DenseNet: 2018

Densely Connected Convolutional Networks

code:https://github.com/liuzhuang13/DenseNet

论文出发点或背景

最近的研究表明，将靠近输入和靠近输出的层通过跳连方式连接起来的网络可以变得更深、准确率更高、训练更高效。

当网络变深的时候，会发生梯度弥散的现象。最近的ResNet和Highway Net是通过跳连的方式解决的。随即深度是通过在训练过程中随机删除层来缩短ResNet以此允许更好的信息和梯度传递。FractalNet通过重复组合多个具有不同卷积块数量的并行层，进而获得更大的深度。

尽管他们的网络拓扑和训练方法上都有所不同，但是他们都通过跳连实现了从前层到后层的连接。

ResNet的优点是梯度可以通过恒等映射的支路直接进入下层，但是恒等映射与输出相加的形式可能会阻碍网络中的信息流。

论文创新思路

为了确保网络层间有最大的信息流动，我们直接将所有层的特征图拼接起来。为了保证前馈特性，每一层都从所有之前的层获得额外的输入，并将自己的特征映射传递到后续所有的层。相比于ResNet，我们没有通过使用加和来组合特征，而是通过在通道上拼接。

为了改善层之间的信息流传递，我们提出了一种新的连接方式 $\mathbf{x}_\ell = H_\ell([\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_{\ell-1}])$,

论文方法大概介绍

普通的卷积L层神经网络只有 L个连接。而我们的DenseNet有$L(L-1)/2$条连接

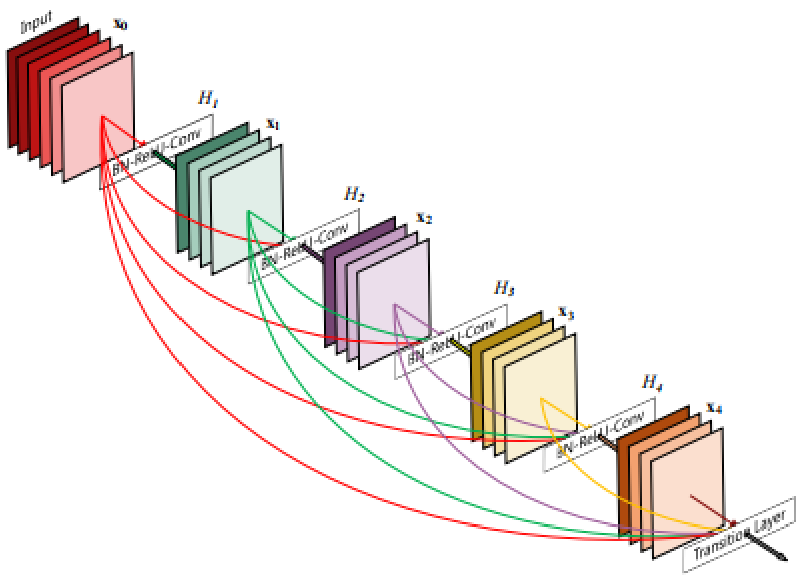


Figure 1: A 5-layer dense block with a growth rate of $k = 4$. Each layer takes all preceding feature-maps as input.

