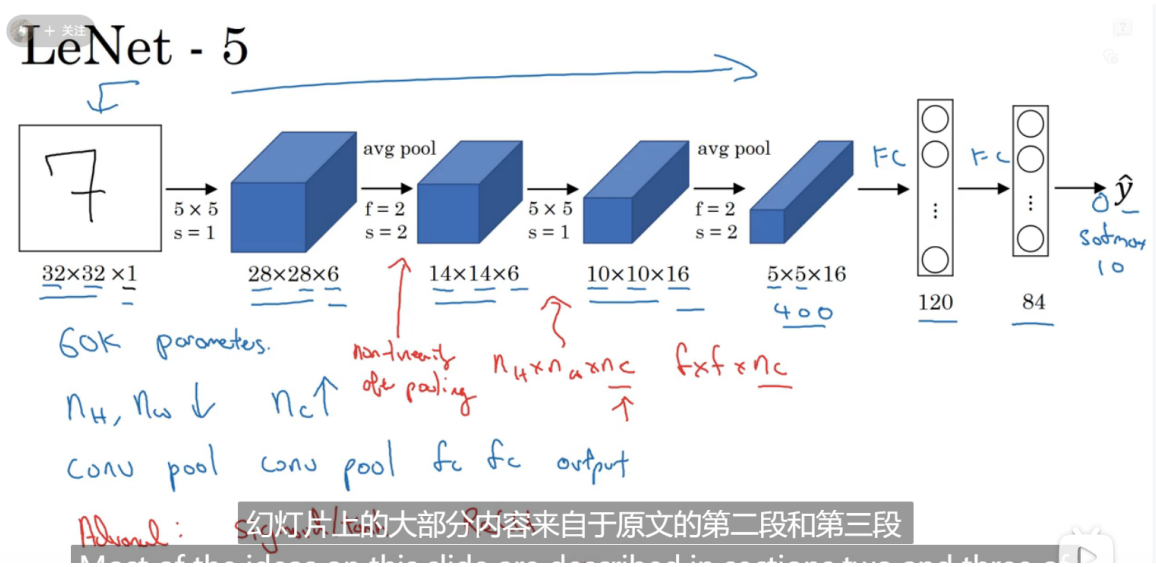
- [ ] 预训练权重
- [x] 数据增强，扩充数据，裁剪翻着缩放旋转
- [x] 加深网络结构，主干，resnet

