基于预训练BERT模型的句子语义等价性判断实验报告

姓名:罗艺超 学号：202300130067 班级：数据班

摘要

本实验利用预训练的BERT-base-uncased模型，在MRPC（Microsoft Research Paraphrase Corpus）数据集上进行微调，实现了一个句子对语义等价性（复述）的二分类模型。实验成功配置了本地GPU训练环境，完整实现了数据加载、模型构建、训练与评估的全流程。经过3个训练周期，模型在验证集上达到了**84.31%**的准确率，与BERT-base在该任务上的基准性能相符。实验深入分析了训练过程中的过拟合现象，并通过具体样例探讨了模型的优势与局限，为理解与使用Transformer模型进行自然语言处理任务提供了实践参考。

---

1. 引言

句子语义等价性判断（Paraphrase Identification）是自然语言处理中的一项基础任务，旨在判断两个句子是否表达相同的语义。该任务在文本去重、问答系统、语义搜索等领域有广泛应用。

---

2. 实验环境与方法

2.1 实验环境

- **硬件**：NVIDIA GeForce RTX 4050 Laptop GPU (6GB显存)
- **核心软件**：
  - Python 3.10
  - PyTorch 2.5.1+cu121
  - Transformers 4.18.0
  - Datasets 库

2.2 数据集

使用**MRPC（Microsoft Research Paraphrase Corpus）** 数据集，该数据集从新闻源中提取句子对，并由人工标注是否构成同义（复述）。

- 训练集：3,668个句子对
- 验证集：408个句子对
- 标签：`1`表示同义，`0`表示不同义

2.3 模型架构

采用 `bert-base-uncased` 预训练模型作为特征提取器，其结构为12层Transformer编码器，隐藏层维度768。

1. **输入处理**：将两个句子拼接为 `[CLS] 句子A [SEP] 句子B [SEP]` 格式，经Tokenizer转换为对应的`input_ids`和`attention_mask`。
2. **特征提取**：将处理后的输入送入BERT模型，取最后一层`[CLS]`标记对应的768维向量作为整个句子对的语义表示。

3. **分类头**：将`[CLS]`向量输入一个自定义的全连接网络进行分类。该网络结构为：`Linear(768->256) + ReLU + Dropout(0.3) + Linear(256->1) + Sigmoid`。

2.4 训练配置

- **优化器**：AdamW
  - BERT参数学习率：2e-5
  - 分类头参数学习率：1e-3
- **损失函数**：二元交叉熵损失（BCELoss）
- **批大小**：16
- **训练周期**：3
- **学习率调度**：StepLR (每2个周期衰减为原来的0.1倍)

---

3. 实验过程与结果

3.1 训练过程动态

模型在3个训练周期内的性能变化如下表所示：

训练周期 (Epoch)	训练损失 (Loss)	训练准确率 (Acc)	验证损失 (Loss)	验证准确率 (Acc)	最佳模型
1	0.5320	72.57%	0.4734	81.86%	
2	0.3348	86.72%	0.4920	**84.31%**	✓ (保存)
3	**0.1473**	**95.07%**	0.4449	84.07%	

分析：

- **快速收敛**：第一个周期后，验证准确率即达到81.86%，表明预训练的BERT模型具有强大的特征迁移能力。
- **过拟合现象**：从第二个周期开始，训练损失持续快速下降，训练准确率急剧上升至95%，而验证准确率的提升却变得极其缓慢，并在第三周期出现轻微回落。同时，验证损失在第二周期后未继续下降。这明确表明模型在训练集上出现了**过拟合**。
- **最佳模型**：根据验证准确率，第二个周期训练结束后保存的模型为最佳模型，准确率为**84.31%**。

3.2 最终性能与样例测试

在保存的最佳模型（Epoch 2）上进行测试，结果如下：

句子 1	句子 2	真实标签	预测标签	置信度	判断
The cat sits on the mat	The cat is on the mat	1 (同义)	1 (同义)	0.9076	**正确**
I love programming	Coding is enjoyable	1 (同义)	0 (不同义)	0.9532	**错误**
The weather is nice	It's raining outside	0 (不同义)	0 (不同义)	0.9156	**正确**

样例分析：

1. **成功案例**：模型能有效处理句法层面的同义转换（“sits on” vs “is on”），并能识别明显的语义矛盾（“nice weather” vs “raining”）。
2. **错误案例**：对于“I love programming”和“Coding is enjoyable”，模型以高置信度判断为不同义。该错误表明模型虽然捕捉到了“love/enjoyable”的情感相似性，但可能未能充分理解“programming/coding”在此上下文中的强关联，或过于依赖严格的词汇匹配。

---

4. 实验总结与心得

1. **流程掌握**：本次实验成功完成了从环境配置、数据预处理、模型构建、训练调试到结果评估的完整深度学习项目流程，巩固了理论知识的工程实践能力。
2. **问题解决**：在实验中遇到了GPU驱动与PyTorch版本不兼容、网络问题导致模型下载失败等典型工程问题。通过查询资料、设置镜像源、调整代码，逐一解决了这些问题，提升了独立排错能力。
3. **理论联系实际**：通过观察训练与验证曲线，直观地理解了**过拟合**的动态表现及其成因。通过对预测样例的分析，认识到即使如BERT这样强大的模型，在细粒度的语义推理上仍有提升空间。
4. **结果验证**：最终模型在MRPC验证集上达到**84.31%**的准确率，与学术界报告的BERT-base在此任务上的性能范围（84%-88%）相符，证明了实验实现的有效性和正确性。

附录

A. 关键代码结构

```
bert_mrpc_experiment/
├── train.py                # 主训练脚本，包含训练循环、验证、保存逻辑
├── MRPCDataset.py          # 数据集加载类（使用HuggingFace datasets库）
├── FCModel.py              # 全连接分类头定义
├── models/                 # 保存的最佳模型（best_model_epoch2.pth）
└── data/                   # 缓存的数据集
```

B. 训练环境验证输出（片段）

```
D:\doubao\lab5\.venv\Scripts\python.exe D:\doubao\lab5\train1.py
使用设备: cuda
GPU信息: NVIDIA GeForce RTX 4050 Laptop GPU
CUDA版本: 12.1
GPU内存: 6.00 GB
创建目录: 'models', 'data'
```

```
Epoch 1 完成:
  时间: 44.57秒
  训练 Loss: 0.5320, Acc: 0.7257
  验证 Loss: 0.4734, Acc: 0.8186
```

```
Epoch 2 完成:
  时间: 43.20秒
  训练 Loss: 0.3348, Acc: 0.8672
  验证 Loss: 0.4920, Acc: 0.8431
```

```
Epoch 3 完成:
  时间: 43.58秒
  训练 Loss: 0.1473, Acc: 0.9507
  验证 Loss: 0.4449, Acc: 0.8407
```