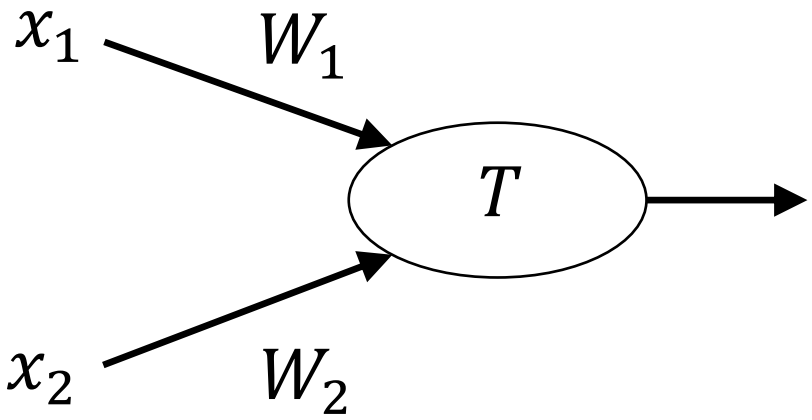
Neural Networks

Chetan Arora

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

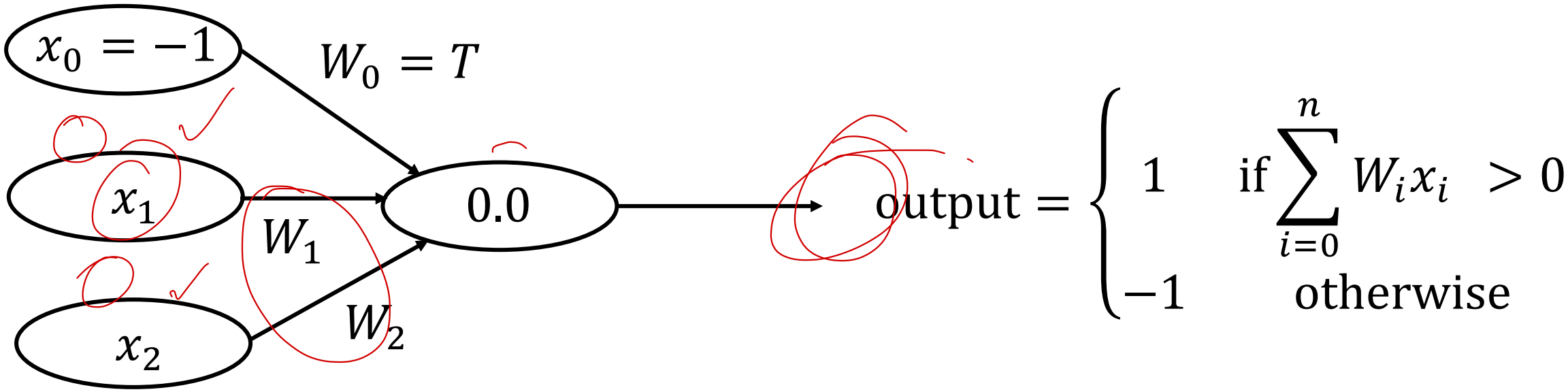
Simple network (AND)



$$\text{output} = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} W_i x_i > T \\ -1 & \text{otherwise} \end{cases}$$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

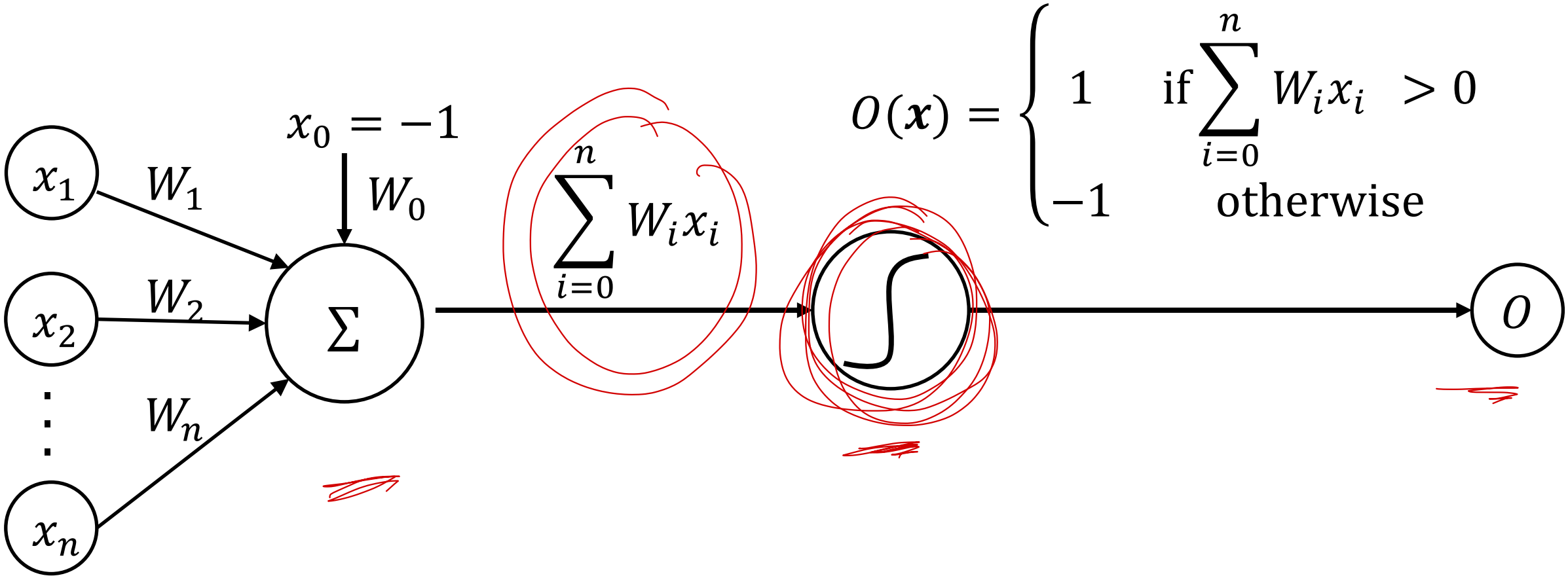
Simple network (AND)



