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：16

小组人员：张佳怿、刘明澄、全诗琪、邵振宇

完成时间：2024.6.21

1． 任务目标

基于暴力图像检测数据集，构建一个 2 分类检测模型，0 代表正常图像、1 代表不良图像。尽量提升分类准确率；具有一定的泛化能力，能够识别类似训练集图像，对于 AIGC 风格变化、图像噪声、对抗样本等具有一定的鲁棒性；有合理的运行时间。

2． 具体内容

（1）实施方案

1) 模型选取

选用 **ShuffleNet v2** 作为基础模型。 ShuffleNet v2 是一种轻量级神经网络架构，引入通道分组和深度可分离卷积等创新设计，实现了高效的计算和较低的参数量，同时保持了较高的精度表现；被广泛应用于图像分类、目标检测等任务。

对其最后的全连接层进行了自定义。self.model.fc=nn.Sequential，nn.Sequential 是一有序的容器，容纳了一系列顺序依次执行的层。其中设置的层依次为：将特征数减半的线性层 nn.Linear(num_ftrs, num_ftrs // 2)、增加模型非线性能力的 nn.ReLU()、防止过拟合的 nn.Dropout(p=dropout_rate)、以及最后一个将特征数映射到分类数的线性层 nn.Linear(num_ftrs // 2, num_classes)。

损失函数采用常用于多分类任务的交叉熵损失, 准确率定义为 2 分类任务的准确率。

2) 图像获取

模型读入 224*224 的 RGB 图像，名称中开头一位作为标注。0 表示非暴力、1 表示暴力。原始训练集加验证集共 8857 张图像，其中非暴力与暴力之比约为 0.9:1；训练集与验证集之比约为 7:1。

测试集分为三个，每个约 801 张图片，非暴力与暴力之比约 0.9:1。与训练集同源的数据作为测试集 1；AIGC 生成图像的数据作为测试集 2；加上图像噪声、对抗噪声之后的数据作为测试集 3。

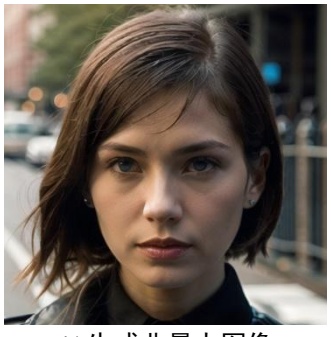
[1] 同源测试集 1:

随机抽取训练集、验证集数据。其中训练集与验证集之比约为 7:1。

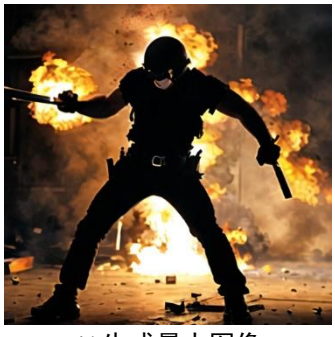
[2] AI 生成测试集 2:

在原 test 图像基础上进行风格变化。利用 stable diffusion 进行图片重绘，重绘度设定为 30%左右以保证相似性，保持暴力/非暴力属性不变。获取 100 张图片。

使用 AI 生成全新图片 701 张。先生成 512*512 的图片以保证图像



AI 生成非暴力图像



AI 生成暴力图像

质量，再批量修改图片大小代码到 224*224。

[3] 图像噪声、对抗样本测试集 3:

添加常见噪声三种，高斯噪声、随机噪声与椒盐噪声。各 150 张图片。随后对高斯噪声图像进行高斯滤波、对椒盐噪声图像进行均值滤波或中值滤波，得到各 150 张对抗样本。

高斯噪声　椒盐噪声　随机噪声
高斯滤波　中值滤波　均值滤波

（2）核心代码分析

1) 接口类

__init__(self, model_path)初始化模型。先定义模型运行的设备，再调用 load_model 加载预训练模型。self.model.eval()将模型设置为评估模式，在评估模式下禁用梯度计算，可减少内存消耗和加速计算。最后定义 self.preprocess 图像预处理为张量转换。

load_model 进行具体的模型加载工作。先 ViolenceClassifier()初始化模型，再通过 torch.load 将模型加载到目标设备，查看 model.load_state_dict 检验是否成功。

classify_folder(self, folder_path: str) -> list 将 folder_path 中的图像读入，转换为 RGB 格式，self.preprocess 后加入图像列表，再转换为 n*3*224*224 的 tensor。末调用 classify。

classify 将输出长度为 n 的 python 列表，每个值为对应的预测类别。先将输入张量移到指定设备，outputs = self.model(input_tensor)通过模型前向传播计算输出。再在输出张量的第一维度上找到最大值的索引，即预测的类别。末将预测结果移到 CPU，并转换为 Python 列表返回。

