**《医学课程项目》小作业实验报告**

**组员：薛峥嵘 519030910349**

**王资 519030910345**

**一、作业目标**

1.熟悉医学图片的构成

2.熟悉神经网络的搭建

3.掌握对医学图片的分类方法

**二、作业要求**

1.按照给出的代码示例，搭建ResNet-18与ResNet-50网络

2.阐述ResNet-18与ResNet-50在分类结果上的差异，并分析差异的原因

3.分析模型输入分辨率对结果产生影响的原因

4.对如何提高分类效果，提出自己的改进方案

**三、数据集简介**

MedMNIST是10类开源医学数据集。该数据集已经做好了预处理，包括：

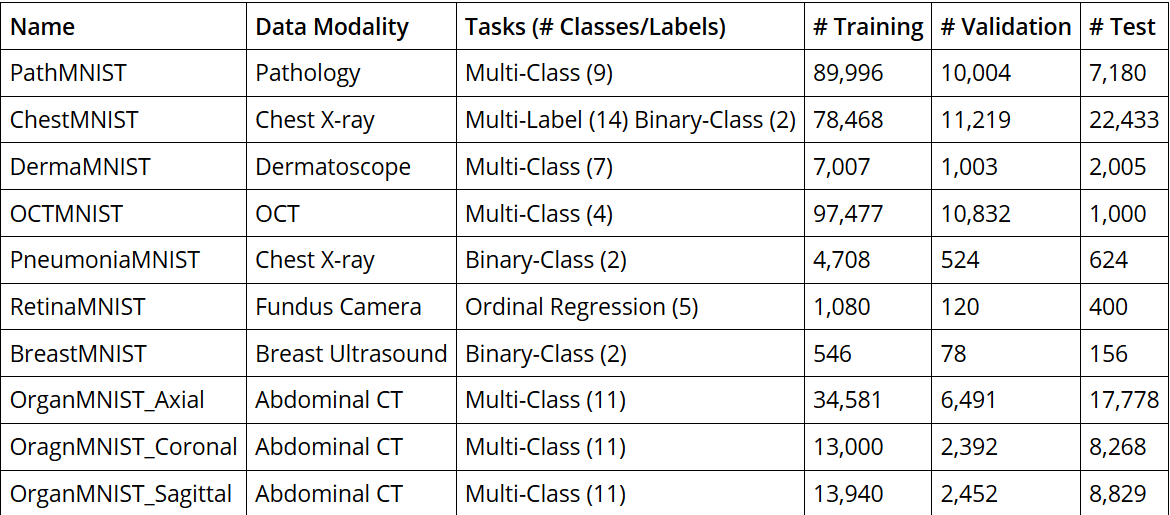
1.已经转换为28\*28的轻量级，方便处理

2.数据规模从100到100000，分类任务包括二分类、多分类、回归。

3.已经做了一些医学层面的预处理，使得参与该项目的人不需要有任何医学相关的背景知识。

MedMNIST数据集总共分成十种：PathMNIST、ChestMNIST、DermaMNIST、OCTMNIST、PneumoniaMNIST、RetinaMNIST、BreastMNIST、OrganMNIST{Axial, Coronal, Sagittal}

各种数据集的相关信息如下表：

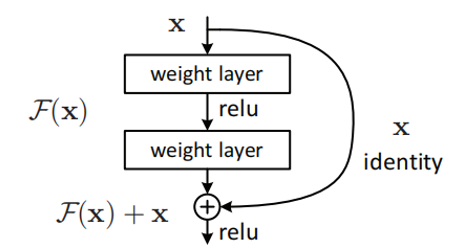


**四、使用模型介绍**

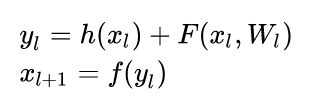
**1.Resnet架构**

**1.1简介**

Resnet的主要贡献是发现了神经网络的退化现象，并针对退化现象提出了shortcut connection，极大的消除了深度过大的神经网络的训练困难问题。神经网络得深度首次突破100层。