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

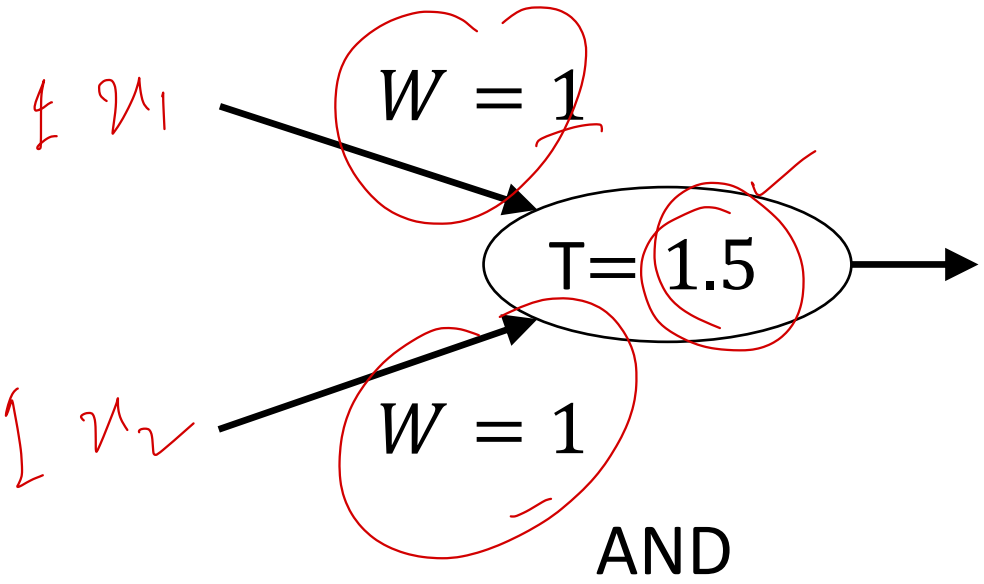
Perceptron

- Linear treshold unit (LTU)



