Convolutional Neural Networks

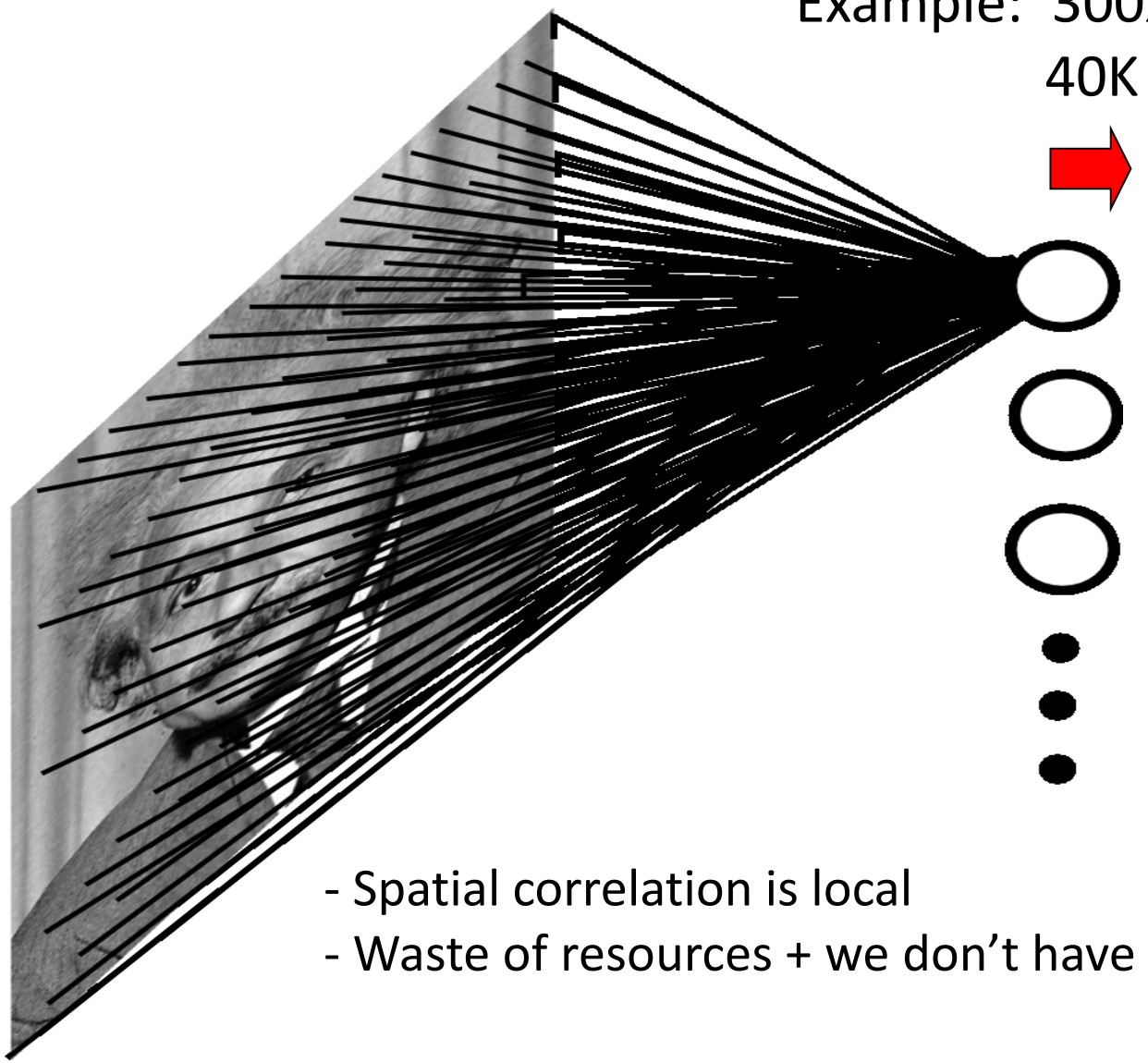
Srikumar Ramalingam

Reference

Many of the slides are prepared using the following resources:

- Marc'Aurelio Ranzato Deep learning tutorial in CVPR 2014

Fully Connected Layer



Example: 300x300 image
40K hidden units

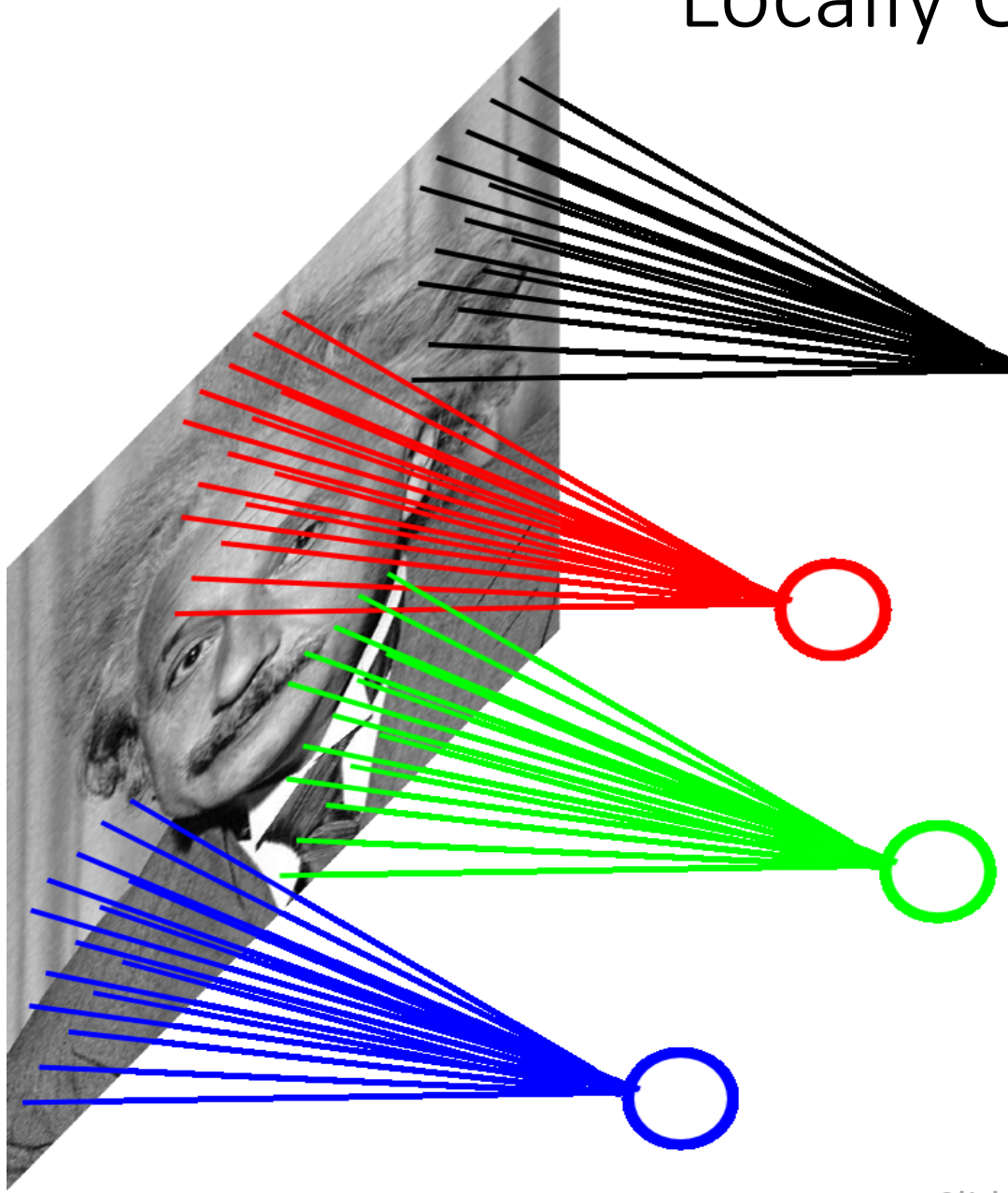
~4B parameters!!!