经典网络

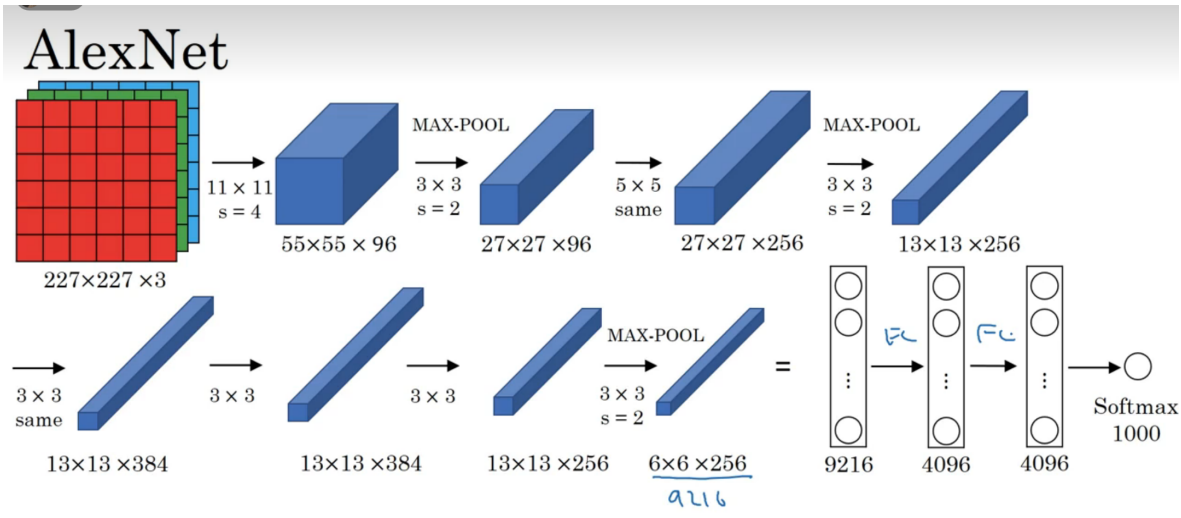
- LeNet-5
- AlexNet
- VGG-16

LeNet-5

网络结构

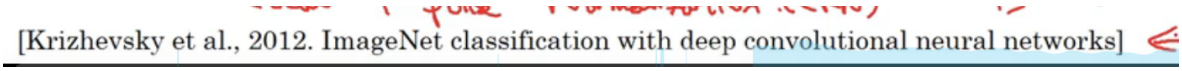


AlexNet

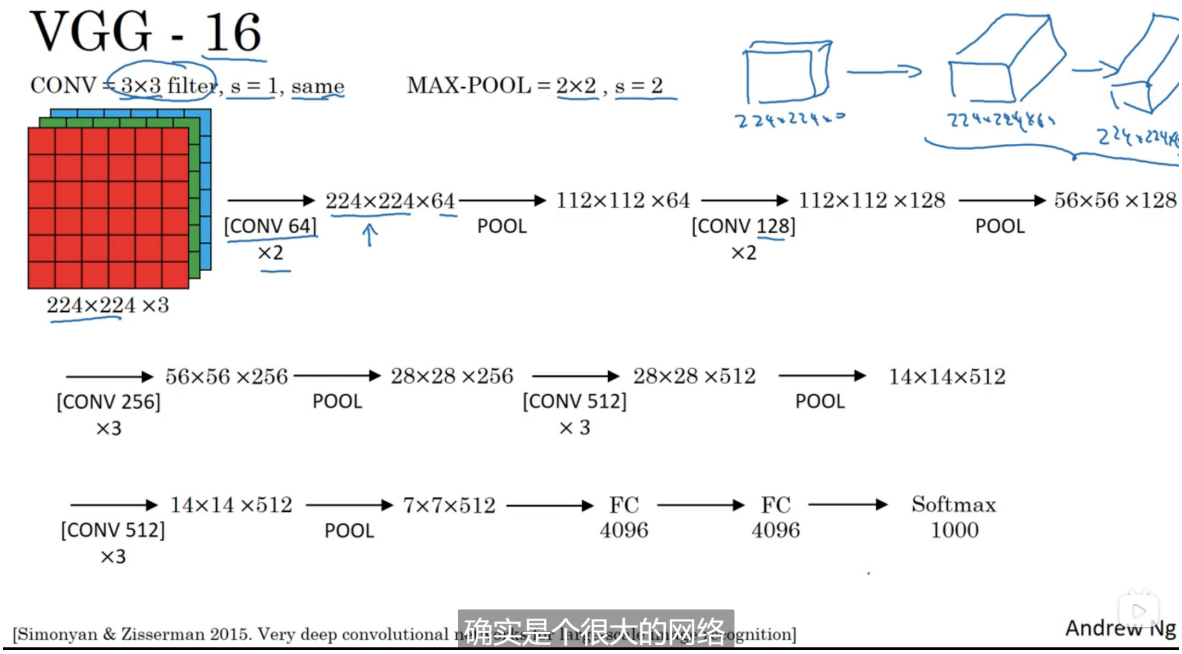


advantage：ReLu function, two GPUs

feature: local response normalization(LRN), look at all the dimension and normalize them. maybe we don't want more neurons with a very high activation. less use now