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

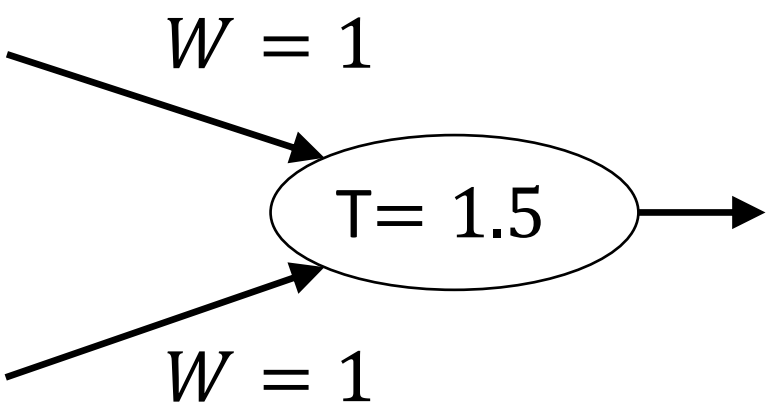
Perceptron



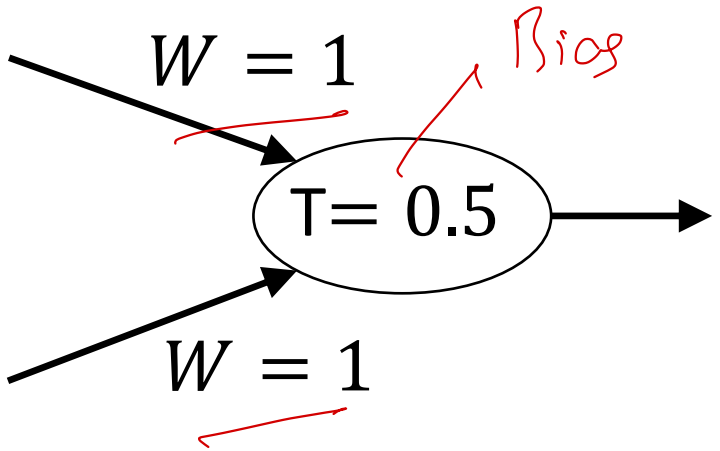
AND

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

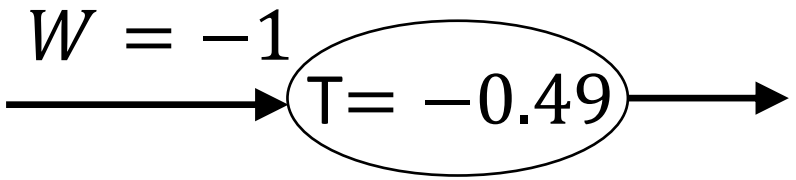
Perceptron



AND



OR



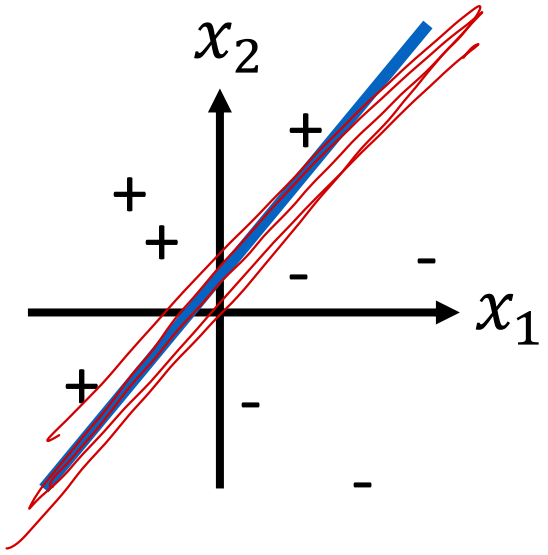
NOT

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

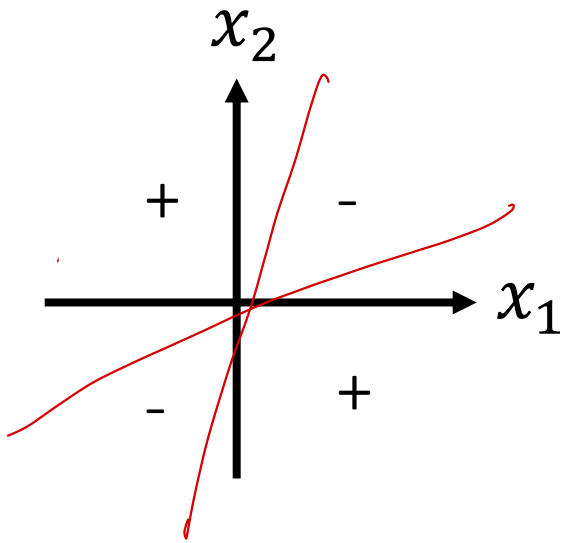
Decision Surface of a Perceptron

- Perceptron can represent many useful functions: AND, OR, NOT etc.

- But functions that are not linearly separable (e.g. XOR) are not representable.



Linearly separable

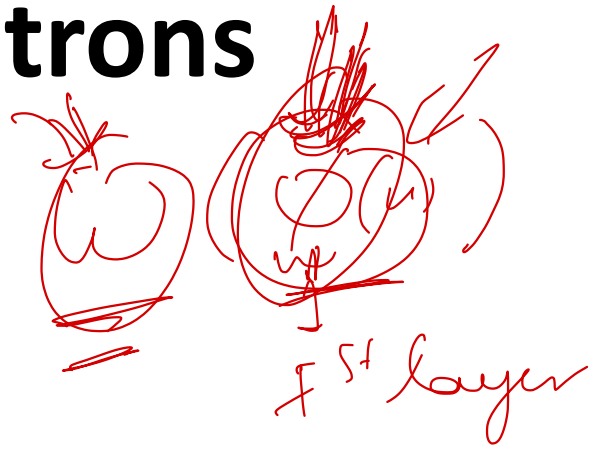


Non-Linearly separable

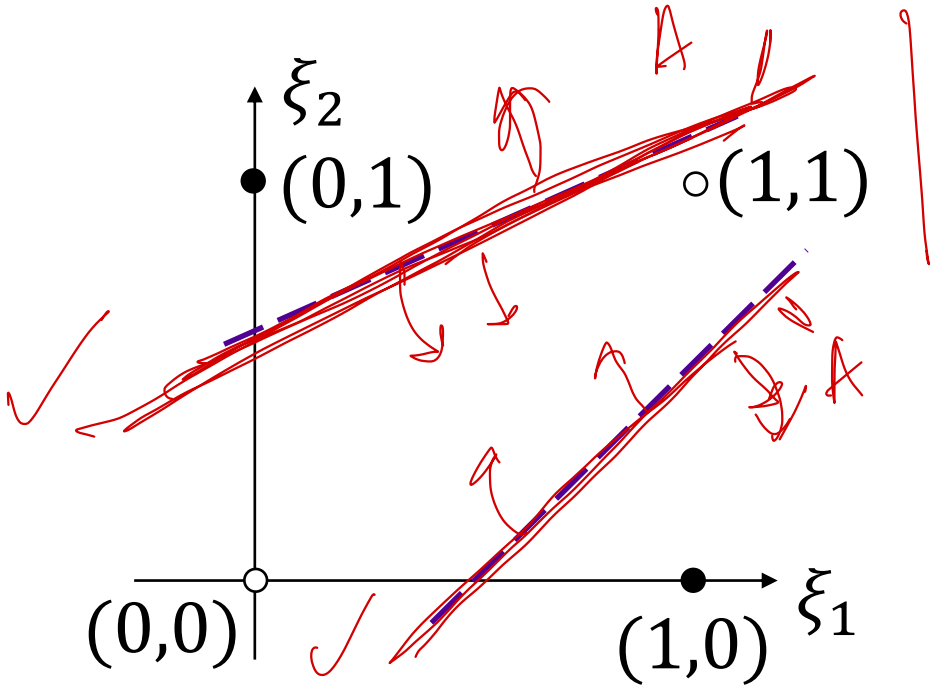
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Solution in 1980s: Multilayer Perceptrons

