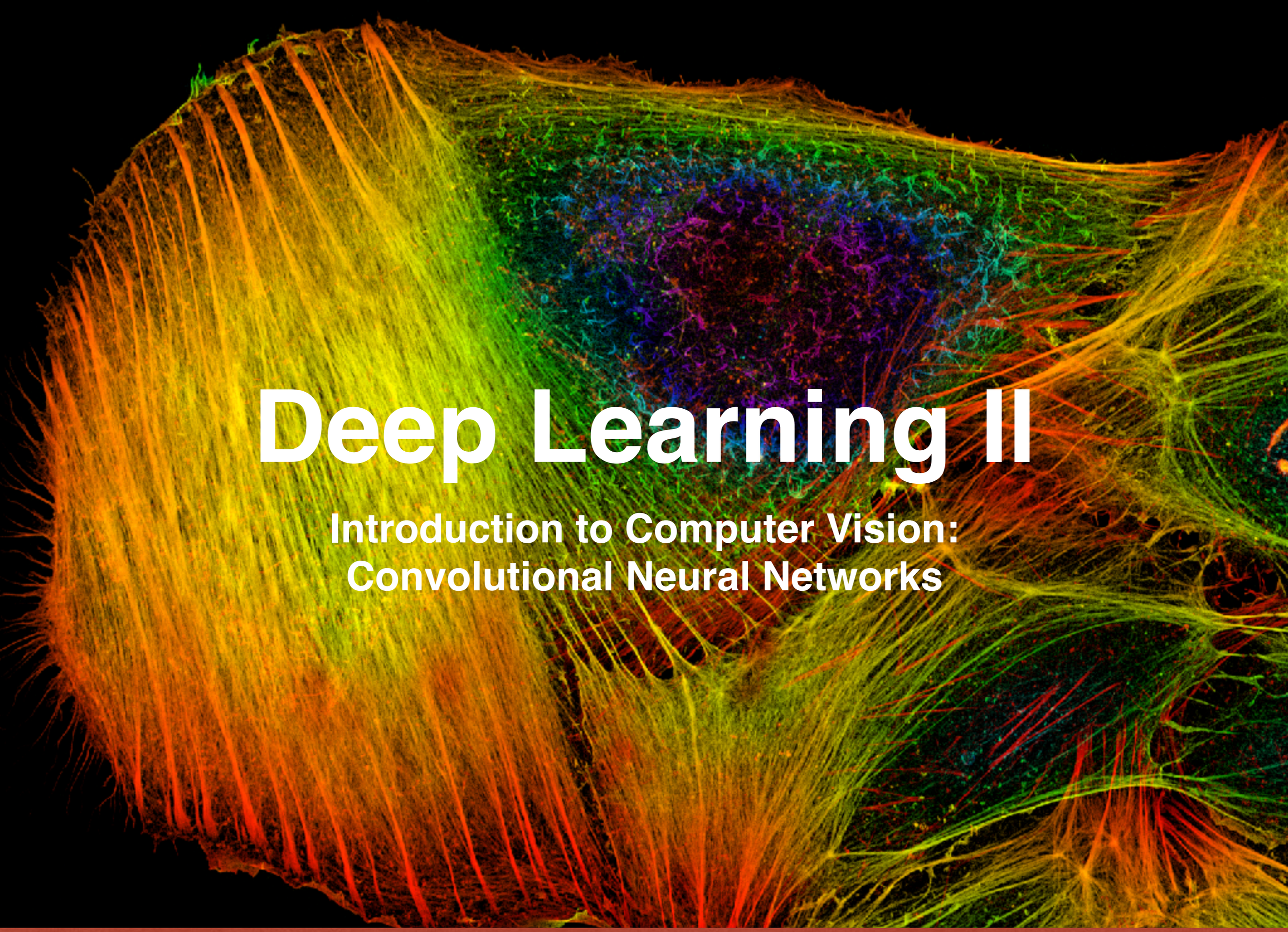
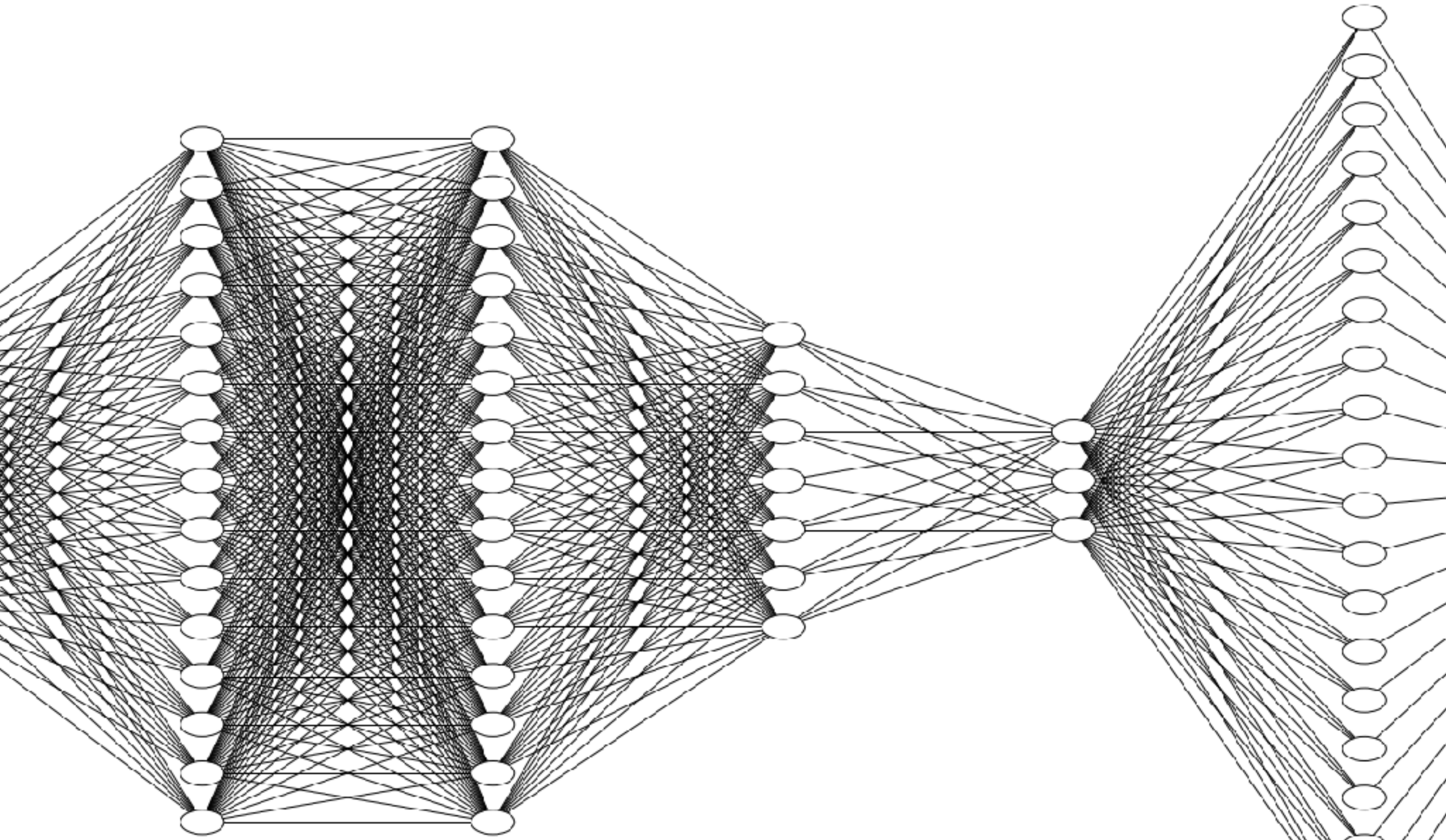