VGG-16



16 layers, simple and uniform structure

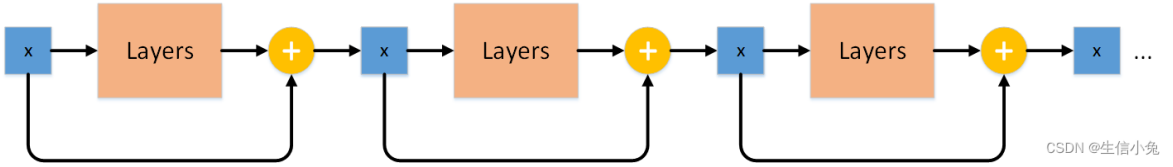
very very deep network

Resnet

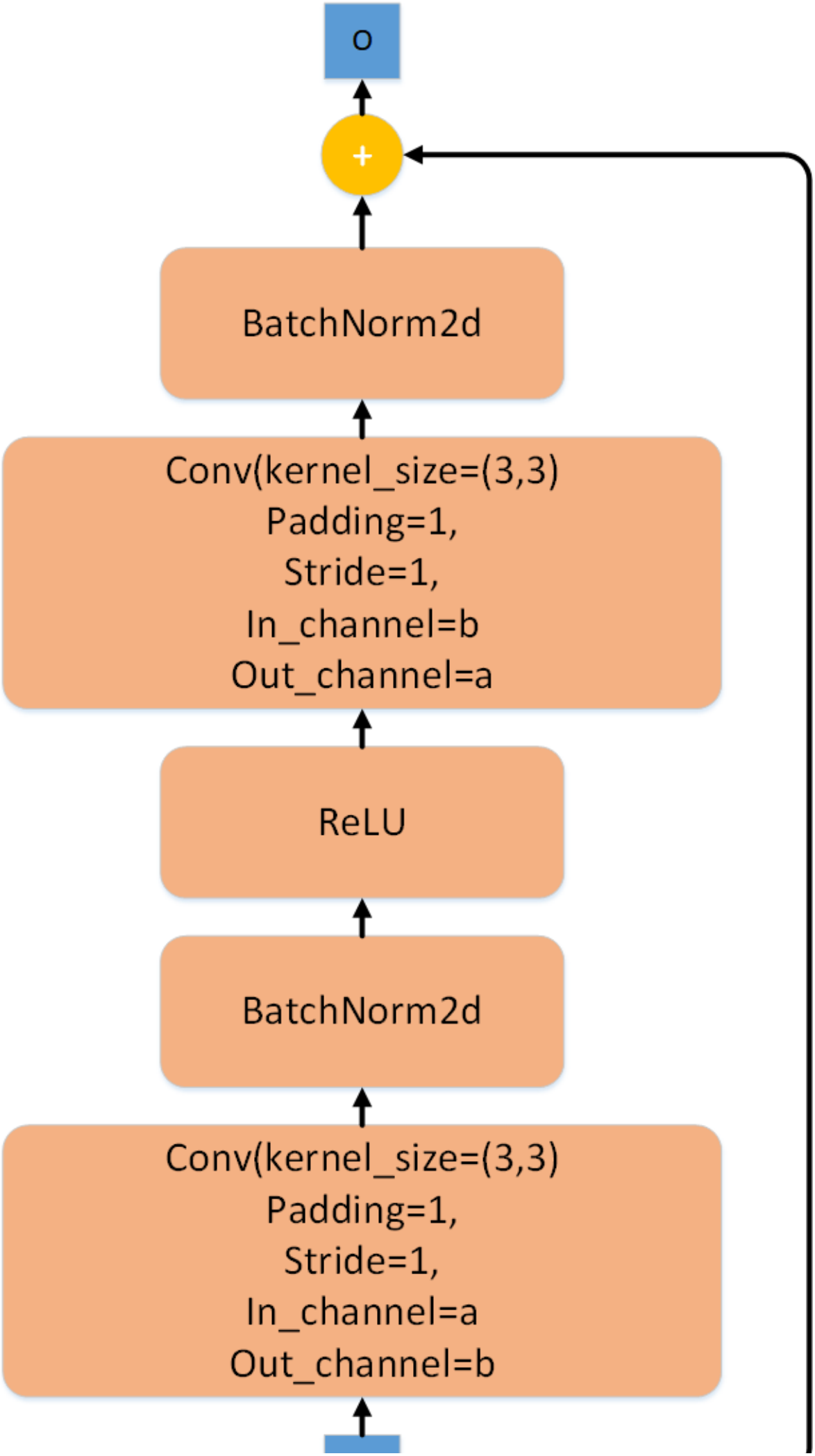
> ResNet-18，ResNet-34，ResNet-50，ResNet-101，ResNet152

Residual block

> shortcut--skip connection: which refers to a[l] just skipping a layer or many layers in order to pass the information deeper into the neural network



