

研究生《深度学习》课程

实验报告

|  |  |
| --- | --- |
| **实验名称：** | **卷积神经网络实验** |
| **姓 名：** | **梁棋棋** |
| **学 号：** | **20120376** |
| **上课类型：** | **专业课** |
| **日 期：** | **2020.8.17** |

## 一、实验内容

#### 1.1 二维卷积实验

1. 手写二维卷积的实现，并在至少一个数据集上进行实验，从训练时间、预测精度、Loss变化等角度分析实验结果（最好使用图表展示）
2. 使用torch.nn实现二维卷积，并在至少一个数据集上进行实验，从训练时间、预测精度、Loss变化等角度分析实验结果（最好使用图表展示）
3. 不同超参数的对比分析（包括卷积层数、卷积核大小、batchsize、lr等）选其中至少1-2个进行分析
4. 使用PyTorch实现经典模型AlexNet并在至少一个数据集进行试验分析 （平台课同学选做，专业课同学必做）（无GPU环境则至少实现模型）
5. 使用实验2中的前馈神经网络模型来进行实验，并将实验结果与卷积模型结果进行对比分析（选作）

#### 1.2 空洞卷积实验（专业课）

1. 使用torch.nn实现空洞卷积，要求dilation满足HDC条件（如1,2,5）且要堆叠多层并在至少一个数据集上进行实验，从训练时间、预测精度、Loss变化等角度分析实验结果（最好使用图表展示）
2. 将空洞卷积模型的实验结果与卷积模型的结果进行分析比对，训练时间、预测精度、Loss变化等角度分析
3. 不同超参数的对比分析（包括卷积层数、卷积核大小、不同dilation的选择，batchsize、lr等）选其中至少1-2个进行分析（选做）

#### 1.3 残差网络实验（专业课）

1. 实现给定结构的残差网络，在至少一个数据集上进行实验，从训练时间、预测精度、Loss变化等角度分析实验结果（最好使用图表展示）
2. 将残差网络与空洞卷积相结合，在至少一个数据集上进行实验，从训练时间、预测精度、Loss变化等角度分析实验结果（最好使用图表展示） （选做）

## 二、实验设计

#### 2.1 二维卷积实验

1. 手动实现二维卷积

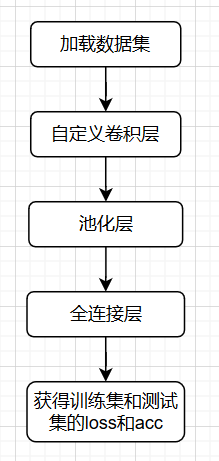


图2.1.1 手动实现二维卷积流程图

其中自定义卷积层设计实现：假设图片的宽为W，高为H，通道数为D（一般RGB图像是3），那么图片的形式就是W\*H\*D。在D的每个通道上使用kernel\_size\*kernel\_siz\*D大小的卷积核（一个卷积核得到一层）去卷积图像得到一层feature，多个卷积核就能得到多个feature。

