2024《人工智能导论大作业》

任务名称：不良内容图像检测

完成组号：9

小组成员：朱施政、朱舟政、赵哲浩、王星淳

完成时间：2024.6.9

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。要求：

- 模型是2分类（0代表正常图像、1代表不良图像），分类准确率越高越好；
- 模型具有一定的泛化能力：不仅能够识别与训练集分布类似的图像，对于AIGC风格变化、图像噪声、对抗样本等具有一定的鲁棒性；
- 运行时间合理。

2.具体内容：

（1）实施方案：

数据集生成：

1. test1数据集：
   从文件夹violence/train中随机选取1000张图片作为test1
   
2. test2数据集：
   AIGC生成test2

- 正样本：

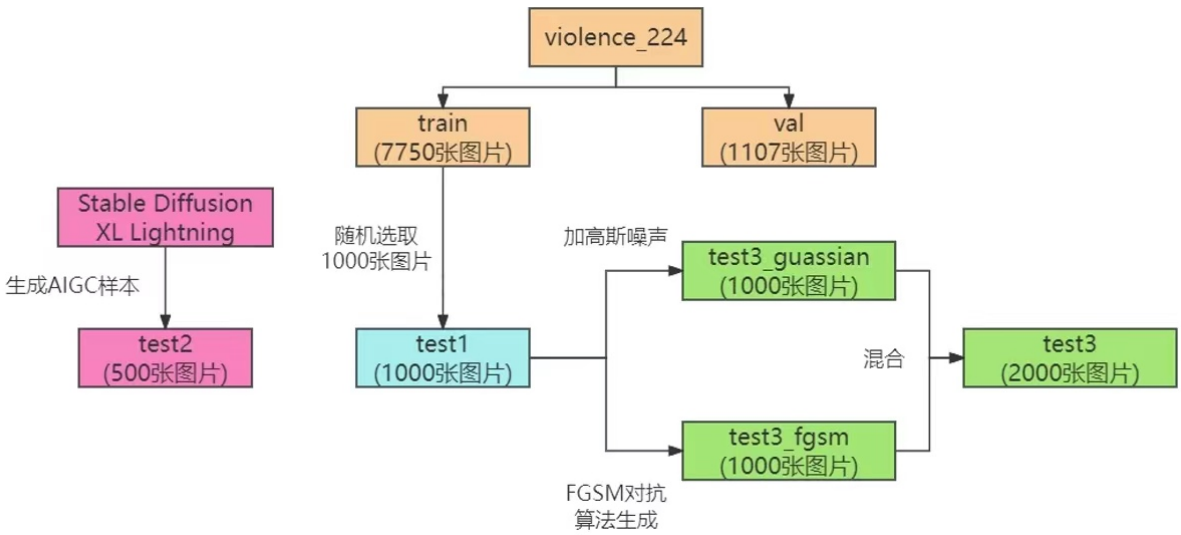
- 负样本：

3. test3数据集：
   test1中选取一张图片经过对抗算法加噪生成test3_fgsm

再在test1中选取另一张图片经过高斯算法加噪生成test3_gaussian，

对已经过对抗算法加噪的test3_fgsm再使用高斯算法加噪生成test3

数据集生成图：



模型结构：

分别使用用resnet18和resnet34（参数均已预训练）

训练测试过程：

使用pytorch中lightning框架训练测试，将训练得到模型和对应的checkpoint进行保存，在测试应用中应用

（2）核心代码分析：

主要代码包括classify.py、dataset.py、fgsm_test.py、get_test2_noise.py、model.py、utils.py等文件，它们的基本框架和作用分别如下所述：

1. classify.py:

```
# 对输入的图像张量进行分类预测，返回预测结果的类别索引列表。
class ViolenceClass:
    # 核心分类函数:参数`imgs`（已处理）
    def classify(self, imgs : torch.Tensor) -> list:
        imgs = imgs.to(self.device)
        # 测试模式（无需更新参数）：调用模型对输入图像进行处理，用torch.max()函数找出每个输出行最大值及其索引。
        with torch.no_grad():
            outputs = self.model(imgs)
            _, preds = torch.max(outputs, 1)
        return preds.cpu().tolist()#输出结果移动到CPU，转换为列表返回。
```