Deep Learning II

Introduction to Computer Vision: Convolutional Neural Networks

Yesterday, we managed to **teach** a fully connected neural network how to recognize handwritten digits



Problems

Fully connected models aren't easy to train when they scale. Problems like vanishing gradient arise

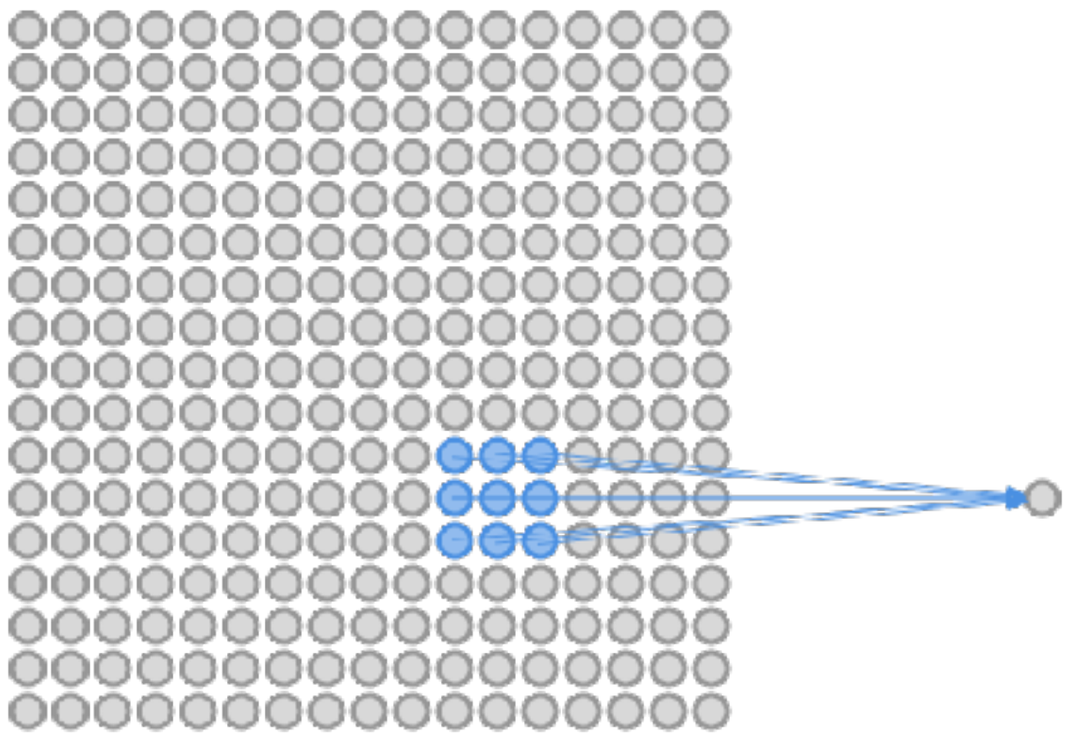
- Does **going deeper** help?
- Does it even *make sense* to use a fully connected network on input images?

We'd rather design an architecture which takes advantage of the **spatial structure** of the input

We'd also like our model to be **lighter** and **more scalable**

Convolutional Nets

- The input neurons can now be arranged according to the pixels position

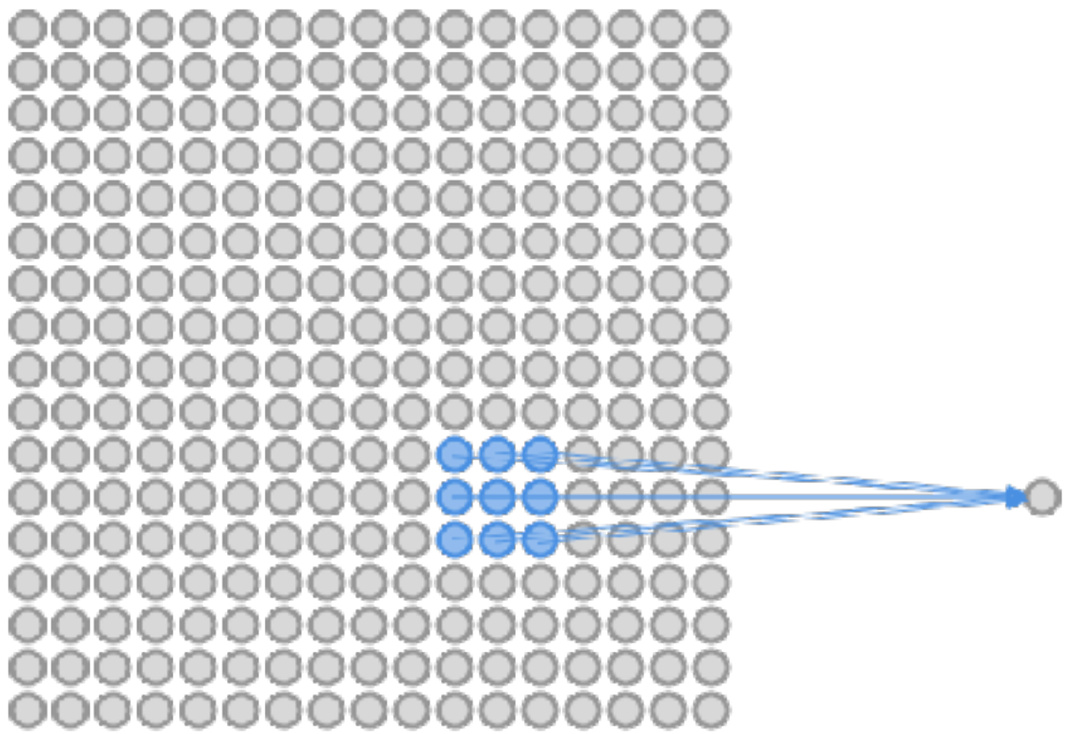


- We won't connect each input pixel to every hidden neuron, we'd rather make localized connections