CSDN @生信小兔


o
+
BatchNorm2d
Conv(kernel_size=(3,3)
Padding=1,
Stride=1,
In_channel=b
Out_channel=a
ReLU
BatchNorm2d
Conv(kernel_size=(3,3)
Padding=1,
Stride=1,
In_channel=a
Out_channel=b

神经网络优化

v1.0

网络结构

第一层卷积：

filter: 5x5x4 ,stride=1,padding=2
output_size=(input_size+2xpadding-kernel_size)/stride+1=(28+2x2-5)/1+1=28
输出：28x28x4

激活函数：**ReLU.**
池化：使用**2x2,stride=1的池化单元进行最大池化**操作（步长小于池化单元大小，采用重叠池化）：
output_size=14x14x4

第二层卷积：

filter: 5x5x8 ,stride=1,padding=2

output_size=(input_size+2xpadding-kernel_size)/stride+1=(14+2x2-5)/1+1=14

输出：14x14x8

激活函数：**ReLU**.

池化：使用**2x2,stride=1的池化单元进行最大池化**操作（步长小于池化单元大小，采用重叠池化）：
output_size=7x7x8=512

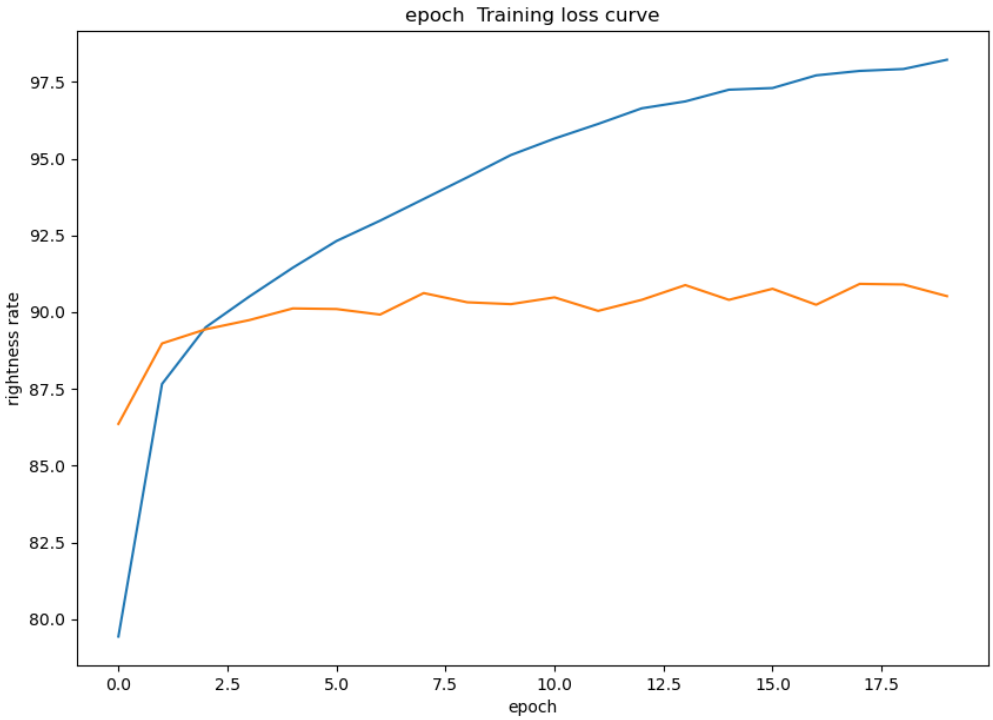
全连接层：输入尺寸为512，输出为要做分类的类别数

进行dropout操作，防止过拟合

输出层为 **log_Softmax**，即概率对数值 log(p(×))。采用log_softmax可以使后面的交叉熵计算更快

优化算法：adam，lr=0.001

> 训练正确率：92.84%，校验正确率：90.48%



V2

第一层卷积：

filter:==5x5x4=32,stride=1,padding=2
output_size=(input_size+2xpadding-kernel_size)/stride+1=(28+2x2-5)/1+1=28
输出：28x28x64

激活函数：**ReLU.**
池化：使用**2x2,stride=1的池化单元进行最大池化**操作（步长小于池化单元大小，采用重叠池化）：
output_size=14x14x64

第二层卷积：

filter: 5x5x64 ,stride=1,padding=2

output_size=(input_size+2xpadding-kernel_size)/stride+1=(14+2x2-5)/1+1=14

输出：14x14x64

激活函数：**ReLU**.