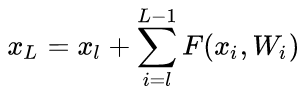


如图所示即为残差单元的模型，可以表示为：

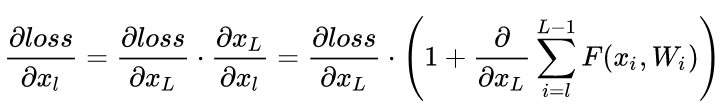


其中，和分别表示第l个残差块的输入和输出，事实上每个残差单元一般包含多层结构。F是残差函数，h表示恒等映射，f是激活函数。

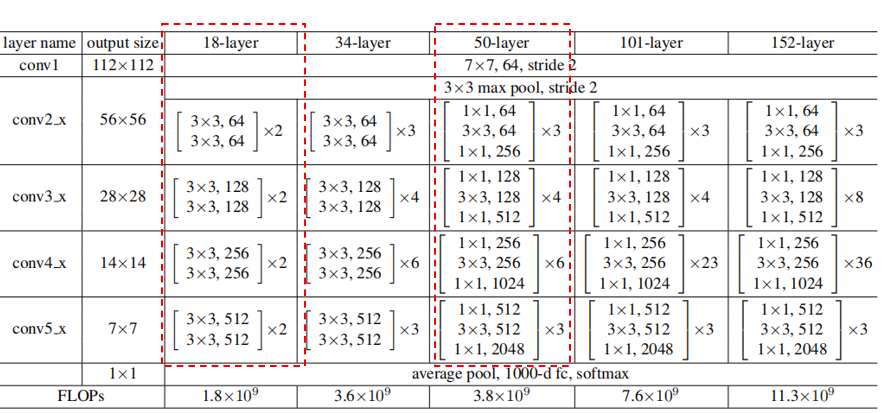
则从浅层l到深层L的学习特征为：



利用链式法则，可以求导得到反向过程的梯度。



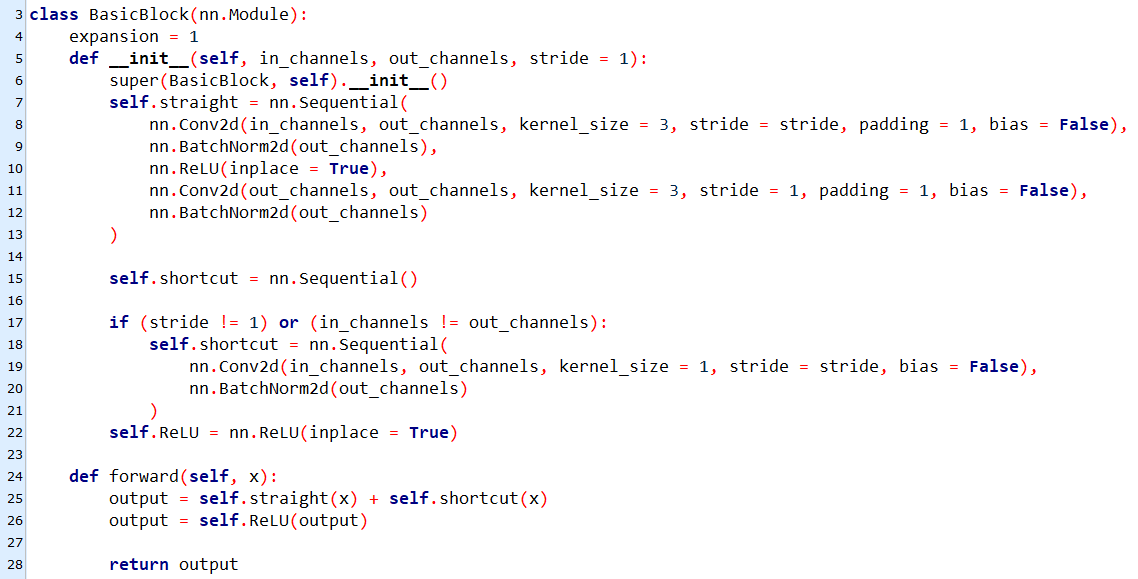
不同resnet如下表：

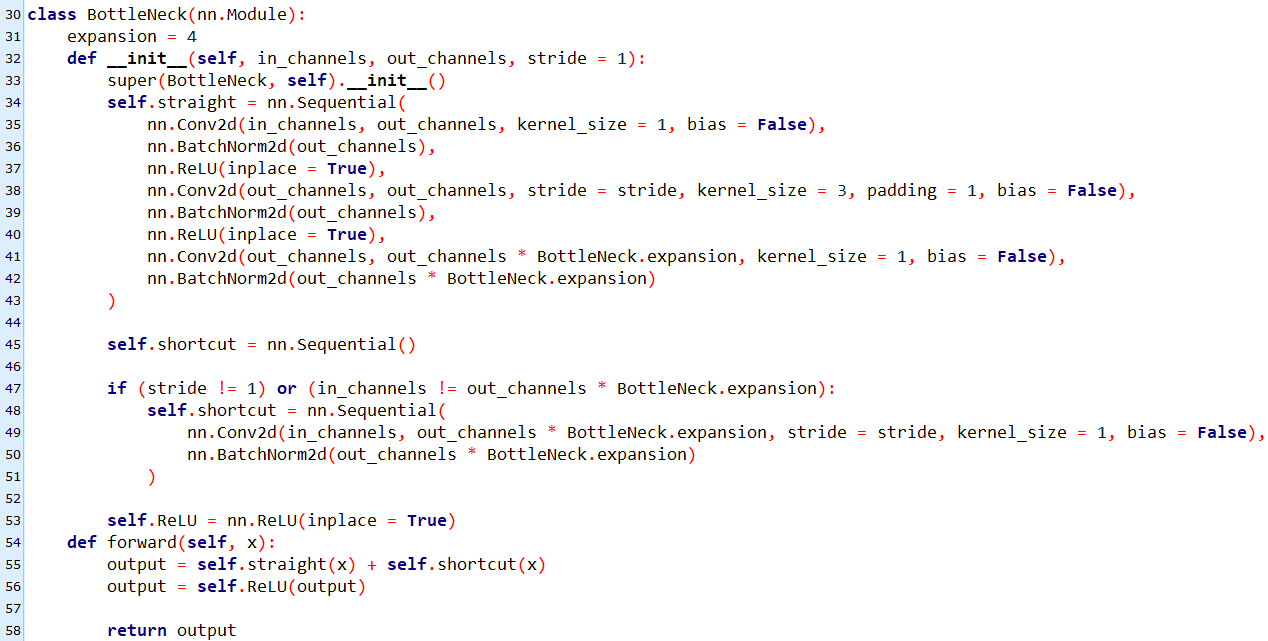


在本任务中，主要使用resnet18和resnet50。

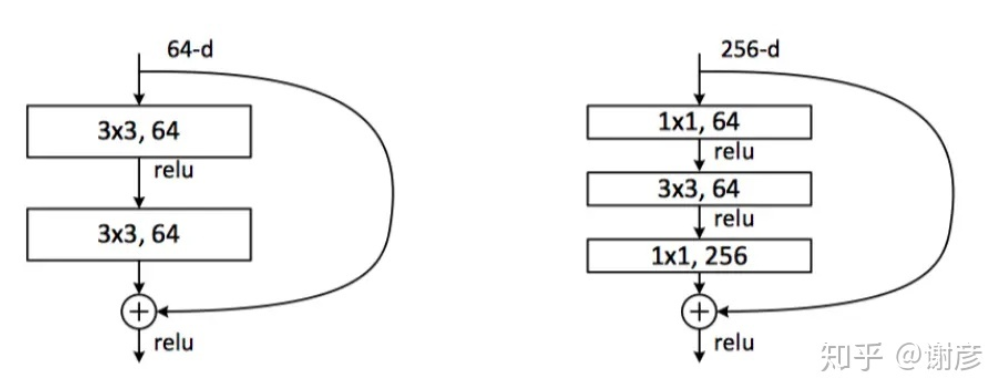
**1.2搭建过程**

实现resnet18和resnet50，首先需要搭建两个block，分别是BasicBlock和BottleNeck



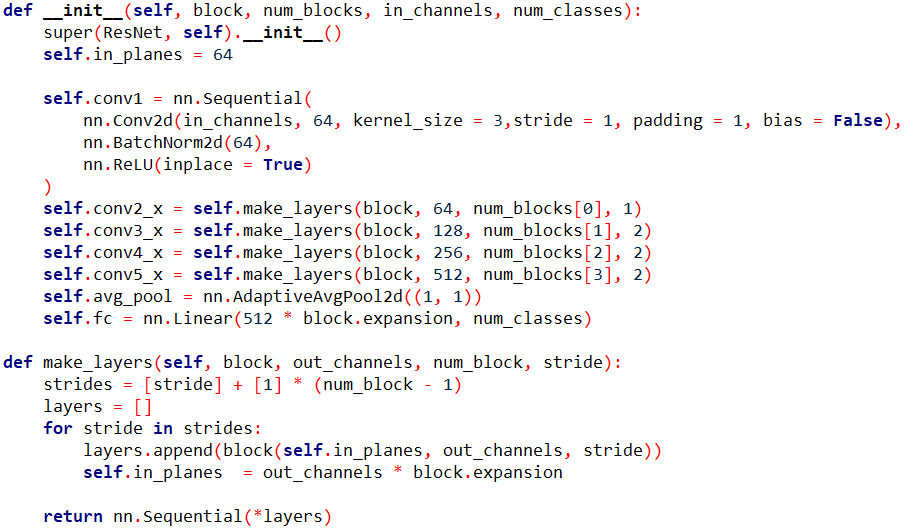


整体思路很类似，都是先构造上述的shortcut connection模块，以便重复利用此模块。但两者的区别在于：