1. 使用torch.nn实现二维卷积

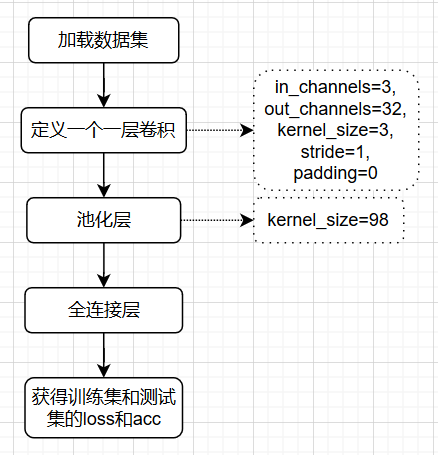


图2.1.2 使用torch.nn实现二维卷积流程图

1. 不同超参数对比分析

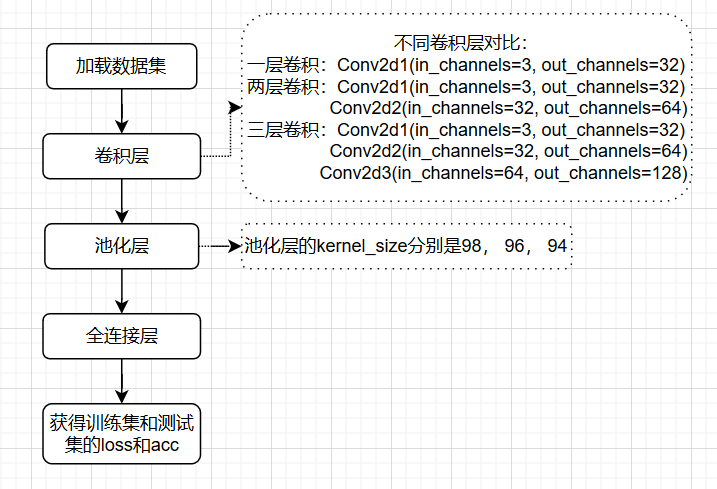


图2.1.3 不同卷积层数对比流程图

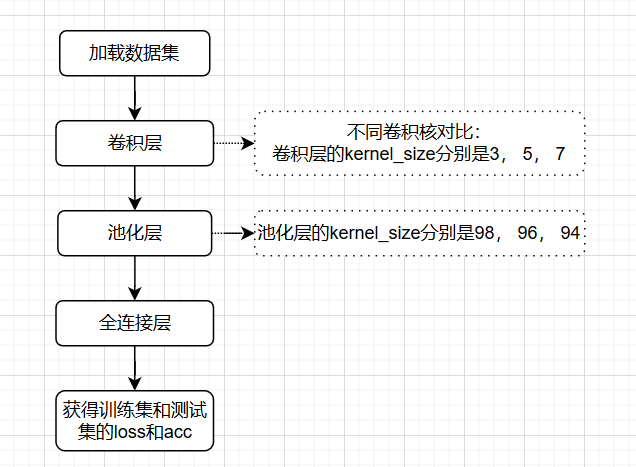


图2.1.4 不同卷积核大小对比流程图

1. 实现经典模型AlexNet

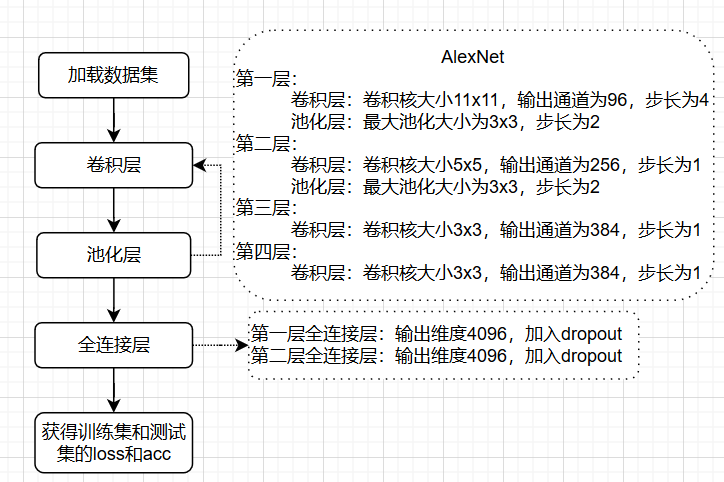


图2.1.5 AlexNet流程图

#### 2.2 空洞卷积实验

1. 利用torch.nn实现空洞卷积

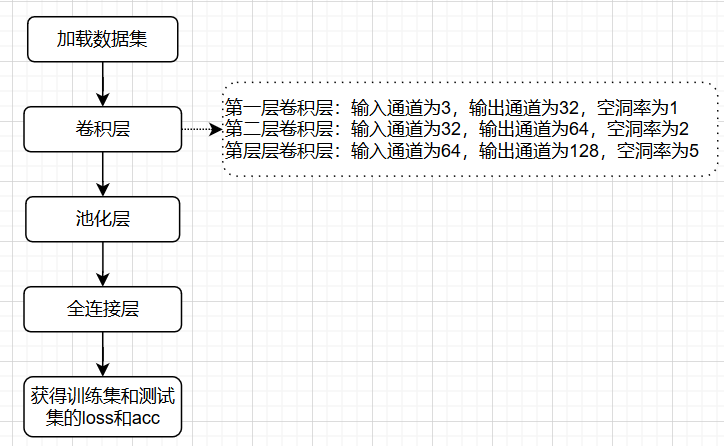


图2.2.1 利用torch.nn实现空洞卷积流程图

1. 空洞卷积和卷积对比



图2.2.2空洞卷积和普通卷积对比流程图

1. 不同参数对比

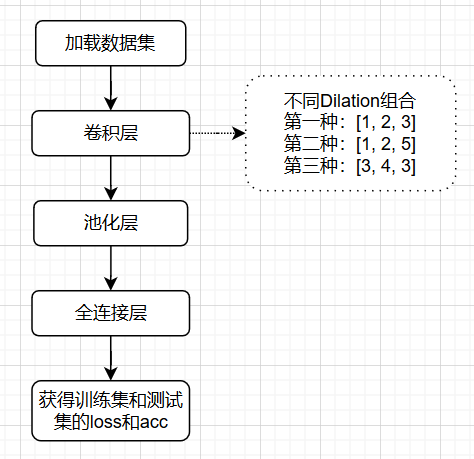


图2.2.3不同Dilation对比流程图

#### 2.3 残差网络实验

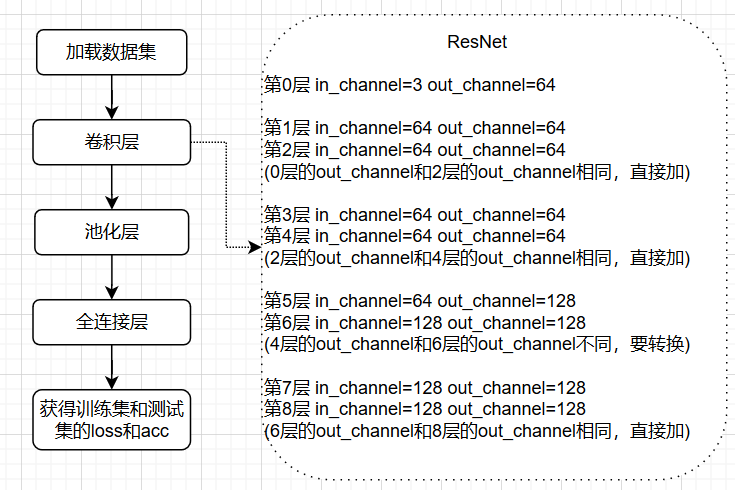


图2.3.1 8层残差网络实现流程图

## 三、实验环境及实验数据集

简单介绍实验环境和涉及的数据集

表3-1 实验环境

|  |  |
| --- | --- |
| 操作系统 | Windows 10 |
| 使用语言 | Python3.6.10 |
| 使用IDE | Jupyter notebook |
| 开发平台 | Pytorch1.5.1 |

实验数据集1：车辆分类数据集

## 四、实验过程

#### 4.1 二维卷积实验

##### 4.1.1 手动实现二维卷积

1. 数据处理：总共218张客车数据，前153张为训练集，后65张为测试集；总共779张汽车数据，前545张为训练集，后234张为测试集；总共360张货车数据，前252张为训练集，后108张为测试集。使用全0矩阵、全1矩阵和全2矩阵分别作为客车，汽车，火车的标签。

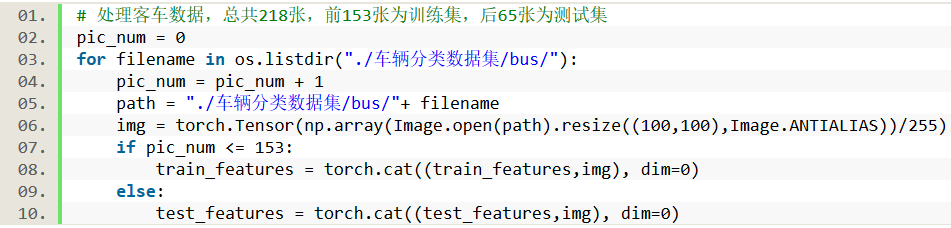


图4.1.1 特征处理代码图

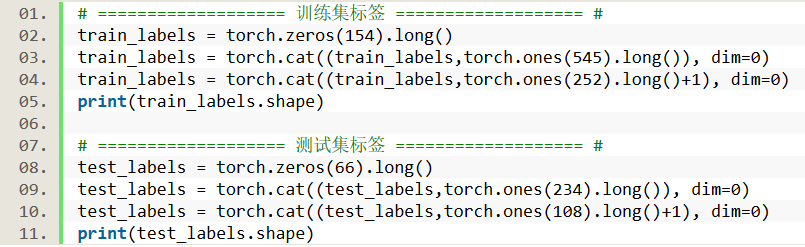


图4.1.2 标签处理代码图

1. 自定义卷积层：图片的宽为100，高为100，通道数为3，图片的形式为3\*100\*100。使用3\*3\*3大小的卷积核去卷积图像得到一层feature，设定的out\_channels=32说明有32层features

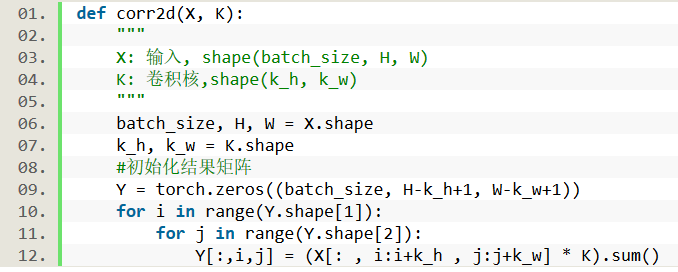


图4.1.3 单通道卷积函数代码图

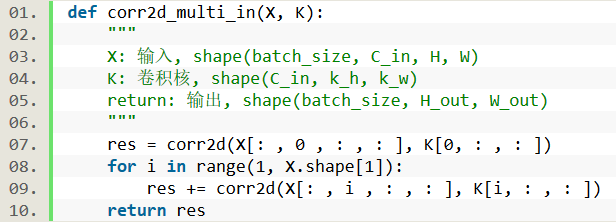


图4.1.4 多通道卷积函数代码图

1. 模型定义：先通过自定义卷积层，添加BatchNorm2d进行数据的归一化处理，这使得数据在进行ReLU之前不会因为数据过大而导致网络性能的不稳定，然后进行ReLU激活。然后经过平均池化层，再使用squeeze函数去掉含1的维度，最后经过一个全连接层输出。

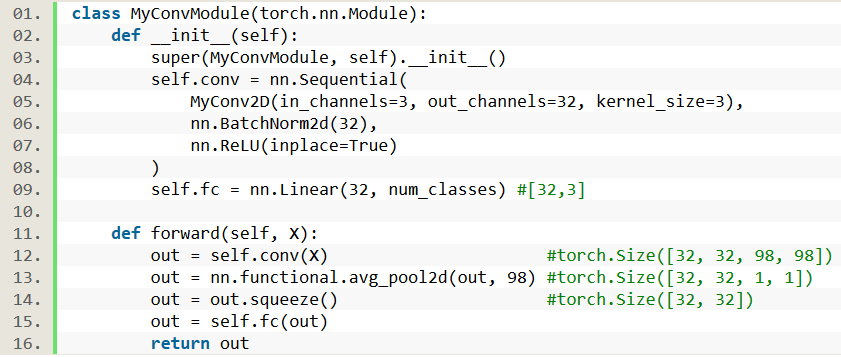


图4.1.5 模型定义代码图

##### 4.1.2 利用torch.nn实现

1. 模型定义：使用nn.Conv2d来实现一个一层卷积，输入通道数为图片的通道数3，输出通道数为32，卷积核大小为3\*3，步长为1，没有填充；然后经过BatchNorm2d进行归一化后再使用ReLU激活函数。由于图片输入为3\*100\*100，所以经过卷积层后是32\*98\*98，使用大小为98\*98的平均池化层变成32\*1\*1的输出。使用squeeze()函数将输出压缩，去掉1的维度，结果输出是32\*32，再经过一个32\*3的全连接层得到模型。

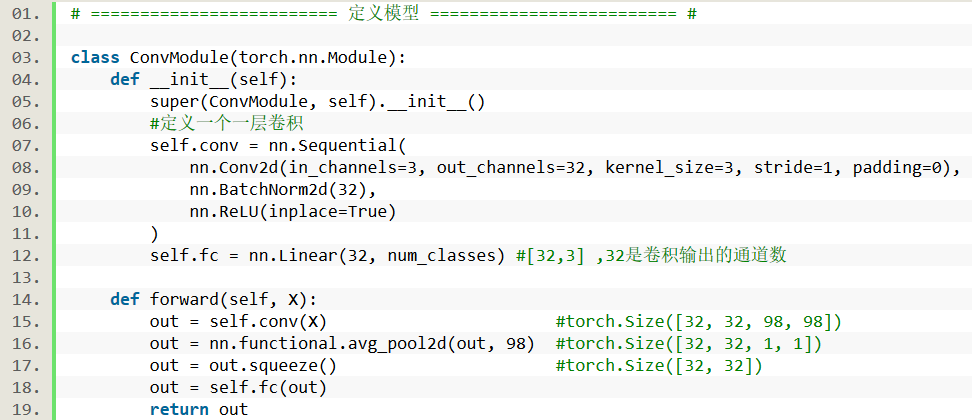


图4.1.6 模型定义代码图

##### 4.1.3 不同卷积层数对比

1. 模型定义：分别定义一个一层卷积，一个二层卷积，一个三层卷积。第一层卷积输入通道为3，输出通道为32，卷积核大小为3\*3，步长为1；第二层卷积输入通道为32，输出通道为64，卷积核大小为3\*3，步长为1；第三层卷积输入通道为64，输出通道为128，卷积核大小为3\*3，步长为1