- Removes many limitations of single-layer networks
  - Can solve XOR



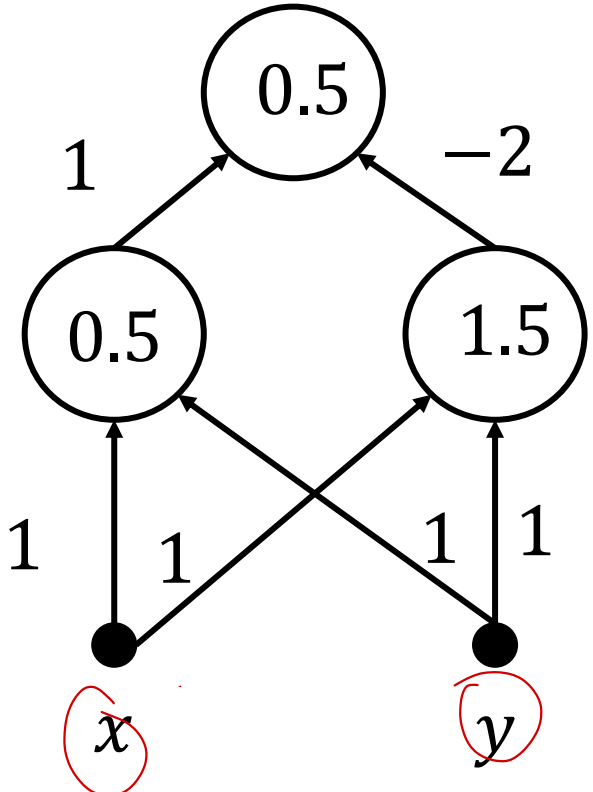
- Exercise: Draw a two-layer perceptron that computes the XOR function
  - 2 binary inputs ξ_1 and ξ_2
  - 1 binary output
  - One "hidden" layer
  - Find the appropriate weights and threshold

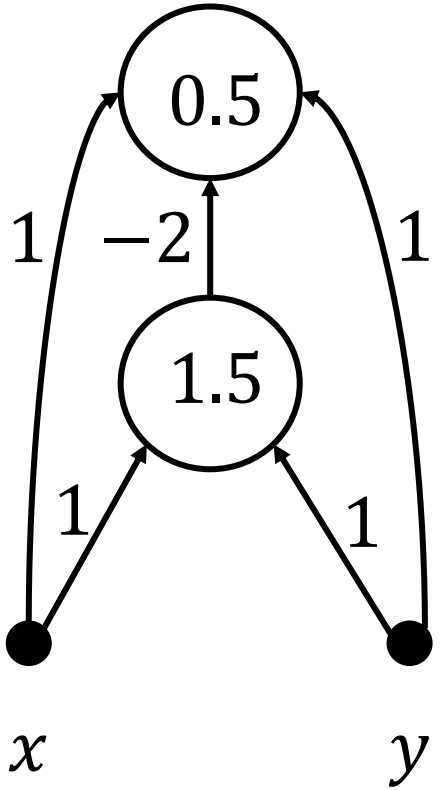


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Solution in 1980s: Multilayer Perceptrons