2. dataset.py:

```
#读取数据集图片进行加载和预处理，返回处理后的图片及其标签
class CustomDataset(Dataset):
    def __getitem__(self, index):
        img_path = self.data[index]
        x = Image.open(img_path)
        # 根据图片路径获取标签值，0代表非暴力，1代表暴力
        y = int(img_path.split("/")[-1][0])
        x = self.transforms(x)  # 对x进行预处理
        return x, y

#管理多个数据集的加载和数据加载器的组织
class CustomDataModule(LightningDataModule):
    # 改写test_dataloader,使之一次能对于三个test集进行测试
    def test_dataloader(self):
        return [
            DataLoader(self.test_dataset_1, batch_size=self.batch_size, shuffle=False, num_workers=self.num_workers),
            DataLoader(self.test_dataset_2, batch_size=self.batch_size, shuffle=False, num_workers=self.num_workers),
            DataLoader(self.test_dataset_3, batch_size=self.batch_size, shuffle=False, num_workers=self.num_workers)
        ]
```

3. fgsm_test.py:

```
 # 采用快速梯度符号法生成对抗样本
 # grad : 模型对输入图像的损失函数相对于输入图像的梯度
def FGSM_attack(imgs,grad,epsilon):
    #根据原始图像、梯度的符号和扰动的大小来生成对抗样本。
    attack_imgs = imgs + epsilon* grad.sign()
    #裁剪生成样本使之像素值大小在[0,1]之内
    attack_imgs = torch.clamp(attack_imgs, 0, 1)
    return attack_imgs

# FGSM对抗攻击，测试模型在对抗样本上的损失和准确率
def FGSM_test(net,eval_loader,device,criterion, output_folder_path, ep = 0.2):
        adv_y = net(adv_x)              # 对抗样本的前向传播和损失计算
        loss= criterion(adv_y,target)  # 计算模型在对抗样本上的损失
        test_loss+=loss.item()          # 累计测试损失
        _,adv_predicted = adv_y.max(1) # 计算并累计预测正确的样本数量
        correct += torch.eq(target,adv_predicted).float().sum().item()
        total+=target.size(0)
```

4. get_test2_noise.py:

```
# 两种方法添加高斯噪声：对所有像素添加噪声或者随机选取部分像素添加
# 函数参数：输入图像，高斯噪声的均值、标准差和数量
def gaussian_noise(self, img, mean=0, std_dev=15, noise_num=None):
    if noise_num is None:
        # 添加噪声到所有像素
        noise = np.random.normal(mean, std_dev, img.shape)
        noisy_img = np.clip(img + noise, 0, 255).astype(np.uint8)
    else:
        # 添加噪声到部分像素
        sz = img.shape
        noise_img = np.zeros_like(img, dtype=np.float32)
        # 随机选择 noise_num 个像素的位置
        indices = np.random.choice(sz[0] * sz[1], noise_num, replace=False)
        for i in indices:
            x, y = np.unravel_index(i, sz[:2])
            # 将一维索引i转换为二维索引(x, y)
            noise_val = np.random.normal(mean, std_dev)
            # 将噪声值添加到噪声图像的对应位置
            noise_img[x, y] = noise_val
        noisy_img = np.clip(img + noise_img, 0, 255).astype(np.uint8)
        return noisy_img
```

5. model.py:

```
#构建和训练一个基于 ResNet-18 架构的暴力分类器模型结构，res34模型同理
class ViolenceClassifier_res18(LightningModule):
```

```
    def __init__(self, num_classes=2, learning_rate=1e-3,version=0):
        super().__init__()
         # 加载预训练模型
        self.model = models.resnet18(pretrained=True)
         # 更改分类头,满足分类问题
        num_ftrs = self.model.fc.in_features
         # 将原始的全连接层替换为一个新的二维输出线性层
        self.model.fc = nn.Linear(num_ftrs, num_classes)

        self.learning_rate = learning_rate #设置学习率
        self.loss_fn = nn.CrossEntropyLoss() #交叉熵损失
        self.accuracy = Accuracy(task="multiclass", num_classes=2)#定义准确率
    # 其他函数
```

