人工智能基础大作业二：图像分类比赛

自75 常成 2017010252 **Chad**

目录

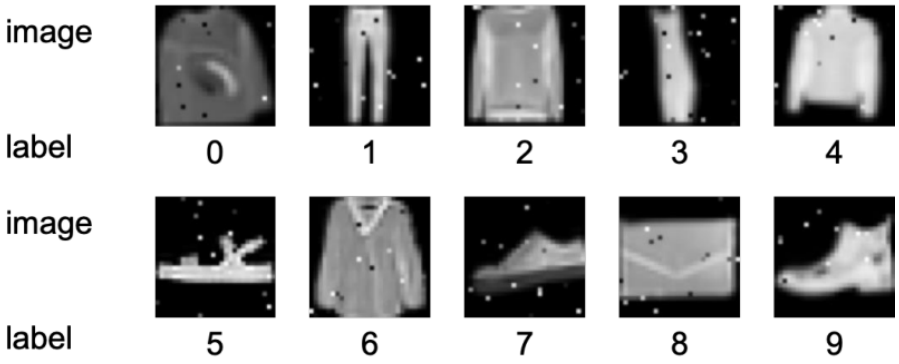
1 任务描述

Team：**Chad**

Public：**90.400**　51/194

Private：**89.971**　73/194

本次大作业要求完成 10 类图片的分类问题，图像示例及类别如下:



训练集由30000张图片组成(train.npy)，测试集由5000张图片组成(test.npy)，每个npy文件包含一个N*784的矩阵，N为图片数量。矩阵每行对应一张28*28的图片。train.csv文件包含训练集的标签，含*image_id*和*label*两列。

在预测环节，我们利用神经网络训练好模型对测试集中的图片进行分类，预测结果生成csv文件提交至kaggle。

本次大作业采用编程语言Python 3.6完成，深度学习框架选用pytorch。

2 网络模型

2.1 Lenet

参考助教提供的tutorial文件，mnist的样例代码中使用的网络模型就是Lenet，Lenet-5是最早的卷积神经网络模型，如图1，即为针对mnist数据集的Lenet网络：

Mnist数据集输入图片的大小为32×32，本次作业项目的输入图片大小为28×28，因此各层的图片大小及参数有所不同，具体见图2:

