

Step I $\delta[n] \rightarrow \boxed{\text{LSI}} \rightarrow h[n]$ "Impulse Response"

Step II $\delta[n-k] \rightarrow \boxed{\text{LSI}} \rightarrow h[n-k]$

delayed the impulse by 'k' units

(Shift Invariance)

i.e., impulse is not at $n=0$, but at k

Step III $x[k] \delta[n-k] \rightarrow \boxed{\text{LSI}} \rightarrow x[k] h[n-k]$

The impulse at k is no longer a unit impulse, but has a magnitude $x[k]$

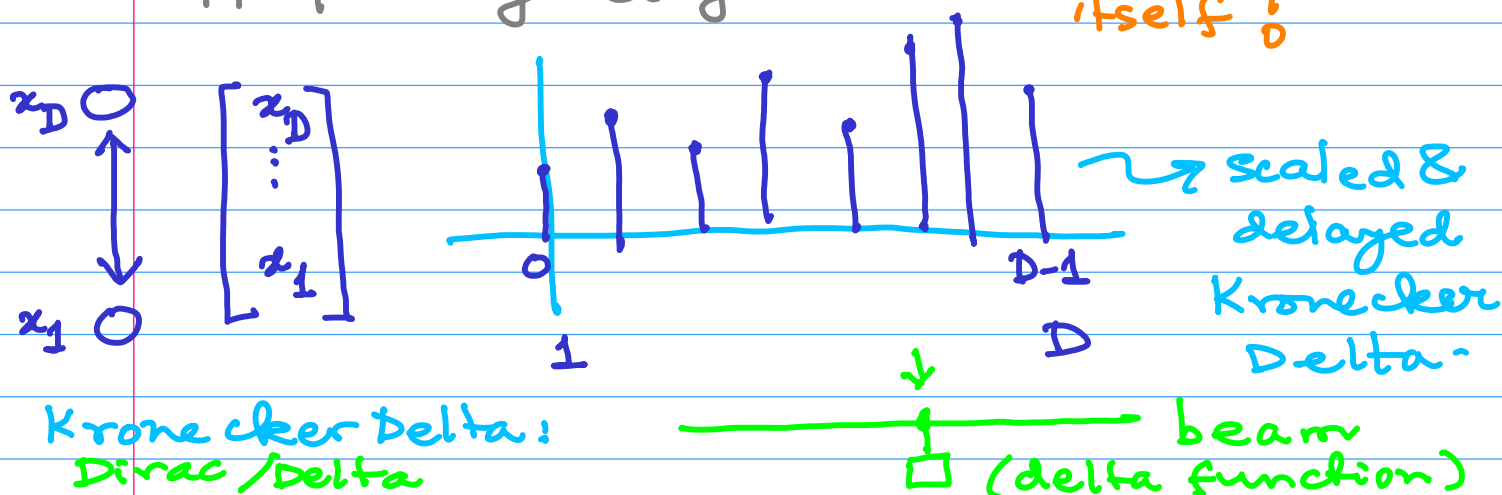
(Homogeneity)

Step IV $\sum_k x[k] \delta[n-k] \rightarrow \boxed{\text{LSI}} \rightarrow \sum_k x[k] h[n-k]$

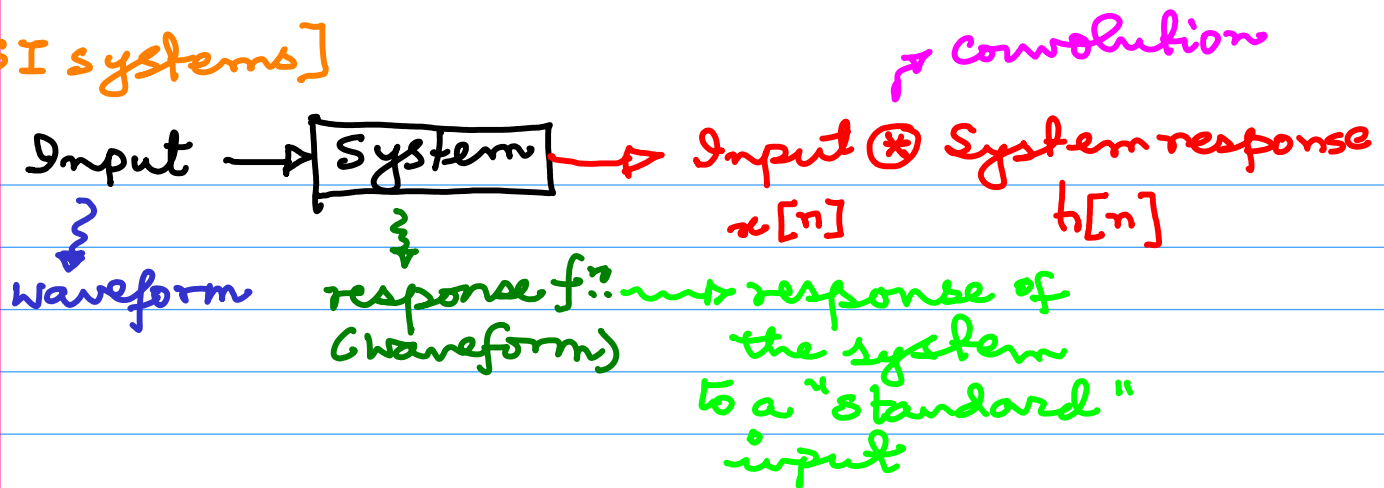
This is the input waveform itself! A collection of scaled unit impulses which are appropriately delayed.

