VGG

Computer Vision

Carnegie Mellon University (Kris Kitani)

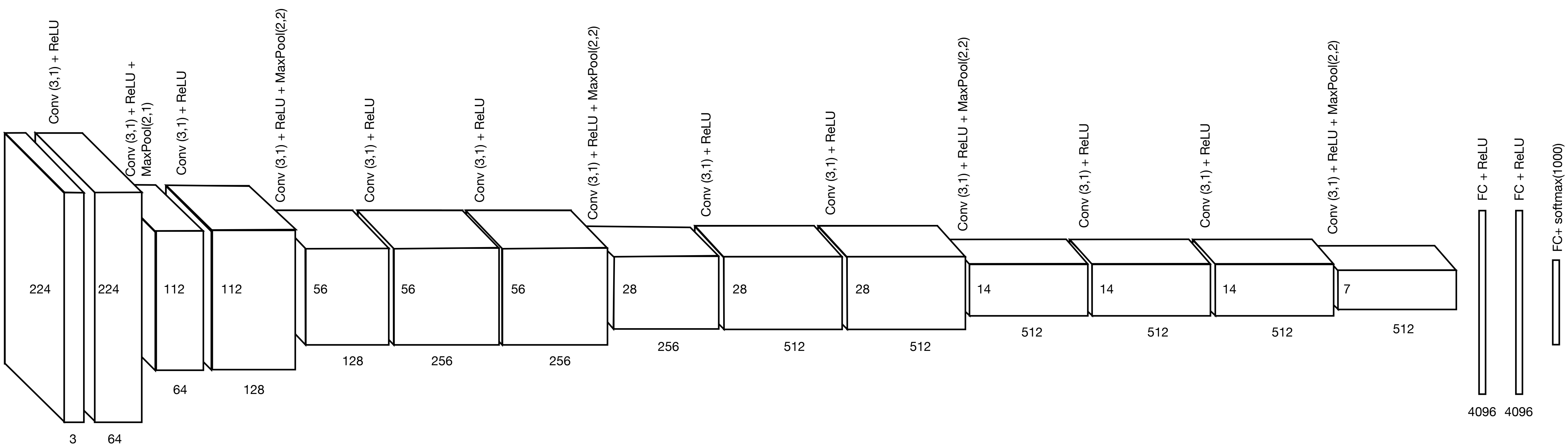
VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION

Karen Simonyan* & **Andrew Zisserman+**
Visual Geometry Group, Department of Engineering Science, University of Oxford
{karen,az}@robots.ox.ac.uk

ABSTRACT

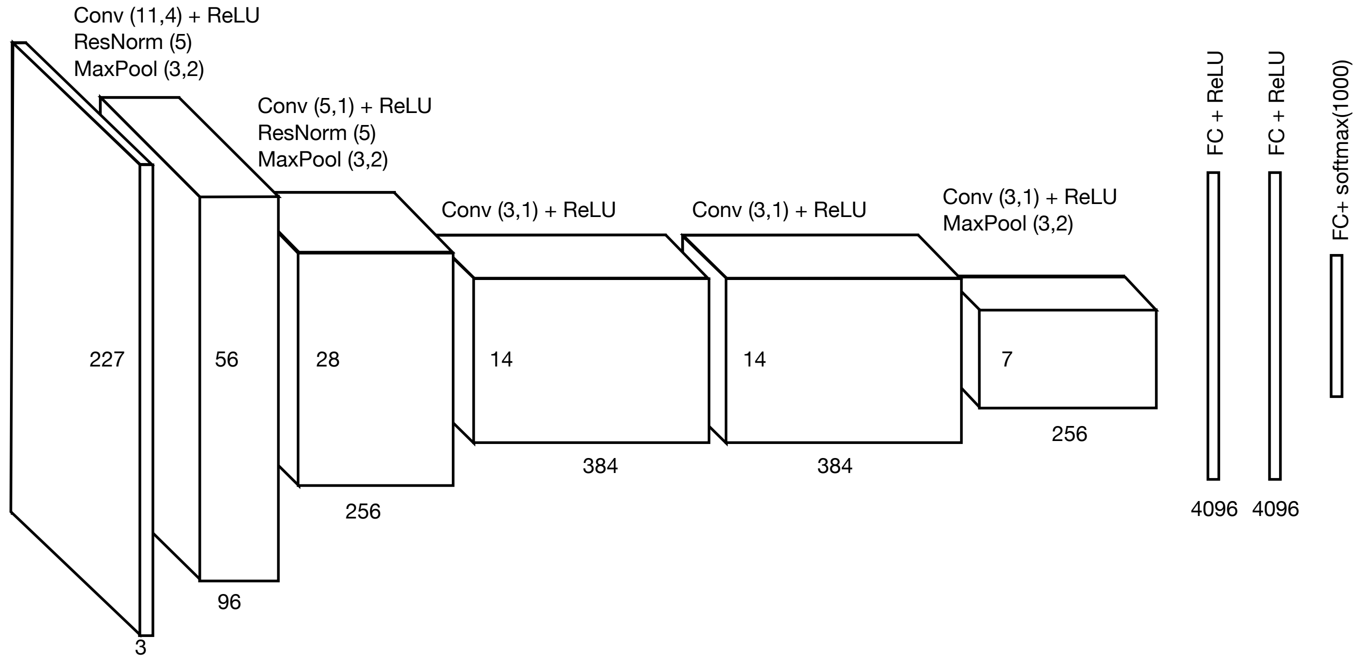
In this work we investigate the effect of the convolutional network depth on its accuracy in the large-scale image recognition setting. Our main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small (3×3) convolution filters, which shows that a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16–19 weight layers. These findings were the basis of our ImageNet Challenge 2014 submission, where our team secured the first and the second places in the localisation and classification tracks respectively. We also show that our representations generalise well to other datasets, where they achieve state-of-the-art results. We have made our two best-performing ConvNet models publicly available to facilitate further research on the use of deep visual representations in computer vision.