复合函数：批归一化、ReLU和3×3卷积

过渡层：批归一化、1×1卷积层、2×2平均池化

瓶颈层：BN-ReLU-Conv（1×1）-BN-ReLU-Conv（3×3）

实际效果

观察到Dense Connection具有正则化效应

比ResNet的参数更少

DenseNets的表现与最先进的ResNets相当，同时需要明显更少的参数和计算来实现类似的性能。

DenseNet的网络结构更为紧凑，加强了特征重用

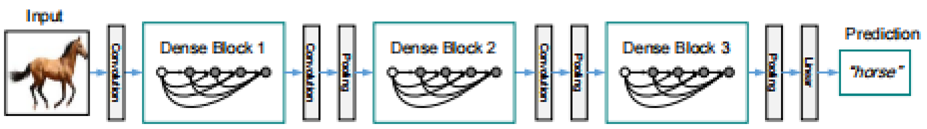
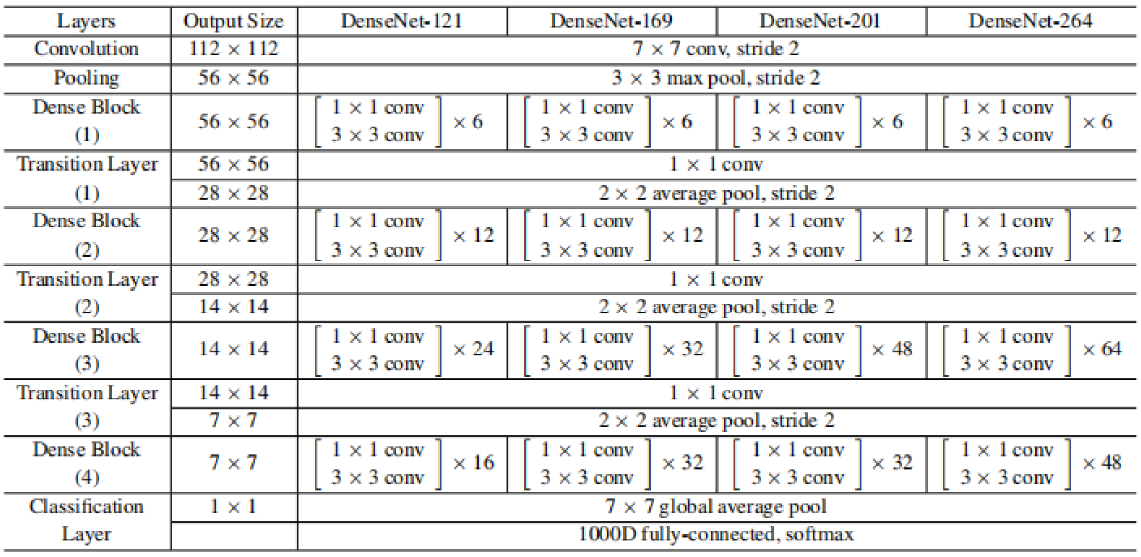
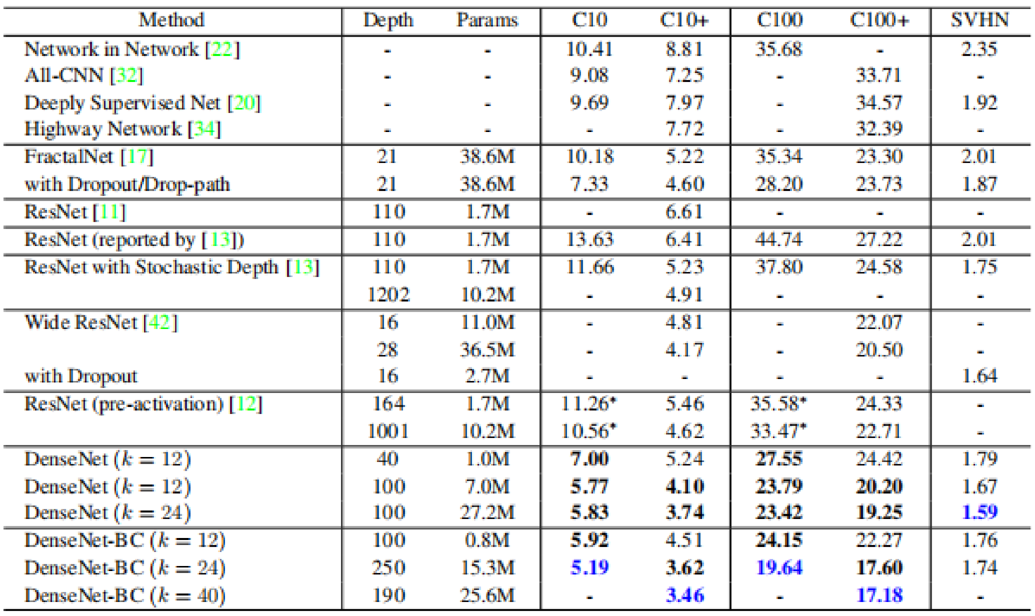


Figure 2: A deep DenseNet with three dense blocks. The layers between two adjacent blocks are referred to as transition layers and change feature-map sizes via convolution and pooling.

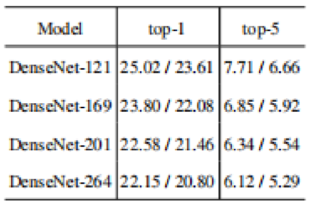


Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times6$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times6$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times6$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times12$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times12$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times12$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times24$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times32$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times48$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times64$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times16$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times32$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times32$	$\begin{bmatrix}1\times1\text{ conv}\\3\times3\text{ conv}\end{bmatrix}\times48$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			



Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26^*	5.46	35.58^*	24.33	-
	1001	10.2M	10.56^*	4.62	33.47^*	22.71	-
DenseNet ($k=12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k=12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k=24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k=12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k=24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k=40$)	190	25.6M	-	**3.46**	-	**17.18**	-

Table 2: Error rates (%) on CIFAR and SVHN datasets. k denotes network's growth rate. Results that surpass all competing methods are **bold** and the overall best results are **blue**. "+" indicates standard data augmentation (translation and/or mirroring). $*$ indicates results run by ourselves. All the results of DenseNets without data augmentation (C10, C100, SVHN) are obtained using Dropout. DenseNets achieve lower error rates while using fewer parameters than ResNet. Without data augmentation, DenseNet performs better by a large margin.



Model	top-1	top-5
DenseNet-121	25.02 / 23.61	7.71 / 6.66
DenseNet-169	23.80 / 22.08	6.85 / 5.92
DenseNet-201	22.58 / 21.46	6.34 / 5.54
DenseNet-264	22.15 / 20.80	6.12 / 5.29

Table 3: The top-1 and top-5 error rates on the ImageNet validation set, with single-crop / 10-crop testing.

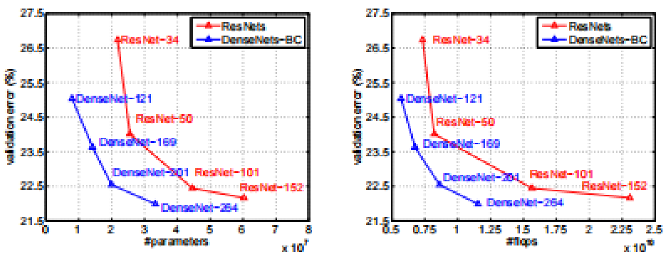


Figure 3: Comparison of the DenseNets and ResNets top-1 error rates (single-crop testing) on the ImageNet validation dataset as a function of learned parameters (*left*) and FLOPs during test-time (*right*).

个人理解

1.相比于ResNet，DenseNet将L层之前的L-1层的特征都得到了利用，在特征重用方面做的比ResNet要好一点，而且相较于直接加和这种信息传递，直接将之前的特征和该层特征拼接起来，信息的完整度更高

2.不足的一点是：由于需要cat操作，数据需要被复制，显存占用很容易增加