《人工智能导论》大作业

任务名称：暴力图像检测

完成组号：2

组员：石梓成、郑凯文、陈向晖、黄宇星

完成时间：2024.6.19

1．任务目标

基于暴力图像检测数据集，构建一个检测模型。该模型可以对数据集的图像进行不良内容检测与识别。

2．具体内容

（1）实施方案

通过使用 resent18 模型进行图像识别，训练测试集采用提供的测试集以及使用 AI 生成的部分图片进行测试，同时取测试集中部分图片，利用 opencv 添加高斯噪声后作为对抗训练集。最后使用训练的模型对与训练集同源的图片进行预测，训练集包括各种图片，最终得到准确率结果。

（2）核心代码分析

在此列出对参考代码做出的修改以及修改的分析。

a)对参考代码 dataset.py 所做的修改

```python
class CustomDataset(Dataset):
    def __init__(self, split):
        assert split in ["train", "val", "test"]
        data_root = "D:\\pytorch\\violence_224\\"
        self.data = [os.path.join(data_root, split, i) for i in os.listdir(data_root + split)]
        if split == "train":
            self.transforms = transforms.Compose([
                transforms.RandomHorizontalFlip(),  # 随机翻转
                transforms.RandomRotation(10),
                transforms.ColorJitter(brightness=0.2, contrast=0.2, saturation=0.2, hue=0.2),
                transforms.RandomResizedCrop(size=224, scale=(0.8, 1.0)),
                transforms.ToTensor(),
                transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
            ])
        else:
            self.transforms = transforms.Compose([
                transforms.Resize((224, 224)),
                transforms.ToTensor(),
                transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
            ])
```

使用数据增强来优化模型，包括随机水平翻转、随机旋转图像、随机改变图像的亮度、对比度、饱和度和色调、随机裁剪图像并

调整到指定大小。

b)对参考代码 model.py 所做的修改

```python
def fgsm_attack(model, loss_fn, images, labels, epsilon):
    images.requires_grad = True
    outputs = model(images)
    loss = loss_fn(outputs, labels)
    model.zero_grad()
    loss.backward()
    data_grad = images.grad.data
    sign_data_grad = data_grad.sign()
    perturbed_image = images + epsilon * sign_data_grad
    perturbed_image = torch.clamp(perturbed_image, min: 0, max: 1)
    return perturbed_image
6 usages
class ViolenceClassifier(LightningModule):
    def __init__(self, num_classes=2, learning_rate=1e-3, dropout_rate=0.5, weight_decay=1e-4,epsilon=0.1):
        super().__init__()
        self.model = models.resnet18(pretrained=True)
        num_ftrs = self.model.fc.in_features
        self.model.fc = nn.Linear(num_ftrs, num_classes)
        self.learning_rate = learning_rate
        self.loss_fn = nn.CrossEntropyLoss()
        self.accuracy = Accuracy(task="multiclass", num_classes=2)
        self.dropout = nn.Dropout(p=dropout_rate)
        self.weight_decay = weight_decay
        self.epsilon = epsilon
```

super().__init__()

调用父类 LightningModule 的初始化方法。

self.model = models.resnet18(pretrained=True)

加载预训练的 ResNet-18 模型。

num_ftrs = self.model.fc.in_features

self.model.fc = nn.Linear(num_ftrs, num_classes)

获取 ResNet-18 模型最后一层全连接层的输入特征数，并将其替换为一个新的线性层，使其输出的类别数与 num_classes 一致（二分类认为中默认为 2）

self.learning_rate = learning_rate

self.loss_fn = nn.CrossEntropyLoss()

self.accuracy = Accuracy(task="multiclass", num_classes=2)

self.dropout = nn.Dropout(p=dropout_rate)

self.weight_decay = weight_decay

Self.epsilon = epsilon

learning_rate：学习率，控制优化器更新权重的步长。

loss_fn：损失函数，这里使用交叉熵损失函数。

accuracy：精度评估器，用于计算多类分类任务的准确度。

dropout：Dropout 层，用于在训练期间随机丢弃一些神经元以防止过拟合。

weight_decay：权重衰减系数，用于 L2 正则化，防止过拟合。

Epsilon：用于控制对输入数据的扰动程度

fgsm_attack 为加入的对抗训练（可能是由于电脑性能有限，加入对抗训练后训练过程很慢，预计超过 12 小时，所以最终没有将其加入模型，加入后对加噪声的图像预测成功率应该会大大提升）

c)接口类 classify.py

```python
import torch
from model import ViolenceClassifier
import torchvision.transforms as transforms
from PIL import Image
import os

1 usage
class ViolenceClass:
    def __init__(self, checkpoint_path: str, device: str = 'cuda:0'):
        """
        初始化接口类，加载模型
        :param checkpoint_path: str, 训练好的模型权重文件的路径
        :param device: str, 指定设备，默认为'cuda:0'
        """
        self.device = device
        self.model = ViolenceClassifier.load_from_checkpoint(checkpoint_path)
        self.model.to(self.device)
        self.model.eval()
        self.transform = transforms.Compose([
            transforms.Resize((224, 224)),
            transforms.ToTensor(),
            transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
        ])

    1 usage
    def classify(self, imgs: torch.Tensor) -> list:
        """
        对输入的图像进行分类
        :param imgs: torch.Tensor, 形状为`n*3*224*224`的tensor，输入图像已经归一化到0-1
        :return: list, 长度为`n`的python列表，每个值为对应的预测类别（0或1）
        """
        imgs = imgs.to(self.device)
        with torch.no_grad():
            logits = self.model(imgs)
            probs = torch.nn.functional.softmax(logits, dim=1)
            preds = torch.argmax(probs, dim=1).cpu().tolist()
        return preds
```