VGG-16



First thing to notice…

AlexNet

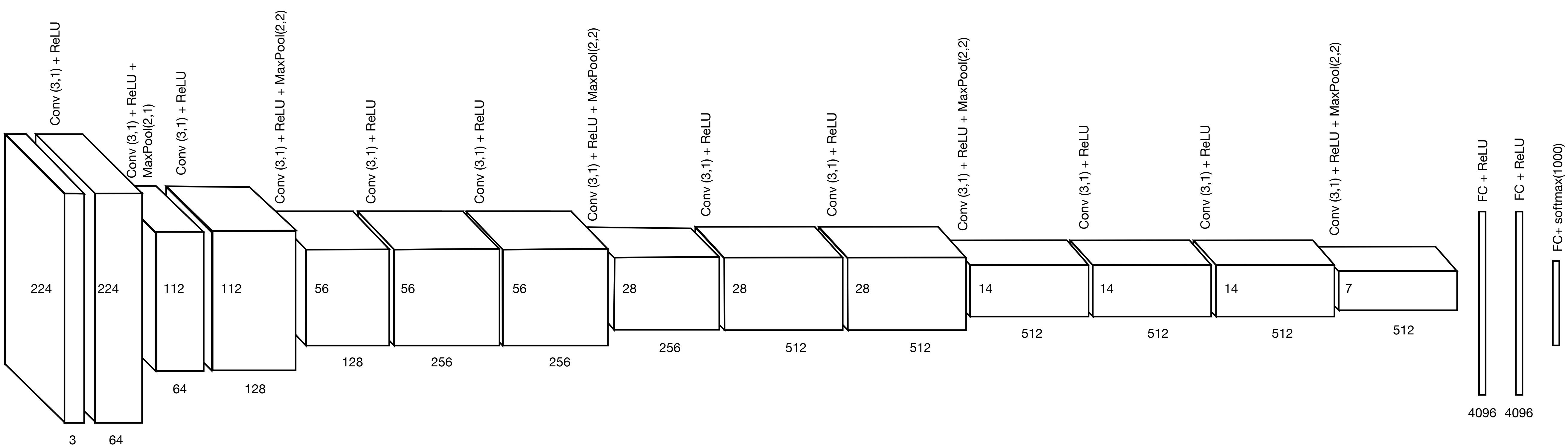


8 layers

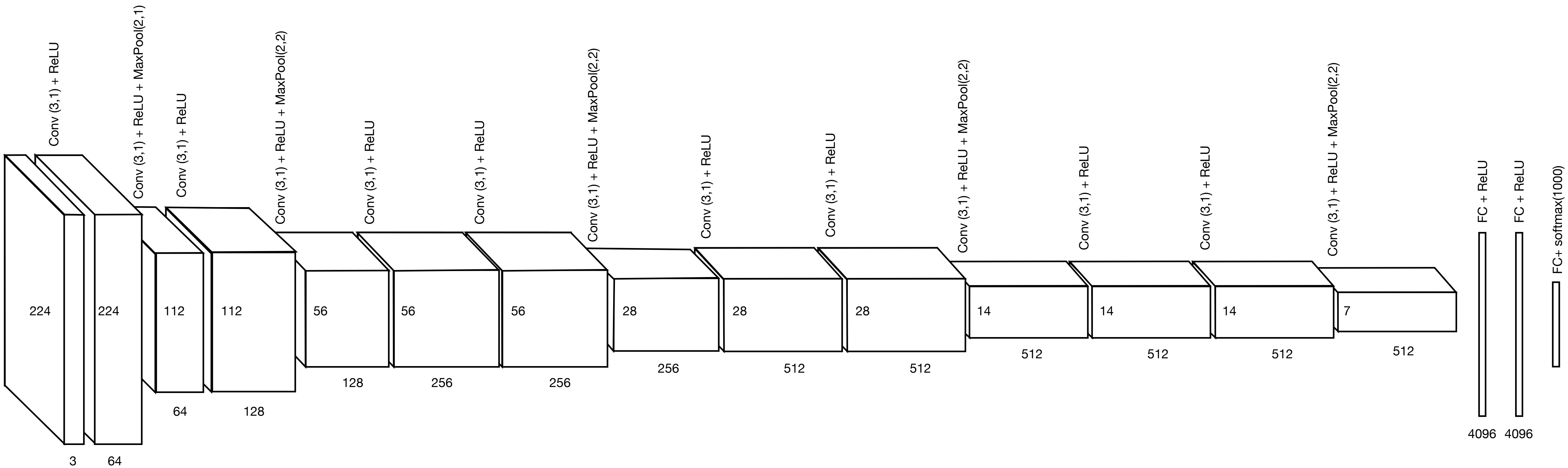
deeper!

16 layers

VGG-16

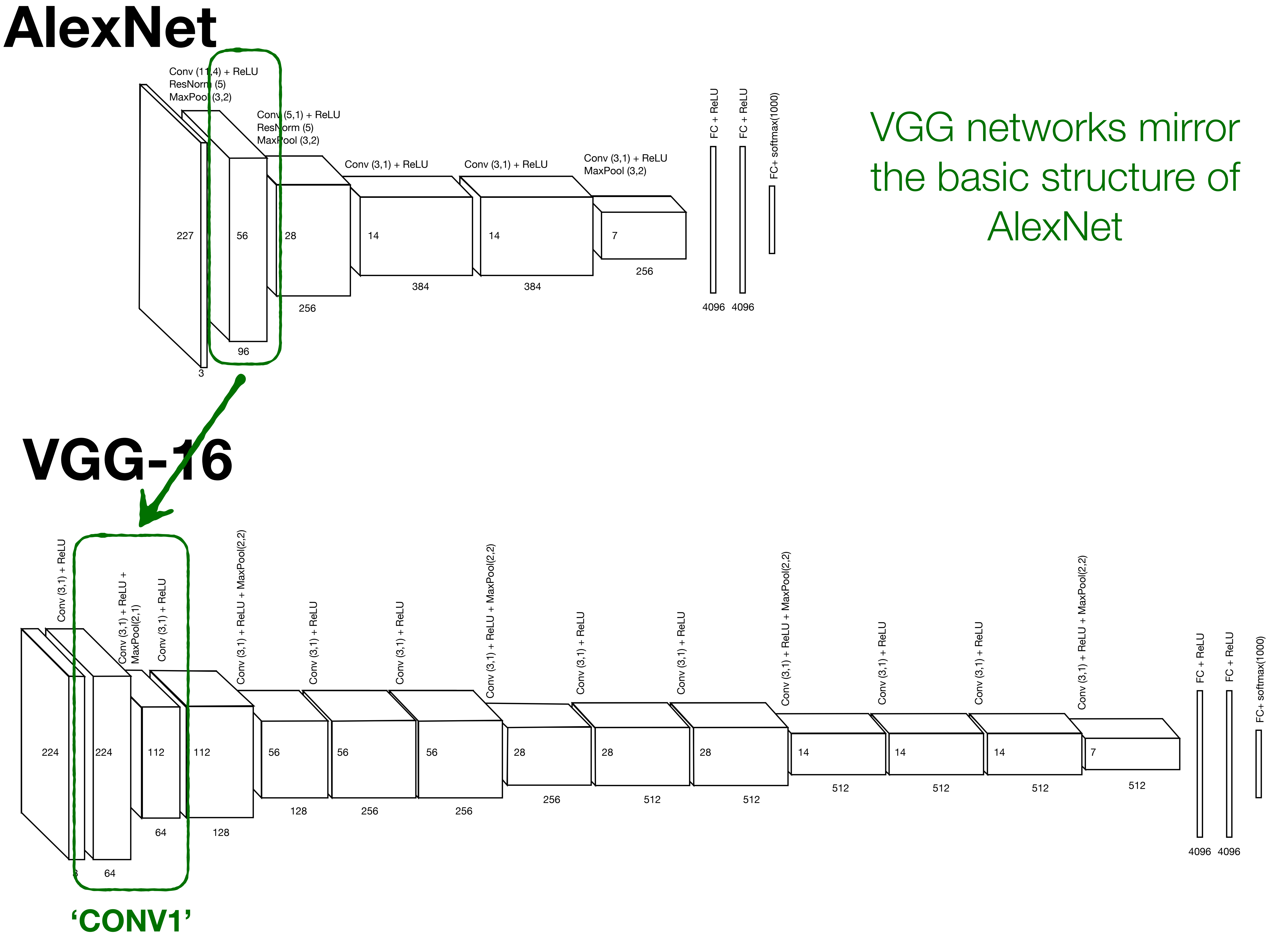


VGG-16



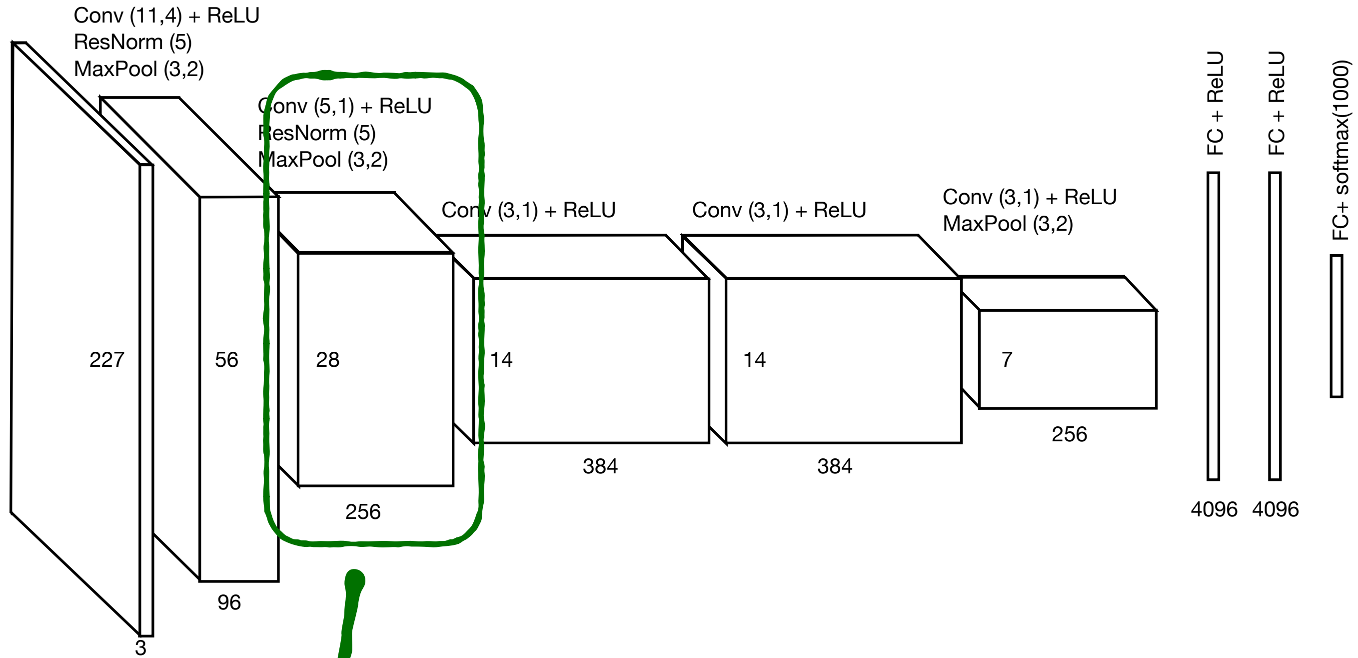
Emerging ‘rule of thumb’

