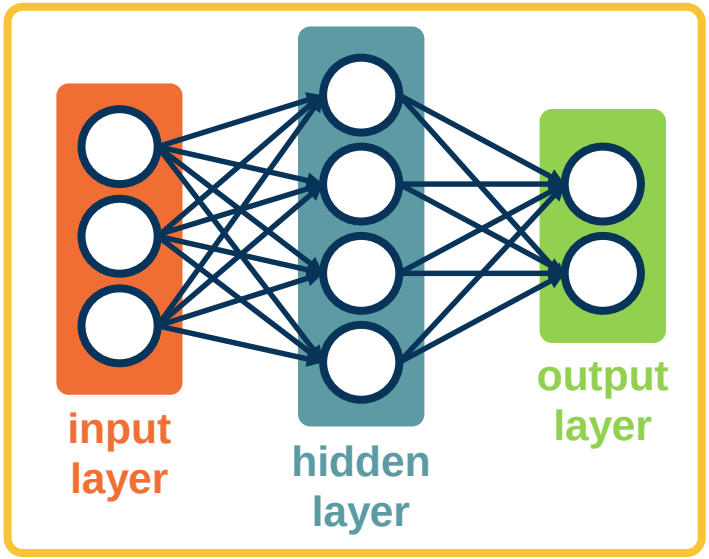
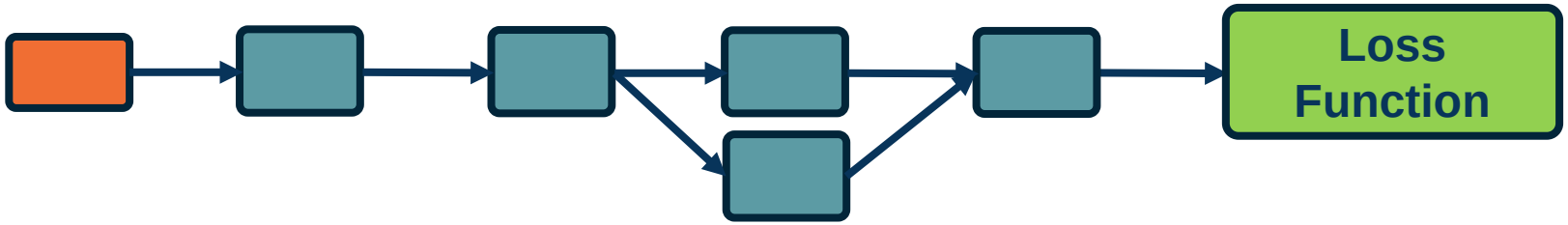
Module 2
Introduction

We have seen how **to build and optimize deep feedforward architectures** consisting of linear & non-linear (e.g. ReLU) layers

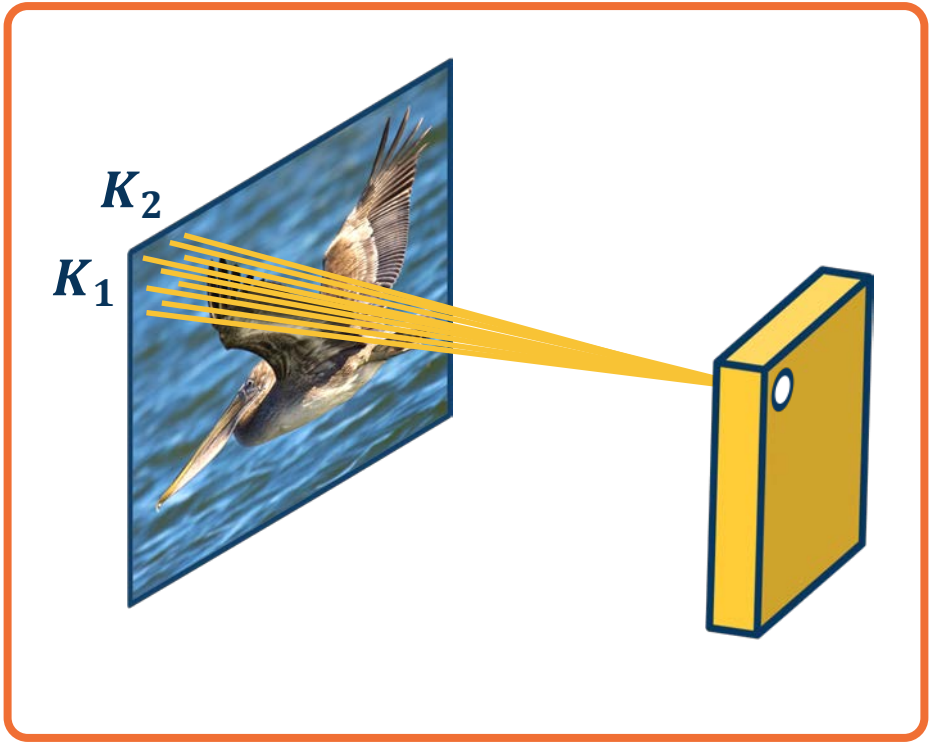
- This can be generalized to **arbitrary computation graphs**
- **Backpropagation and automatic differentiation** can be used to optimize all parameters via **gradient descent**