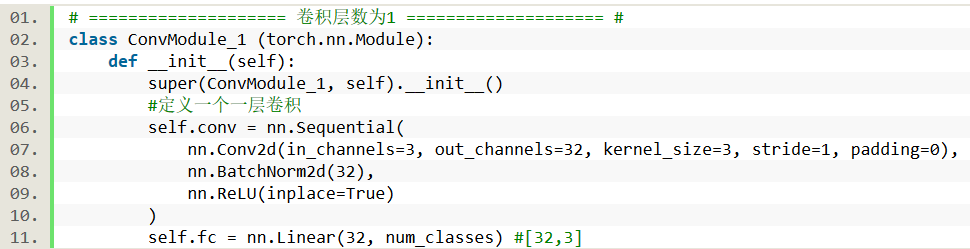


图4.1.7 一层卷积模型定义代码图

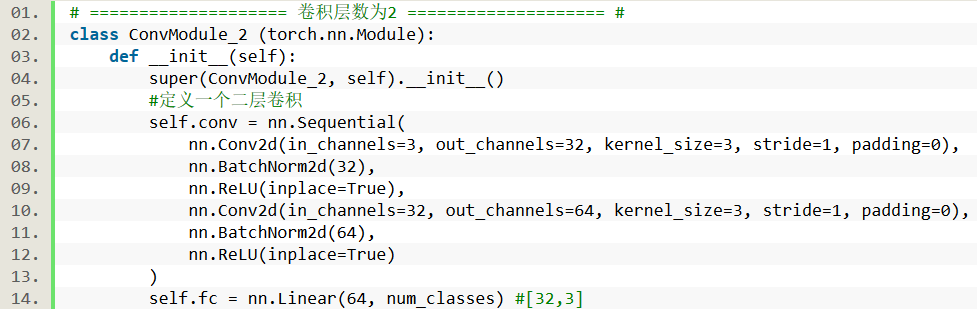


图4.1.8 二层卷积模型定义代码图

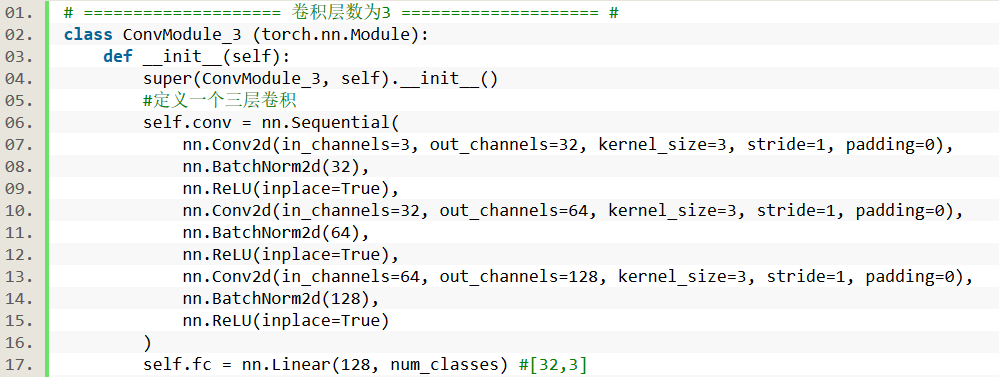


图4.1.9 三层卷积模型定义代码图

##### 4.1.4 不同卷积核大小对比

1. 卷积层定义：输入通道都为图片通道3，输出通道都为32，步长都为1，无填充，卷积核大小分别为3，5，7。

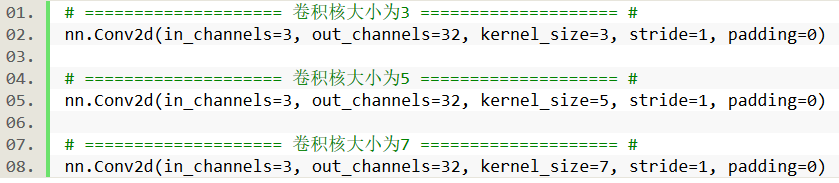


图4.1.10 不同卷积层模型定义代码图

##### 4.1.5 AlexNet实现

1. 模型定义：输入为100x100x3的图片，经过第一个卷积层：11x11的卷积和，输出通道为96，步长为4，3x3的最大池化，步长为2；再经过第二个卷积层：5x5的卷积和，输出通道为256，步长为1，3x3的最大池化，步长为2；经过第三个卷积层：3x3的卷积和，输出通道为384，步长为1；经过第四个卷积层：3x3的卷积和，输出通道为384，步长为1；然后经过两个全连接层，全连接层的输出维度为4096，并且都加入dropout，最后输出层。

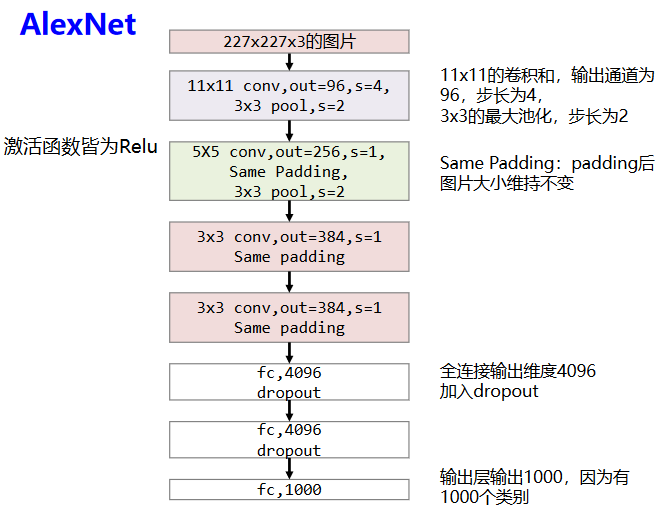


图4.1.11 AlexNet经典模型流程图



图4.1.12 AlexNet模型实现代码图

#### 4.2 空洞卷积实验

##### 4.2.1 利用torch.nn实现空洞卷积

1. 模型定义：定义一个三层空洞卷积，空洞率分别为[1, 2, 5]。由于普通空洞卷积会造成像素连接处的不连续性，重复堆叠会进一步加剧网格的形成，会造成细节丢失，所以我们采用HDC，通过设计锯齿波的空洞率序列达到感受野覆盖全图的目的。接下来验证该序列的合理性：

所以，目标是让（其中K是卷积核大小），所以，该组序列合理

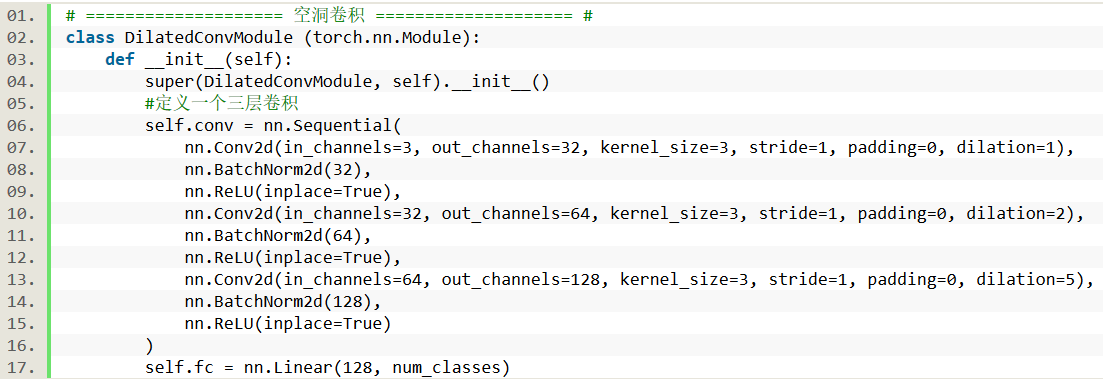


图4.2.1空洞卷积模型定义代码图

##### 4.2.2 空洞卷积与卷积模型对比

1. 模型定义：空洞卷积相当于在普通卷积的每一个卷积层上加上空洞率，这个空洞率是满足HDC的合理序列，由于输入输出通道不变，卷积核大小也不变，所以二者的池化层的维度也不变。图片输入为[3, 100, 100]，经过卷积层后维度变成[128, 84, 84]，经过池化层后变成[128, 1, 1]，经过squeeze压缩操作后变成[128]，再经过128个单元的全连接层。

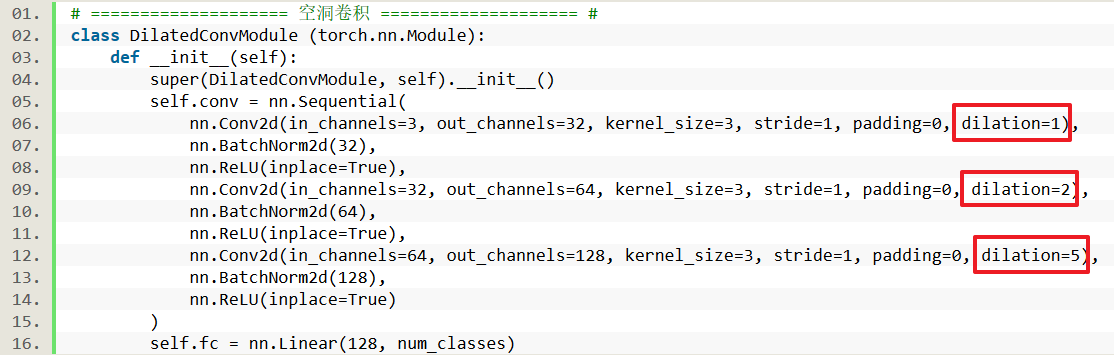


图4.2.2 空洞卷积模型定义代码图

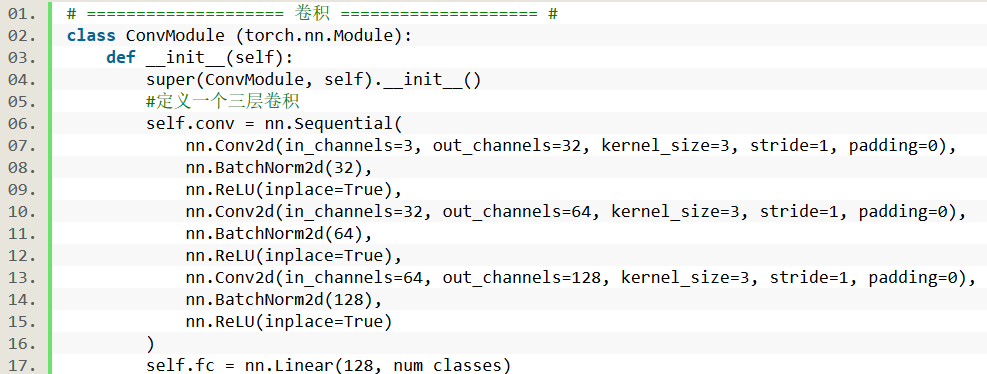


图4.2.3 卷积模型定义代码图

##### 4.2.3 不同HDC序列对比

1. 模型定义：4.2.1介绍了HDC序列组合的合理性判断，所以可以得到[1,2,3], [3,4,5], [1,2,5], [5,9,17], [1,2,5,9]都是合理的空洞卷积序列。故此选择[1,2,3], [3,4,5], [1,2,5]序列来进行模型对比。