Fully connected layers do not take into account the spatial structure of the images. For instance, it treats input pixels which are far apart and close together on exactly the same footing.

- Spatial correlation is local
- Waste of resources + we don’t have enough training samples anyway..

Slide Credit: Marc'Aurelio Ranzato

Locally Connected Layer

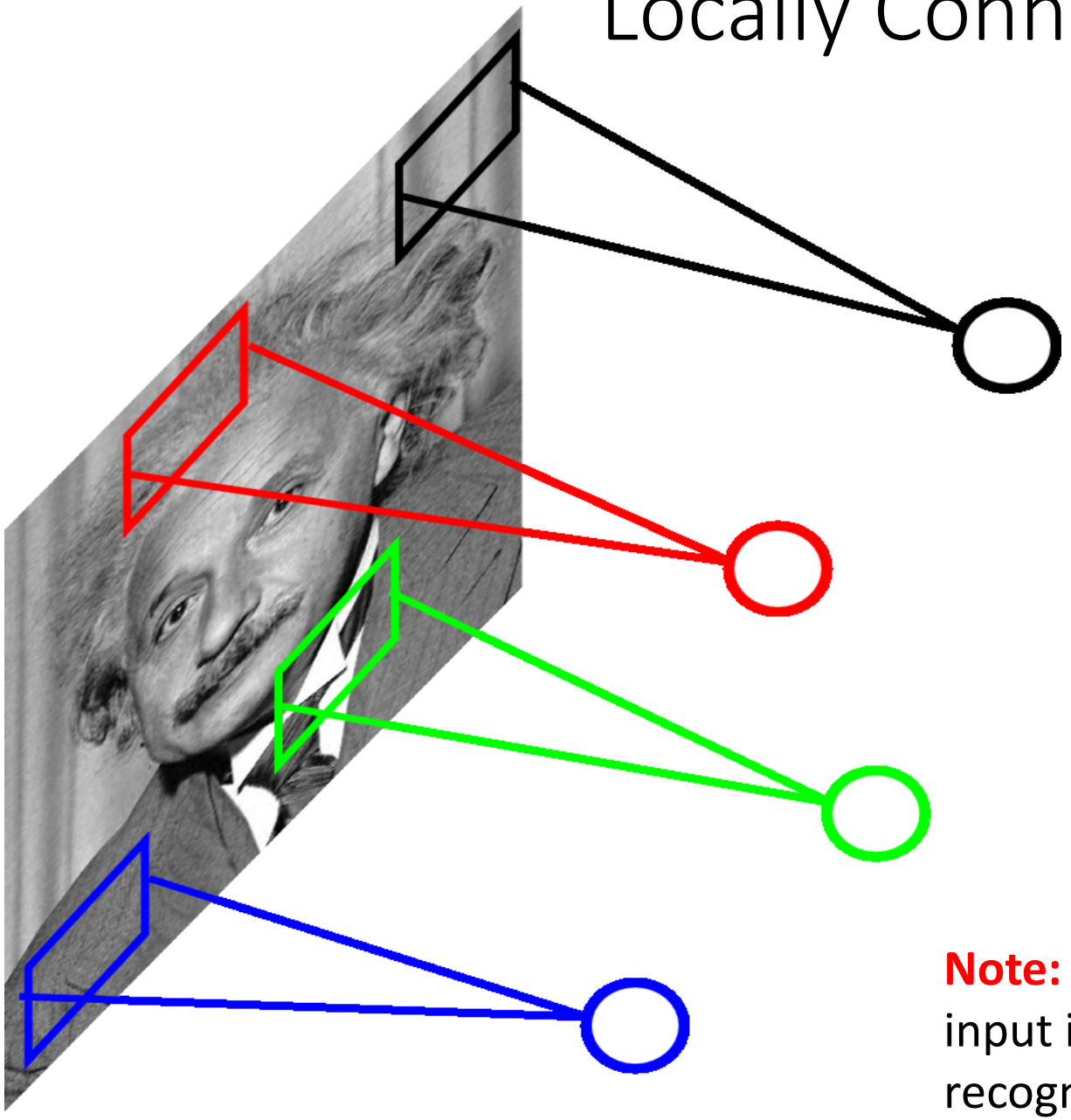


Example: 300x300 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good when input image is registered (e.g., face recognition).

