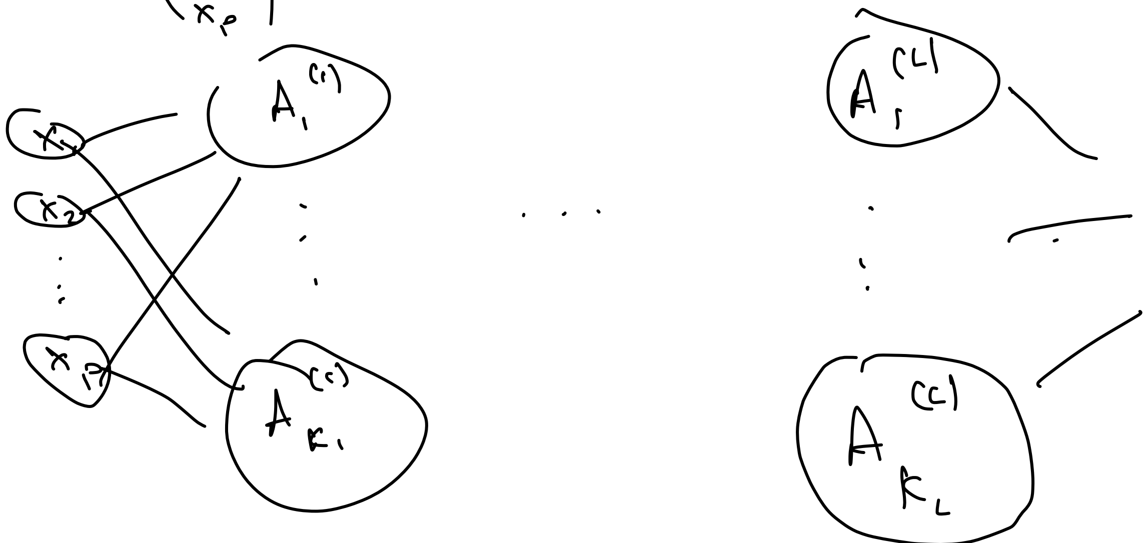
$$\underline{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{pmatrix} \qquad Y = \underline{f(\underline{x})} + \underline{\varepsilon}$$



$$f(\underline{x})$$
$$\parallel$$
$$f(\underline{x}) = \beta_0 + \sum_{k=1}^{K_L} \beta_k A_k^{(L)}$$

$$A_k^{(1)} = g\left(w_{k0} + \sum_{j=1}^{p} w_{kj} x_j\right)$$

g activation fcn (ReLU)

10.4 Convolutional Neural Networks (CNNs)

CNNs are state-of the art neural networks models for image data that exploits the spatial structure of images.

<u>Data</u> (raster image, not vector graphics)

8

pixel

6

1	5	50	52	1	2		
7	6	51					
110	112	26					

In grey scale image, typically represent pixel values as

0, 1, 2, …, 255

black

white

$\boxed{\text{Ex}}$ 3x3 filter

Apply a 3x3 convolutional filter, of form:

data matrix (for one channel)

$$\begin{pmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

First entry of resulting matrix is sum of

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix} \odot \begin{pmatrix} a & b & c \\ e & f & g \\ i & j & k \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & f & g \\ 0 & 0 & k \end{pmatrix} \rightarrow f+g+k$$

element-wise product

Convolved matrix is:

$$\begin{pmatrix} f+g+k & g+h+l \\ j+k+o & k+l+p \end{pmatrix}$$

Examples of 3x3 convolutional filters

$$\begin{pmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \qquad \begin{pmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \end{pmatrix}$$

local averaging or blurring of image

estimating a derivative in x-direction $\longrightarrow$ y-direction

Which convolutions would be useful for image data?

Let the data inform them!

In CNNs, after a convolution step it is typical to reduce the dimension of that matrix through max pooling.

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ 6 & 3 & 4 & 7 \\ 10 & 5 & 8 & 2 \\ 9 & 8 & 11 & 2 \end{pmatrix} \longrightarrow \begin{pmatrix} 6 & 7 \\ 10 & 11 \end{pmatrix}$$

where we apply a 2x2 matrix that takes the maximal value of original matrix

$\boxed{\text{Ex}}$ If start with images of dim 32x32 + do a 3x3 convolutional filter, and a 2x2 maxpooling step, we would matrices that go:

$$(32 \times 32) \rightarrow (30 \times 30) \rightarrow (15 \times 15)$$

(note maxpooling halves dim of matrix)