



山东大学  
SHANDONG UNIVERSITY

# 信息基础II 机器学习、模式识别与深度学习

## 实验三

陈雷

山东大学信息科学与工程学院

## 实验内容

实验一：常规神经网络函数逼近实验

实验二：基于LeNet-5的MNIST字符识别

实验三：ResNet-18

实验四：Selective search

实验五：Yolo V3

期末项目：命题项目和自选项目



## 实验三：ResNet

**ResNet（Residual Network，残差网络）**是由Microsoft Research的4位学者（何凯明，张翔宇，任少卿，孙剑）在2015年共同提出的。

该网络出自论文**Deep Residual Learning for Image Recognition（CVPR 2016最佳论文）**。

提出的深度残差网络在当年取得了多个图像方面比赛的冠军。



## 实验三：ResNet

### 深层次网络训练瓶颈：梯度消失，网络退化

随着网络深度增加，模型会产生退化现象。它不是由过拟合产生的，而是由冗余的网络层学习了不是恒等映射的参数造成的。

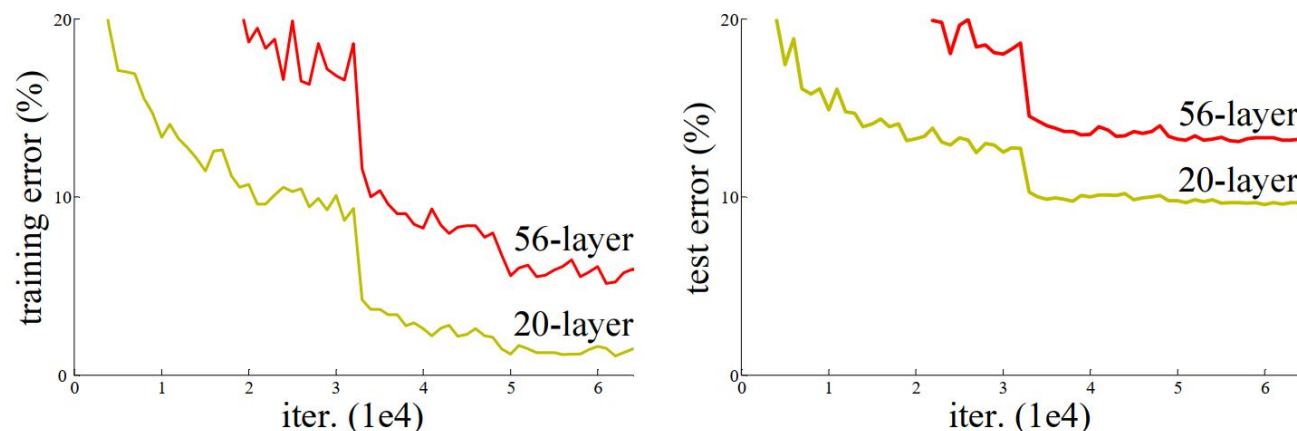
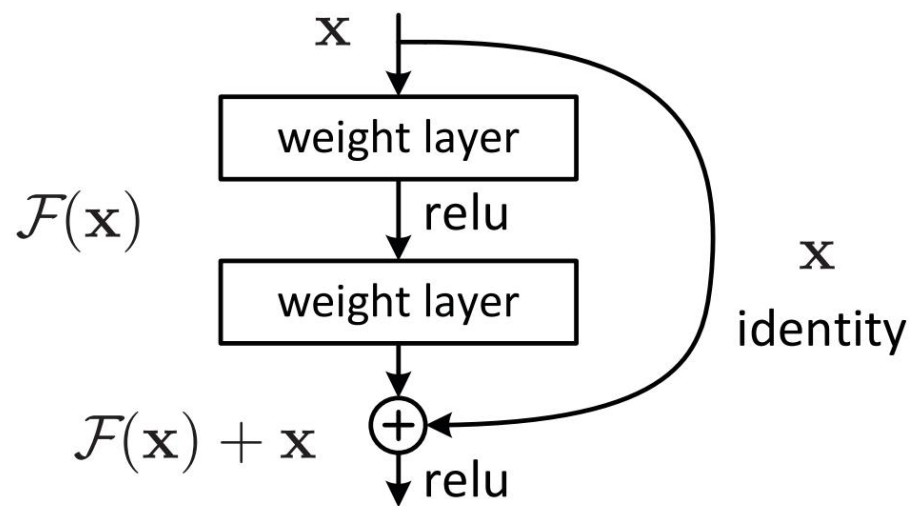


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.