Slide Credit: Marc'Aurelio Ranzato

Locally Connected Layer

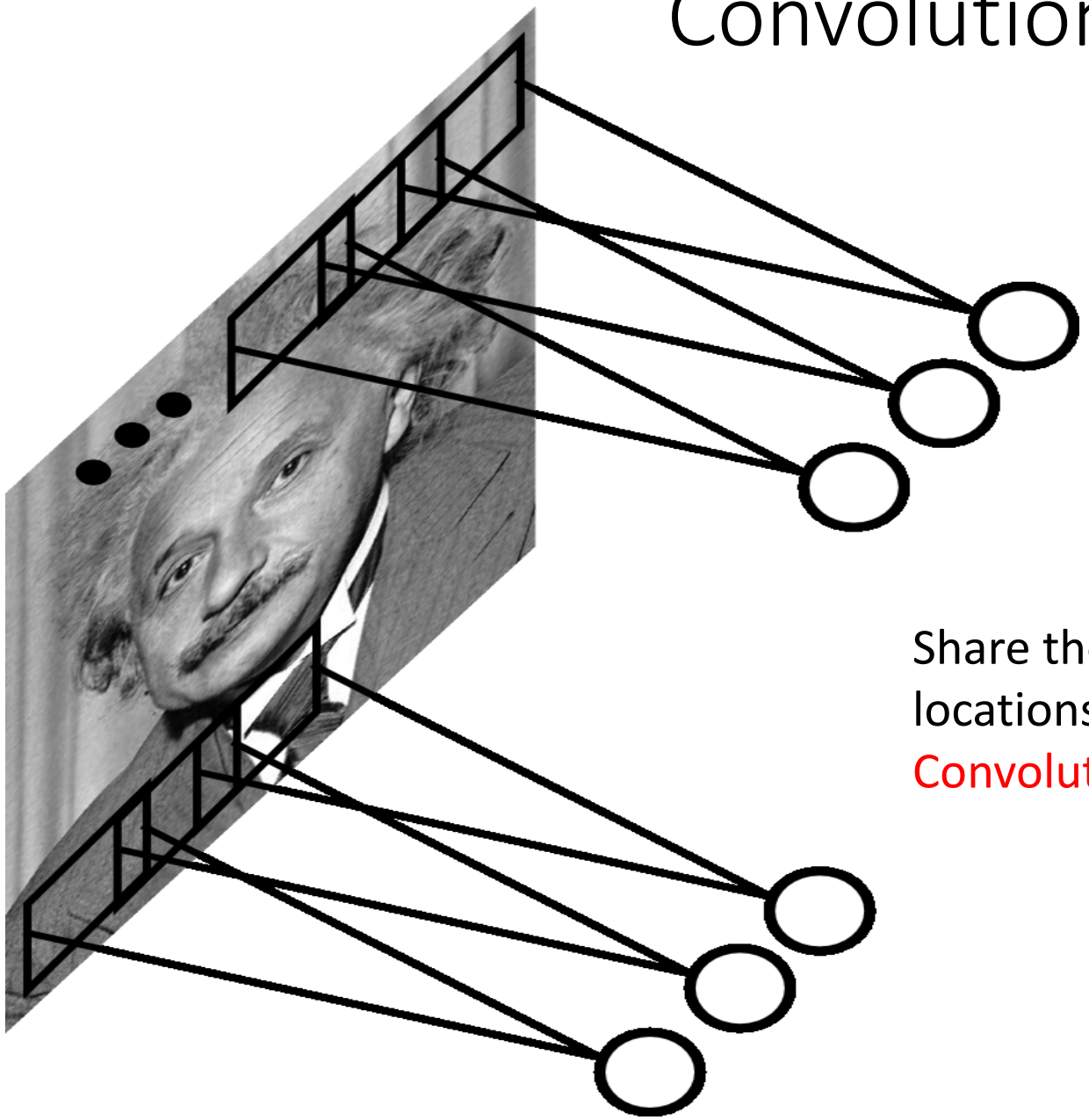


Example: 300x300 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good when input image is registered (e.g., face recognition).

Slide Credit: Marc'Aurelio Ranzato

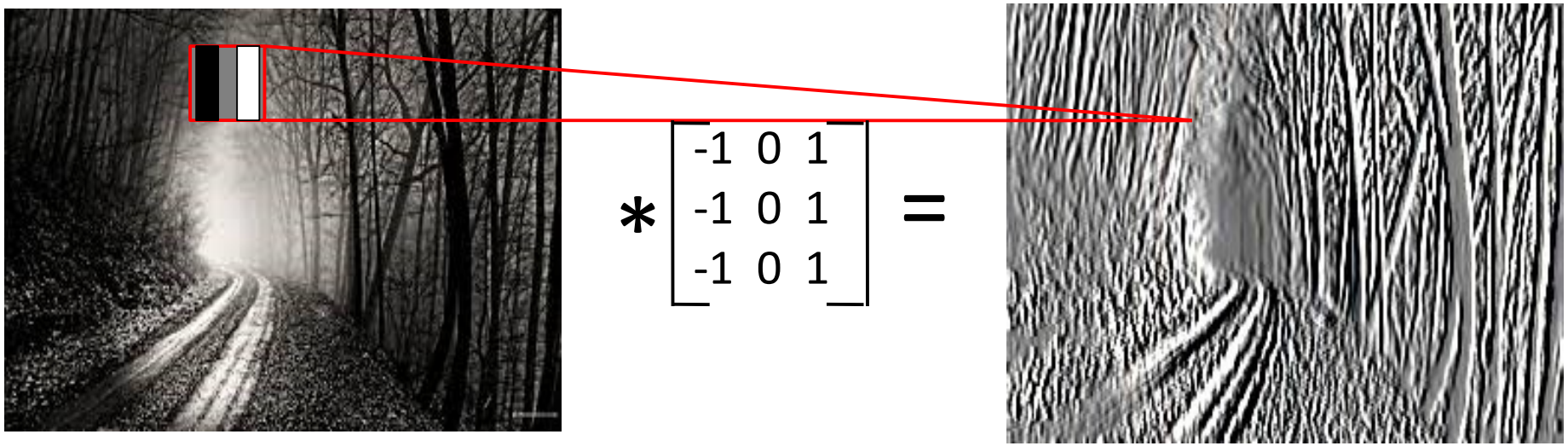
Convolutional Layer



Share the same parameters across different locations (assuming input is stationary):
Convolutions with learned kernels