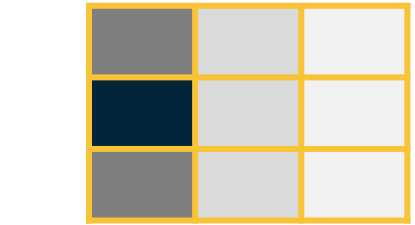
But layers don't have to be fully connected – **other connectivity structures are possible**

For images, it makes sense for output nodes to **consider only small patches of inputs**



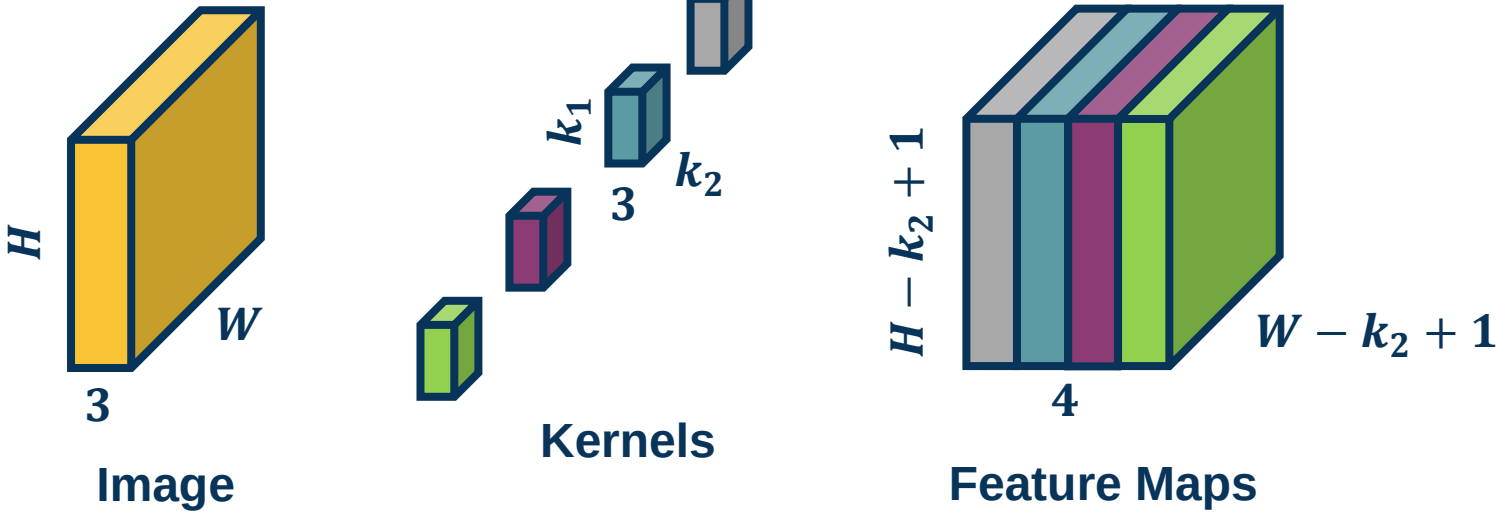
This operation, which can be another layer in the neural network, can be formalized as **convolution operations**

