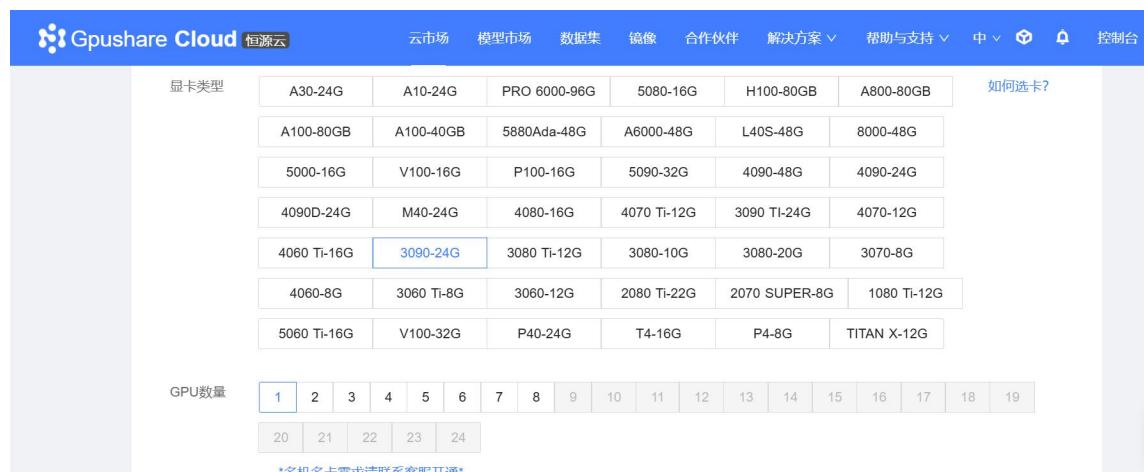


山东大学计算机科学与技术学院

大数据分析与实践实验报告

学号: 202300130214	姓名: 于博雅	班级: 23级数据班
实验题目: BERT环境配置		
实验学时: 2	实验日期: 2025. 11. 7	
实验目标: 创建大规模数据的处理平台实例，利用远程环境进行代码的调试		
实验步骤与结果: 选择云服务器平台恒源云，选择各项参数，创建实例:		
		
		
		

正在创建中：

The screenshot shows the Gpushare Cloud interface with a sidebar on the left containing '我的实例' (My Instances), '数据集' (Data Sets), '据集' (Dataset), '据集' (Dataset), '像' (Image), '心' (Heart), and '心' (Heart). The main area displays an instance creation form with fields for '地区/实例ID/名称' (Region/Instance ID/Name) set to '西南' (Sichuan) and 'Linux'. Other fields include '状态' (Status) '正常' (Normal), '实例存储' (Instance Storage) '3090-24G * 1 系统磁盘 正常', '操作模式' (Operation Mode) '按量付费', '机器到期时间' (Machine Expiry Time) '2026-10-13', and '登录指令' (Login Command) 'ssh***** 复制'. A red vertical bar on the right labeled '注册礼包' (Registration Gift) contains a gift icon and a sign-in icon.

最后创建完成如下图所示：

The screenshot shows the same instance creation interface, but the status has changed to '运行中' (Running) with a green circle icon. The instance details now include 'I2562e4f31701501068' and 'yby'. The '登录指令' (Login Command) field now shows 'ssh***** 复制' and '余额不足72小时' (Insufficient balance for 72 hours).

使用VSCode进行连接：

输入密码：（复制这一栏）



出现如下图界面即代表连接成功：

The screenshot shows a VSCode interface connected to 'root [SSH: i-1.gpushare.com]'. The left sidebar shows a file tree with files like '.cache', '.conda', '.config', '.dotnet', '.hycloud_ossutil_log', '.ipython', 'jupyter', '.local', '.nv', '.pip', '.ssh', '.vscode-server', '.bashrc', '.condarc', '.history_command', '.hycloud_ossutil_config', '.profile', '.wget-hsts', 'test_bert.py', and 'test_cuda.py'. The main editor pane displays Python code for sentiment analysis:

```
model_name = "uer/roberta-base-finetuned-jd-binary-chinese"
tokenizer = BertTokenizer.from_pretrained(model_name)
model = BertForSequenceClassification.from_pretrained(model_name)
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model.to(device)

def predict_sentiment(text):
    inputs = tokenizer(text, return_tensors='pt', padding=True, truncation=True,
                       inputs = {key: value.to(device) for key, value in inputs.items()}
    with torch.no_grad():
        outputs = model(**inputs)
    logits = outputs.logits
    probs = torch.softmax(logits, dim=-1)
    sentiment = torch.argmax(probs, dim=-1).item()
    return "积极" if sentiment == 1 else "消极", probs

text = "今天的天气真好，我感到非常开心！"
sentiment, probability = predict_sentiment(text)
print(f"文本: {text}")
```

The bottom status bar indicates the connection is 'Remote - SSH'.

