

- CNN (functional view)**
- recall fully connected layer

$32 \times 32 \times 3$  image (linear)  $\downarrow$

$3072 \times 1$

input  $3072 \rightarrow 1024 \rightarrow 512 \rightarrow 10$   
weights

- convolutional layer  $32 \times 32 \times 3$  image (preserves shape)

$32 \times 32 \times 3$  image  
fibers always extend the full depth of the input volume

$32 \times 32 \times 3$  image  
5x5x3 filter  $\Rightarrow$  1 neuron  
the number of taking a dot product between the filter and the input volume is the same (i.e.  $5^2 \times 3 = 15$ -dimensional dot product + bias)

$32 \times 32 \times 3$  image  
activation map

$32 \times 32 \times 3$  image  
5x5x3 filter  
convolve (slide over all spatial locations)

$32 \times 32 \times 3$  image  
can be different filter!!!

For example, if we had 5 filters, we'll get 5 separate activation maps:

activation maps  
Convolution Layer

We will then up to get a "new" image of size  $28 \times 28 \times 5$ !

Posterior ConvNet is a response of Convolutional Layers interpreted with activation functions

basically look for features! (high/medium/lows)

Preview

$5 \times 5 \times 3 \times 3 \times 3 \times 3 \times 3$   
=  $5^{10}$  neurons

down sample

Input volume:  $32 \times 32 \times 3$   
10 5x5 filters with stride 1, pad 2

Output volume size:  
 $(32+2*2-5)^2+1 = 32$  spatially, so  
 $32 \times 32 \times 10$

Q. no. of parameters? (do weights)

A:

Examples time:

Input volume:  $32 \times 32 \times 3$   
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?  
each filter has  $5^2 \times 3 + 1 = 76$  params (+1 for bias)

$\geq 76 \times 10 = 760$

Summary To summarize, the Conv Layer

• Accepts a volume of size  $W_1 \times H_1 \times D_1$

• Produces a volume of size  $W_2 \times H_2 \times D_2$

• Number of filters:  $K$

• the stride  $S$

• the padding  $P$

• the kernel size  $F$

•  $W_2 = (W_1 - F + 2P) / S + 1$  width and height are computed equally by symmetry

• With parameter sharing, it introduces  $F \times D_1$  weights per filter for a total of  $(F \times F \times K)$   $K$  weights

• In the output volume, the  $d$  is depth slice of size  $W_2 \times H_2 \times D_2$  is the result of performing a valid convolution

• If the input volume has  $N$  channels, then the output volume has  $N$  channels

• you can do 1x1 conv

Example: CONV layer in Torch

**SpatialConvolution**  
Applies a 2D convolution over an input image composed of several input planes. The output tensor's dimensionality is expected to be a 3D tensor of size  $(\text{output\_size} \times \text{output\_size} \times \text{depth})$ .

Parameters are the following

- $\text{in\_size}$ : The input size (number of input image planes).
- $\text{out\_size}$ : The number of output planes.
- $\text{K}$ : The number of filters.
- $\text{F}$ : The kernel height of the convolution.
- $\text{H}$ : The kernel width of the convolution.
- $\text{S}$ : The stride length of the convolution. Default is 1.
- $\text{P}$ : The padding length of the convolution. Default is 0.
- $\text{bias}$ : The additive term added per plane to the input planes. Default is 0, a global number.
- $\text{group}$ : The additional groups added per plane to the input planes. Default is 1, a global number.

Note that the size of the sum of all terms involved in the calculation is  $\text{out\_size} \times \text{out\_size} \times \text{depth}$ . If the input image is a 3D tensor (spatial\_size  $\times$  height  $\times$  width), the output image will be a 3D tensor (spatial\_size  $\times$  height  $\times$  width  $\times$  depth).

If the input image is a 4D tensor (batch\_size  $\times$  spatial\_size  $\times$  height  $\times$  width), the output image will be a 4D tensor (batch\_size  $\times$  out\_size  $\times$  out\_size  $\times$  depth).

Summary To summarize, the Conv Layer

• Accepts a volume of size  $W_1 \times H_1 \times D_1$

• Requires three hyperparameters:

- the spatial extent  $F$ , **kernel size**
- the stride  $S$
- the depth  $D_2$

• Produces a volume of size  $W_2 \times H_2 \times D_2$  where:

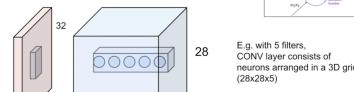
- $W_2 = (W_1 - F) / S + 1$
- $H_2 = (H_1 - F) / S + 1$
- $D_2 = D_1$

• Introduces zero parameters since it computes a fixed function of the input

• Note that it is not common to use zero-padding for Pooling layers

- **six filter**  $\rightarrow$  six receptive field for each neuron

The brain/neuron view of CONV Layer



There will be 5 different neurons all looking at the same region in the input volume

- **fully connected network** (takes the entire input domain)

- **Pooling layer**

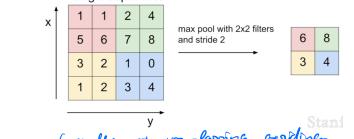
- makes the representations smaller & more manageable
- operates over each activation map independently

$224 \times 224 \times 64$   $\rightarrow$   $112 \times 112 \times 64$



- **max pooling** (instead of dot-product, we do maximization)

Single depth slice



(usually not overlapping sampling)

- **averaging pooling**

• Accepts a volume of size  $W_1 \times H_1 \times D_1$

• Requires three hyperparameters:

- the spatial extent  $F$ , **kernel size**
- the stride  $S$
- the depth  $D_2$

• Produces a volume of size  $W_2 \times H_2 \times D_2$  where:

- $W_2 = (W_1 - F) / S + 1$
- $H_2 = (H_1 - F) / S + 1$
- $D_2 = D_1$

• Introduces zero parameters since it computes a fixed function of the input

• Note that it is not common to use zero-padding for Pooling layers

prev: down-sampling



- **Fully connected layer**

CNN output  $N \times H \times D$   $\rightarrow$  map  $\rightarrow$  result space

Examples time:

Input volume:  $32 \times 32 \times 3$

10 5x5 filters with stride 1, pad 2

Output volume size:

$(32+2*2-5)^2+1 = 32$  spatially, so

$32 \times 32 \times 10$

Q. no. of parameters? (do weights)

A:

Examples time:

Input volume:  $32 \times 32 \times 3$

10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?

each filter has  $5^2 \times 3 + 1 = 76$  params (+1 for bias)

$\geq 76 \times 10 = 760$

Summary To summarize, the Conv Layer

• Accepts a volume of size  $W_1 \times H_1 \times D_1$

• Produces a volume of size  $W_2 \times H_2 \times D_2$

• Number of filters:  $K$

• the stride  $S$

• the padding  $P$

• the kernel size  $F$

•  $W_2 = (W_1 - F + 2P) / S + 1$  width and height are computed equally by symmetry

• With parameter sharing, it introduces  $F \times D_1$  weights per filter for a total of  $(F \times F \times K)$   $K$  weights

• In the output volume, the  $d$  is depth slice of size  $W_2 \times H_2 \times D_2$  is the result of performing a valid convolution

• If the input volume has  $N$  channels, then the output volume has  $N$  channels

• you can do 1x1 conv