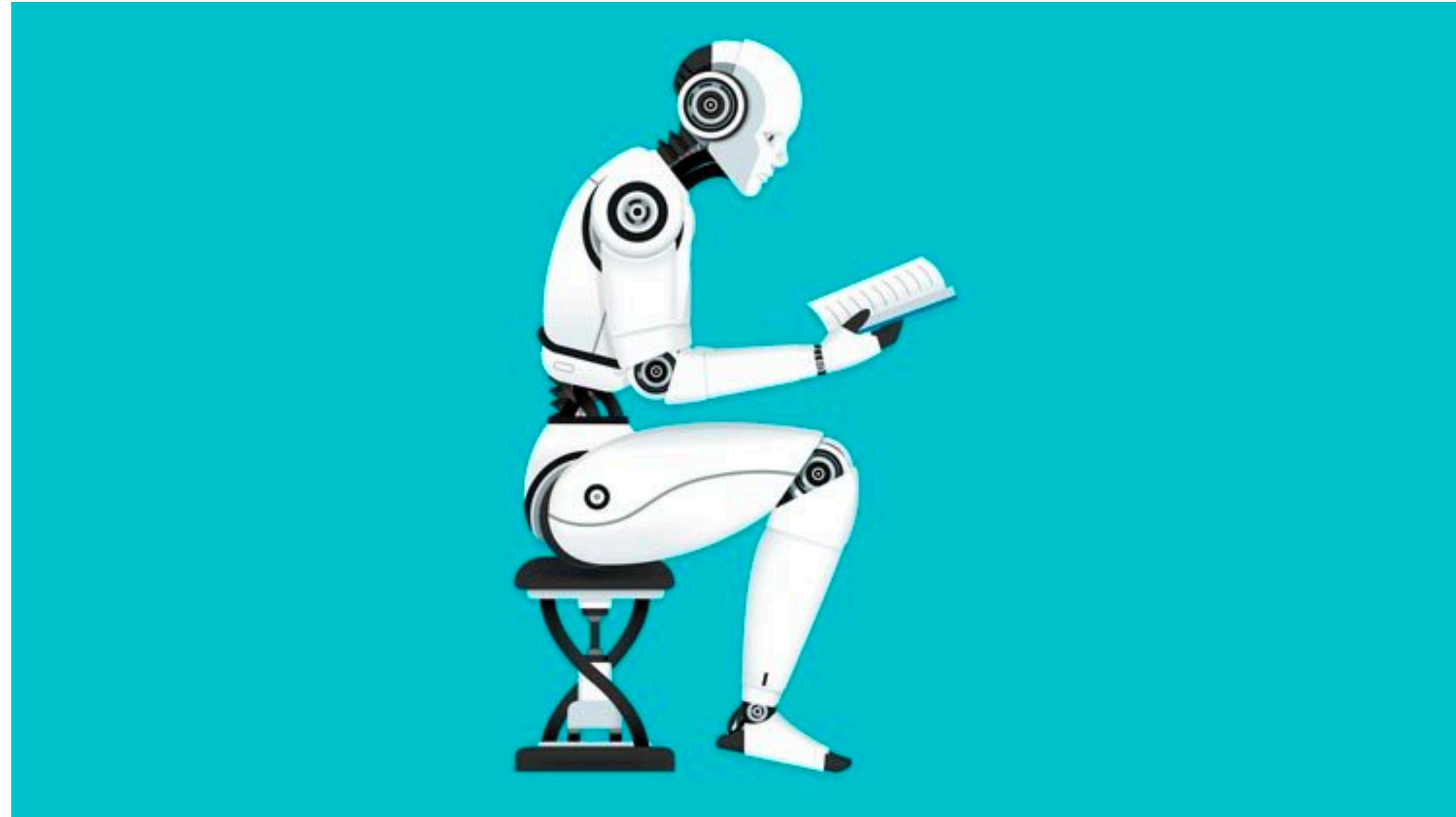




UNIVERSITAT DE
BARCELONA



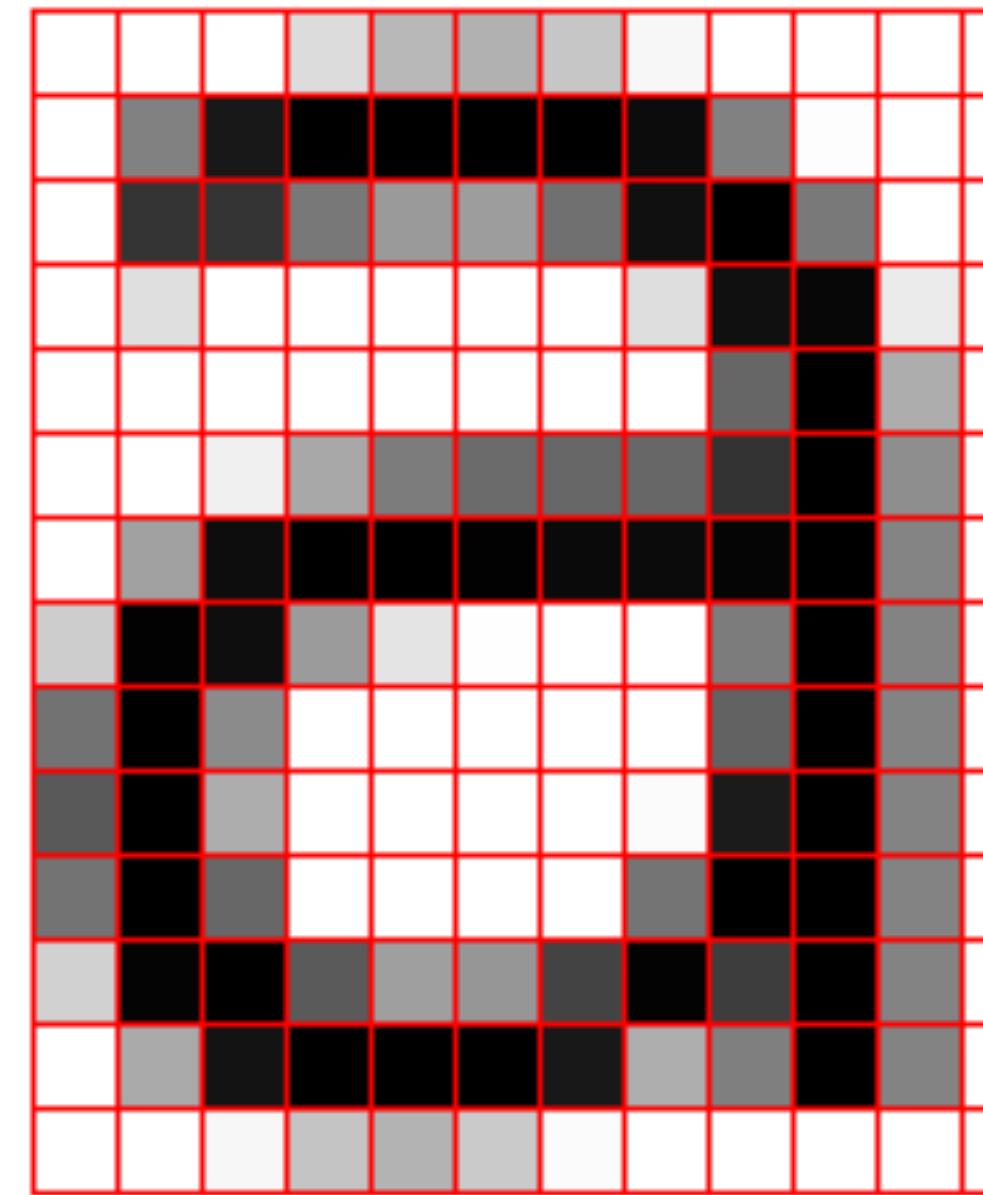
Neural Networks

Machine Learning | Enginyeria Informàtica

Santi Seguí | 2021-2022

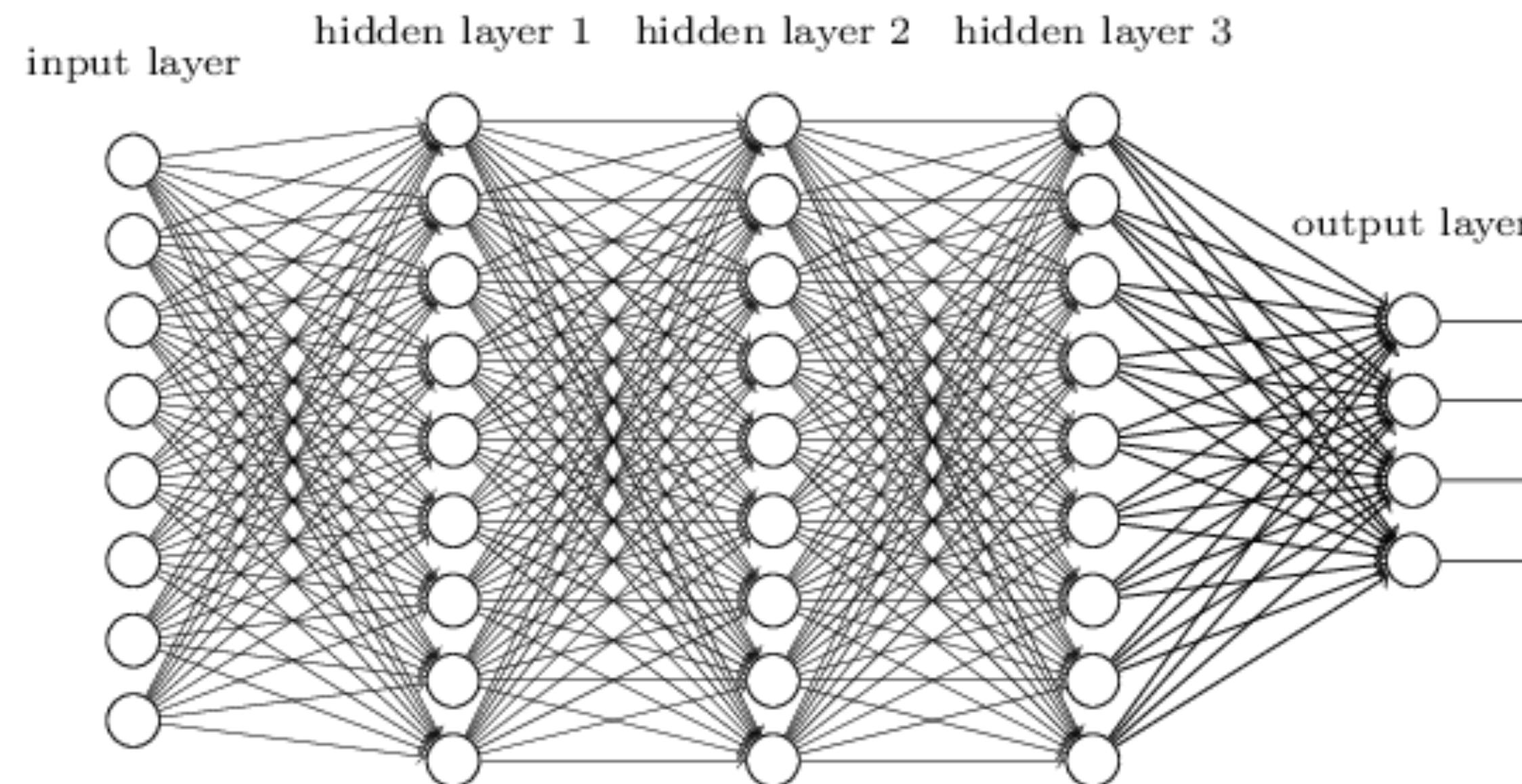
Neural Networks for Images?

