

Deep Learning From Scratch

Neural Networks: Part 2



Santi Seguí

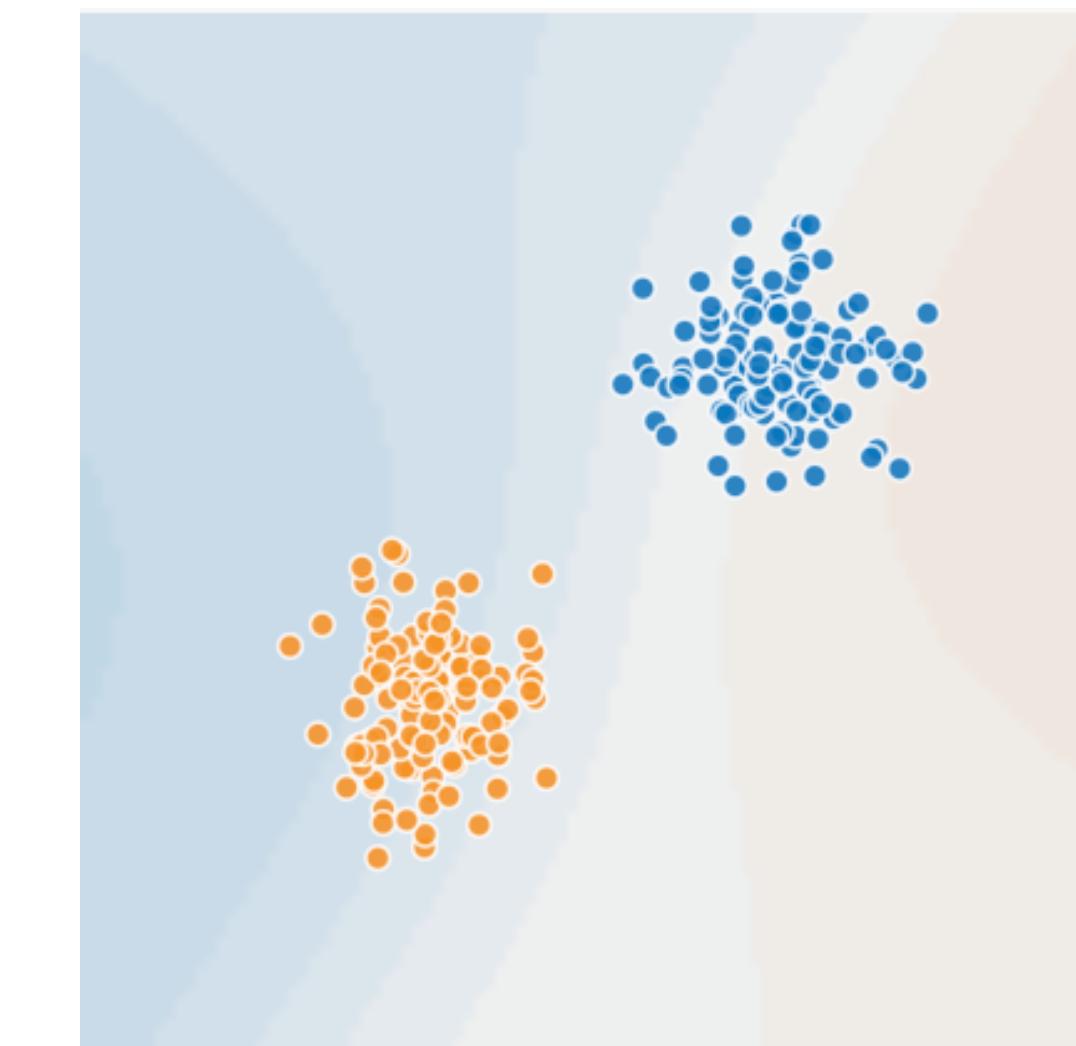
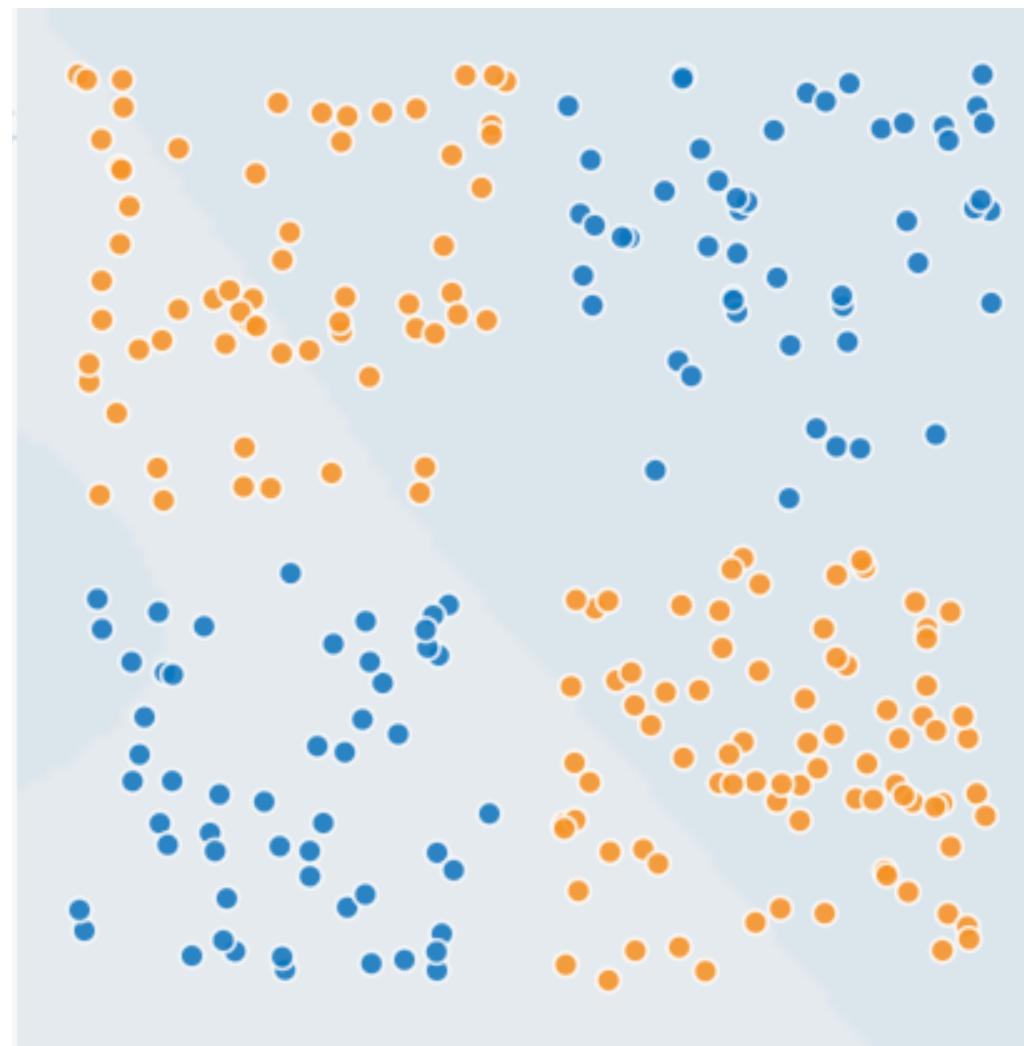


Associate Professor at the Department of Mathematics and Computer Science
from the University of Barcelona.

PhD in 2011 on Computer Vision and Machine Learning
Graduate on Computer Science in 2007

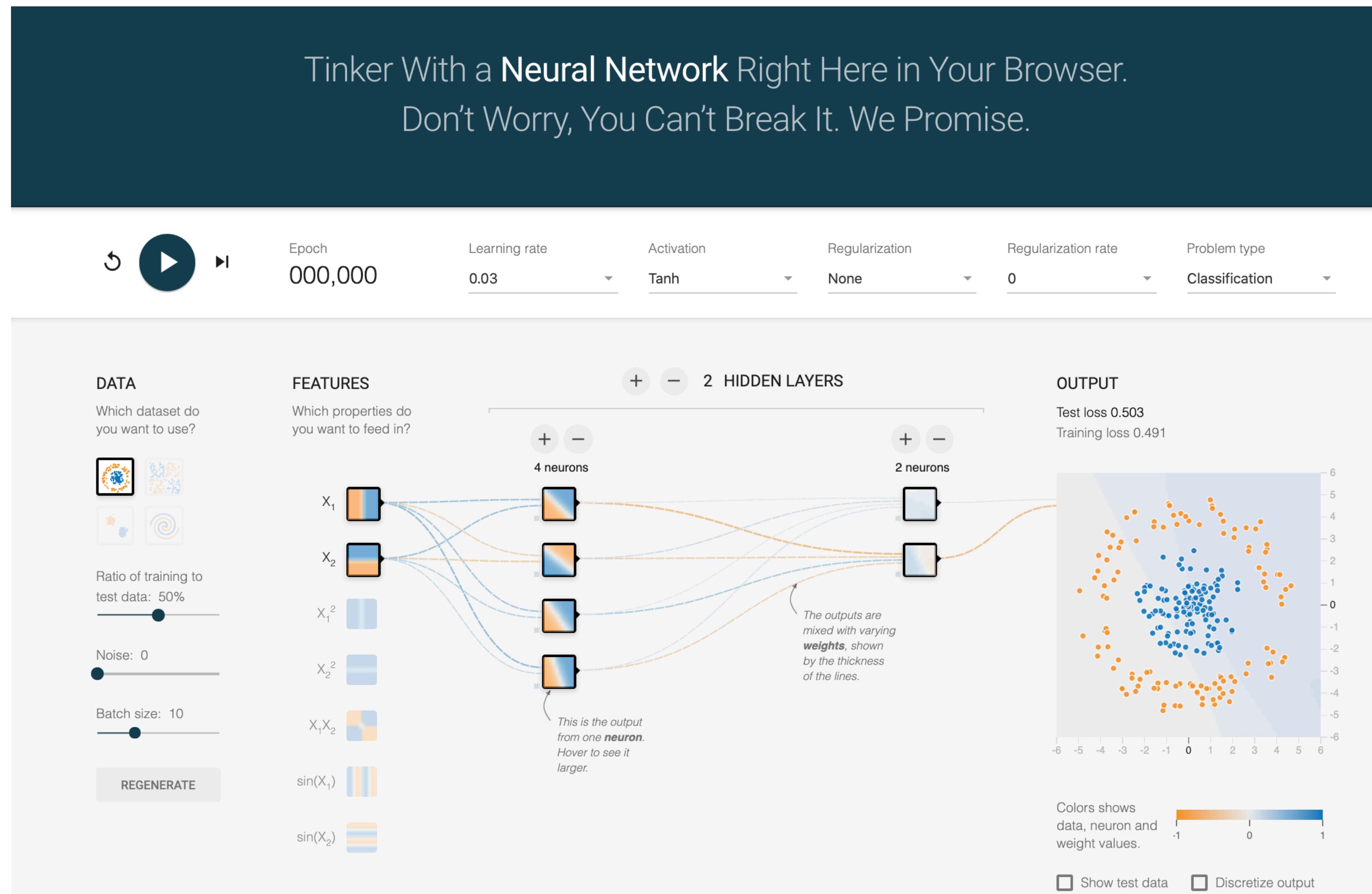
Santi Seguí Email: santi.segui@ub.edu

wrap up!

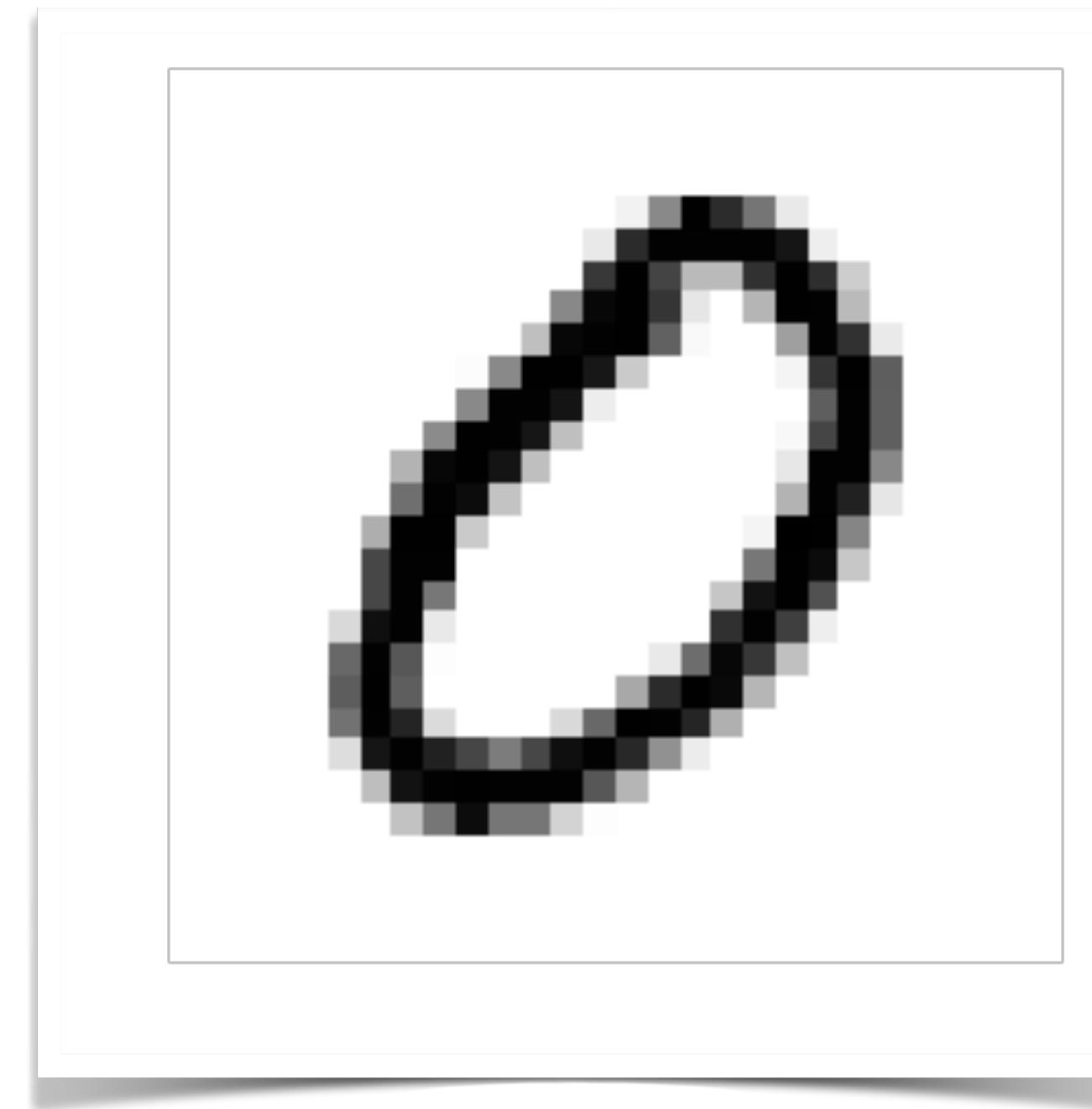


How can we create a Neural Network able to deal with these datasets?

<https://playground.tensorflow.org/>



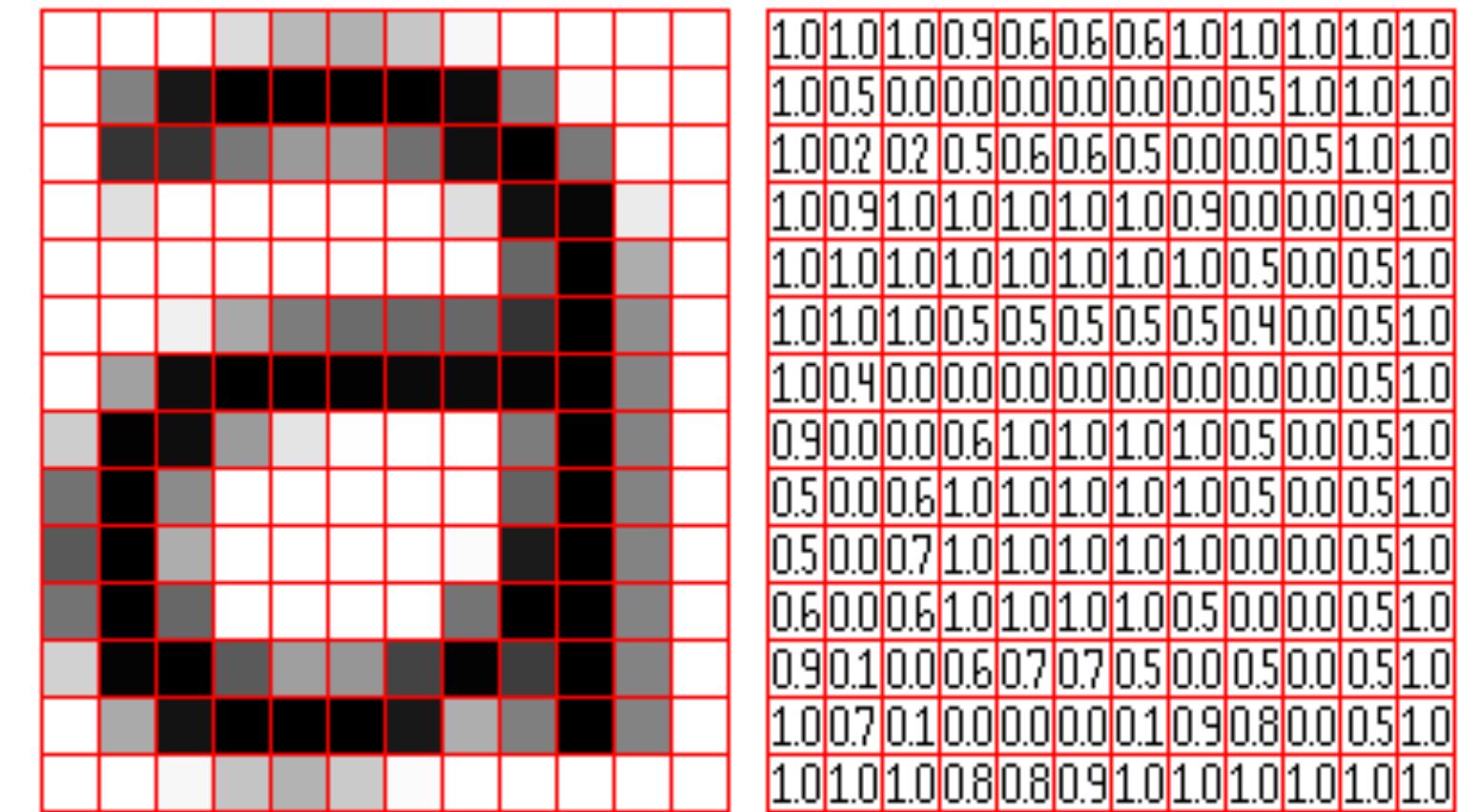
wrap up!