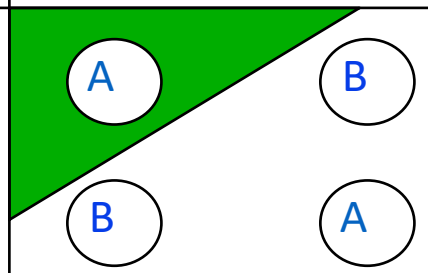
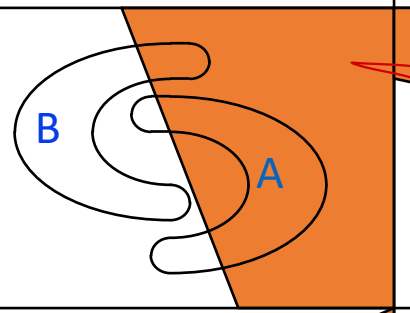
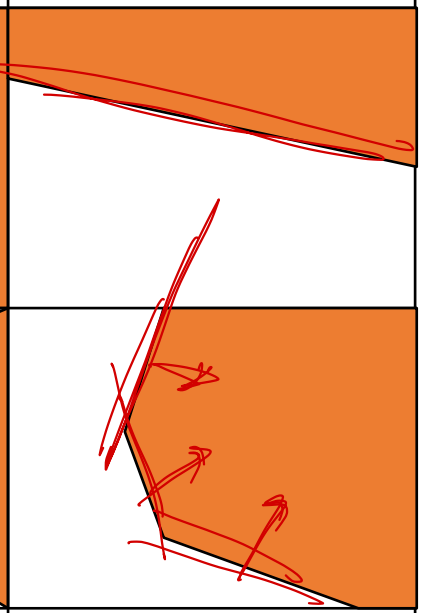
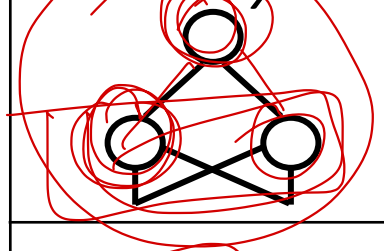
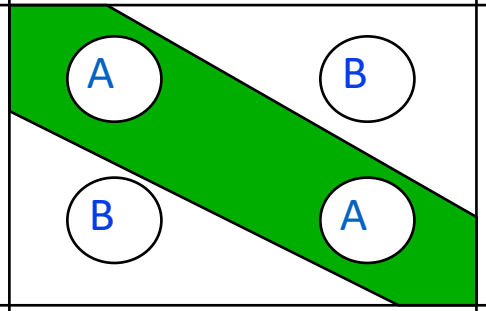
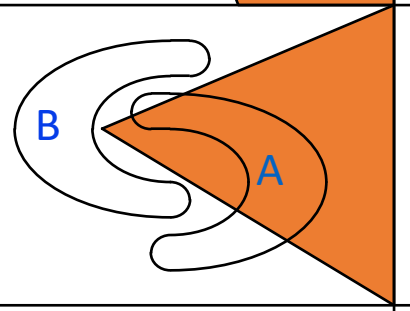
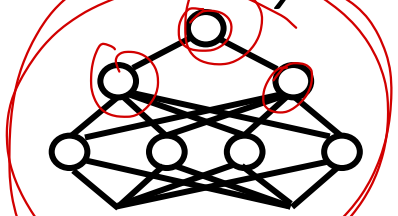
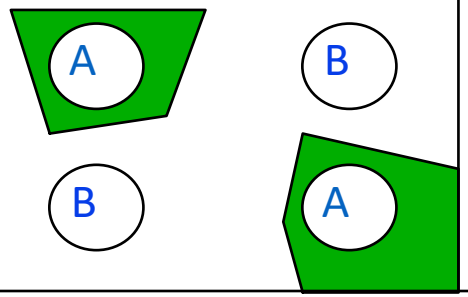
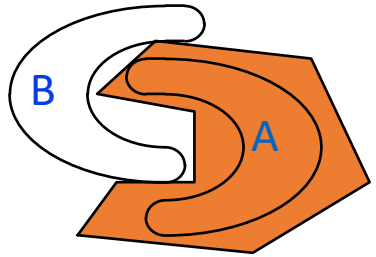
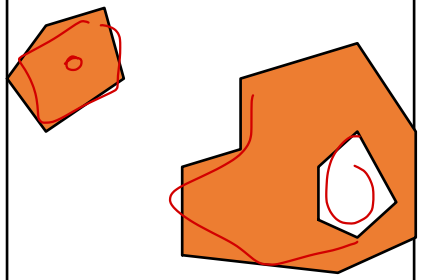
- Examples of two-layer perceptrons that compute XOR
- E.g. Right side network
  - Output is 1 if and only if $x + y - 2(x + y - 1.5 > 0) - 0.5 > 0$

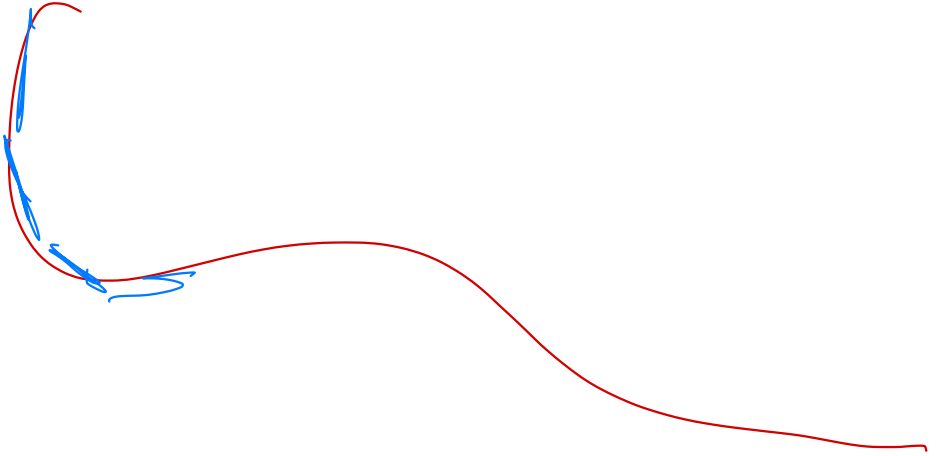




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Different Non-Linearly Separable Problems

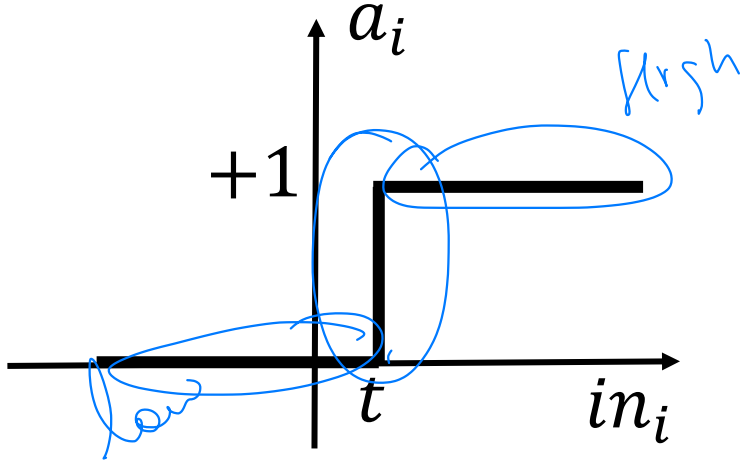
Structure	*Types of Decision Regions*	*Exclusive-OR Problem*	*Classes with Meshed regions*	*Most General Region Shapes*
Single-Layer 	*Half Plane Bounded By Hyperplane*			
Two-Layer 	*Convex Open Or Closed Regions*			
Three-Layer 	Arbitrary (Complexity Limited by No. of Nodes)			