左边的BasicBlock包含两个3\*3的卷积层，右图的BottleNeck包括了三个卷积层，第一个1\*1的卷积层用于降维，第二个3\*3层用于处理，第三个1\*1卷积层用于升维。这样减少了计算量。BasicBlock主要用于resnet18和resnet34，剩下更深的resnet都会利用BottleNeck结构。

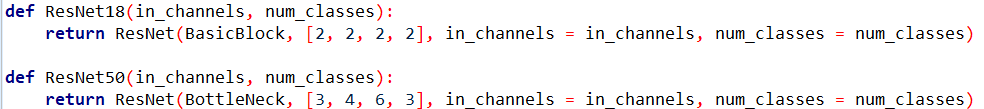
然后按表中所示层顺序搭建resnet



这里按原论文的结构是先做了7\*7的卷积，再做了stride为2的maxpool，但由于28\*28的图像本身就比较小，如果按原结构做的话就在第一层直接缩小了四倍，可能损失了不少信息，得不偿失，所以这里只做了一次加了padding之后的卷积。图像大小不变，但通过卷积操作提取出了一部分特征。接着就是调用make\_layers函数，根据BasicBlock和BottleNeck类制作出整个resnet。

前传操作就是把定义的各层都调用一遍，这里不需要赘述。

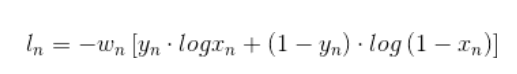
按照如下定义即可调用不同的resnet。



**2.loss函数**

Baseline中主要用了两种loss函数，BCEwithlogitsLoss和CrossEntropyLoss.

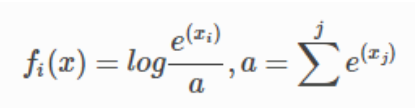
对于BECwithlogitsLoss，输入参数是(predict\_result, ground\_truth)，其作用是将predict\_result先通过sigmoid映射到0~1之间的值，再通过公式