## 实验三：ResNet

ResNet的思想是假设我们涉及一个网络层，存在最优化的网络层次，往往设计的深层次网络是有很多网络层为冗余层。那么我们希望这些冗余层能够完成恒等映射（Identity Mapping），保证经过该恒等层的输入和输出完全相同。



## 实验三：ResNet

### 例子：

假设：在引入残差之前，输入 $x=6$ ，要拟合的函数 $H(x)=6.1$ ，也就是说普通网络找到了一组 $w'$ 使得 $F'(x,w') \rightarrow H(x)=6.1$ 。引入残差后，输入不变 $x=6$ ，要拟合的函数 $H(x)=6.1$ ，变化的是 $F(x,w)+x \rightarrow H(x)=6.1$ ，可得 $F(x,w) \rightarrow 0.1$ 。

如果需拟合的函数 $H(x)$ 增大了0.1，对普通网络来说 $F'(x,w')$ 就是从6.1变成了6.2，增大了1.6%。而对于ResNet来说， $F(x,w)$ 从0.1变成了0.2，增大了100%。

在残差网络中输出的变化对权重的调整影响更大，也就是说反向传播的梯度值更大，训练就更加容易。



## 实验三：ResNet

残差网络的特点是容易优化，并且能够通过增加相当的深度来提高准确率。其内部的残差块使用了跳跃连接（shortcut connection）：shortcut连接的结构保证了反向传播更新参数时，很难有梯度为0的现象发生，不会导致梯度消失。

当网络自行决定了哪些层为冗余层后，通过学习残差 $F(x)=0$ 来让该层网络恒等映射上一层的输入，使得有了这些冗余层的网络效果与没有这些冗余层的网络效果相同，这样很大程度上解决了网络退化问题。





## 实验三：ResNet

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				
		$\begin{bmatrix} 3\times 3, 64 \\ 3\times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, 64 \\ 3\times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 64 \\ 3\times 3, 64 \\ 1\times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times 3, 128 \\ 3\times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, 128 \\ 3\times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times 1, 128 \\ 3\times 3, 128 \\ 1\times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3\times 3, 256 \\ 3\times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, 256 \\ 3\times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1\times 1, 256 \\ 3\times 3, 256 \\ 1\times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3\times 3, 512 \\ 3\times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times 3, 512 \\ 3\times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times 1, 512 \\ 3\times 3, 512 \\ 1\times 1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		$1.8 \times 10^9$	$3.6 \times 10^9$	$3.8 \times 10^9$	$7.6 \times 10^9$	$11.3 \times 10^9$





## 实验三：ResNet

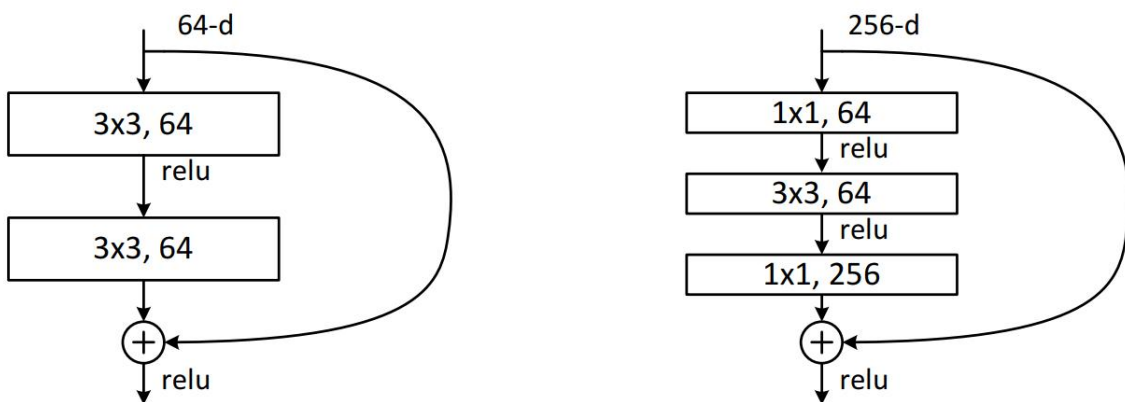
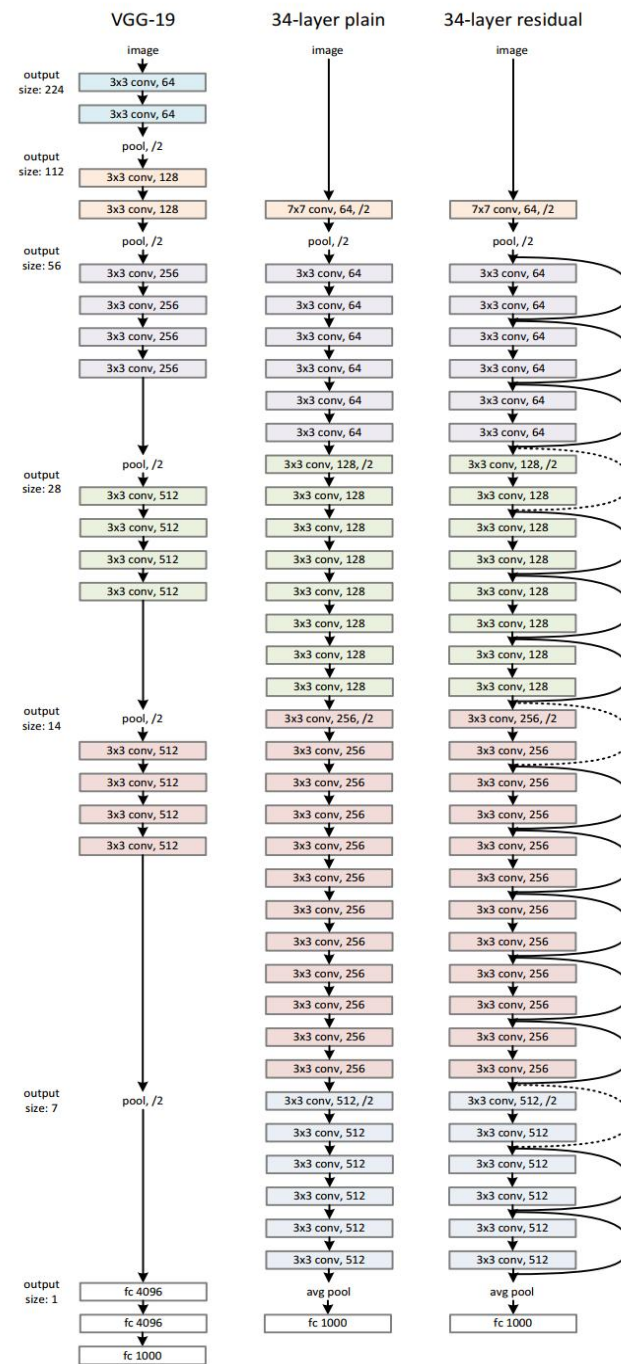


