神经网络 Assignment1 实验报告

181240047 匡亚明学院 戎奕名

一、 实验环境

操作系统: Windows 10 中文版

CPU: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz

GPU: NVIDIA GeForce MX150

Python package: Pytorch

二、 模型架构

我们的任务是将 48×48 的脸部灰度图像根据表情分为七类，是比较典型的图像分类问题，因此准备使用卷积神经网络模型。

数据预处理时，我将 train.csv 文件和 test.csv 文件中的数据经过分割和类型转换存储在了 train.json 文件和 test.json 文件中。一开始选取 train 中后 2800 个作为 validation set，但发现后 2800 个中类型 1 偏多，因此选取了前 2800 张图片作为 validation set，其余为 training set。

1. Model1

首先，我仿照了 LeNet 实现了一个最为简单的神经网络，由两个卷积层和两个线性层组成，激活函数选择了 Relu，激活之后又通过一个池化层，训练 2 个 epoch 的模型的 validation accuracy 仅有 20%~30%，且训练过程中 loss 的值经过几次迭代后常为 NAN，因此我考虑用 BatchNormalization 进行处理。处理后 loss 稳定了，validation accuracy 约为 40%~50%。

2. Model2

为了进一步提升网络的性能，仿照 AlexNet 的实现，添加了 Dropout 和 L2 正则化，最终 training accuracy 约为 93%时 validation accuracy 在 56%左右。

3. Model3

我自己书写了一个 ResNet18 以加深对残差神经网络的理解。

根据通用近似定理，对于具有线性输出层和至少一个有“挤压”（如 Sigmoid, Relu 等）性质的激活函数组成的前馈神经网络，在隐藏层神经元数量足够多的情况下，可以以任意精度近似任何一个定义在实数空间的有界闭集函数。但是随着层数的增加，学习会变得十分困

难。残差网络的核心思想在于让非线性单元去近似残差函数，这相当于用非线性单元加恒等函数近似目标函数，而残差函数在实际中更容易学习。

$$h(\boldsymbol{x}) = \underbrace{\boldsymbol{x}}_{\text{恒等函数}} + \underbrace{(h(\boldsymbol{x}) - \boldsymbol{x})}_{\text{残差函数}}.$$

最终的训练结果为在 training accuracy 为 93.52%时 validation accuracy 为 54.3%。但是如果添加 Dropout 和 L2 正则化解决过拟合的问题，训练速度就会变得很慢。并且需要分类的图片本来只有 48×48，网络层数过多不便于观测每层的 output。迫于硬件设备和时间，没有继续调整 ResNet。

4. Final Model

最终在 Model2 的基础上改进得到了最终的网络结构如下：

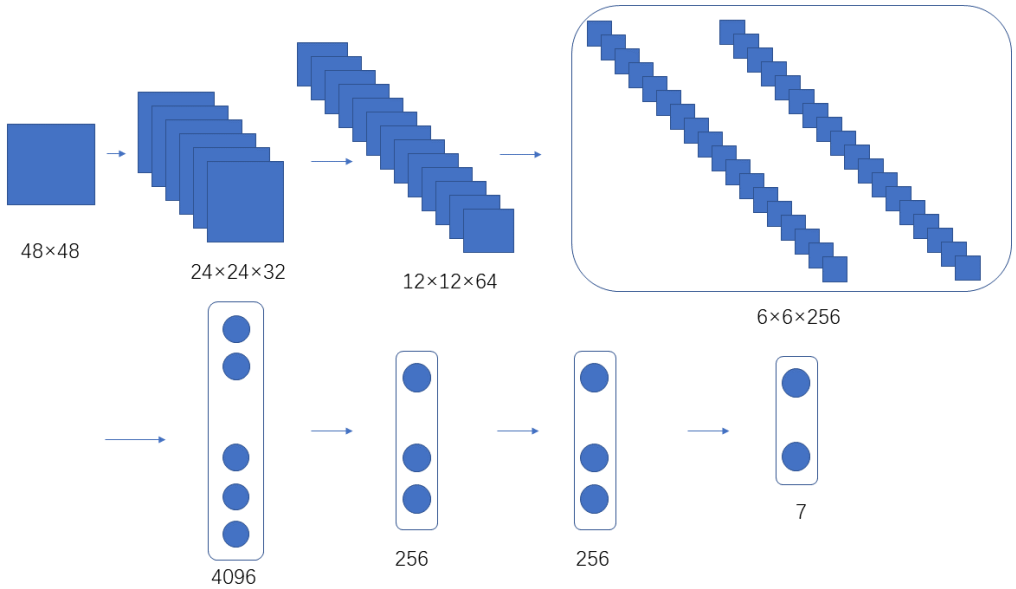
卷积层由三个 block 构成：

```
self.conv1 = nn.Sequential(
    nn.Conv2d(1, 32, 3, stride=1, padding=1),
    nn.BatchNorm2d(num_features=32),
    nn.RReLU(inplace=True),
    nn.MaxPool2d(2, 2),
    nn.Dropout(0.1)
)

self.conv2 = nn.Sequential(
    nn.Conv2d(32, 64, 3, stride=1, padding=1),
    nn.BatchNorm2d(num_features=64),
    nn.RReLU(inplace=True),
    nn.MaxPool2d(2, 2),
    nn.Dropout(0.1)
)

self.conv3 = nn.Sequential(
    nn.Conv2d(64, 256, 3, stride=1, padding=1),
    nn.BatchNorm2d(num_features=256),
    nn.Conv2d(256, 256, 3, stride=1, padding=1),
    nn.BatchNorm2d(num_features=256),
    nn.RReLU(inplace=True),
    nn.MaxPool2d(2, 2),
)
```