将训练集随机分出3000张组成验证集，余下的27000张图片仍作为正常的训练集，这样在训练阶段可以用验证集的预测结果来检验已有网络模型的性能。

为检验Lenet的性能，采用SGD优化器方法，设置学习率为0.01，epoch数

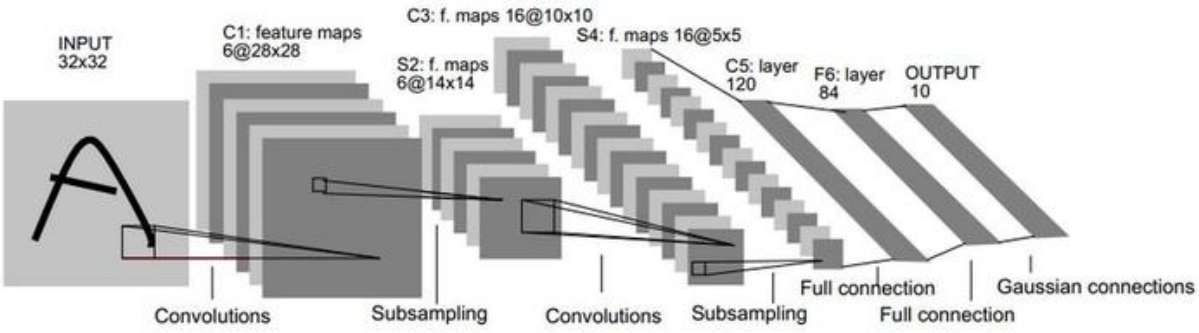


图 1: Lenet Mnist数据集网络结构

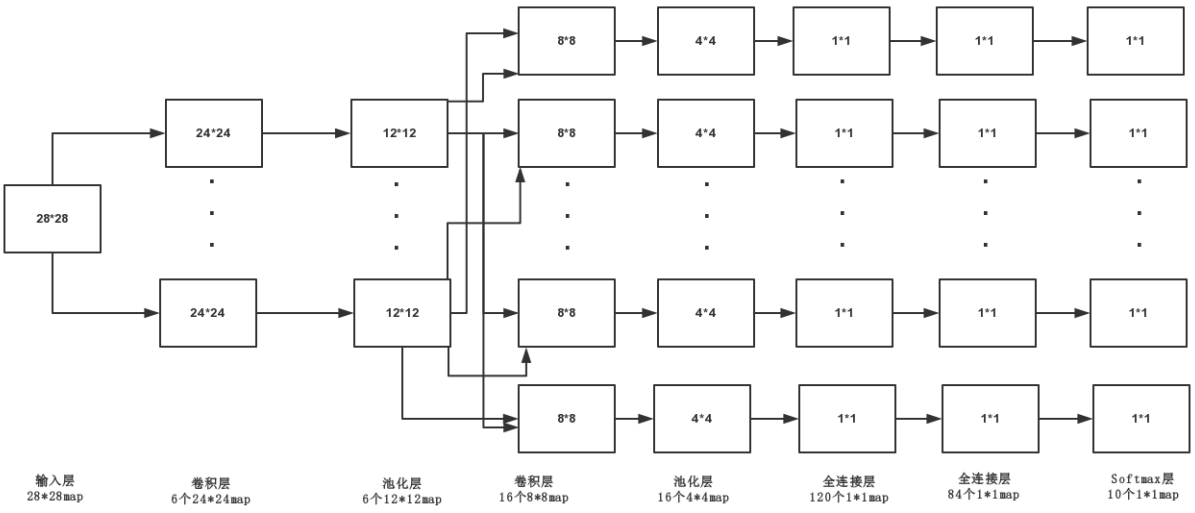


图 2: 针对28*28图片设计的Lenet结构

为25，最后5个epoch以及验证集上的预测结果如图3

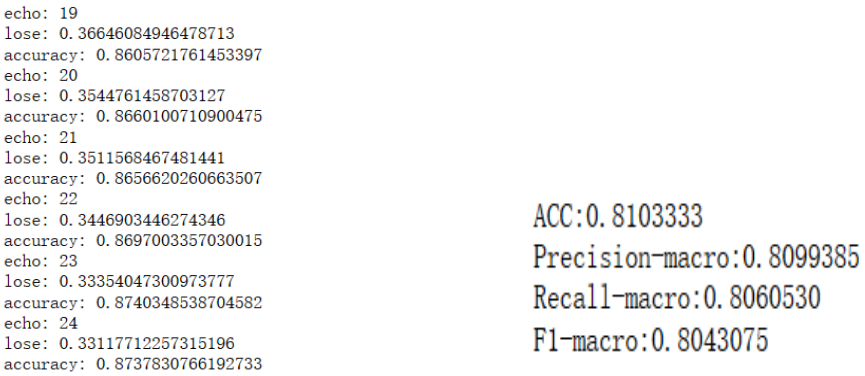


图 3: lenet在训练集和验证集上的结果

可以看到，Lenet模型对于本次作业项目的分类问题效果并不是很好，为进一步将模型”好坏”指标量化，引入两条曲线测量指标：ROC曲线及PR曲线。

ROC曲线：ROC为二维平面上的曲线，平面的横坐标是false positive rate(FPR)，纵坐标是true positive rate(TPR)。对某个分类器而言，我们可以根据其在测试样本上的表现得到一个TPR和FPR点对。这样，此分类器就可以映射成ROC平面上的一个点。这些点组成的曲线可以反映敏感性和特异性连续变量的综合指标。ROC模型越凸向左上方说明模型效果越好。