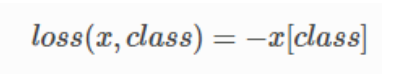
逐元素计算Loss，最后求和。

对于CrossEntropyLoss，在pytorch的设定中，该函数的主要作用是将logsoftmax-NLLLoss合并在一起得到的结果。将softmax之后的结果取Log，减少了计算量，同时保障了函数的单调性。

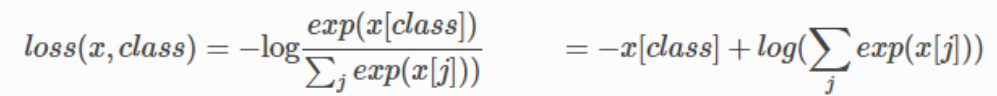
其中logsoftmax的定义如下：



NLLLoss的定义如下：



Pytorch中CrossEntropyLoss的定义如下：



输入为(input, target)，其中input为模型的输出，包含每个类的得分。Shape为batch\*n类，target的大小为n，包含类别的索引（不是one-hot编码的形式）。

**3.不同模型产生差异的原因**

Resnet18和Resnet50对比，Resnet18的网络层数更少，那么其表征能力注定比不上50的表征能力。有了shortcut connection之后，在resnet中几乎更深的网络表征能力、泛化能力更强。所以观察到训练相同epoch数量的时候，resnet50的效果往往更好一些。但也正是由于Resnet50的网络更加庞大，所以训练时间大约是Resnet18的两到三倍，这也是大型网络的一个劣势。