An image is a matrix of size $m \times n \times c$ pixels



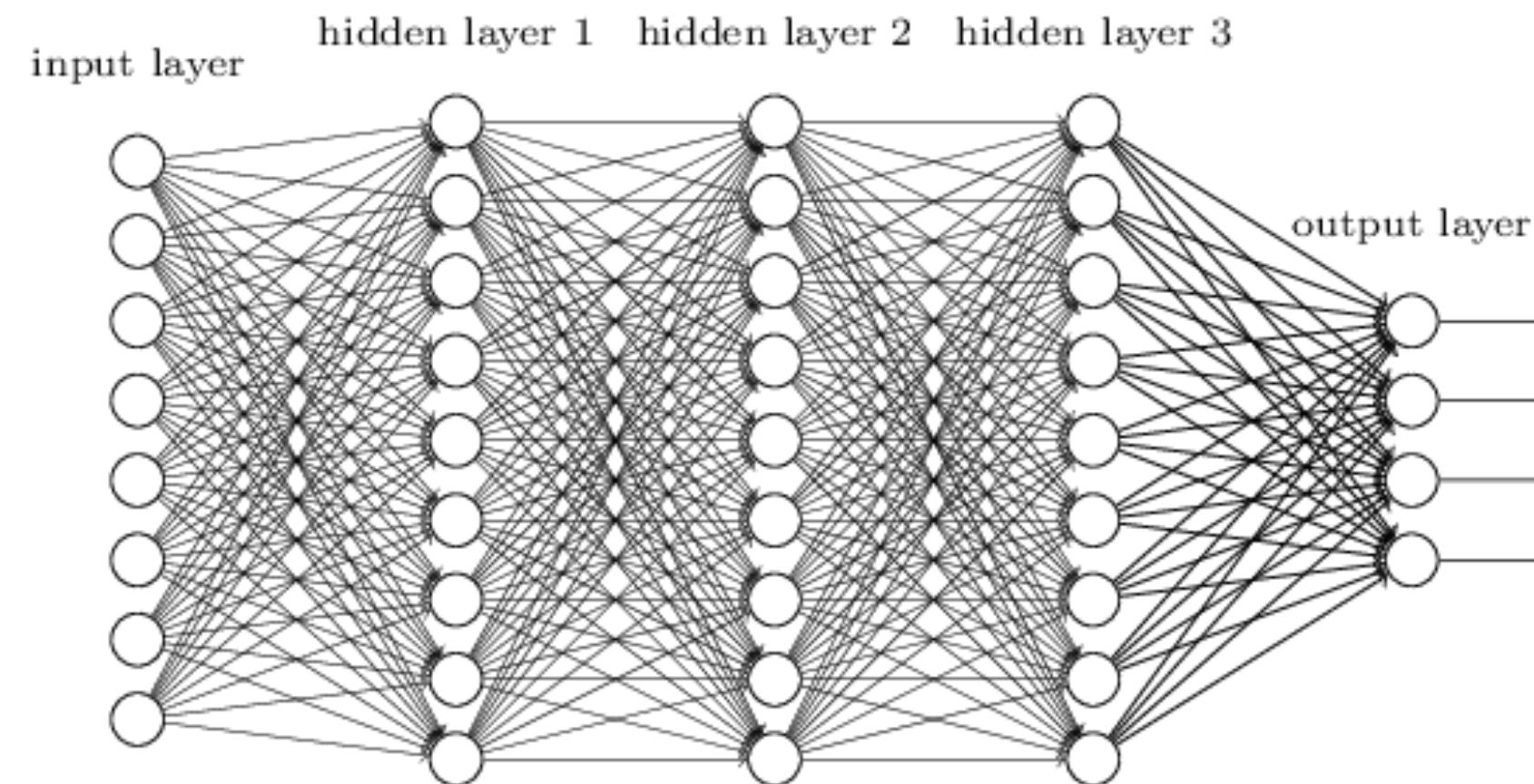
1.0	1.0	1.0	0.9	0.6	0.6	0.6	1.0	1.0	1.0	1.0	1.0
1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	1.0	1.0
1.0	0.2	0.2	0.5	0.6	0.6	0.5	0.0	0.0	0.5	1.0	1.0
1.0	0.9	1.0	1.0	1.0	1.0	1.0	0.9	0.0	0.0	0.9	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.0	0.5	1.0
1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.4	0.0	0.5	1.0	1.0
1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0
0.9	0.0	0.0	0.6	1.0	1.0	1.0	0.5	0.0	0.5	1.0	1.0
0.5	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.5	1.0	1.0
0.5	0.0	0.7	1.0	1.0	1.0	1.0	0.0	0.0	0.5	1.0	1.0
0.6	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.0	0.5	1.0
0.9	0.1	0.0	0.6	0.7	0.7	0.5	0.0	0.5	0.0	0.5	1.0
1.0	0.7	0.1	0.0	0.0	0.0	0.1	0.9	0.8	0.0	0.5	1.0
1.0	1.0	1.0	0.8	0.8	0.9	1.0	1.0	1.0	1.0	1.0	1.0

Neural Networks for Images?

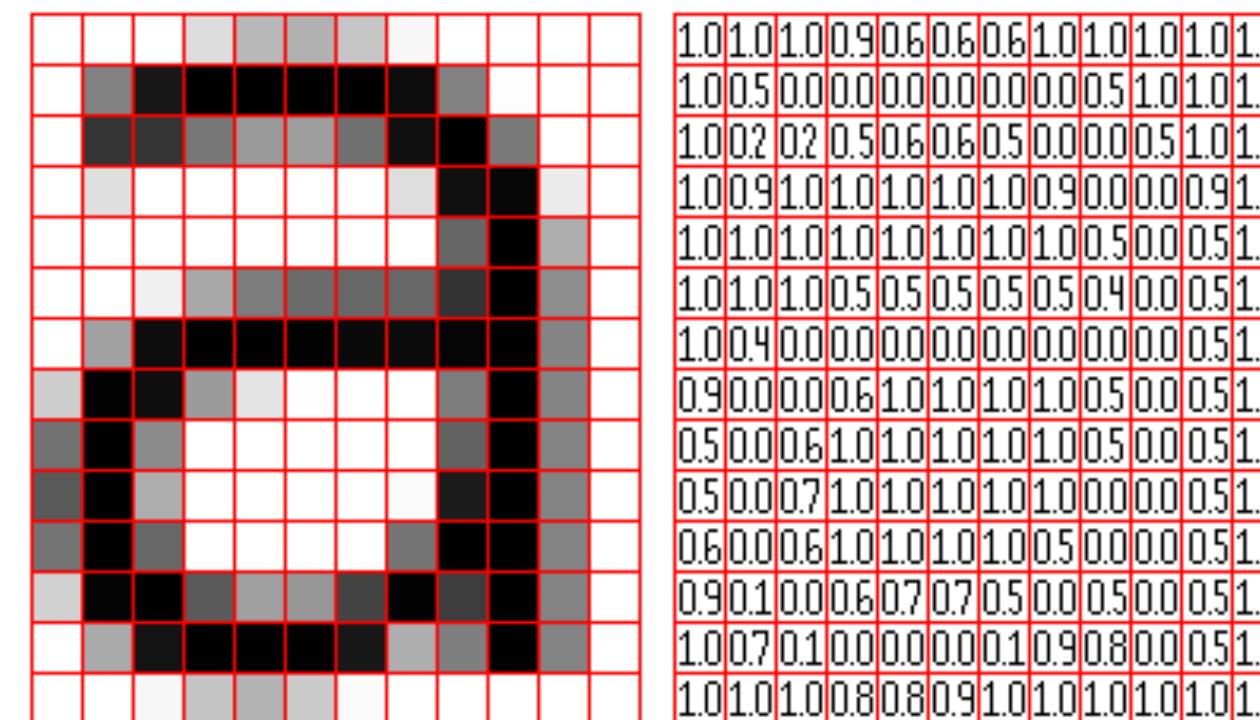


How many parameter does this MLP has?

Neural Networks for Images?



What happens if your data is an image like this one?



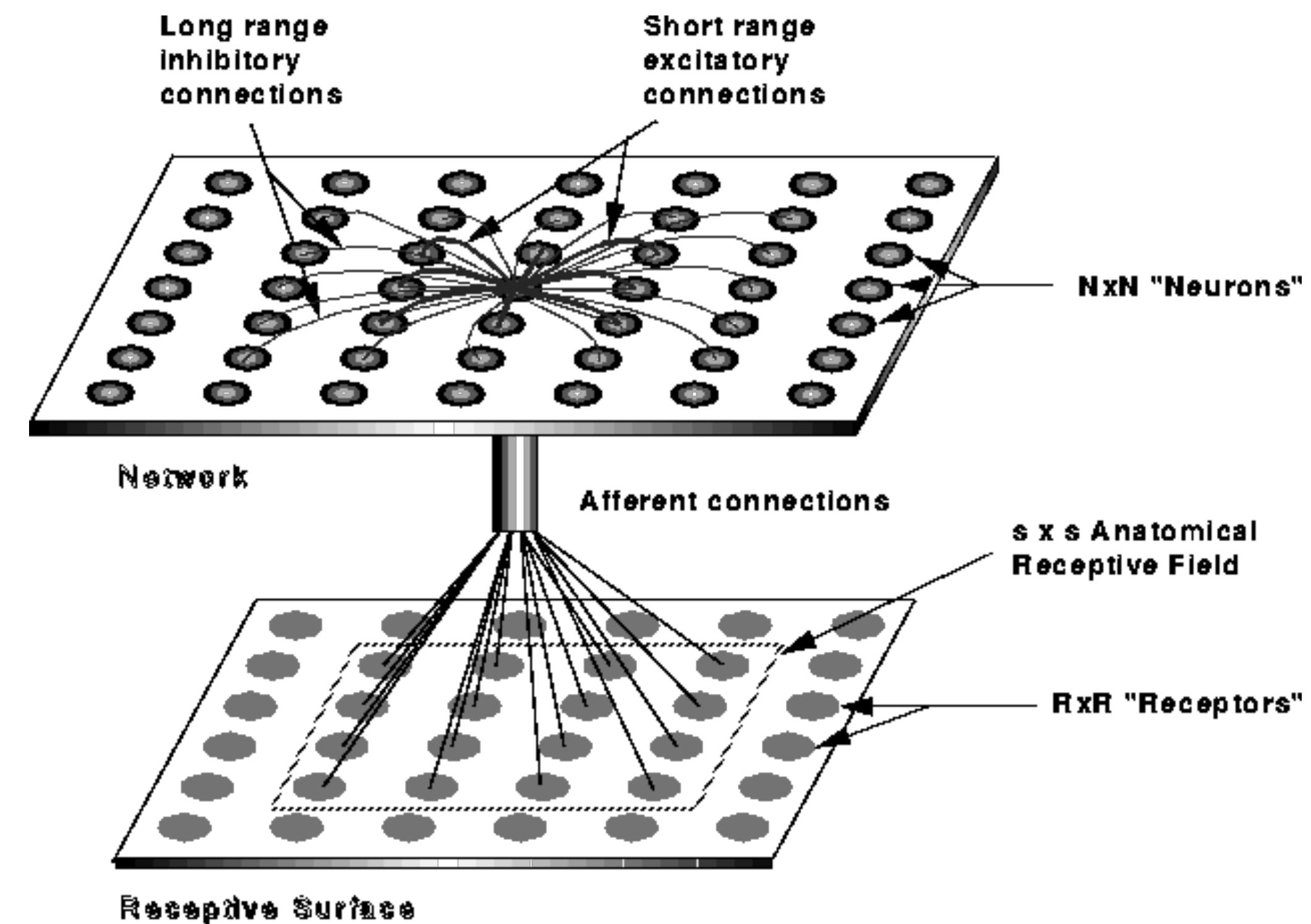
Neural Networks for Images?



Neural Networks for Images

- Input is a standard vector of size $N \times M \times C$
 - Imagine an medium resolution color image of 256x256 pixels
 - If we think on a Multi Layer Perceptron with just one hidden layer of 256 neurons + an output layer of 1 neuron it will have more than **48 million** parameters.
 - **Does it make sense? Can we do it better?**

Local Receptive Fields



David Hunter Hubel and Torsten Nils Wiesel, 1968

But, in an image:
A dog can appear **anywhere** in the image!