PR曲线：PR曲线是以召回率(Recall)为x轴，预测率(Precision)为y轴，二者的计算公式为：

$$Recall = \frac{TP}{TP + FN}$$

$$Precision = \frac{TP}{TP + FP}$$

PR曲线越凸向右上方证明模型效果越好。

Lenet网络生成的预测结果的ROC曲线和PR曲线如图4 5

可以看到效果确实很差，尤其是class 6类别，ROC和AP的score值均为最低，且严重偏离左上方和右上方，由此可见Lenet不能很好地解决作业项目的问题。

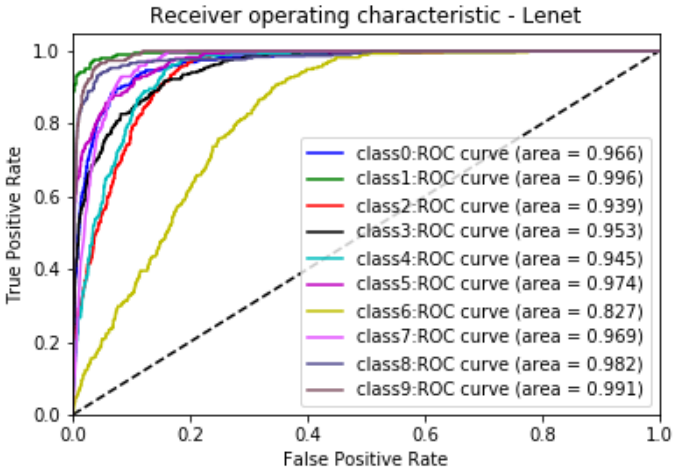


图 4: Lenet的ROC曲线

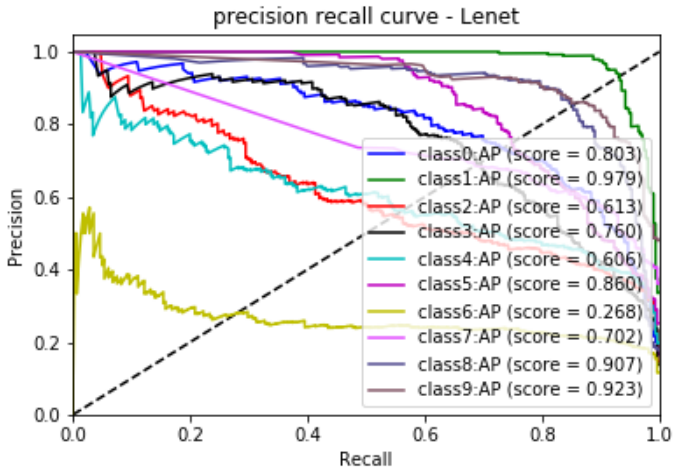


图 5: Lenet的PR曲线

2.2 Alexnet

AlexNet诞生于2012年，使用了8层卷积神经网络，首次在CNN中成功应用了ReLU、Dropout和LRN等技巧，其网络结构如图6所示：

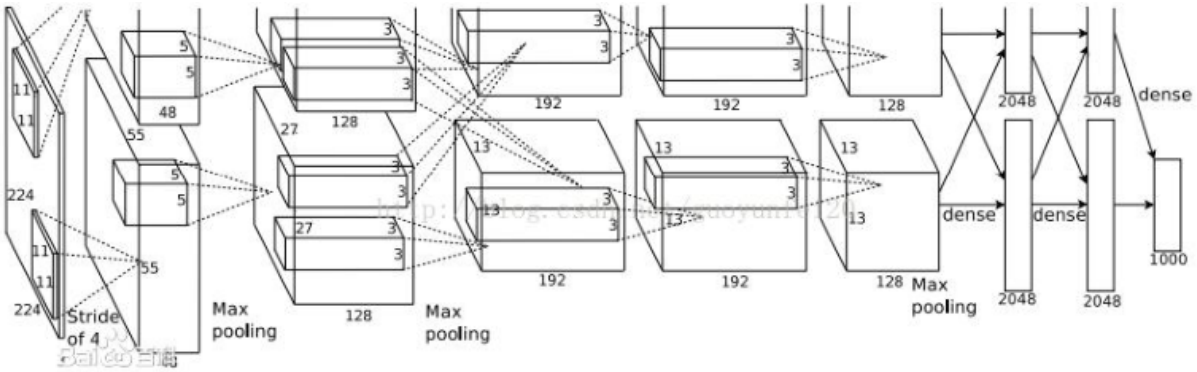


图 6: ALexnet针对224*224图片的网络结构

由于本次作业项目的图片均为28*28，卷积核、通道数和步长都需要有所调整才可以使图片正常输入网络模型，因此修改Alex的结构如图7所示：

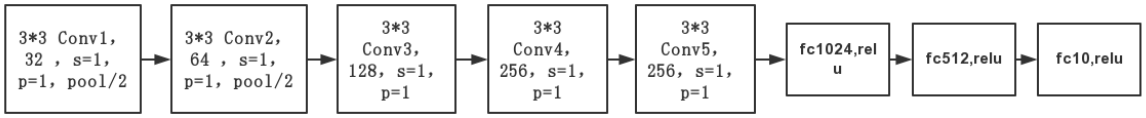


图 7: ALexnet针对28*28图片的网络结构

