Deep Learning for Computer Vision

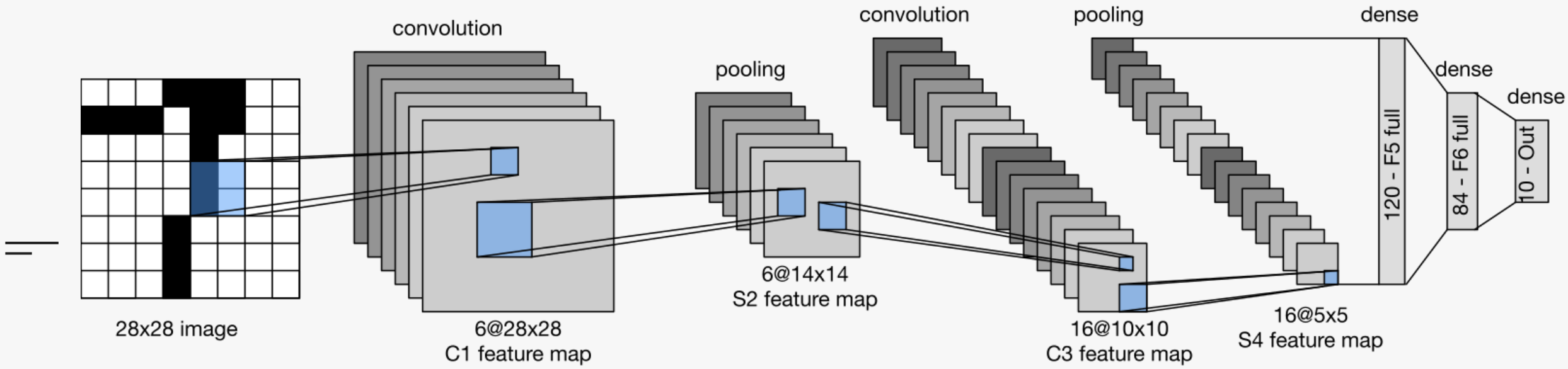
Ahmed Hosny Abdel-Gawad

Senior AI/CV Engineer

ConvNets Meta Architectures

LeNet

Vanilla ConvNet (LeNet)

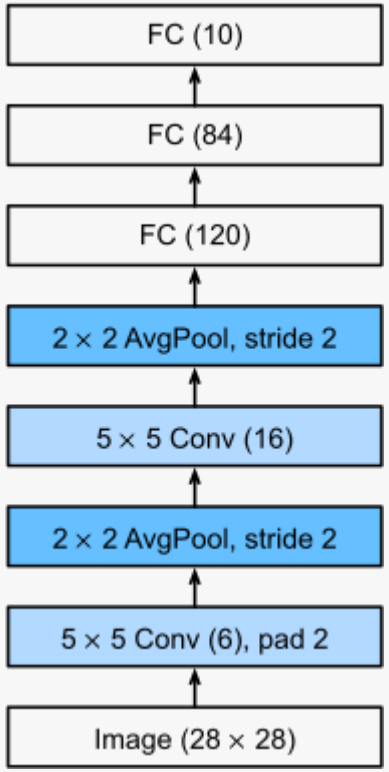


Instead of At a high level, LeNet (LeNet-5) consists of two parts:

1. A convolutional encoder consisting of **two** convolutional layers
2. A dense block (decoder) consisting of **three** fully connected layers

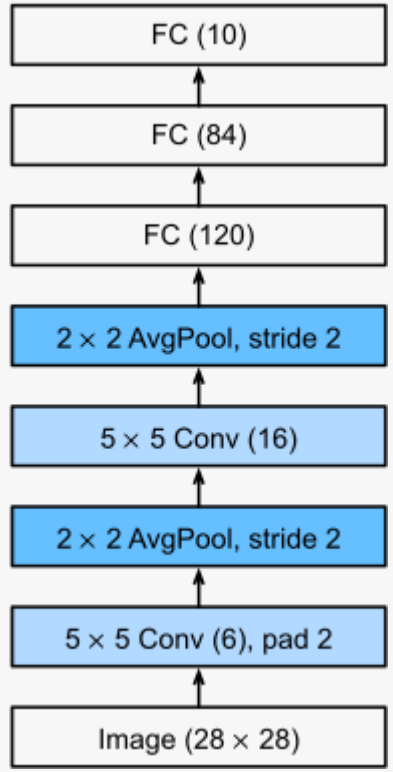
Vanilla ConvNet (LeNet)

- The **input** is images of size 28×28
- **C1** is the first convolutional layer with 6 convolution kernels of size 5×5.
- **S2** is the pooling layer that outputs 6 channels of 14×14 images. The pooling window size, in this case, is a square matrix of size 2×2.
- **C3** is a convolutional layer with 16 convolution kernels of size 5×5. Hence, the output of this layer is 16 feature images of size 10×10.
- **S4** is a pooling layer with a pooling window of size 2×2. Hence, the dimension of images through this layer is halved, it outputs 16 feature images of size 5×5.