$$K = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

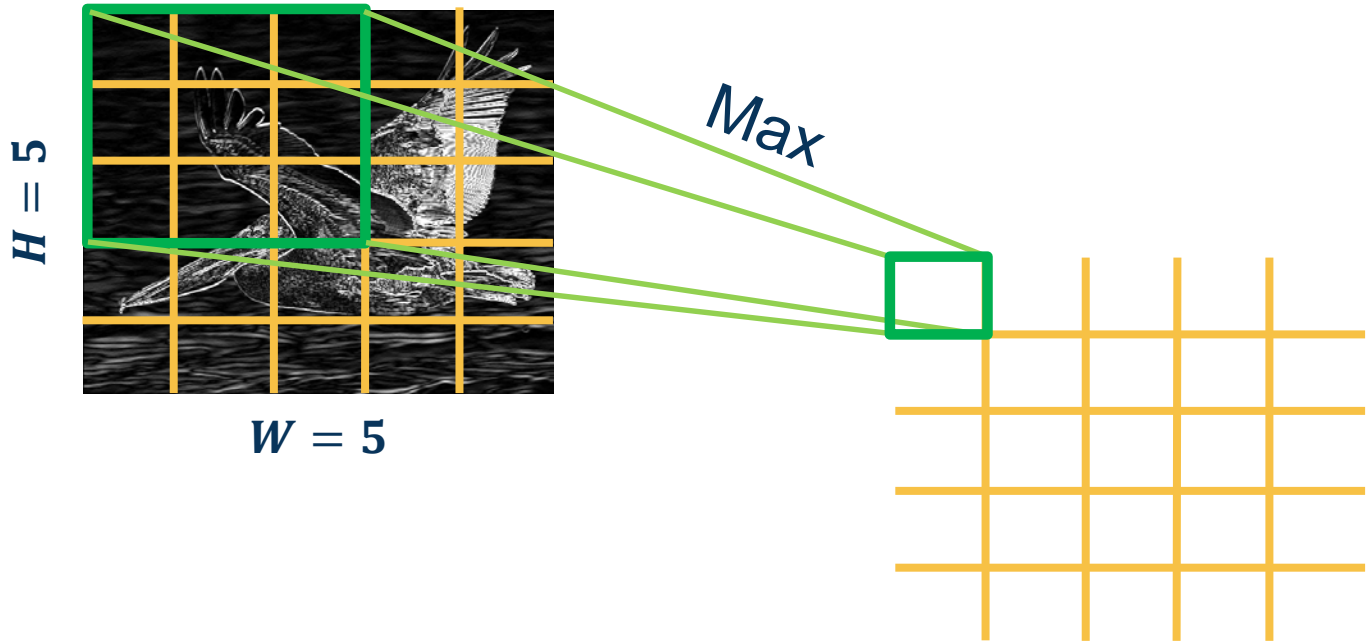


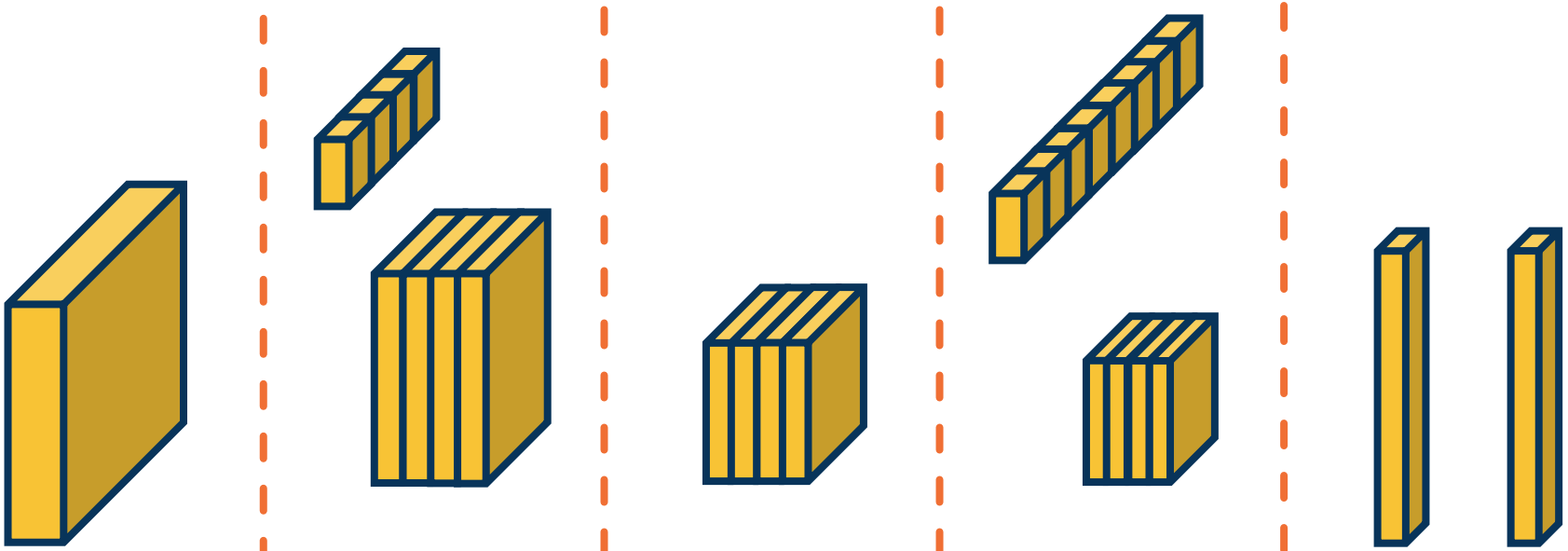
This new convolution layer can take **any input 3D tensor (say, RGB image)** and **output another similarly-shaped output**