（1）导入必要的库

（2）__init__ 方法：初始化类实例，加载训练好的模型权重文件并将模型设置为评估模式。

checkpoint_path：训练好的模型权重文件的路径。

device：指定设备（默认为 'cuda:0'）。

self.transform：定义图像预处理变换，包括调整大小、转换为 tensor 和归一化。

（3）classify 方法：对输入的图像进行分类。

imgs：输入的图像 tensor，形状为 n*3*224*224。

imgs.to(self.device)：将图像 tensor 移动到指定设备。

with torch.no_grad()：在不计算梯度的上下文中进行推理。

logits：模型输出的未归一化的得分。

probs：对得分进行 softmax 转换得到概率。

preds：通过 argmax 得到预测的类别索引，并转换为列表返回。

```
if __name__ == "__main__":
    import os
    from PIL import Image

    # 加载并预处理图像
    img_dir = "D:\\pytorch\\violence_224\\test"
    img_files = [os.path.join(img_dir, f) for f in os.listdir(img_dir) if f.endswith('.jpg') or f.endswith('.png')]

    imgs = []
    transformer = transforms.Compose([
        transforms.Resize((224, 224)),
        transforms.ToTensor(),
        transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
    ])

    for img_file in img_files:
        img = Image.open(img_file).convert("RGB")
        img_tensor = transformer(img)
        imgs.append(img_tensor)

    imgs_tensor = torch.stack(imgs)  # 形状为 n*3*224*224 的tensor

    # 实例化ViolenceClass并进行分类
    checkpoint_path = "train_logs/resnet18_pretrain_test/version_21/checkpoints/resnet18_pretrain_test-epoch=38-val_loss=0.06.ckpt"
    violence_classifier = ViolenceClass(checkpoint_path, device='cuda:0')
    predictions = violence_classifier.classify(imgs_tensor)

    # 打印预测结果
    i = 0
    for img_file, pred in zip(img_files, predictions):
        lable = int(img_file[29])
        print(f"Image: {img_file}, Predicted class: {pred}")
        if(lable!=pred): i = i + 1

    print(i)
```

主函数：加载图像并进行预处理。

img_dir：测试图像目录。

img_files：测试图像文件列表。

imgs：存储预处理后的图像 tensor 列表。

transformer：定义图像预处理变换。

将图像加载为 RGB 模式，应用预处理变换，并添加到 imgs 列

表。

torch.stack(imgs)：将列表中的 tensor 堆叠为一个 batch tensor。

实例化 ViolenceClass 并加载模型权重文件，调用 classify 方法对图像进行分类，并得到预测结果。

最后输出预测结果，统计正确率。

预测正确率：

```
Image: D:\pytorch\violence_224\val\1_0576.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0577.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0578.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0579.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0580.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0581.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0582.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\val\1_0583.jpg, Predicted class: 1
0.974706413730804

Process finished with exit code 0
```

对与训练集同源的数据的正确率约为 0.97

```
Image: D:\pytorch\violence_224\test\1_006.png, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_007.png, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_008.png, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_009.png, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_010.png, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_011.png, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_012.png, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_013.png, Predicted class: 0
0.726027397260274

Process finished with exit code 0
```

对 AIGC 生成图像的正确率约为 0.73

```
Image: D:\pytorch\violence_224\test\1_0576.jpg, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_0577.jpg, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_0578.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_0579.jpg, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_0580.jpg, Predicted class: 1
Image: D:\pytorch\violence_224\test\1_0581.jpg, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_0582.jpg, Predicted class: 0
Image: D:\pytorch\violence_224\test\1_0583.jpg, Predicted class: 0
0.5700090334236676

Process finished with exit code 0
```

对加噪声（标准差为 9）的测试正确率约为 0.57

3．工作总结

（1）收获、心得

通过这次课程项目，我们学会了如何选择一个人工智能模型来进行图像的识别与分类，并且通过使用不同的训练集来增强该模型预测的正确性，在人工智能实践方面有了巨大的收获；同时，利用人工智能模型进行暴力图像分类本身是一个非常具有应用场景的项目，能够协助网络环境的管理和整治，具有非凡的实用价值，能够通过课程的学习去应用所学知识，贡献于当代社会，令小组成员都十分振奋，所学有所用，实践出真知。

（2）遇到问题及解决思路

问题 1：报错 OSError: [WinError 1455] 页面文件太小,无法完成操作。

解决：原因是系统自动为其分配的分页文件大小不够，手动设置更大即可。

问题 2：参考代码中的 y = int(img_path.split("/")[-1][0])无法正

确读取标签位

解决：修改为 file_name = os.path.basename(img_path)

y = int(file_name[0])

4．课程建议

人工智能导论课程拓宽了我们的视野，向我们战士了人工智能这一新兴领域的部分内容。尽管这只是冰山一角，却也给我们带来了巨大的启发。我们希望课程以后能更深入地带领同学们了解人工智能，多介绍一些新兴的先进的人工智能技术，同时也讲解人工智能在生活中的运用，既与科学前沿接轨，也关注现实生活的方方面面。