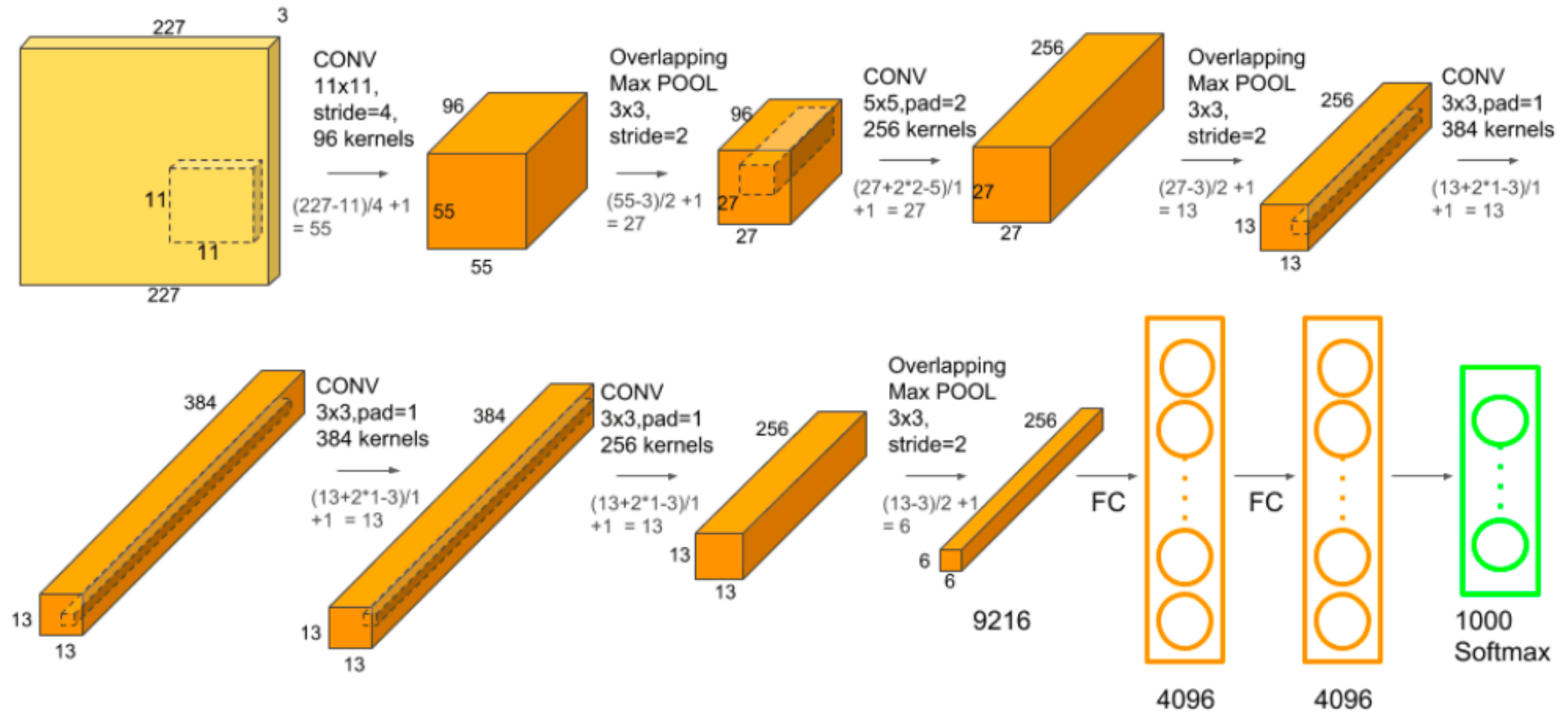
Vanilla ConvNet (LeNet)

- **F5** is a fully connected layer with 120 neurons which are all connected to the flattened output of C4.
- **F6** is a fully connected layer with 84 neurons which are all connected to the output of F5.
- The **output** layer consists of 10 neurons corresponding to the number of classes (numbers from 0 to 9).



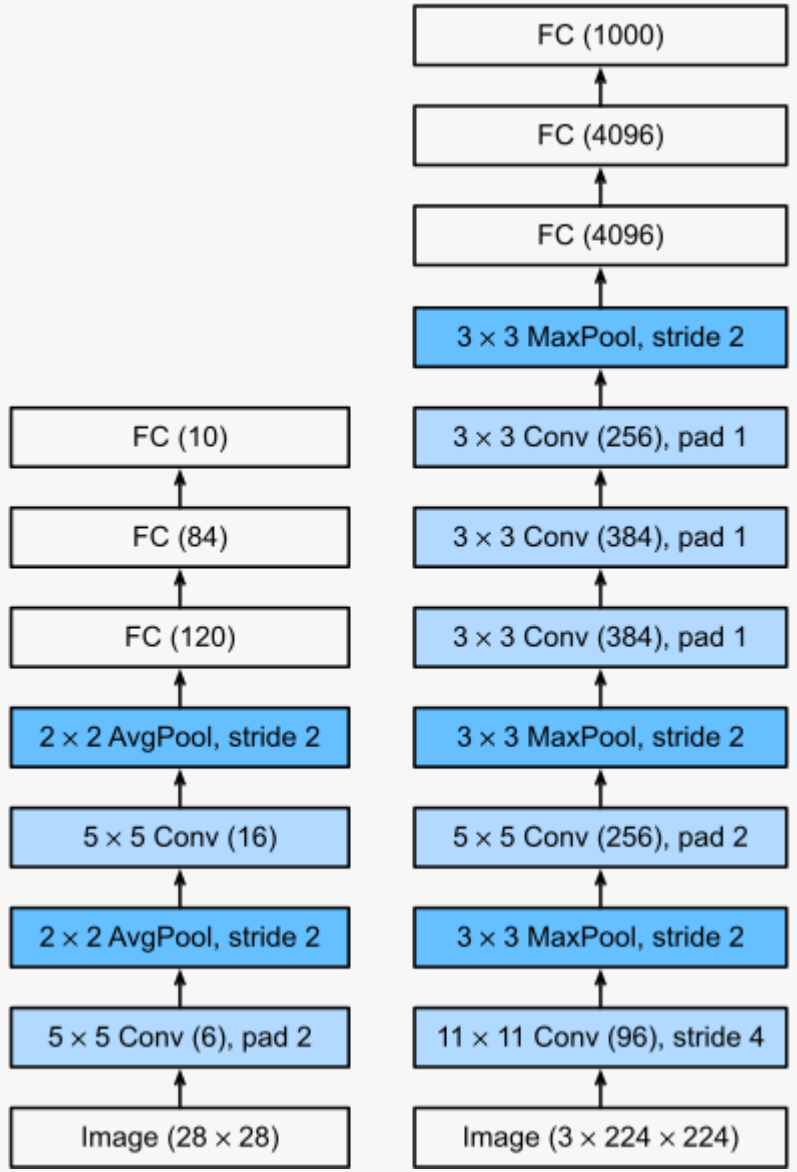

AlexNet

Deep ConvNet (AlexNet)