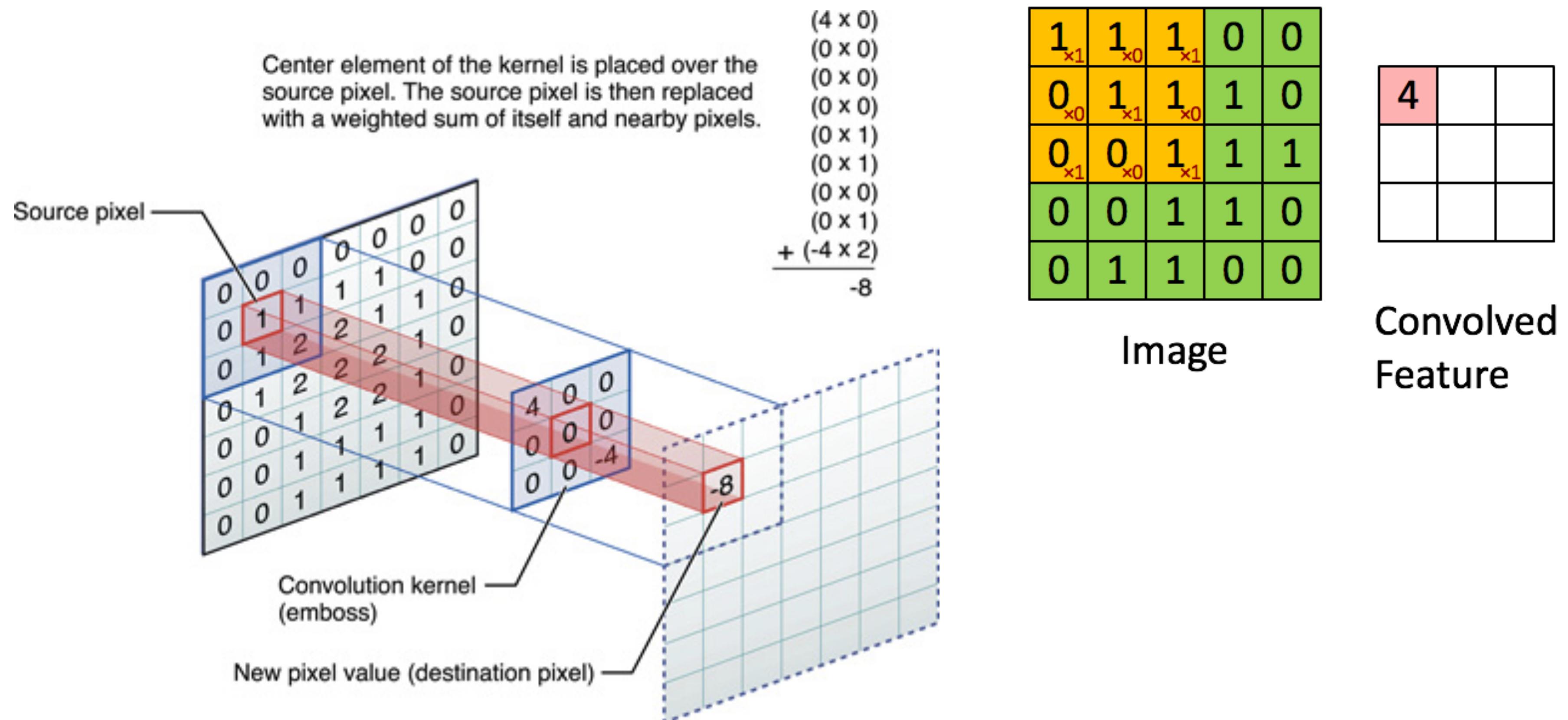


Doesn't matter where they appear,
they look similar anywhere!

Convolutional Neural Networks (CNNs)

- Three main ideas:
 1. **local receptive fields**,
 2. **shared weights**,
 3. **sub-sampling**

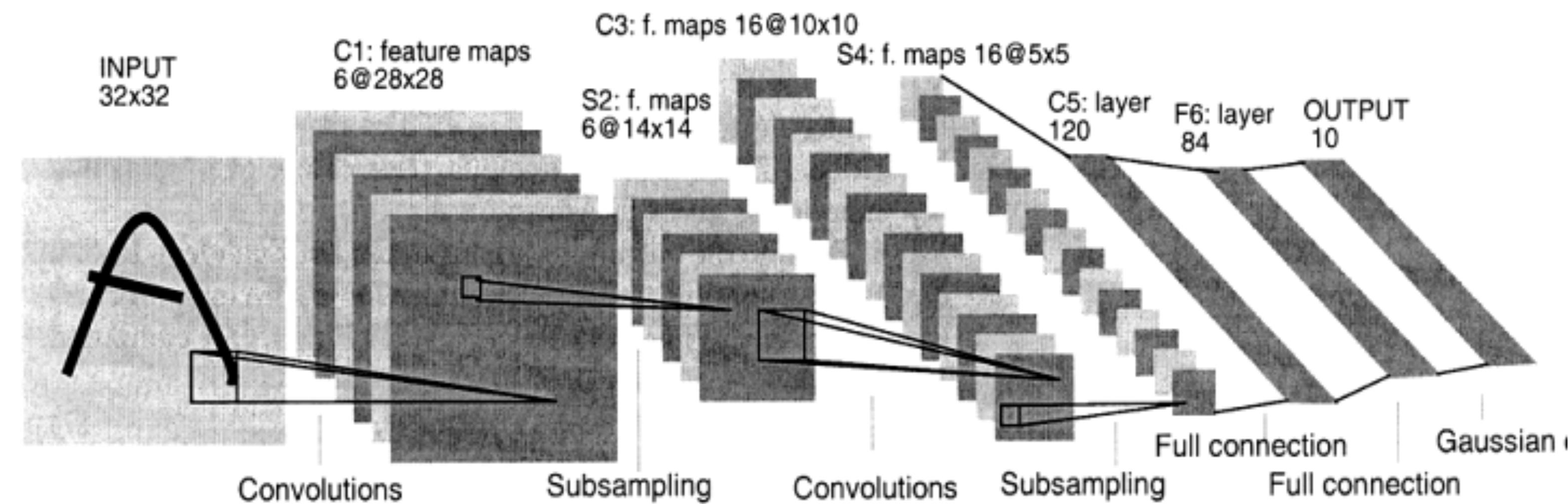
What is an image convolution?



What is an image convolution?



“Nothing New”

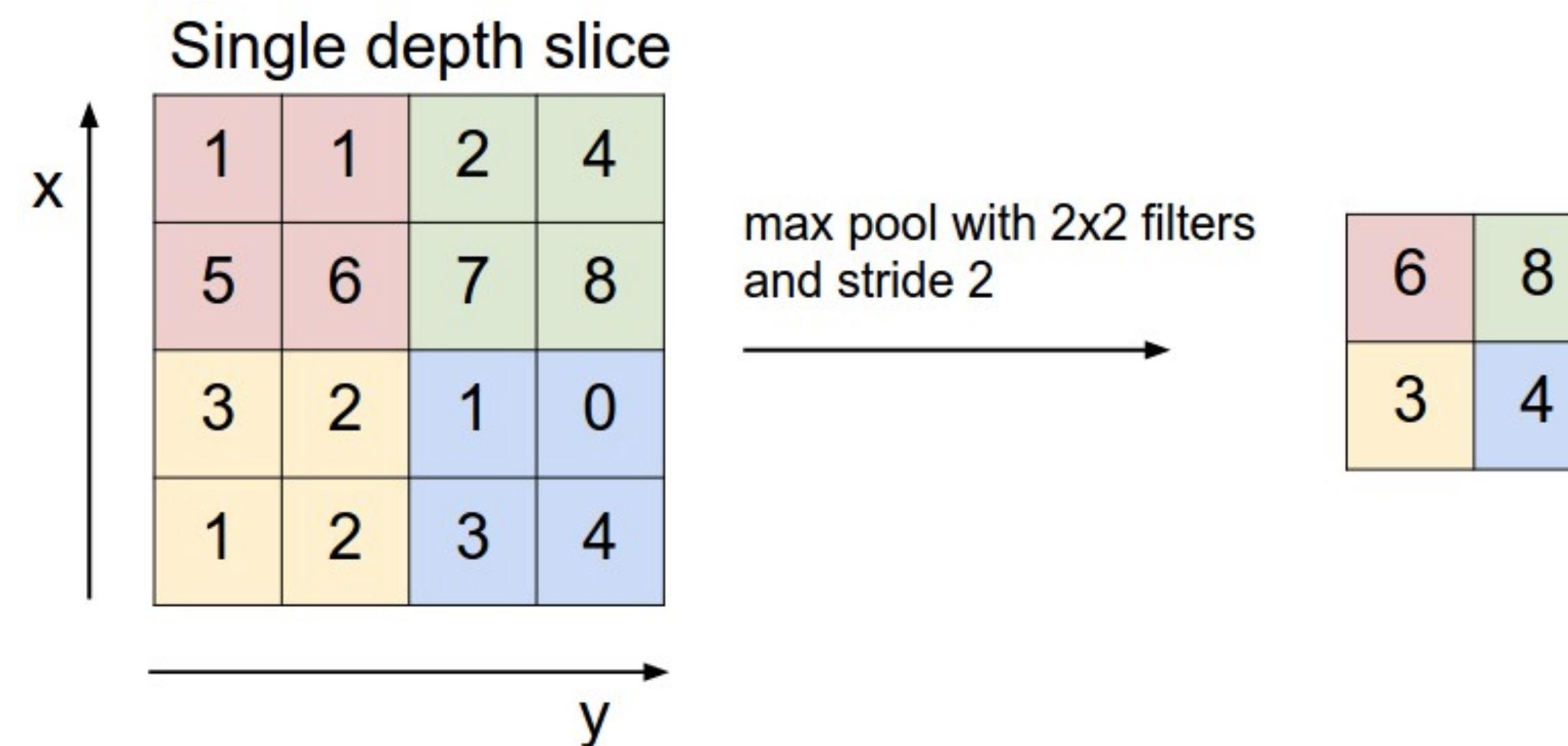


LeCun et al. 1998

Max pooling

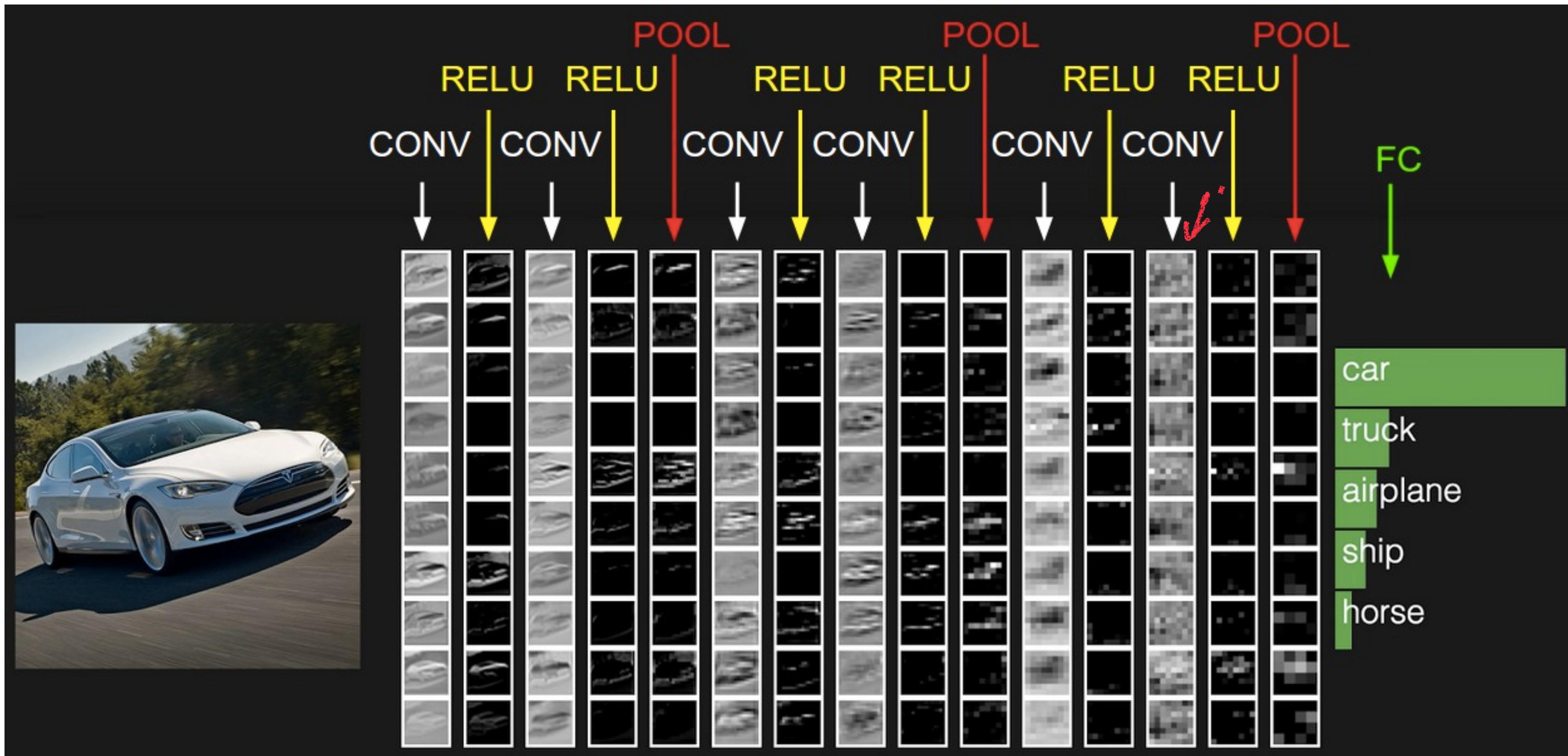
Pooling is a way of sub-sampling, i.e. reducing the dimension of the input (or at some hidden layer).