全连接层的结构如下：

```
self.fully_connect = nn.Sequential(
    nn.Dropout(0.5),
    nn.Linear(256*6*6, 4096),
    nn.BatchNorm1d(4096),
    nn.RReLU(inplace=True),
    nn.Linear(4096, 256),
    nn.BatchNorm1d(256),
    nn.RReLU(inplace=True),
    nn.Linear(256, 256),
    nn.BatchNorm1d(256),
    nn.RReLU(inplace=True),
    nn.Linear(256, 7)
)
```

示意图如下：



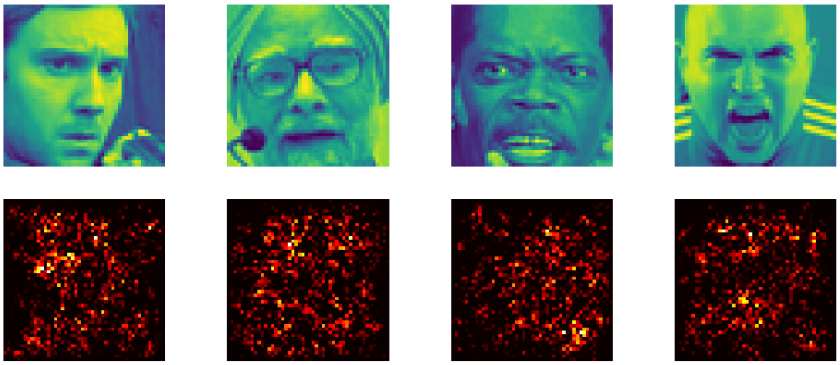
训练 70 个 epoch 后 training accuracy 约为 87.9%，validation accuracy 约为 60%，kaggle 上的预测成绩为 0.61772。

三、 观察模型

1. Saliency Maps

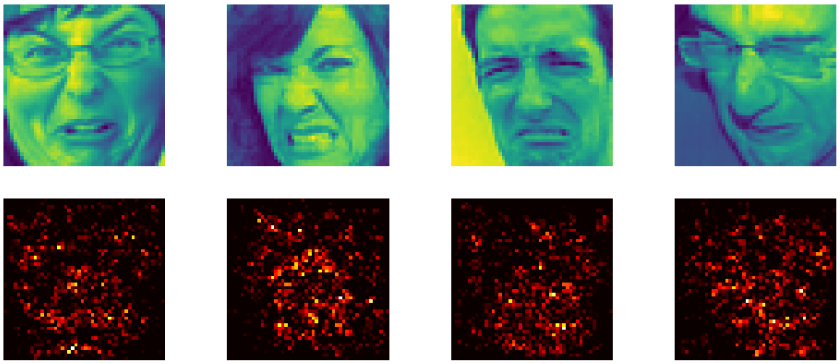
Saliency Maps 就是计算网络对每个像素的偏导，来观察网络对那些像素的注意力比较高。我在 validation set 选取了每种类型的前四张图片的 Saliency Maps 作为例子。