接下来是环境的搭配：

因为在创建服务器时就已经装好了 PyTorch，所以运行以下测试代码： test1.py

```
test1.py > ...
1 import torch
2 # 检查CUDA是否可用
3 cuda_available = torch.cuda.is_available()
4 print(f"CUDA 是否可用: {cuda_available}")
5 if cuda_available:
6     # 检查当前可用的GPU数量
7     gpu_count = torch.cuda.device_count()
8     print(f"可用的 GPU 数量: {gpu_count}")
9     # 获取当前GPU名称
10    for i in range(gpu_count):
11        print(f"GPU {i} 名称: {torch.cuda.get_device_name(i)}")
12    # 检查当前设备
13    current_device = torch.cuda.current_device()
14    print(f"当前使用的设备索引: {current_device}")
15 else:
16     print("未检测到可用的 CUDA 设备")
```

得到如下结果：

```
(base) root@I2562e4f31701501068:~# /usr/local/miniconda3/bin/python /root/test1.py
CUDA 是否可用: True
可用的 GPU 数量: 1
GPU 0 名称: NVIDIA GeForce RTX 3090
当前使用的设备索引: 0
(base) root@I2562e4f31701501068:~#
```

得到如上图输出，说明 torch包 的 CUDA 和 GPU 都是可用的。

但想要运行 BERT，还需要安装transformers

在终端输入 pip install transformers 安装完成后运行以下测试代码： test2.py

```
test2.py > predict_sentiment
6 import os
7 os.environ['HF_ENDPOINT'] = 'https://hf-mirror.com'
8
9 from transformers import BertTokenizer, BertForSequenceClassification
10 import torch
11
12 model_name = "uer/roberta-base-finetuned-jd-binary-chinese"
13
14 tokenizer = BertTokenizer.from_pretrained(model_name)
15 model = BertForSequenceClassification.from_pretrained(model_name)
16 device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
17 model.to(device)
18
19 def predict_sentiment(text):
20     inputs = tokenizer(text, return_tensors='pt', padding=True, truncation=True, max_length=512)
21     inputs = {key: value.to(device) for key, value in inputs.items()}
22     with torch.no_grad():
23         outputs = model(**inputs)
24         logits = outputs.logits
25         probs = torch.softmax(logits, dim=-1)
26         sentiment = torch.argmax(probs, dim=-1).item()
27         return "积极" if sentiment == 1 else "消极", probs
28
29 text = "今天的天气真好，我感到非常开心！"
30 sentiment, probability = predict_sentiment(text)
31 print(f"文本: {text}")
32 print(f"预测的情感: {sentiment}")
33 print(f"概率: {probability}")
```

得到如下的结果：

```
(base) root@I2562e4f31701501068:~# /usr/local/miniconda3/bin/python /root/test2.py
文本: 今天的天气真好，我感到非常开心！
预测的情感: 积极
概率: tensor([[0.0114, 0.9886]], device='cuda:0')
```

通过以上操作，BERT 的基础环境已经配好

遇到的问题以及解决方法：

```
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

之前报错的关键原因之一是服务器无法访问 HuggingFace 国外源（huggingface.co），导致模型下载超时 / 失败，现在将 HuggingFace 的模型下载端点强制切换为国内镜像源（hf-mirror.com），替代国外的 huggingface.co 帮助服务器快速拉取模型文件。

结论分析与体会：

本次实验成功在恒源云搭建远程实例，经 VSCode 连接调试，配好 PyTorch+transformers 的 BERT 环境，验证了 GPU 可用。遇到的 HuggingFace 源问题，换国内镜像解决。云镜像省环境配置，远程工具适配高效，国内源是海外工具国内用的关键。