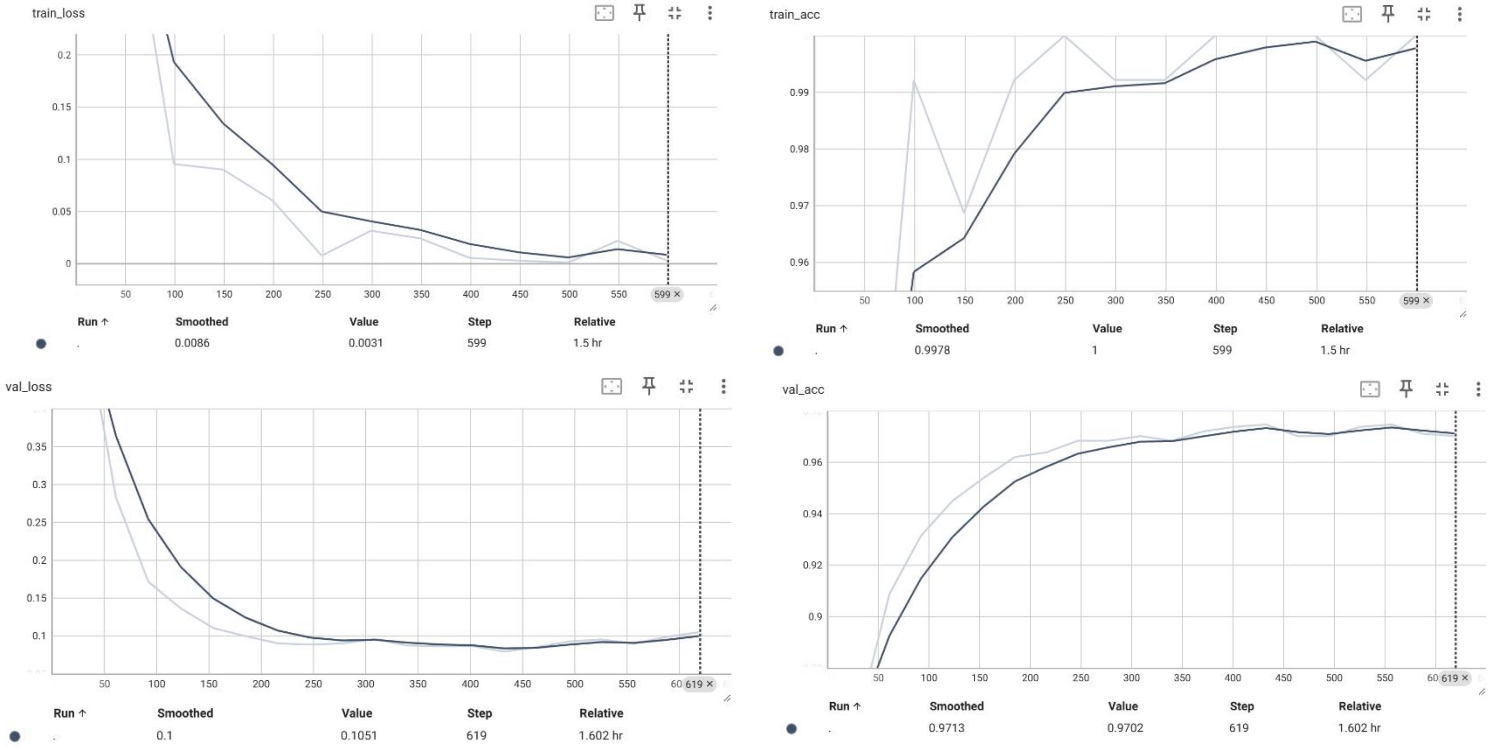
2) 模型训练与测试

[1] 训练

Model.py 中 ViolenceClassifier 类创建模型并定义损失函数、准确率。self.model = models.shufflenet_v2_x1_0(pretrained=True)加载预训练的 ShuffleNet v2 模型。随后用自定义的全连接层替换原来的全连接层。

