山东大学 计算机科学与技术 学院

大数据分析实践 课程实验报告

学号：202300130092	姓名：王俊磊	班级： 23 数据
实验题目： 利用预训练的 BERT 模型对 MRPC 数据集进行语义相似性判别（同义预测）实验		
实验学时：2	实验日期：2025.11.12	
实验目的：对动手实践利用机器学习方法分析大规模数据有进一步了解，并学习如何利用远程环境进行工程代码的调试.		
软件环境： PyCharm,PyTorch 2,HuggingFace Transformers,GPU cuda 13.0		
实验步骤与内容： 1.数据准备：自定义类用于数据加载 2.文本处理：自动添加 [CLS] sentence1 [SEP] sentence2 [SEP]		

```python
class MRPCDataset(Dataset):  2 用法
    def __init__(self, path="dataset/msr_paraphrase_train.txt"):
        self.data = []
        with open(path, encoding="utf-8") as f:
            reader = csv.reader(f, delimiter='\t')
            next(reader)
            for row in reader:
                if len(row) < 5:
                    continue
                label = int(row[0])
                s1 = row[3]
                s2 = row[4]
                self.data.append((s1, s2, label))
```

```
sentence1, sentence2 = sentence
encoding = tokenizer(
    sentence1,
    sentence2,
    return_tensors='pt',
    padding=True,
    truncation=True
).to(device)
```

3.预训练 BERT 模型：

加载

```
bert_model = BertModel.from_pretrained("bert-base-uncased").to(device)
```

使用 pooler_output 作为句子对的语义表示

```
pooled = bert_output.pooler_output
```

4. 构建分类器：定义两层全连接神经网络（MLP）完成二分类

```
self.classifier = nn.Sequential(
    nn.Linear( in_features: 768,  out_features: 256),
    nn.ReLU(),
    nn.Dropout(0.3),
    nn.Linear( in_features: 256,  out_features: 1)
)
```

5. 模型训练

```
loss = crit(predict, label) loss.backward()  optimizer.step()
```

```
bert_optimizer.step() scheduler.step()
```

损失函数使用

```
crit = torch.nn.BCEWithLogitsLoss()
```

概率和准确度

```
prob = torch.sigmoid(predict)
acc = binary_accuracy(prob, label)
```

6 训练日志节选：

```
batch 47 loss=0.6728 acc=0.5625
batch 48 loss=0.5928 acc=0.8125
batch 49 loss=0.6811 acc=0.6875
batch 50 loss=0.6936 acc=0.6250
batch 51 loss=0.4892 acc=0.7500
batch 52 loss=0.4326 acc=0.8125
batch 53 loss=0.8783 acc=0.5000
batch 54 loss=0.5896 acc=0.7500
```

结论分析与体会：

实验成功完成了利用预训练 BERT 模型对 MRPC 数据集进行语义相似性判别任务，实现了完整的 NLP fine-tuning pipeline。

从训练日志可观察到，模型初始准确率波动大，但随着训练推进逐渐稳定并提升，说明微调策略有效。

BERT Tokenizer 自动完成 [CLS]、[SEP]、padding、分段编码等操作，使得句子对任务的数据处理大幅简化。

通过对 BERT 与自定义全连接分类层的联合训练，完成了句子级 representation 到语义相似度判别的映射。