How can we create a Neural Network able to detect the image digit?

An image

- An image is a matrix of size **$m \times n \times C$** pixels

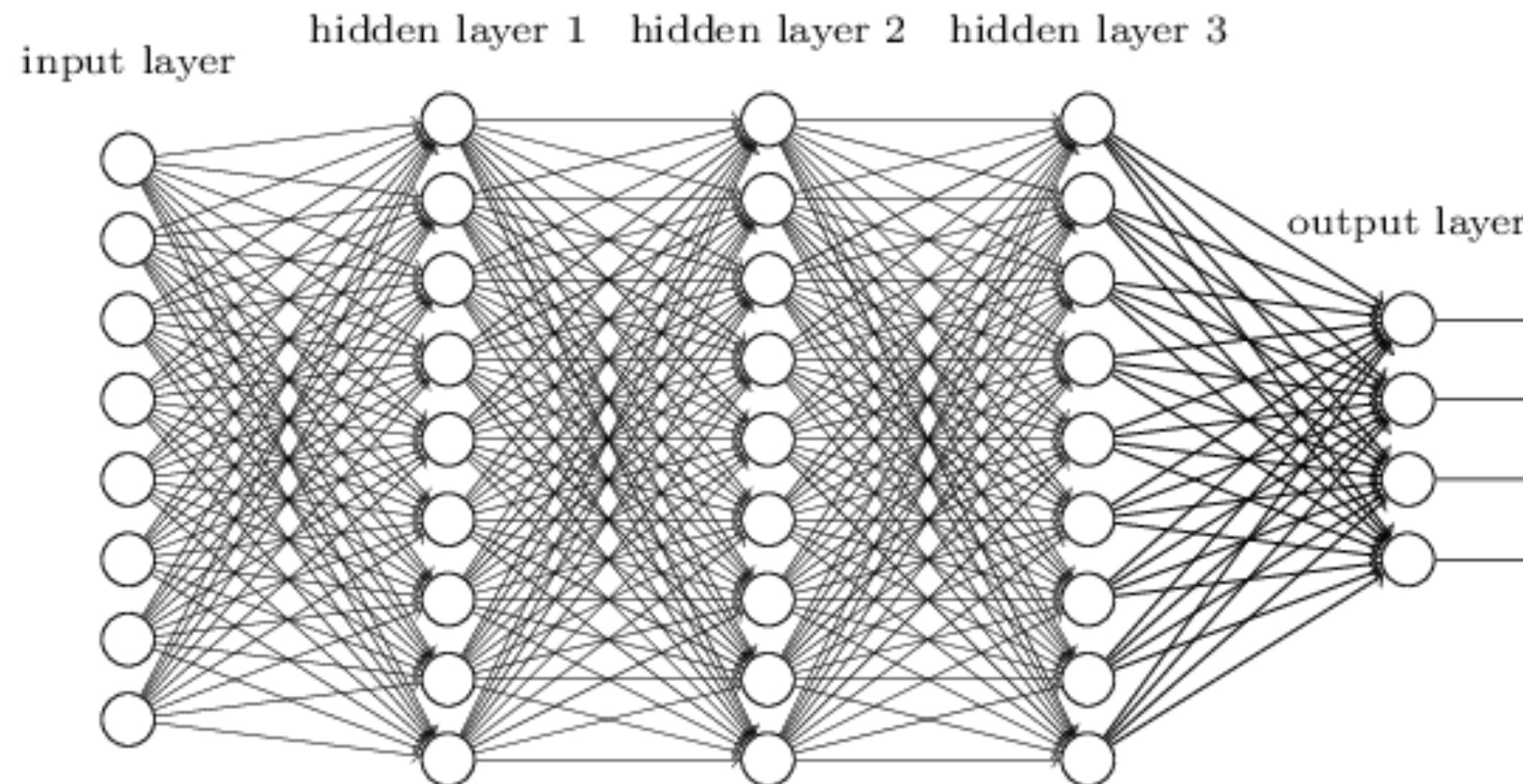


Neural Networks for Images



Neural Networks for Images

Multi Layer Perceptron



How many parameter does this MLP has?

$$8 * 9 + 9 * 9 + 9 * 9 + 9 * 4 = 270$$

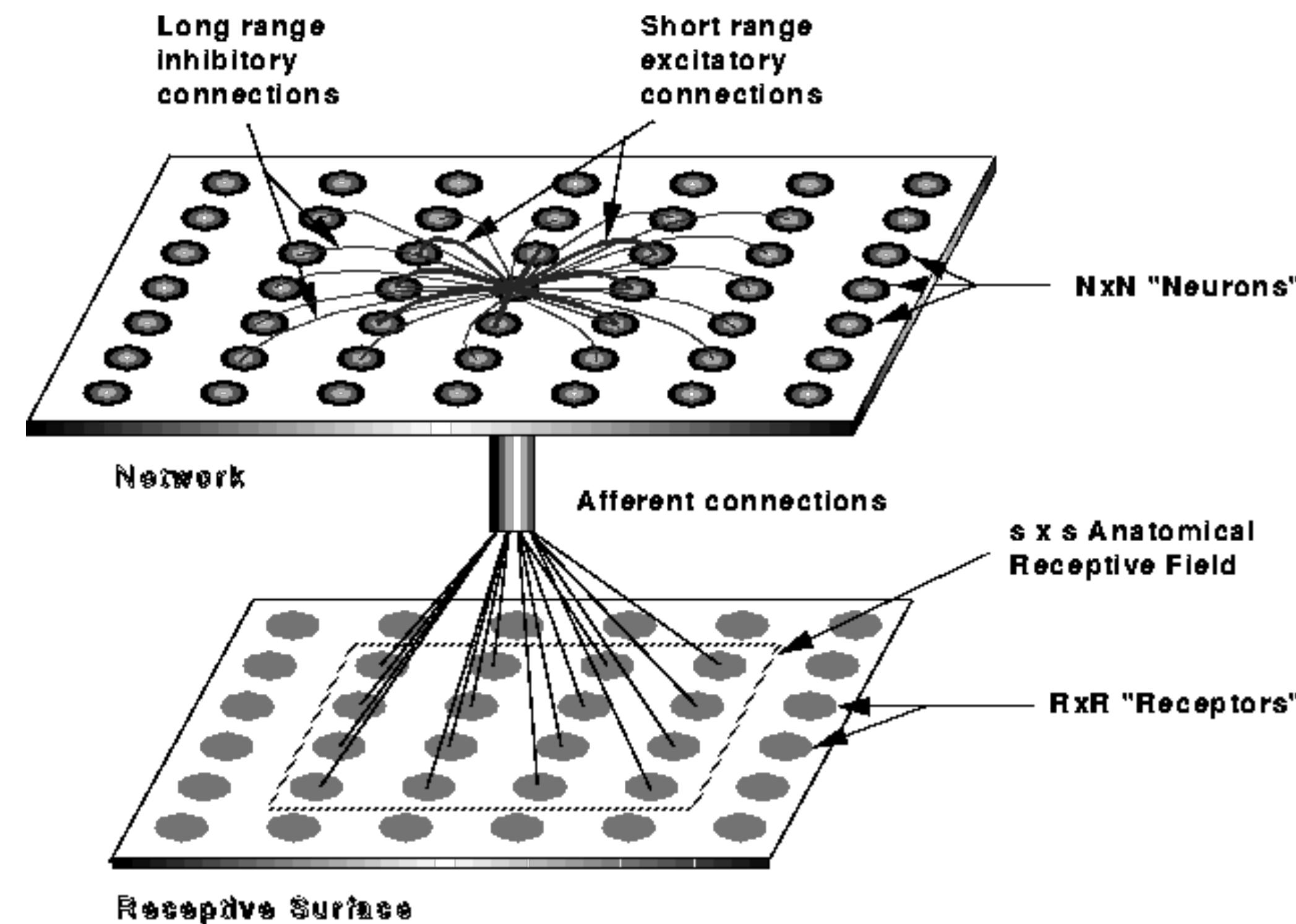
Neural Networks for Images



Neural Networks for Images

- Input is a standard vector of size $N \times M \times C$
 - Imagine a 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!

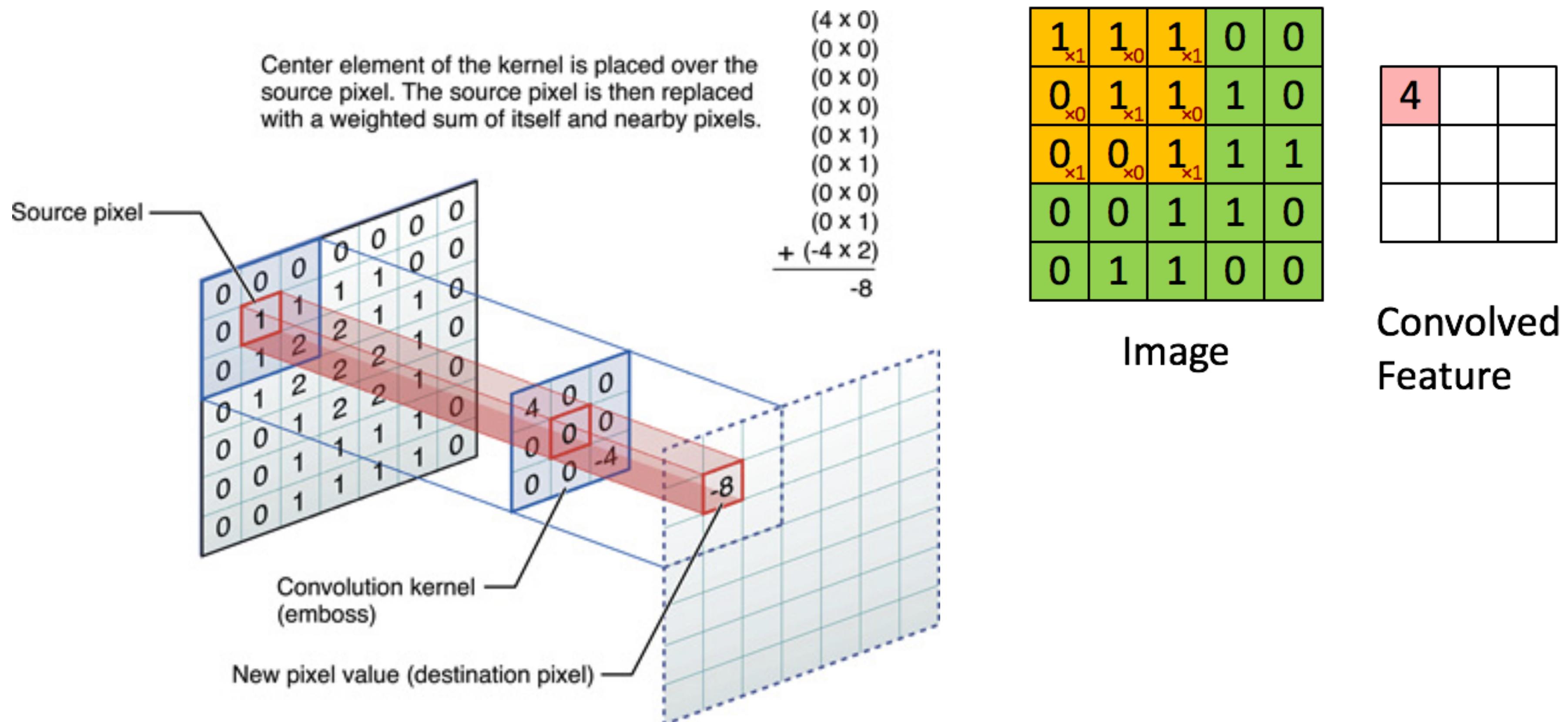
Convolutional Neural Networks (CNNs)

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

Convolutional Neural Networks (CNNs)

- Repetitive blocks of neurons that are applied across space (for images) or time (for audio signals etc).
- For images, these blocks of neurons can be interpreted as 2D convolutional kernels, repeatedly applied over each patch of the image.
- For speech, they can be seen as the 1D convolutional kernels applied across time-windows.
- At training time, the weights for these repeated blocks are 'shared', i.e. the weight gradients learned over various image patches are averaged.

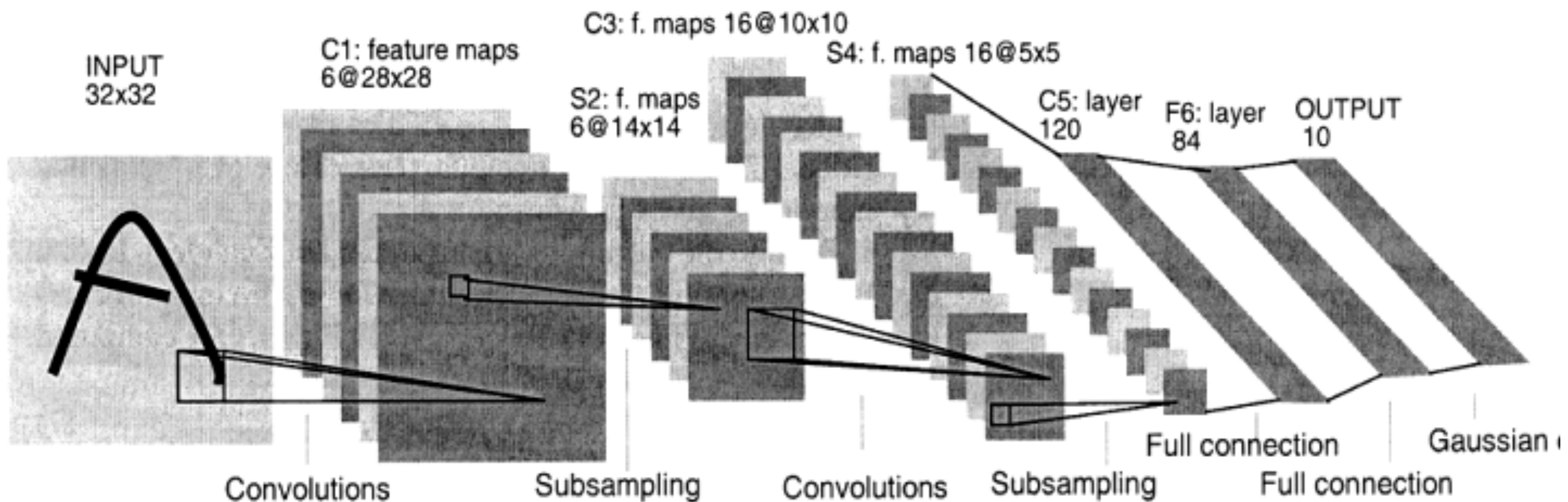
What is an image convolution?



What is an image convolution?



“Nothing New”