training_step(self, batch, batch_idx)定义在训练过程中每个批次调用的方法。x, y = batch 将批次数据解包为输入数据 x 和对应的标签 y。logits = self(x)将输入数据 x 传递给模型，得到预测的 logits 值。 self(x)调用模型的 forward 方法，前向传播。最后计算损失，并返回当前批次的损失值，用于优化步骤，以更新模型参数。

训练过程中，主要调节 lr 学习率、batch_size 与 epoch。最后设置 lr=6e-5, batch_size = 128。



train_loss/train_acc/val_loss/val_acc 随 epoch 增长的变化曲线

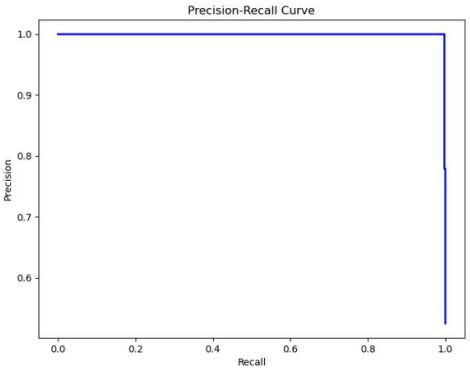
[2] 测试

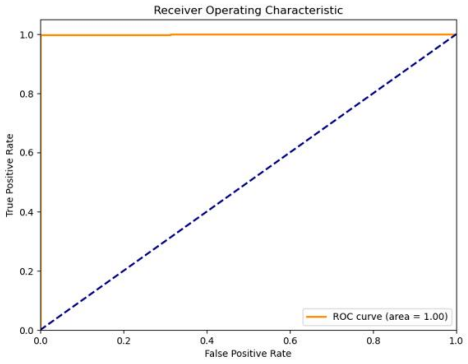
test_step 定义在训练过程中每个批次调用的方法。self.model.eval()将模型设置为评估模式，关闭 Dropout 和 BatchNorm 层的训练行为。将批次数据解包为输入数据 x 和对应的标签 y。前向传播得到预测的 logits。计算损失和准确率并记录。最后返回一个包含测试损失、准确率、logits 和真实标签的字典。

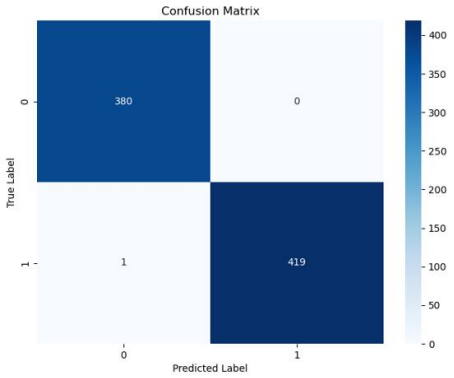
测试阶段结束时调用 test_epoch_end(self, outputs)，利用 torch.cat 将所有批次的 logits 以及真实标签合并为一个张量。y_pred = torch.argmax(logits, dim=1)通过在维度 1 上取最大值的索引，计算预测标签。y_score = torch.softmax(logits,

dim=1).cpu().numpy()计算预测分数，并将其转换为 NumPy 数组。

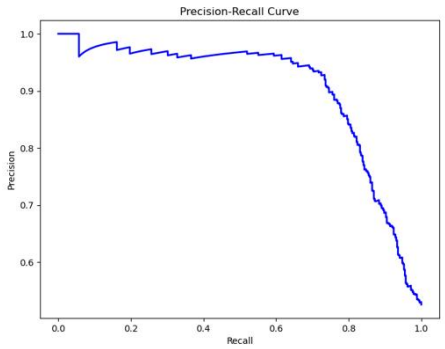
最终模型上测试集准确率为 test1 0.99875，test2 0.81024，test3 0.89000。