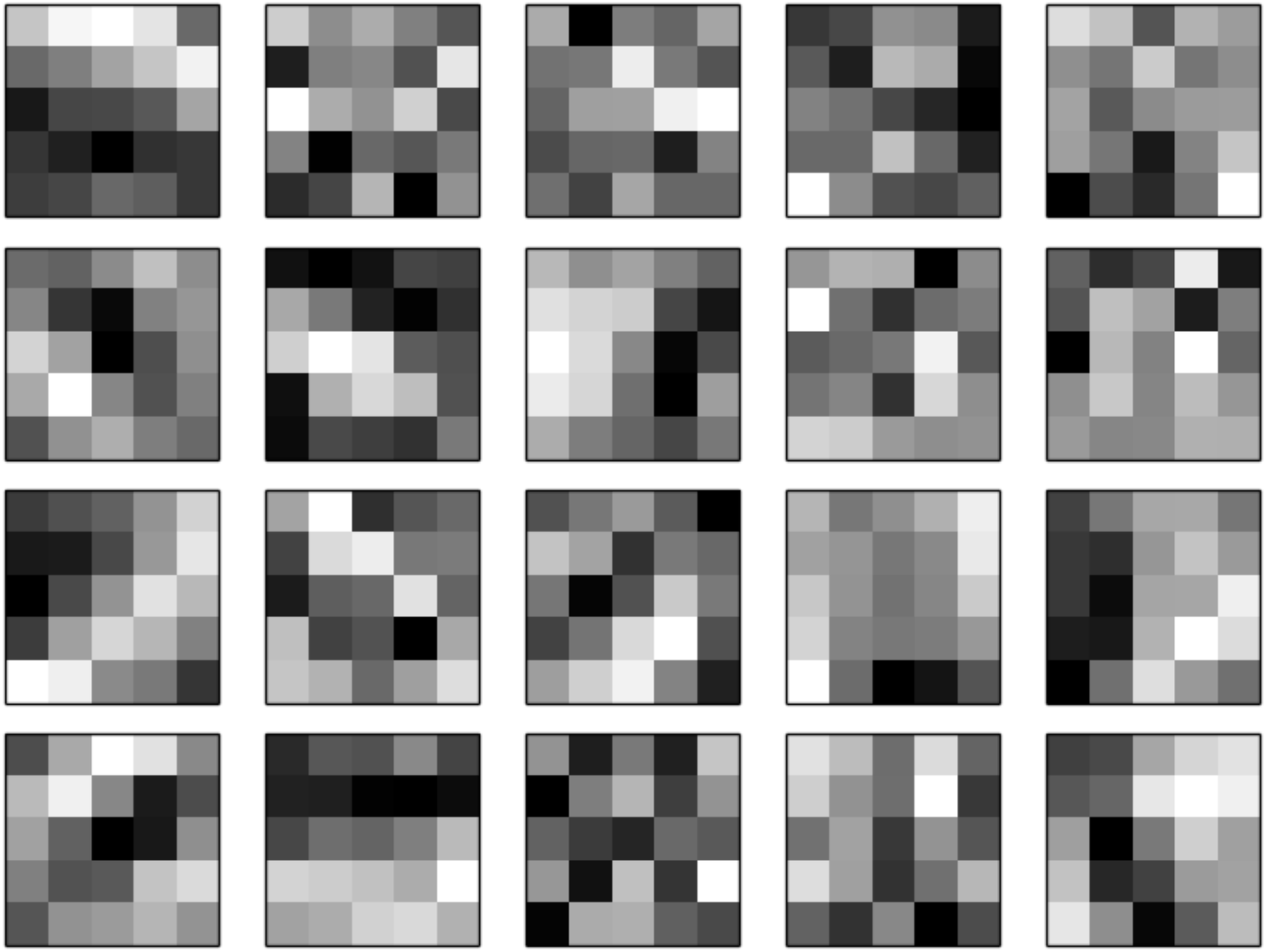
- The input region that maps to the hidden neuron is called **local receptive field** of the hidden neuron

Convolutional Nets

- Receptive fields slide through the image until they cover it all
- The step size of this slide is called **stride**
- The map from the input layer to the hidden layer is called **feature map**
- Multiple feature maps share weights (called **kernels or filters**) and biases



Filters, kernels, shared weights



We can think of a convolution as the complete **mapping** between the input space and the hidden space

The term *convolution* is used because of the mathematical expression of this mapping

$$\sigma\left(b + \sum_{l=0}^{f} \sum_{m=0}^{f} w_{l,m}\, a_{j+l,k+m}\right)$$

Convolutional neural networks also contain **pooling layers**, usually immediately after convolutional layers