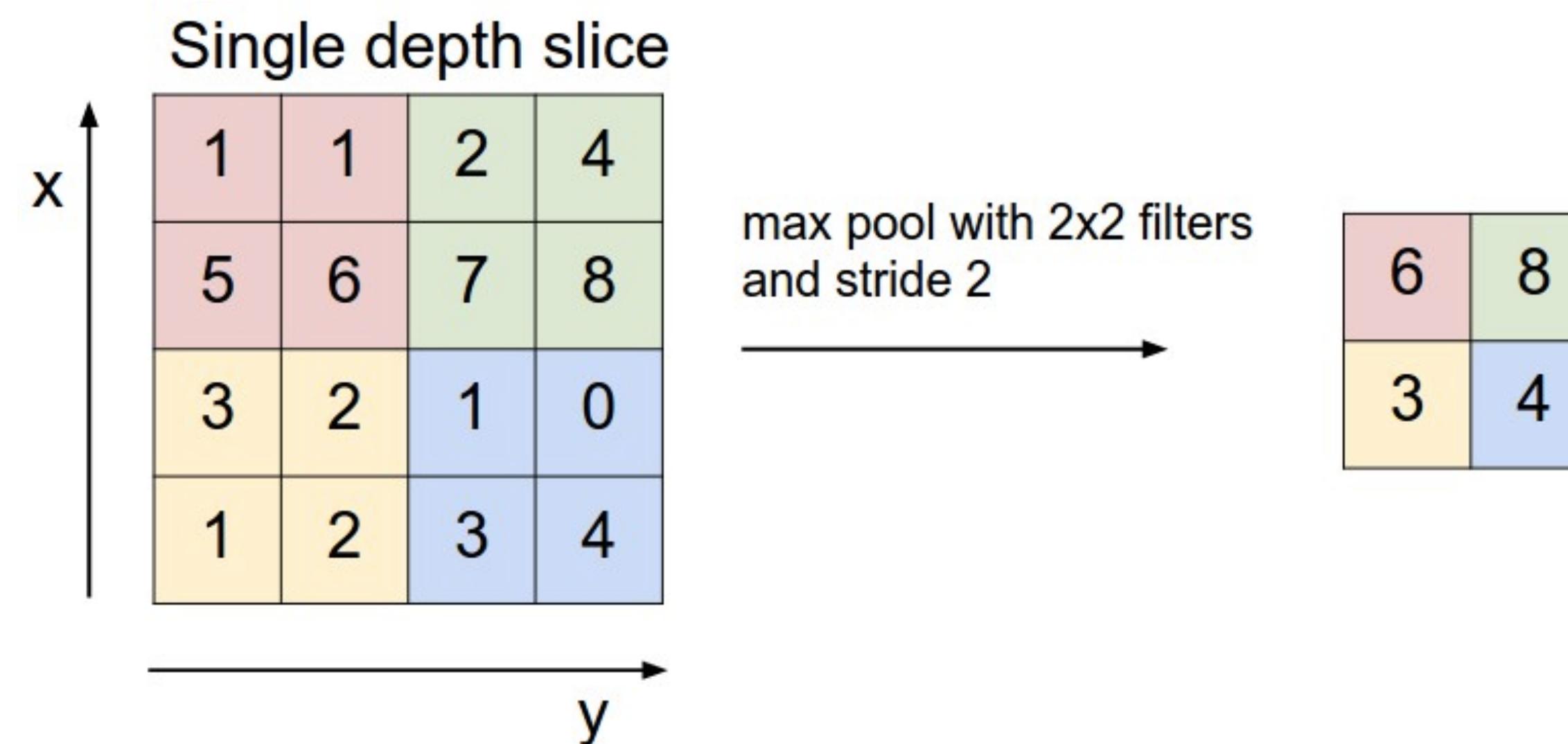


LeCun et al. 1998

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



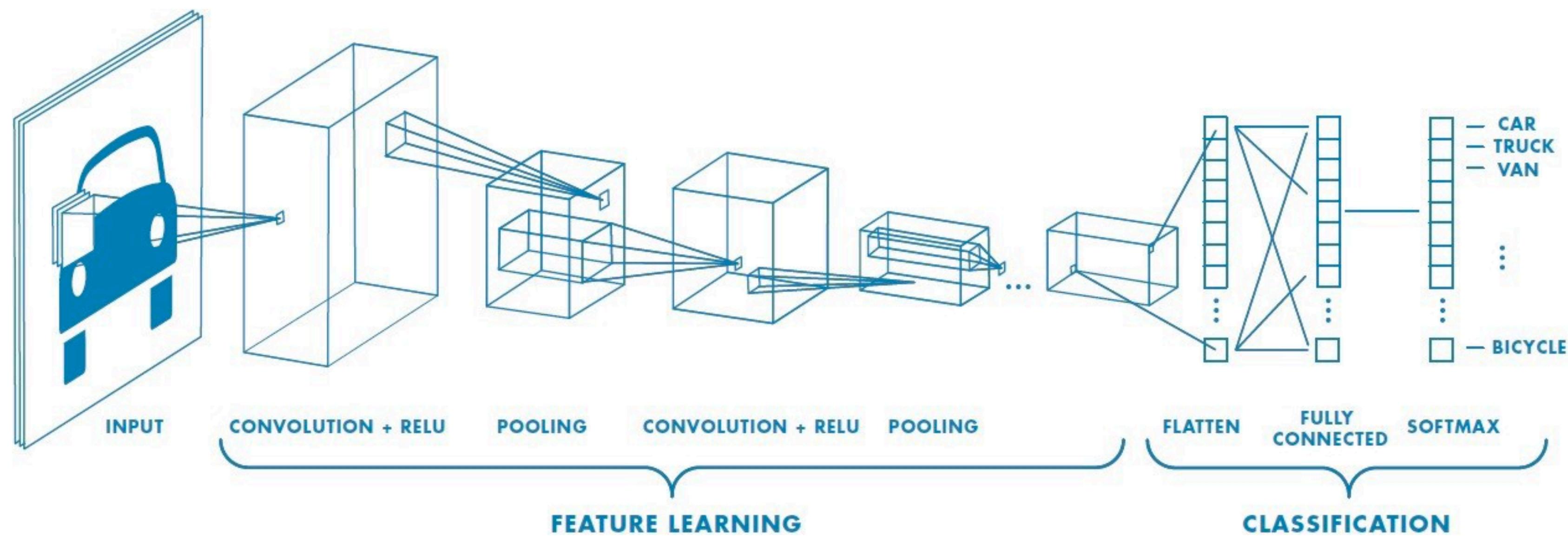
But it is also useful since it provides a form of translation **invariance**

Question:

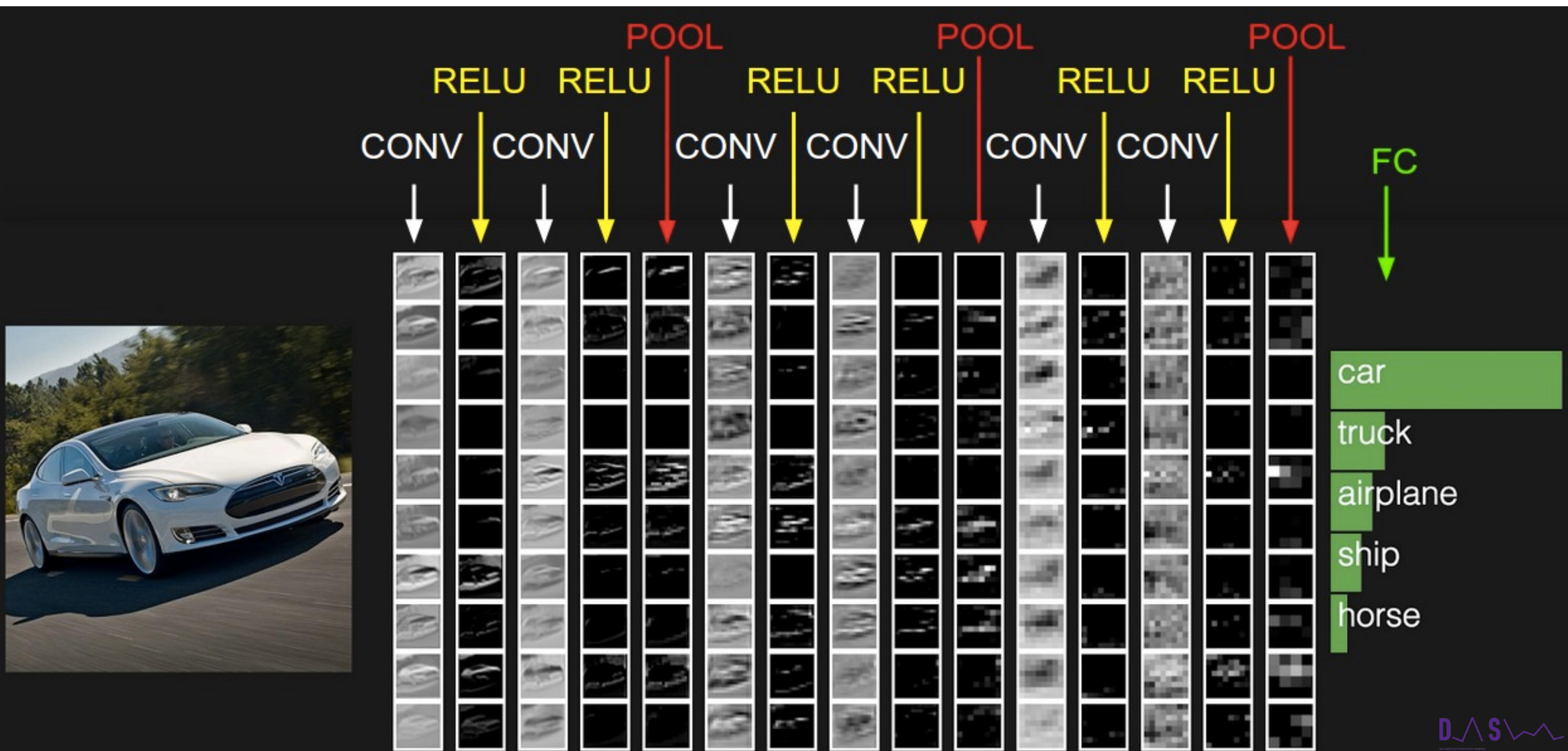
Why MAX Pooling

and not

Average Pooling?

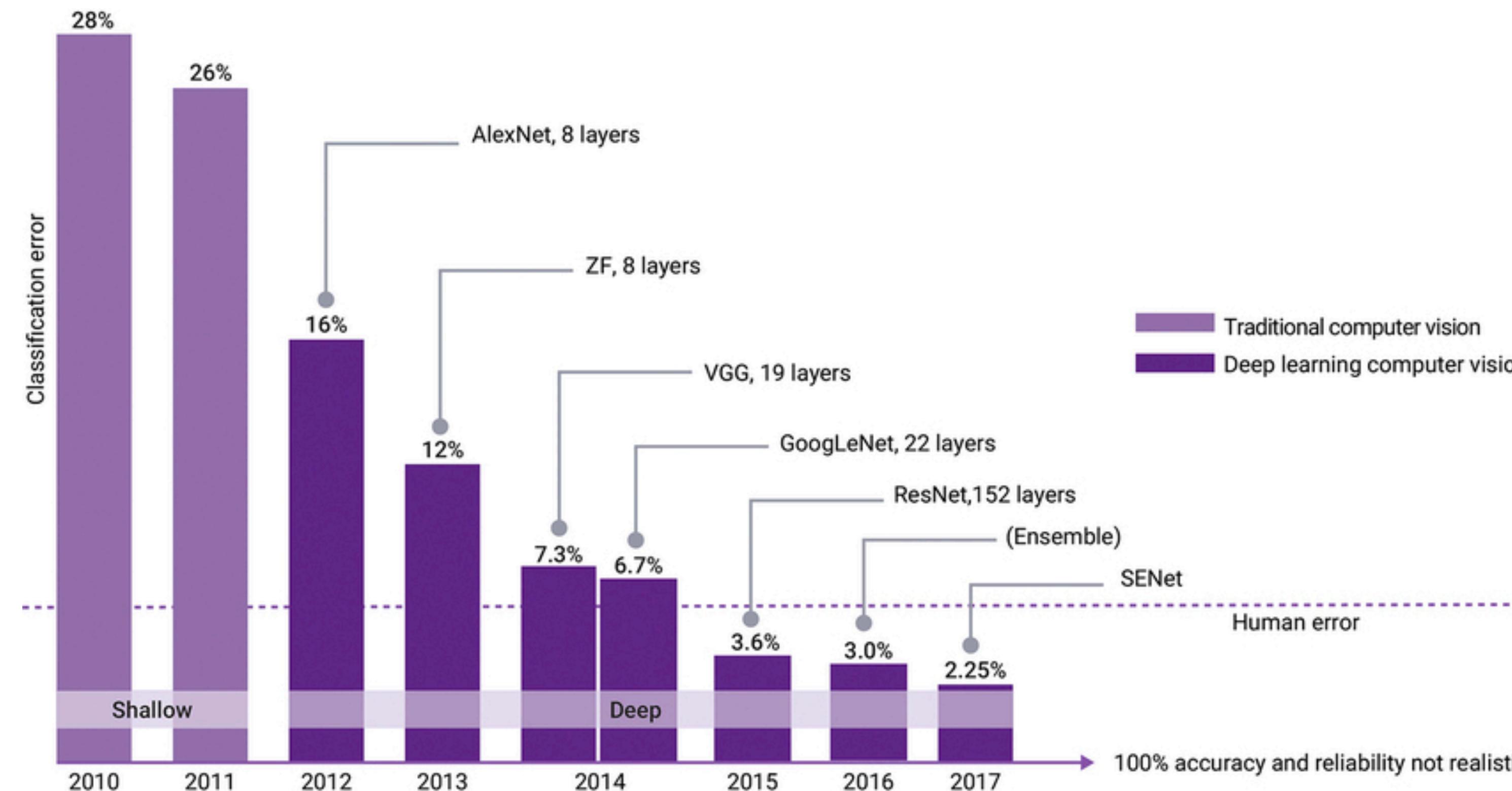


Finally..



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:

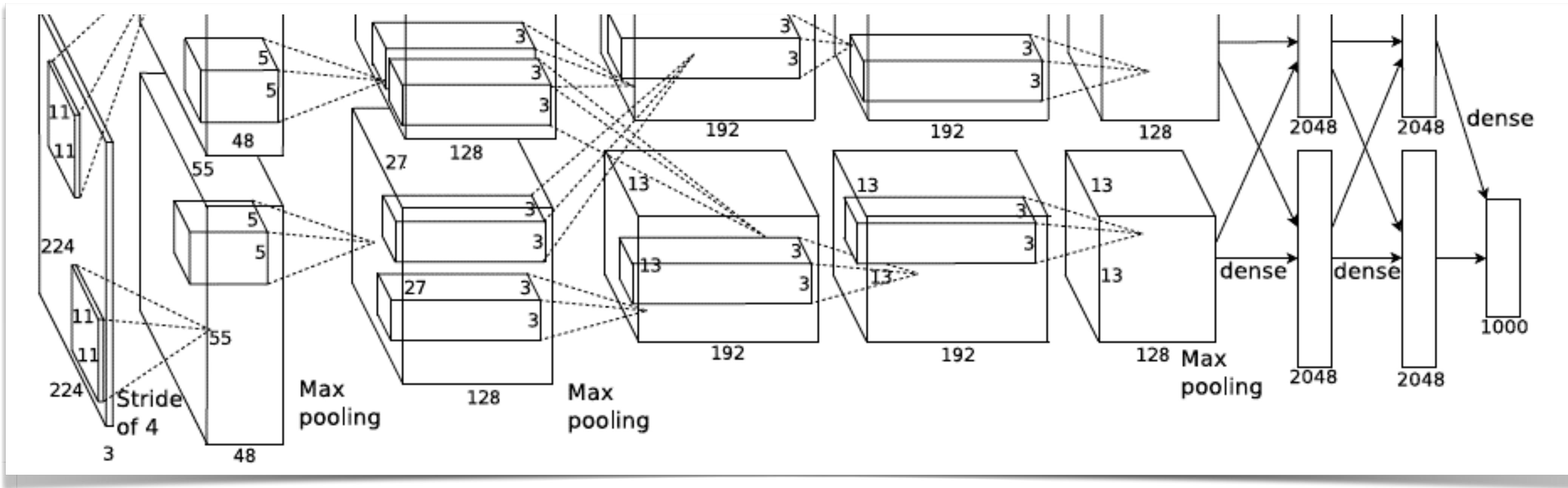
Bigger model:

7 hidden layers, 650.000 units, 60 million parameter

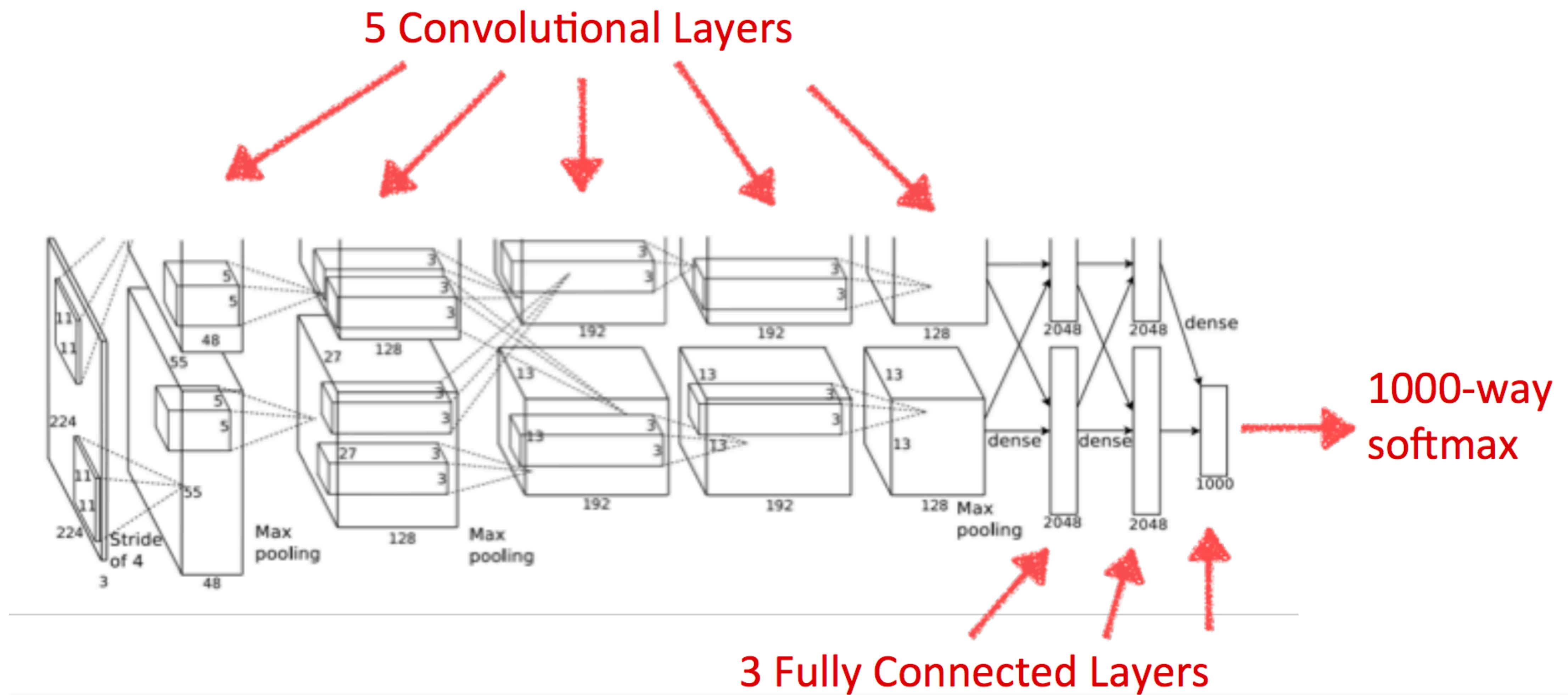
More Data:

10^6 vs 10^3 images

GPU implementation (50x speedup over CPU)