- Convolutions all standardized to 3 x 3
- Convolutions always followed by ReLU
- Stack several convolutions at the same resolution
- Downsample by 2, increase channels by 2
- No local response normalization



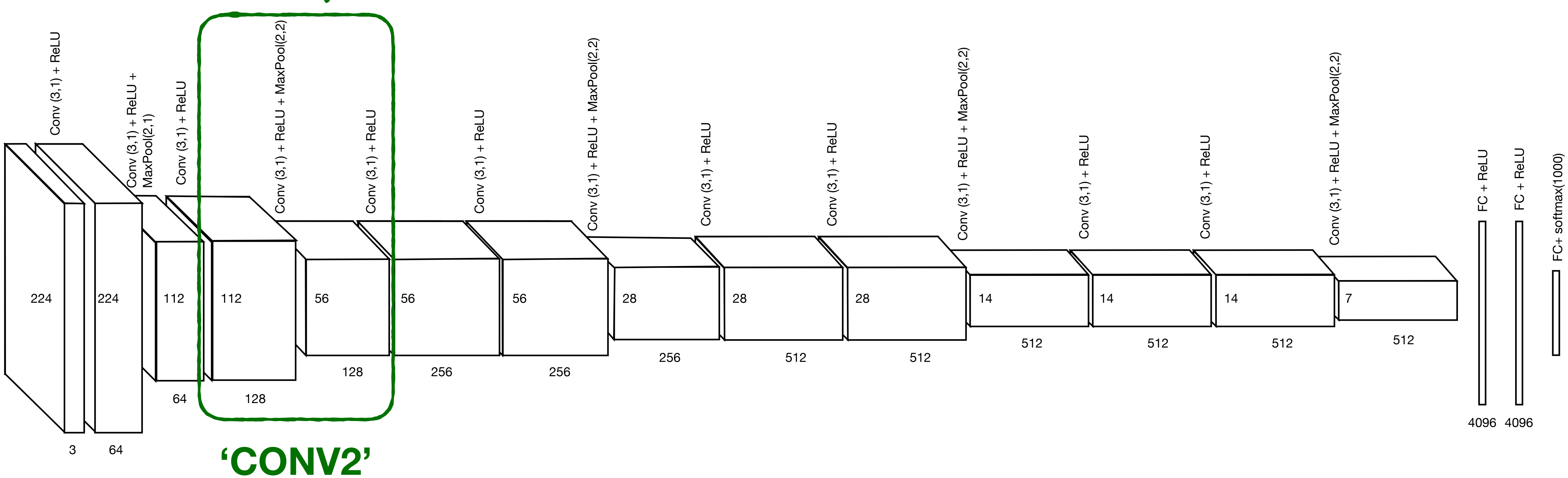
AlexNet
Conv (11,4) + ReLU
ResNorm (5)
MaxPool (3,2)
Conv (5,1) + ReLU
ResNorm (5)
MaxPool (3,2)
Conv (3,1) + ReLU
Conv (3,1) + ReLU
Conv (3,1) + ReLU
MaxPool (3,2)
FC + ReLU
FC + ReLU
FC+ softmax(1000)
227
3
56
96
28
256
14
384
14
384
7
256
4096
4096
VGG networks mirror the basic structure of AlexNet
VGG-16
Conv (3,1) + ReLU
Conv (3,1) + ReLU + MaxPool(2,1)
Conv (3,1) + ReLU
Conv (3,1) + ReLU + MaxPool(2,2)
Conv (3,1) + ReLU
Conv (3,1) + ReLU
Conv (3,1) + ReLU + MaxPool(2,2)
Conv (3,1) + ReLU
Conv (3,1) + ReLU
Conv (3,1) + ReLU + MaxPool(2,2)
Conv (3,1) + ReLU
Conv (3,1) + ReLU
Conv (3,1) + ReLU + MaxPool(2,2)
FC + ReLU
FC + ReLU
FC+ softmax(1000)
224
3
224
64
112
64
112
128
56
128
56
256
56
256
28
256
28
512
28
512
14
512
14
512
14
512
7
512
4096
4096
'CONV1'

