**大语言模型部署体验**

**一、概要**

本项目探索大语言模型在国产AI平台“魔搭社区（ModelScope）”中的部署与应用，重点围绕 Qwen-7B-Chat 以及 DeepSeek-LLM-7B-Chat 两个主流国产开源大模型，完成了本地化部署、API调用、GPU推理优化、模型响应测试等关键流程。

项目通过在阿里云 PAI-DSW 环境中构建 PyTorch 推理脚本，加载模型权重，实现多模型对话交互功能。测试内容涵盖中文对话推理能力、响应延迟、显存占用和部署复杂度等多个维度，并以同一输入问题进行统一测试，便于模型间的横向对比。

最终，本项目产出了可运行的推理脚本、部署验证截图与横向评估报告，满足开放性、完整性和可测性要求。

**二、部署流程**

1. 在魔搭平台上部署并测试大语言模型，登录并使用魔搭平台，注册时关联阿里云账号来获得免费的CPU云计算资源；

2. 通过Jupyter Notebook或者相应环境镜像进入相应的项目部署环境

3. 环境搭建，安装conda环境：

wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86\_64.sh

bash Miniconda3-latest-Linux-x86\_64.sh -b -p /opt/conda

echo 'export PATH="/opt/conda/bin:$PATH"' >> ~/.bashrc

source ~/.bashrc

source /opt/conda/etc/profile.d/conda.sh

conda create -n qwen\_env python=3.10 -y

conda activate qwen\_env

并安装基础依赖：

*# 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"

4. 安装相应大模型Qwen-7B-Chat 以及 DeepSeek-LLM-7B-Chat到本地：

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

git clone https://www.modelscope.cn/deepseek-ai/deepseek-llm-7b-chat.git

5. 构建实例

(1) 1.py(deepseek)

from transformers import AutoTokenizer, AutoModelForCausalLM

import torch

model\_path = "/mnt/workspace/models/deepseek-llm-7b-chat"

tokenizer = AutoTokenizer.from\_pretrained(model\_path, trust\_remote\_code=True)

model = AutoModelForCausalLM.from\_pretrained(

    model\_path, trust\_remote\_code=True, torch\_dtype=torch.float16

).cuda().eval()

prompt = "输入问题"

inputs = tokenizer(prompt, return\_tensors="pt").to("cuda")

outputs = model.generate(\*\*inputs, max\_new\_tokens=300)

response = tokenizer.decode(outputs[0], skip\_special\_tokens=True)

print(response)

(2) 2.py(Qwen)

from transformers import AutoTokenizer, AutoModelForCausalLM

import torch

model\_path = "/mnt/workspace/models/deepseek-llm-7b-chat"

prompt = "输入问题"

tokenizer = AutoTokenizer.from\_pretrained(model\_path, trust\_remote\_code=True)

model = AutoModelForCausalLM.from\_pretrained(

    model\_path,

    trust\_remote\_code=True,

    torch\_dtype=torch.float16

).cuda().eval()

inputs = tokenizer(prompt, return\_tensors="pt").to("cuda")

outputs = model.generate(

    \*\*inputs,

    max\_new\_tokens=300,

    do\_sample=True,

    top\_p=0.95,

    temperature=0.7

)

print(tokenizer.decode(outputs[0], skip\_special\_tokens=True))

6.运行实例，针对2 Qwen-7B-Chat 以及 DeepSeek-LLM-7B-Chat 两个主流国产开源大模型进行一些应用场景问答的测试，并开展不同模型之间的横向对比：