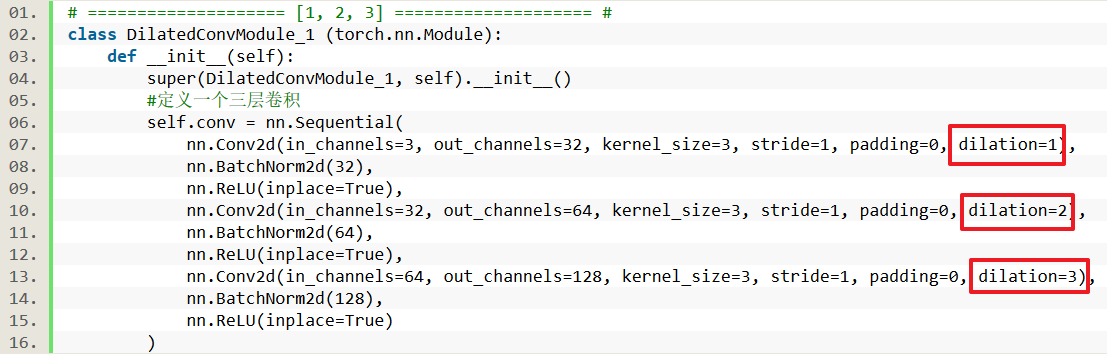


图4.2.4 序列[1,2,3]定义代码图

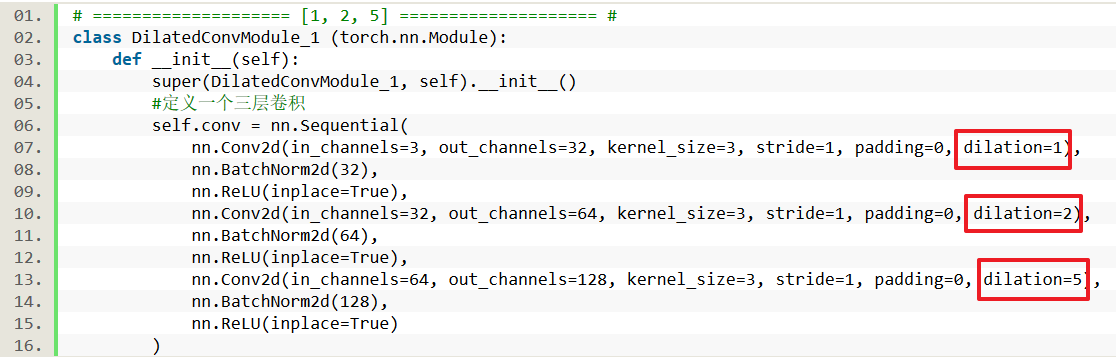


图4.2.5 序列[1,2,5]定义代码图

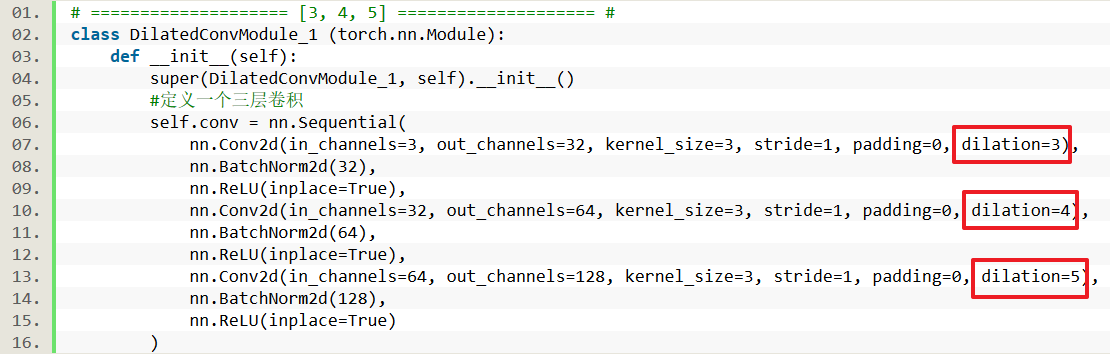


图4.2.6 序列[3,4,5]定义代码图

#### 4.3 残差网络实验

##### 4.3.1 实现给定结构的残差网络

1. 模型定义：图片的输入为[3, 100, 100]，经过第0层卷积层，将通道变成64，则输出图片为[64, 100, 100]，每经过一层卷积层（步长为1，填充为1），图片的size不变。所以在经过第一块（第1层卷积层和第2层卷积层）的时候，由于第0层的输出通道为64，而第2层的输出通道为64，二者channel维度相同，所以在shortcut的时候可以直接相加，第2层后图片为[64, 100, 100]。同理在第4层后图片维度为[64, 100, 100]。在第三块（第5层和第6层）结束后，由于步长是2，填充为1，所以图片的size要减半，再根据输出通道为128，所以第8层后图片维度为[128, 50, 50]。由于第4层的输出通道为64，而第6层的输出通道为128，所以不能直接相加，需要将第4层的维度更改成128，并且图片大小改成50才可以相加。

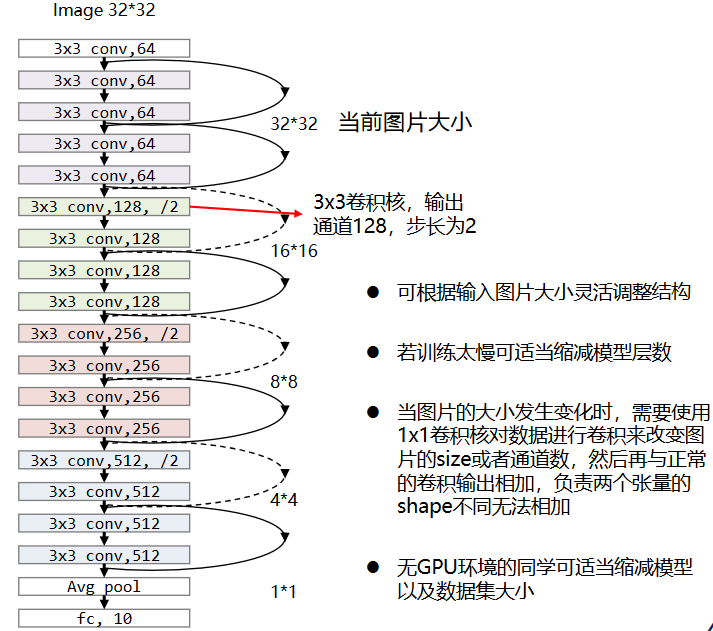


图4.3.1 ResNet经典模型流程图

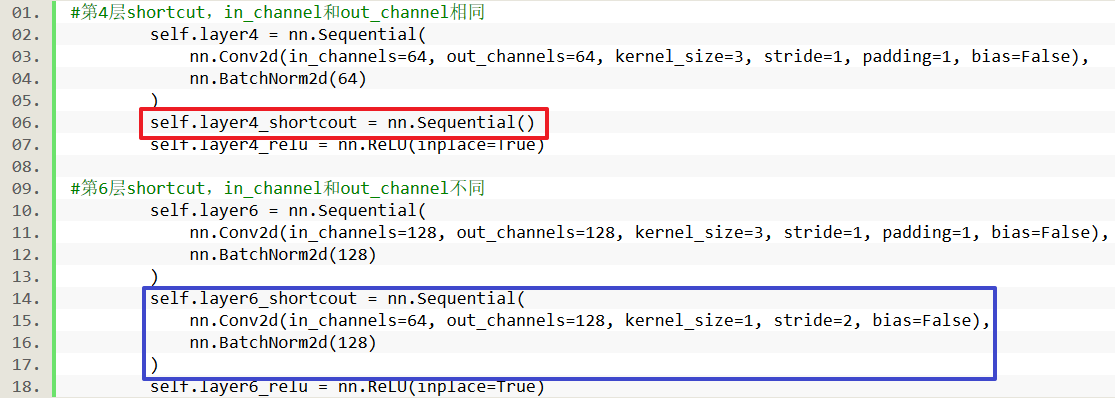


图4.3.2 ResNet经典模型部分代码图

##### 4.3.2 残差网络与空洞卷积结合

1. 模型定义：

图片的输入为[3, 100, 100]，经过第0层卷积层，将通道变成64，则输出图片为[64, 100, 100]。

第一块卷积（包含第一层，第二层，第三层，且空洞率分别为1，2，3）

经过第一层卷积层（步长为1，填充为1，空洞率为1），图片的size不变。经过第二层卷积层（步长为1，填充为2，空洞率为2），图片的size不变。经过第三层卷积层（步长为1，填充为3，空洞率为3），图片的size不变。由于第0层的输出通道为64，而第3层的输出通道为64，二者channel维度相同，所以在shortcut的时候可以直接相加，第三层后图片为[64, 100, 100]。