- Where **high values represent strong responses** of the learned features



Convolution layers can be combined with non-linear and **pooling layers** which reduce the dimensionality of the data





Alternating Convolution and Pooling

These architectures have existed **since 1980s**

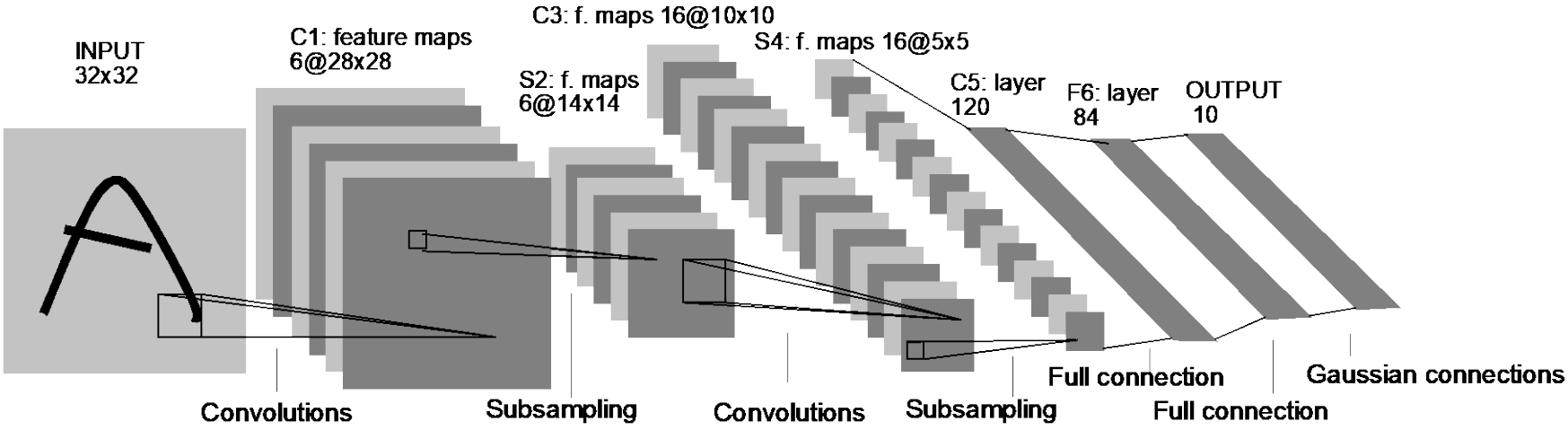
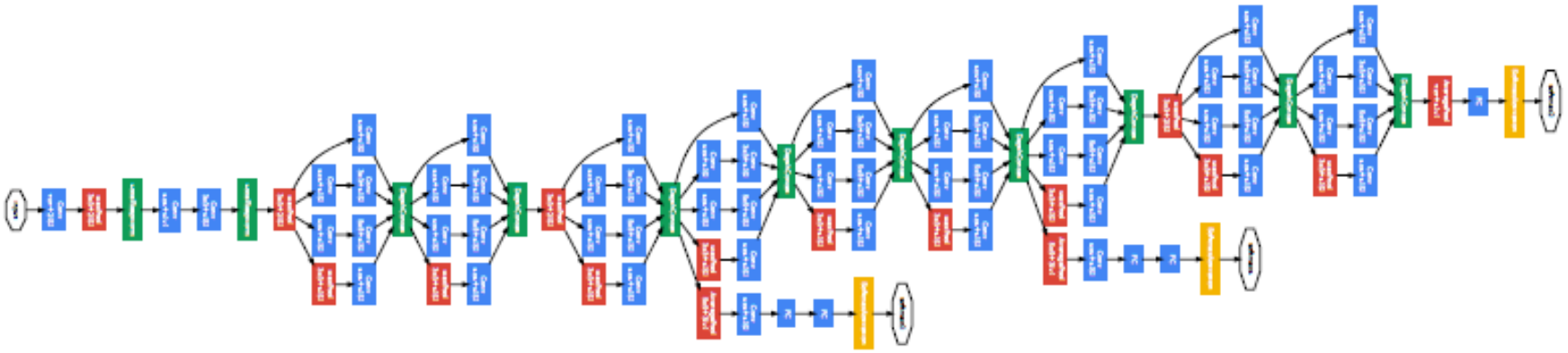