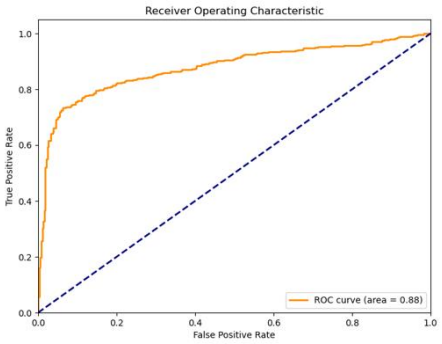


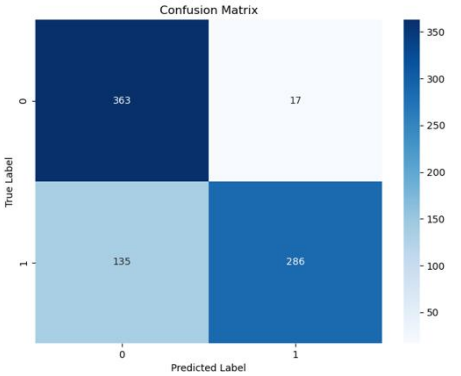




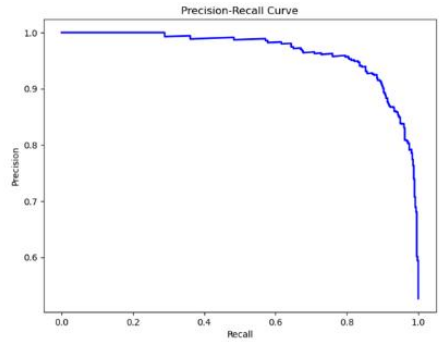
测试集 1　　Precision-Recall **曲线**　　ROC **曲线**　　Confusion Matrix

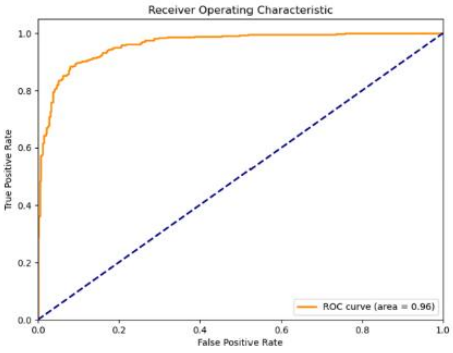


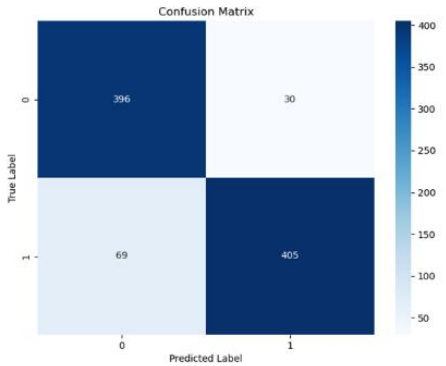




测试集 2　　Precision-Recall **曲线**　　ROC **曲线**　　Confusion Matrix







测试集 3　　Precision-Recall **曲线**　　ROC **曲线**　　Confusion Matrix

3．　工作总结

（1）收获、心得

通过选取图像分类模型，加深了我们对课堂内容的理解，帮助我们更好地巩固理论知识。动手创建、训练模型锻炼了我们的资料查阅能力与实践能力。同时，通过小组协作，增进了同学间交流，也提升了合作能力。

（2）遇到问题及解决思路

[1] 数据增强方法选取

一开始同时应用了 transforms 随机仿射变换、随机调整亮度、对比度、饱和度和色调等多种数据增强方法，结果训练集和验证集效果都不好，最后只保留了高斯模糊。

[2] AIGC 测试集准确率低

可能由 AIGC 生成图像与原训练集图像内容差异较大导致。生成非暴力图像时，所采用的 AIGC 生成图像所用提示词较为单一，图片集中于油画与人物；生成的暴力图像大部分都通过黑暗火焰表达电影风格的暴力，而非日常斗殴等暴力活动。

4． 课程建议

增加实践环节：熟悉人工智能技术需要更多的实践，由此也能够帮助我们更好的巩固理论知识。可以在课程中增加更多的小习题项目，强化所学知识。

引入前沿研究：人工智能领域发展迅速，适时引入最新的研究进展和实际案例，能够在激发兴趣的同时体现出人工智能发展方向，使得课程内容更加丰富。