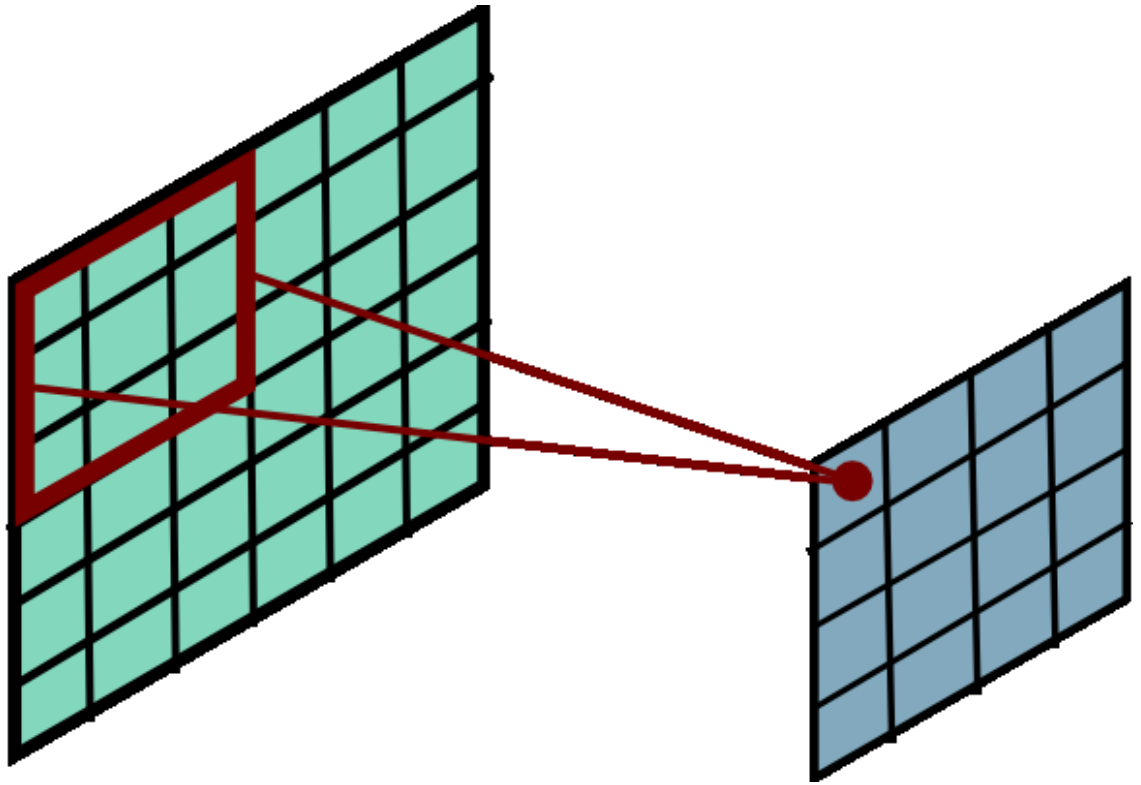
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



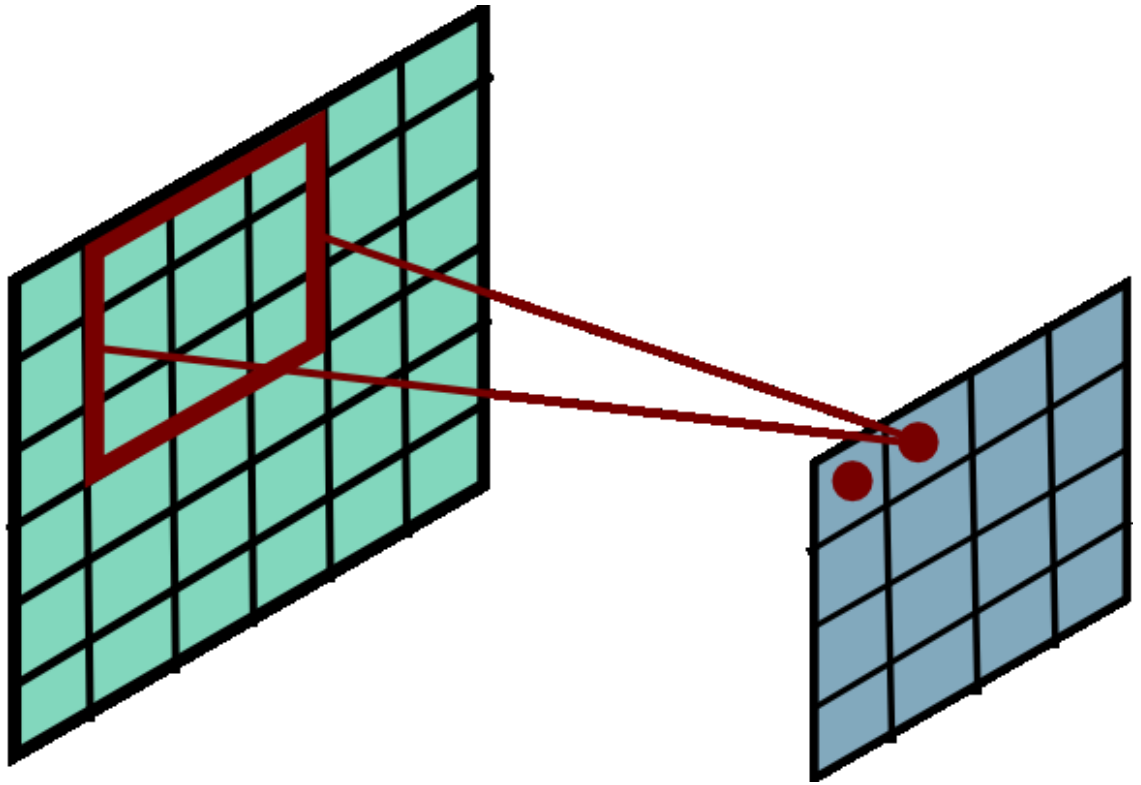
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



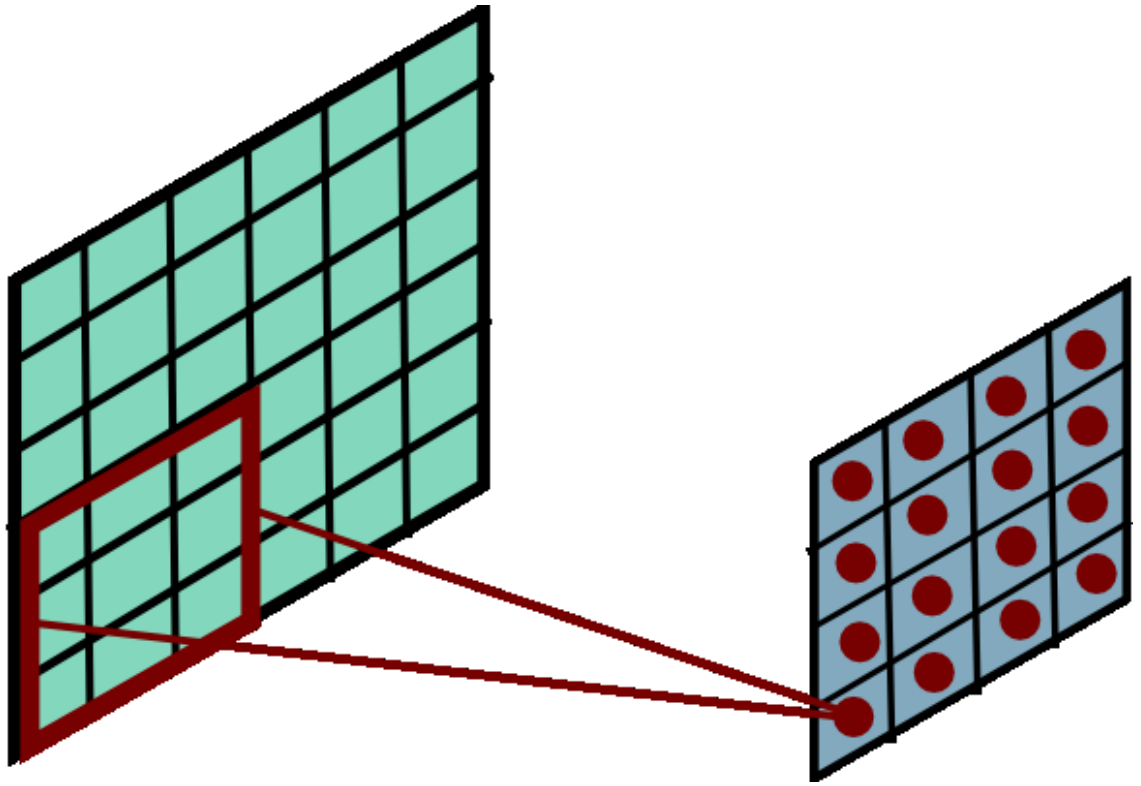
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