It is usually done after some of the convolutional layers



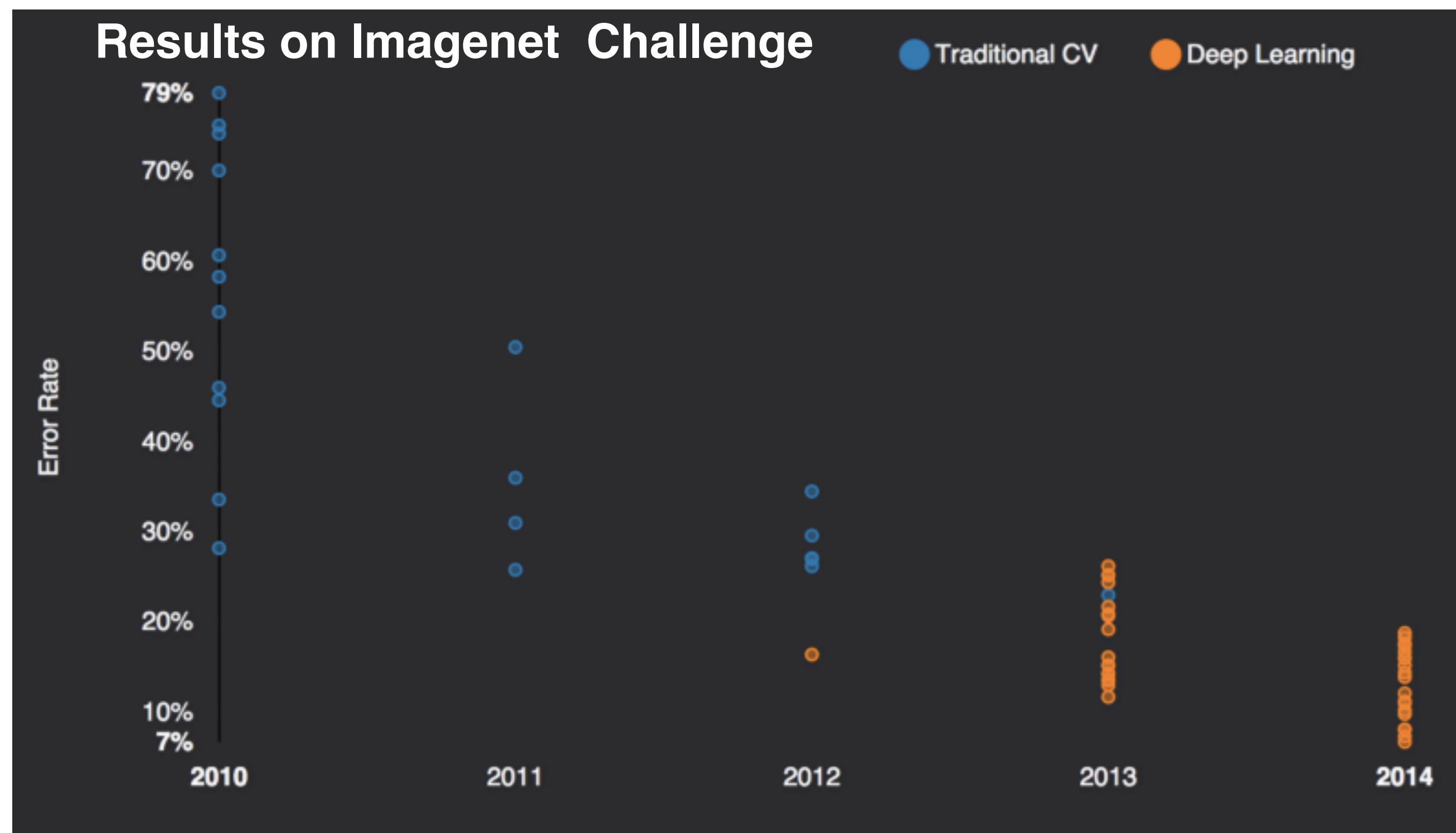
But it is also useful since it provides a form of translation **invariance**

Finally..



Convolutional Neural Networks (CNNs)

In computer Vision the breakthrough resulted in 2011 when Ciresan et.al introduced an algorithm to train these networks by using graphical cards (GPUs)



AlexNet

Similar framework to LeCun'98 but:

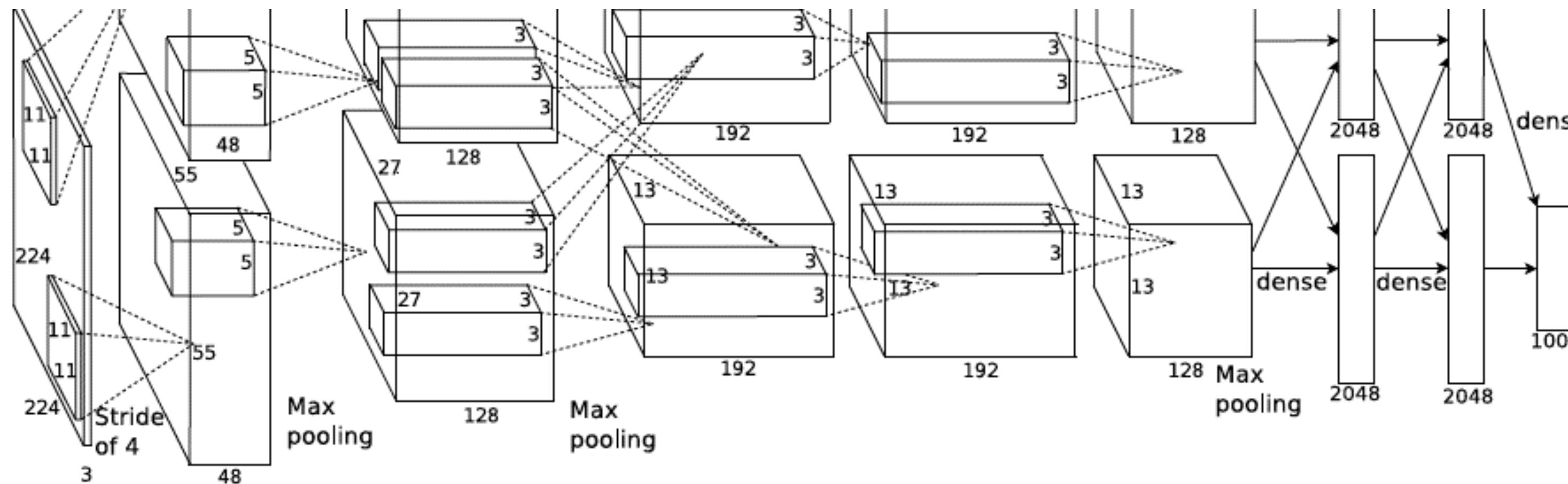
Bigger model:

7 hidden layers, 650.000 units, 60 million parameter

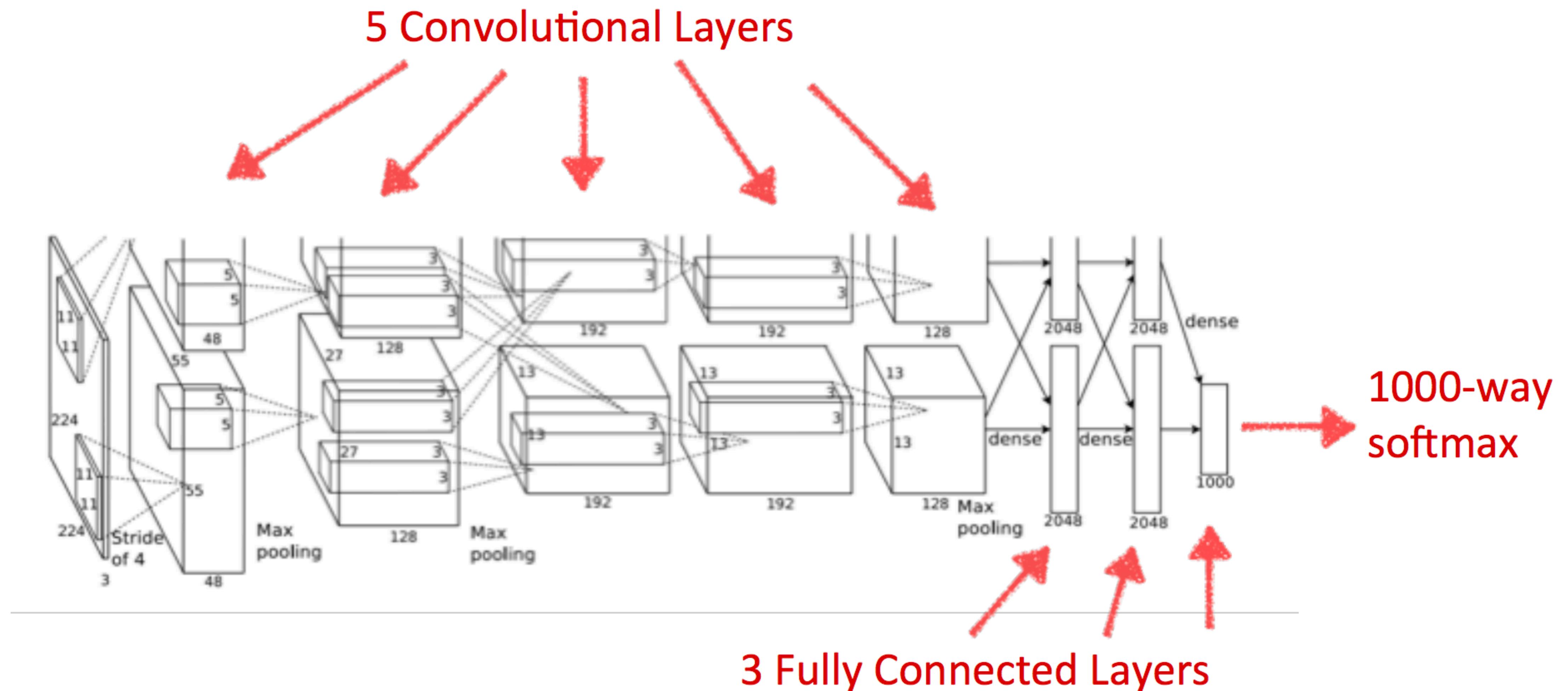
More Data:

10^6 vs 10^3 images

GPU implementation (50x speedup over CPU)