第二块卷积（包含第四层，第五层，第六层，且空洞率分别为1，2，3）

经过第四层卷积层（步长为2，填充为1，空洞率为1），图片的size减半。经过第五层卷积层（步长为1，填充为2，空洞率为2），图片的size不变。经过第六层卷积层（步长为1，填充为3，空洞率为3），图片的size不变。由于第3层的输出通道为64，而第6层的输出通道为128，二者channel维度不同，所以在shortcut的时候需要转换，第6层后图片为[128, 50, 50]。

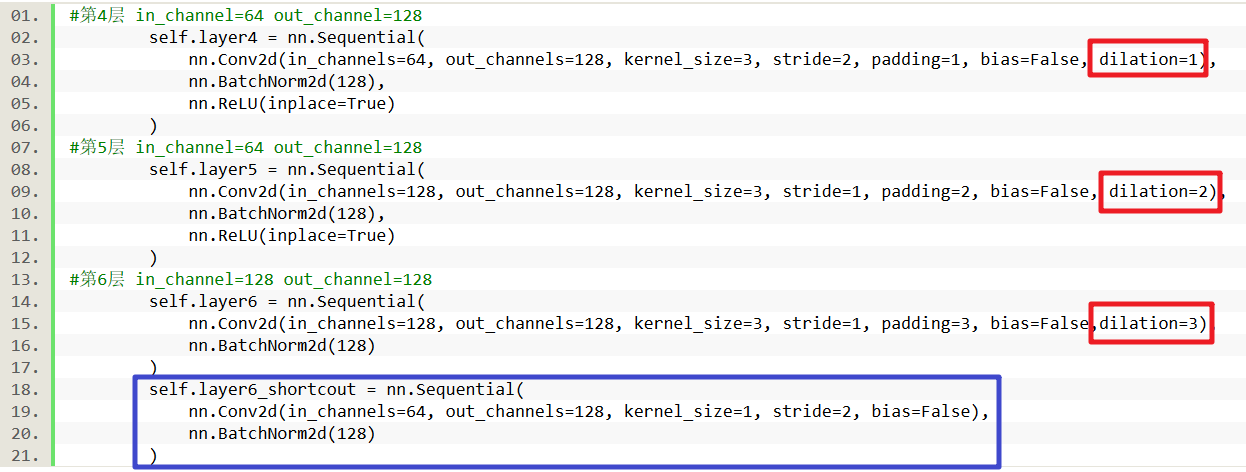


图4.3.3 残差与空洞卷积结合部分代码图

## 五、实验结果

#### 5.1 二维卷积实验

##### 手动实现二维卷积

由于手动实现二维卷积实验时间过长（一个batch\_size运行半小时，跑不完一个epoch），所以只输出3个batch的loss和acc用于实现，结果仅供参考，不具有太大的意义。3个batch训练时间：1小时30分钟。

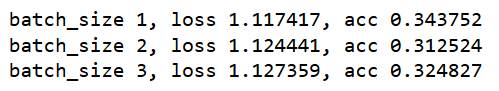


图5.1.1 手动实现二维卷积结果图

##### 5.1.2 利用torch.nn实现

由loss对比可以得到，训练集loss在稳定下降，测试集loss在波动下降，在100个epoch的时候训练集loss小于测试集loss；由acc对比可以得到，训练集acc在稳定上升，测试集loss在波动上升，在100个epoch的时候训练集acc大于测试集acc。造成测试集结果波动的原因可能是batch\_size=128设置得太小，一定范围内，batch\_size越大，其确定的下降方向就越准，震荡就越小。

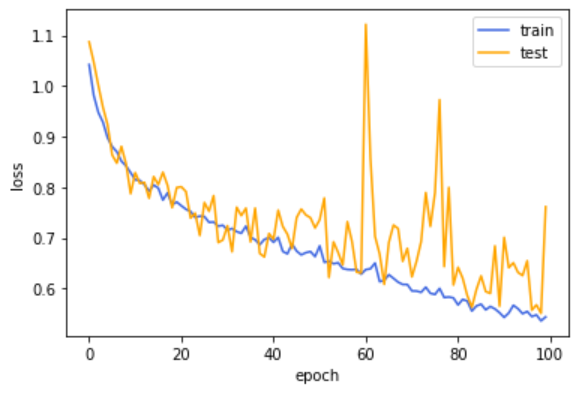


图5.1.2 训练集和测试集loss对比结果图

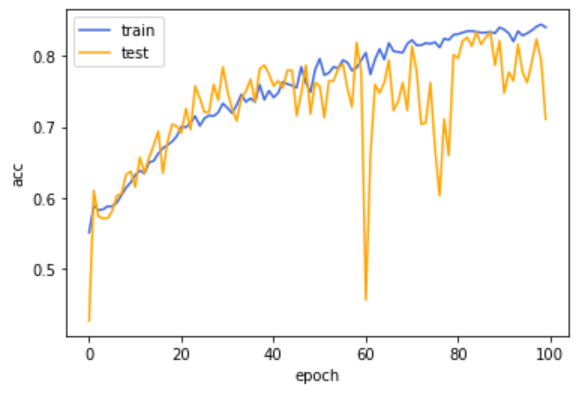


图5.1.3 训练集和测试集acc对比结果图

100个epoch的执行时间为7m57s，远远高于手动实现的效率。



图5.1.4 利用torch.nn实现训练时间图

##### 5.1.3 不同卷积层数对比

从训练集loss可看出，三层卷积的效果最好。三层卷积的loss一开始是最低，并且在100个epoch之后loss也是三者之中最低的；其次是二层卷积，最后是一层卷积。

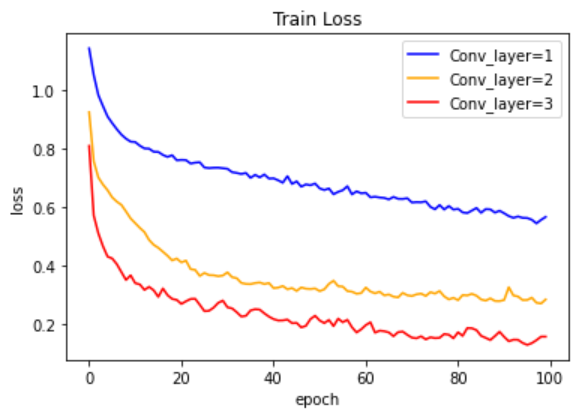


图5.1.5 不同卷积层数训练集loss结果图

从测试集loss可看出，三层卷积的效果最好。对于稳定性来说，是一层卷积的效果最好，但是对于最终结果来说，是三层卷积的效果最好。

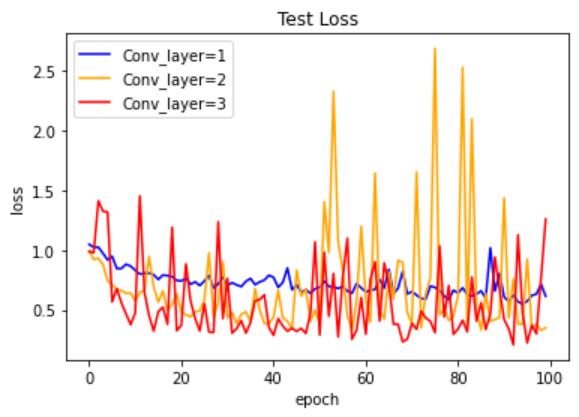


图5.1.6 不同卷积层数测试集loss结果图

从训练集acc可看出，三层卷积的效果最好。三层卷积的acc一开始是最高，并且在100个epoch之后acc也是三者之中最高的；其次是二层卷积，最后是一层卷积。三层卷积的训练集准确率最高能达到96%，二层卷积的训练集准确率最高能达到90%，而一层卷积的训练集准确率最高仅能达到83%。

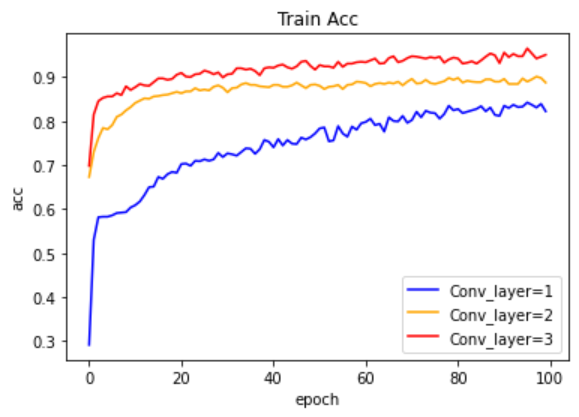


图5.1.7 不同卷积层数训练集acc结果图

从测试集acc可看出，三层卷积的效果最好。对于稳定性来说，是一层卷积的效果最好，但是对于最终结果来说，是三层卷积的效果最好。二层卷积的波动率太大。三层卷积的测试集准确率最高能达到91%，二层卷积的测试集准确率最高能达到89%，而一层卷积的测试集准确率最高仅能达到83%。