(Additivity)

→ this is the definition of Convolution itself!



[LSI systems]



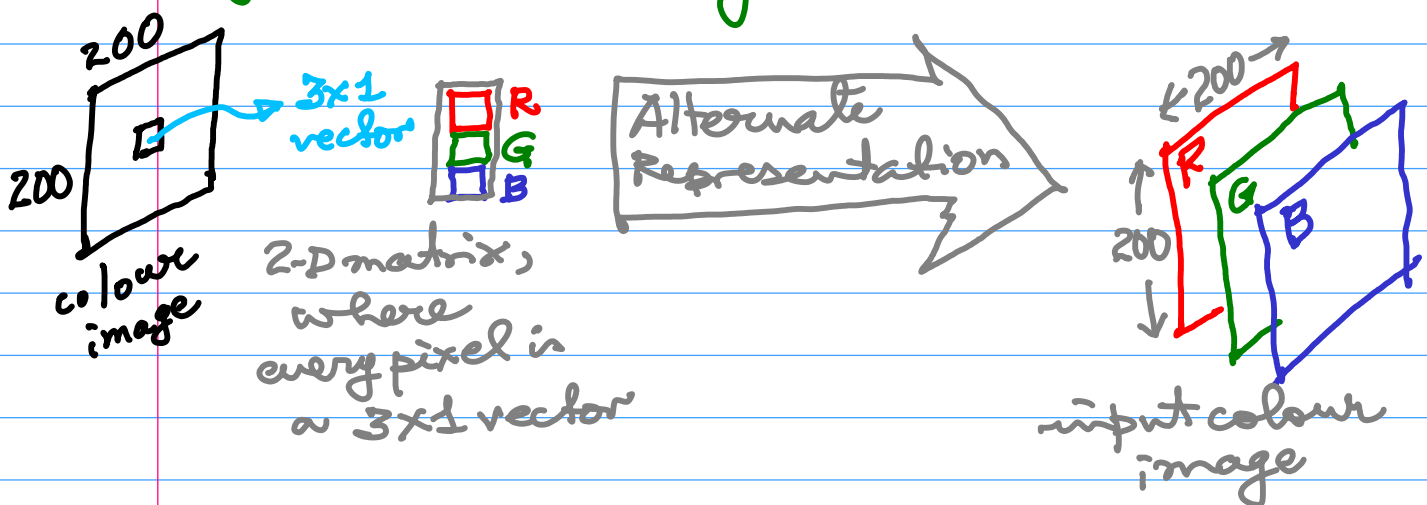
Motivation for CNNs (Simple Deep NNs)