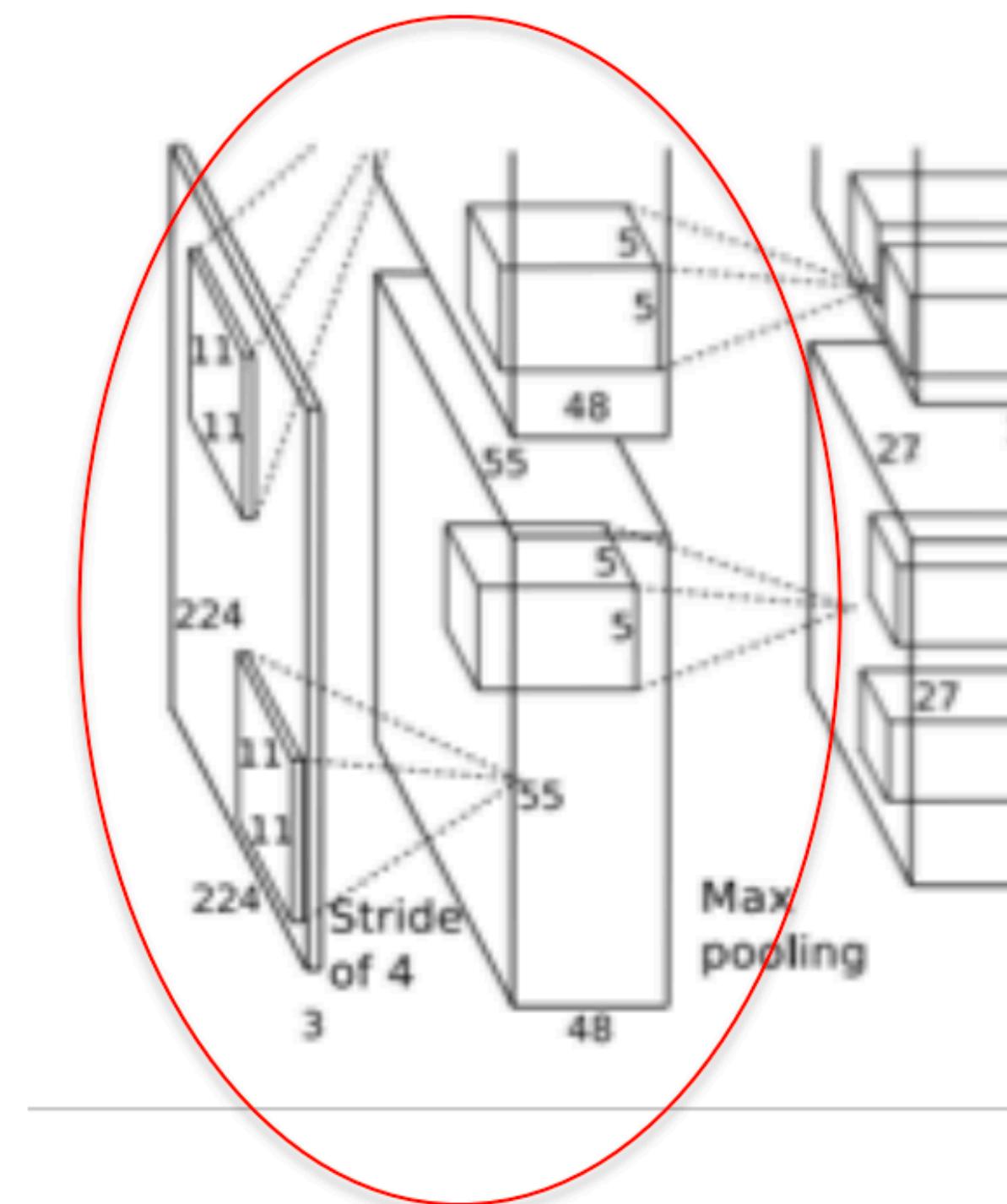


AlexNet



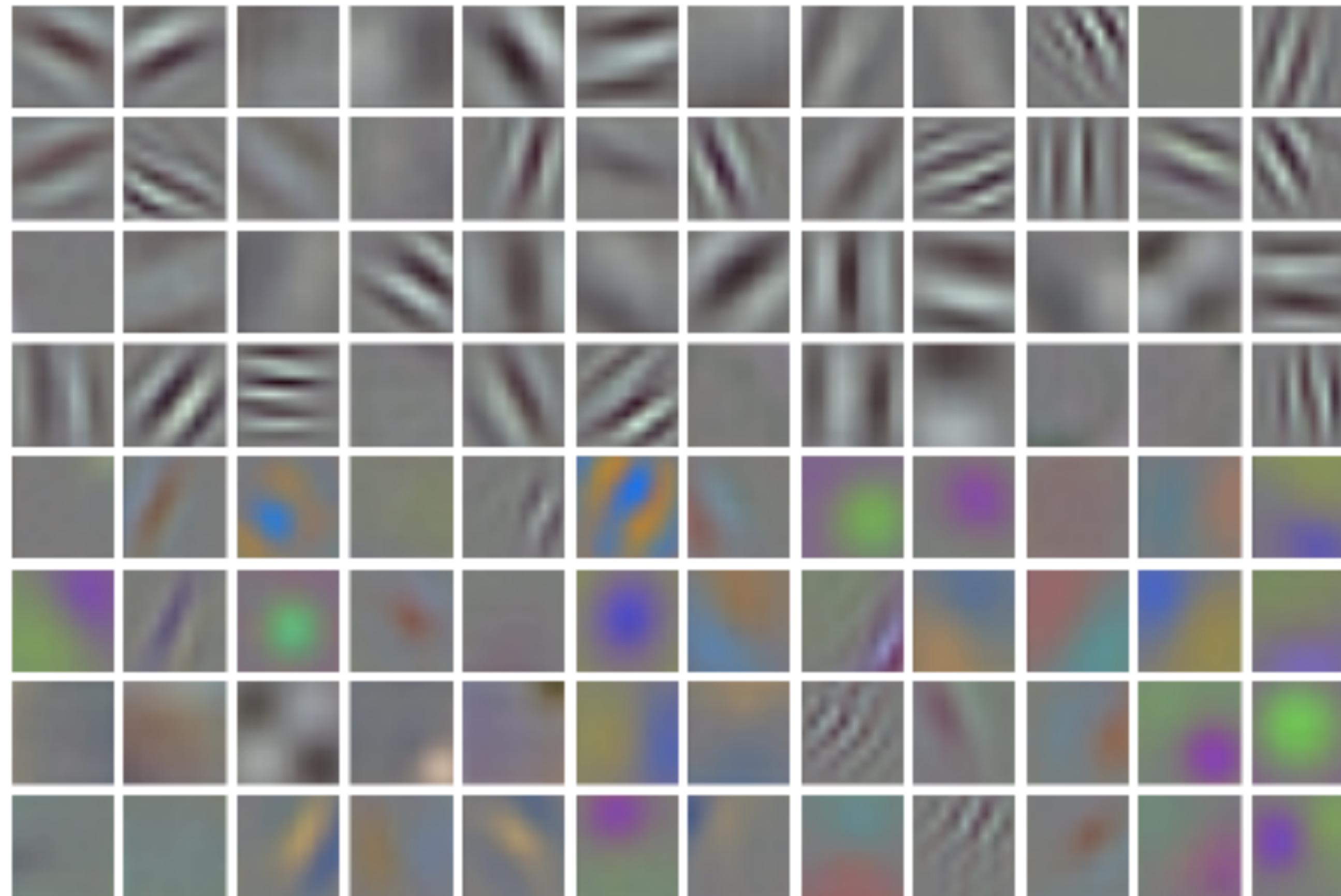
AlexNet

1st Convolution Layer

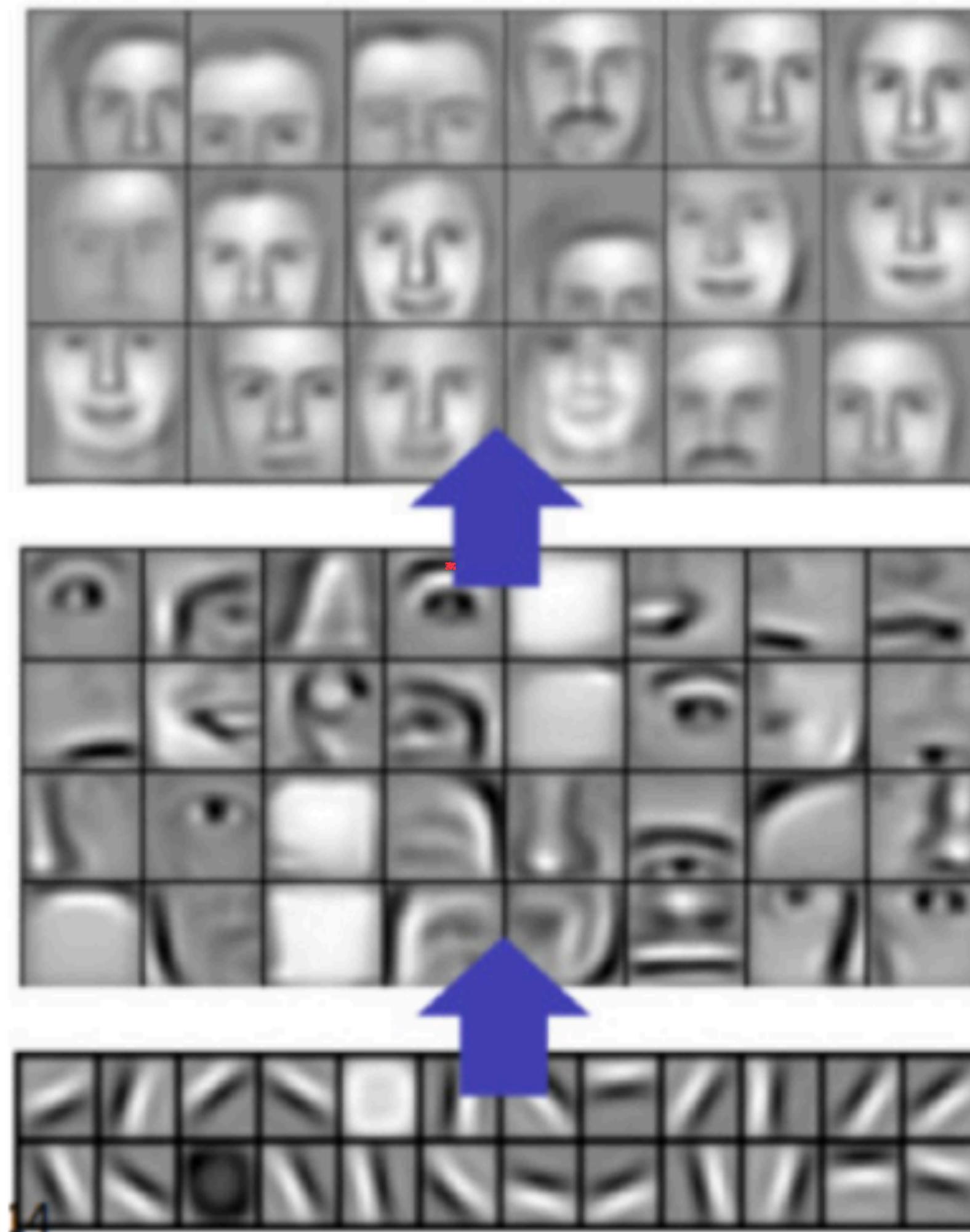


- Images: 227x227x3
- F (receptive field size): 11
- S (stride) = 4
- Conv layer output: 55x55x96

