

# 浙江大学



## 本科实验报告

姓名： 严轶凡

---

学院： 控制科学与工程学院

---

系： 自动化

---

专业： 自动化（控制）

---

学号： 3200100917

---

指导教师： 周建光

---

年    月    日

# 浙江大学 实验报告

## 一、实验目的和要求（必填）

- （1）建立深度学习模型，检测出图中的人是否佩戴了口罩，并将其尽可能调整到最佳状态。
- （2）学习经典的模型 MTCNN 和 MobileNet 的结构。
- （3）学习训练时的方法。

## 二、实验内容和原理（必填）

### 实验内容：

针对目标检测的任务，可以分为两个部分：目标识别和位置检测。

通常情况下，特征提取需要由特有的特征提取神经网络来完成，如 VGG、MobileNet、ResNet 等，这些特征提取网络往往被称为 Backbone。而在 Backbone 后面接全连接层(FC)就可以执行分类任务。

但 FC 对目标的位置识别乏力。经过算法的发展，当前主要以特定的功能网络来代替 FC 的作用，如 Mask-Rcnn、SSD、YOLO 等。

我们选择充分使用已有的人脸检测的模型，再训练一个识别口罩的模型，从而提高训练的开支、增强模型的准确率。

### 实验原理：

本次实验中使用 MobileNetV1 训练模型进行口罩识别。

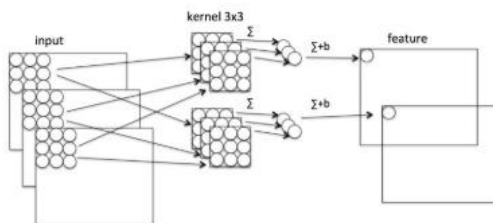
传统卷积神经网络，内存需求大、运算量大，而 MobileNet 网络是由 google 团队在 2017 年提出的，专注于移动端或者嵌入式设备中的轻量级 CNN 网络。相比传统卷积神经网络，在准确率小幅降低的前提下大大减少模型参数与运算量。

MobileNetV1 中使用了传统的普通卷积和深度可分离卷积。

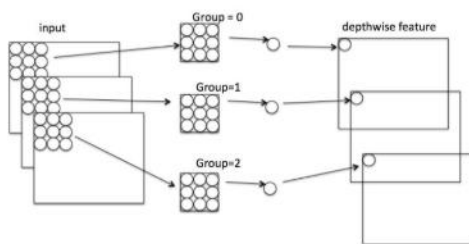
传统卷积方式的卷积核做卷积运算时得同时考虑对应图像区域中的所有通道（channel），而深度可分离卷积对不同的通道采用不同的卷积核进行卷积，它将普通卷积分解成了深度卷积（Depthwise Convolution）和逐点卷积（Pointwise Convolution）两个过程，这样可以将通道（channel）相关性和空间（spatial）相关性解耦。

深度可分离卷积将以往普通卷积操作同时考虑通道和区域改变（卷积先只考虑区域，然后再考虑通道），实现了通道和区域的分离。

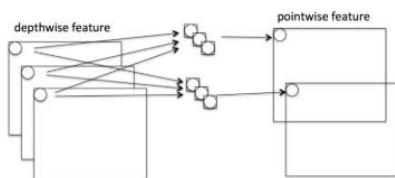
下图所示为标准卷积过程：



下图所示为深度卷积过程：



下图所示为逐点卷积过程：



MobileNetV1 的网络结构如图所示，前面的卷积层中除了第一层为标准卷积层外，其他都是深度可分离卷积（Conv dw + Conv/s1），卷积后接了一个 7\*7 的平均池化层，之后通过全连接层，最后利用 Softmax 激活函数将全连接层输出归一化到 0-1 的一个概率值，根据概率值的高低可以得到图像的分类情况。

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32 \text{ dw}$	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64 \text{ dw}$	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128 \text{ dw}$	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256 \text{ dw}$	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5×	Conv dw / s1	$3 \times 3 \times 512 \text{ dw}$
	Conv / s1	$1 \times 1 \times 512 \times 512$
	Conv dw / s2	$3 \times 3 \times 512 \text{ dw}$
	Conv / s1	$1 \times 1 \times 512 \times 1024$
	Conv dw / s2	$3 \times 3 \times 1024 \text{ dw}$
	Conv / s1	$1 \times 1 \times 1024 \times 1024$
	Avg Pool / s1	Pool $7 \times 7$
	FC / s1	$1024 \times 1000$
	Softmax / s1	Classifier

### 三、实验环境

Python	3.8.15
numpy	1.23.4
torch	1.13.0+cu116
matplotlib	3.5.3

## 四、操作方法与实验步骤

### （一）面部识别

本次实验中，调用 MTCNN 进行人脸检测，能够从含有多副人脸的图像中提取出人脸图像，以便于后续人脸口罩识别。

### （二）数据集划分

使用 processing\_data 函数将数据集文件夹中的数据划分为训练集和验证集，同时将图片进行尺寸上的调整，以及数据归一化处理，以便于后续 pytorch 的模型训练。

```
train_data_loader, valid_data_loader = processing_data(  
    data_path=data_path,  
    height=train_height,  
    width=train_width,  
    batch_size=train_batch_size  
)
```