为检验Alexnet的性能，采用SGD优化器方法，设置学习率为0.007，epoch数为30，最后5个epoch以及验证集上的预测结果如图8

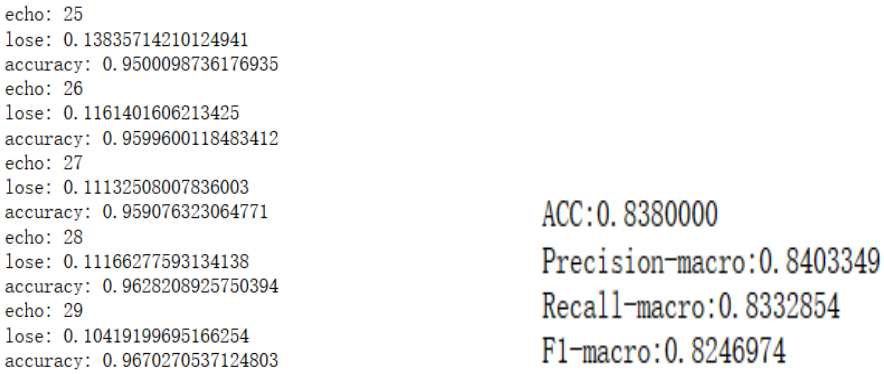


图 8: Alexnet在训练集和验证集上的结果

直观上看Alexnet的性能强于Lenet，在训练集上可以更快的达到较高的准确率，在验证集上的准确率也会提高一些，再来看的Alexnet的ROC曲线及PR曲线，如图9 10

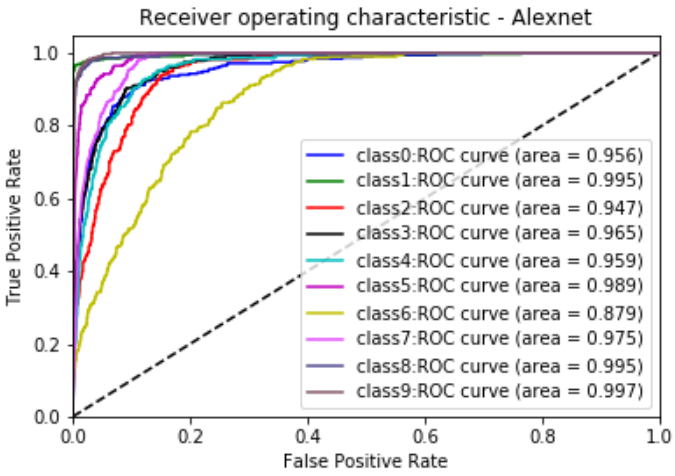


图 9: Alexnet的ROC曲线

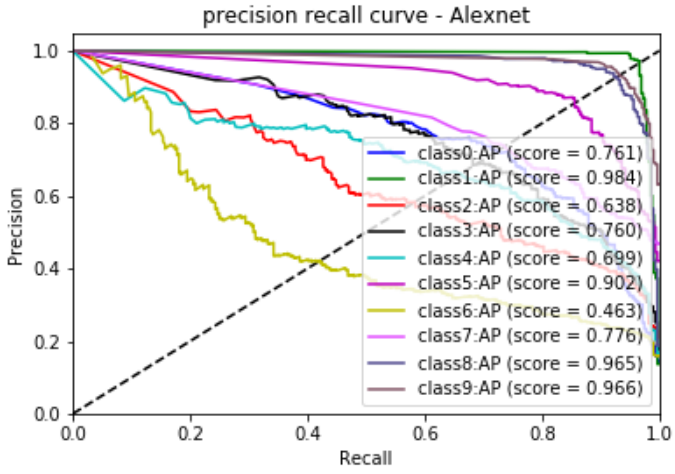


图 10: Alexnet的PR曲线

可以看到相对于lenet，Alexnet的ROC曲线更加向左上方靠拢，PR曲线也更加向右上方靠拢，对于不易分辨的class 6，效果也得到了较为明显的改善。

2.3 Resnet

Resnet网络在2015年由何恺明提出，参考论文Deep Residual Learning for Image Recognition，之前的神经网络均显示网络有足够深度是模型表现良好的前提，但是当网络叠加到一定深度后，简单的网络堆叠会使效果变差，产生更

多的训练误差。

产生误差的原因是当网络结构加深，梯度会出现消失的现象，在后向传播的时候，无法有效的把梯度更新到前面的网络层，就会导致训练和测试结果变差，Resnet解决该问题的方法是引入残差网络，如图11