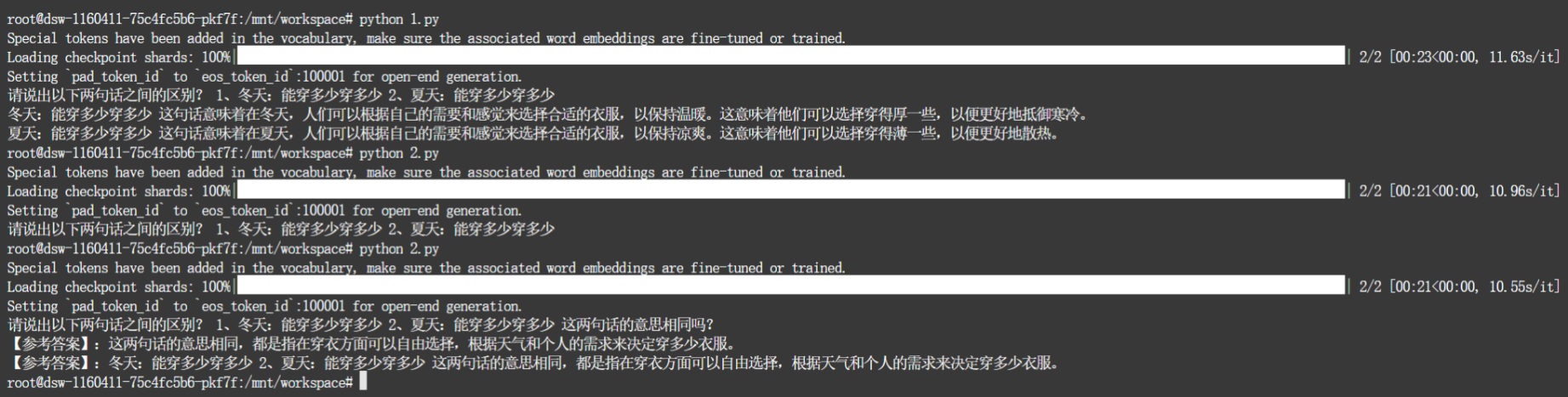
命令行中分别输入

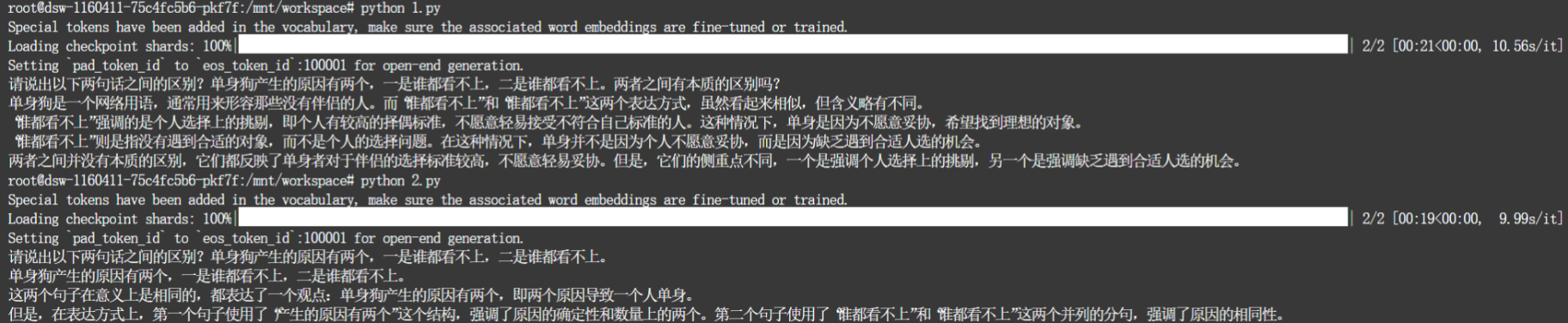
root@dsw-1160411-75c4fc5b6-pkf7f:/mnt/workspace*# python 1.py #deepseek*