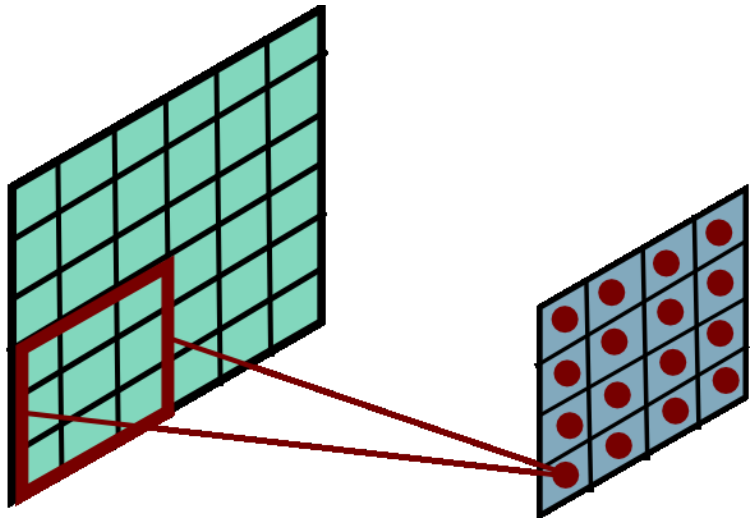
Slide Credit: Marc'Aurelio Ranzato

Convolutional Layer



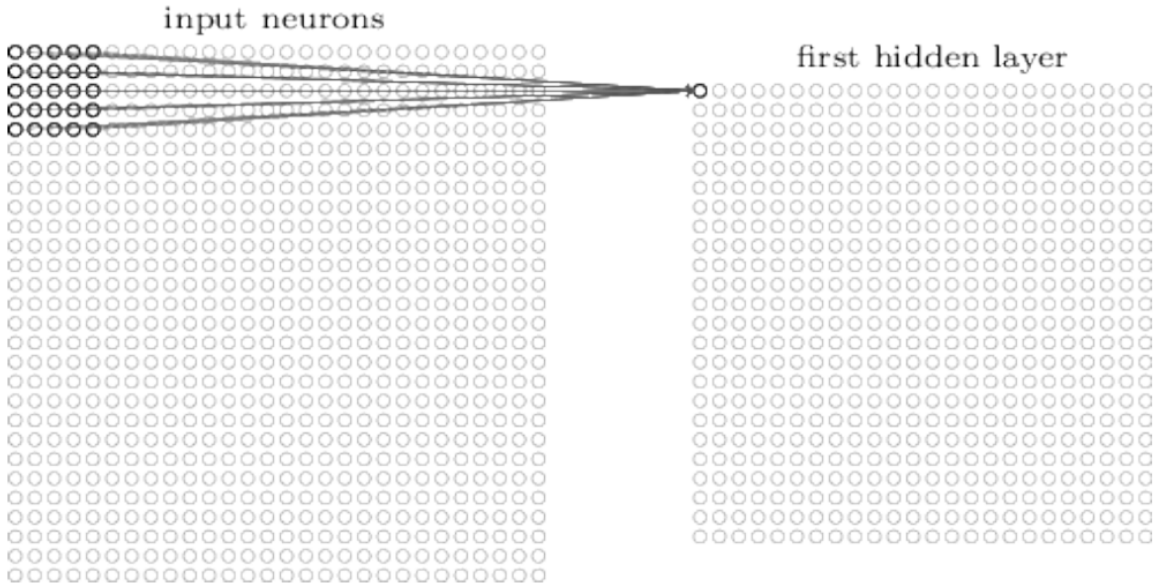
Slide Credit: Marc'Aurelio Ranzato

Local Receptive Fields in CNNs



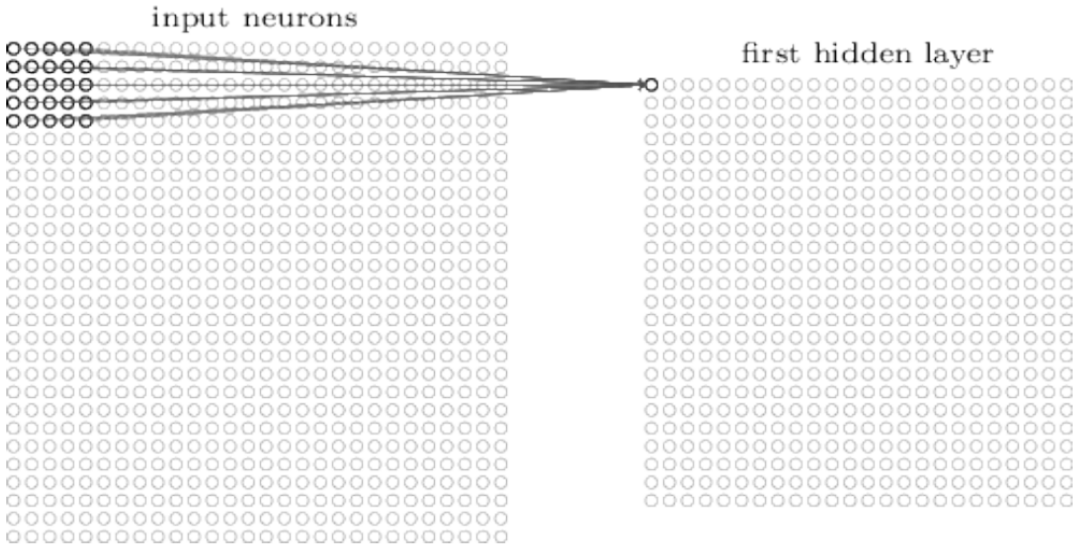
- Local receptive fields – each neuron in the hidden layer will be connected to a small window of input neurons, say 5x5 region, corresponding to 25 input neurons. This input area of a neuron is called its *receptive field*.
- Each hidden neuron can be thought of analyzing its local receptive field.