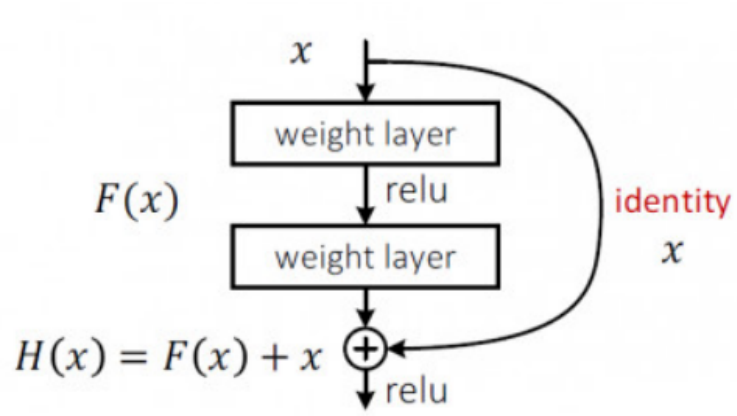


图 11: Resnet的残差网络

论文中给出几种resnet的结构示意图，如图12：

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图 12: Resnet的结构图

考虑到本次作业项目的图片均为28*28,结构图中的卷积核、通道数及步长等参数均需要调整，对Resnet18的结构，设计参数结构如图13所示

为检验Resnet的性能，采用SGD优化器方法，设置学习率为0.05，epoch数为10，最后5个epoch以及验证集上的预测结果如图14

再来看的Resnet的ROC曲线及PR曲线，如图15 16

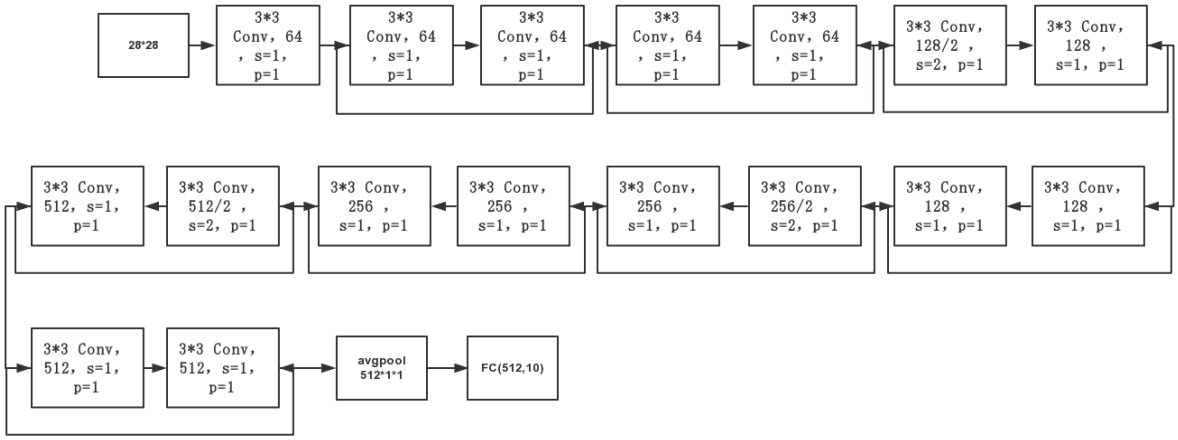


图 13: Resnet18针对28*28图片的网络结构

```
echo: 5
lose: 0.1492287660662032
accuracy: 0.9466330963665087
echo: 6
lose: 0.11315783911265469
accuracy: 0.9591380331753555
echo: 7
lose: 0.06358666349890955
accuracy: 0.9777744865718799
echo: 8
lose: 0.051457812983136605
accuracy: 0.9823706556082148
echo: 9
lose: 0.030335172375231558
accuracy: 0.9901856240126383
```

```
ACC:0.8550000
Precision-macro:0.8542710
Recall-macro:0.8540529
F1-macro:0.8498081
```

