Last time

- → Fully connected layer
- → Activation functions

Today

- → Convolutional (2D) layer
- → pooling layers

Ex: $p = 9, \; d_1 = 16, \; d_2 = 8, \; d_3 = 1$

$$f_{dense}(X) = [\max(0, \max(0, XW_1 + B_1)W_2 + B_2)]W_3 + B_3$$

$X = [n, p]$

$X_i = [1, p]$

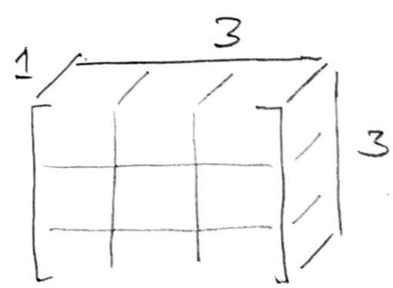
Total parameter count $= 305$

1st layer parameter count $= 160$

Now

$X = [n, \sqrt{p}, \sqrt{p}, 1]$

$X_i = [1, \sqrt{p}, \sqrt{p}, 1]$

$X_i =$ 

$$f_{conv}(X) = [\max(0, \text{flatten}(\text{Conv2D}(X))W_2 + B_2)]W_3 + B_3$$

2D Convolutional Layer

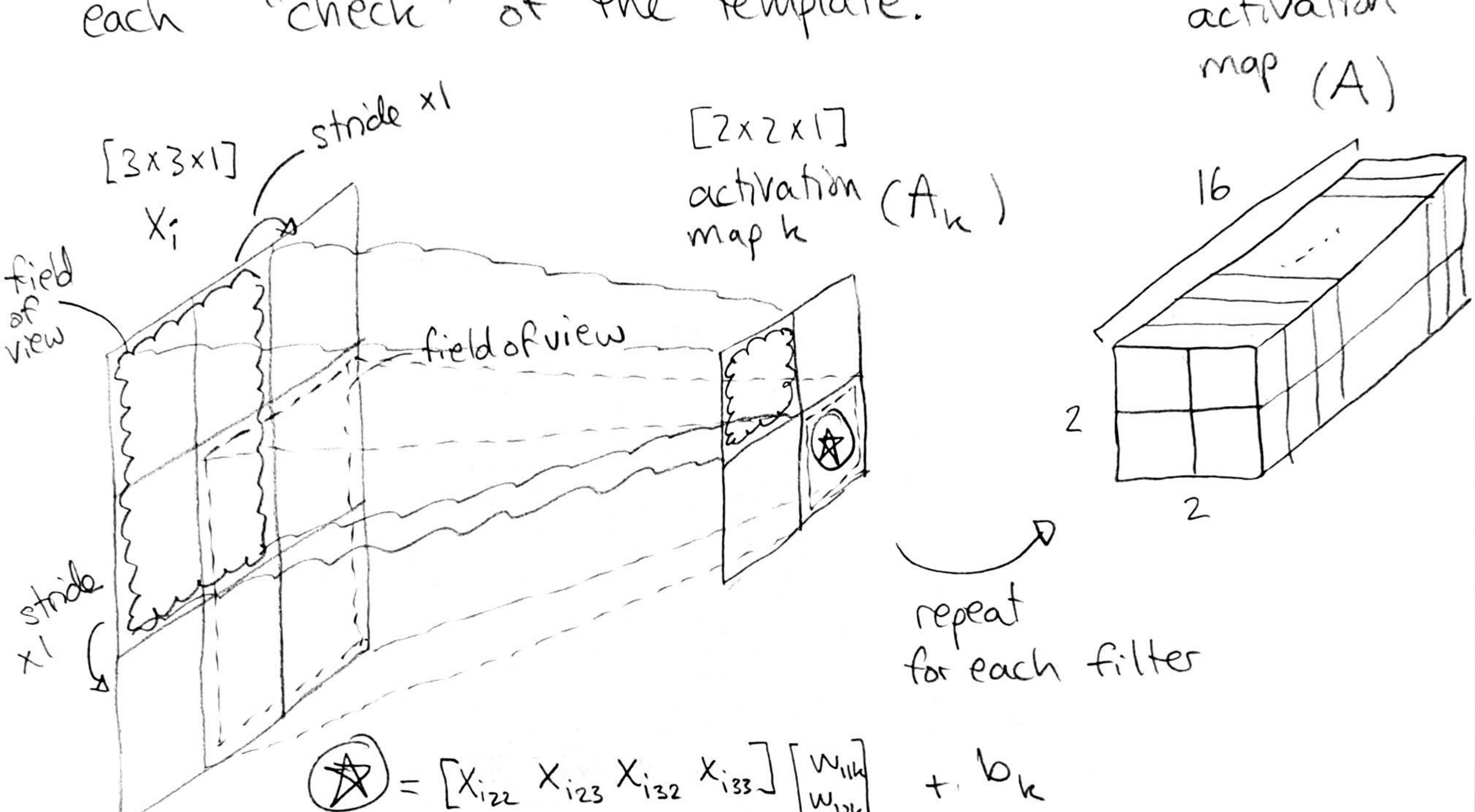
- a restricted type of template checking
- assumes spatial structure, templates are only checked against a spatially-restricted window referred to as the <u>field of view</u>.
- commonly used to process image data
- <u>filter</u>: (or kernel) the template being checked for. Analogous to a neuron in the dense layer.
  - W : [filter height, filter width, filter depth]
  - B : [1]

  <u>Note</u>: layer's parameter size is not a function of p (the input size)

- <u>strides</u>: the step sizes, in both the height and width directions, that are used to move the field of view across the input.