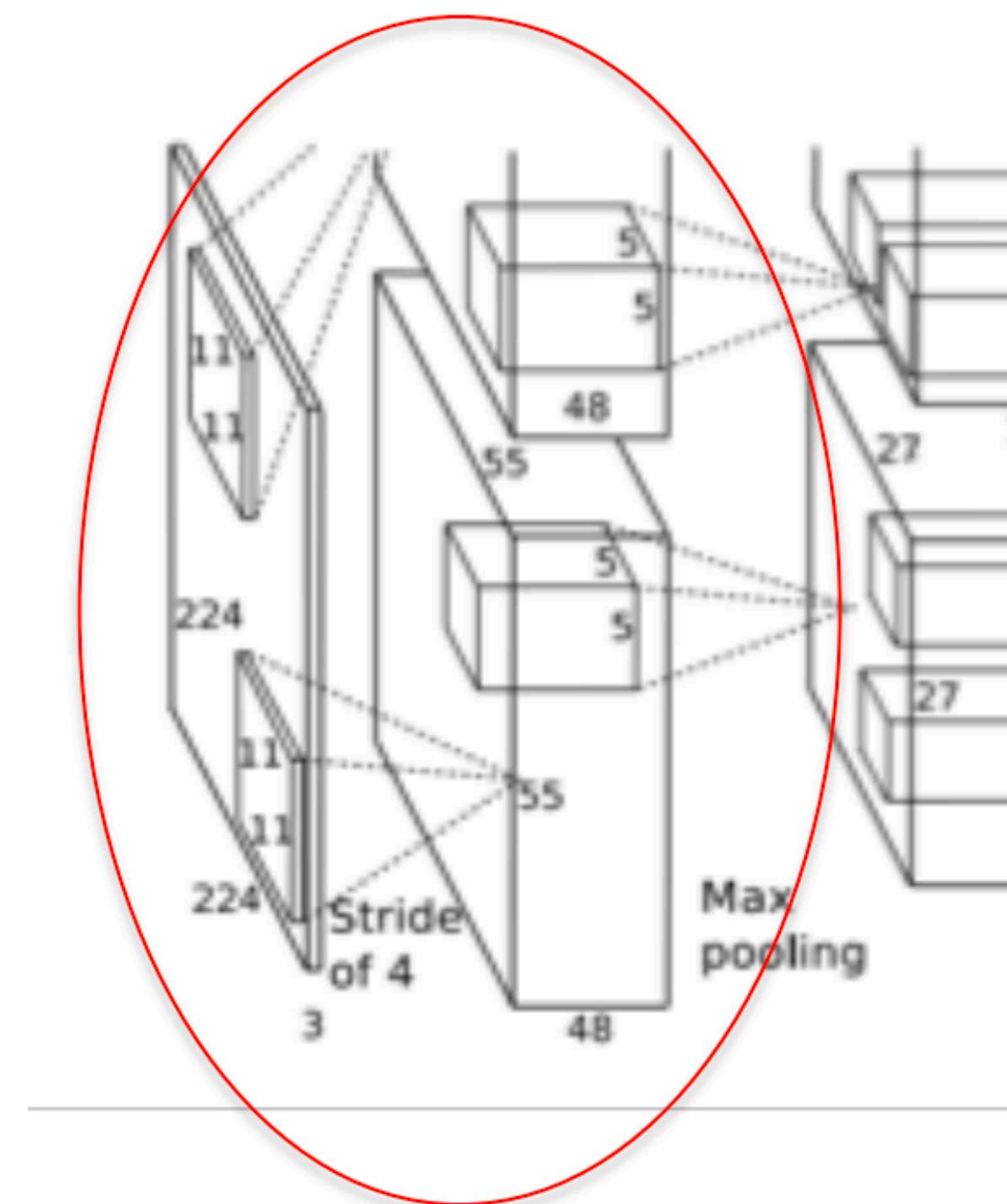


AlexNet



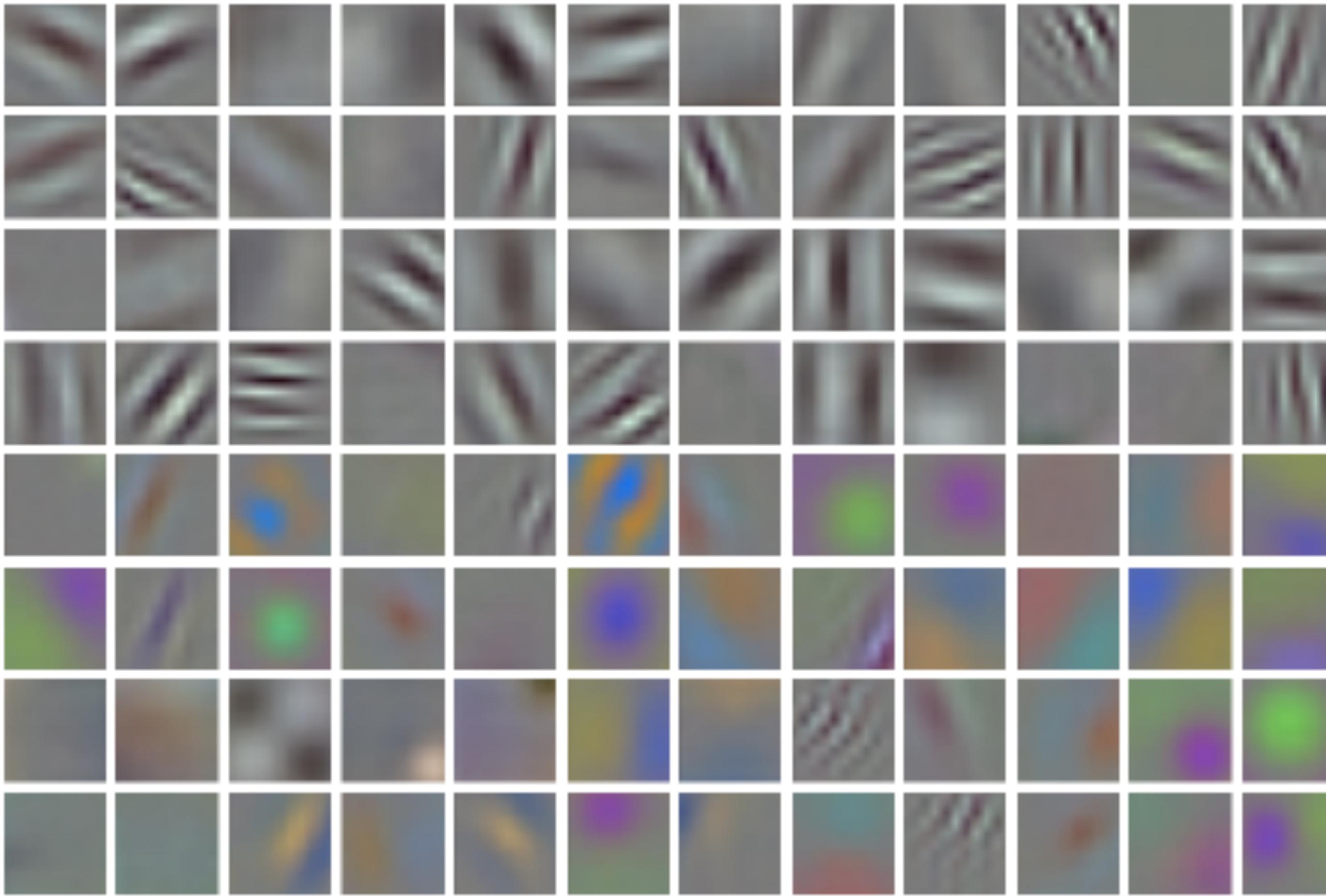
AlexNet

1st Convolution Layer

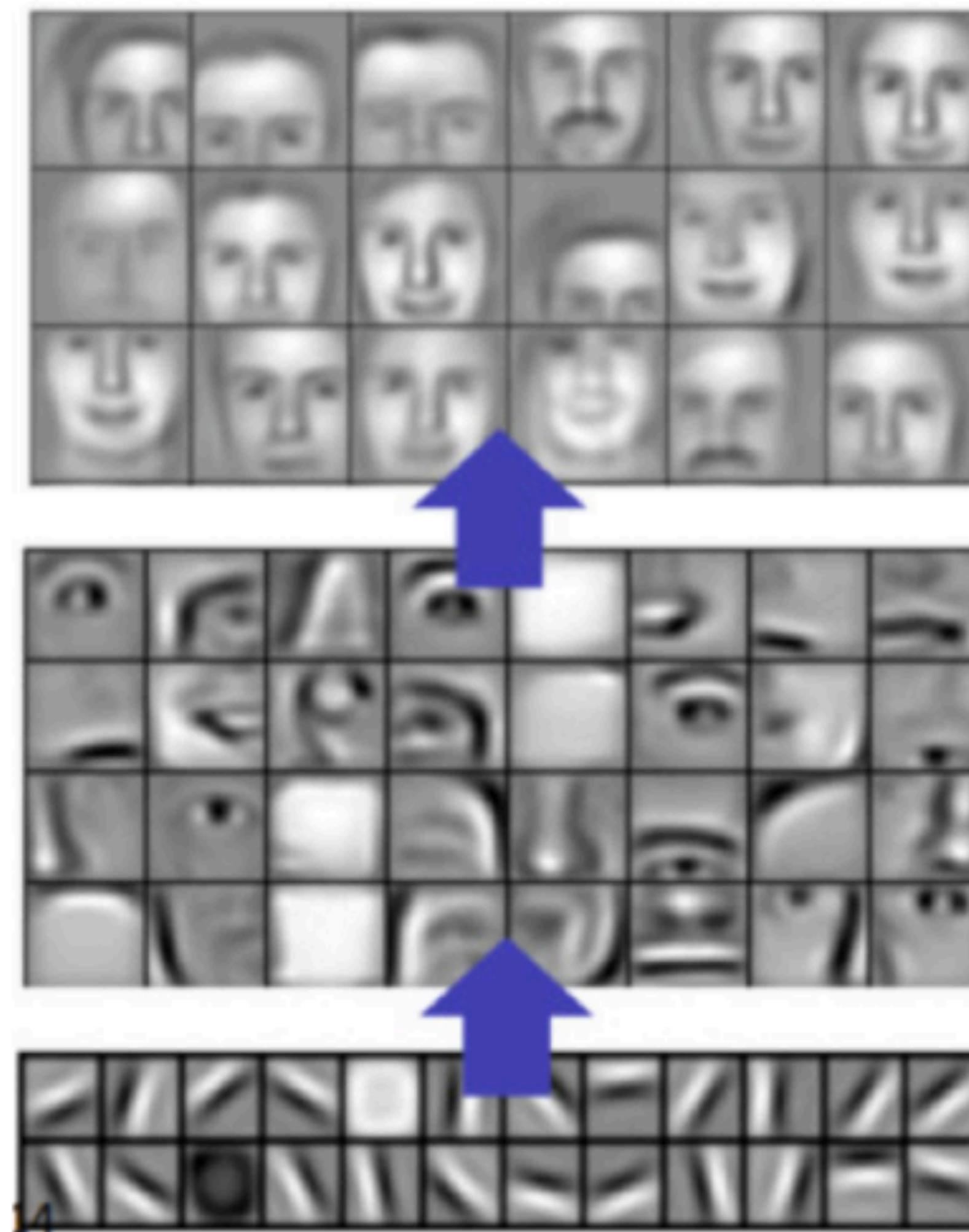


- Images: 227x227x3
- F (receptive field size): 11
- S (stride) = 4
- Conv layer output: 55x55x96

Alexnet 1st Conv Filters



Alexnet



Layer 3

Layer 2

Layer 1

Reducing Overfitting

Data Augmentation

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

Crop 224x224 patches (and their horizontal reflections)

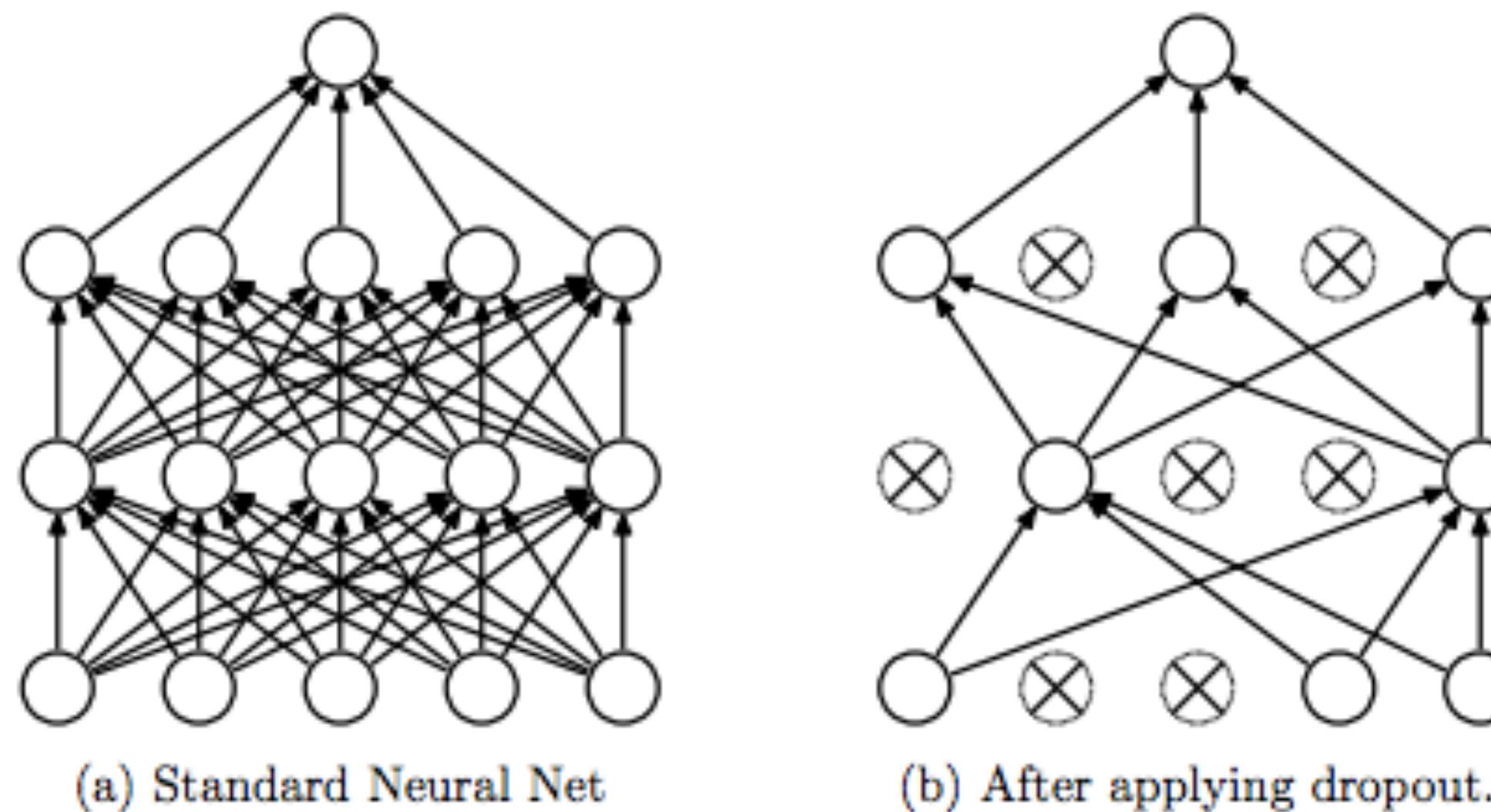
Data Augmentation at test

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

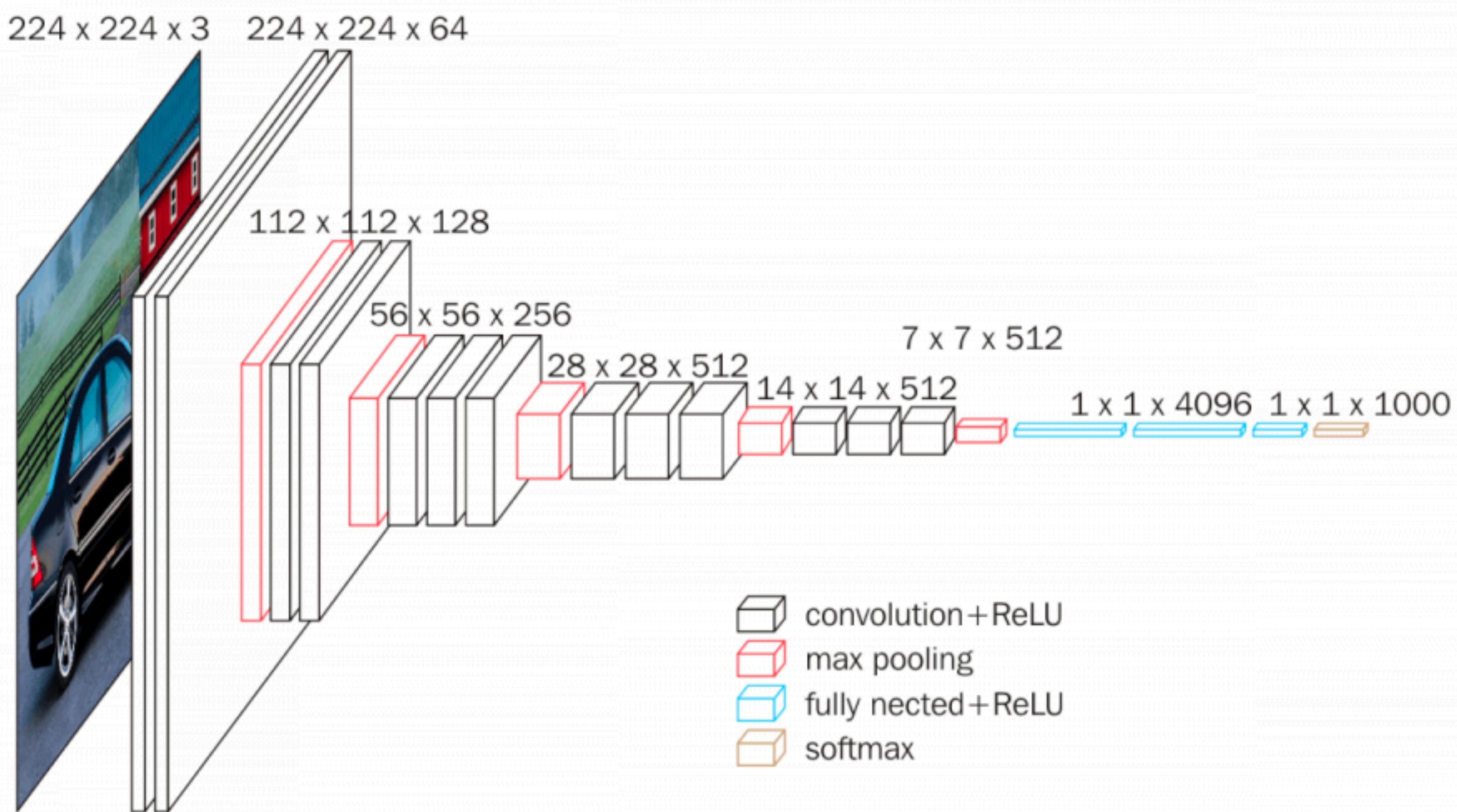
- How many parameter do the model is a RGB image of 50x50:
 - 1 convolution layer with 10 filters of size 5x5
 - Max-pooling 2x2 with stride 2
 - 1 convolution layer with 20 filters of size 3x3
 - Dense Layer with 100 neurons
 - Dense Layer with 10 neurons