(1) Emotion==angry:



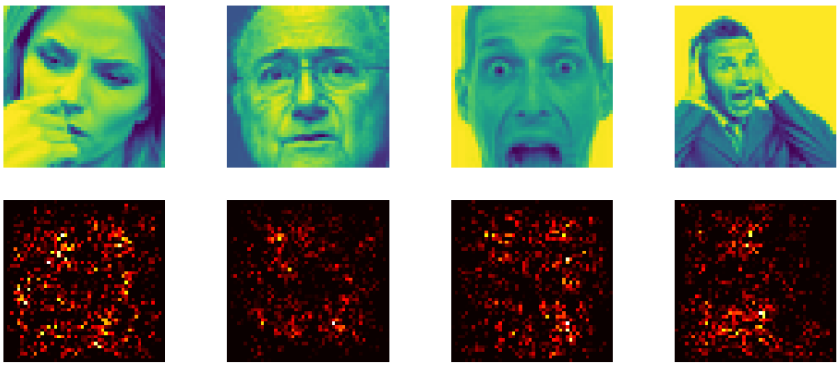
主要 focus 在面部较为中心的部分，尤其是眼睛和嘴，但特征不是特别明显，因此准确率也不是太高。

(2) Emotion==disgusted:



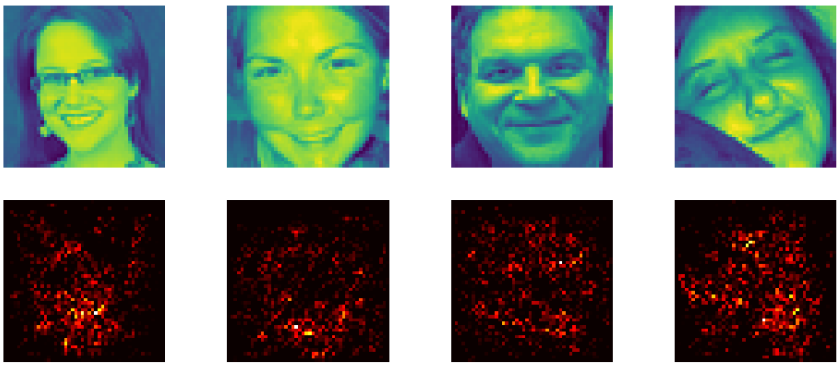
主要分布在面部中心，尤其是鼻子的部分。

(3) Emotion==terrified:



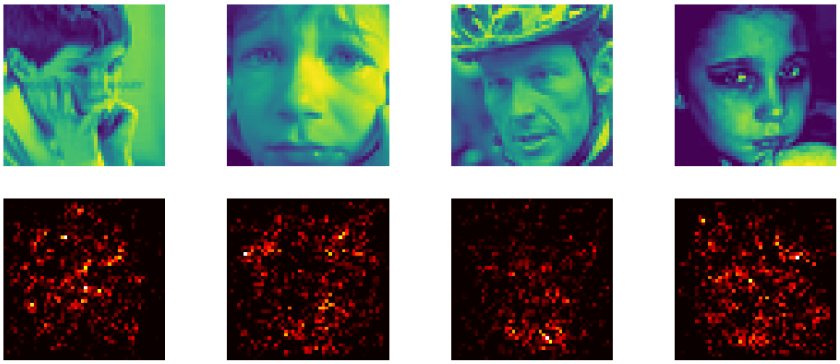
主要分布在面部，但是不是十分明显，可以看出结果不是非常好。尤其是最后一张图片，由于图片中不完全是人脸，似乎没有完全 focus 在面部。

(4) Emotion==happy:



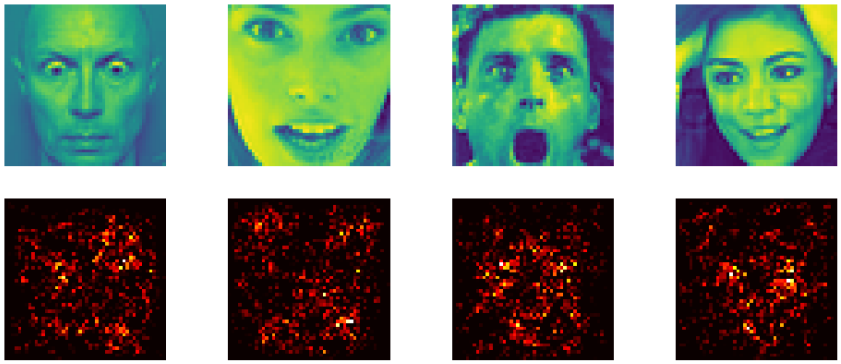
Happy 类型的 validation accuracy 最高。可以看出基本都 focus 在了嘴上的笑容，并且眼睛也有一定的注意力。开心的表情人为来分类的准确率也会比较高，因为特征比较明显。