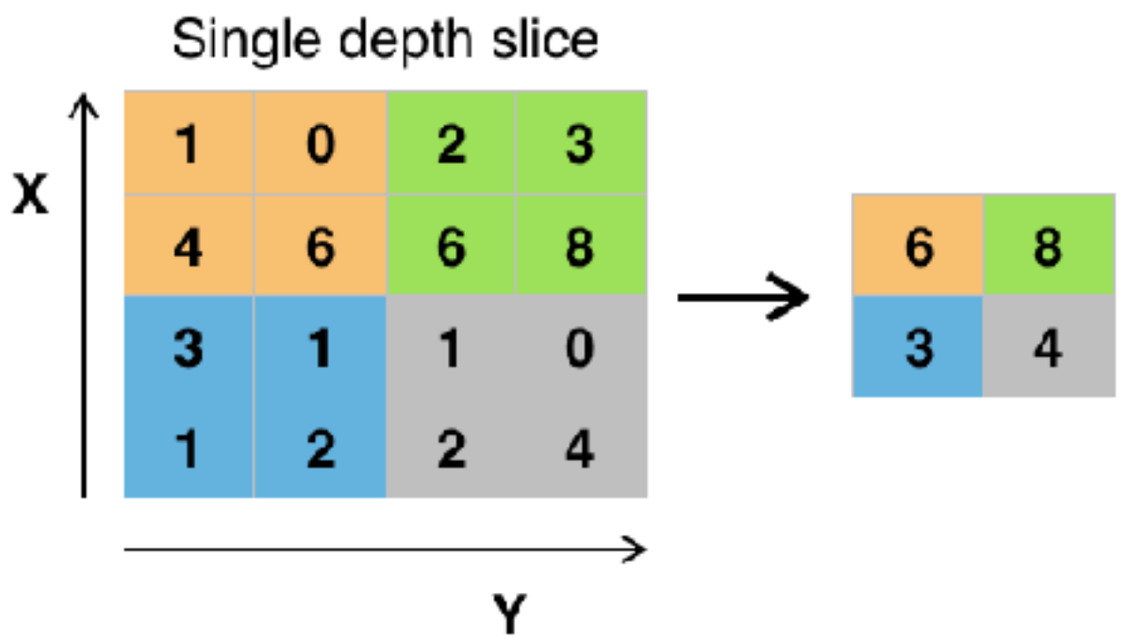
Pooling layers map *feature map outputs* to a ***condensed feature map***, depending on a particular pooling function

Pooling

- **Max-pooling** extracts the maximum activation among a pooling region

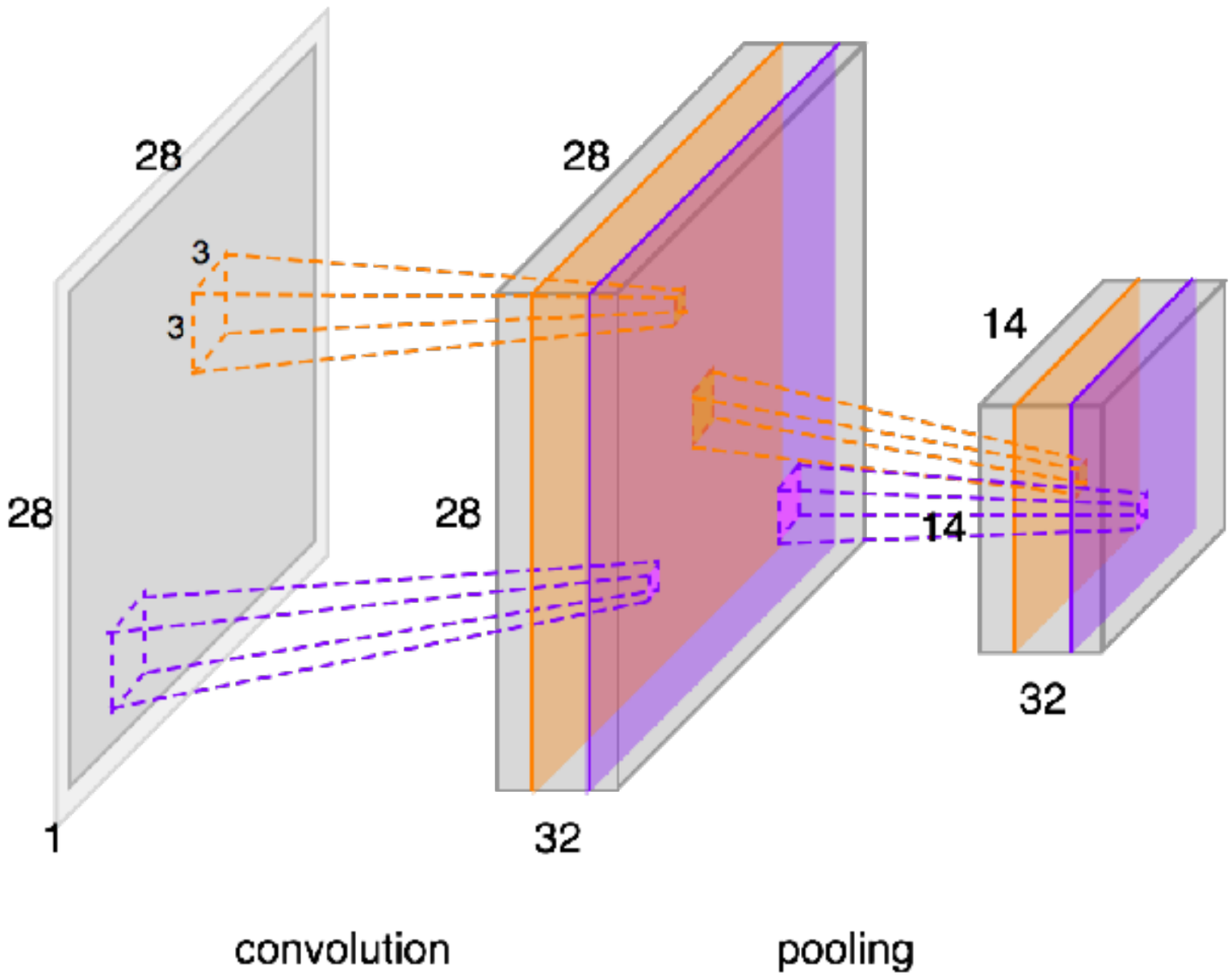
- **Average-pooling** calculates the average activation among a pooling region

- More complex pooling layers perform different operations. Any kind of (reasonable) pooling layer can be designed



All pooling layers share one characteristic: they **lose spatial information** while **reducing the parameters** in the network