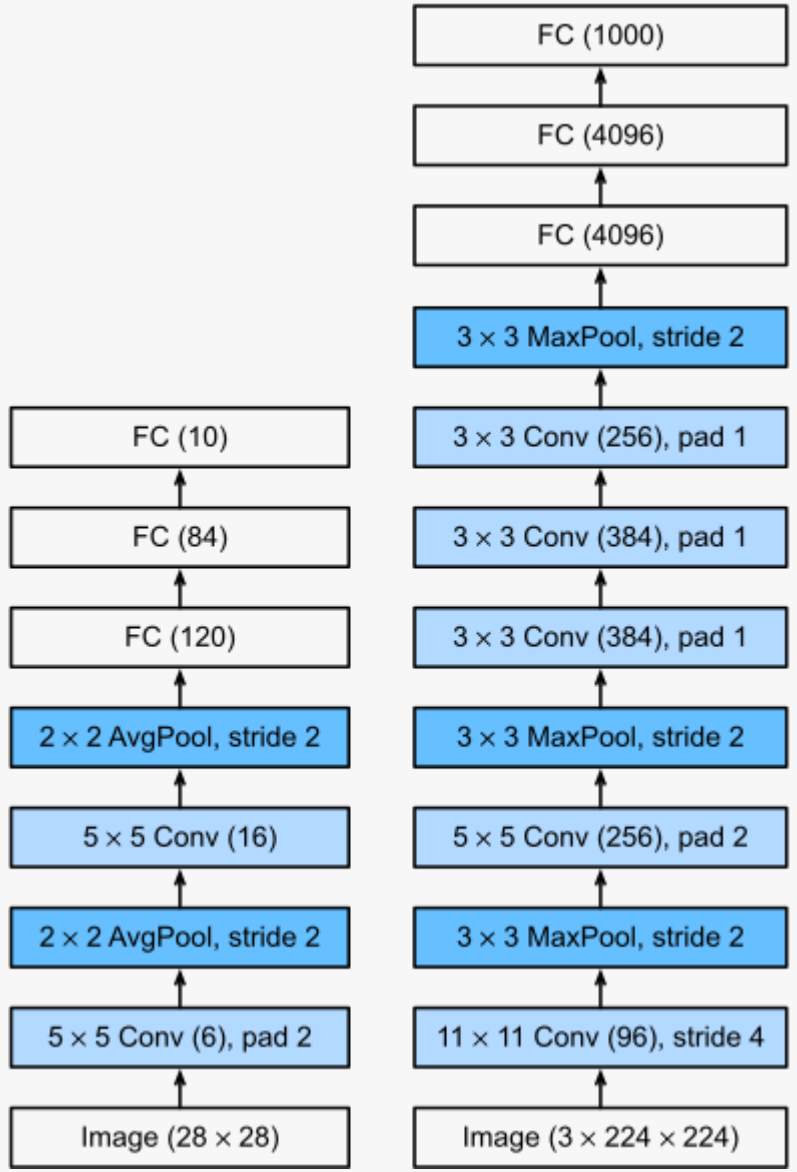
Deep ConvNet (AlexNet)

- In AlexNet's first layer, the convolution window shape is **11x11**, It is because of input size is large, so we need to use a large kernel to capture the object.
- The convolution window shape in the second layer is reduced to **5x5**, followed by **3x3**.
- In addition, after the first, second, and fifth convolutional layers, the network adds max-pooling layers with a window shape of and a stride of 2.
- Moreover, AlexNet has ten times more convolution channels than LeNet.



Deep ConvNet (AlexNet)

- AlexNet controls the model complexity of the fully connected layer by dropout with ratio of 50%.
- To augment the data even further, the training loop of AlexNet added a great deal of image augmentation, such as flipping, clipping, and color changes.

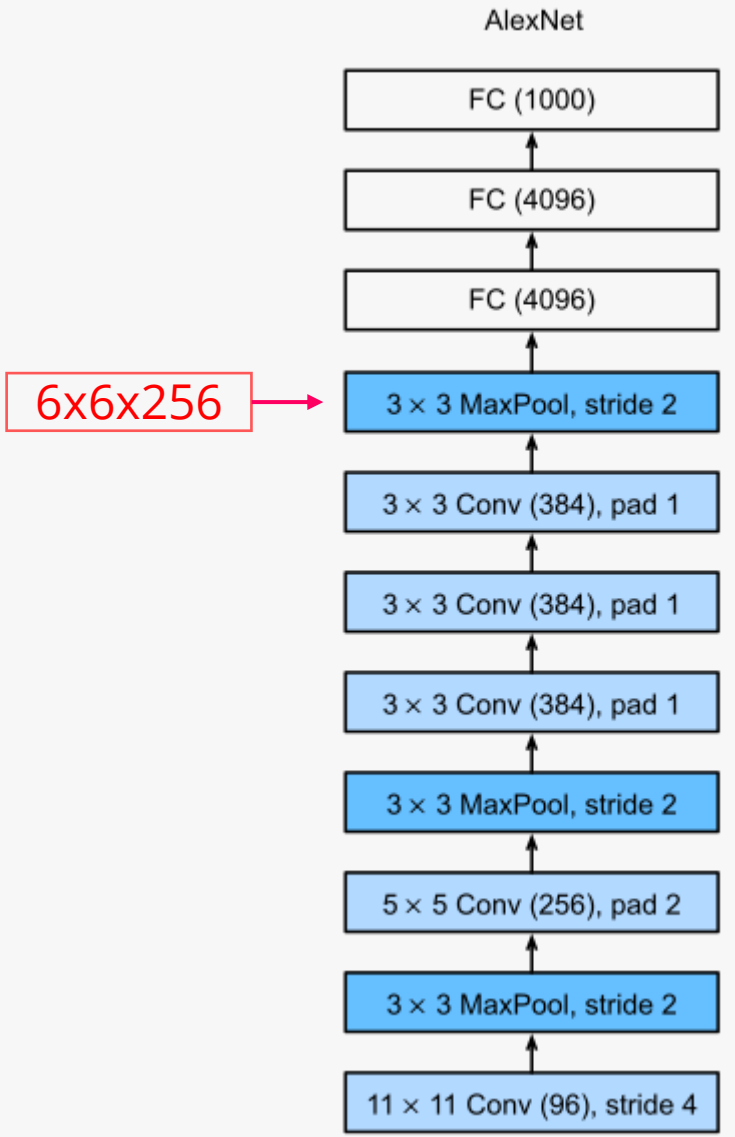


VGGNet

VGG Block

One of the problems with this approach is that the spatial resolution decreases quite rapidly.