Alexnet 1st Conv Filters



Alexnet



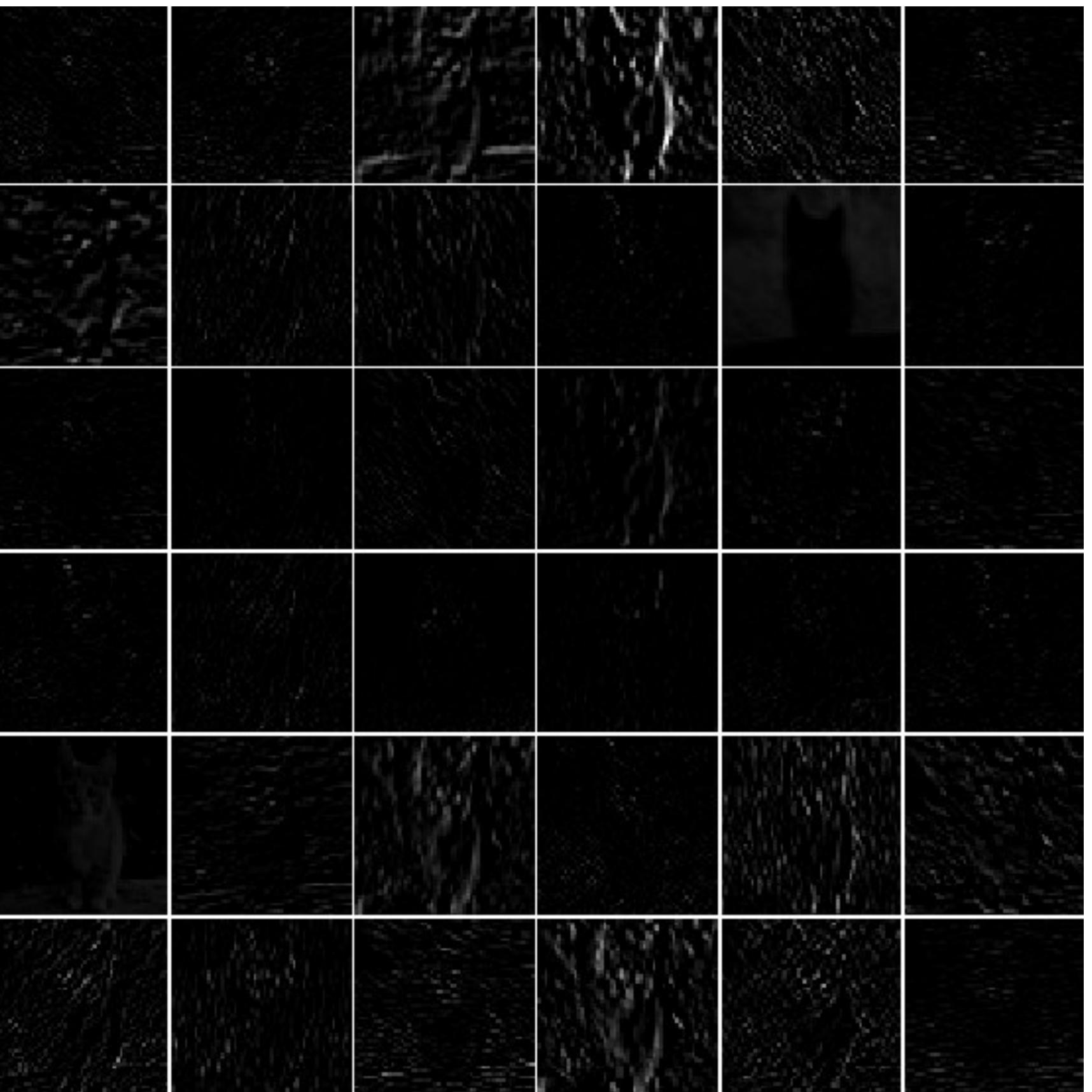
Layer 3

Layer 2

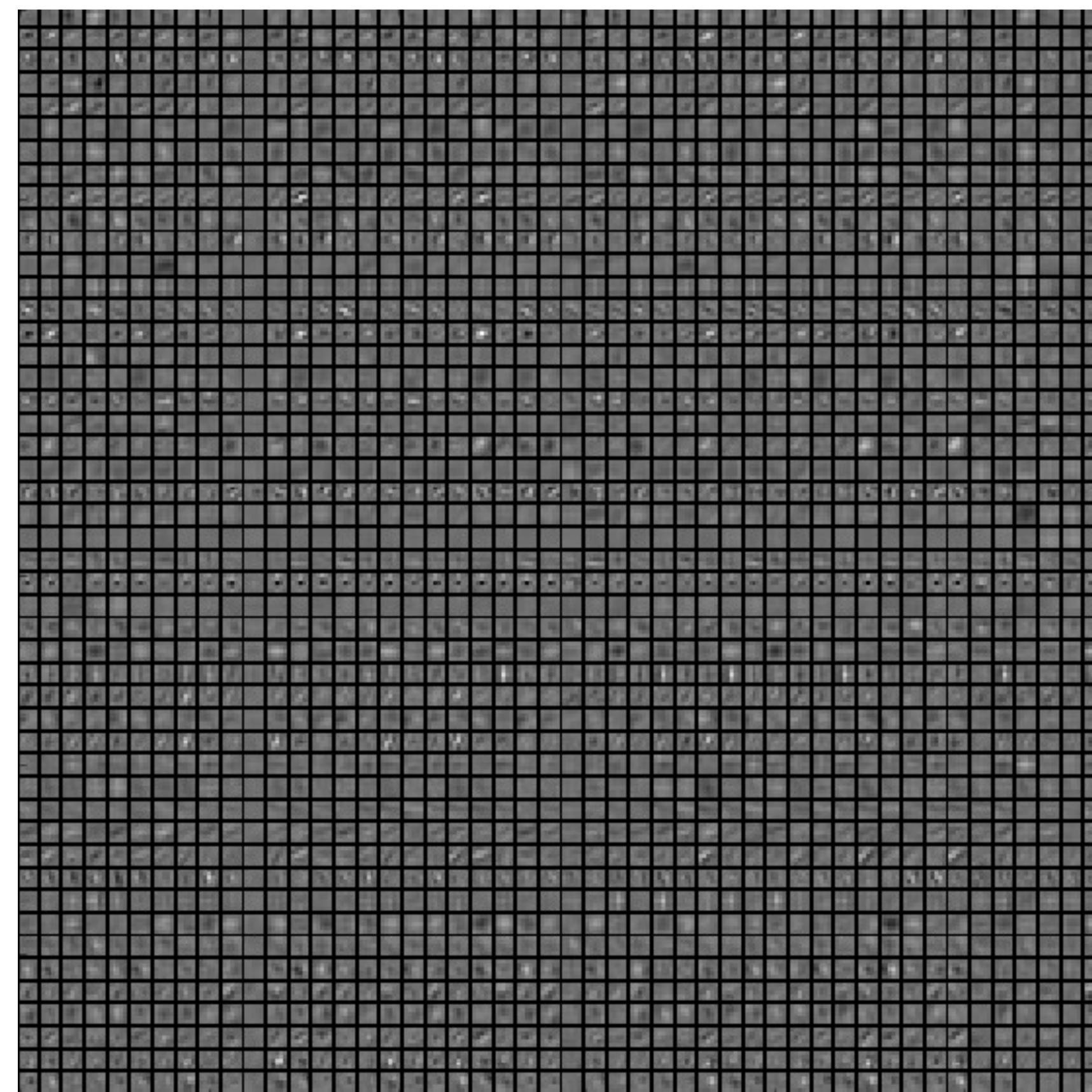
Layer 1

Alexnet

Feature Map Conv1



Conv2



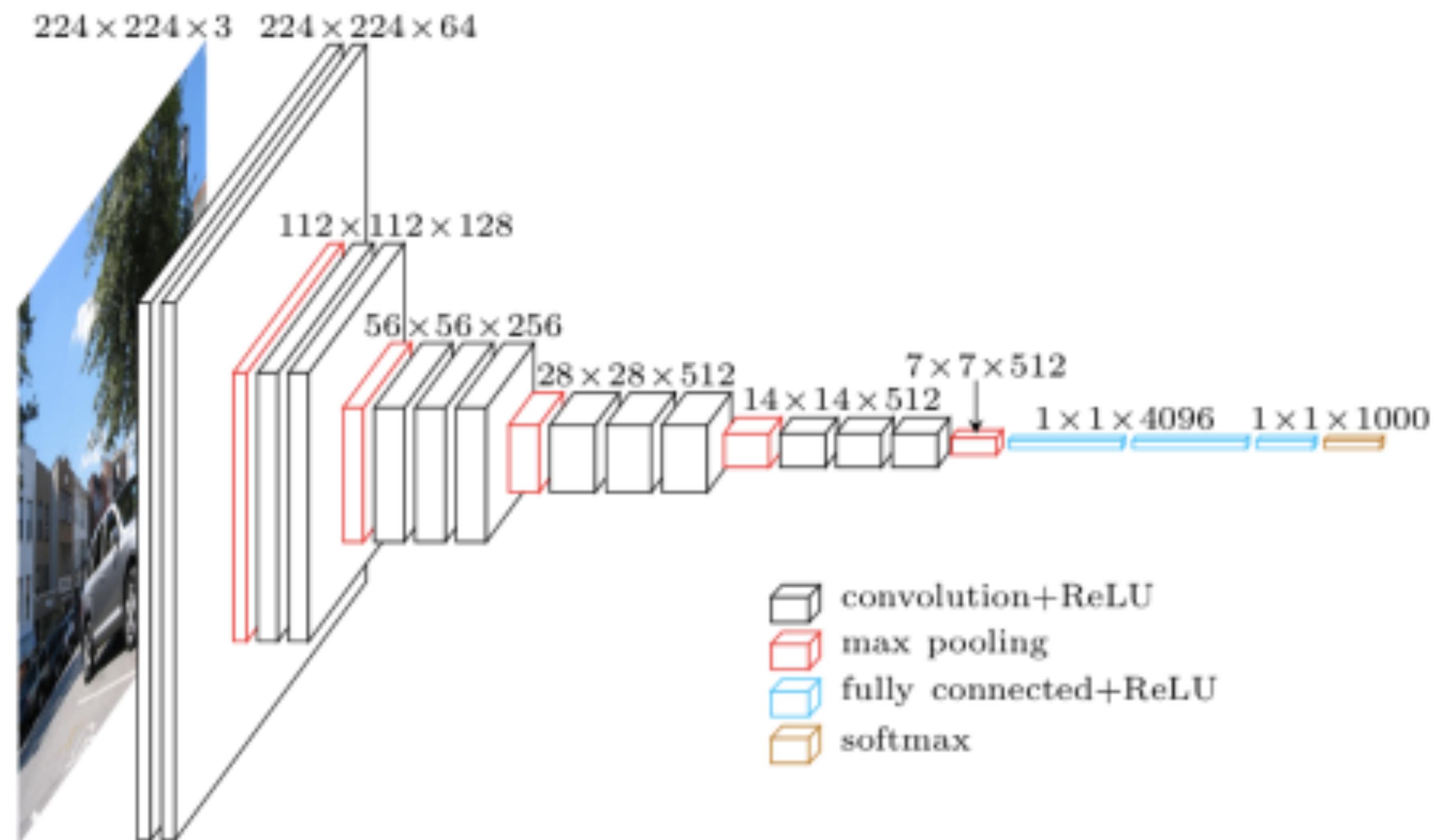


WE NEED TO GO

DEEPER

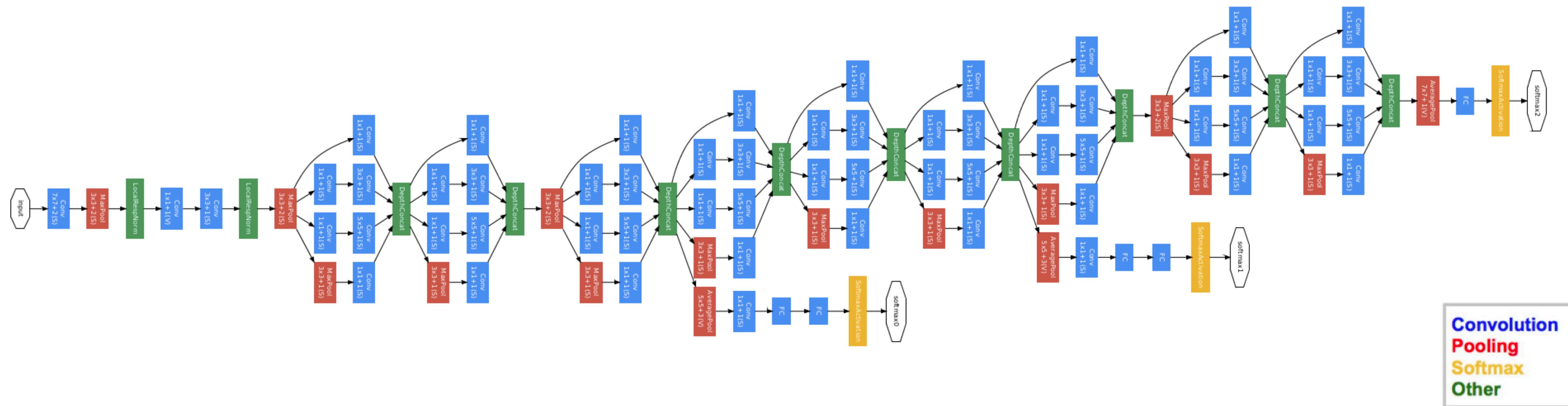
troll-face.ru

VGG Net

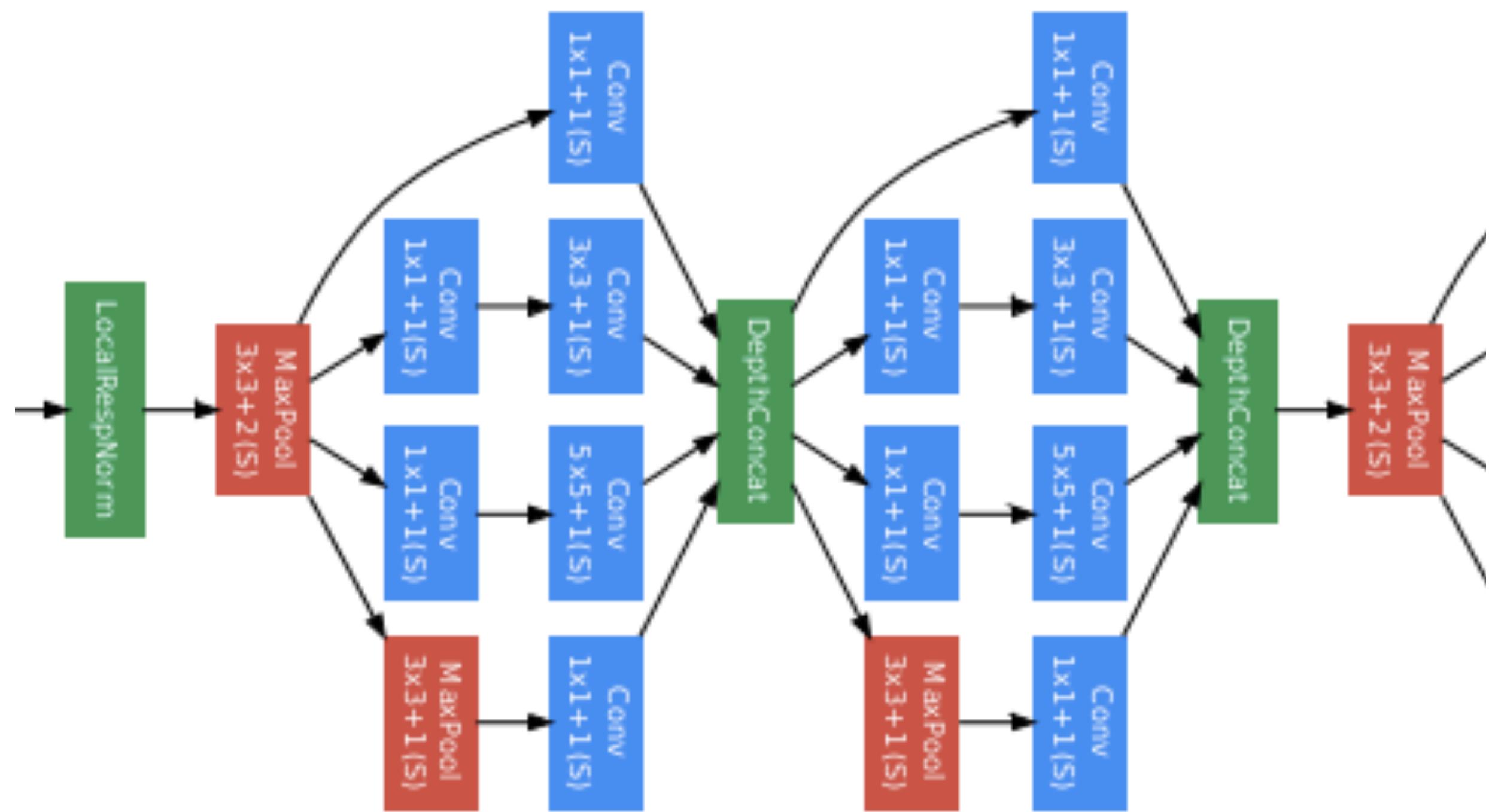


GoogLeNet