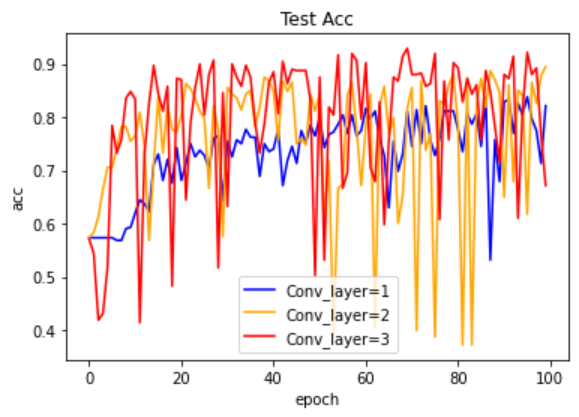


图5.1.8 不同卷积层数测试集acc结果图

由下图可看出层数越高，层数越多，执行时间越长。两层卷积的执行时间是一层卷积的两倍，三层卷积的执行时间又是两层卷积的两倍。



图5.1.9 不同卷积层数执行时间结果图

综上所述，三层卷积的效果最好，且层数越高，执行时间越长。

##### 5.1.4 不同卷积核大小对比

从训练集loss可看出，卷积核大小为5\*5和7\*7的效果最好。卷积核大小为5\*5的loss一开始是最低，并且在100个epoch之后loss也是三者之中最低的；其次是卷积核大小为7\*7，卷积核大小为3\*3。

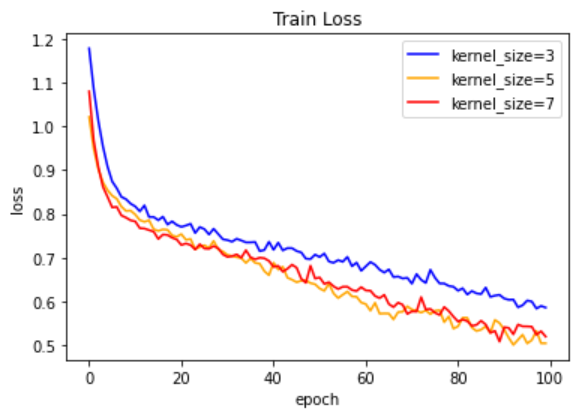


图5.1.10 不同卷积核大小训练集loss结果图

从测试集loss可看出，卷积核大小为7\*7的效果最好。对于稳定性来说，是卷积核大小为3\*3的效果最好，但是对于最终结果来说，是卷积核大小为5\*5的效果最好，卷积核大小为7\*7的loss波动比较大。

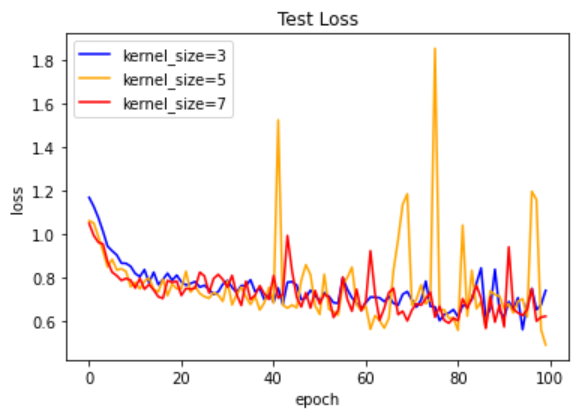


图5.1.11 不同卷积核大小测试集loss结果图

从训练集acc可看出，卷积核大小为5\*5和7\*7的效果最好。卷积核大小为5\*5的acc一开始是最高，并且在100个epoch之后acc也是三者之中最高的；其次是卷积核大小为7\*7，最后是卷积核大小为3\*3。在100个epoch之后三者的准确率差不多，都在83%左右

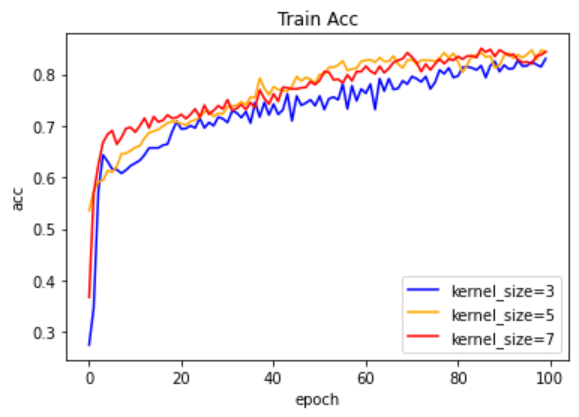


图5.1.12 不同卷积层数训练集acc结果图

从测试集acc可看出，卷积核大小为7\*7的效果最好。对于稳定性来说，是卷积核大小为3\*3的效果最好，但是对于最终结果来说，是卷积核大小为7\*7的效果最好。卷积核大小为5\*5的波动率太大。三者的测试集acc最高都能达到84%左右。

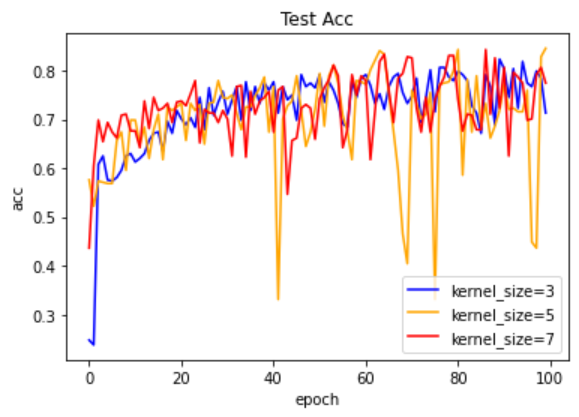


图5.1.13 不同卷积核大小测试集acc结果图

在时间对比上，不管卷积核大小是3，5还是7，训练时间都差别不大，都在30s左右就可执行好。

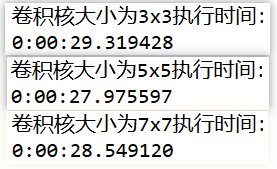


图5.1.14 不同卷积核大小执行时间结果图

综上所述，卷积核大小为7的效果最好。

##### 5.1.5 AlexNet实现

由Loss结果图可得到训练集loss一直在下降并趋于稳定，但是测试集loss在下降一段时间后波动回升，这是明显的过拟合现象。产生过拟合的原因是因为模型层数太多，模型过于复杂，但是数据集样本数太少（车辆训练集仅为950，测试集仅为407）。

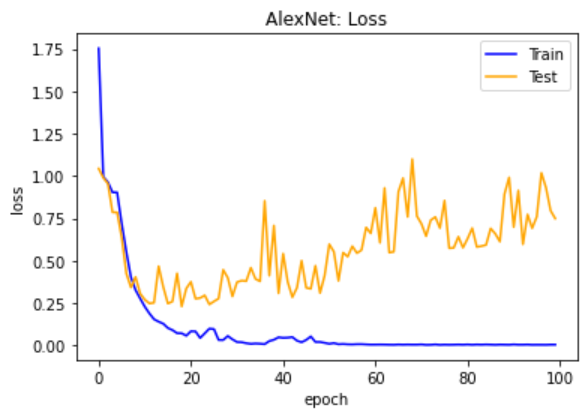


图5.1.14 AlexNet训练集和测试集loss结果图

由Acc结果图可得到训练集acc效果好且准确率最高能达到99%，在测试集上效果没有训练集好，准确率最高能达到93%并趋于稳定。测试集的acc低于训练集的acc说明该模型的泛化能力还需要提升。

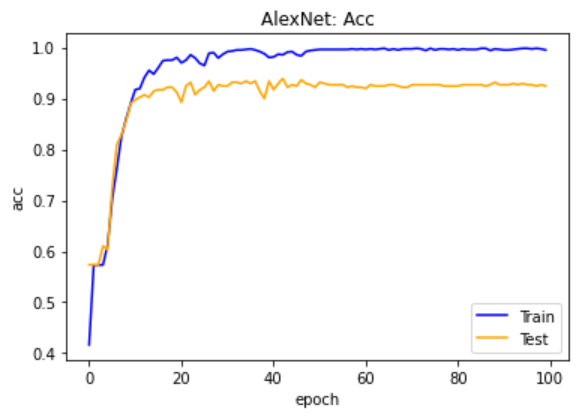


图5.1.14 AlexNet训练集和测试集acc结果图

模型运行时间为14m49s，运行时间较长



#### 5.2 空洞卷积实验

##### 5.2.1 利用torch.nn实现空洞卷积

由loss对比可以得到，训练集loss小幅度稳定下降，测试集loss一开始波动幅度较大，在30epoch之后虽然有小幅度波动，但总体趋于稳定，说明模型训练得还不错。

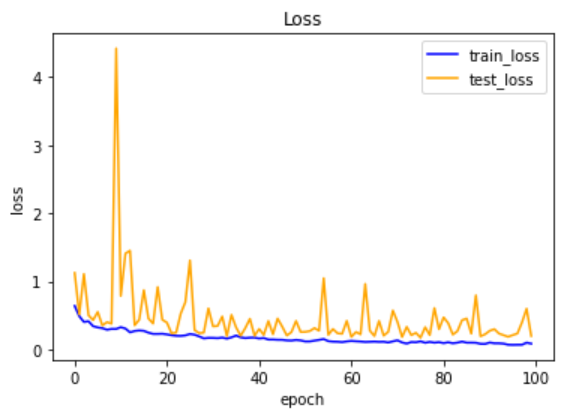


图5.2.1 空洞卷积实现loss结果图

由acc对比可以得到，训练集acc小幅度稳定上升，测试集acc一开始波动幅度较大，后面有小幅度波动，说明模型泛化能力还不够。训练集的准确率最高在97%，测试集的准确率最高在94%。