For instance, in the case of ImageNet, it would be impossible to have more than 8 convolutional layers in this way.



VGG Block

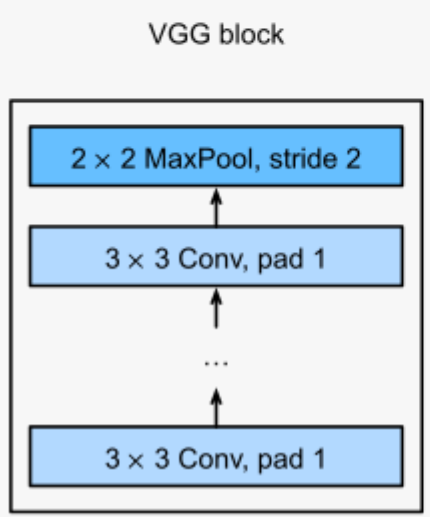
The key idea was to use multiple convolutions in between down-sampling via max-pooling in the form of a block. **The primarily interest in whether deep or wide networks perform better?**

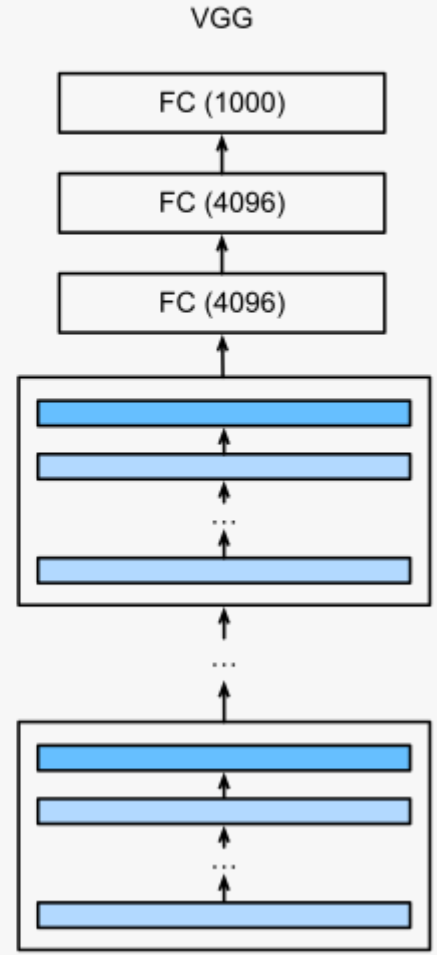
For instance, the successive application of **two 3×3** convolutions touches the **same** pixels as a single **5×5** convolution does.

The **5×5** uses approximately $1\text{x}(5\text{x}5\text{x}c) + 1 = 25c + 1$ parameters as three convolutions do $2\text{x}(3\text{x}3\text{x}c) + 2 = 18c + 2$.

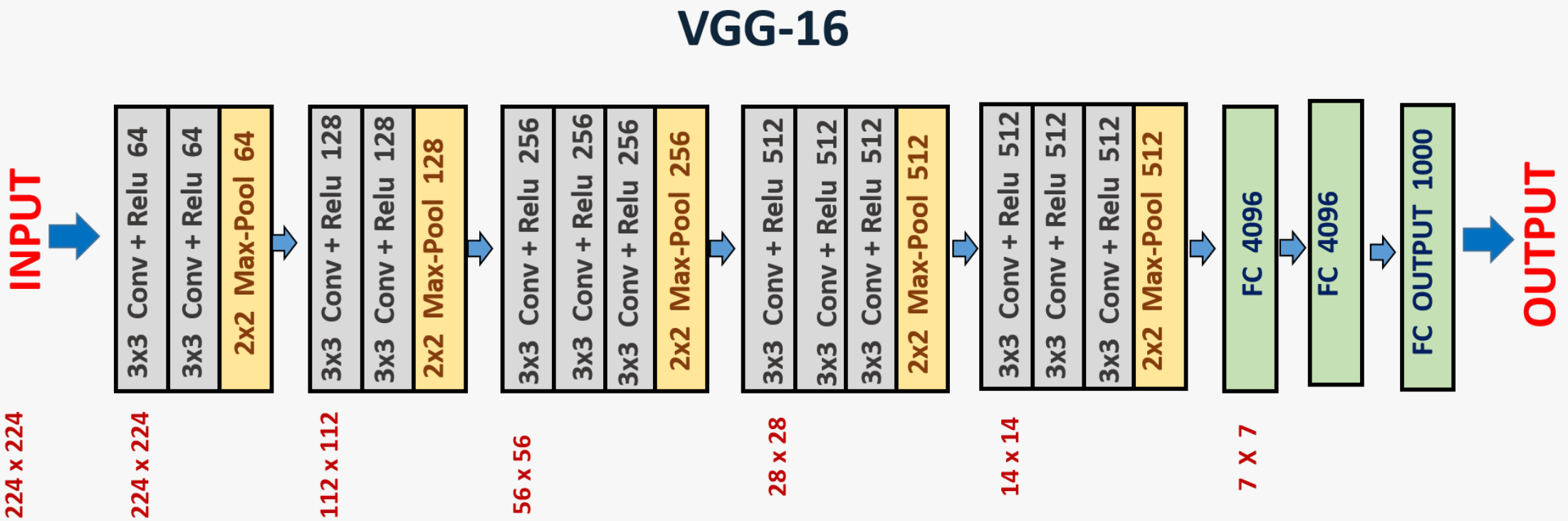
$$\text{Filters} * (K\text{x}K\text{x}C) + \text{Filters}$$

VGG block consists of a sequence of convolutions with **3×3** kernels with padding of **1** (keeping height and width) followed by a **2×2** max-pooling layer with stride of **2** (halving height and width after each block).