### （三）模型结构修改

考虑实际性能，将 MobileNetV1 模型结构设置如下：

```
class MobileNetV1(nn.Module):  
    def __init__(self, num_classes=2):  
        super(MobileNetV1, self).__init__()  
        self.mobilebone = nn.Sequential(  
            self._conv_bn(3, 32, 2),  
            self._conv_dw(32, 64, 1),  
            self._conv_dw(64, 128, 2),  
            self._conv_dw(128, 128, 1),  
            self._conv_dw(128, 256, 2),  
            self._top_conv(256, 256, 5),  
            self._conv_dw(256, 512, 2),  
            self._top_conv(512, 512, 1),  
            # self._conv_dw(512, 1024, 2),  
            # self._conv_dw(1024, 1024, 1),  
        )  
  
        self.avg_pool = nn.AdaptiveAvgPool2d(1)  
        self.fc = nn.Linear(512, num_classes)  
        for m in self.modules():  
            if isinstance(m, nn.Conv2d):  
                n = m.kernel_size[0] * m.kernel_size[1] * m.out_channels  
                m.weight.data.normal_(0, (2. / n) ** .5)  
            if isinstance(m, nn.BatchNorm2d):  
                m.weight.data.fill_(1)  
                m.bias.data.zero_()
```

```
def forward(self, x):
    x = self.mobilebone(x)
    x = self.avg_pool(x)
    x = x.view(x.size(0), -1)
    out = self.fc(x)
    return out
```

#### （四）模型训练

训练轮数 epoch 设置为 100;

优化器 optimizer 使用 Adam 优化器;

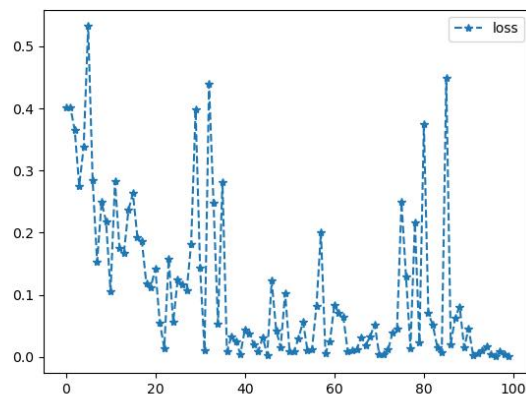
损失函数 criterion 使用交叉熵函数 CrossEntropyLoss();

在模型训练过程中，分别记录每一轮训练过程中的训练误差 loss 以及模型在验证集上预测的正确率 correct，并且以正确率为评价指标，记录正确率最高时的模型结构参数。

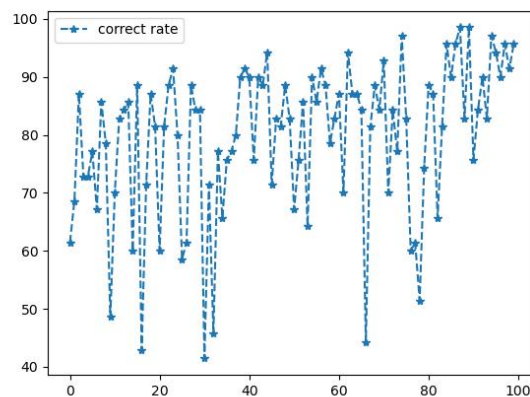
在模型训练结束后将模型权重参数以文件形式进行保存。

#### 五、实验数据记录和处理


在训练过程中，训练集的交叉熵损失函数变化曲线图如图所示，可见虽然损失虽然存在波动，但整体趋势随着训练过程逐渐下降。



在训练过程中，模型在验证集上的正确率变化曲线图如图所示，最高正确率为 98.57%，该正确率最高的模型被保存以便于后续预测。



最终模型在平台上的测试效果如下，可见模型在口罩预测上实现了较好的实际预测效果。

测试点	状态	时长	结果
在 5 张图片上 测试模型		6s	得分:100.0

### 六、讨论、心得

通过本次实验，我对机器学习训练深度模型的基本步骤有了更深的理解，同时掌握了使用 pytorch 框架进行深度学习框架搭建的流程和原理。

同时在本次口罩识别实验中，我对 MTCNN 人脸识别框架和 MobileNetV1 轻量神经网络的原理和结构有了更进一步的了解，也积累了在深度学习代码编写中调整参数的相关经验。

同时在本次实验中。MobileNetV1 取得了较好的预测效果，分析 MobileNet 能够加快网络训练速度的原因，首先就是使用了深度可分离卷积，这极大的减少了计算量（Mult-Adds）。另一方面，是网络的实现方式。

除此以外，设计者还应用了 MobileNet 的核心超参数，也就是宽度乘子和分辨率乘子，在 MobileNet 的调节过程中起到了关键作用。