

大数据分析实践实验报告

实验四 BERT 环境配置

一、实验目标

配置 BERT 环境，并验证能否成功运行

二、实验环境

python3.8, Anaconda Prompt

三、实验过程

1. 打开 Anaconda Prompt，并创建虚拟环境（Python 3.8）

```
(base) C:\Users\边鑫磊>conda create -n bert_env python=3.8
Retrieving notices: ...working... done
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done
```

Downloading and Extracting Packages:

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

2. 激活虚拟环境

```
(base) C:\Users\边鑫磊>conda activate bert_env
```

3. 安装深度学习框架（这里选择 PyTorch）

```
(bert_env) C:\Users\边鑫磊>conda install pytorch torchvision torchaudio cpuonly -c pytorch
Channels:
- pytorch
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done
```

4. 安装 Hugging Face Transformers 库

```
(bert_env) C:\Users\边鑫磊>pip install transformers
Collecting transformers
  Downloading transformers-4.46.3-py3-none-any.whl.metadata (44 kB)
Requirement already satisfied: filelock in d:\zhenanaconda\envs\bert_env
Collecting huggingface-hub<1.0,>=0.23.2 (from transformers)
  Downloading huggingface_hub-0.36.0-py3-none-any.whl.metadata (14 kB)
Requirement already satisfied: numpy>=1.17 in d:\zhenanaconda\envs\bert_
3)

Installing collected packages: safetensors, regex, packaging, fsspec, colorama, tqdm, huggingface-hub, tokenizers, trans
formers
Successfully installed colorama-0.4.6 fsspec-2025.3.0 huggingface-hub-0.36.0 packaging-25.0 regex-2024.11.6 safetensors-
0.5.3 tokenizers-0.20.3 tqdm-4.67.1 transformers-4.46.3
```

5. 创建一个简单的 Python 脚本 (test_bert.py)，验证 BERT 模型能否正常加载和运行

```
1  from transformers import BertTokenizer, BertModel
2
3  # 加载预训练BERT模型和分词器 (bert-base-uncased为基础英文模型)
4  tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
5  model = BertModel.from_pretrained('bert-base-uncased')
6
7  # 测试文本
8  text = "Hello, this is a simple BERT test."
9
10 # 分词并转换为模型输入格式
11 inputs = tokenizer(text, return_tensors='pt') # 'pt'表示PyTorch张量 (若用TensorFlow则为'tf')
12
13 # 前向传播 (获取BERT输出)
14 outputs = model(**inputs)
15
16 # 输出结果形状 (验证模型运行正常)
17 print("BERT输出形状:", outputs.last_hidden_state.shape)
~~
```

```
(bert_env) C:\Users\边鑫磊>python test_bert.py
tokenizer_config.json: 100%|██████████| 48.0/48
D:\zhenanaconda\envs\bert_env\lib\site-packages\huggingface_hub\file_download.py:143: UserWarning:
he-system uses symlinks by default to efficiently store duplicated files but your machine doe
rs\边鑫磊\.cache\huggingface\hub\models--bert-base-uncased. Caching files will still work bu
t might require more space on your disk. This warning can be disabled by setting the 'HF_HUB_
nvironment variable. For more details, see https://huggingface.co/docs/huggingface_hub/how-to
To support symlinks on Windows, you either need to activate Developer Mode or to run Python a
r to activate developer mode, see this article: https://docs.microsoft.com/en-us/windows/app
vice-for-development
    warnings.warn(message)
vocab.txt: 232kB [00:01, 186kB/s]
tokenizer.json: 466kB [00:01, 287kB/s]
model.safetensors: 100%|██████████| 440M/440M
BERT输出形状: torch.Size([1, 11, 768])
```

可以看到，运行后输出了形状: torch.Size([1, 11, 768]) (这里 1 为批次大小，11 为分词后的 token 数量，768 为 BERT 隐藏层维度)，与预期一致，说明 BERT 环境搭建成功

四、实验总结

本次实验成功在 Anaconda 环境中搭建了基于 PyTorch 和 Hugging Face Transformers 库的 BERT 运行环境，通过创建独立虚拟环境避免了依赖冲突，并针对网络访问 Hugging Face 模型仓库超时的问题，采用设置国内镜像源的方法解决，最终运行测试脚本验证了 BERT 模型的正常加载与前向传播，输出了符合预期的模型隐藏层张量形状，为后续基于 BERT 的

自然语言处理任务（如文本分类、命名实体识别等）奠定了稳定的环境基础