Networks Using Blocks (VGG)



VGG16 is composed of 13 convolutional layers, 5 max-pooling layers, and 3 fully connected layers. Therefore, the number of layers having tunable parameters is 16 (13 convolutional layers and 3 fully connected layers).

VGG Networks

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

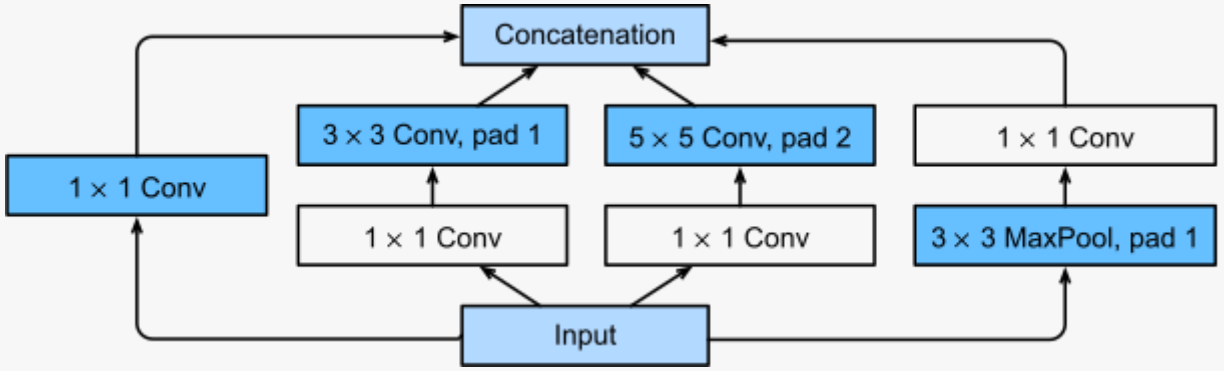
Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144


InceptionNet

Inception Blocks

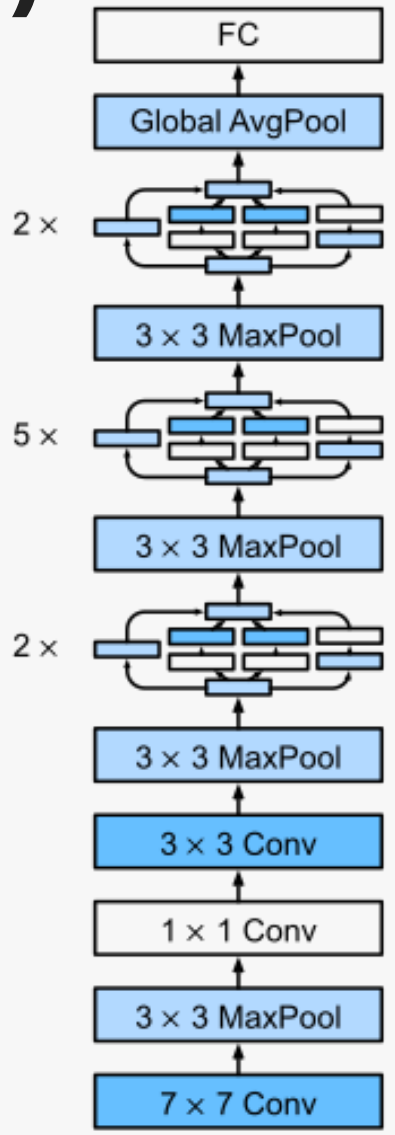
Inception block consists of four parallel paths at which convolution layers with different kernel sizes:

- The first path uses a convolutional layer with a window size of 1 × 1.
- In the second and the third paths, a convolutional layer of size 1 × 1 is used before applying two expensive 3 × 3 and 5 × 5 convolutions. The 1×1 convolution helps to reduce the number of filter channels, thus reducing the model complexity.
- The fourth path uses a max-pooling layer to reduce the resolution of the input, and it is followed by a 1 × 1 convolutional layer to reduce the dimension.