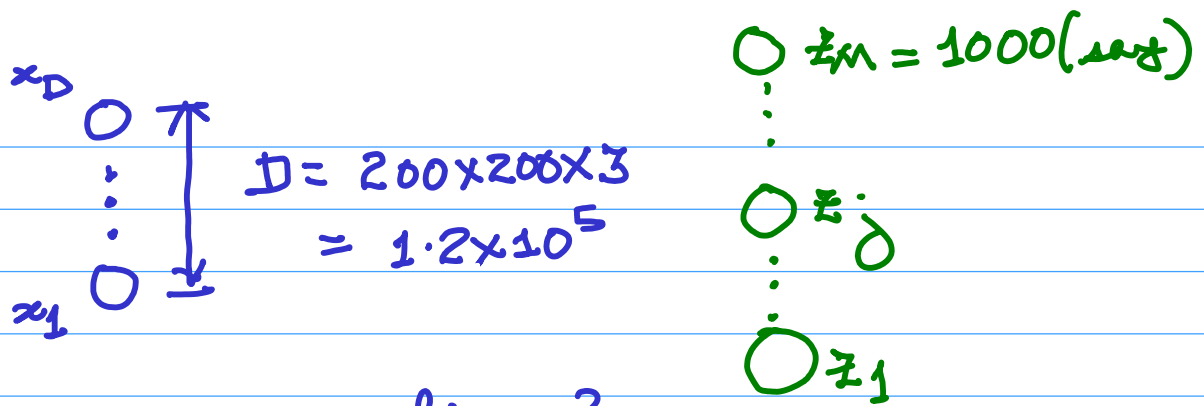
(*) Feedforward MLP: Too many connections (each neuron to every neuron: between layers)

(*) Images/Videos (2-D/3-D): inputs \rightarrow vectorise them \rightarrow discards spatial connectivity/neighbourhood information \rightarrow Need for a structure which can naturally account for peculiarities of 2-D inputs/ 3-D inputs/...

with a 'reasonable' complexity.

Images (2-D) are very common \rightarrow LeNet (1989)





How many connections?

Each hidden layer neuron z_j is connected to 1.2×10^5 neurons
 $\equiv 1.2 \times 10^5$ weights + 1 bias term

Even if we forget the bias term,

of parameters $\approx 1.2 \times 10^5 \times 10^3$ weights
 $= 1.2 \times 10^8$ weights.

→ Too many parameters,
 chances of overfitting

(*) Invariance Issues: we often look for certain patterns in 1-D inputs (e.g., doctors looking at ECGs) or 2-D inputs (e.g., doctors looking at X-ray images or USG)

These patterns can be anywhere in the 1-D waveform / input / signal, or a 2-D image

Simplest Invariance issue: translation

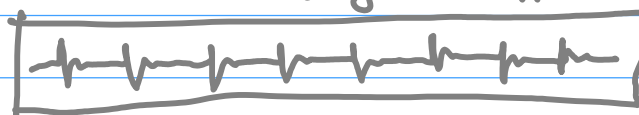
Others: rotations, scaled, shear, projective effects (2-D)

["Shift Invariance"] → comes naturally

ECG



an abnormality can appear anywhere!



ECG print

Euclidean transform

2-D transformations (Simplest possible)

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\underline{x}' = R \underline{x} + \underline{t}$$

rotation

translation

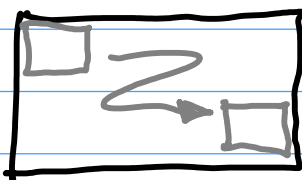
resilience to translated patterns of interest in 1-D/2-D/... inputs is usually built into the CNN structure itself

(for other transformations → heuristics e.g.

- feed transformed variants into the NN & hope that it learns them
- Regularisation (penalise a NN which doesn't account for these)
- Pre-processing the input, & feeding the processed input to the NN)

2-D example: "Where's Wally/Where's Waldo"

□ "Wally/waldo detector" → sweep the entire image with this detector (correlation score over a square patch of the input image) → how likely is Wally/waldo present in this part of the image



→ convolution / shift invariance

Usually, we consider two forms of convolution:

(1) "wherever fits"

$$\begin{bmatrix} 1 & -1 & 2 & 3 \end{bmatrix} * \begin{bmatrix} 1 & -1 \end{bmatrix}$$

← A → ← B →

↓ flip

$$\begin{bmatrix} -1 & 1 \end{bmatrix}$$

Also called

"Zero padding"

$O \begin{bmatrix} 1 & -1 & 2 & 3 \end{bmatrix} O$

$\begin{bmatrix} 1 & 1 \end{bmatrix} \rightarrow 1$
 $\begin{bmatrix} -1 & 1 \end{bmatrix} \rightarrow -2$
 $\begin{bmatrix} 1 & 1 \end{bmatrix} \rightarrow 3$
 $\begin{bmatrix} -1 & 1 \end{bmatrix} \rightarrow -5$
 $\begin{bmatrix} -1 & 1 \end{bmatrix} \rightarrow 3$

longer than the
 $[1 \ -2 \ 3 \ -5 \ 3]$
 larger array
 input

$$\text{size} = A + B - 1$$

$$= 4 + 2 - 1 = 5$$

(2) "Perfect fit" (generally used in NNs / image processing)

$$\begin{bmatrix} 1 & -1 & 2 & -3 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \rightarrow -2$$

$$\begin{bmatrix} -1 & 1 \end{bmatrix} \rightarrow 3$$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \rightarrow -5$$