Putting it all together

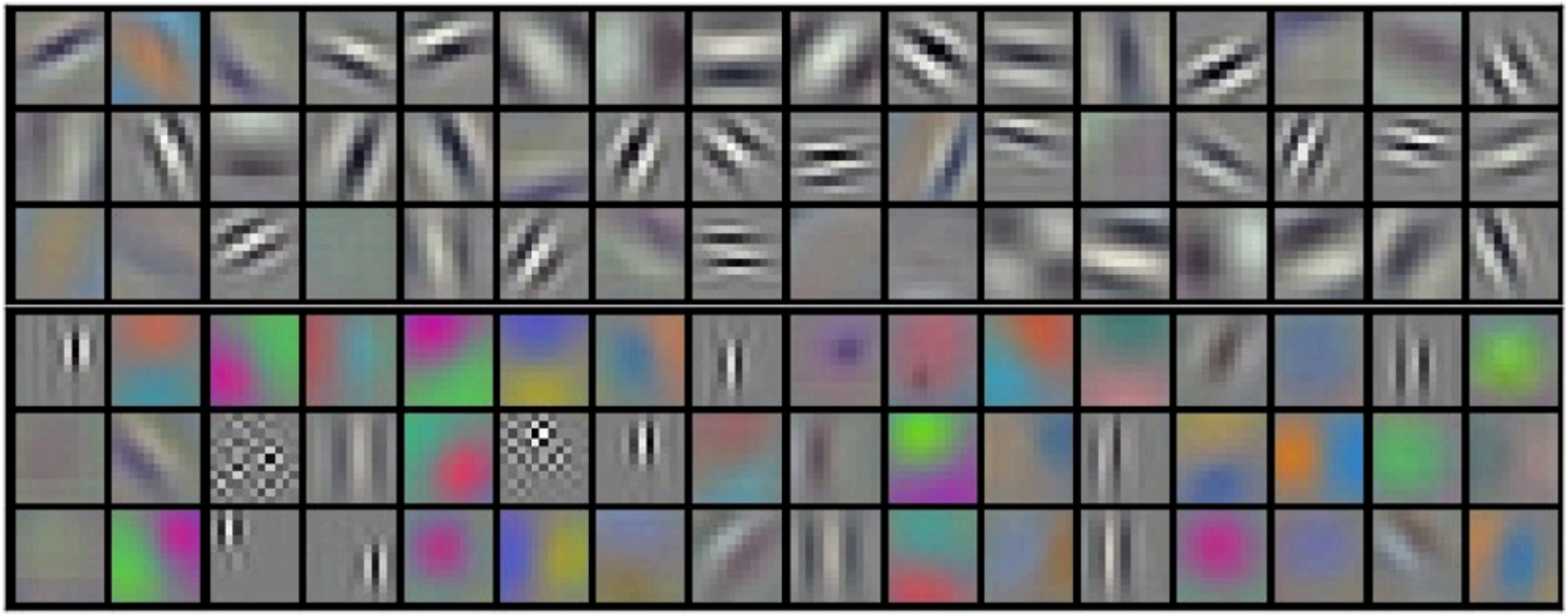


We can still use fully connected layers, dropout, softmax activations within CNNs

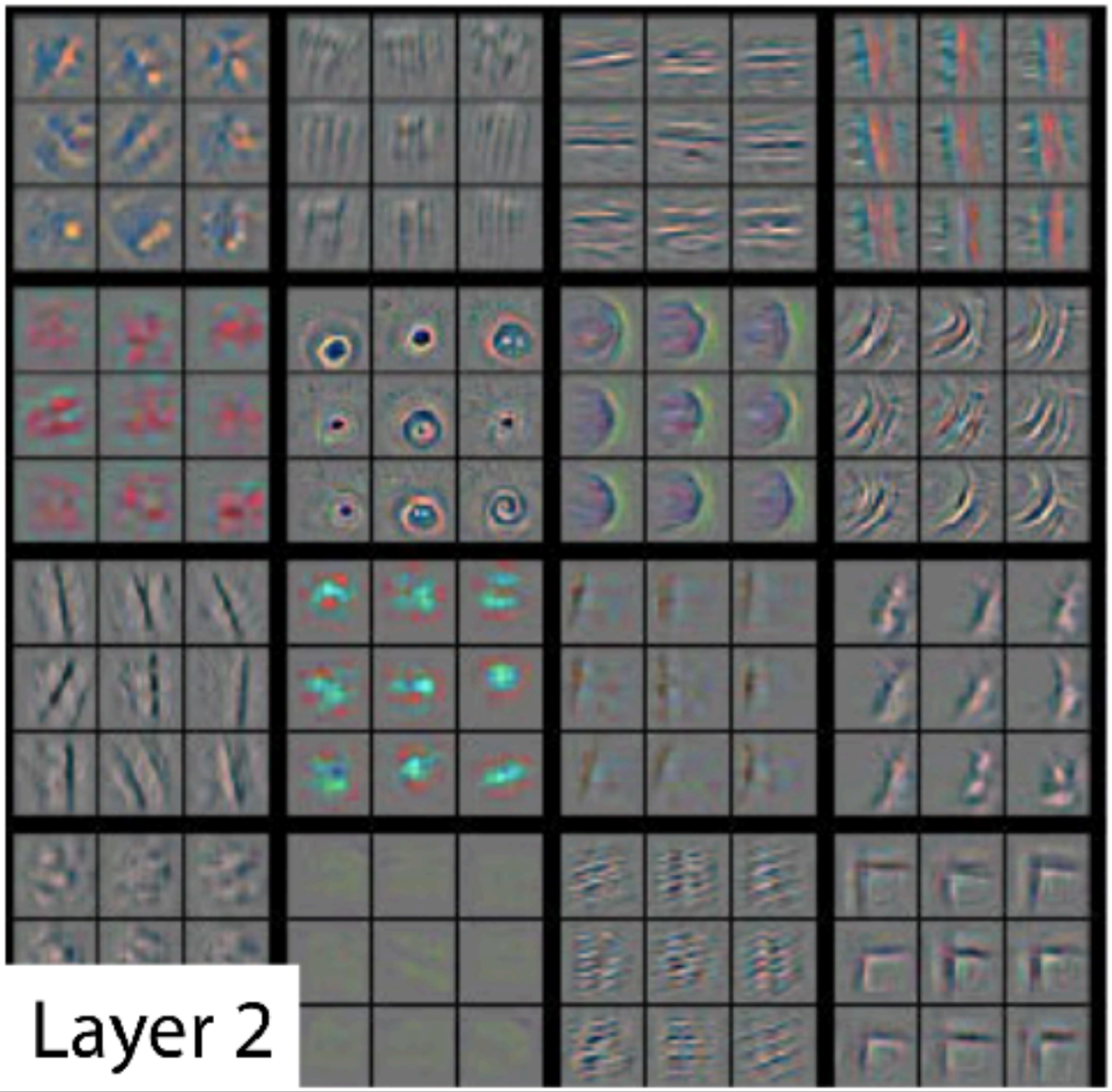
You still train a convolutional neural network in the same way we trained fully connected ones:

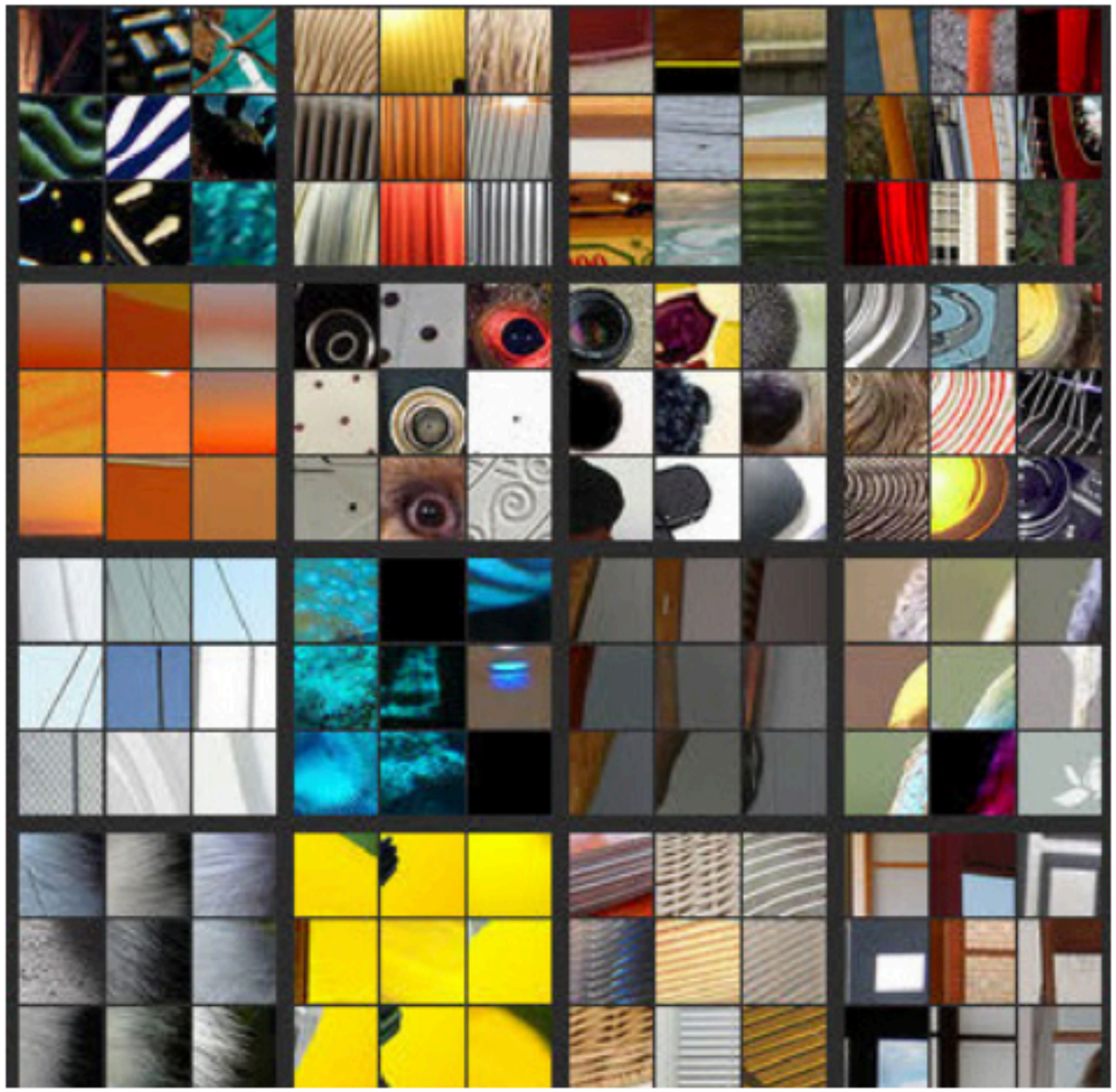
backpropagation

Visualize learning



Visualize learning

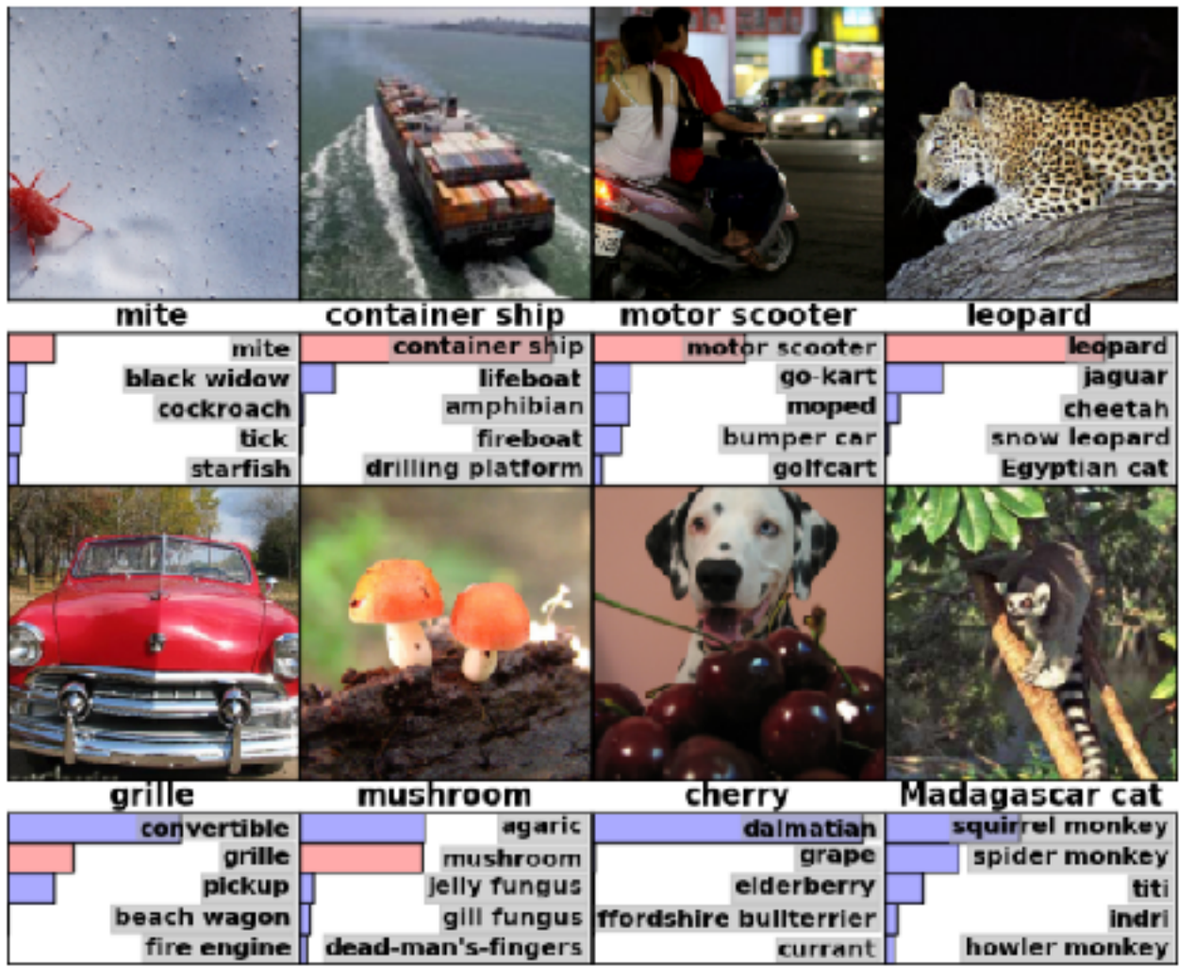




Krizhevsky et al. (2012)

Alexnet is able to give the right answer 63% of the time

One of the top 5 answers it gives is right 85% of the time