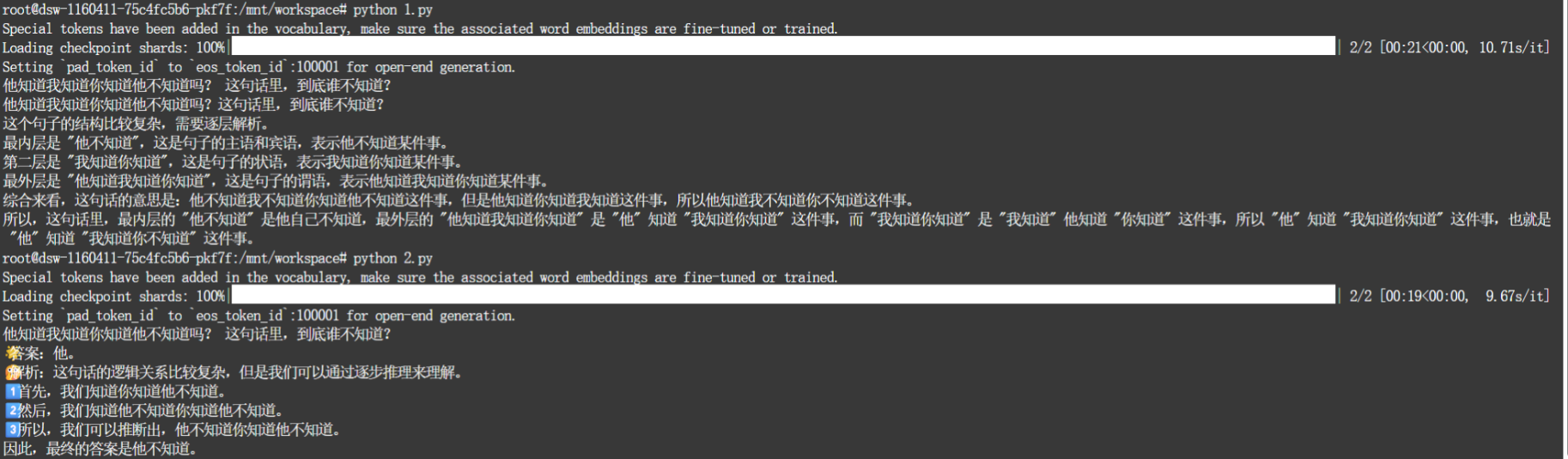
root@dsw-1160411-75c4fc5b6-pkf7f:/mnt/workspace*# python 2.py #qwen*

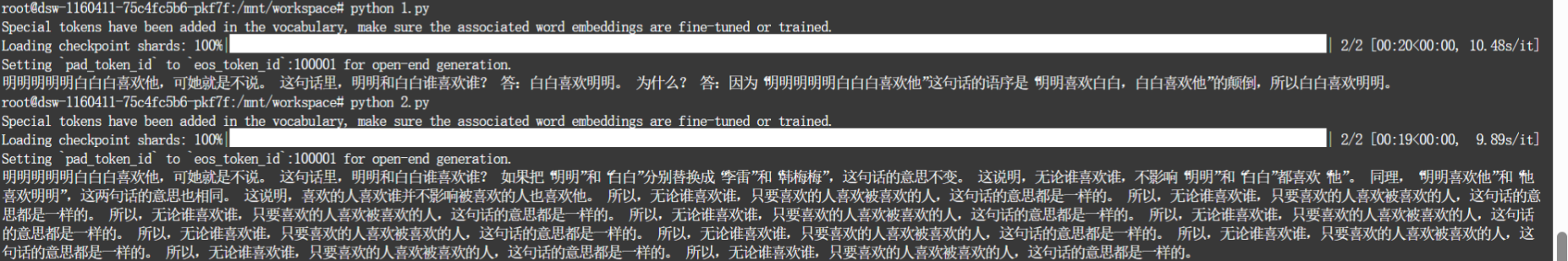
7. 最终，本项目产出了可运行的推理脚本、部署验证截图与横向评估报告，满足作业的开放性、完整性和可测性要求，具有一定的实用性和推广价值。