AlexNet

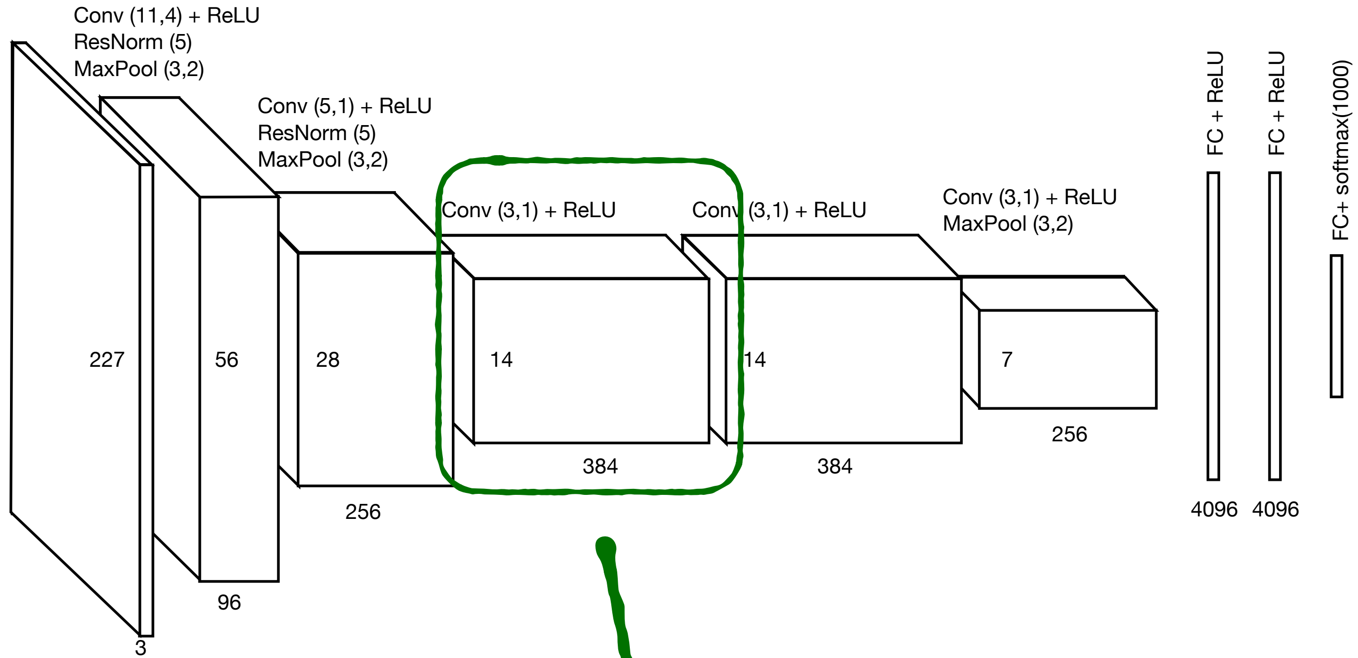


VGG networks mirror the basic structure of AlexNet

VGG-16

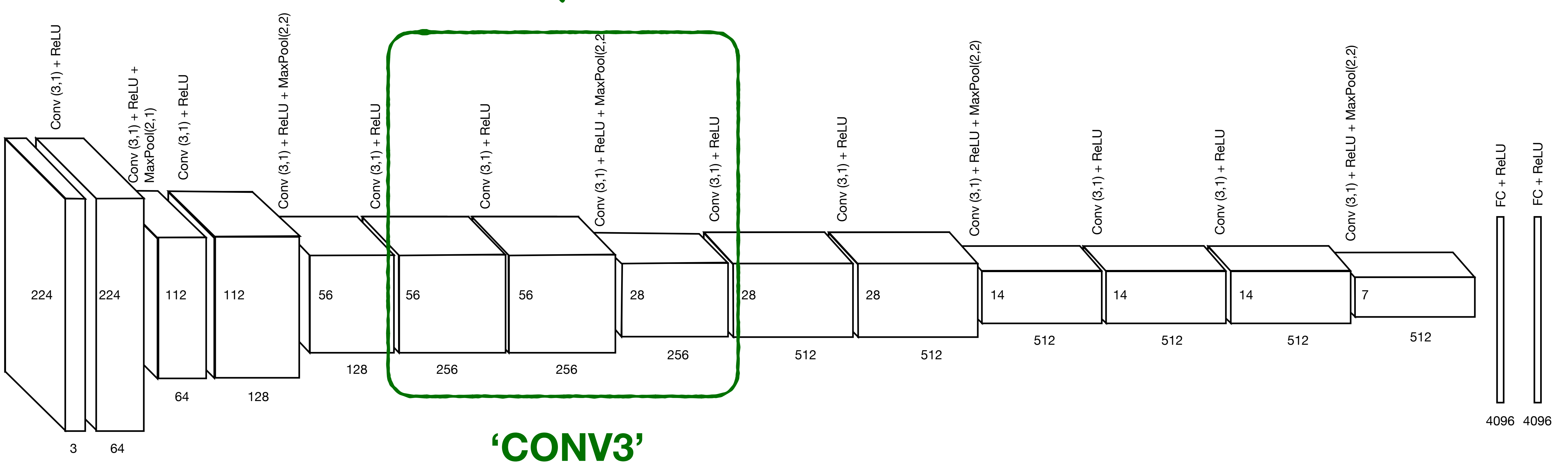


AlexNet

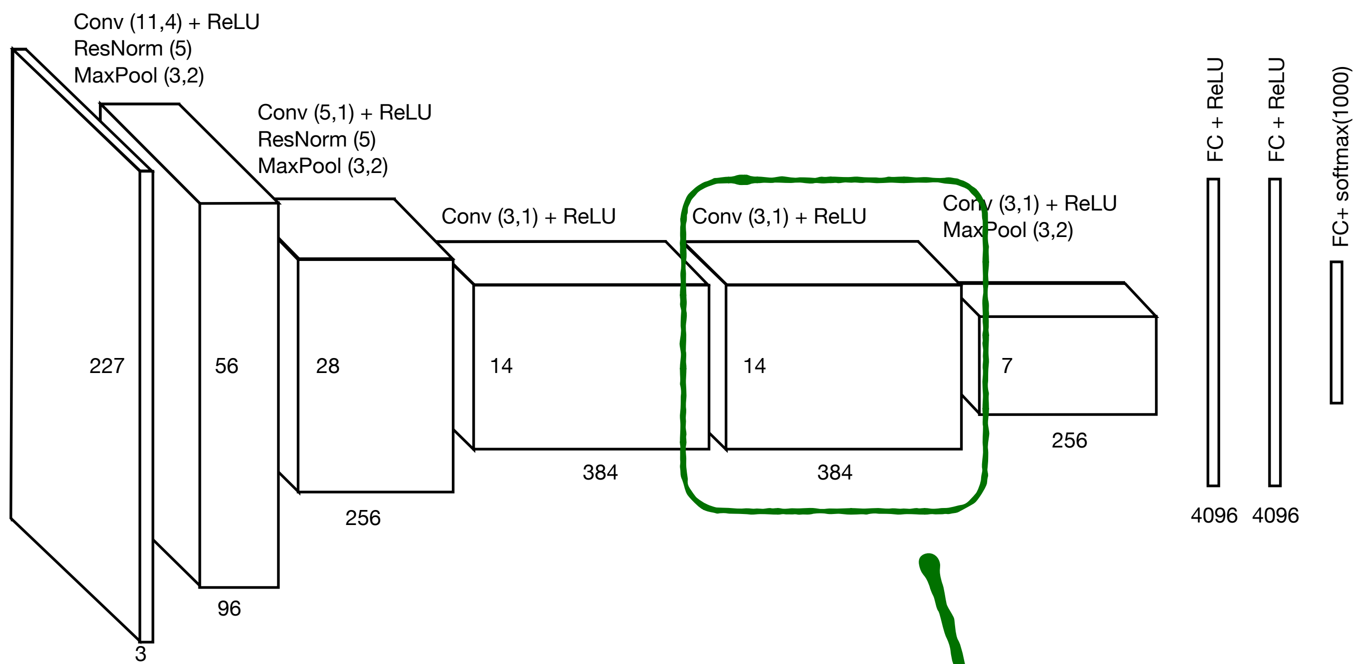


VGG networks mirror the basic structure of AlexNet