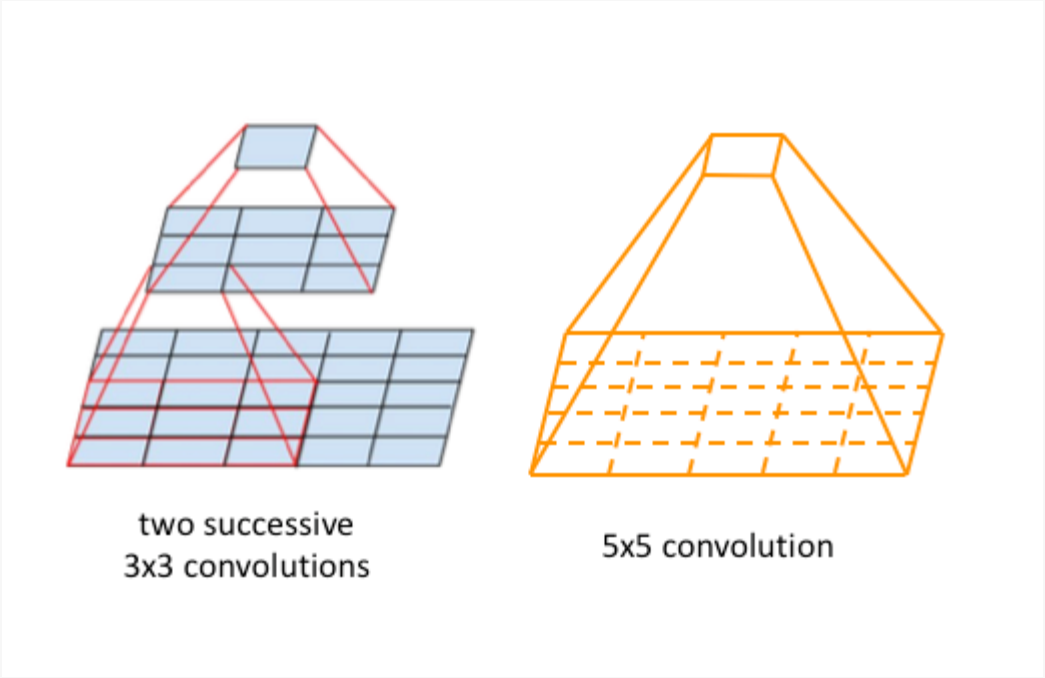


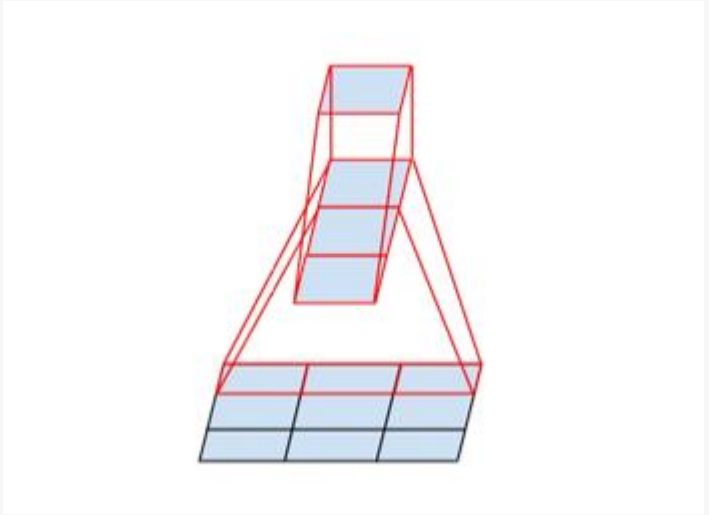
Multi-Branch Networks (GoogLeNet)

- The input size image is 224 × 224.
- There are nine Inception blocks in this network.
- There are four max-pooling layers outside the Inception blocks, in which two layers are located between blocks 3–4 and block 7–8. These max-pooling layers help to reduce the size of the input data, thus reduce the model complexity as well as the computational cost.
- This network uses the idea of using an average pooling layer, which helps to improve the model performance and reduce overfitting.
- A dropout layer (with 40%) is utilized before the linear layer. This is also an efficient regularization method to reduce the overfitting phenomena.
- The output layer uses the softmax activation function to give 1000 outputs which are corresponding to the number of categories in the ImageNet dataset.



Modified GoogLeNet

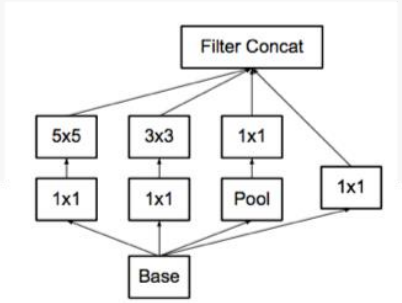


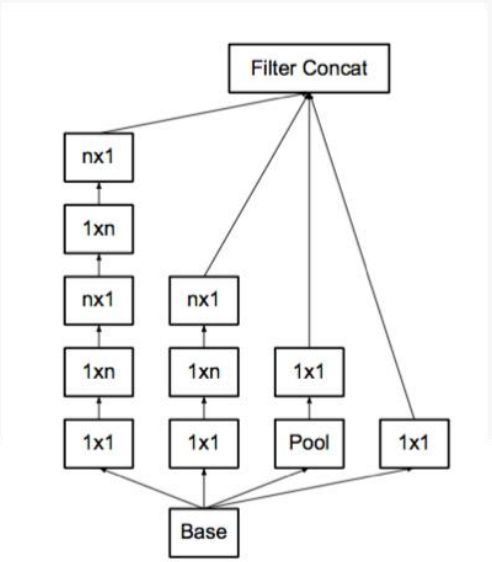


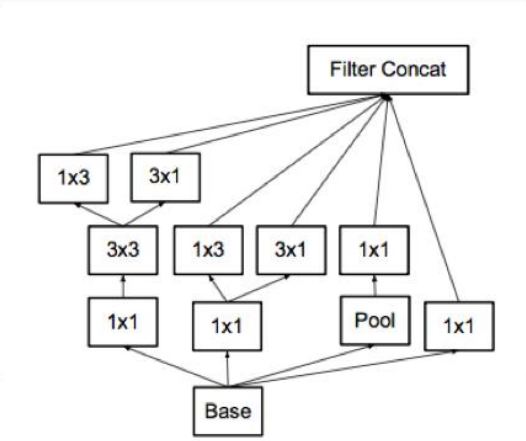
For instance, the successive application of **two 3×3** convolutions touches the **same** pixels as a single **5×5** convolution does.