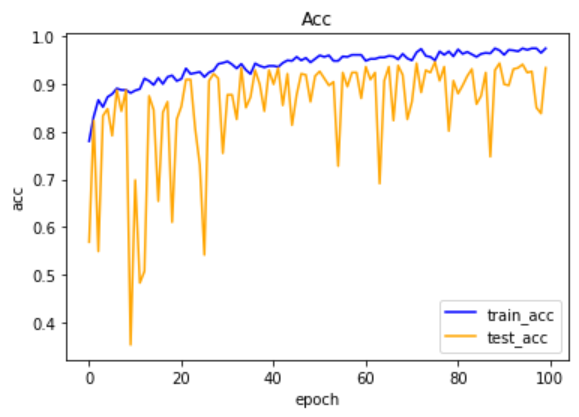


图5.2.2 空洞卷积实现acc结果图

空洞卷积的训练时间为10m19s，高于普通卷积的7m57s，二者相差2分钟



图5.2.3 空洞卷积实现训练时间结果图

##### 5.2.2 空洞卷积与卷积模型对比

因为空洞卷积模型要比普通卷积复杂，一开始普通卷积的训练集loss要低于空洞卷积的训练集loss，但是在5个epoch之后空洞卷积的loss就要普遍低于普通卷积，二者的loss都在小幅度波动中稳定下降。总体来看，空洞卷积的训练效果要比普通卷积要好。

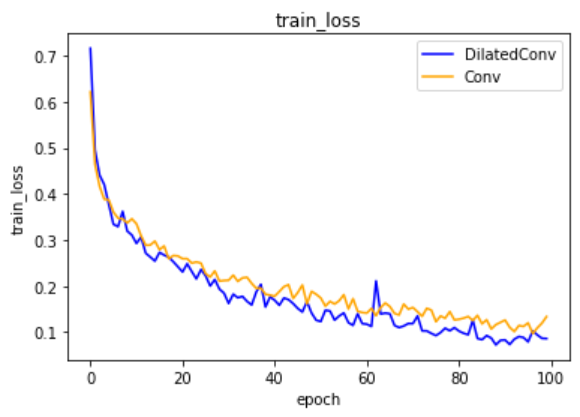


图5.2.4 空洞卷积与卷积训练集loss对比结果图

二者的测试集loss波动明显且基本持平不下降，说明模型的超参数设置得不好，模型泛化能力不够，出现了过拟合的现象。由于测试集样本太小，所以震荡现象较为明显。空洞卷积的震荡幅度要小于普通卷积，说明空洞卷积在一定程度上缓解了过拟合。

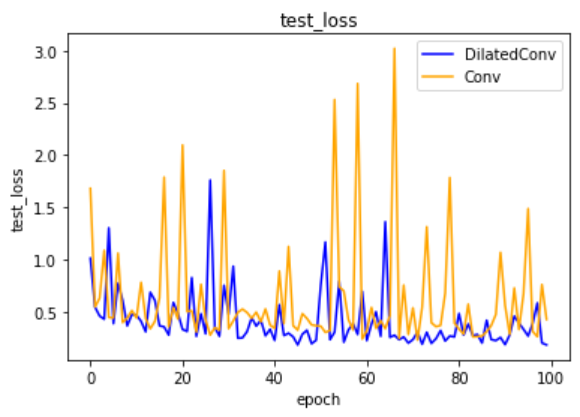


图5.2.5 空洞卷积与卷积测试集loss对比结果图

因为空洞卷积模型要比普通卷积复杂，一开始普通卷积的训练集acc要高于空洞卷积的训练集acc，但是在5个epoch之后空洞卷积的acc就要普遍高于普通卷积，二者的acc都在小幅度波动中上升。总体来看，空洞卷积的训练效果要比普通卷积要好。空洞卷积的准确率最高能达到97%，普通卷积的准确率最高能达到96%，说明训练集的训练效果是很好的。

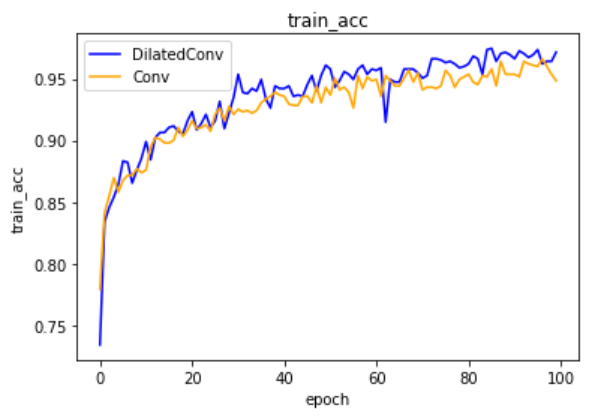


图5.2.6 空洞卷积与卷积训练集acc对比结果图

二者的测试集acc波动明显但总体小幅度上升，说明模型的超参数设置得不好，模型泛化能力不够，出现了过拟合的现象。由于测试集样本太小，所以震荡现象较为明显。空洞卷积的震荡幅度要小于普通卷积，说明空洞卷积在一定程度上缓解了过拟合。空洞卷积的准确率最高能达到94%，普通卷积的准确率最高能达到93%。

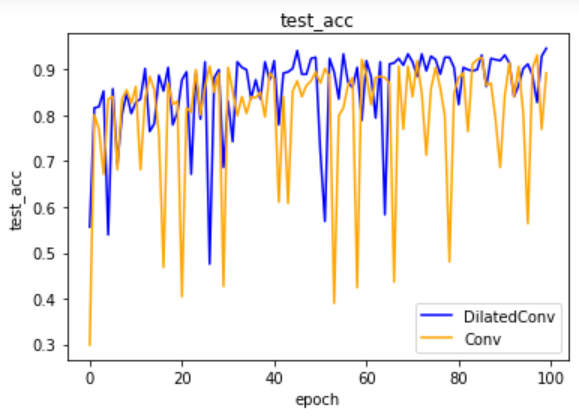


图5.2.7 空洞卷积与卷积测试集acc对比结果图

因为空洞卷积模型要比普通卷积复杂，所以空洞卷积的训练时间要远远多于普通卷积的执行时间，空洞卷积训练时间为10m19s，卷积训练时间为2m4s，空洞卷积100个epoch的训练时间是卷积100个epoch训练时间的5倍。

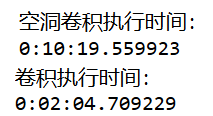


图5.2.8 空洞卷积与卷积训练时间对比结果图

##### 5.2.3 不同HDC序列对比

由下图可以得到，序列[3, 4, 5]的训练效果最好。一开始的loss是序列[1, 2, 5]最低，但是在2个epoch之后序列[3, 4, 5]就是最低，序列[1, 2, 3]的效果不如序列[1, 2, 3]和序列[3, 4, 5]。

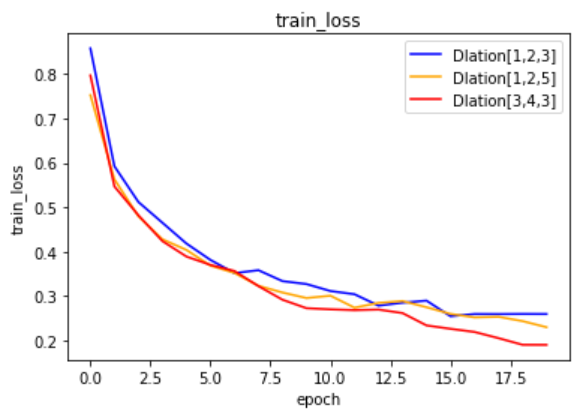


图5.2.9 不同HDC序列训练集loss对比结果图

由于测试集样本太小，所以震荡现象较为明显，但大体上是呈现下降的趋势。由测试集loss图可以看出序列[3, 4, 5]普遍较低，序列[1, 2, 3]的效果在开始的5个epoch比较好，5个epoch之后效果就不如序列[3, 4, 5]。

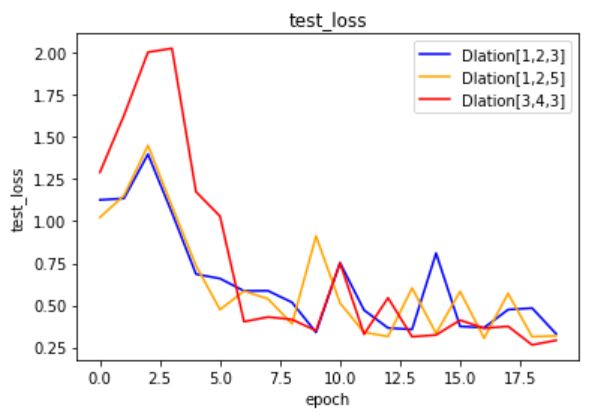


图5.2.10 不同HDC序列测试集loss对比结果图

由下图可以得到，序列[3, 4, 5]的训练效果最好，其次是序列[1, 2, 5]，最后是序列[1, 2, 3]。一开始的准确率比较高的是序列[1, 2, 5]，然后逐渐被序列[3, 4, 5]超过。序列[3, 4, 5]的训练集准确率最高在94%，序列[1, 2, 5] 的训练集准确率最高在92%，序列[1, 2, 3] 的训练集准确率最高在91%。