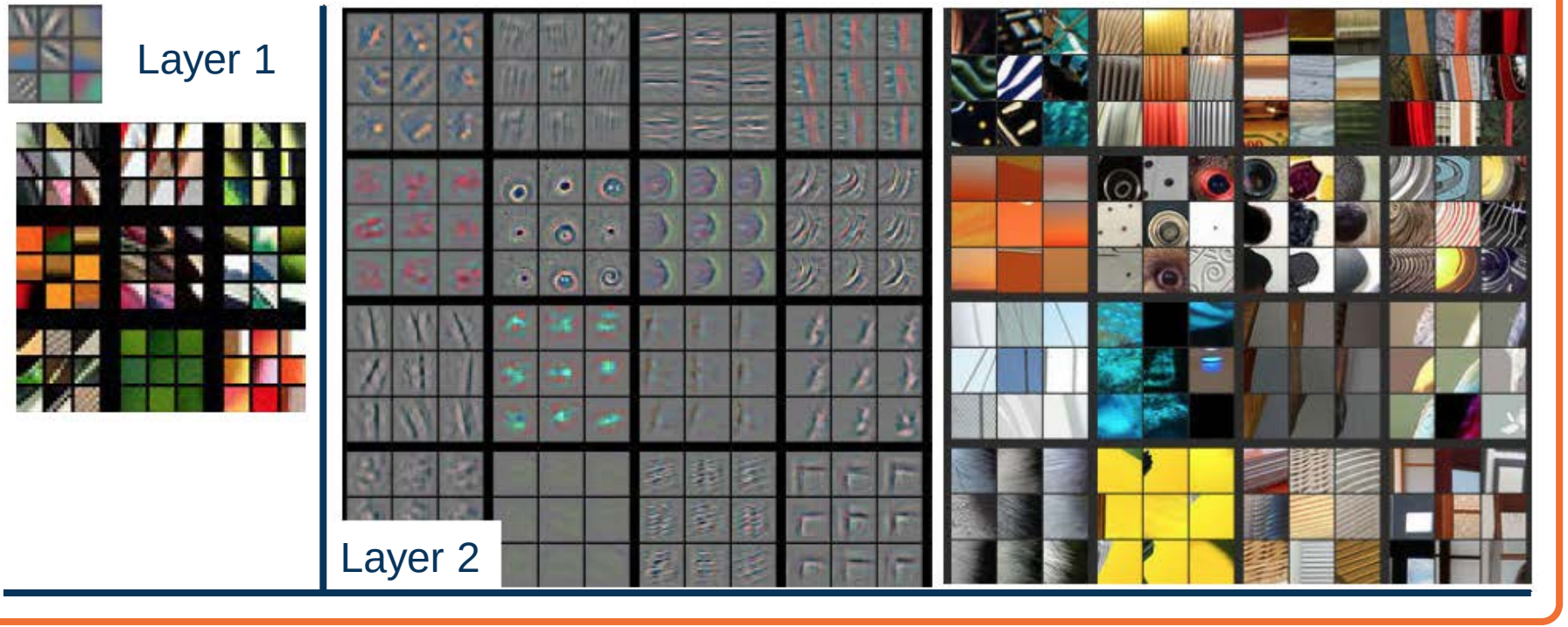
Image Credit: Yann LeCun, Kevin Murphy

LeNet Architecture

But have become **deeper and more complex**



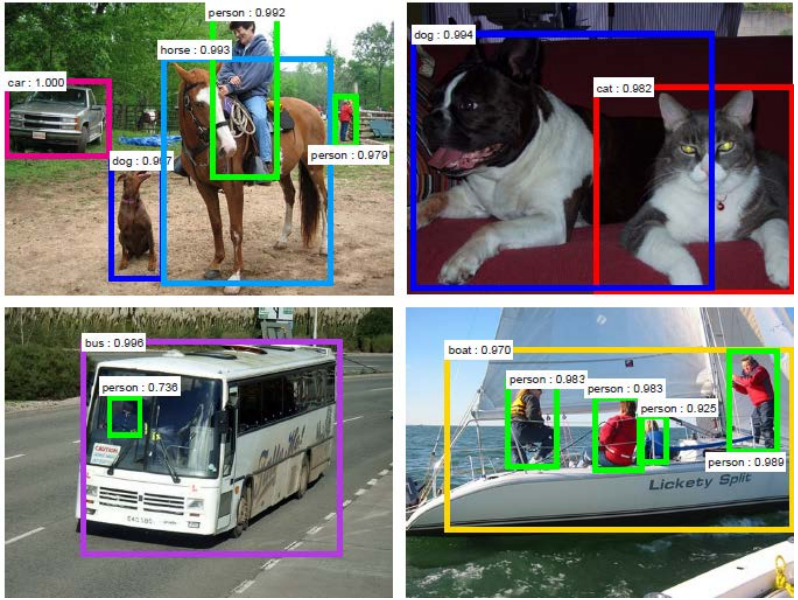
From: Szegedy et al. Going deeper with convolutions