WE NEED TO GO

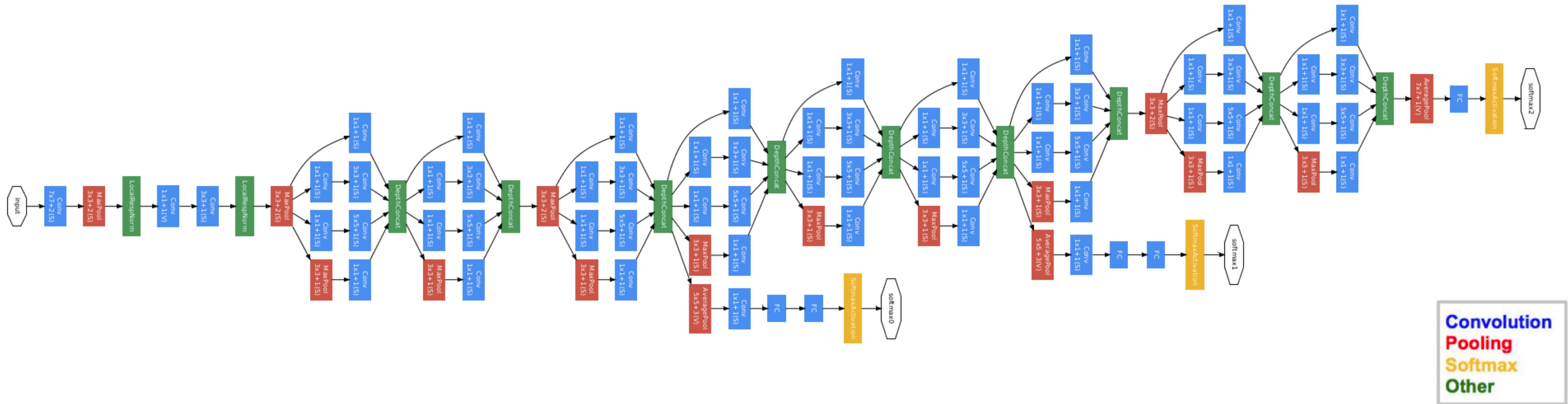
DEEPER

VGG Net



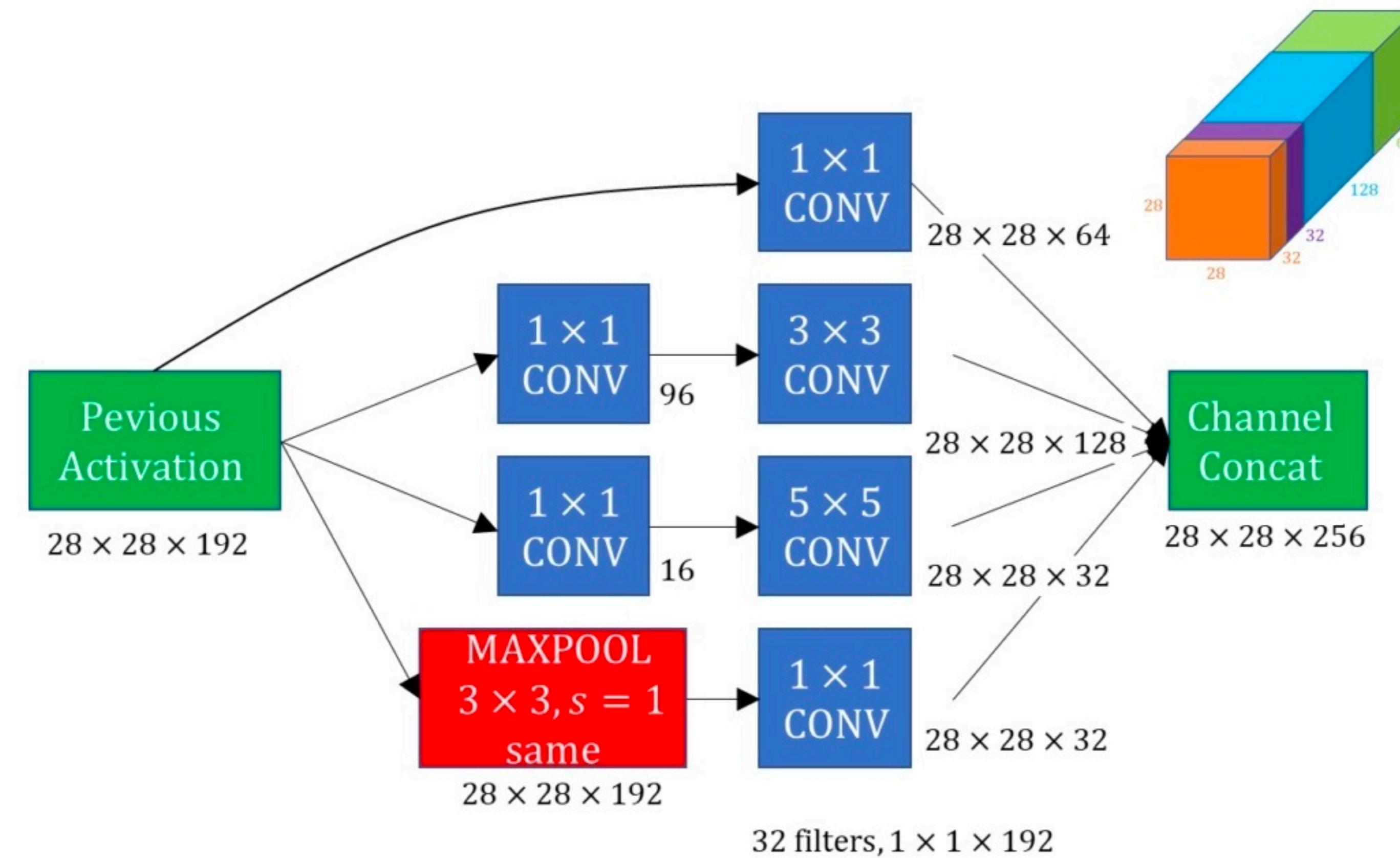
GoogLeNet

22 layers, but 12 times less parameters than AlexNet



GoogleNet

An example of an Inception module

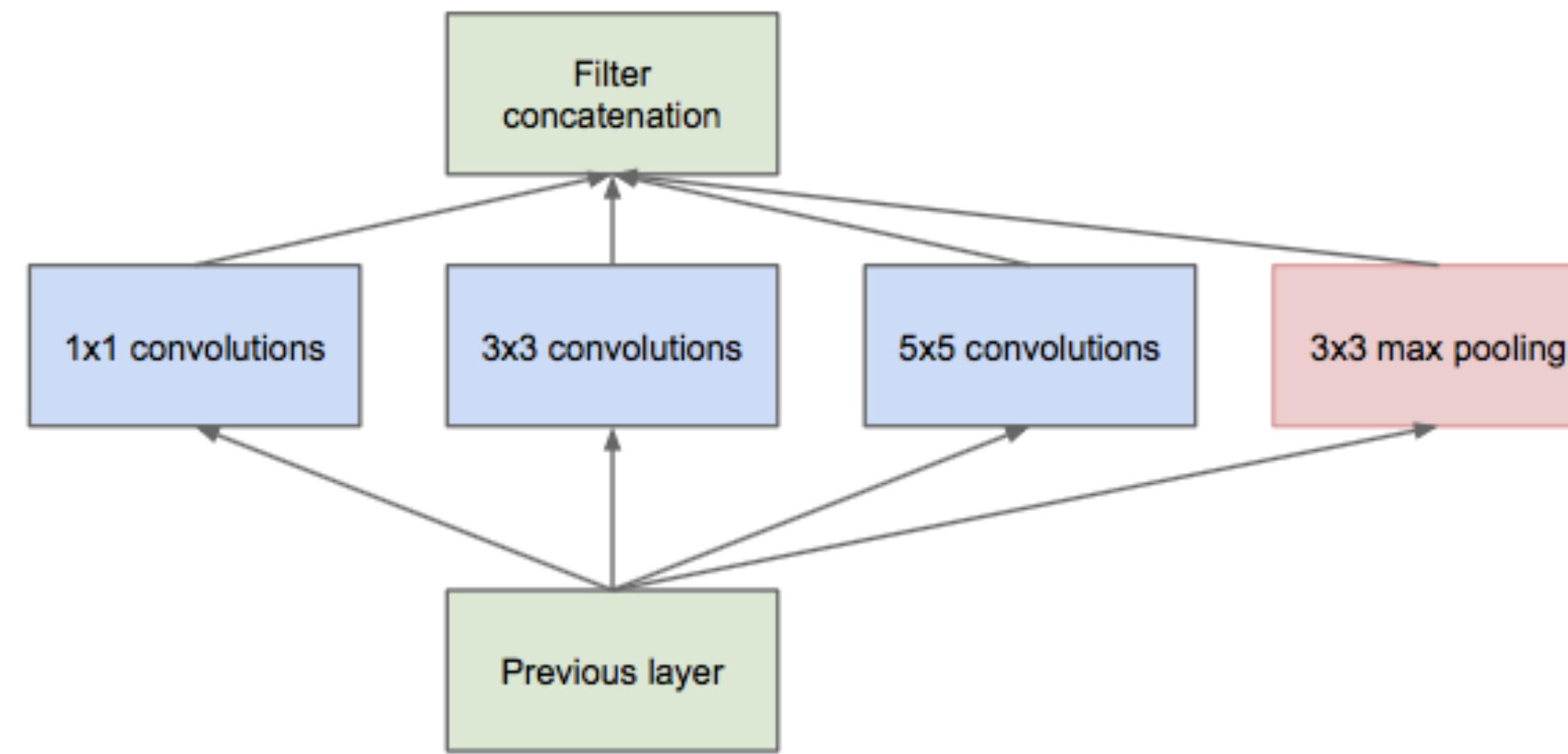


Inception Module: Naive Version

Convolutional filters with different sizes can cover different clusters of information.

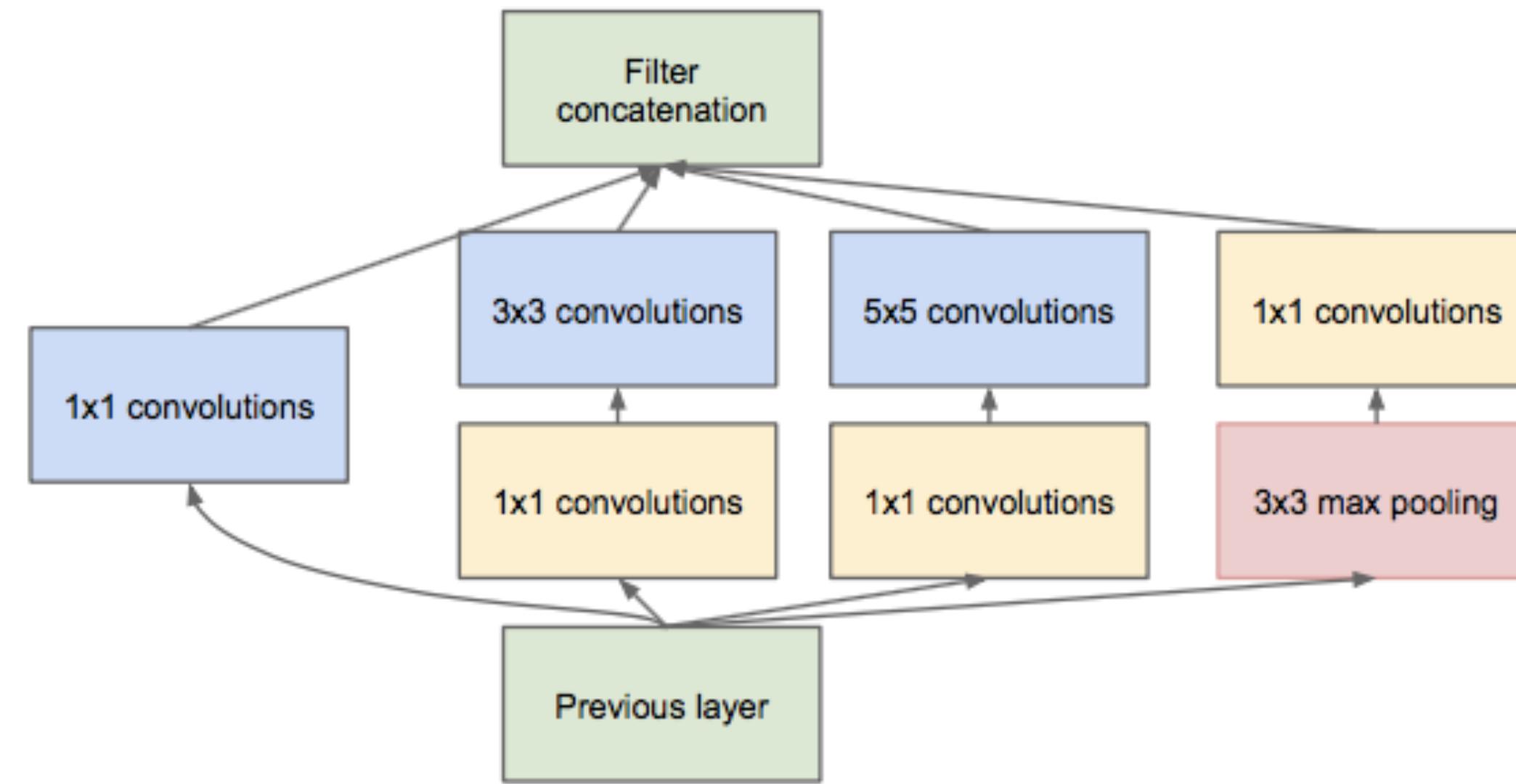
By finding the optimal local construction and repeating it spatially, they approximate the optimal sparse structure with dense components.

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



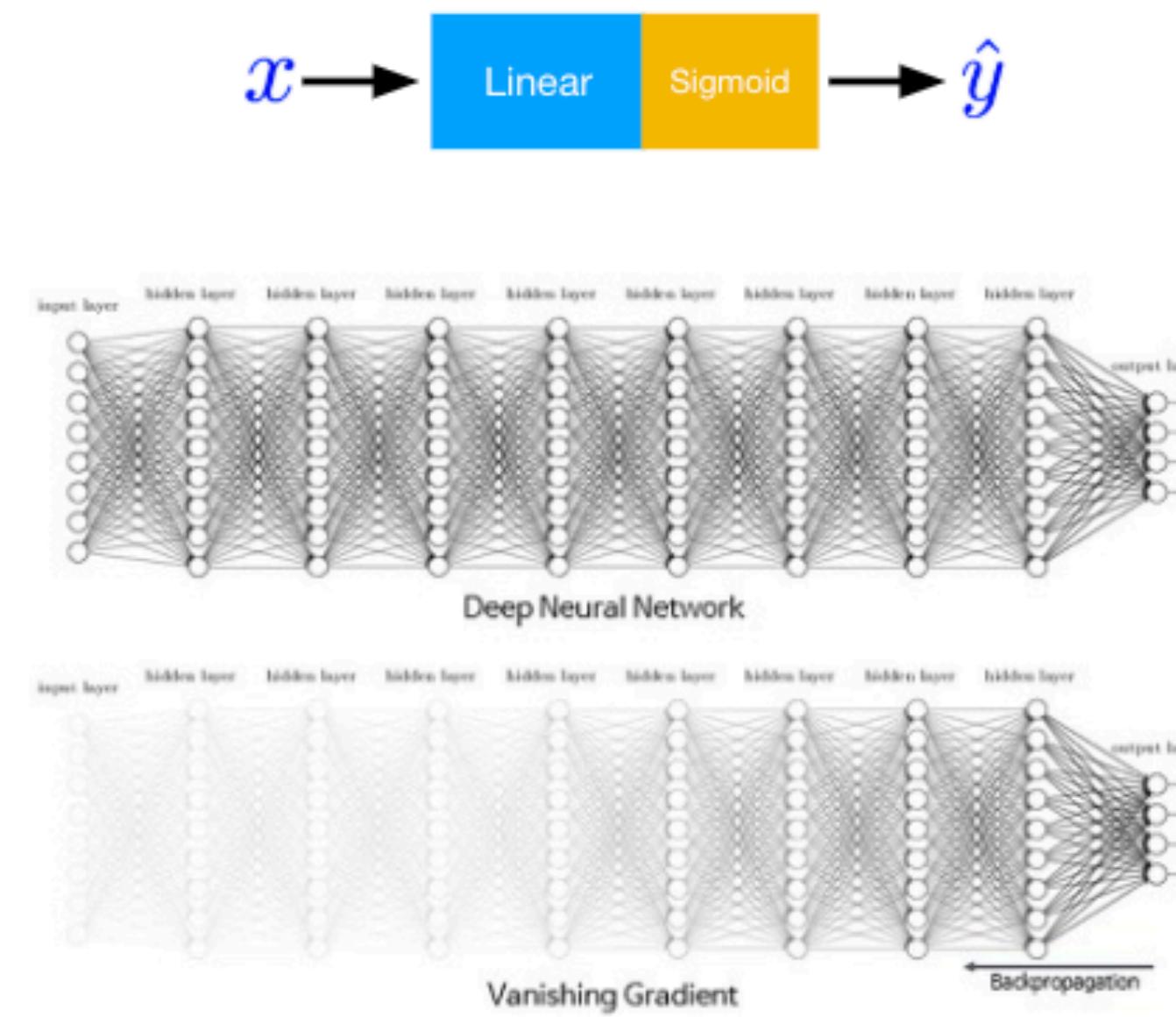
Conclusion:
More layers is better

A close-up shot of two men in dark suits and ties. They are looking directly at each other with serious expressions. The lighting is dramatic, casting shadows on their faces.