Local Receptive Fields in CNNs



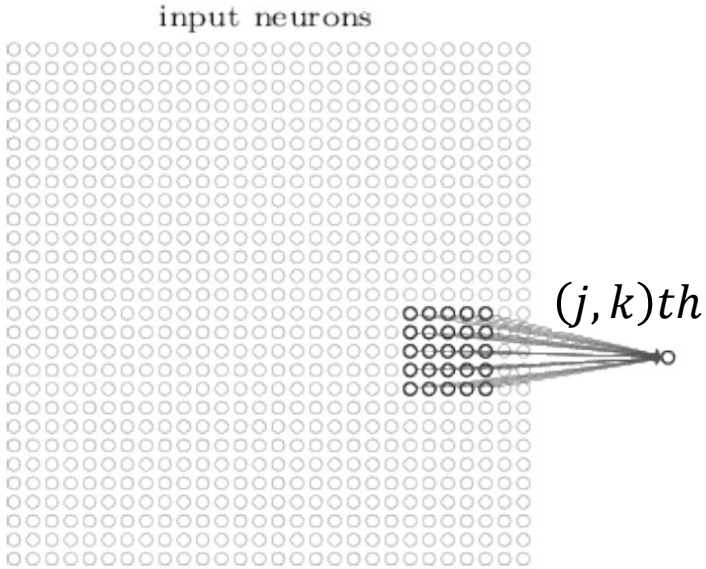
- We slide the local receptive field over by one pixel to the right (i.e., by one neuron), to connect to a second hidden neuron.
- If we have a 28x28 image and 5x5 receptive field, we will have 24x24 hidden neurons.

Stride Length



- Sometimes we slide the local receptive field over by more than one pixel to the right (or down). In that case the stride length could be 2 or more.
- This will lead to fewer hidden neurons. For example, in the case of a 28x28 image with 5x5 receptive field and stride length 2, we will have just 12x12 hidden neurons.

Shared Weights and Biases

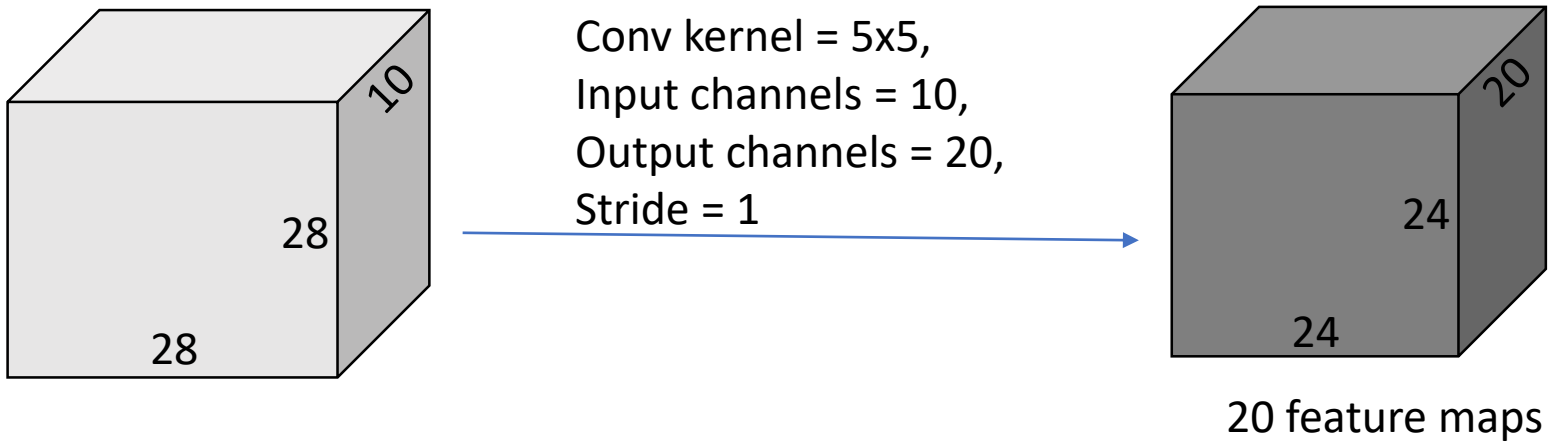


$(j,k)th\ hidden\ neuron$

$$\sigma\left(b+\sum_{l=0}^{4}\sum_{m=0}^{4}w_{l,m}a_{j+l,k+m}\right)$$

- σ is the activation function such as sigmoid unit, b is the bias unit, $w_{l,m}$ is the weight term, and $a_{x,y}$ is the input variable at location (x, y).
- Note that the weights and the bias terms are the same for all the hidden neurons in one feature map.
- All the neurons in the first hidden layer detect exactly the same feature, just at different locations in the image, e.g, cats.

Feature maps



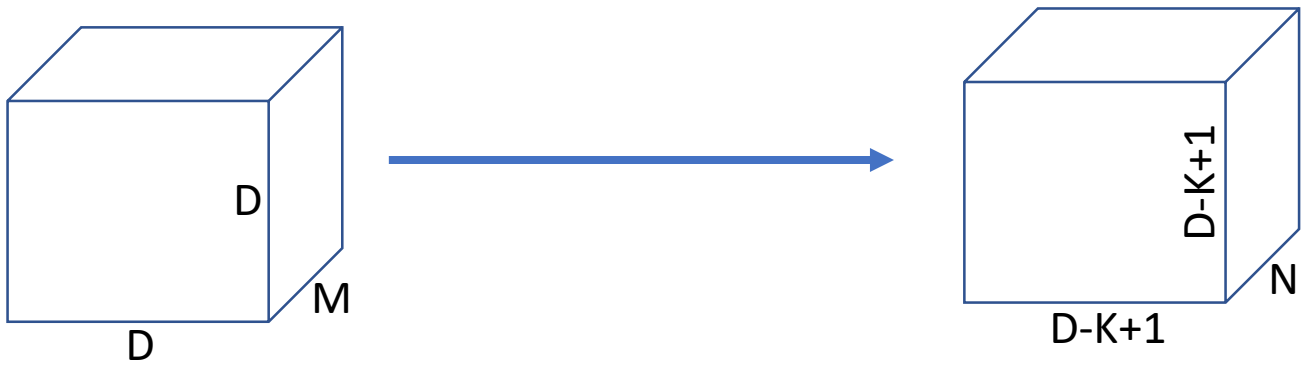
- We call this map from the input to the hidden layer as the feature map.
- The weights are called shared weights and biases are called shared biases, and they are only shared in one feature map. Different feature maps have different weights and biases. We use 20 different filters and each filter is of dimension 5x5x10.

Convolutional Layer

Question: What is the size of the output? What's the computational cost?