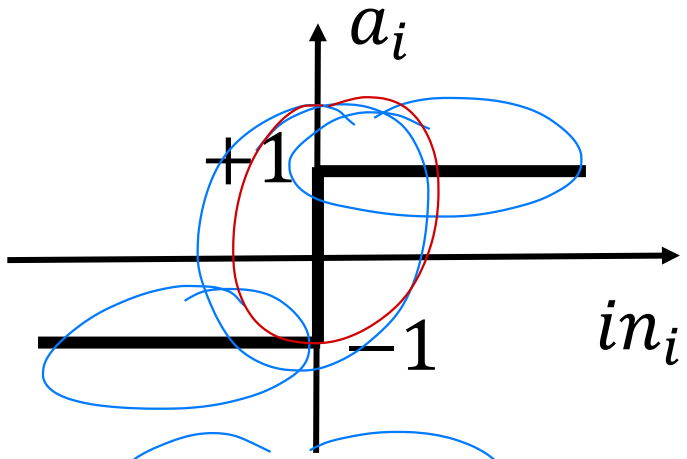
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Activation functions

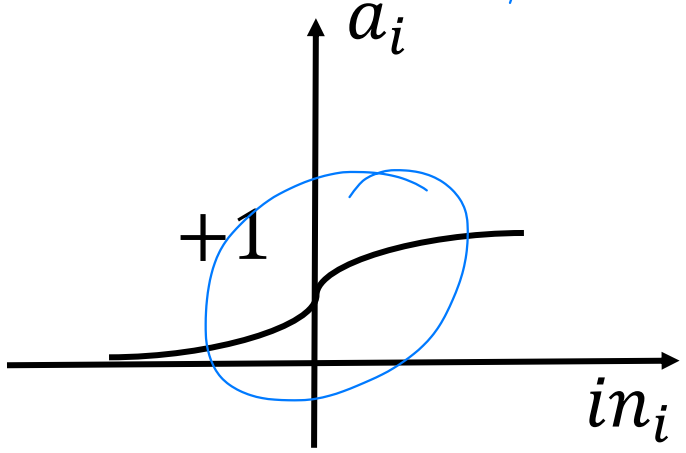
- Transforms neuron's input into output.
- Features of activation functions:
  - A squashing effect is required
    - Prevents accelerating growth of activation levels through the network.
  - Simple and easy to calculate



Step function



Sign function



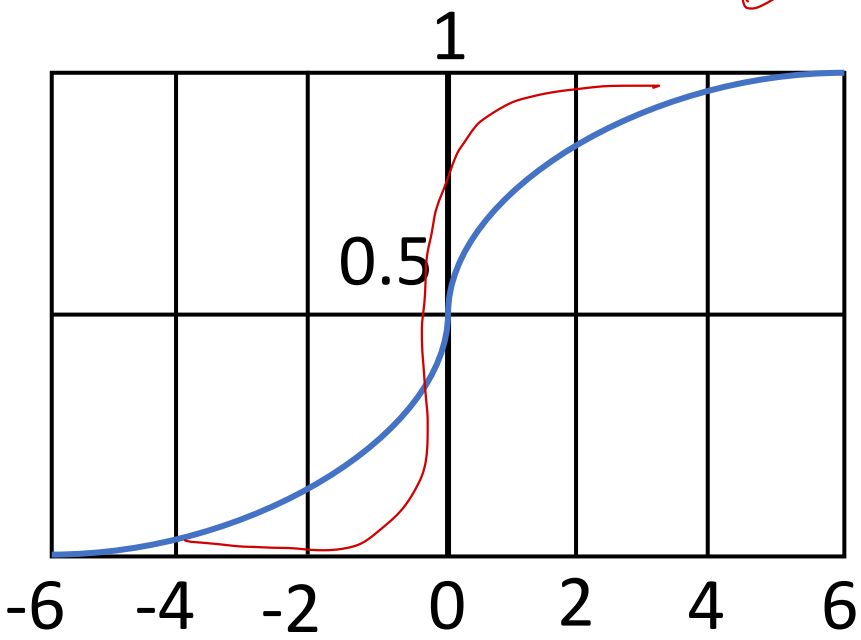
Sigmoid function

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Standard activation functions

- The hard-limiting threshold function
  - Corresponds to the biological paradigm
    - either fires or not
- Sigmoid functions ('S'-shaped curves)
  - The logistic function
  - The hyperbolic tangent (symmetrical)
  - Both functions have a simple differential
  - Only the shape is important

$$\phi(x) = \frac{1}{1 + e^{-x}}$$

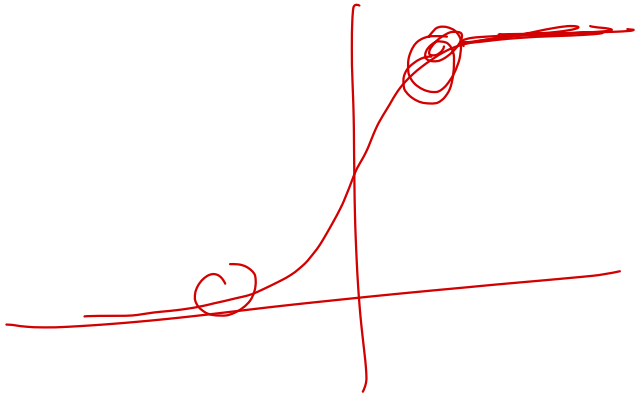


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Standard activation functions

