**程序报告**

学号：2013750 姓名：管昀玫

1. **问题重述**

（简单描述对问题的理解，从问题中抓住主干，必填）

====================================================================

今年一场席卷全球的新型冠状病毒给人们带来了沉重的生命财产的损失。

有效防御这种传染病毒的方法就是积极佩戴口罩。

我国对此也采取了严肃的措施，在公共场合要求人们必须佩戴口罩。

在本次实验中，我们要建立一个目标检测的模型，可以识别图中的人是否佩戴了口罩。

1. **设计思想**

（所采用的方法，有无对方法加以改进，该方法有哪些优化方向（参数调整，框架调整，或者指出方法的局限性和常见问题），伪代码，理论结果验证等… **思考题，非必填**）

====================================================================

1. 首先需要训练正样本和负样本，需要调整图片的尺寸。

2. 制作训练时所需要的批量数据集：需要创建一个Dataset对象和Dataloader对象

3. 进行MTCNN人脸检测：通过搭建 MTCNN 网络实现人脸检测

4. 进行口罩识别：加载预训练模型MobileNet，并手动调整学习率。、

5. 进行模型训练

1. **代码内容**

（能体现解题思路的主要代码，有多个文件或模块可用多个"===="隔开，必填）

====================================================================

def processing\_data(data\_path, height=224, width=224, batch\_size=64,

test\_split=0.1):

"""

数据处理部分

:param data\_path: 数据路径

:param height:高度

:param width: 宽度

:param batch\_size: 每次读取图片的数量

:param test\_split: 测试集划分比例

:return:

"""

transforms = T.Compose([

T.Resize((height, width)),

T.RandomHorizontalFlip(0.1), # 进行随机水平翻转

T.RandomVerticalFlip(0.1), # 进行随机竖直翻转

T.ToTensor(), # 转化为张量

T.Normalize([0], [1]), # 归一化

])

dataset = ImageFolder(data\_path, transform=transforms)

# 划分数据集

train\_size = int((1-test\_split)\*len(dataset))

test\_size = len(dataset) - train\_size

train\_dataset, test\_dataset = torch.utils.data.random\_split(dataset, [train\_size, test\_size])

# 创建一个 DataLoader 对象

train\_data\_loader = DataLoader(train\_dataset, batch\_size=batch\_size,shuffle=True)

valid\_data\_loader = DataLoader(test\_dataset, batch\_size=batch\_size,shuffle=True)

return train\_data\_loader, valid\_data\_loader

data\_path = './datasets/5f680a696ec9b83bb0037081-momodel/data/image'

train\_data\_loader, valid\_data\_loader = processing\_data(data\_path=data\_path, height=160, width=160, batch\_size=64)

def show\_tensor\_img(img\_tensor):

img = img\_tensor[0].data.numpy()

img = np.swapaxes(img, 0, 2)

img = np.swapaxes(img, 0, 1)

img = np.array(img)

plot\_image(img)

for index, (x, labels) in enumerate(train\_data\_loader):

print(index, "\nfeature:",x[0], "\nlabels:",labels)

show\_tensor\_img(x)

break

# 加载 MobileNet 的预训练模型权

device = torch.device("cuda:0") if torch.cuda.is\_available() else torch.device("cpu")

train\_data\_loader, valid\_data\_loader = processing\_data(data\_path=data\_path, height=160, width=160, batch\_size=64)

modify\_x, modify\_y = torch.ones((64, 3, 160, 160)), torch.ones((64))

epochs = 30

model = MobileNetV1(classes=2).to(device)

optimizer = optim.Adam(model.parameters(), lr=1e-3) # 优化器

print('加载完成...')

# 学习率下降的方式，acc三次不下降就下降学习率继续训练，衰减学习率

scheduler = optim.lr\_scheduler.ReduceLROnPlateau(optimizer,

'max',

factor=0.5,

patience=10)

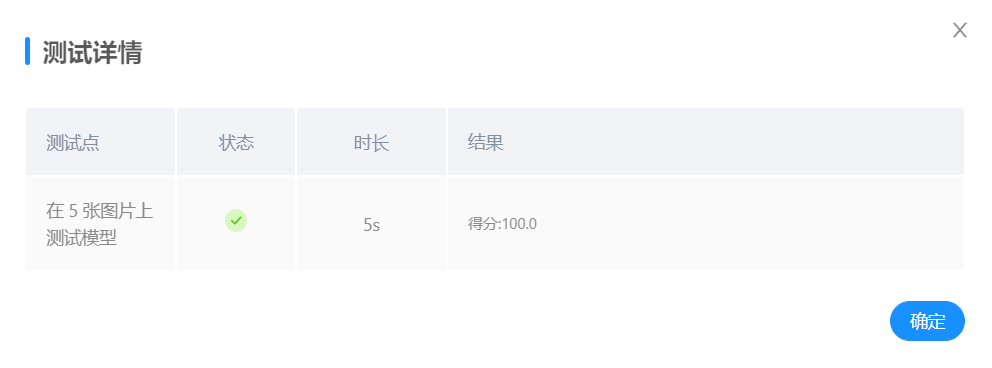
# 损失函数

criterion = nn.CrossEntropyLoss()

1. **实验结果**

（实验结果，必填）

====================================================================



1. **总结**

（自评分析（是否达到目标预期，可能改进的方向，实现过程中遇到的困难，从哪些方面可以提升性能，模型的超参数和框架搜索是否合理等），**思考题，非必填**）

====================================================================

本次实验总体还算可以。

一些优化的想法：人脸检测的正确率还不是很高，通过查阅资料发现了另外一种人脸检测算法：RetinaFace

RetinaFace: Single-stage Dense Face Localisation in the Wild

以下为摘要：

虽然在未受控制的人脸检测方面取得了巨大进步，但野外准确有效的面部定位仍然是一个开放的挑战。这篇文章提出了一个强大的单阶段人脸检测器，名为RetinaFace，它利用联合监督和自我监督的多任务学习，在各种人脸尺度上执行像素方面的人脸定位。具体来说，我们在以下五个方面做出了贡献：（1）我们在WIDER FACE数据集上手动注释五个面部标志，并在这个额外的监督信号的帮助下观察硬面检测的重要改进。 （2）我们进一步增加了一个自监督网格解码器分支，用于与现有的受控分支并行地预测像素三维形状的面部信息。 （3）在WIDER FACE硬测试装置上，RetinaFace的性能优于现有技术平均预测（AP）1.1％（达到AP等于91.4％）。 （4）在IJB-C测试集上，RetinaFace使最先进的方法（ArcFace）能够改善他们在面部验证中的结果（FAR = 1e-6的TAR = 89.59％）。 （5）通过采用轻量级骨干网络，RetinaFace可以在单个CPU内核上实时运行，以实现VGA分辨率的显示。