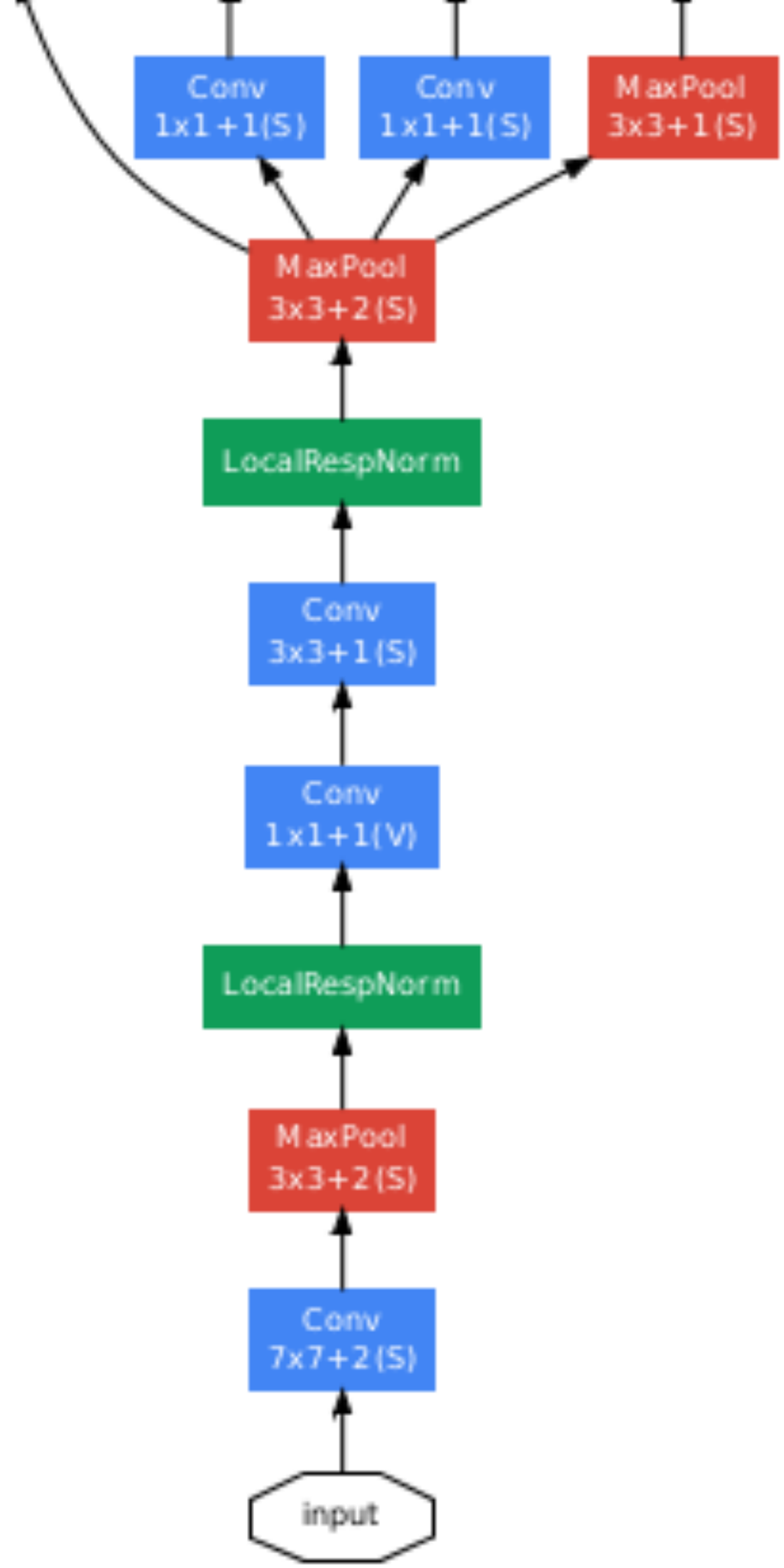


Why 2012?

The origins of convolutional neural networks go back to the 70s, and developed as we know them today during the late 90s. So *why* did we need to wait until 2012 to see their potential?

- Large datasets (e.g. ImageNet)
- Cheap parallel computation (e.g. GPUs)