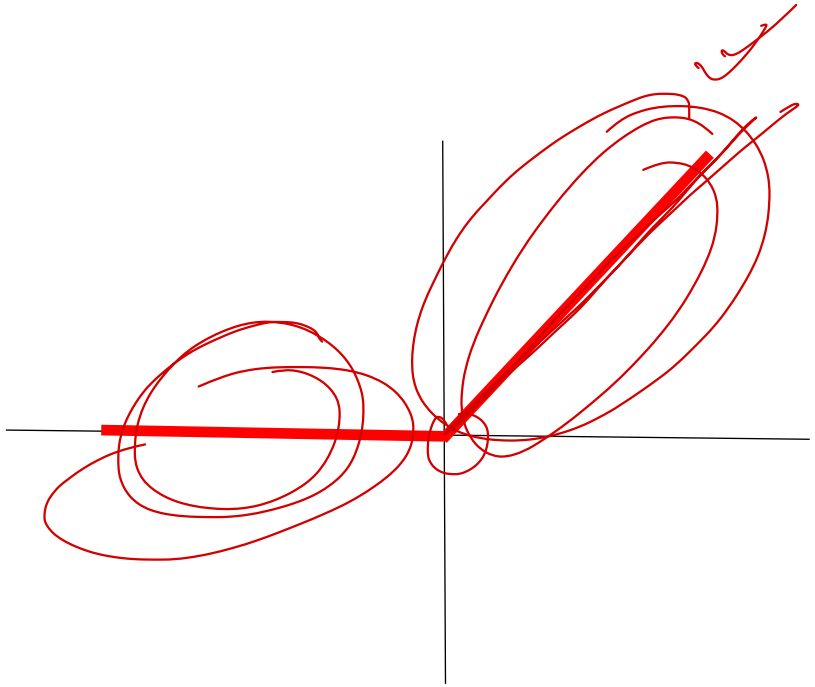
- tanh

$$\frac{\exp(z)-\exp(-z)}{\exp(z)+\exp(-z)}$$



- ReLU

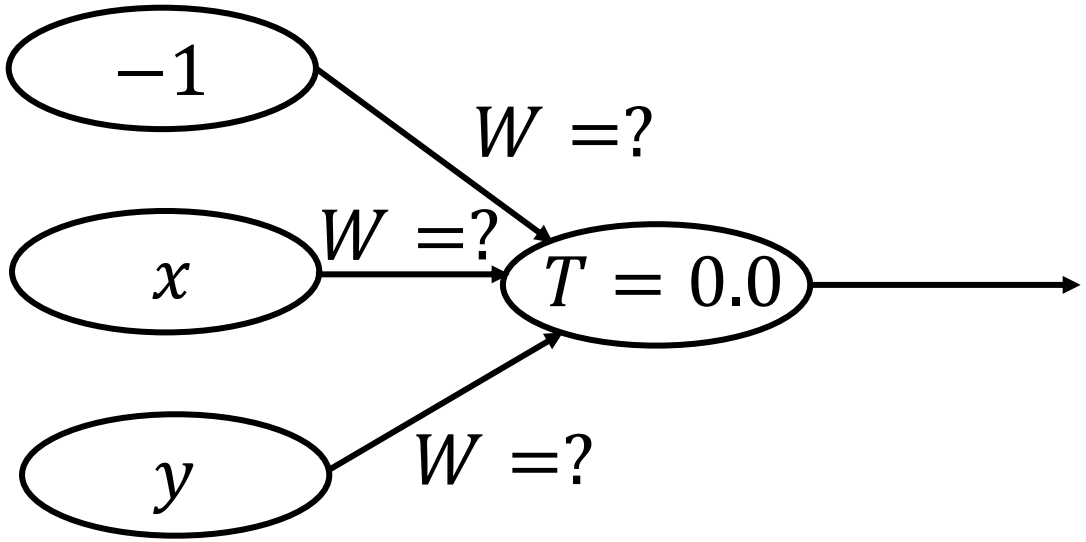
$$relu(z) = \max(0, z)$$



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Training Perceptrons

- What are the weight values?



For AND

A	B	Output
0	0	0
0	1	0
1	0	0
1	1	1


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Training Algorithms

- Perceptron Learning Algorithm: for linearly separable samples.

- Backpropagation: Coming soon.