- <u>activation map</u>: the spatially organized output of the application of a filter to the input.

Note: Conv layer is not fully connected because each element of an input is not considered in each "check" of the template.



$$\star = \begin{bmatrix} X_{i22} & X_{i23} & X_{i32} & X_{i33} \end{bmatrix} \begin{bmatrix} w_{11k} \\ w_{12k} \\ w_{21k} \\ w_{22k} \end{bmatrix} + b_k$$

strides : [1x1]

filter size : [2x2x1] $\left(\begin{bmatrix} w_{11k} & w_{12k} \\ w_{21k} & w_{22k} \end{bmatrix}, b_k \right)$

\# filters : 16

The output is an "image" of size [2x2x16]

$$\mathrm{Conv2D}(X) = \sigma(A(X))$$

Notes :

- The number of parameters in this layer ($80 = (4+1) \times 16$) is much smaller than the Dense layers used in the example last time which contained the same number of templates (16) and operated on the same sized input (i.e. 1×9 vs $1 \times \sqrt{9} \times \sqrt{9} \times 1$).

- The total parameter size of $f_{conv}(X)$:

$$= 80 + (16 \times 4 \times 8) + 8 + (8 \times 1) + 1$$

$$= 80 + 520 + 9 = 609 \ll f_{dense}$$

- The parameter count for a 2D Conv layer :

$$= (\text{filter height} \times \text{filter width} \times \text{filter depth} + 1) \times \#\text{filters}$$

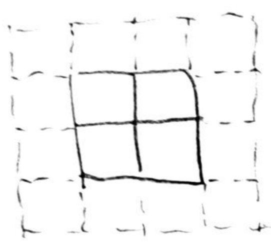
- filter size and strides has a big influence on output size.

- output size $= 1 + \left\lfloor \frac{\text{input size} - \text{filter size} + 2 \times \text{padding}}{\text{stride}} \right\rfloor$

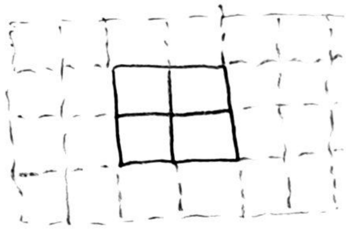
for each dimension (ie height and width) padding: the concatenation of additional height or width elements to the input before template checks

e.g.

padding = [1x1]



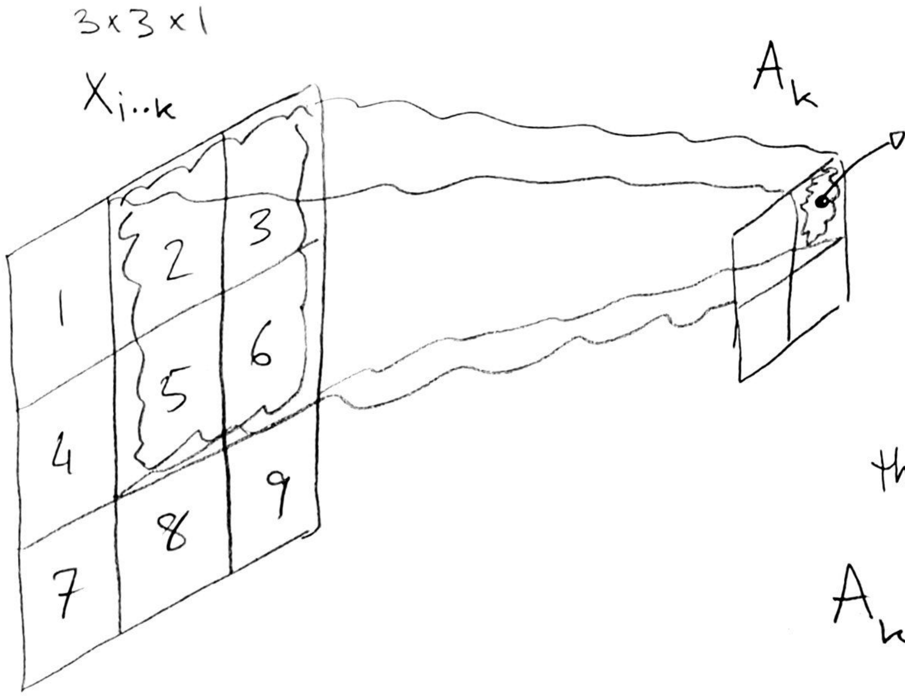
padding = [2x1]



- used to evenly apply filters to each element of the input
- padded elements are usually zero or the mean of mirror-flipping of adjacent elements.

Pooling Layers (2D)

- used to downsample / summarize inputs
- (usually) contain no learnable parameters
- characterized by their pooling operation (g)
  - max, mean
- operate similarly to conv2D layers and are also thus parameterized by filter size, padding and strides.



$$g\begin{pmatrix} x_{i12k}, x_{i13k}, \\ x_{i22k}, x_{i23k} \end{pmatrix}$$

if $g = \max$

then

$$A_k = \begin{array}{|c|c|} \hline 5 & 6 \\ \hline 8 & 9 \\ \hline \end{array}$$

if $g = \text{mean}$

$$A_k = \begin{array}{|c|c|} \hline 3 & 4 \\ \hline 6 & 7 \\ \hline \end{array}$$

strides : [1x1]

filter size : [2x2]

→ Note : Unlike convolutional layers, Pooling layers operate on a single channel at a time.