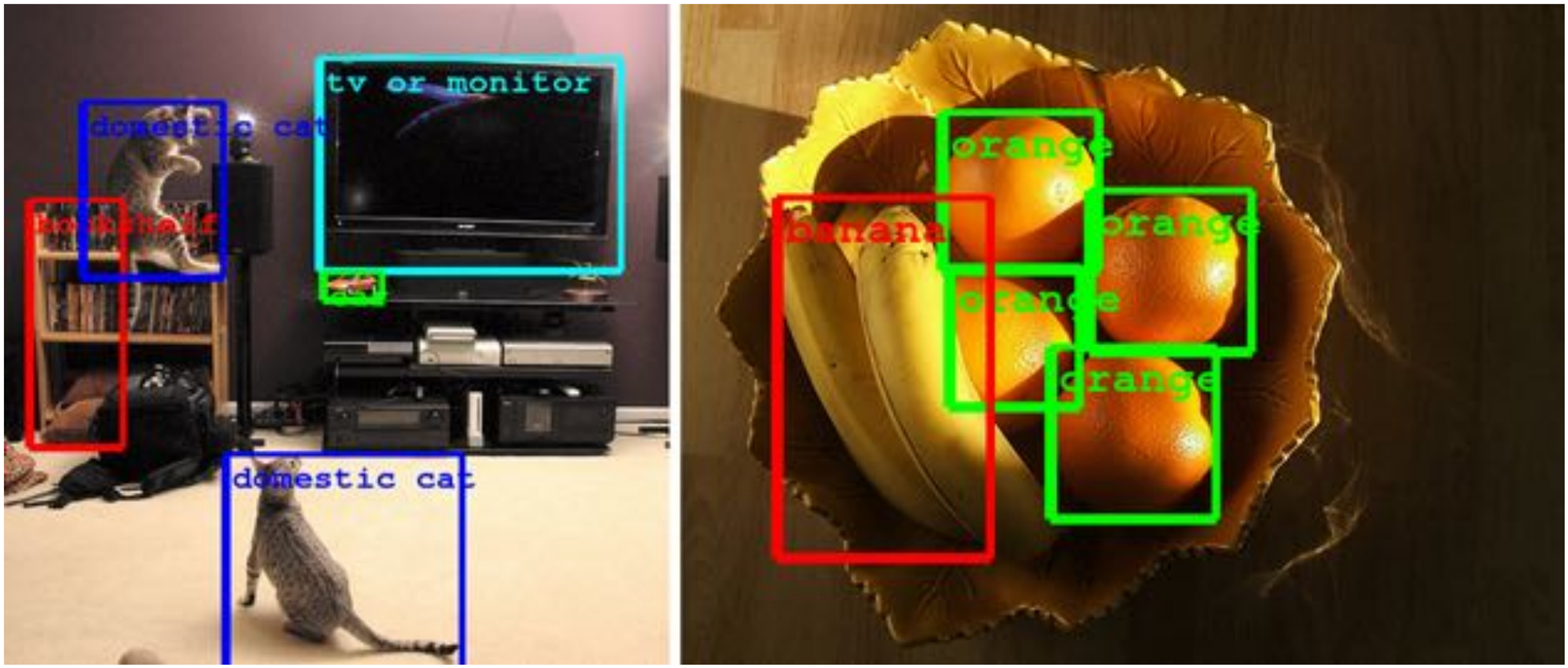
GoogLeNet (2014)



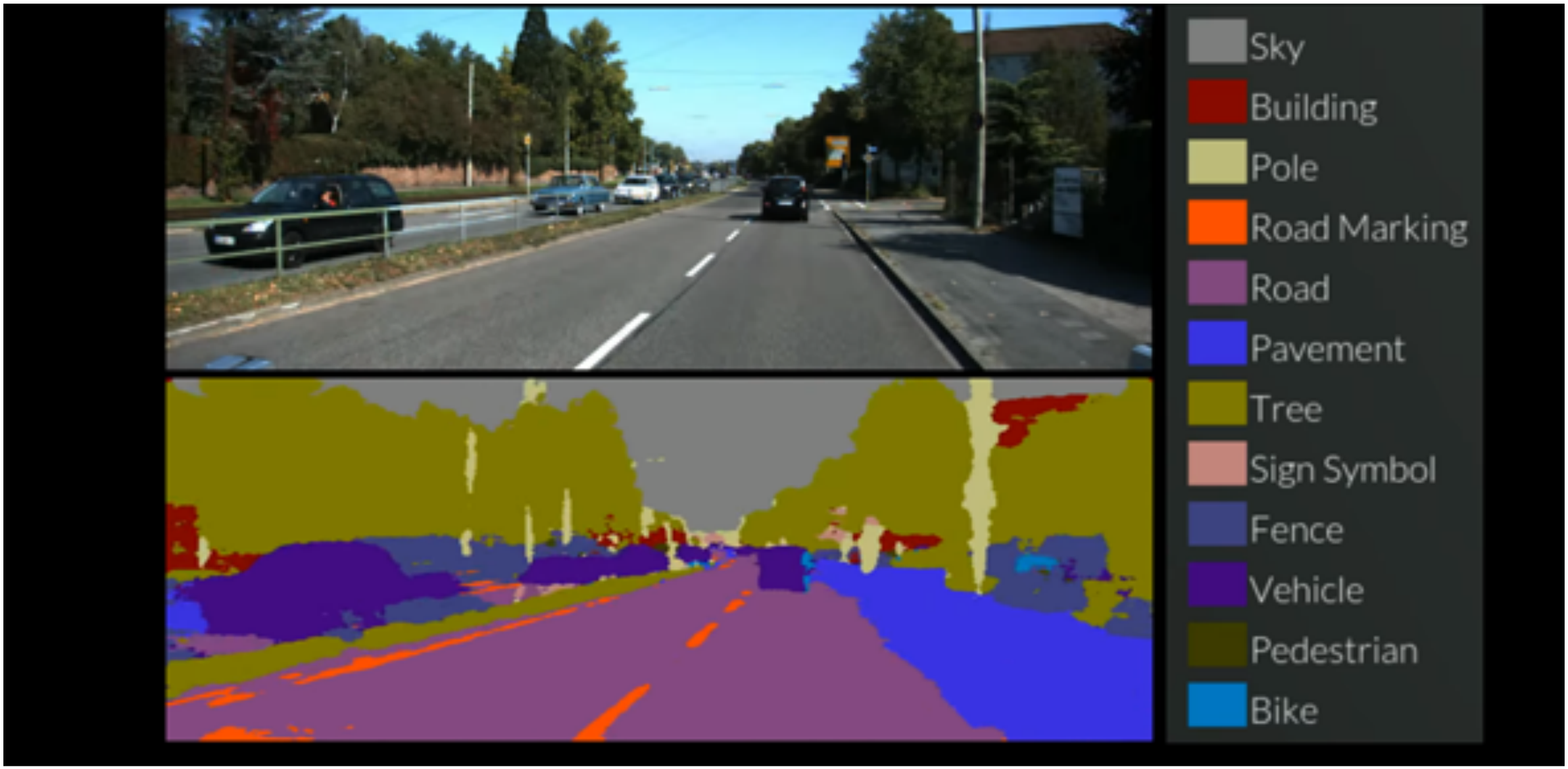
CNNs are everywhere now

Image classification

Object detection



Semantic segmentation



Let's get our hands dirty with some TensorFlow