VGG-16

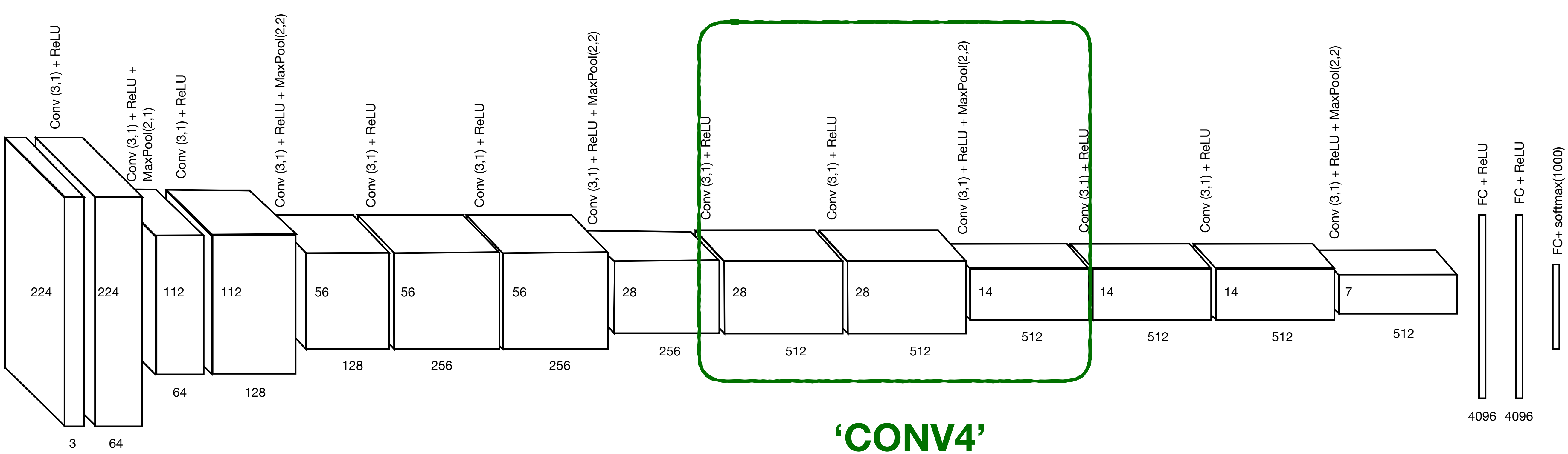


AlexNet



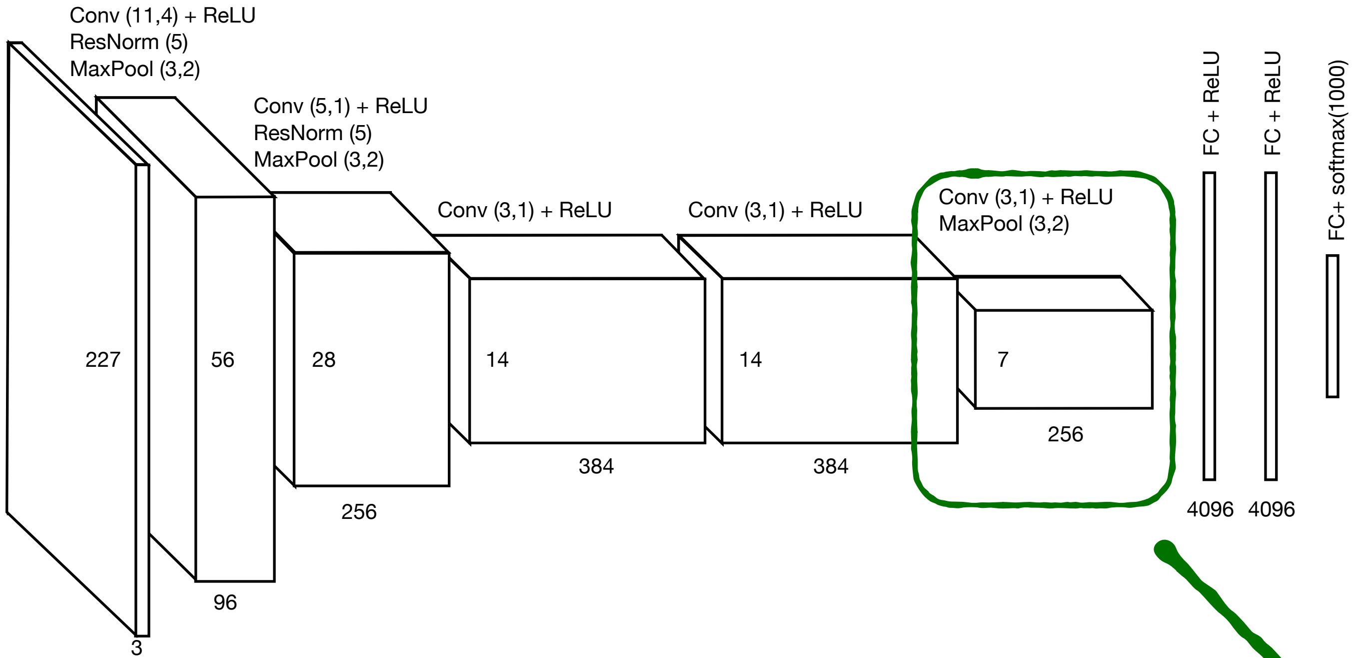
VGG networks mirror the basic structure of AlexNet

VGG-16



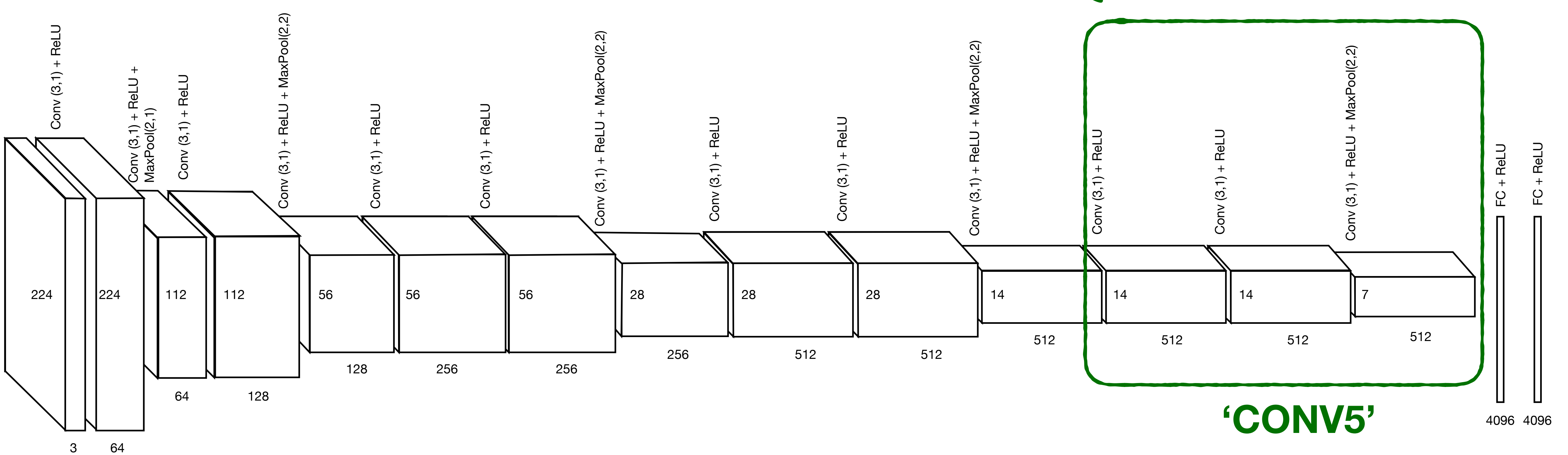
‘CONV4’