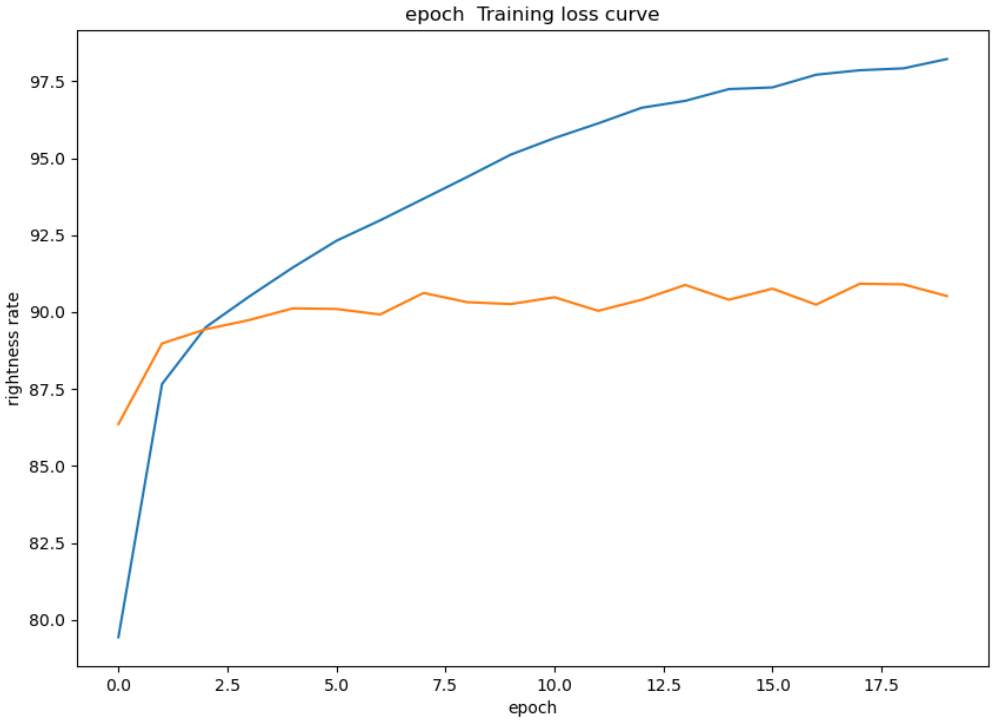
池化：使用**2x2,stride=1的池化单元进行最大池化**操作（步长小于池化单元大小，采用重叠池化）：
output_size=512