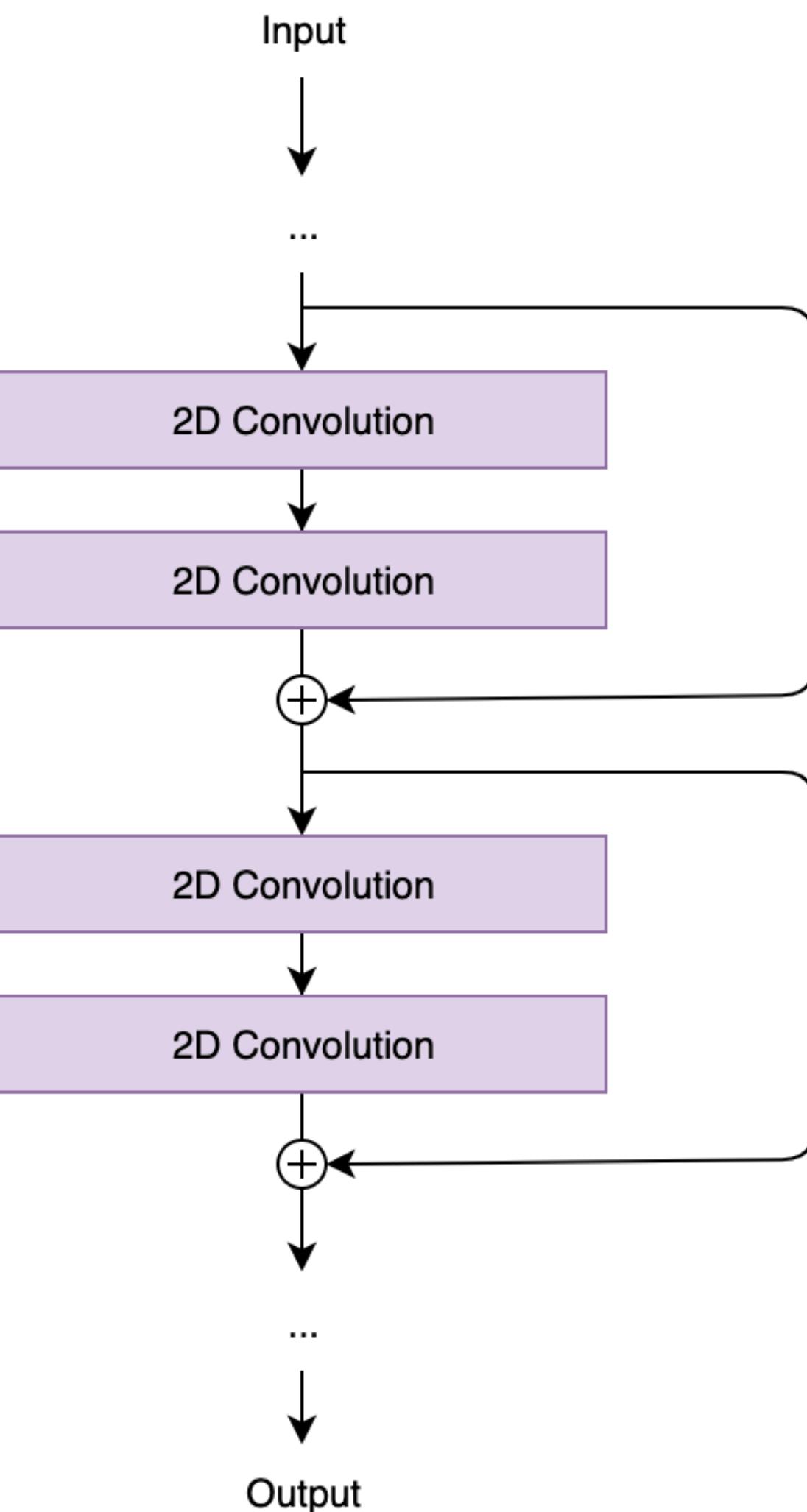
WE NEED TO GO

DEEPER

Not everything is that simple: Vanishing Gradient Problem

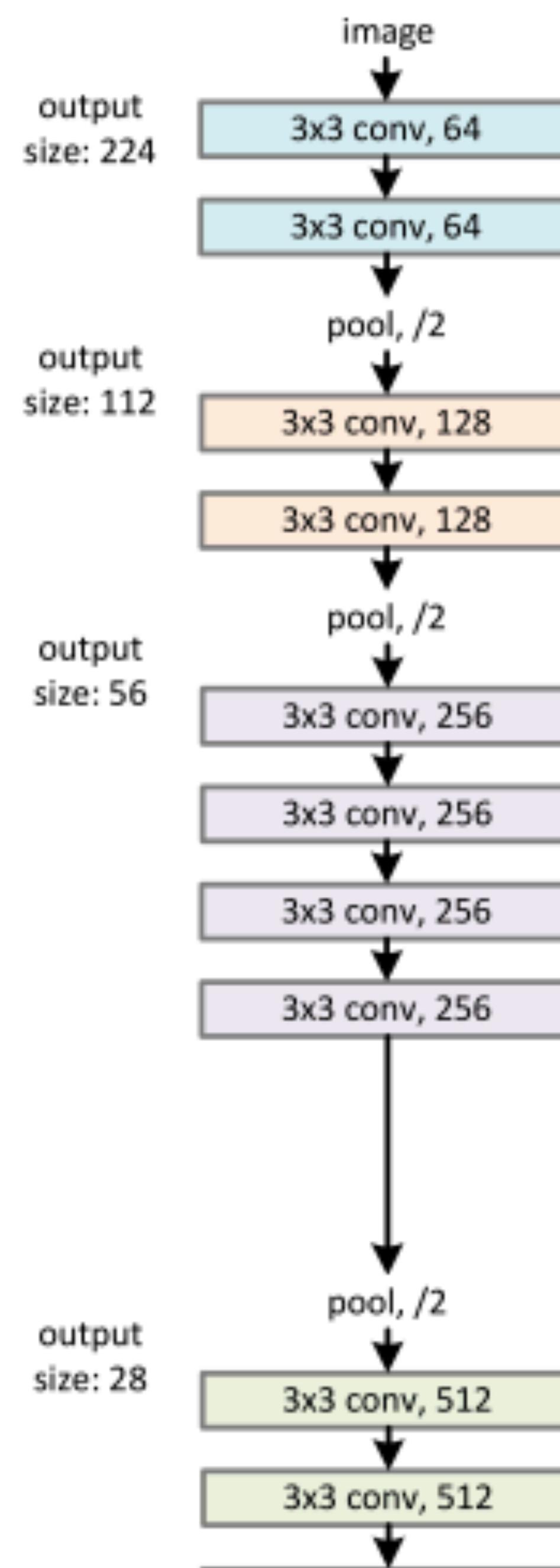


As the gradient keeps flowing backward to the initial layers, this value keeps getting multiplied by each local gradient. Hence, the gradient becomes smaller and smaller, making the updates to the initial layers very small, increasing the training time considerably.

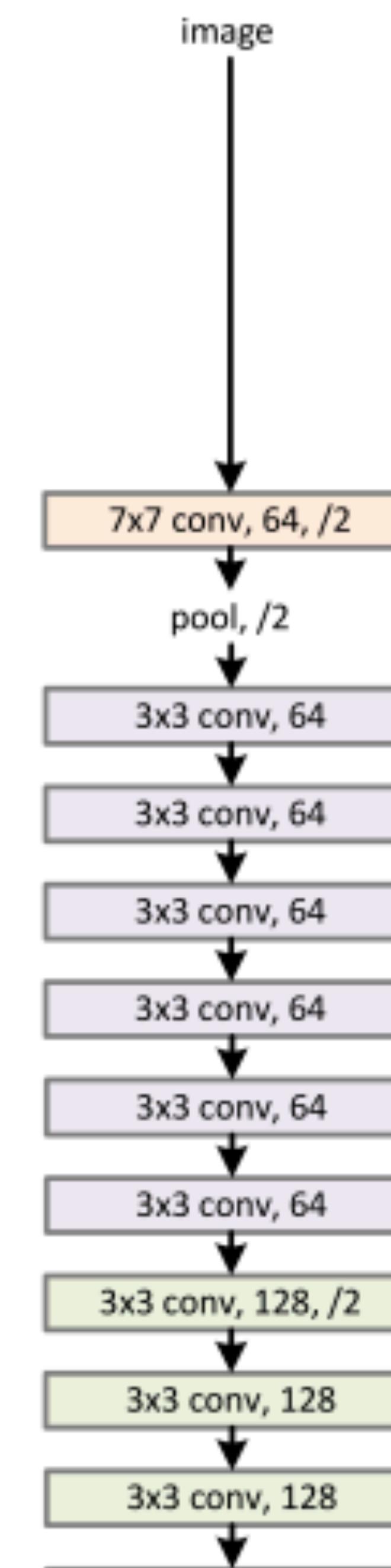


These ***skip connections*** act as gradient *superhighways*, allowing the gradient to flow unhindered.

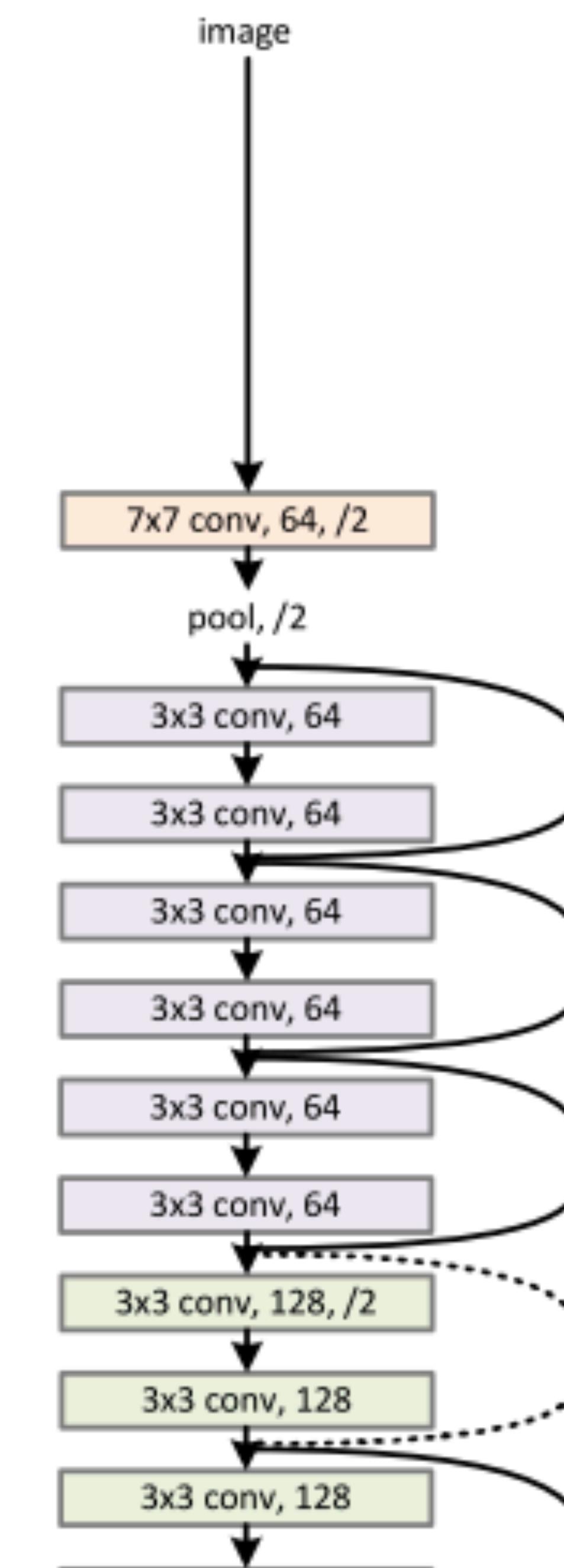
VGG-19



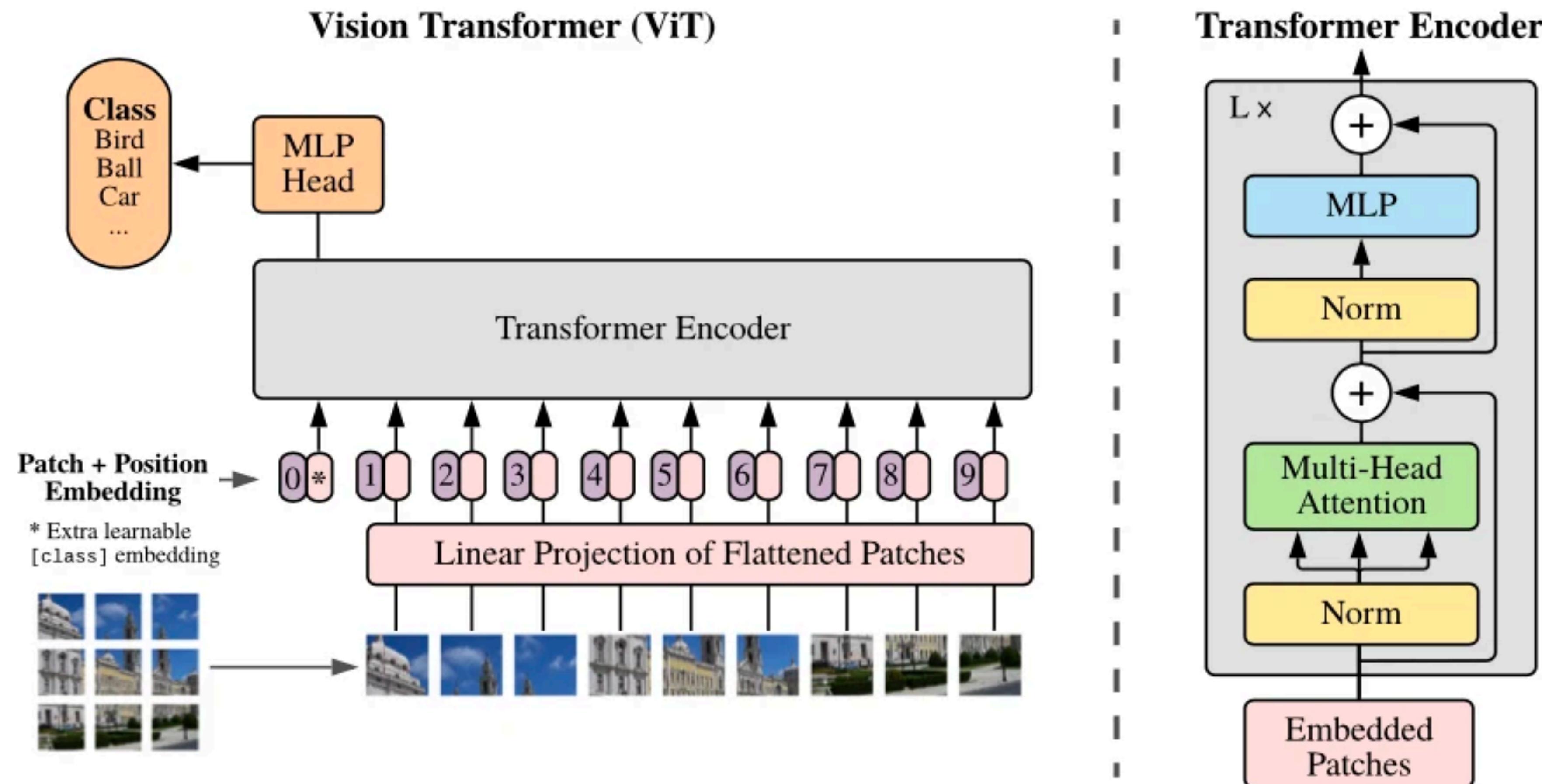
34-layer plain



34-layer residual



Visual TRANSFORMERS



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.

Unsupervised learning

Layout

- Autoencoders
- Learning unsupervised representations
- Sparse coding
- A manifold learning view
- Deep patient
- Self-Supervised Learning

Unsupervised learning tries to understand the properties of a particular set of data. There are different ways of doing this

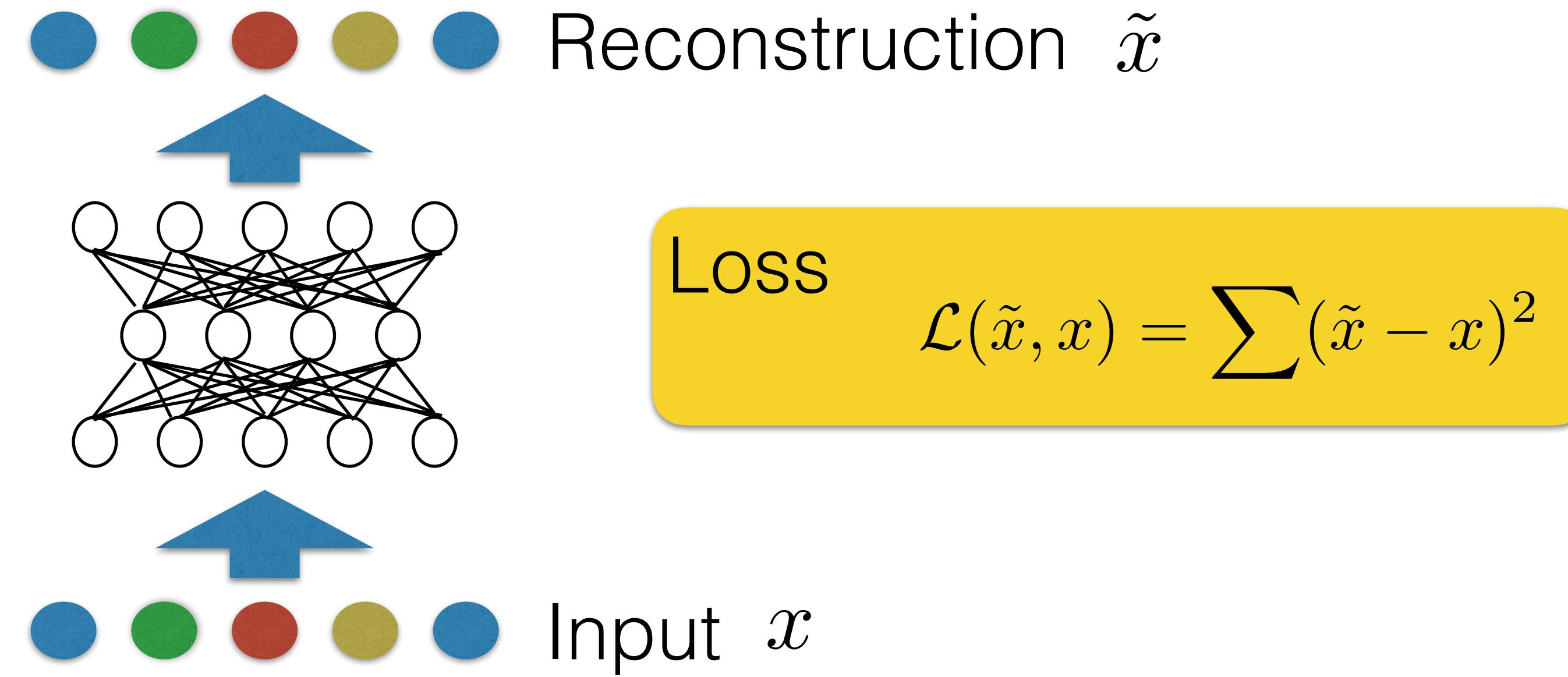
- Clustering - Divide data in groups according to some notion of similarity.
- Manifold learning - Understanding how data is distributed in the space, parameterising a manifold.

Autoencoder

- Build a network with the aim of reconstruction.

D.E. Rumelhart, G.E. Hinton, and R.J. Williams. Learning internal representations by error propagation. In Parallel Distributed Processing. Vol 1: Foundations. MIT Press, Cambridge, MA, 1986.