AlexNet

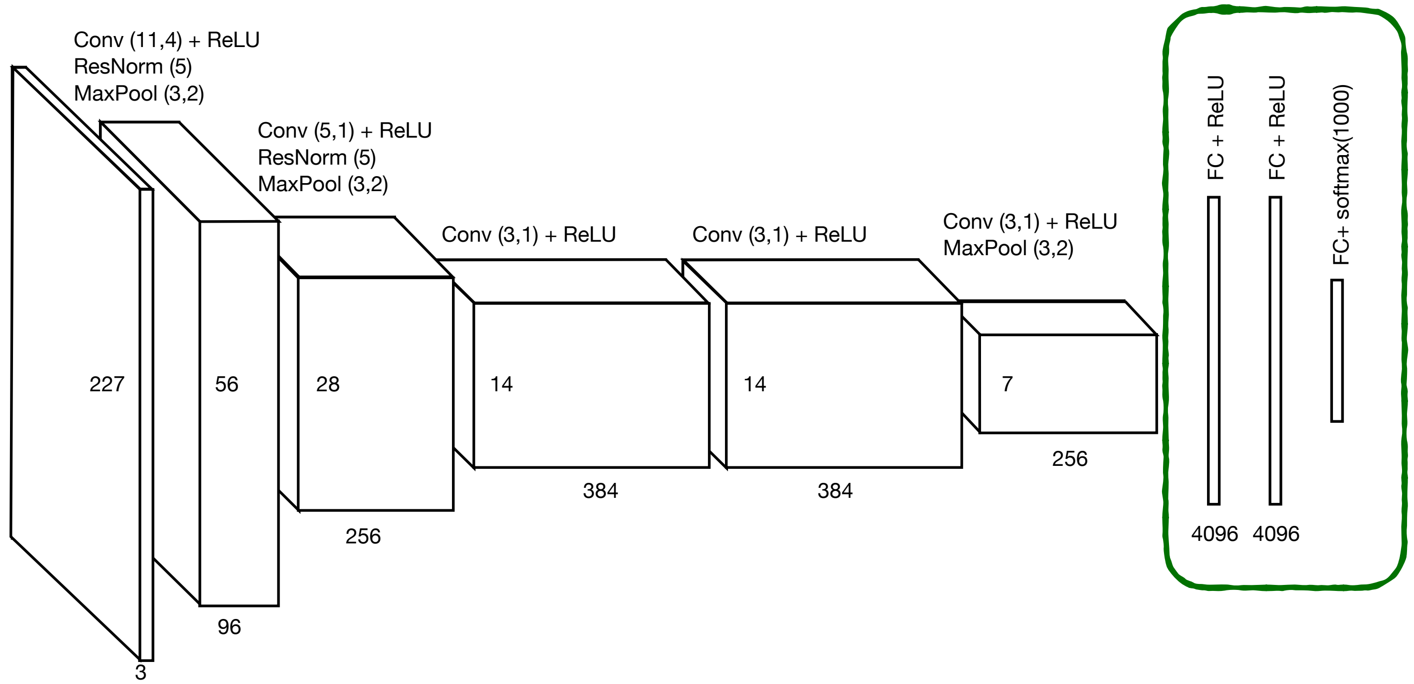


VGG networks mirror the basic structure of AlexNet

VGG-16

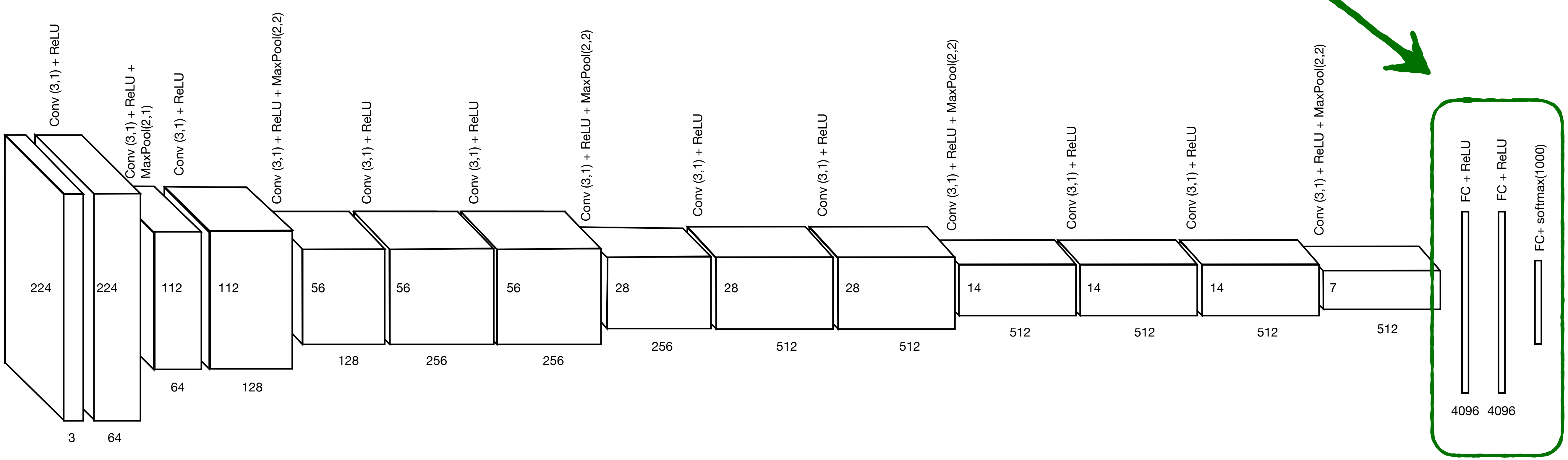


AlexNet



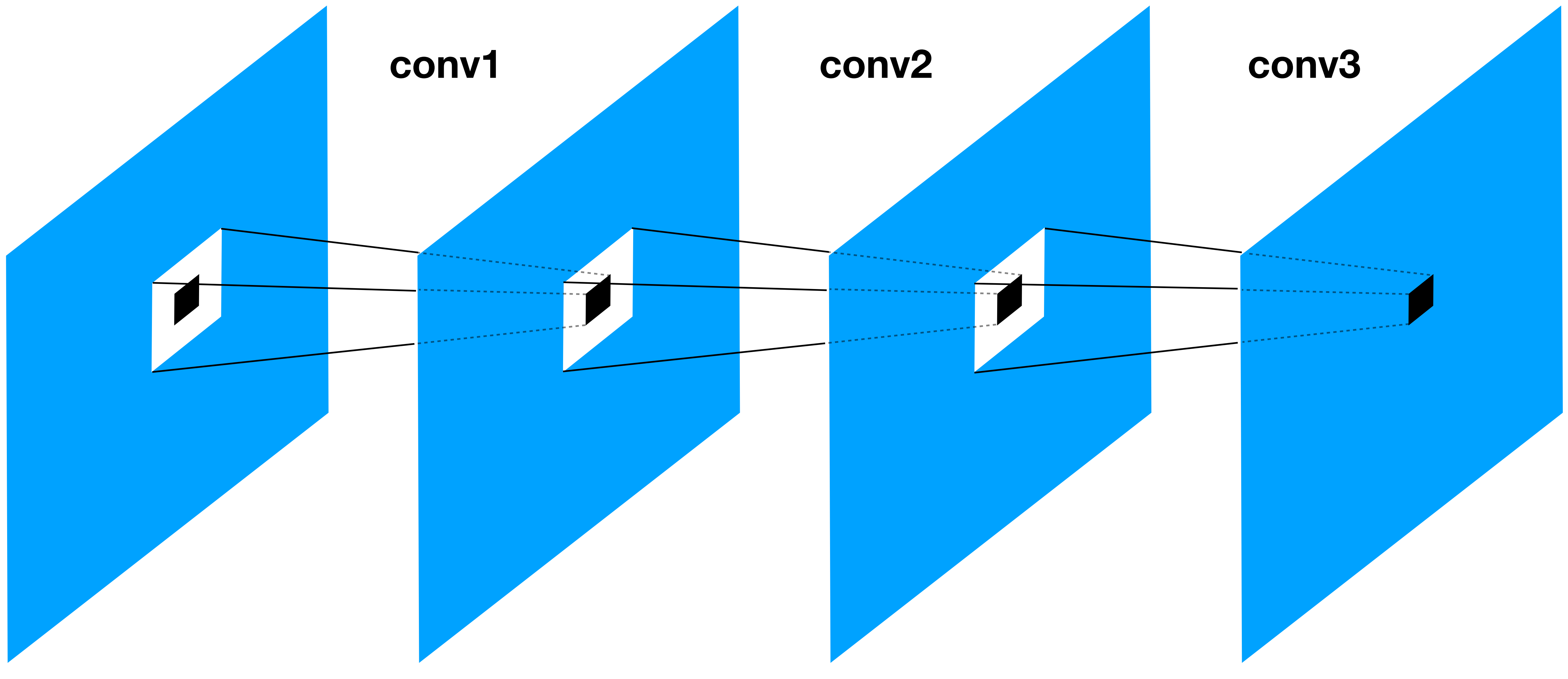
VGG networks mirror the basic structure of AlexNet

VGG-16

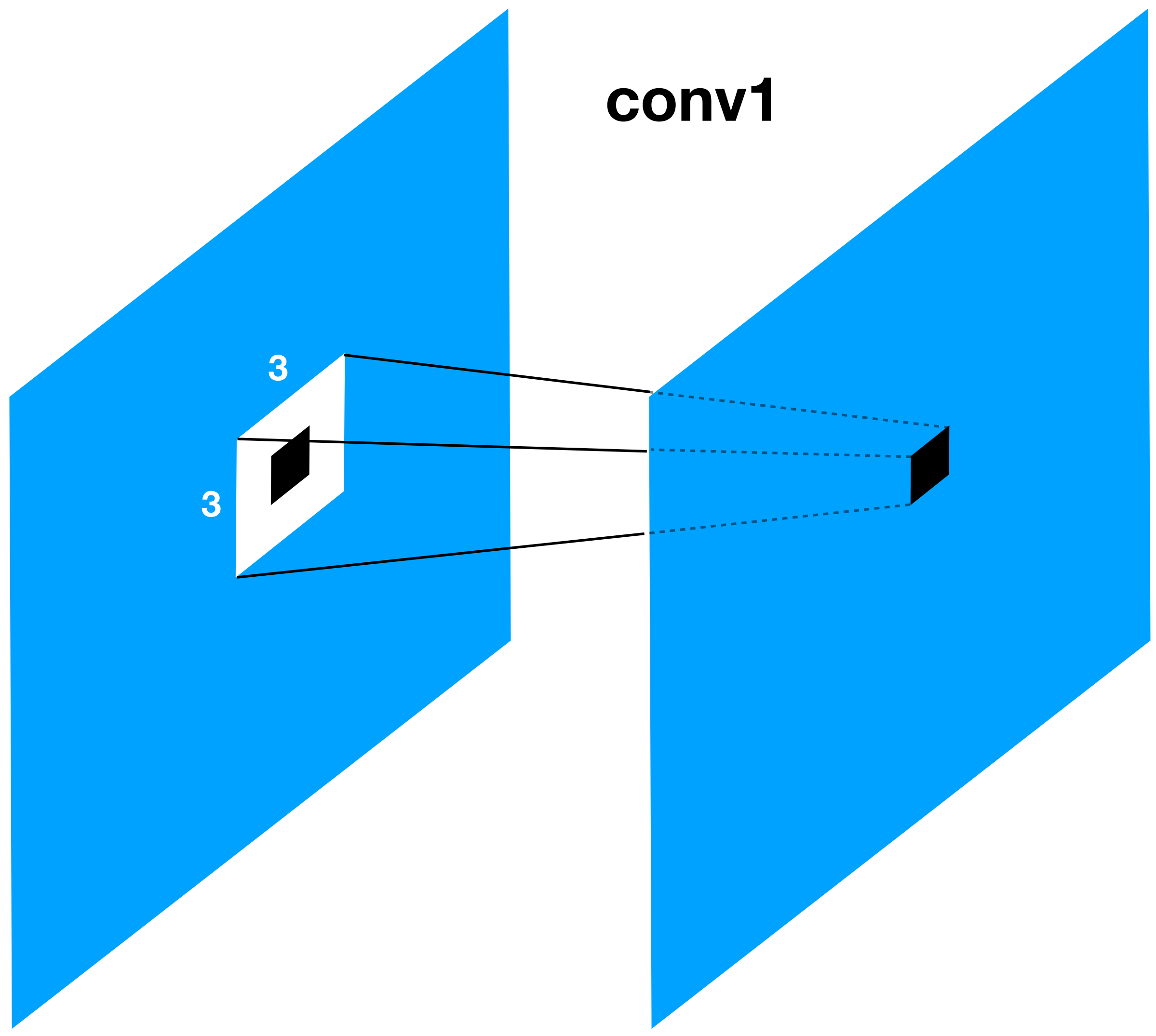


‘FC6/7/8’