(5) Emotion==sad:



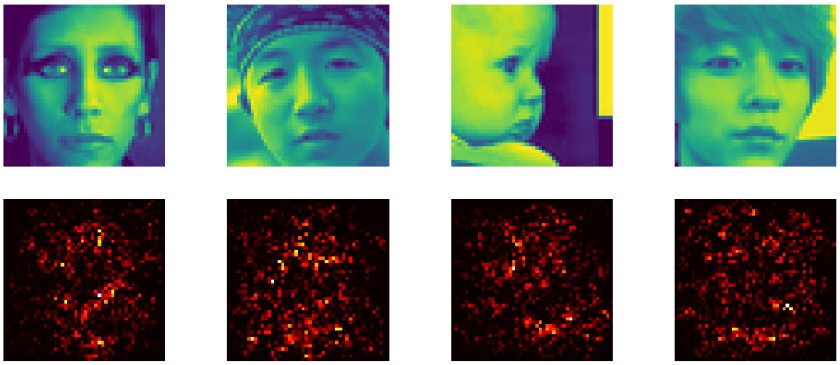
主要分布在眼睛和嘴，尤其第二张图片，可以非常明显地看出眼睛的轮廓。第一张图片因为不完全是人脸，也没有正确 focus 在人脸上。

(6) Emotion==surprised



明显 focus 在眼睛和嘴上，基本都可以看出面部的基本轮廓。

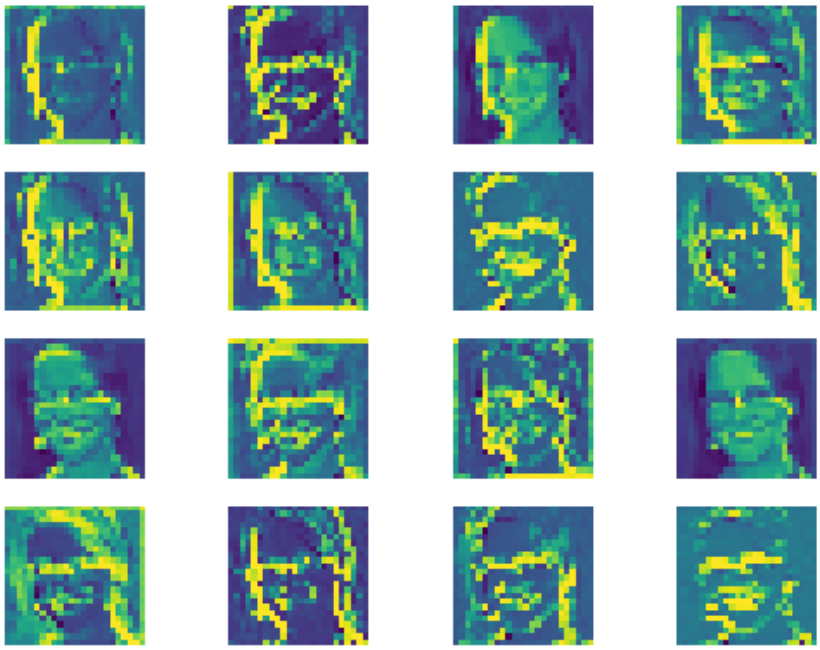
(7) Emotion==neutral:



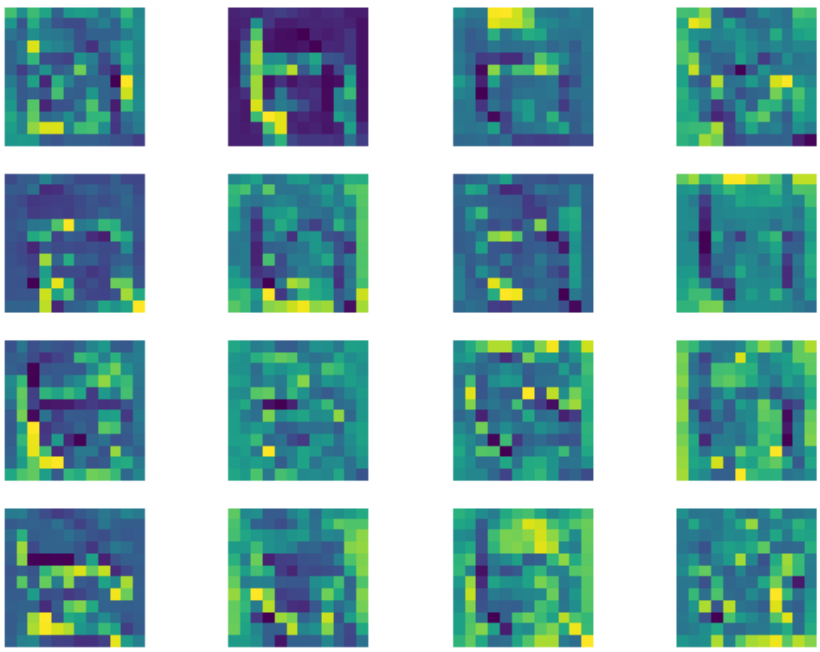
没有非常几种地分布在面部的某个区域，分布相对均匀。第三张图片由于拍摄的角度，也没有非常好地分布在面部。

2. Filter output

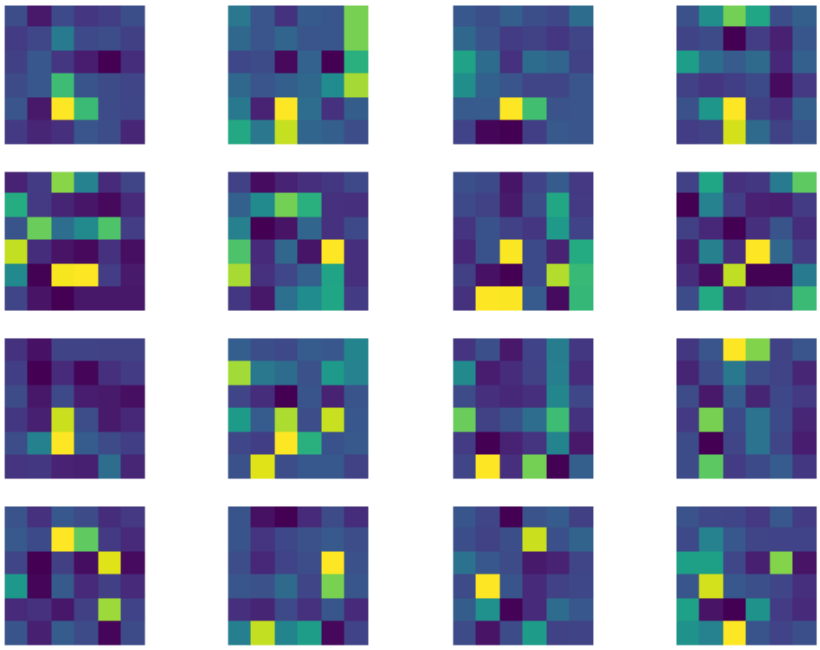
选择了每个 block（经过卷积、BN、RRelu）的前 16 个 filter 的 output 作为展示。



Happy 类型的第一张图片经过第一个 block 的结果。可以看出图片的大致轮廓，但是每个 filter 所关注的轮廓不尽相同。第一个 block 可以说是一个比较全局的判断。



每个 filter 捕捉的特征却别更加明显，有的集中在嘴、有的几种在头部等等。



第三个 block 的 output 就只是 6×6 的像素块了，可以看成是比较基本的线条。

3. 表情判断方式

我的模型对各种 label 的 training accuracy 和 validation accuracy 如下：

```
The training accuracy of 3581 angry pictures is 98.3244903658196
The training accuracy of 398 disgusted pictures is 98.99497487437185
The training accuracy of 3672 terrified pictures is 97.57625272331155
The training accuracy of 6526 happy pictures is 99.31045050566964
The training accuracy of 4378 sad pictures is 98.81224303334855
The training accuracy of 2873 superised pictures is 99.58231813435432
The training accuracy of 4481 neutral pictures is 99.08502566391431
```

```
The validation accuracy of 414 type0 pictures is 43.96135265700483.
The validation accuracy of 38 type1 pictures is 42.10526315789473.
The validation accuracy of 425 type2 pictures is 41.647058823529406.
The validation accuracy of 689 type3 pictures is 79.9709724238026.
The validation accuracy of 452 type4 pictures is 49.336283185840706.
The validation accuracy of 298 type5 pictures is 78.18791946308725.
The validation accuracy of 484 type6 pictures is 61.57024793388429.
```