图 14: Resnet在训练集和验证集上的结果

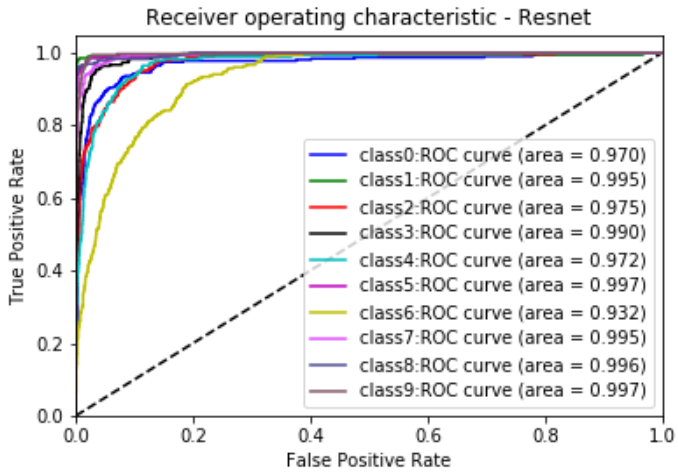


图 15: Resnet的ROC曲线

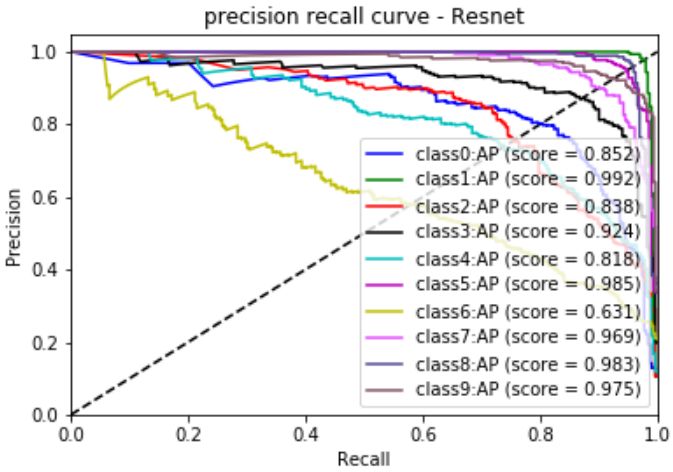


图 16: Resnet的PR曲线

由此我们可以看到，Resnet可以在训练集上较快达到很高的准确率，在验证集上的准确率相对于Alexnet有了进一步的提升，观察ROC曲线和PR曲线也可看到，图像显示的性能有了进一步的提升。

综上所述的三种模型，我们选择Resnet作为本次比赛的基本网络模型，并在下文进行改善和性能提升。

3 数据增强

深层的神经网络需要大量的数据训练才可以获得比较理想的结果，因此我们可以通过数据增强来增加训练样本的多样性，同时增强模型的鲁棒性，防止训练过拟合，进一步提高泛化能力，常用的数据增强方法是：

- 翻转。将图像沿水平或垂直随机翻转一个角度；
- 旋转。将图像按顺时针或逆时针方向随机旋转一定角度；
- 随机裁剪或补零。将图像随机裁剪或补零到指定大小；
- 加噪声。在图像中加入随机噪声；
- 缩放。将图像放大或缩小。

在本次作业项目的数据增强环节中，我首先选择加入翻转和旋转，使图片依概率0.5水平翻转，顺时针或逆时针随机旋转0到20度，将优化器调整为Adam，学习率为0.001，epoch设置为40，得到训练集及验证集的结果为图17

```
echo: 35
lose: 0.13077417498021895
accuracy: 0.9512983807266983
echo: 36
lose: 0.12602413772336116
accuracy: 0.9550083925750394
echo: 37
lose: 0.11616872699481051
accuracy: 0.9564820300157978
echo: 38
lose: 0.1112595141393031
accuracy: 0.9574052132701422
echo: 39
lose: 0.11253310644654867
accuracy: 0.9570003949447078
```

ACC:0.8936667
Precision-macro:0.8932252
Recall-macro:0.8927241
F1-macro:0.8924767

图 17: Resnet在训练集和验证集上的结果

此时的ROC和PR曲线图18 19

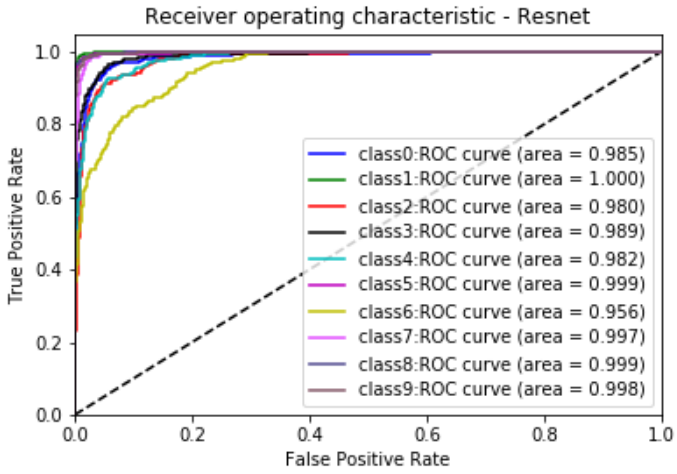


图 18: Resnet增强数据集的ROC曲线

在验证集上的结果达到了89%，且曲线显示的网络性能更加优良。

对于加噪声部分的数据增强处理，我考虑了椒盐噪声和高斯噪声两种，分别选择3%图像加入椒盐噪声，和($\mu = 0$,$\sigma = 0.5$)的高斯随机噪声，epoch数目增加10，其余参数不变，在3000张图片的验证集上结果如图20：