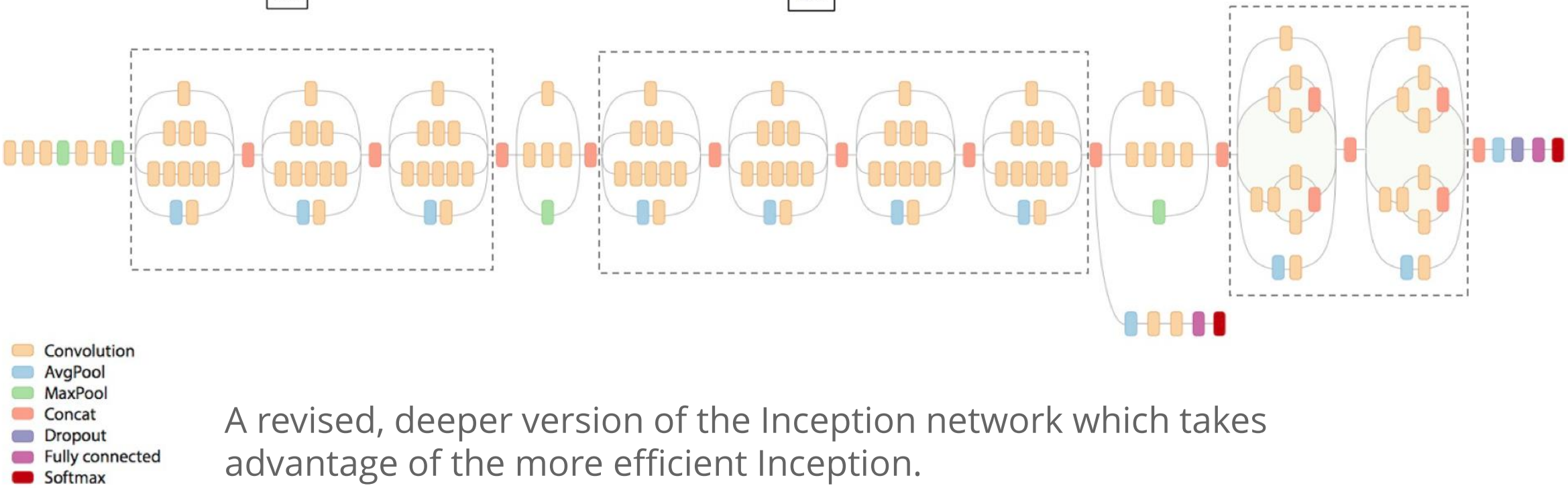
It was also shown that **3x3** convolutions could be further deconstructed into successive **3x1** and **1x3** convolutions.

Deeper GoogleNet



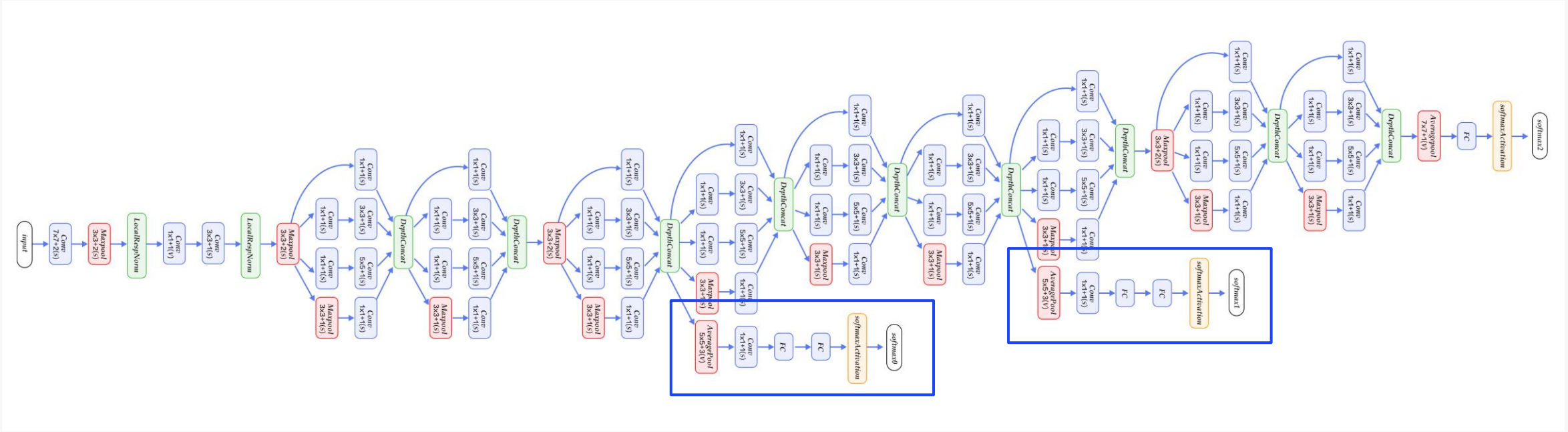






A revised, deeper version of the Inception network which takes advantage of the more efficient Inception.

GoogLeNet (Auxiliary Loss)



link...

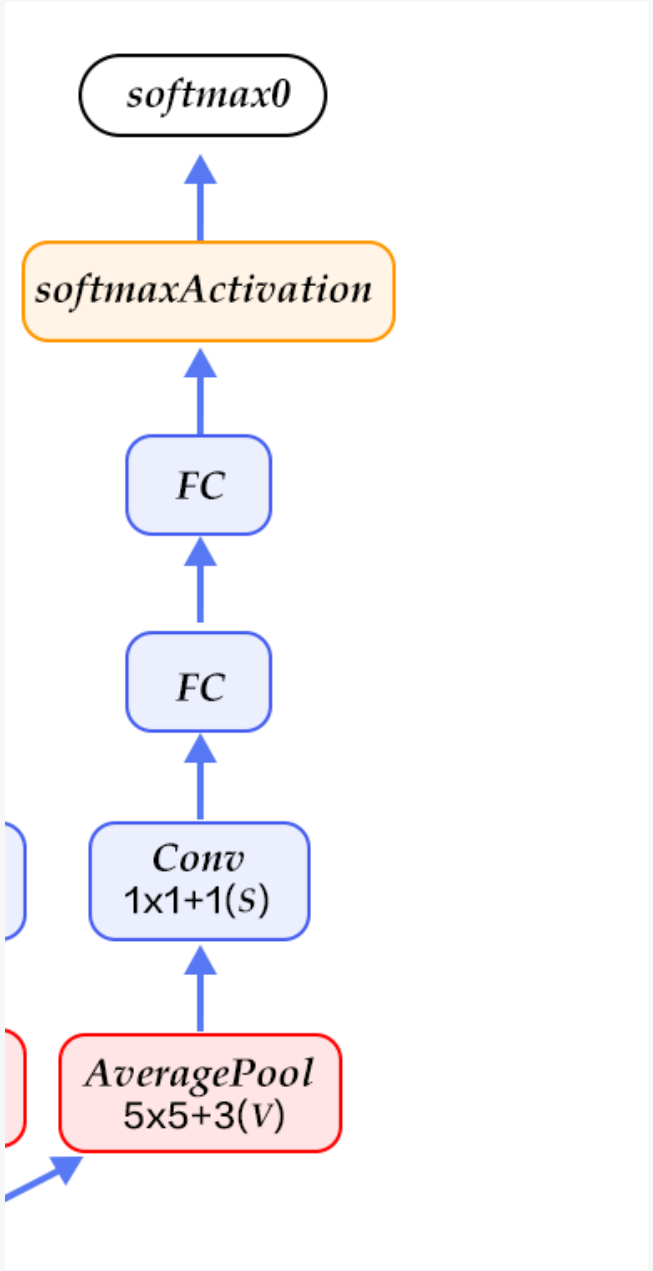
GoogLeNet (Auxiliary Loss)