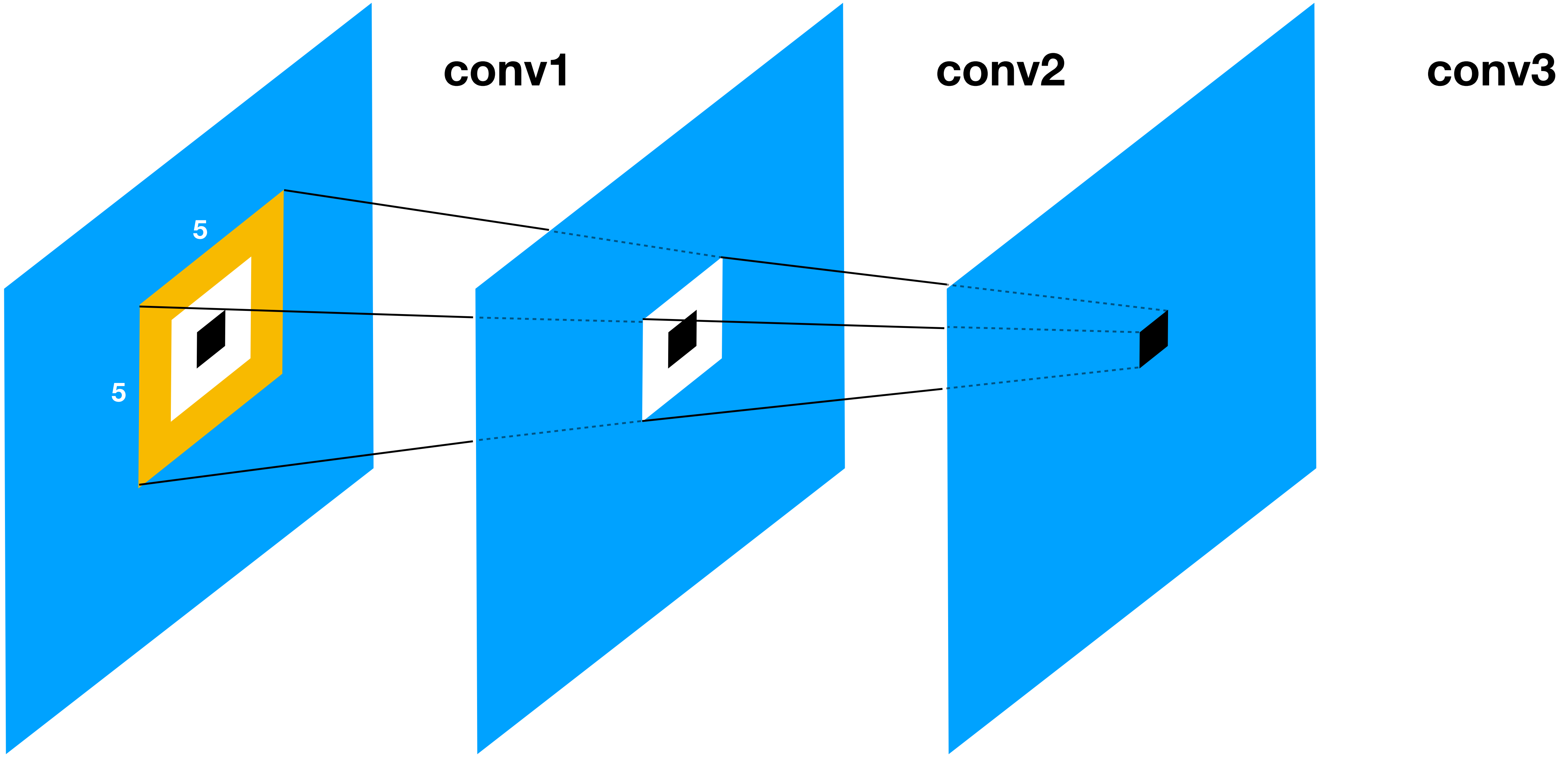
What is the receptive field of 3 stacks of 3x3 convolutions?



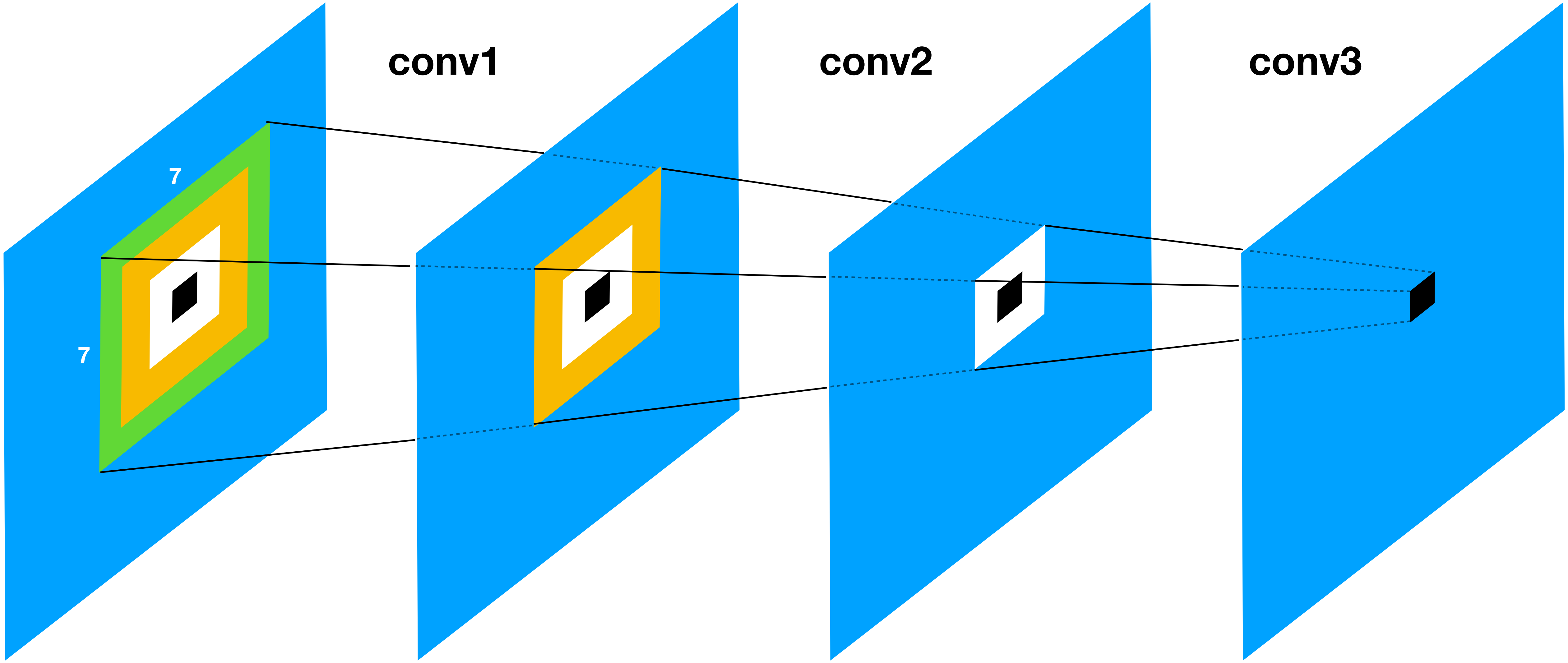
What is the receptive field of 3 stacks of 3x3 convolutions?



What is the receptive field of 3 stacks of 3x3 convolutions?



What is the receptive field of 3 stacks of 3x3 convolutions?



What do you gain by stacking convolutions?

What do you gain by stacking convolutions?

Decision function is more discriminative.

What do you gain by stacking convolutions?

Decision function is more discriminative.

Two linear functions

$$b \cdot (a \cdot x)$$

Same as just one linear function

Two non-linear functions

$$F(b \cdot F(a \cdot x))$$

Can capture more complex decision boundary

What do you gain by stacking convolutions?