**4.改进方案**

**4.1 epoch数**

我们发现epoch==100的时候，最优的auc往往在比较低的epoch数时找到，说明出现了过拟合的现象。所以我们减少了epoch数，Resnet18大约训练50轮，Resnet50大约训练10~20轮即可到达很好的效果，不会过拟合。

**4.2数据增强（对比度，随机旋转）**

观察本数据集，发现有的数据集对比度很低，基本颜色相近。所以我们使用了torchvision.transforms.ColorJitter进行调整，主要改善了数据集中的亮暗、对比度。

另外，有些数据集的样本比较少，如BreastMNIST，所以需要做一些随机旋转的操作来间接增大数据量。但这里我们做的是随机水平旋转，因为医学图像本身的特性不能被改变。如果随机竖直旋转，训练出来的结果虽然能提升一定的泛化能力，但事实上这样的数据是无效的，所以做法不可取。这里调用的是torchvision.transforms.RandomVerticalFlip函数。

总之，数据增强操作能够提高整个模型的泛化能力。

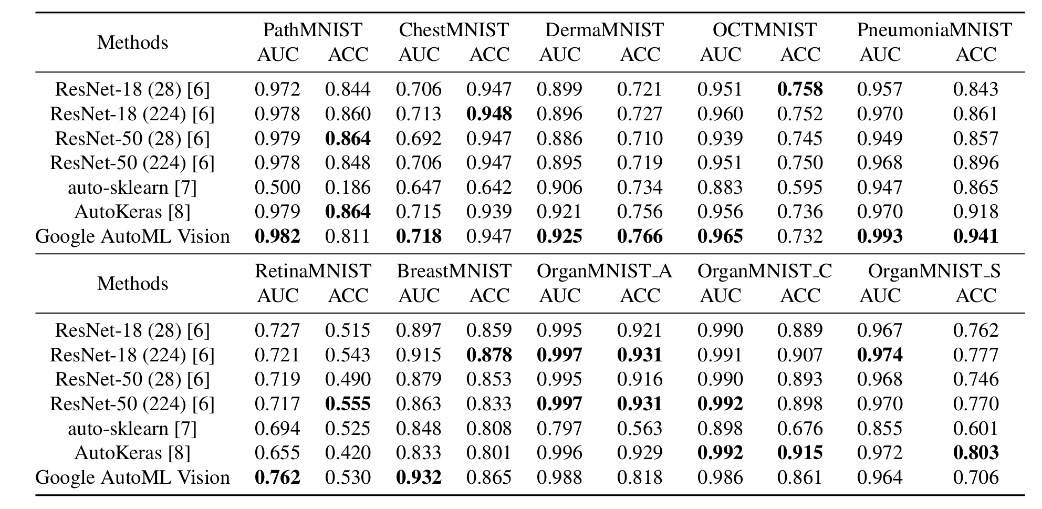
**4.3优化器**

SGD的收敛速度较慢，我们换成了Adam优化器，优化速度明显提升，这也是能够减少epoch数目的原因之一。

**五、实验结果（与论文原文的结果进行对照）**

由于训练步骤重复且冗长，在征得助教同意后，我们选取了不同数据规模、不同任务的7种数据进行测试，那些具有相同任务、相近数据规模的任务没有做，但并不难看出我们的模型和训练步骤的正确性。

下表为论文中的指标：



下表为我们的结果：

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | AUC | ACC |
| PathMNIST | Resnet18 | 0.978 | 0.864 |
|  | Resnet50 | 0.985 | 0.836 |
| ChestMNIST | Resnet18 | 0.766 | 0.947 |
|  | Resnet50 | 0.988 | 0.871 |
| DermaMNIST | Resnet18 | 0.915 | 0.741 |
|  | Resnet50 | 0.903 | 0.728 |
| PneumoniaMNIST | Resnet18 | 0.960 | 0.837 |
|  | Resnet50 | 0.960 | 0.897 |
| RetinaMNIST | Resnet18 | 0.740 | 0.503 |
|  | Resnet50 | 0.745 | 0.525 |
| BreastMNIST | Resnet18 | 0.883 | 0.827 |
|  | Resnet50 | 0.864 | 0.821 |
| OrganMNIST\_C | Resnet18 | 0.990 | 0.884 |
|  | Resnet50 | 0.975 | 0.746 |

可见，我们的实验结果大部分与论文中的相近，还有几组实验的结果优于论文的结果。实验较为成功。