Ans: $A - B + 1$

$$[-2 \quad 3 \quad -5]$$

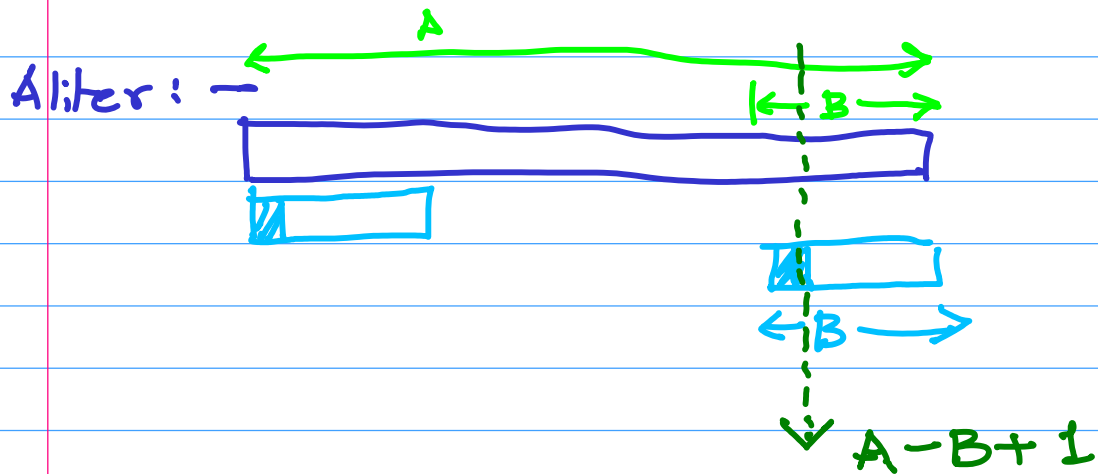
shorter than the
larger array

logic \rightarrow the previous 'wherever fits' strategy
has $(B-1)$ more overlaps to the left &
 $(B-1)$ more to the right

$$\text{perfect fit} + (B-1) + (B-1) = A + B - 1$$

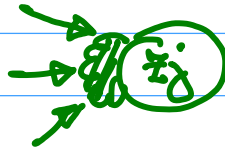
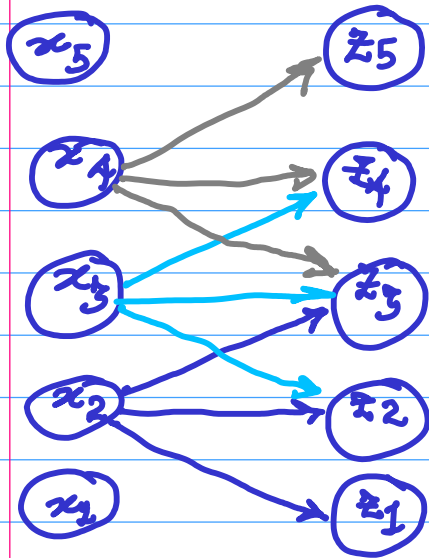
$$\Rightarrow \text{perfect fit} = A + B - 1 - 2B + 2$$

$$= A - B + 1$$



Characteristics of Deep Networks

① Local Receptive fields (Sparse connections)



"receptive field of z_j "

