多模态情感分析实验报告

10203330409 王越月

实验代码见：https://github.com/Eunhapada/exp5.git

一、实验目的

本次实验通过构建多模态融合模型，通过对给定图片及其描述、情感描述进行训练，并对测试集给出的无标签的图片代码进行情感倾向的预测。此外，希望在消融实验结果，也即分别只输入文本或图片训练模型时得到较好的预测结果。

二、实验环境

实验在 jupyter notebook 中完成，在 pytorch 环境下使用 CPU 运行程序（GPU 运行时本机超出内存范围）。

三、实验过程及分析

（一）模型设计

本次实验实现了一个基于 Bert 和 ResNet50 两个预训练模型的多模态融合模型。选择这两个模型的原因是因为，Bert 和 ResNet50 都是经过大规模预训练的模型，在大型数据集上进行了无监督学习，学习到了丰富的语义和视觉特征。通过在这些预训练模型的基础上进行参数微调和特征提取，可以充分利用模型的学习能力，提高模型的泛化能力。

ResNet50 是一种深度卷积神经网络，具有出色的图像处理能力。通过使用 ResNet50 模型，我们可以从输入的图像中提取高维的视觉特征，这些特征可以捕捉到图像中的边缘、纹理、形状等信息。

Bert 模型使用了之前下载过的“bert-based-uncased”，由于训练数据为英文文本，而这一模型是基于英文语料库得到的预训练模型，因而在处理文本数据时具有较强的建模能力和语义理解能力。通过使用 Bert 模型，我们可以对输入的文本进行编码和表示学习，从而捕捉到更丰富的文本特征。

通过编写两种模型分别的类函数，以及融合模型的类函数，实现对不同模态特征的融合，从而提高模型的性能和表现。

（二）实验过程

1. 数据集

本次实验使用图片和文本混合的数据，其中 train.txt 包含了图片的编号（guid）和情感倾向（label）的对应关系，作为所有模型的 label 储存；data 文件夹内包括 guid 对应的图片（.jpg）和文本描述(.txt)。

2. 数据预处理

（1） 图片处理

读入图片后转成 256*256 的大小并转为 tensor 数据类型，以便后续生成 Dataloader。调整图片尺寸是因为分类任务转成 256*256 大小是因为输出特征图如果尺寸太小，那么信息就丢失太严重，如果尺寸太大，信息的抽象层次不够高，计算量也更大。查阅到有人测试过不同尺寸投入模型时的测试结果，发现 resize 到这一尺寸时在 resnet 上表现最好。get_ImagePath() 函数用于获取图像文件的路径列表。

（2） 文本处理

get_txt 函数用于获取名为 guid 的文本文件，并返回文本内容列表。此外定义了一个继承自 torch.utils.data.Dataset 的新类 Dataset，返回图像数据、input_ids、attention_mask 和 label 值。

（3） 划分验证集

将图像、文本、label 三类数据按 20%的比例划分验证集，随机种子设置为学号 9.

3. 模型训练

（1）搭建多模态模型

定义了三个模型类：ImgModel、TxtModel 和 FusionModel。

ImgModel 类是图像特征提取模型，它使用预训练的 ResNet-50 作为图像特征提取器。在初始化函数中，通过 resnet50(pretrained=True)加载了预训练的 ResNet-50 模型。forward 方法将传入的图像数据作为输入，通过 ResNet-50 模型提取图像特征并返回。

TxtModel 类是文本特征提取模型，它应该使用了一个名为 bert_model 的预训练 BERT 模型。在初始化函数中，通过 bert_model 实例化了一个 BERT 模型 forward 方法接收 input_ids 和 attention_mask 作为输入，将其传入 BERT 模型，并返回 BERT 模型的输出。

FusionModel 类是多模态融合模型，它包含了一个图像特征提取模型和一个文本特征提取模型，以及一个用于融合特征并进行分类的线性层。在初始化函数中，创建了一个 ImgModel 实例和一个 TxtModel 实例。根据给定的选择，forward 方法会选择使用图像模型、文本模型或拼接两类特征，然后将经过特征提取和融合的结果传入线性分类器，并返回输出。

（2）定义训练和预测函数

train_model 函数用于训练模型，函数内会将模型设置为训练模式，然后开始迭代训练数据加载器。对于每个批次，它会将图像数据、文本数据、注意力掩码和标签数据移动到指定的设备上。然后，根据模型的输入进行前向传播，并计算预测结果和损失值。接下来，它会计算准确率，并进行反向传播和参数更新。在迭代结束后，计算并返回平均损失和准确率。

predict_model 函数用于对模型进行预测，它将模型设置为评估模式，然后开始迭代测试数据加载器。对于每个批次，它将图像数据、文本数据和注意力掩码移动到指定的设备上，并通过模型进行前向传播以生成预测结果。最后，它将预测结果转移到 CPU 上并返回预测列表。

训练过程中模型会根据损失值进行反向传播和参数更新，而预测过程中模型不会进行梯度计算和参数更新。

4. 超参数调试

本次实验主要调整学习率，分别测试了两个直结果如下：

```
batch size: 64, lr: 1e-06, Epoch 1/10, Train Loss: 1.1073, Train Acc: 0.4188, Val Acc: 0.5450, Best Val Acc:0.5450
batch size: 64, lr: 1e-06, Epoch 2/10, Train Loss: 1.0369, Train Acc: 0.5028, Val Acc: 0.6025, Best Val Acc:0.6025
batch size: 64, lr: 1e-06, Epoch 3/10, Train Loss: 1.0118, Train Acc: 0.5325, Val Acc: 0.6000, Best Val Acc:0.6025
batch size: 64, lr: 1e-06, Epoch 4/10, Train Loss: 0.9624, Train Acc: 0.5722, Val Acc: 0.5962, Best Val Acc:0.6025
batch size: 64, lr: 1e-06, Epoch 5/10, Train Loss: 0.9693, Train Acc: 0.5719, Val Acc: 0.5975, Best Val Acc:0.6025
```

```
batch size: 64, lr: 0.001, Epoch 1/10, Train Loss: 1.1333, Train Acc: 0.5894, Val Acc: 0.5962, Best Val Acc:0.5962
batch size: 64, lr: 0.001, Epoch 2/10, Train Loss: 1.0986, Train Acc: 0.5972, Val Acc: 0.5962, Best Val Acc:0.5962
batch size: 64, lr: 0.001, Epoch 3/10, Train Loss: 1.0986, Train Acc: 0.5972, Val Acc: 0.5962, Best Val Acc:0.5962
batch size: 64, lr: 0.001, Epoch 4/10, Train Loss: 1.0986, Train Acc: 0.5972, Val Acc: 0.5962, Best Val Acc:0.5962
batch size: 64, lr: 0.001, Epoch 5/10, Train Loss: 1.0986, Train Acc: 0.5972, Val Acc: 0.5962, Best Val Acc:0.5962
```

四、遇到的问题及解决方式

1. Given groups=1, weight of size [64, 3, 7, 7], expected input[100, 224, 224, 3] to have 3 channels, but got 224 channels instead

resnet 卷积层 nn.conv（input_size）通道数为 3，需要输入数据维度为【N,C,H,W】中 C=3，而初始数据处理得到 train_loader 维度为 3200*224*224*3，通道数维度错误。

解决方案为在预处理时将图片数据 reshape（3,224,224）即可解决。

2. expected scalar type Long but found Int

问题出现在 nn.CrossEntropyLoss()里，其接受的 label 类型为 torch.LongTensor，因此在放入损失函数前需要类型转换 labels.type(torch.LongTensor)。