由于没有时间进行 LIME 分析了，我只做一些比较主观且浅显的分析。

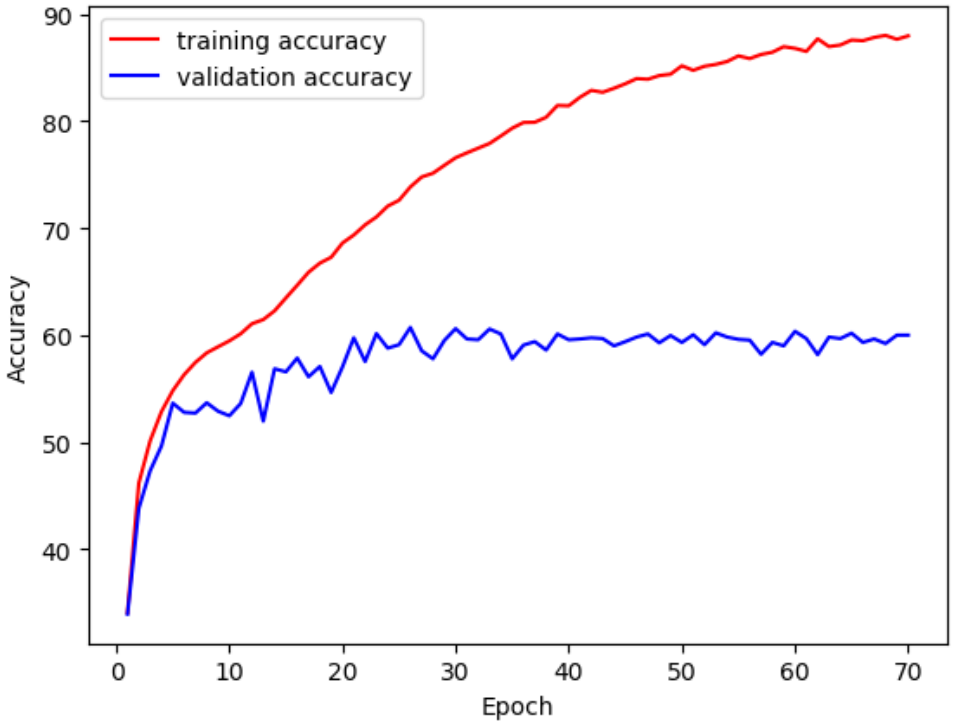
首先是训练数据量的问题。Happy 类型的训练数据最多，Disgusted 类型的训练数据最

少，只有几百张，因此从总体上看 happy 类型的准确率最高，disgusted 类型的准确率最低。

其次，有些表情的特征本来可能就比较细微，而有些表情的特征很明显。从人类的判断方式来说，happy 的表情一看眼睛和嘴就可以判断，而 sad、terrified 的表情相对比较细微。同时有些情绪的表情较难分，人类进行判断的准确率也可能不高。比如我请几位同学在不知情的情况下判断几张 terrified 类型的照片，由于表情不明显他们都认成了 neutral。

4. 训练过程

训练过程中 training loss 一直下降，validation loss 先下降后上升。validation accuracy 最后基本收敛到 60%。



在模型的调整过程中，dropout 和 L2 正则化的加入会减慢训练的速度。ResNet18 层数较多，dropout 加入的比较多收敛速度就会很慢。

四、 实验中遇到的困难及未来的问题

1. 没有理解“用 gradient ascent 方法观察特定层 filter 被图片 activate”的含义
2. 有些图片并不完全是面部，或面部的角度不一样。如果在可以 focus 到面部的模型的基础上进行分类可能会有一些更好的效果。

3. 请教了一些使用 ResNet 的同学，他们的模型似乎在不需要 dropout 的情况下就可以达到 62%左右的 validation accuracy，但是我的 ResNet 在实现上似乎没有很大的错误。由于时间关系没有继续调整 ResNet。