22 layers, but 12 times less parameters than AlexNet



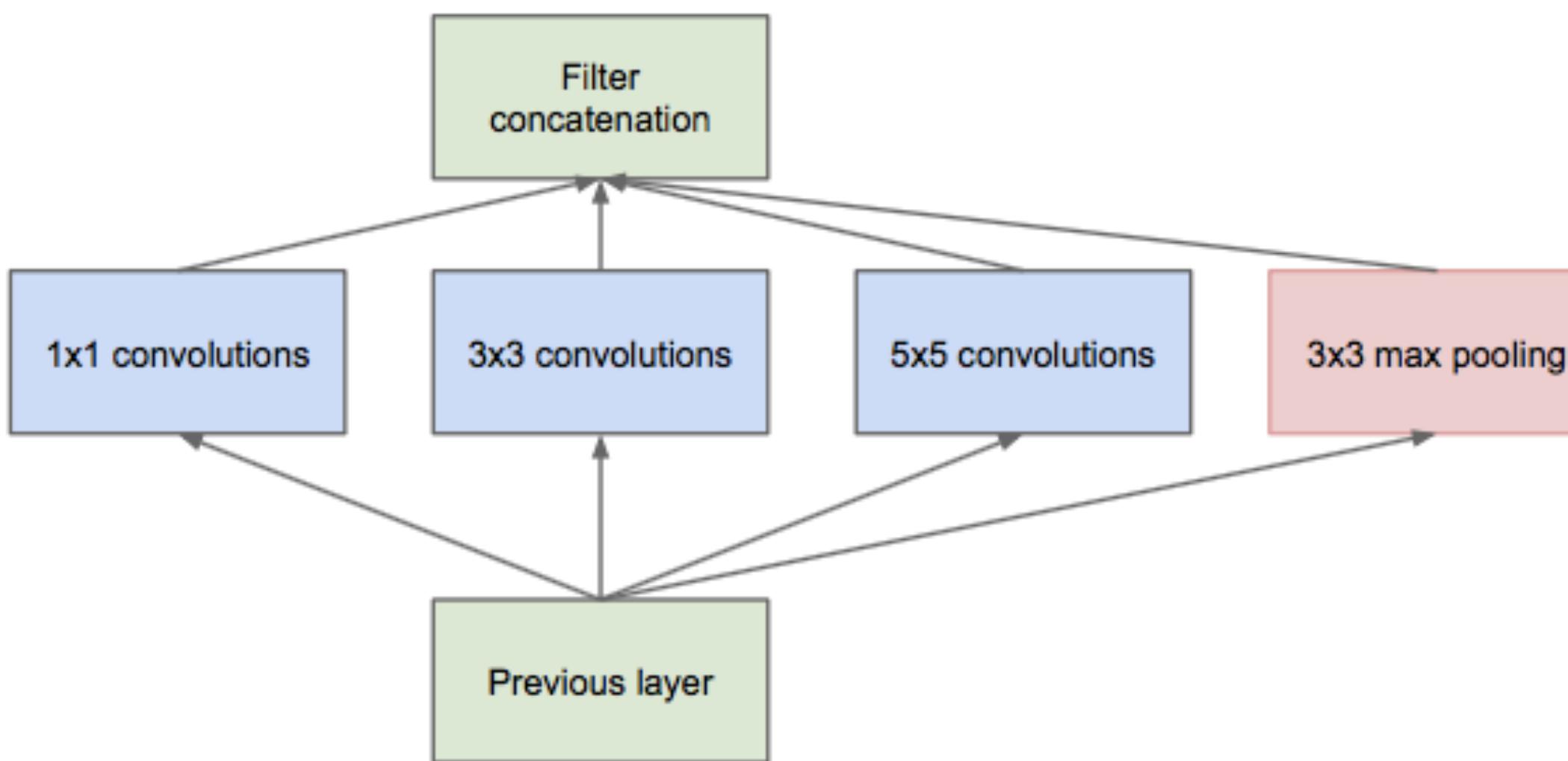
GoogleNet



Inception Module: Naive Version

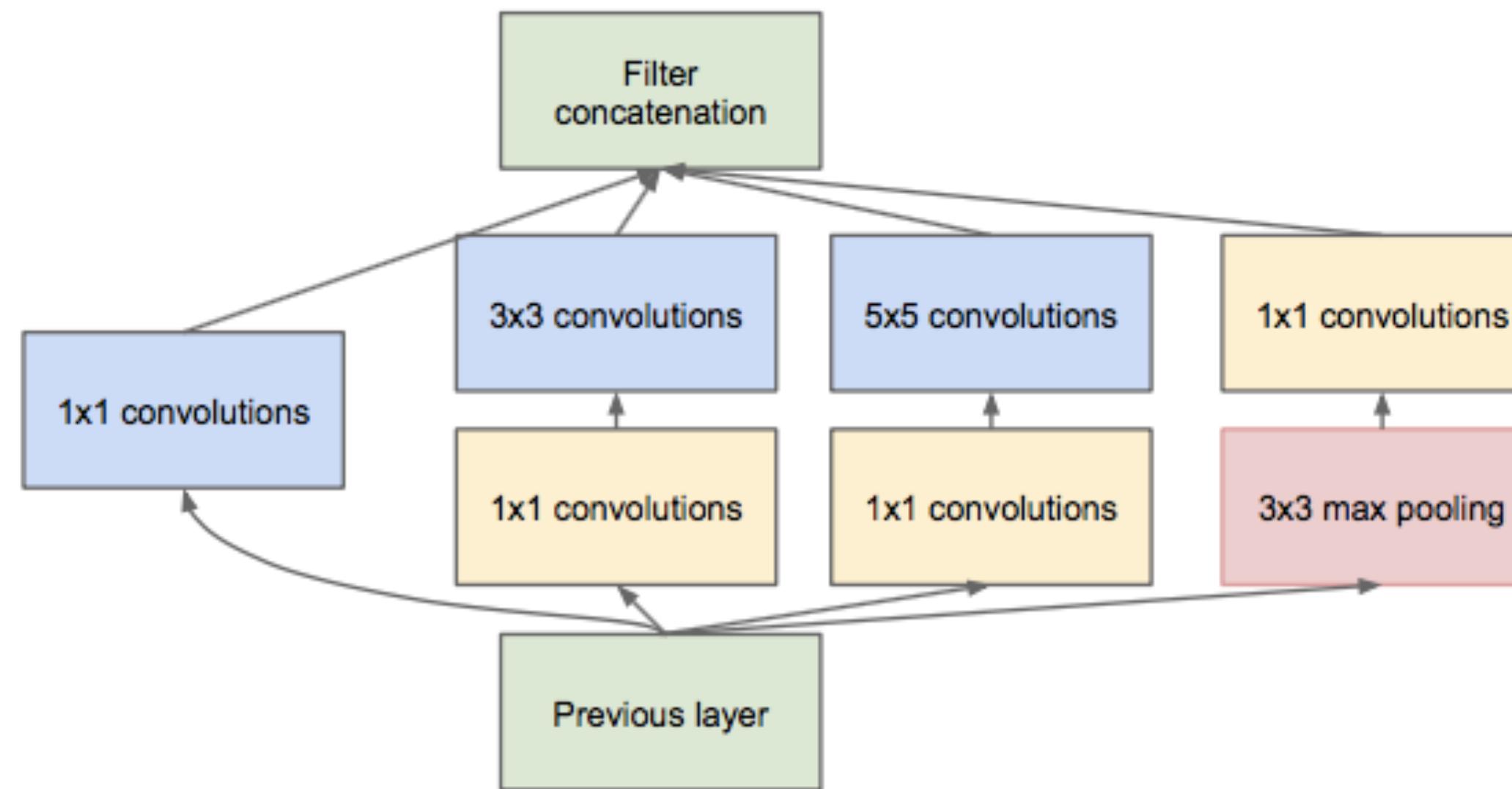
Convolutional filters with different sizes can cover different clusters of information. By finding the optimal local construction and repeating it spatially, they approximate the optimal sparse structure with dense components.