可见加入随机噪声对于训练效果并没有明显的改进，反而出现了退化，而输入图片大小本身为28*28，无需进行放缩和裁剪，因此在数据增强部分，我只选择添加水平翻转与随机旋转两种。

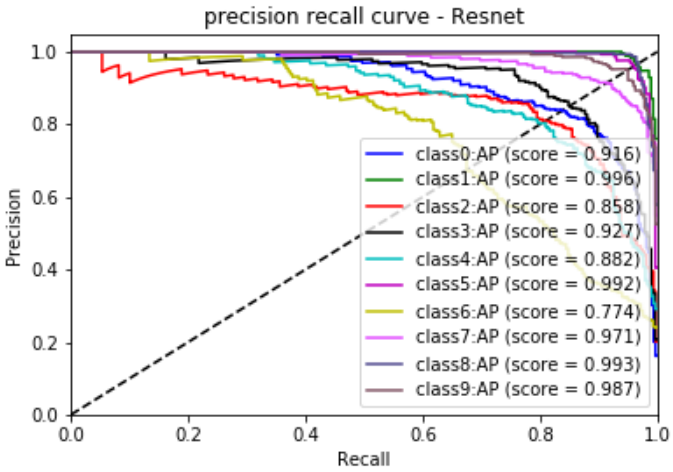


图 19: Resnet增强数据集的PR曲线

ACC:0.8833333
Precision-macro:0.8837185
Recall-macro:0.8818403
F1-macro:0.8823314

ACC:0.8750000
Precision-macro:0.8749027
Recall-macro:0.8741643
F1-macro:0.8740761

图 20: Resnet添加椒盐和高斯噪声后在验证集的结果

4 训练方法的改进

在之前的训练过程中，均是设置一定的学习率和层数，使模型一直按照此参数运行到最后，这样做是存在不合理的地方的，我们知道，学习率($learining \quad rate$)对于机器学习训练模型是极其重要的参数，如果最初学习率设置过大，会导致训练时的loss和accuracy左右摇摆，得不到很好的效果，而如果lr设置过小，则会使更新迭代的过程大大增加，所以我们可以选择采用学习率衰减的方法，刚开始选择较大的学习率，不会引起很大影响，还可以很快地更新，之后再视情况降低学习率，使训练效果趋向最佳。

学习率更新也有两种方式：

- 平原衰减。即采用监听的方式，当发现训练过程中存在一段时间的训练集accuracy不上升或loss下降的情况，利用optim中的ReduceLROnPlateau自动降低学习率。
- 定时衰减。每训练一定层数的epoch，降低部分学习率。

在尝试第一种方式没有取得明显提升效果后，我选择了第二种降低学习率的方法，并在经过反复试验和调整参数后，经验证，按照图21的学习率衰减函数会取得不错的效果(优化器选用Adam，横轴代表epoch，纵轴代表lr)：

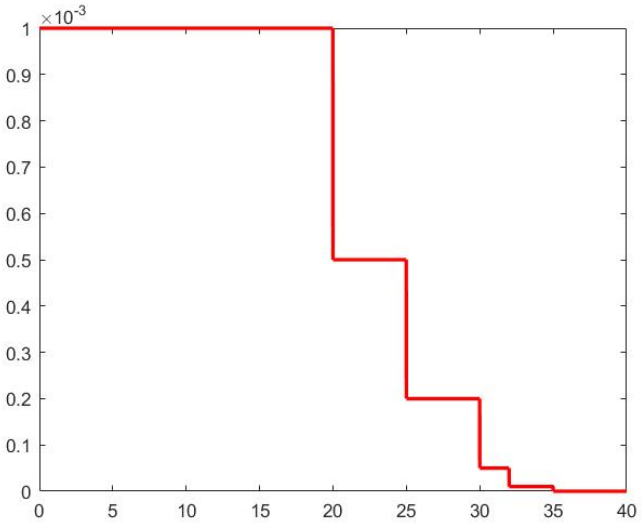


图 21: 学习率衰减

按照这种方法，最后5个epoch以及验证集上的预测结果如图22

```
echo: 0
lose: 0.05254228861552279
accuracy: 0.981891785150079
echo: 1
lose: 0.050537977685436816
accuracy: 0.9819238744075829
echo: 2
lose: 0.04716542033452135
accuracy: 0.983370359399684
echo: 3
lose: 0.04323366321005386
accuracy: 0.9851895734597157
echo: 4
lose: 0.041559164602115256
accuracy: 0.985814079778831
```

```
ACC:0.9056667
Precision-macro:0.9062864
Recall-macro:0.9067153
F1-macro:0.9063743
```