In order to improve overall network performance, two auxiliary outputs are added throughout the network.

It was later discovered that the earliest auxiliary output had no discernible effect on the final quality of the network.

The addition of auxiliary outputs primarily benefited the end performance of the model, converging at a slightly better value than the same network architecture without an auxiliary branch.

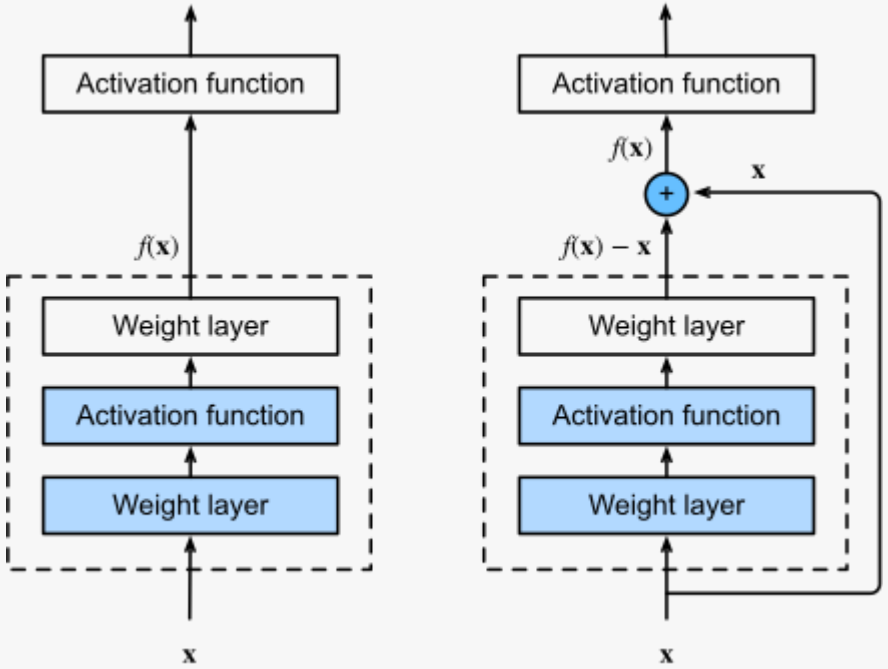
It is believed the addition of auxiliary outputs had a regularizing effect on the network.



ResNet

Residual Blocks

- On the left, the portion within the dotted-line box must directly learn the mapping .
- On the right, the portion within the dotted-line box needs to learn the residual mapping , which is how the residual block derives its name.
- Till GoogleNet we can build **18 layers** only, with the skip connection we can build **150 layers**!



ResNet Model

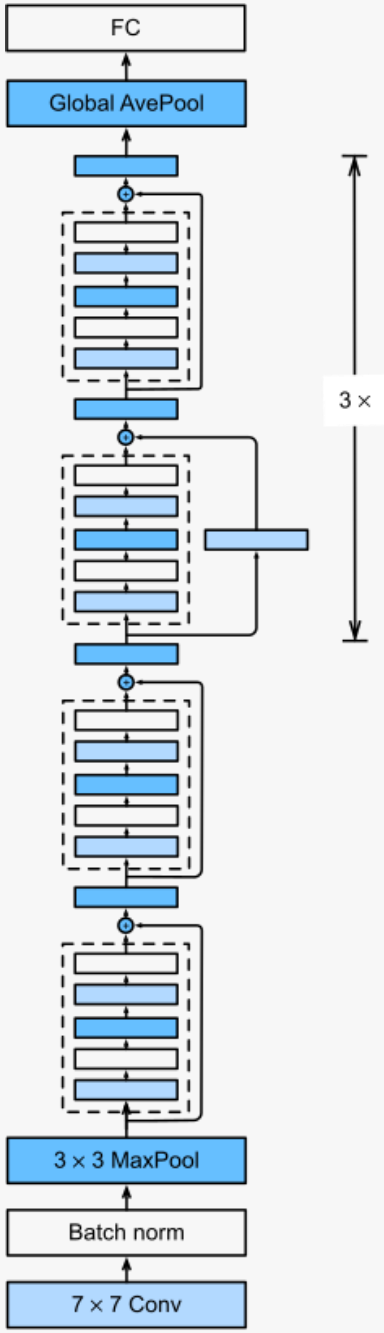
The first two layers of ResNet are the same as those of the GoogLeNet we described before: the **7×7** convolutional layer with 64 output channels and a stride of 2 is followed by the **3×3** max-pooling layer with a stride of 2.

The difference is the **batch normalization** layer added after each convolutional layer in ResNet.

ResNet uses four modules made up of residual blocks, each of which uses several residual blocks with the same number of output channels.

The number of channels in the first module is the same as the number of input channels. Since a max-pooling layer with a stride of 2 has already been used, it is not necessary to reduce the height and width.

In the first residual block for each of the subsequent modules, the number of channels is doubled compared with that of the previous module, and the height and width are halved.



Thank You!