Figure 5. A deeper residual function  $\mathcal{F}$  for ImageNet. Left: a building block (on  $56 \times 56$  feature maps) as in Fig. 3 for ResNet-34. Right: a “bottleneck” building block for ResNet-50/101/152.

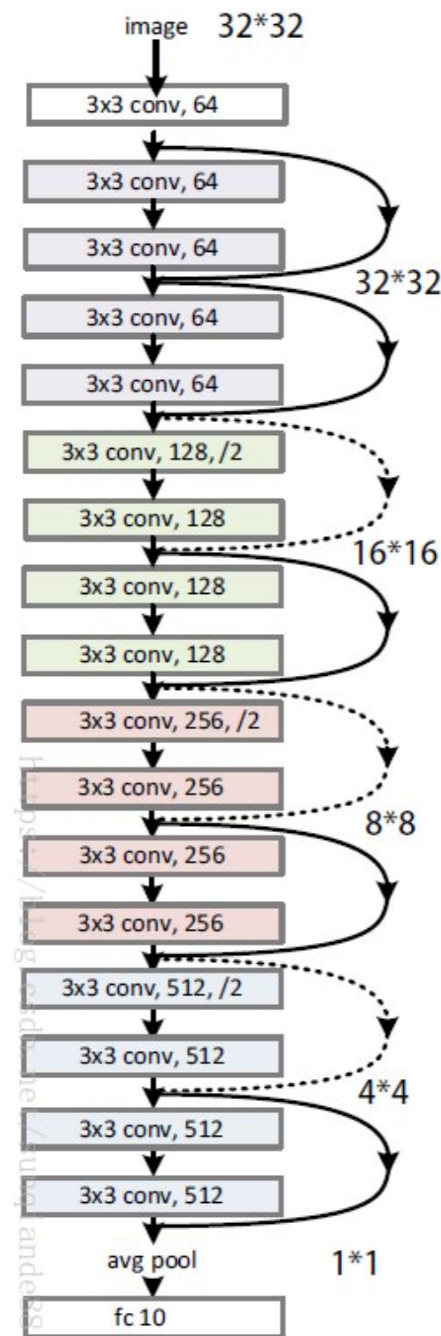


## 实验三：ResNet-18

### ResNet-18实现CIFAR-10图像分类

#### 1. 搭建ResNet-18。

2. 所选数据集为**CIFAR-10**，该数据集共有60000张带标签的彩色图像，这些图像尺寸 $32 \times 32$ ，分为10个类，每类6000张图。这里面有50000张用于训练，每个类5000张，另外10000用于测试，每个类1000张。



## 实验要求:

1. 要求自己编程实现网络结构;
2. 开发语言为python或者C/C++;
3. 要求提交报告和代码。