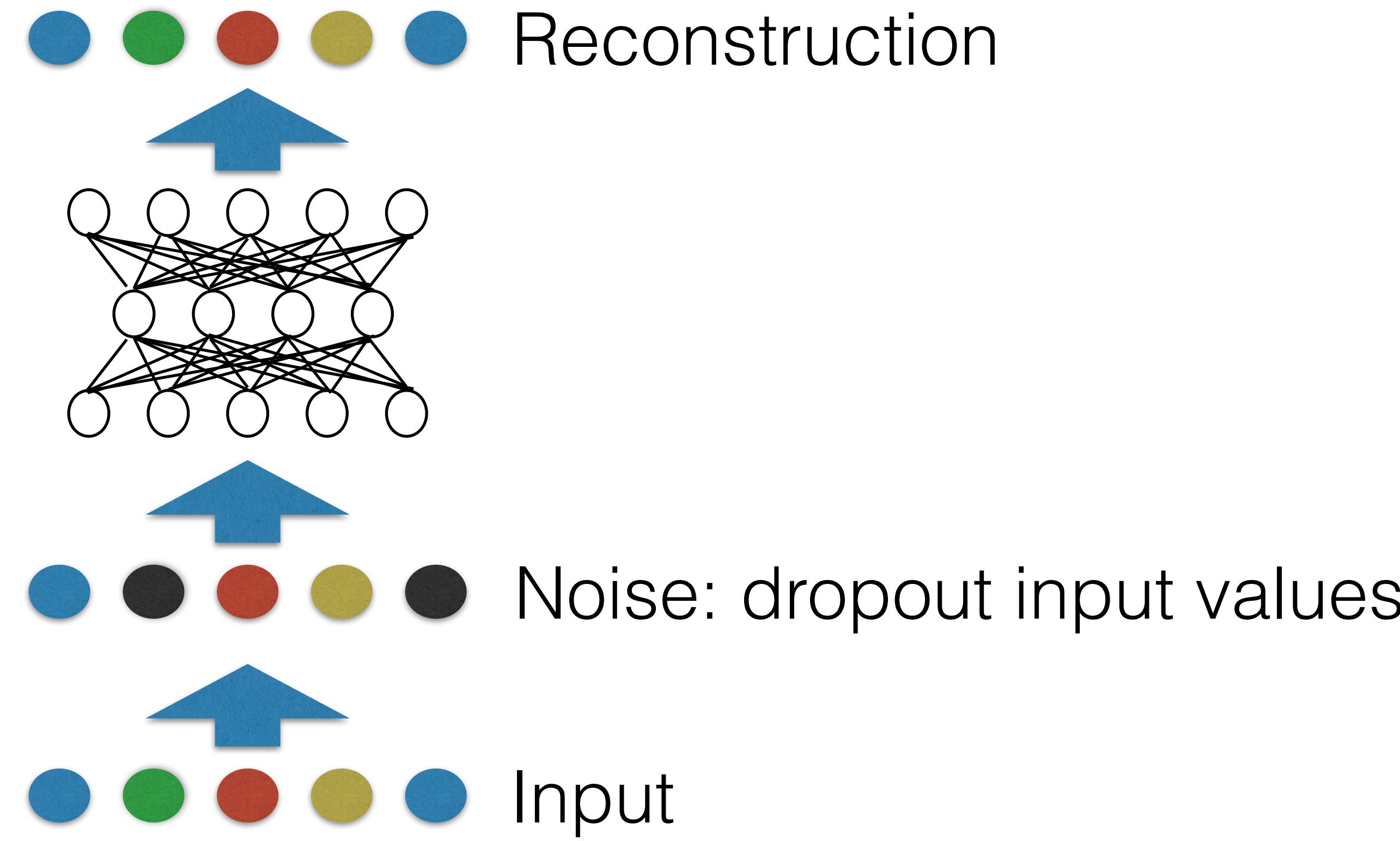
Autoencoders



Problems

- In large networks it may learn the identity mapping rendering the auto encoder representation useless.
- In order to correct this issue and furthermore give robustness to the auto encoder, denoising auto encoders are proposed.

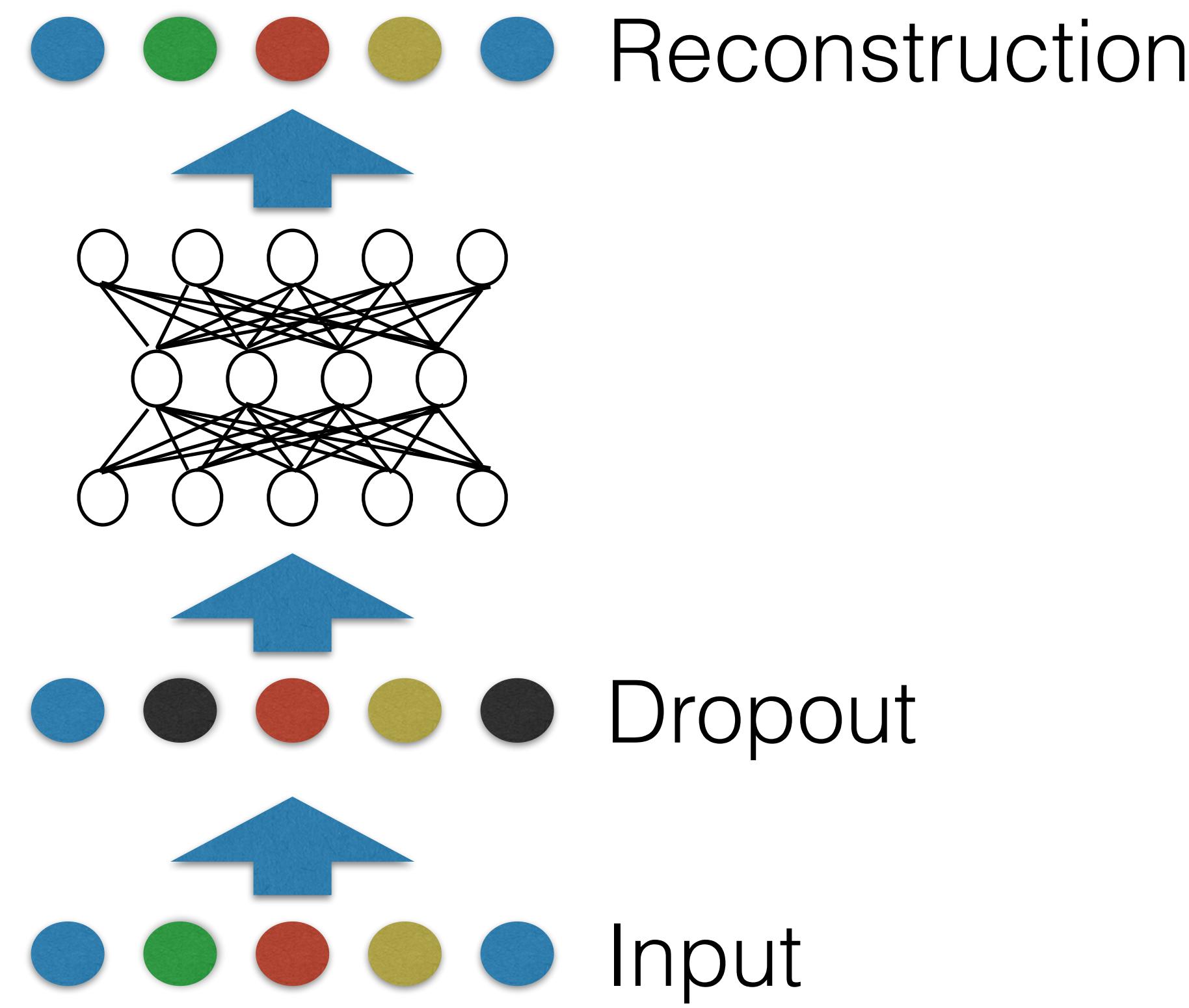
Denoising Autoencoders



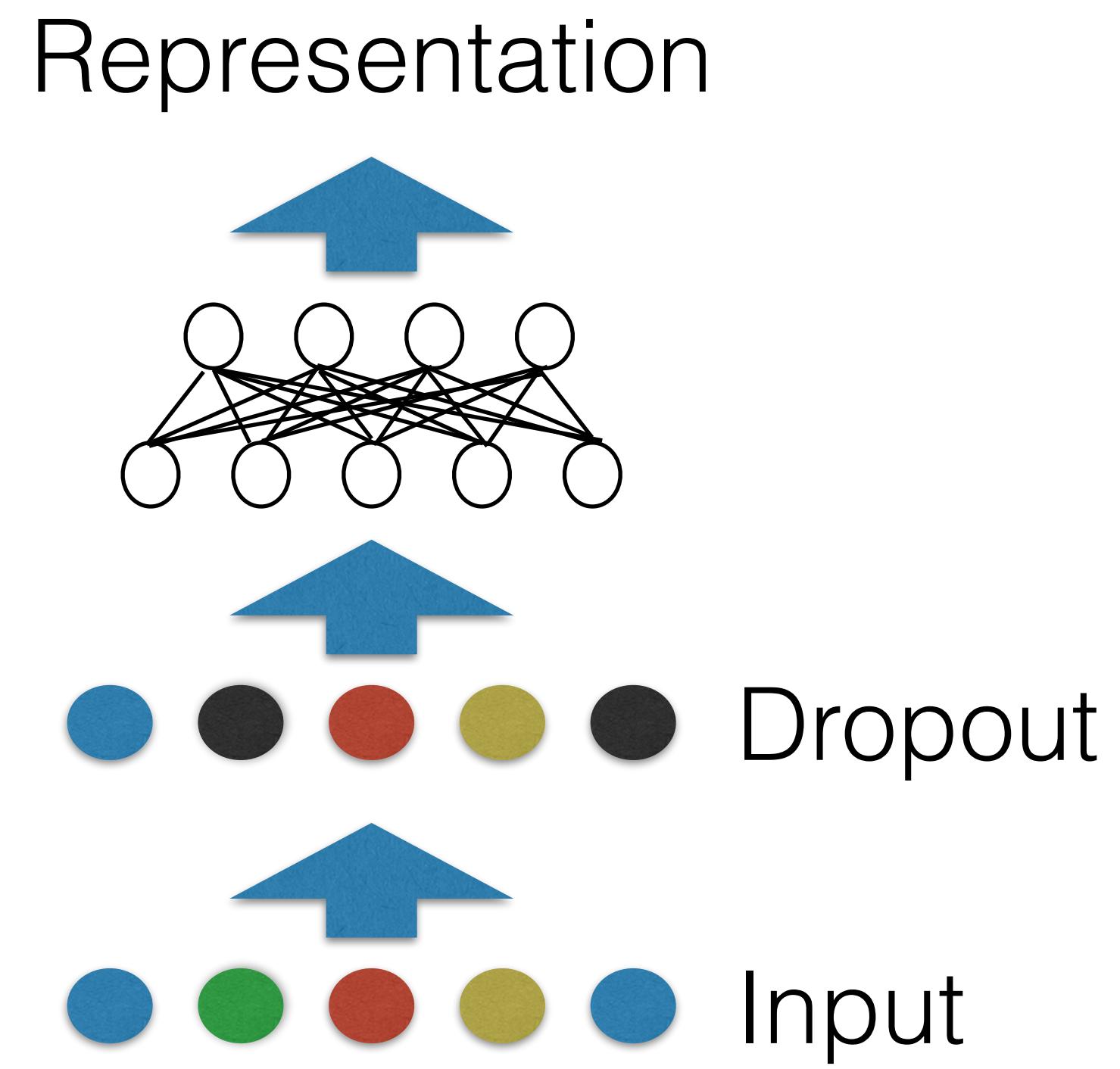
Learning representations

- What can we use these representations for?
 - Transfer learning
 - Pure transfer
 - Pretraining
 - Compression