Decision function is more discriminative.

Two linear functions

$$b \cdot (a \cdot x)$$

Same as just one linear function

Two non-linear functions

$$F(b \cdot F(a \cdot x))$$

Can capture more complex decision boundary

Decrease number of parameters.

What do you gain by stacking convolutions?

Decision function is more discriminative.

Two linear functions

$$b \cdot (a \cdot x)$$

Same as just one linear function

Two non-linear functions

$$F(b \cdot F(a \cdot x))$$

Can capture more complex decision boundary

Decrease number of parameters.

Three layer 3 x 3 x C convolution has $3 \cdot 3^2 \cdot c = 27 \cdot c$ parameters.

What do you gain by stacking convolutions?

Decision function is more discriminative.

Two linear functions

$$b \cdot (a \cdot x)$$

Same as just one linear function

Two non-linear functions

$$F(b \cdot F(a \cdot x))$$

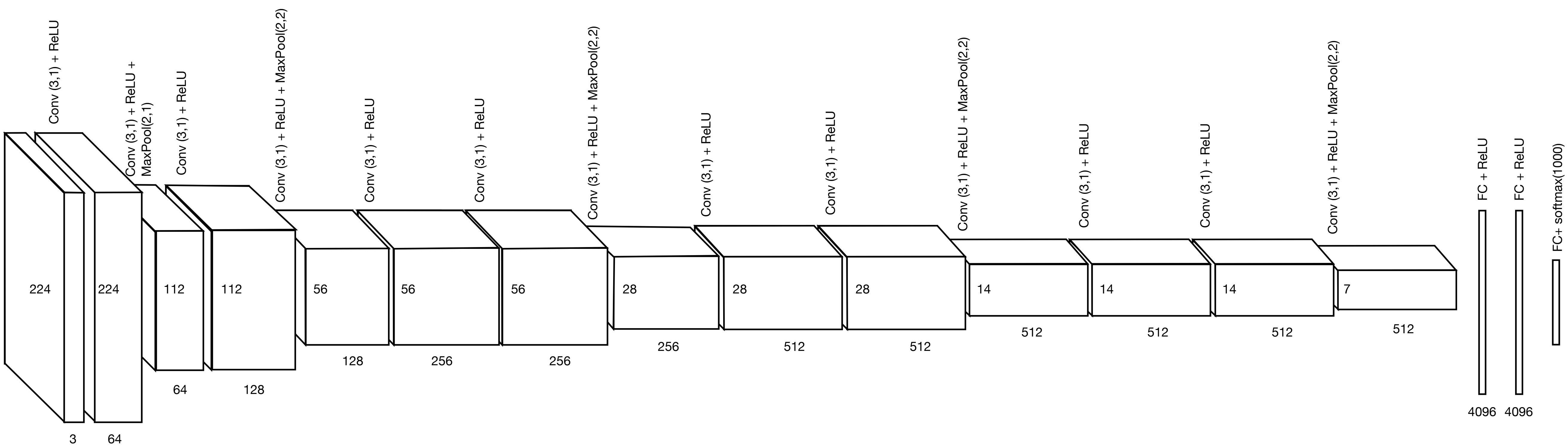
Can capture more complex decision boundary

Decrease number of parameters.

Three layer 3 x 3 x C convolution has $3 \cdot 3^2 \cdot c = 27 \cdot c$ parameters.

One layer 7 x 7 x C convolution has $7^2 \cdot c = 49 \cdot c$ parameters.

VGG-16



How easy is it to train VGG?

How easy is it to train VGG?

6 different VGG networks in the paper

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					

How easy is it to train VGG?

A can be trained from scratch

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					

How easy is it to train VGG?

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					

These cannot be trained from scratch

What would you do?

How easy is it to train VGG?

Use A to initialize bigger 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					

What would you do?

Tricks for Training VGG

Pre-training

Data augmentation (multi-crops, flips, color, scale)

Dropout

SGD with momentum


4 GPUs
21 days
to train VGG19

VGG Testing Details

- Convert model to fully convolutional network.
  Generates a pixel-wise heat maps with channels equal to number of classes
  (no need to sample multiple crops like AlexNet)

- Average pool final output to obtain class score

- Hack! Softmax class posteriors of original and flipped images averaged

VGG Results

Single VGG Network
(ILSVRC 2012)

Table 5: **ConvNet evaluation techniques comparison.** In all experiments the training scale S was sampled from $[256; 512]$, and three test scales Q were considered: $\{256, 384, 512\}$.

ConvNet config. (Table 1)	Evaluation method	top-1 val. error (%)	top-5 val. error (%)
D	dense	24.8	7.5
	multi-crop	24.6	7.5
	multi-crop & dense	**24.4**	**7.2**
E	dense	24.8	7.5
	multi-crop	24.6	7.4
	multi-crop & dense	**24.4**	**7.1**

Multiple VGG Networks
(ILSVRC 2012)

Table 6: **Multiple ConvNet fusion results.**

Combined ConvNet models	Error		
	top-1 val	top-5 val	top-5 test
ILSVRC submission			
(D/256/224,256,288), (D/384/352,384,416), (D/[256;512]/256,384,512) (C/256/224,256,288), (C/384/352,384,416) (E/256/224,256,288), (E/384/352,384,416)	24.7	7.5	7.3
post-submission			
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), dense eval.	24.0	7.1	7.0
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop	23.9	7.2	-
(D/[256;512]/256,384,512), (E/[256;512]/256,384,512), multi-crop & dense eval.	**23.7**	**6.8**	**6.8**

Table 7: **Comparison with the state of the art in ILSVRC classification**. Our method is denoted as "VGG". Only the results obtained without outside training data are reported.

<table>
<tr><th>Method</th><th>top-1 val. error (%)</th><th>top-5 val. error (%)</th><th>top-5 test error (%)</th></tr>
<tr><td>VGG (2 nets, multi-crop & dense eval.)</td><td>23.7</td><td>6.8</td><td>6.8</td></tr>
<tr><td>VGG (1 net, multi-crop & dense eval.)</td><td>24.4</td><td>7.1</td><td>7.0</td></tr>
<tr><td>VGG (ILSVRC submission, 7 nets, dense eval.)</td><td>24.7</td><td>7.5</td><td>7.3</td></tr>
<tr><td>GoogLeNet (Szegedy et al., 2014) (1 net)</td><td>-</td><td colspan="2">7.9</td></tr>
<tr><td>GoogLeNet (Szegedy et al., 2014) (7 nets)</td><td>-</td><td colspan="2">6.7</td></tr>
<tr><td>MSRA (He et al., 2014) (11 nets)</td><td>-</td><td>-</td><td>8.1</td></tr>
<tr><td>MSRA (He et al., 2014) (1 net)</td><td>27.9</td><td>9.1</td><td>9.1</td></tr>
<tr><td>Clarifai (Russakovsky et al., 2014) (multiple nets)</td><td>-</td><td>-</td><td>11.7</td></tr>
<tr><td>Clarifai (Russakovsky et al., 2014) (1 net)</td><td>-</td><td>-</td><td>12.5</td></tr>
<tr><td>Zeiler & Fergus (Zeiler & Fergus, 2013) (6 nets)</td><td>36.0</td><td>14.7</td><td>14.8</td></tr>
<tr><td>Zeiler & Fergus (Zeiler & Fergus, 2013) (1 net)</td><td>37.5</td><td>16.0</td><td>16.1</td></tr>
<tr><td>OverFeat (Sermanet et al., 2014) (7 nets)</td><td>34.0</td><td>13.2</td><td>13.6</td></tr>
<tr><td>OverFeat (Sermanet et al., 2014) (1 net)</td><td>35.7</td><td>14.2</td><td>-</td></tr>
<tr><td>Krizhevsky et al. (Krizhevsky et al., 2012) (5 nets)</td><td>38.1</td><td>16.4</td><td>16.4</td></tr>
<tr><td>Krizhevsky et al. (Krizhevsky et al., 2012) (1 net)</td><td>40.7</td><td>18.2</td><td>-</td></tr>
</table>

AlexNet

Important Concepts

Emerging 'rule of thumb'

Convolutions all standardized to 3 x 3

Convolutions always followed by ReLU

Stack several convolutions at the same resolution

Downsample by 2, increase channels by 2

No local response normalization

Training Tricks

Initialize bottom layers with smaller network

Dropout is important

Data augmentation is important