图 22: 学习率衰减的Resnet在训练集和验证集上的结果

此时的ROC和PR曲线图23 24

可见网络训练效果变得更好，在训练集上可以较快达到很高的准确率和较低的loss，且在验证集上的预测准确率可以超过90%，由此模型生成的预测csv文件也可以在public榜单上超过90%。

综上所述，采用Resnet18基本网络模型，采用随机翻转和随机旋转等数据增强方式，并且应用学习率衰减的训练方法，可以取得不错的分类效果。

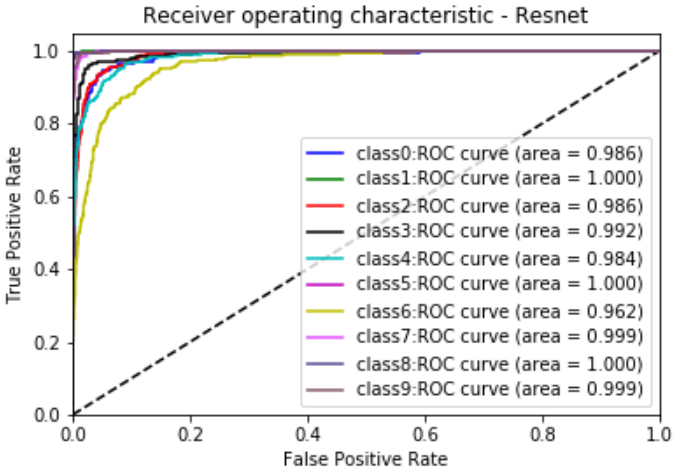


图 23: Resnet学习率衰减的ROC曲线

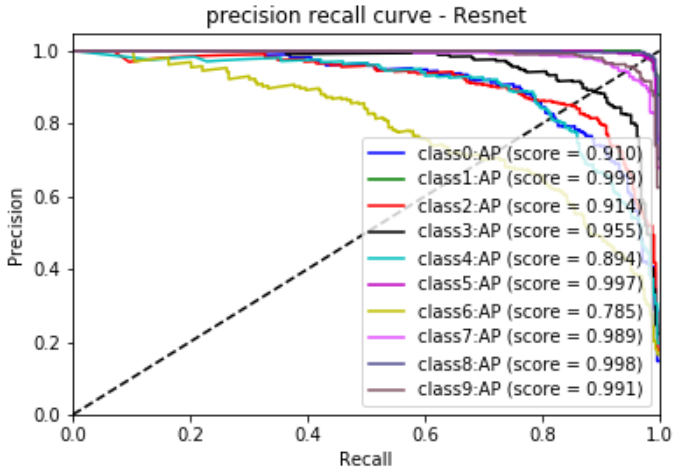


图 24: Resnet学习率衰减的PR曲线

5 项目总结

首先，我选中的csv评测模型在public榜达到了90.400%的准确率，但是在放榜后，private榜掉到了89.971%，名次下降20+，我还发现图25

Submission and Description	Private Score	Public Score	Use for Final Score
submit33.csv 4 hours ago by Chad add submission details	0.90285	0.89133	☐
submit25.csv 2 days ago by Chad add submission details	0.89971	0.90400	✔

图 25: 让人心痛的kaggle

不得不说还是存在遗憾的，最后一次提交的正是我private榜的最高分数，可以达到90.285%，然而我被该文件在public榜的低分”迷惑”，没有把它选进最终评测。

不过大作业做到这里，比赛排名已经没那么重要了，这是我第一次训练机器学习模型，第一次参与kaggle比赛，体验感十足。起初接触各种网络的时候，参数调整时经常会出现数据维数报错，在弄清网络结构之后，调整参数也变得很容易了。通过这次大作业，我对卷积神经网络有了更深的认识，对pytorch框架也使用得更加熟练，感谢助教和老师几次tutorial的分享和课堂指导！

参考文献

[1] *ai_tutorial2*

[2] Deep Residual Learning for Image Recognition@microsoft.com

[3] https://blog.csdn.net/qq_36338754/article/details/92799540 CSDN文章：pytorch对resnet的官方实现

[4] https://blog.csdn.net/sunqiande88/article/details/80100891 CDSN:Pytorch实战2：ResNet-18实现Cifar-10图像分类（测试集分类准确率95.170

[5] https://blog.csdn.net/dcrmg/article/details/79241211CSDN:AlexNet神经网络结构