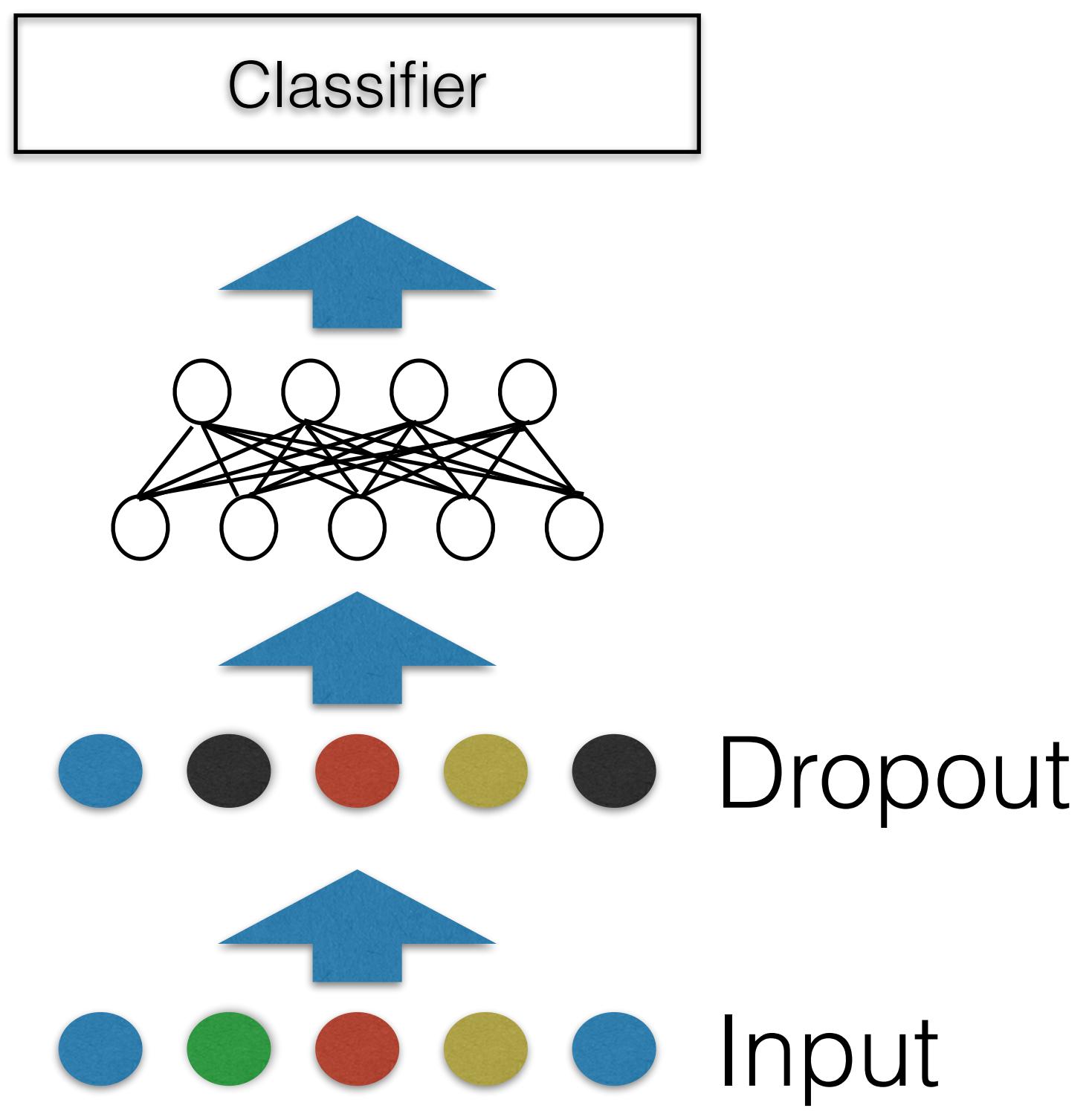
Pretraining and transfer



Pretraining and transfer

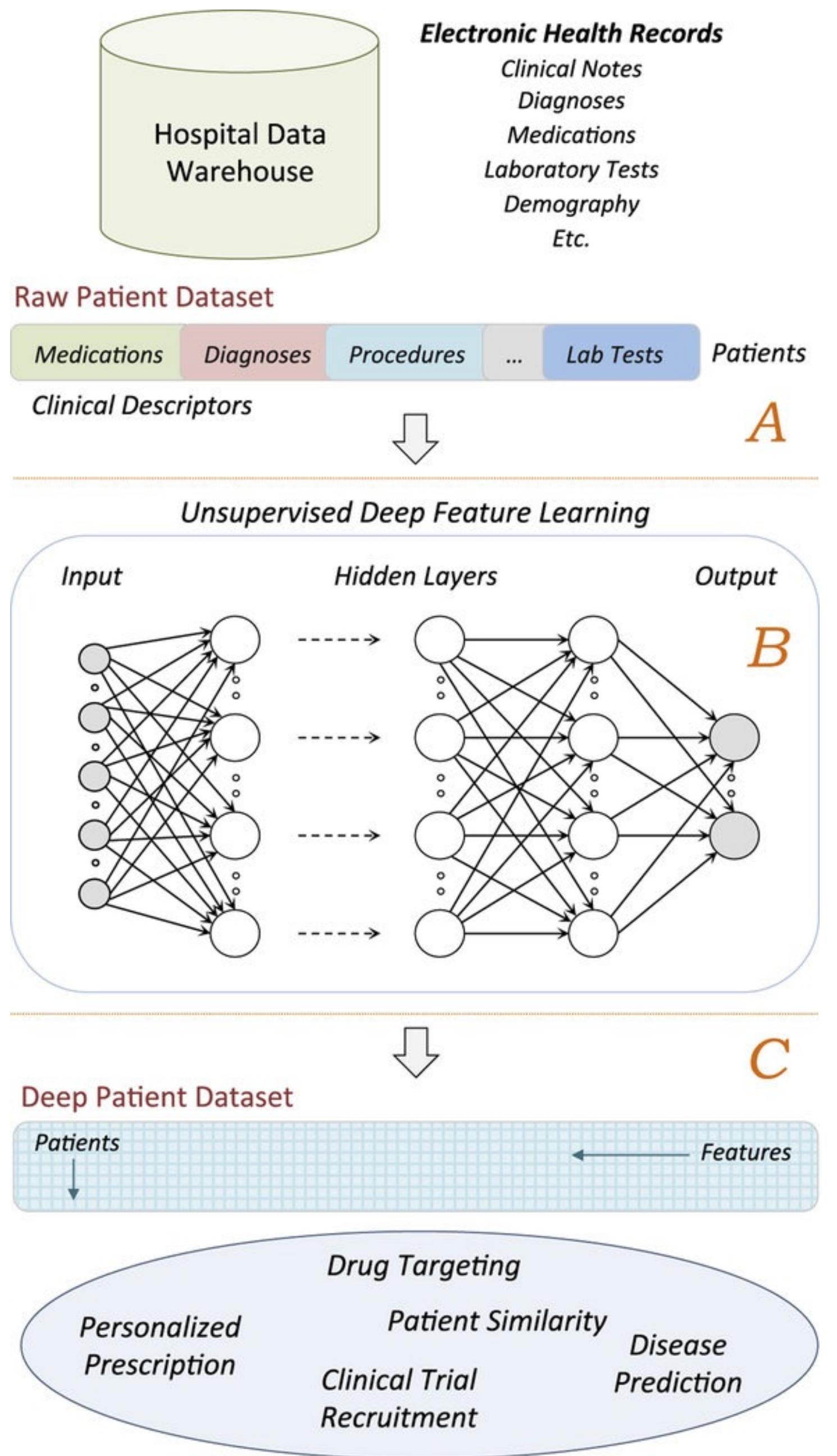


Pretraining and transfer

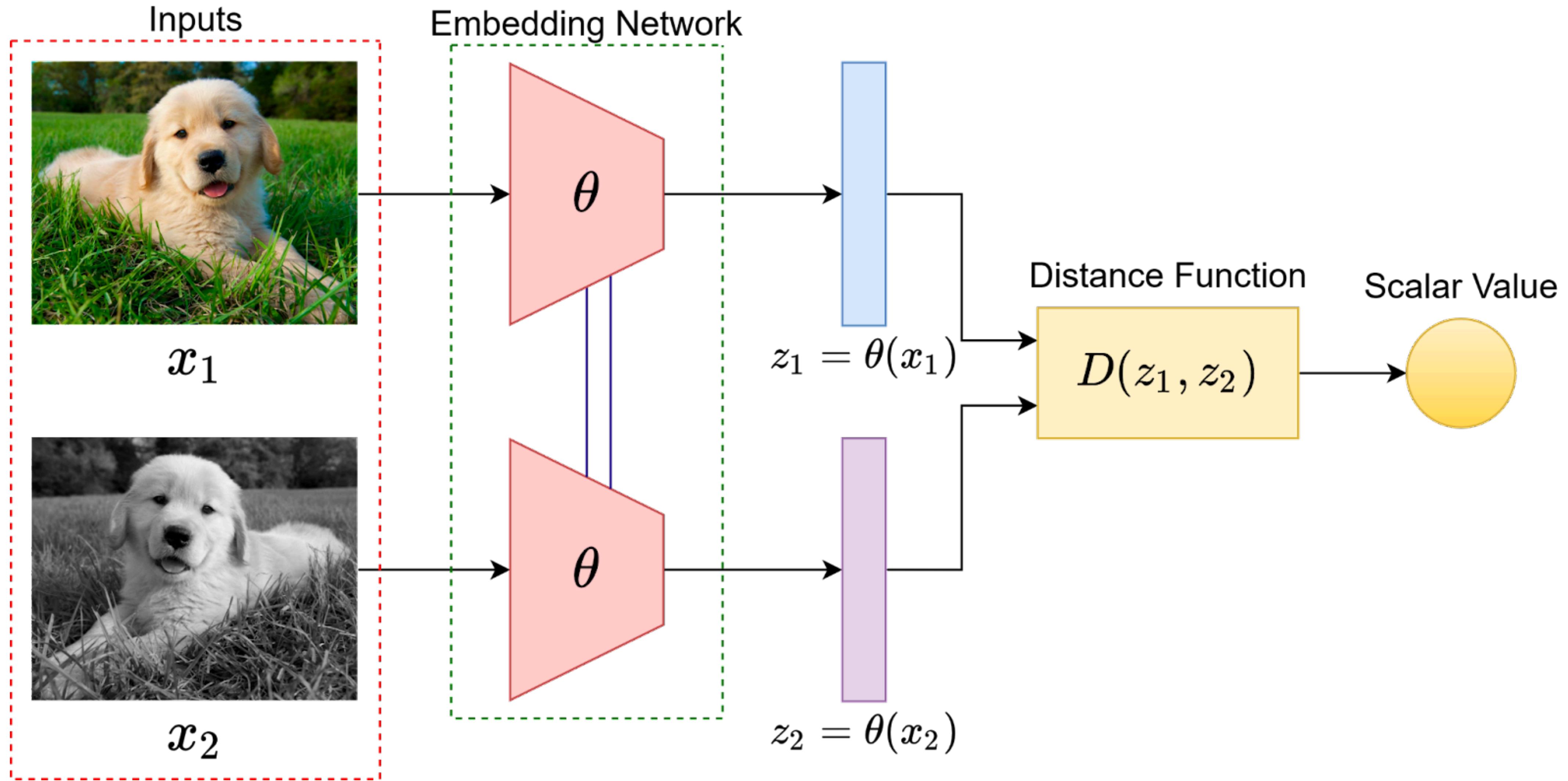


Application

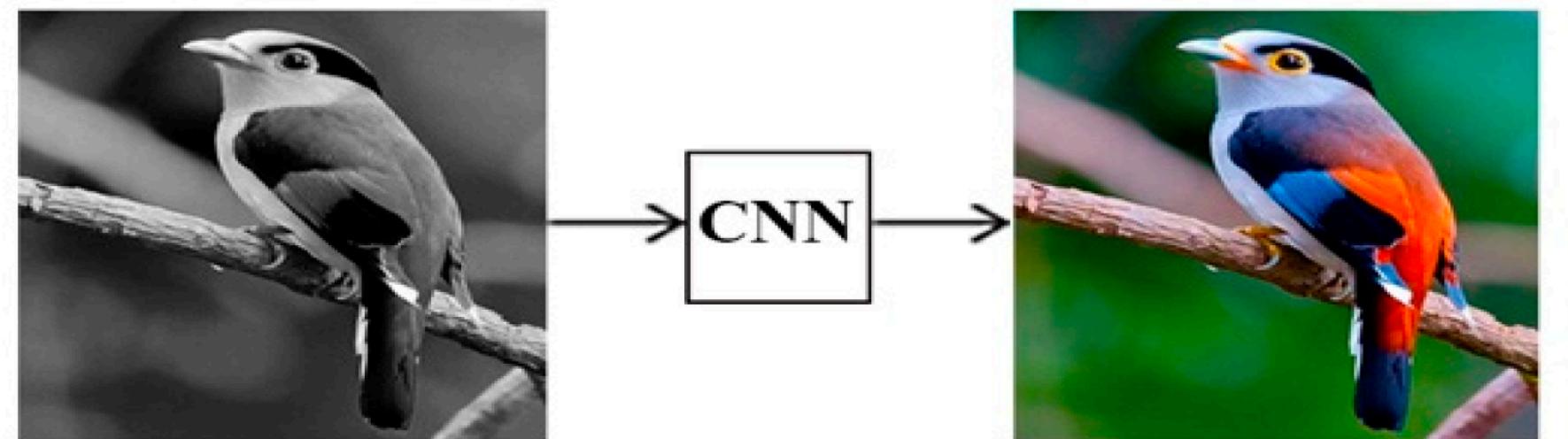
- Deep Patient: An Unsupervised Representation to Predict the Future of Patients from the Electronic Health Records



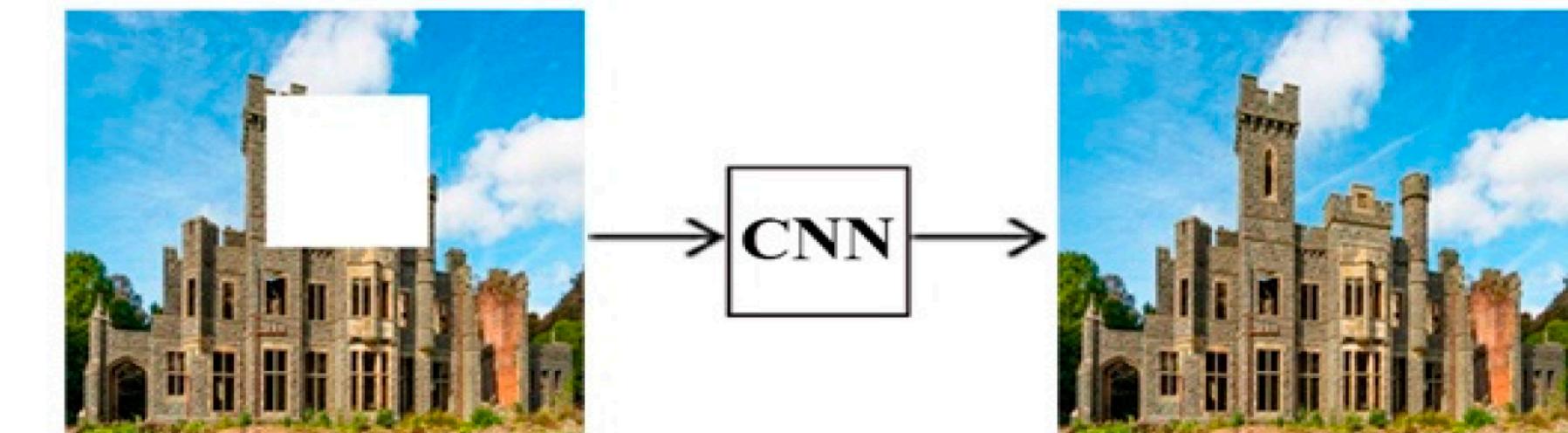
Self-Supervised Learning



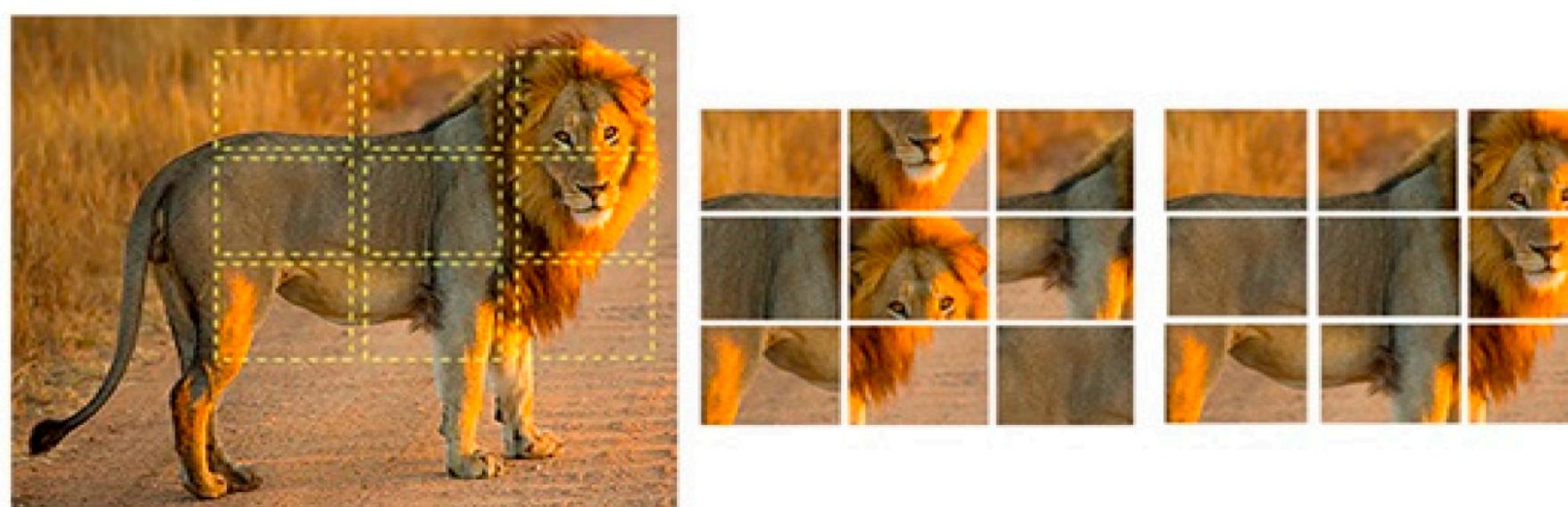
Self-Supervised Learning



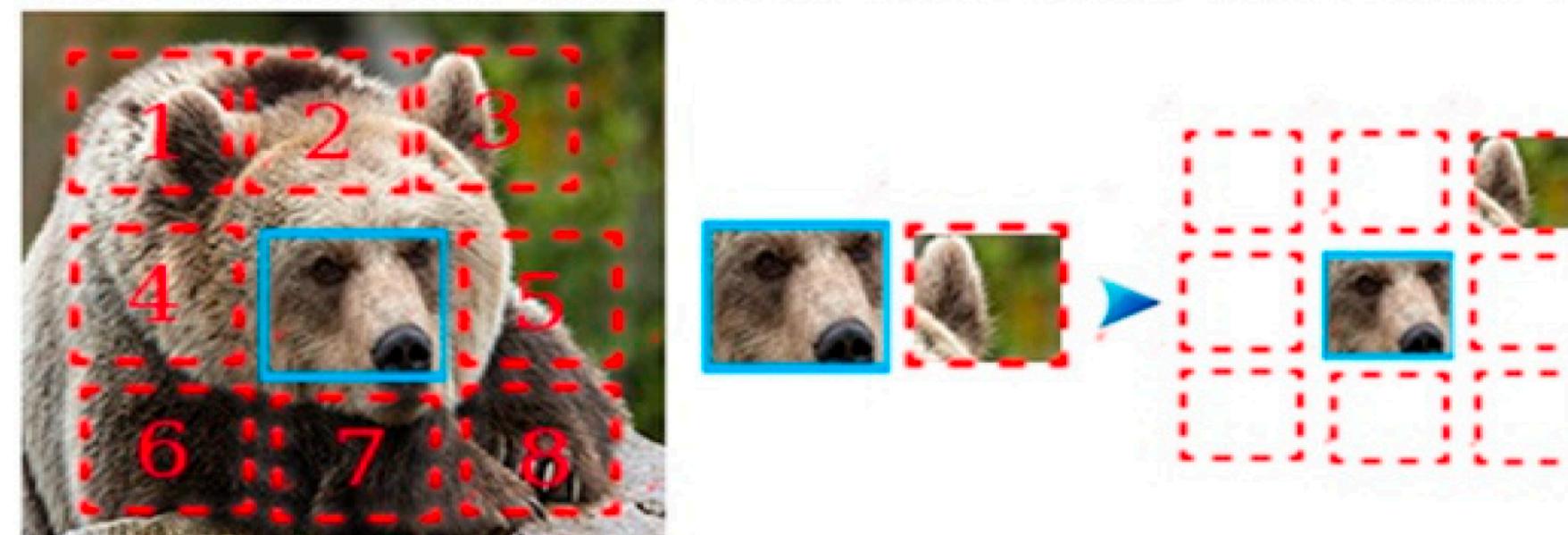
a) Colorizing an image



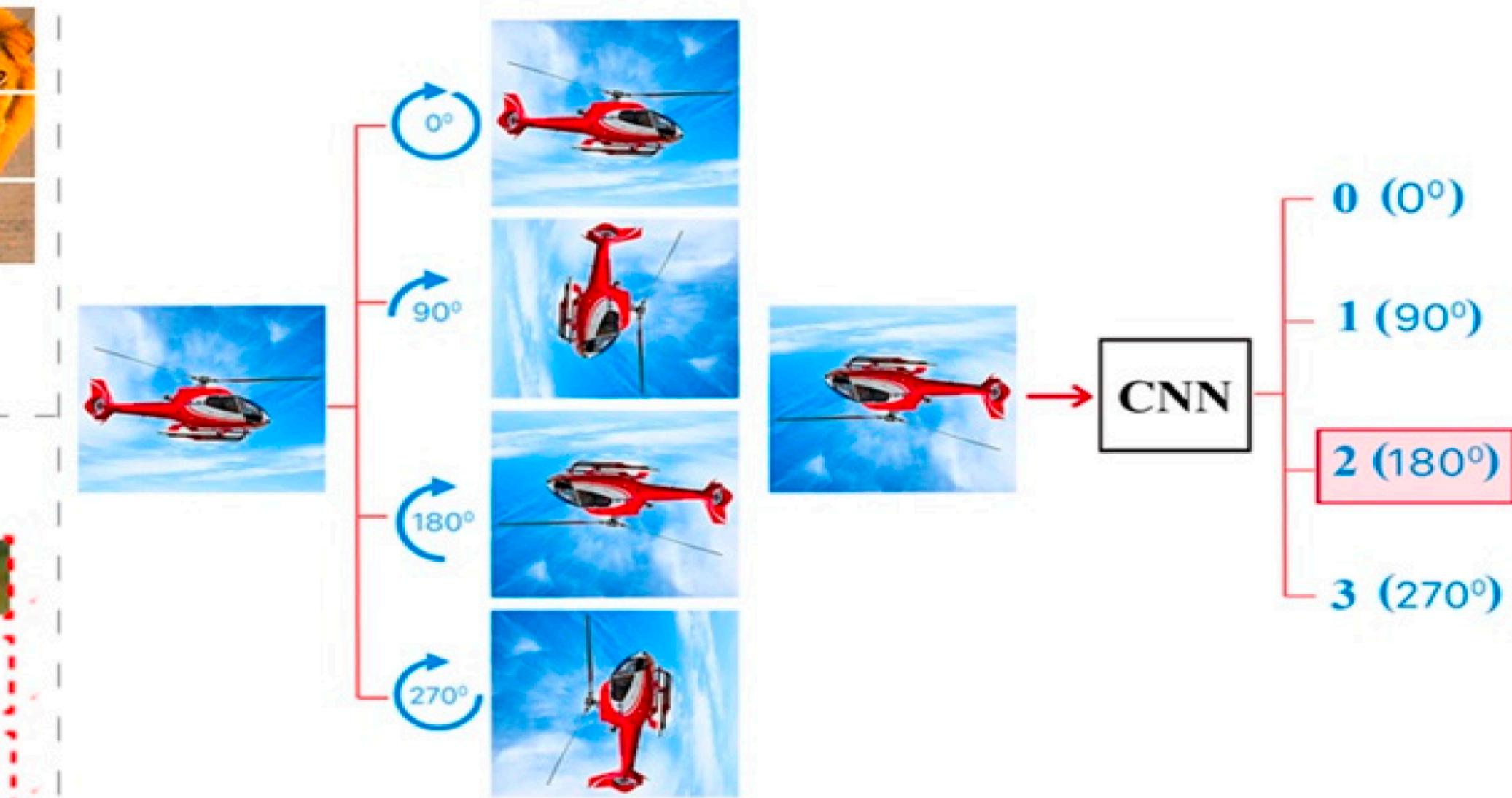
b) Inpainting



c) Solving Jigsaw Puzzle



d) Predicting relative position



e) Estimating the rotation angle