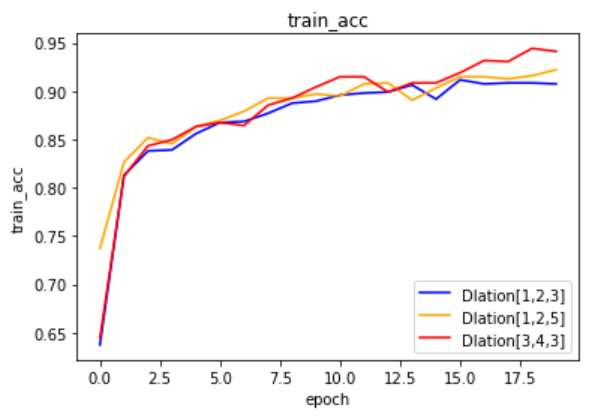


图5.2.11 不同HDC序列训练集acc对比结果图

由下图可以得到，序列[3, 4, 5]的训练效果最好，其次是序列[1, 2, 5]，最后是序列[1, 2, 3]。一开始的准确率比较高的是序列[1, 2, 5]，然后逐渐被序列[3, 4, 5]超过。序列[3, 4, 5]的训练集准确率最高在90%，序列[1, 2, 5] 的训练集准确率最高在90%，序列[1, 2, 3] 的训练集准确率最高在88%。准确率有波动说明应该在后期训练时适当降低学习率等超参数。

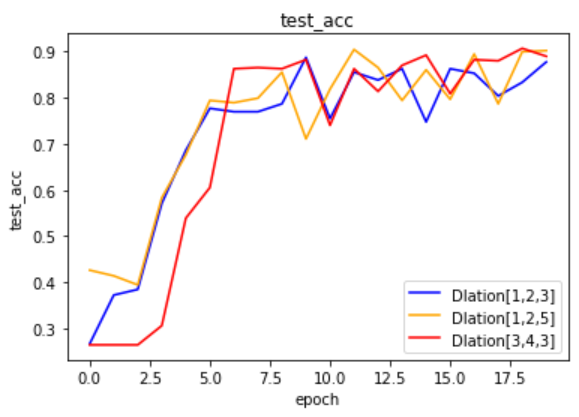


图5.2.12 不同HDC序列测试集acc对比结果图

由下图可以得到，不同HDC序列的训练时间相差无几，因为模型大小相同，复杂度相同，所以训练时间相差不大。

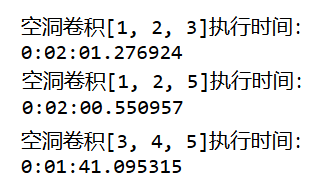


图5.2.13 不同HDC序列训练时间对比结果图

#### 5.3 残差网络实验

##### 5.3.1 实现给定结构的残差网络

由于有18层卷积，模型过于复杂，但是数据集样本数太少，导致测试集的loss有频繁波动，但是总体上来看，模型训练效果较好。

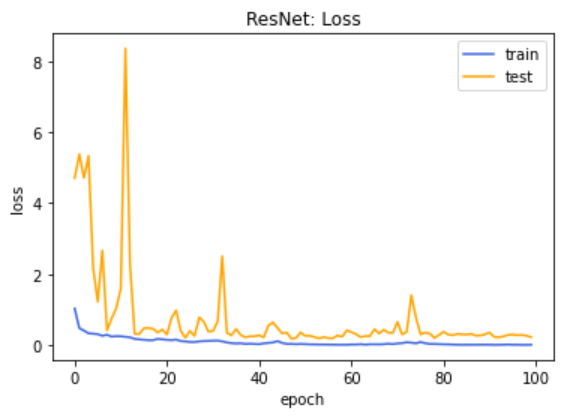


图5.3.1 给定结构残差网络loss结果图

由于有18层卷积，模型过于复杂，但是数据集样本数太少，导致测试集的acc有频繁波动，但是总体上来看，模型训练效果较好。训练集的准确率在100个epoch之后能够达到99%，测试集的准确率在100个epoch之后最高能达到94%。

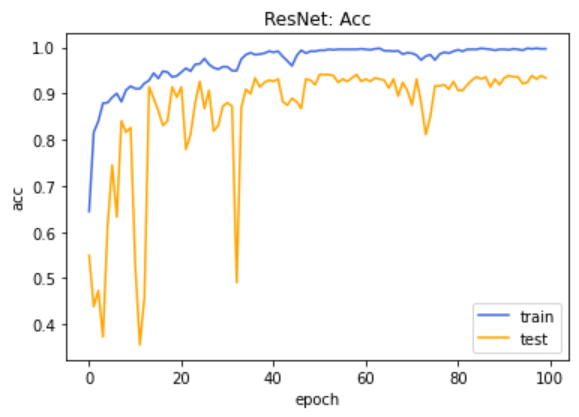


图5.3.2 给定结构残差网络acc结果图

18层的残差网络在GPU下跑100个epoch训练时间为6m47s，是3层的空洞卷积训练时间的两倍。由于残差连接使得信息前后向传播更加顺畅，所以即使模型有18层但丝毫不影响速度。



图5.3.3 给定结构残差网络训练时间结果图

##### 5.3.2 残差网络与空洞卷积结合

由于残差网络的卷积层有6层，且每一层都加入了空洞卷积（空洞率分为[1, 2, 3, 1, 2, 3]），所以网络较为复杂，测试集的loss有频繁波动，但训练效果好。

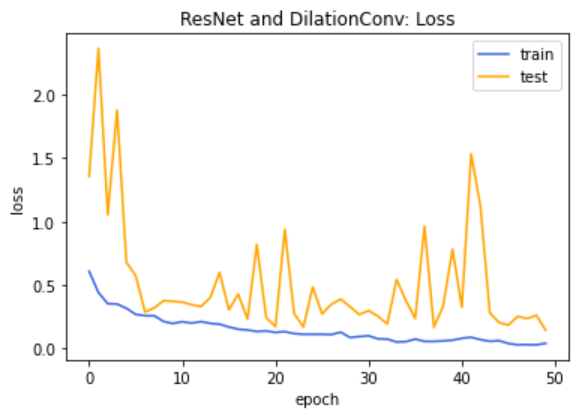


图5.3.4 残差网络与空洞卷积结合loss结果图

由于残差网络的卷积层有6层，且每一层都加入了空洞卷积（空洞率分为[1, 2, 3, 1, 2, 3]），所以网络较为复杂，但是数据集样本数太少，导致测试集的acc有频繁波动，但是总体上来看，模型训练效果较好。训练集的准确率在50个epoch之后能够达到99%，测试集的准确率在50个epoch之后最高能达到96%。

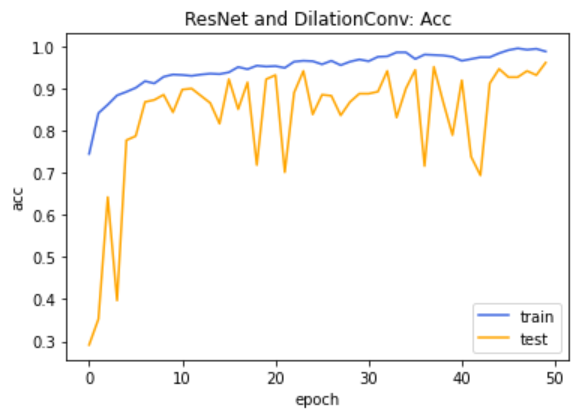


图5.3.5 残差网络与空洞卷积结合acc结果图

6层的残差网络与空洞卷积结合在GPU下跑50个epoch训练时间为9m36s，相比起仅仅使用18层的残差网络的6m47s来说要久一点，主要是因为加入了空洞率元素导致运行时间变长。



图5.3.3 残差网络与空洞卷积结合训练时间结果图

## 六、实验心得体会

1. BatchNorm2d层的作用：添加BatchNorm2d进行数据的归一化处理，这使得数据在进行ReLU激活之前不会因为数据过大而导致网络性能的不稳定。
2. torch.squeeze这个函数主要对数据的维度进行压缩，，去掉维数为1的的维度

torch.unsqueeze()这个函数主要是对数据维度进行扩充。给指定位置加上维数为一的维度

1. HDC序列的合理性验证：假设，卷积核大小K为3

所以，目标是让（其中K是卷积核大小），所以，该组序列合理。所以 [1,2,3], [3,4,5], [1,2,5], [5,9,17], [1,2,5,9]都是合理的空洞卷积序列。

1. 错误提示： one of the variables needed for gradient computation has been modified by an inplace operation

错误原因：某个变量在前向计算时是一个值，在求梯度时变成了另一个值（inplace 操作导致），使得pytorch在反向梯度求导时产生了错误。

解决方法：残差网络中tranceback报错时只提示loss.backward()这一行产生了错误，导致很难debug，用 torch.autograd.set\_detect\_anomaly(True) 可回溯问题语句。然后替换所有的替换所有的inplace操作：x += 1 改成 x = x + 1，也就是将out3 += self.layer3\_shortcout(out0)（inplace操作）改成out3 = out3 + self.layer3\_shortcout(out0)（不是inplace操作）

## 七、参考文献

无

## 八、附录

略。

**实验报告编写要求**

1. 正文要求小四号宋体，行间距1.5倍；

2. 英文要求小四号Times New Roman；

3. 在实验内容、实验过程、实验结果三部分需要针对当次实验不同的实验内容分别填写（模版以实验一为例），实验设计中如有必要也可以分开填写；

4. 实验报告配图的每幅图应有编号和标题，编号和标题应位于图下方处，居中，中文用五号宋体；

5. 表格应为三线表，每个表格应有编号和标题，编号和标题应写在表格上方正中，距正文段前0.5倍行距。表格中量与单位之间用“/”分隔，编号与标题中的中文用五号宋体；

6. 图、表、公式、算式等，一律用阿拉伯数字分别依序连续编排序号。其标注形式应便于互相区别，可分别为：图1、表2、公式(5)等。