全连接层：输入尺寸为512，输出为要做分类的类别数

进行dropout操作，防止过拟合

输出层为 **log_Softmax**，即概率对数值 log(p(×))。采用log_softmax可以使后面的交叉熵计算更快

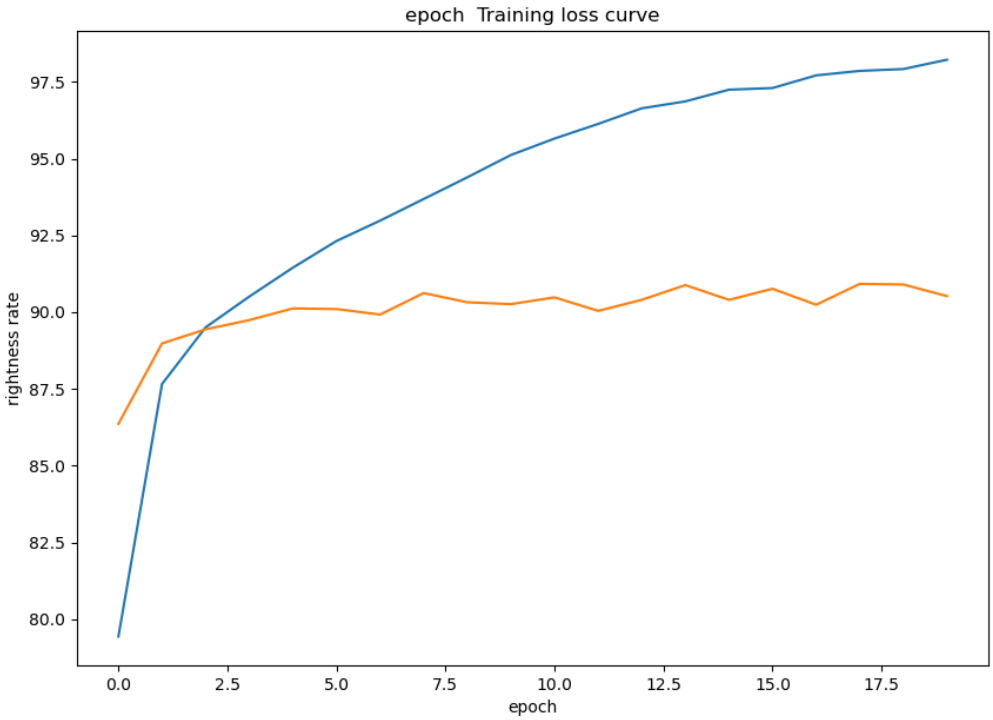
优化算法：adam，lr=0.003



训练正确率: 93.40%　　校验正确率: 90.26%

v3.0

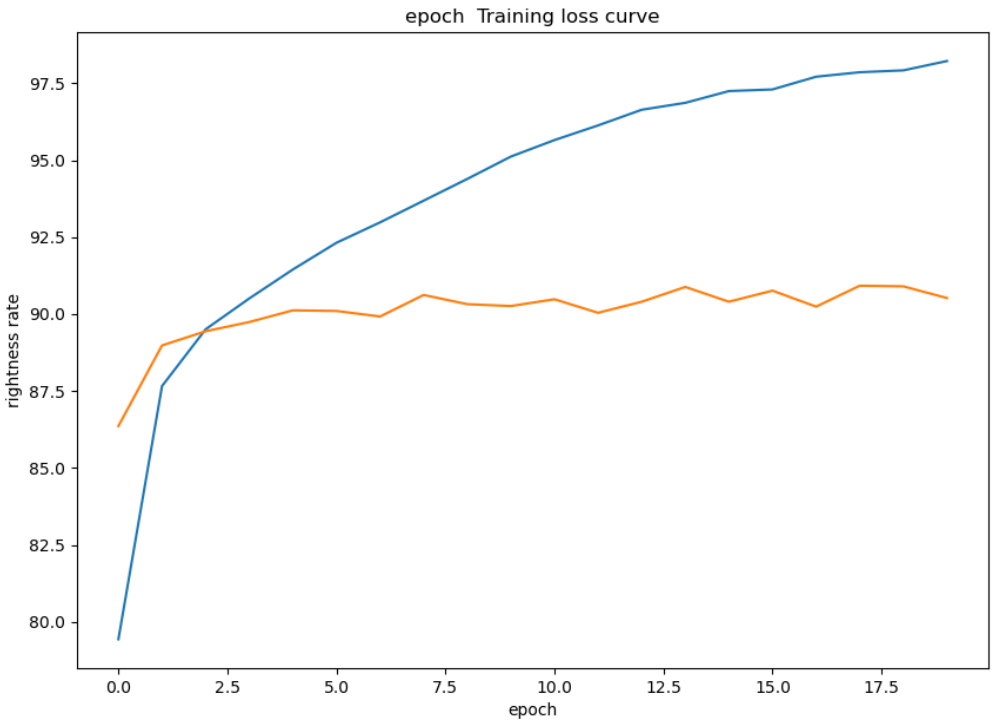
其他类似，不过增加了网络的深度，分别为depth=[4,8,16,32,64]，但是全连接层仍为512个节点。



训练正确率: 96.88%　　校验正确率: 91.16%

v4.0

depth=[4,8,16,32,64] ,且把全连接层的节点扩展到了7x7x64=4096个



```
训练正确率: 98.22%      校验正确率: 90.52%
```