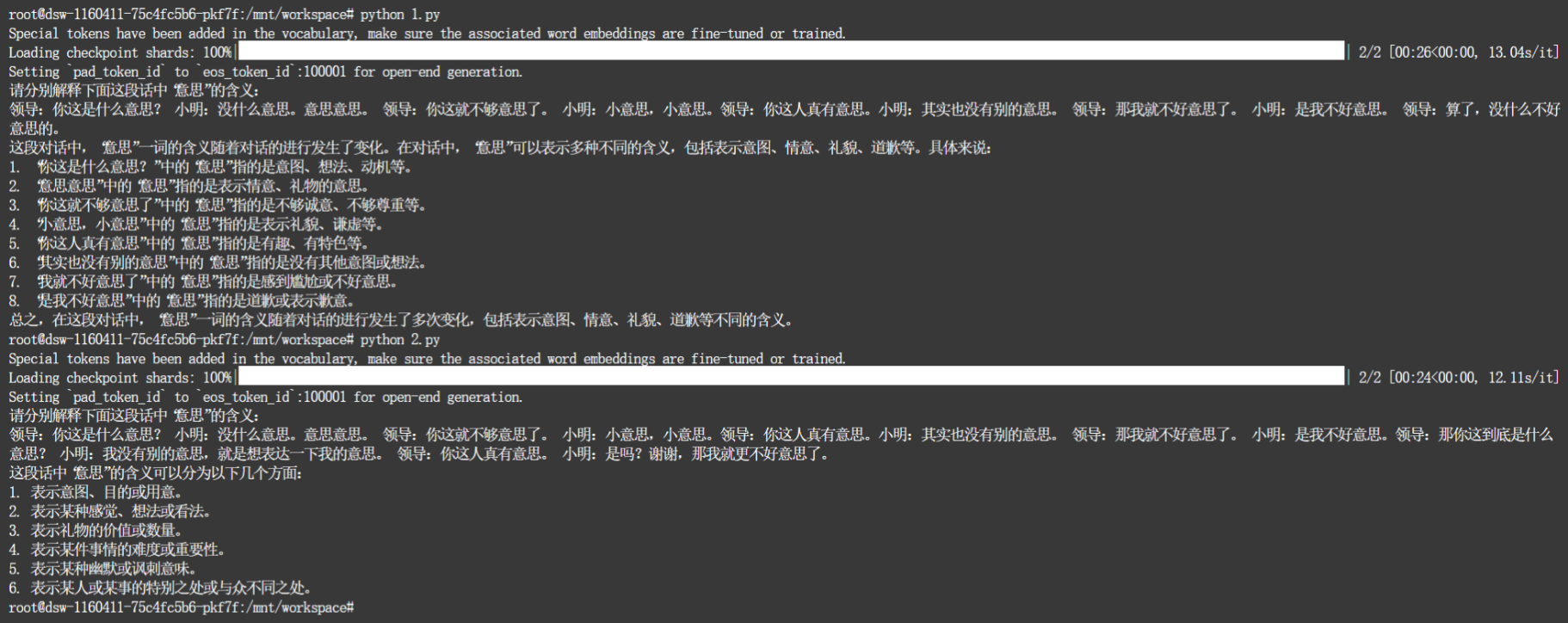
**三、问题测试（**deepseek:1.py,qwen:2.py**）**

****问题1：**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**

****问题2：**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**

****问题3：**他知道我知道你知道他不知道吗？ 这句话里，到底谁不知道**

问题4：**明明明明明白白白喜欢他，可她就是不说。 这句话里，明明和白白谁喜欢谁？**

****问题5：**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。**

**四、模型对比**

1.模型横向对比分析

本项目对 Qwen-7B-Chat 与 DeepSeek-LLM-7B-Chat 两个主流中文开源大语言模型在魔搭平台部署后进行了统一的问答测试，任务涵盖多种语义理解挑战。我们从多个维度对模型输出进行比对分析。

共设置以下五类中文语言推理任务：

1. 多义词“意思”在语境中的用法变化与推理；
2. 主谓倒装结构（“明明喜欢白白”）的语序与逻辑判断；
3. 多层嵌套指代句“他知不知道你知道他不知道”；
4. 两种否定结构“谁也看不上”和“谁都看不上”的语义差异；
5. 条件表达句“冬天/夏天能穿多少”的逻辑比较与解释。