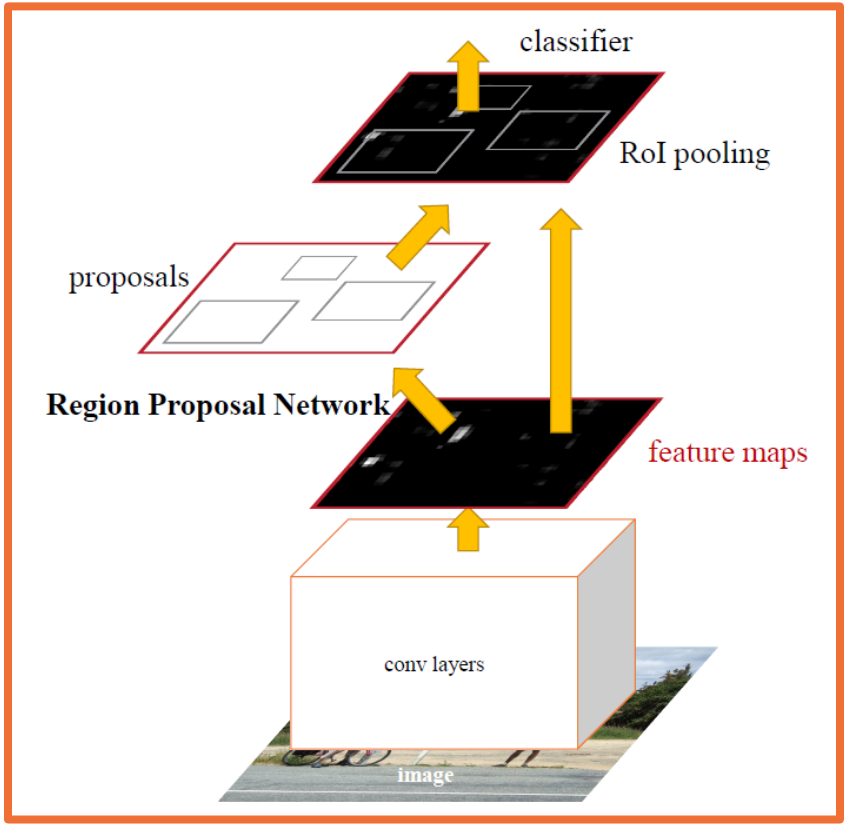


From: Zeiler & Fergus, Visualizing and Understanding Convolutional Networks

Visualization of Learned Features

More complex tasks, like **object detection**, show the real power of deep learning





From: Ren et al., Faster R-CNN