Answer: It is proportional to the number of filters and depends on the stride. If kernels have size KxK, input has size DxD, stride is 1, and there are M input feature maps and N output feature maps then:

- the input has size M@DxD
- the output has size N@(D-K+1)x(D-K+1)
- the kernels have MxNxKxK coefficients (which have to be learned)
- cost: M*K*K*N*(D-K+1)*(D-K+1)



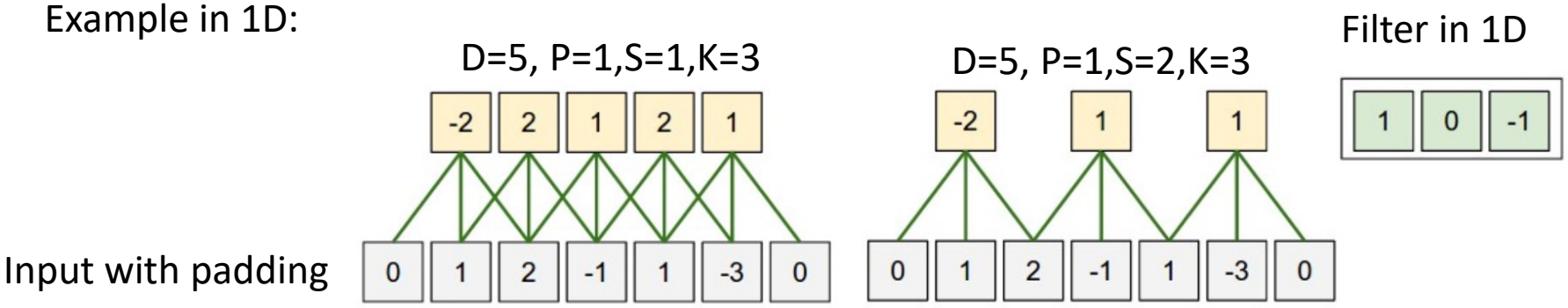
Assumption: zero-padding and stride 1

With padding P and Stride S

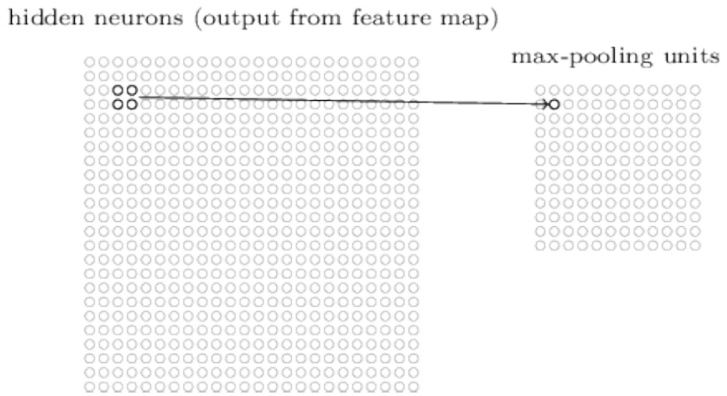
- Input : D x D x M – there are M input channels (M@D x D)
- Let us assume that we have kernels of size K x K
- Let us assume that we have N output channels. What is the dimensions of the output?

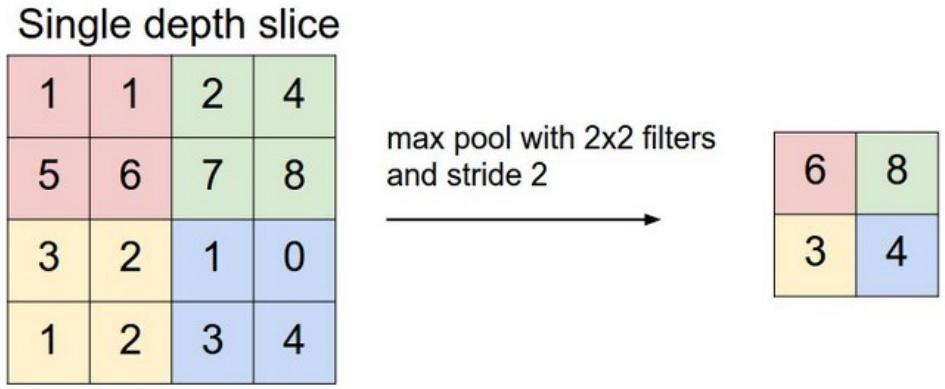
$$N@((D - K + 2P)/S + 1) \times (D - K + 2P)/S + 1))$$

Example in 1D:



Pooling Layers



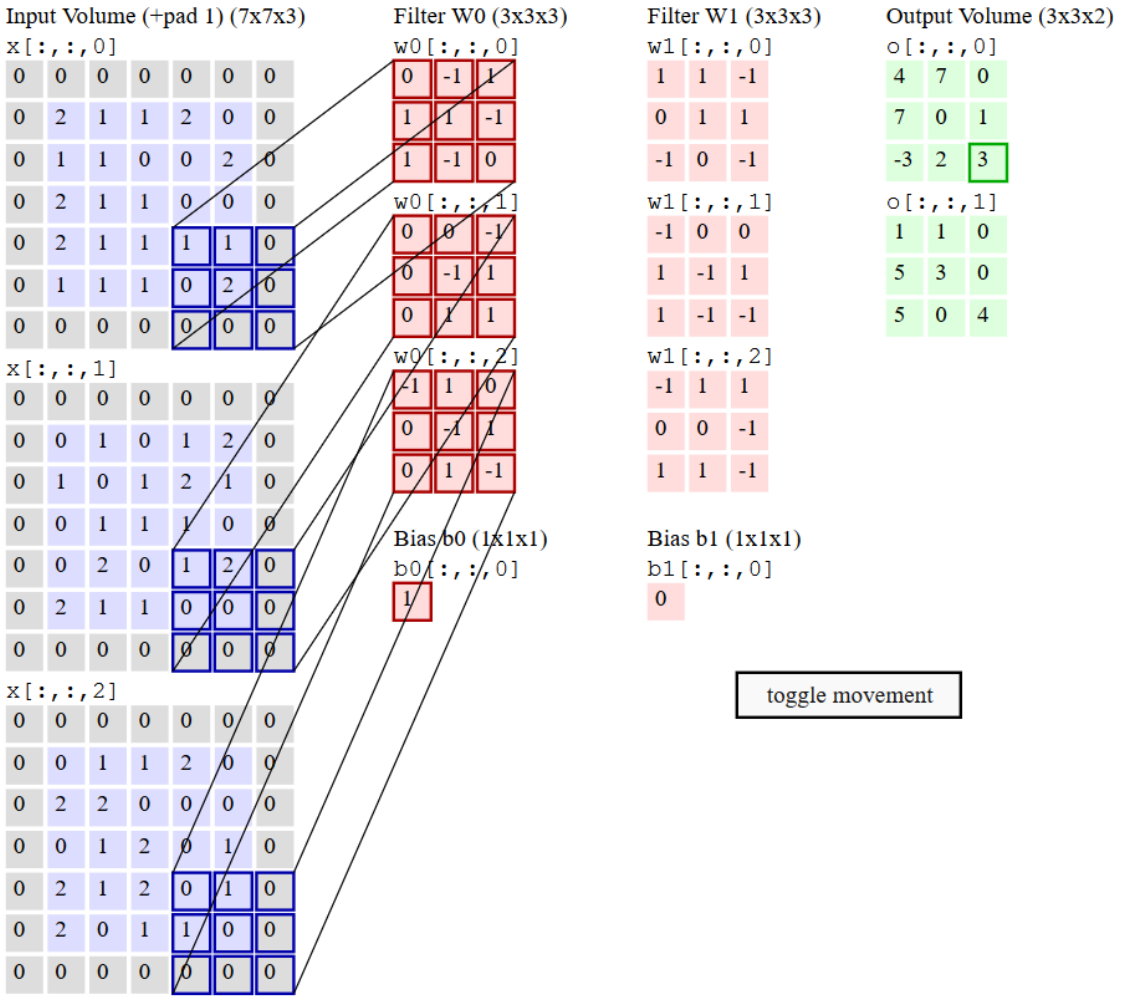


- A pooling layer takes each feature map output from the convolutional layer and prepares a condensed feature map.
- For instance, each unit in the pooling layer may summarize a region of neurons in the previous layer.
- As a concrete example, one common procedure for pooling is known as *max-pooling, e.g., a pooling unit can output the maximum activation in the 2x2 input region.*

Pooling Layers

- L2 pooling – square root of the sum of the squares of the 2x2 regions – several pooling options exist (e.g., average pooling)

Stride S = 2, Padding P = 1, input size D=5, filter size K=3



Why use padding?

- If there is no padding, then the size of the output would reduce by a small amount after each CONV, and the information at the borders would be “washed away” too quickly.

Pooling Layer

Question: What is the size of the output? What's the computational cost?

Answer: The size of the output depends on the stride between the pools. For instance, if pools do not overlap and have size KxK, and the input has size DxD with M input feature maps, then:

- output is M@(D/K)x(D/K)
- the computational cost is proportional to the size of the input (negligible compared to a convolutional layer)

With overlap?

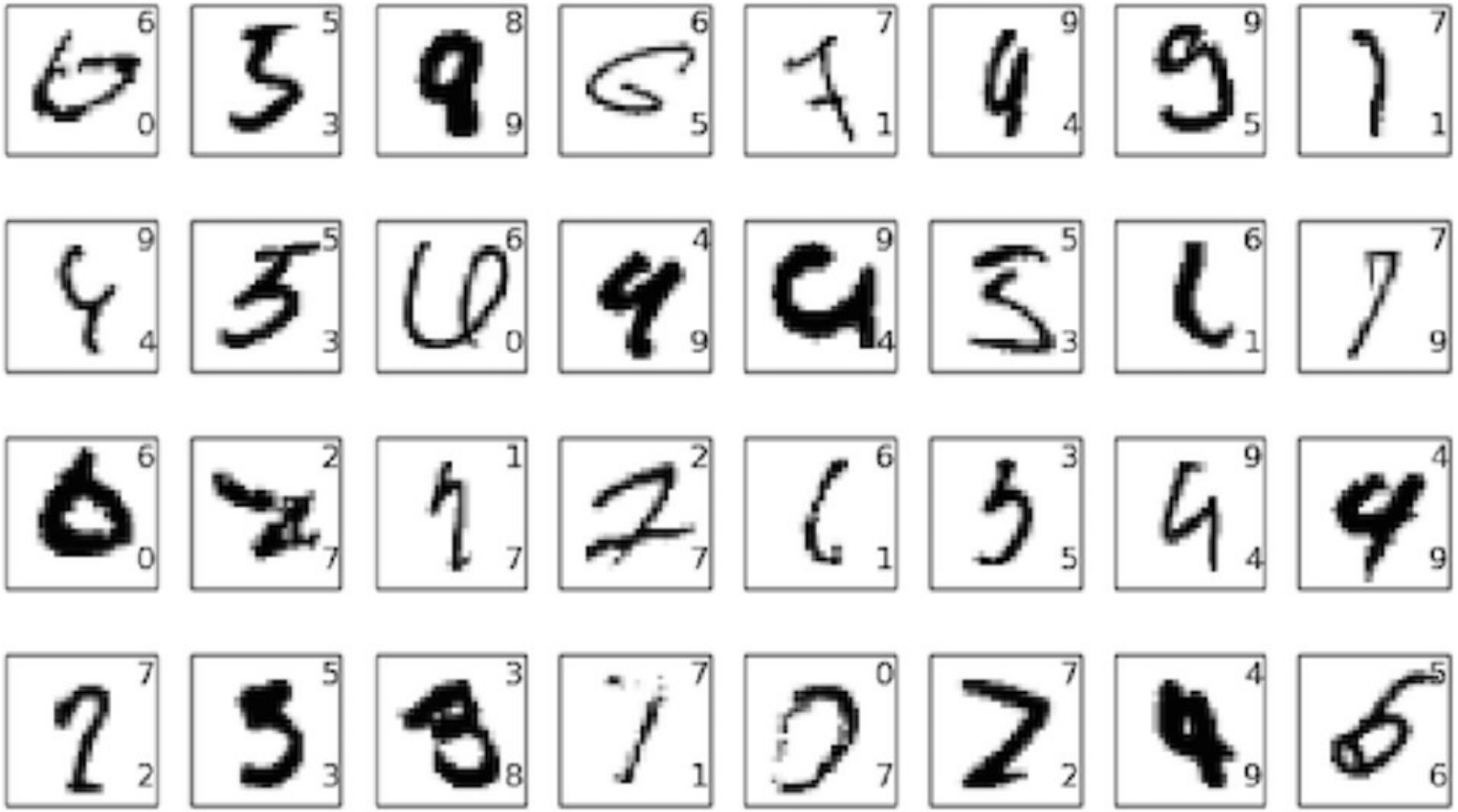
- Stride S and Filter size K
- Output dimension?

$$M@(((D-K)/S+1) \times ((D-K)/S + 1))$$

MNIST data

- Each grayscale image is of size 28x28.
- 60,000 training images and 10,000 test images
- 10 possible labels (0,1,2,3,4,5,6,7,8,9)

Performance on MNIST is near perfect



33 out of 10,000 images are misclassified.
Top Right: correct
Bottom Right: misclassfied

Using several ideas: convolutions, pooling, the use of GPUs to do far more training than we did with shallow networks, the algorithmic expansion of our training, dropouts, etc.

Thank You