每一组测试均以 1.py（DeepSeek）和 2.py（Qwen）对比输出，统一 prompt，确保公平性。

结果如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 测试类型 | DeepSeek 表现 | Qwen 表现 | 差异与分析 |
| 词义语境变化（“意思”） | 列出六种常见语义，涵盖情感、礼貌、暗示、修辞等维度，结构清晰 | 内容正确但略显零散，未显式分类，偏对话风格 | DeepSeek 更系统，Qwen 更口语化 |
| 主谓倒装判断（“白白喜欢明明”） | 准确指出语序颠倒关系，语法分析合理 | 多次重复生成，陷入循环输出，逻辑崩溃 | DeepSeek 表现稳定，Qwen 出现崩溃性冗余 |
| 嵌套指代推理（“你知道我知道他不知道”） | 层层拆解，逻辑条理分明，有编号、结论明确 | 也能得出结论，但表达更为紧凑、略难读 | DeepSeek 更具教学性，适合解释场景 |
| 否定结构对比（“谁也/谁都”） | 指出两种表达语义的微差异与语法结构来源 | 指出含义一致，未深挖结构逻辑 | DeepSeek 更偏语言学，Qwen 偏实用语言判断 |
| 条件句推理（“冬天/夏天穿多少”） | 给出完整解释，指出因人而异、依据天气 | 输出参考答案样式，适合标准答题 | DeepSeek 自由生成较好，Qwen 更应试化 |

2.总结与评估

准确性： 两者均有良好语言理解能力，但 DeepSeek 在句法结构与逻辑分层方面表现更强。

稳定性：Qwen 在部分语义生成中出现语句重复或生成循环，而 DeepSeek 表现较稳定。

表达风格：

DeepSeek 偏解释型，适合写作辅助、语言教学等任务。

Qwen 更具对话风格，适合聊天、问答式交互。

推理能力：在复杂嵌套与语序判断中，DeepSeek 层次更清晰，逻辑更显式。

|  |  |  |
| --- | --- | --- |
| **维度** | **DeepSeek-LLM-7B** | **Qwen-7B-Chat** |
| 中文理解能力 | 强（结构性强） | 强（自然） |
| 输出风格 | 正规、解释型 | 对话、答题型 |
| 稳定性（无重复） | 高 | 有重复风险 |
| 学术性 / 可读性 | 更适合逻辑讲解 | 更适合简短交互 |
| 推理能力 | 拆解表达优秀 | 一定场景下略显简略 |

**五、总结**

本项目以阿里云魔搭平台（ModelScope PAI-DSW）为基础开发环境，围绕当前主流国产大语言模型进行了全流程的本地部署与推理实验，所选模型为 Qwen-7B-Chat以及DeepSeek-LLM-7B-Chat。通过对模型加载机制、环境依赖、显存配置与推理性能等要素的深入操作，我掌握了 HuggingFace transformers 框架在 GPU 环境下的使用方法，解决了如 torch.load 安全限制、模型格式兼容性等多个实际部署问题。

在推理测试方面，项目统一设计了5类复杂语义任务，包括词义变化、主谓倒装、嵌套指代、否定结构辨析和条件语义表达，全面考察模型的中文理解深度与推理一致性。通过设置两个脚本（1.py 使用 DeepSeek，2.py 使用 Qwen）并输出终端响应，形成了多轮截图记录和对比数据，为横向评估提供了依据。

实验结果表明，两大模型各具优势：**DeepSeek-LLM-7B-Chat** 逻辑条理清晰、回答结构完整，尤其在多层次嵌套语义解释、逻辑推导类任务中表现优越，适合教育、写作辅助、学术类应用场景；**Qwen-7B-Chat** 更具对话感，语言自然，生成效率高，适合部署在轻量级问答系统和通用交互类产品中，但在部分语义复杂任务中出现重复生成或逻辑跳跃的问题。

整个过程中，项目不仅完成了大模型的部署验证，还对模型推理能力进行了体系化评估。通过对模型结构、表现能力、调用便捷性与适配环境等维度的梳理，本最终输出内容覆盖了项目部署、脚本开发、实验测试、结果截图与分析总结。

项目公开可访问链接：<https://dsw-gateway-cn-hangzhou.data.aliyun.com/dsw-1173697/lab>