For convenience of computation, they use 1×1 , 3×3 and 5×5 filters + pooling. Together these made up the naive Inception module.



Inception Module: Naive Version

Stacking these inception modules on top of each would lead to an exploding number of outputs
Solution: inspired by "Network in Network" add 1x1 convolutions for dimensionality reduction



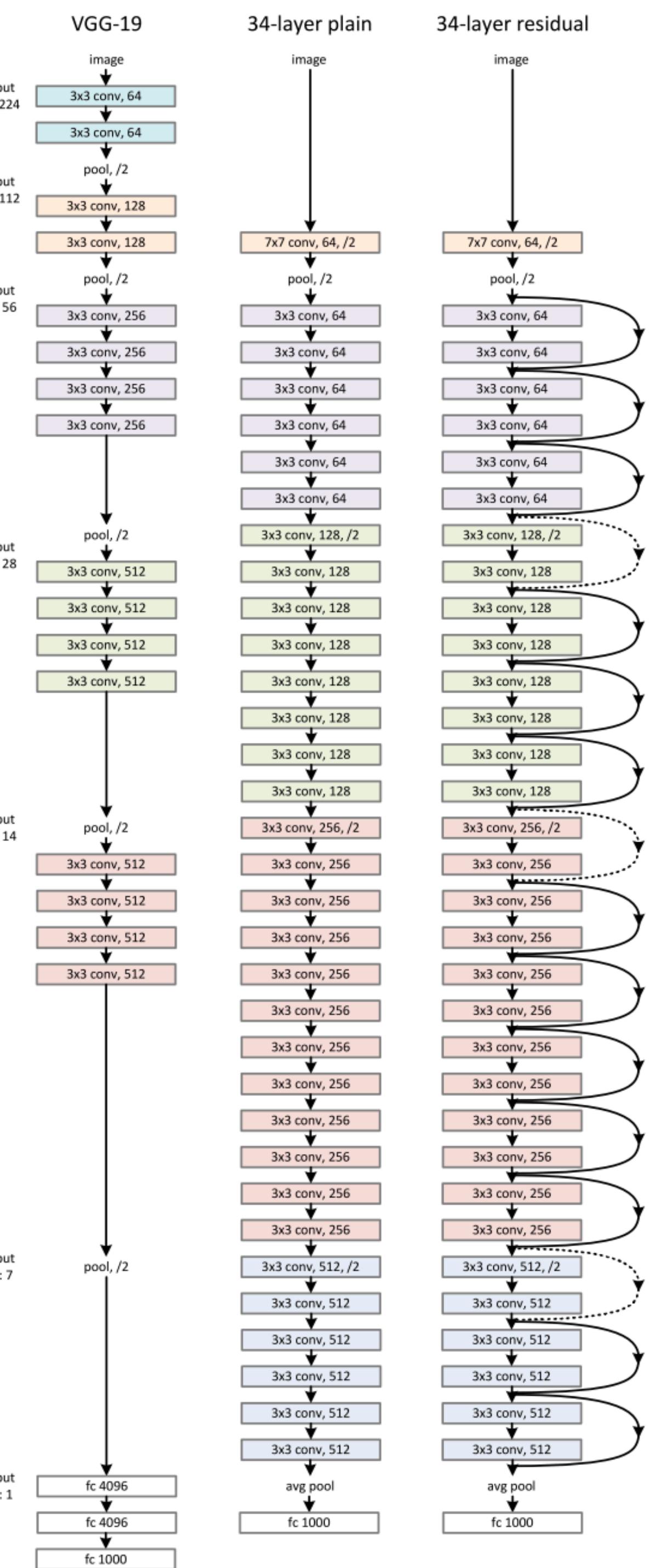
GoogleNet

type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	$7 \times 7 / 2$	$112 \times 112 \times 64$	1							2.7K	34M
max pool	$3 \times 3 / 2$	$56 \times 56 \times 64$	0								
convolution	$3 \times 3 / 1$	$56 \times 56 \times 192$	2		64	192				112K	360M
max pool	$3 \times 3 / 2$	$28 \times 28 \times 192$	0								
inception (3a)		$28 \times 28 \times 256$	2	64	96	128	16	32	32	159K	128M
inception (3b)		$28 \times 28 \times 480$	2	128	128	192	32	96	64	380K	304M
max pool	$3 \times 3 / 2$	$14 \times 14 \times 480$	0								
inception (4a)		$14 \times 14 \times 512$	2	192	96	208	16	48	64	364K	73M
inception (4b)		$14 \times 14 \times 512$	2	160	112	224	24	64	64	437K	88M
inception (4c)		$14 \times 14 \times 512$	2	128	128	256	24	64	64	463K	100M
inception (4d)		$14 \times 14 \times 528$	2	112	144	288	32	64	64	580K	119M
inception (4e)		$14 \times 14 \times 832$	2	256	160	320	32	128	128	840K	170M
max pool	$3 \times 3 / 2$	$7 \times 7 \times 832$	0								
inception (5a)		$7 \times 7 \times 832$	2	256	160	320	32	128	128	1072K	54M
inception (5b)		$7 \times 7 \times 1024$	2	384	192	384	48	128	128	1388K	71M
avg pool	$7 \times 7 / 1$	$1 \times 1 \times 1024$	0								
dropout (40%)		$1 \times 1 \times 1024$	0								
linear		$1 \times 1 \times 1000$	1							1000K	1M
softmax		$1 \times 1 \times 1000$	0								

ImageNet Challenge 2012-2014

Team	Year	Place	Error (top-5)	External data
SuperVision – Toronto (7 layers)	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai – NYU (7 layers)	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
VGG – Oxford (16 layers)	2014	2nd	7.32%	no
GoogLeNet (19 layers)	2014	1st	6.67%	no
<u>Human expert*</u>			5.1%	

ResNet



Training a CNN

- Back-propagation + stochastic gradient descent with momentum
- Dropout
- Data Augmentation
- Batch Normalization
- Initialization
- Transfer Learning
- The make use of **GPU's** in order to optimize the weights.

Reducing Overfitting

Data Augmentation

60 million parameters, 650,000 neurons
-> overfits a lot

Crop 224x224 patches (and their horizontal reflections)

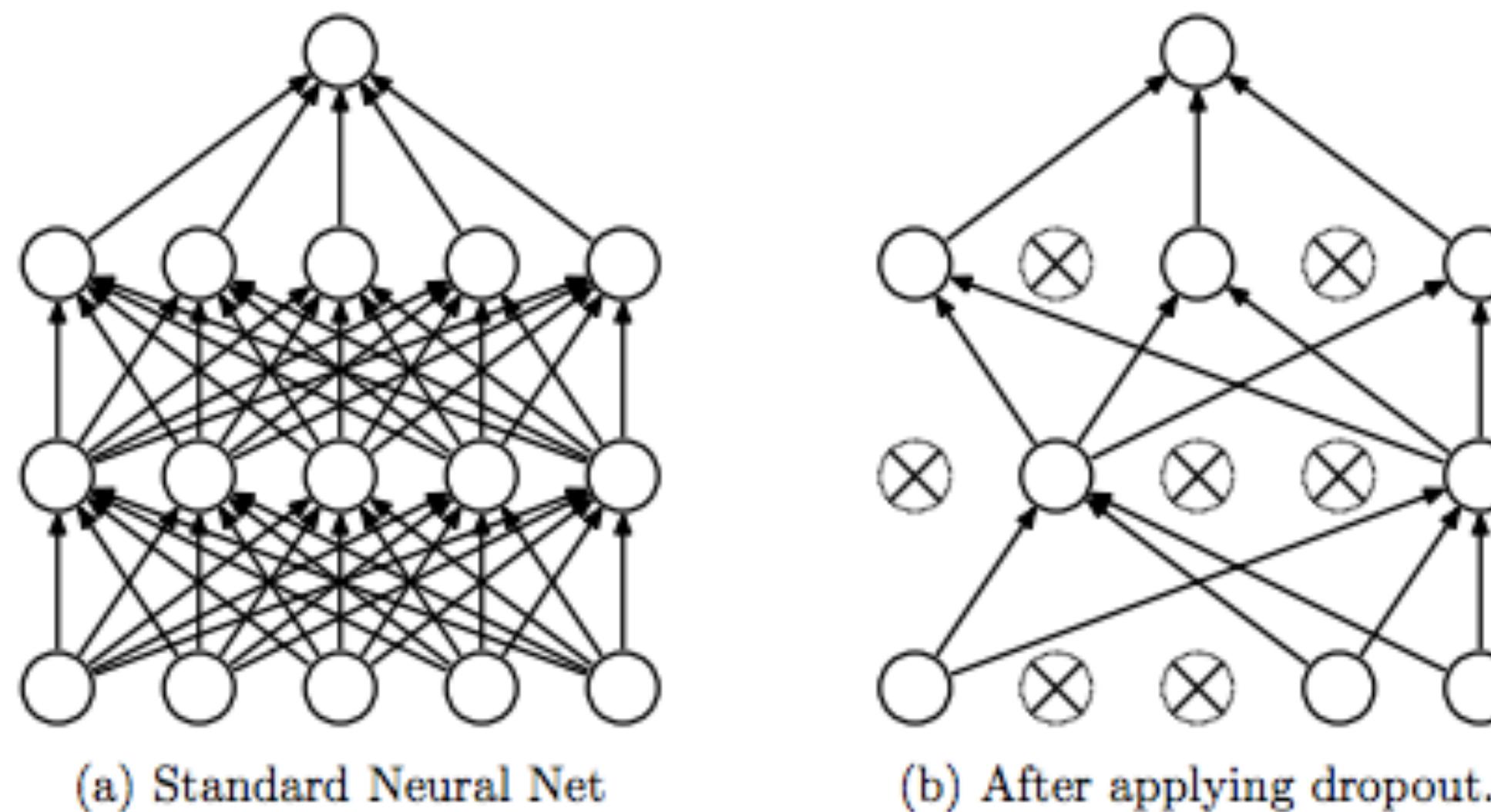
Data Augmentation at test

average the predictions on the 10 patches.

Reducing Overfitting

Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Journal of Machine Learning Research 15 (2014) 1929-1958



Exercice

- How many parameter do the model has if:
 - it has 10 input values
 - 2 neurons in one hidden layer
 - 1 neuron in the output layer

Exercice

- How many parameter do the model has if:
 - it has 10 input values
 - 2 neurons in one hidden layer
 - 1 neuron in the output layer
- How many parameter do the model has if:
 - it has 10 input values
 - 25 neurons in the first hidden layer
 - 25 neurons in the second hidden layer
 - 1 neuron in the output layer

Exercice

- How many parameter do the model is a RGB image of 50x50:
 - 1 convolution layer with 10 filters of size 5x5
 - Max-pooling 2x2 with stride 2
 - 1 convolution layer with 20 filters of size 3x3
 - Dense Layer with 100 neurons
 - Dense Layer with 10 neurons

Learning algorithm

- Learning the parameters is a complex and time-consuming task.
- **Huge amount** of **data** is needed.
- **Back propagation** method is used.
- Great **packages** like Caffe, Theano or TensorFlow.
- The make use of **GPU's** in order to optimize the weights.