模型能力的展现，支持 FlashAttention v2 和多种推理加速库，尽管对显卡有一定要求，但在长上下文建模和复杂推理任务中表现更优。此外，Qwen 也在逐步接入工具调用、API 接口化等能力，虽然目前还不如 ChatGLM3 在“插件调用”方面灵活，但整体开放程度高、兼容 HuggingFace 等生态系统，适合进行大规模定制开发。

五、总结

通过本次“大语言模型部署体验”作业，我对当前主流中文大语言模型——Qwen-7B-Chat 与 ChatGLM3-6B 的架构设计、对话机制、训练策略以及部署能力进行了系统性的对比与分析。在实际部署与交互过程中，我不仅观察到两者在任务理解和生成风格上的细微差异，也深刻体会到模型架构背后对于性能、效率与应用场景的权衡取舍。Qwen-7B-Chat 展现出更强的推理能力与上下文理解深度，适合复杂任务与研发场景；而 ChatGLM3-6B 则以其轻量化设计和本地部署优势，在日常问答、嵌入式交互等应用中表现出色。

这次体验不仅加深了我对 Transformer 系列语言模型内部机制的理解，也提升了我对于模型部署链路、推理优化和资源匹配的工程实践认知。在未来的学习和研究中，我将继续关注大模型的发展趋势，并尝试结合具体应用需求，选择合适的模型进行场景适配和性能调优，为实际项目提供更具性价比的智能方案。