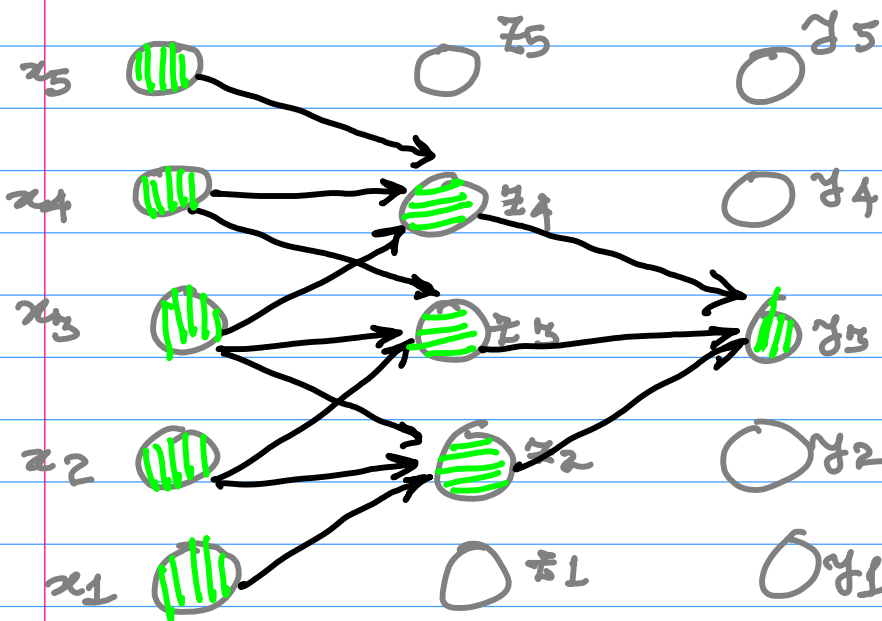
Complexity: connectivity
is now $O(M)$

M : # of neurons in the hidden layer

in place of the general MLP case of $O(DM)$

D : # of neurons in the input layer

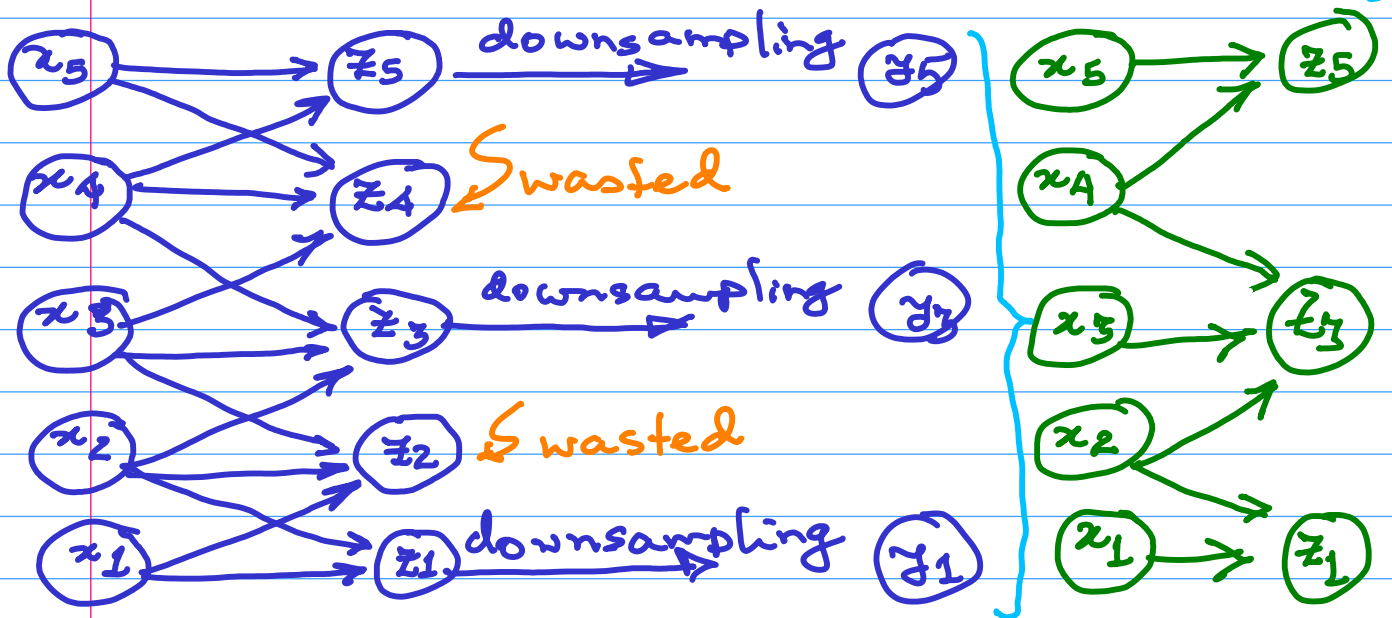
* The sparsity is not really that restrictive!



y_3 interacts indirectly with 5 in the input layer!

- Units in deeper layers interact **indirectly** with a larger portion of the input
- * Efficient representation of complicated interactions among many variables, by using simple building blocks which have sparse interactions.

(2) 'Strided' Convolutions



'stride' = 2 in this case mathematically equivalent but less wasteful

(3) Parameter Sharing