6. utils.py:

```
# 训练的两个模型分别对应0和1的version值，根据version值选择checkpoint路径
def get_checkpoint_dir(version):
    if version == 0: #res18的checkpoint保存路径
        dir =
"train_logs/resnet18_pretrain_test/version_3/checkpoints/resnet18_pretrain_test-
epoch=03-val_loss=0.21.ckpt"
    elif version == 1: #res34的checkpoint保存路径
        dir =
"train_logs/resnet34_pretrain_test/version_0/checkpoints/resnet34_pretrain_test-
epoch=14-val_loss=0.05.ckpt"
    else:
        raise ValueError("Unsupported version: {}".format(version))
    return dir
```

(3)结果准确率表格：

Dataset	Accuracy
test1	0.926
test2	0.663
test3_fgsm	0.342
test3_gaussian	0.583
test3	0.516

可以看到五个数据集的准确性都不错，证明模型鲁棒性不错。

(4)创新点：

1. 训练了两个模型:res18和res34，可以比对不同模型的识别能力
2. 设计了命令行的接口：

- classify.py中可以使用命令行格式,通过参数args指定多个图片文件夹路径,获取结果,不需要将图片进行ToTensor预处理

```
#calssify.py部分代码
def main(args):
  # 加载本地图片并进行预处理
  image_paths = []
  # 遍历全部图片路径
  for directory in args.dir:
      if os.path.isdir(directory):
          for root, _, files in os.walk(directory):
              for file in files:
                  if file.lower().endswith(('.png', '.jpg', '.jpeg')):
                      image_paths.append(os.path.join(root, file))
      elif os.path.isfile(directory) and directory.lower().endswith(('.png',
'.jpg', '.jpeg')):
          image_paths.append(directory)
```

- train.py中实例化训练器时epoch可以自己用命令行格式指定。
  eg: `max_epochs=args.epochs`，其中args是命令行传入参数
- test.py中可以使用命令行格式,自主指定模型结构和相应的batch_size,方便测试

```
#test.py部分代码
def get_args_parser():
    parser = argparse.ArgumentParser(description="Violence Test")
    # version : model版本
    parser.add_argument("--version", type=int, default=0, help="Model version")
    parser.add_argument("--bs", type=int, default=4, help="batch size")
    return parser
```

3.工作总结:

（1）收获、心得

在训练模型识别暴力图片的过程中，加深了对暴力行为定义、类型的了解，也进一步认识了暴力问题领域的背景。其次，通过训练不同的模型，锻炼掌握了机器学习和深度学习技术，包括数据预处理、模型构建、模型调优等。在解决遇到的各种问题的过程中，如标签噪声、过拟合等，了解了不同问题的出现原因和背景，增加了经验、磨炼了心境。最后，在团队成员协作的过程中，提高了团队合作和沟通能力。

（2）遇到问题及解决思路

- 问题：标签噪声：数据集中的标签部分标记错误。
  解决方法：进行数据集清理，人工审核并纠正错误标签。
- 问题：过拟合：模型在训练集上表现良好，但在测试集上表现不佳。
  解决方法：增加数据量或采用数据增强技术以增加数据的多样性，或者采用早停策略来避免过拟合
- 问题：模型偏差：对不同类型的暴力行为的识别准确率不同。
  解决方法：确保训练数据集尽可能涵盖各种暴力行为和背景，训练提高模型的识别度。

4.课程建议：

导论课程内容以人工智能各个领域的理论讲述为主，作业布置大多也偏理论。但是大作业的操作难度对同学们的AI领域基础和操作能力有一定的要求。建议可以把理论授课+大作业的形式改成按模块教学的形式，如机器学习、人工智能、计算机视觉等不同的模块，每个模块均采取理论+代码实践的形式，最后选取学生得分最高的两三个模块平均分即可。这样可以适当降低每个模块的实操难度，使学生不过多拘泥于知识细节，能更好了解人工智能技术的发展。