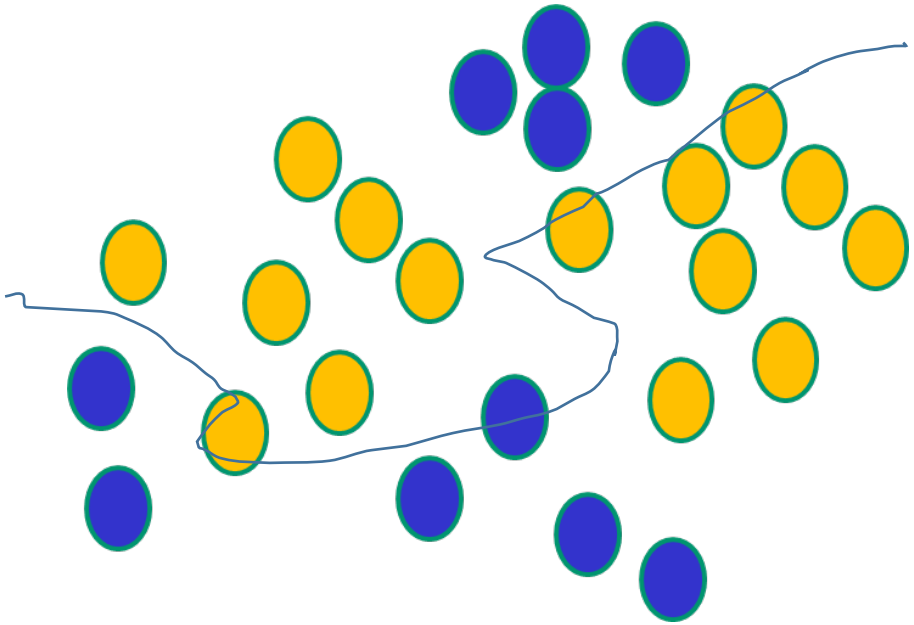
Neural Networks as Feature detectors

$w(\phi(x))$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Neural Networks: Two Perspectives

Use nonlinear $f(x)$ to draw complex boundaries, but keep the data unchanged

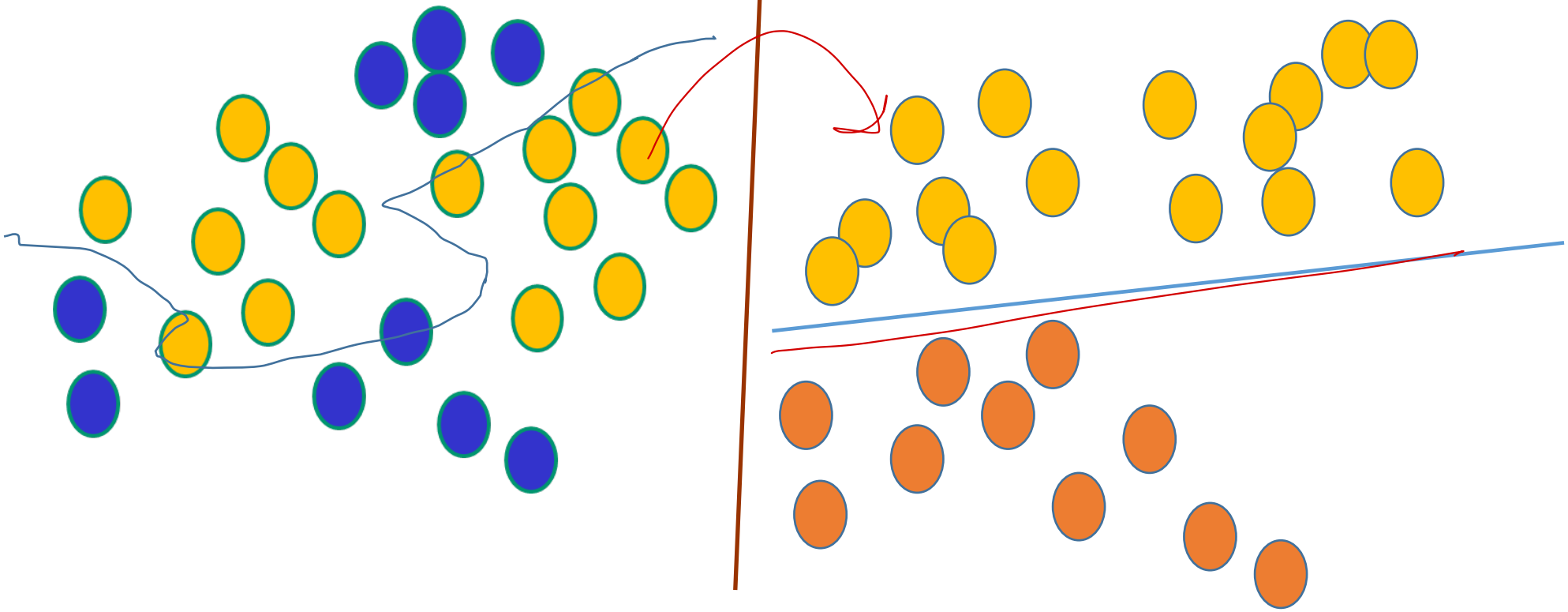


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Neural Networks: Two Perspectives

Use nonlinear $f(x)$ to draw complex boundaries, but keep the data unchanged

Find linear boundaries only, but transform the data first in a way that makes linear boundaries OK



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Neural Networks as Feature detectors

Handwritten images

MNIST

32x32

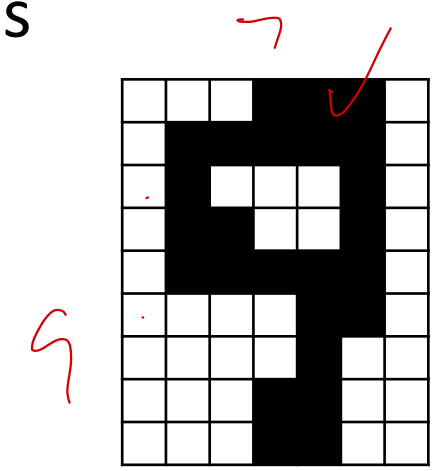
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Neural Networks as Feature detectors

Handwritten images

■ +1

□ -1



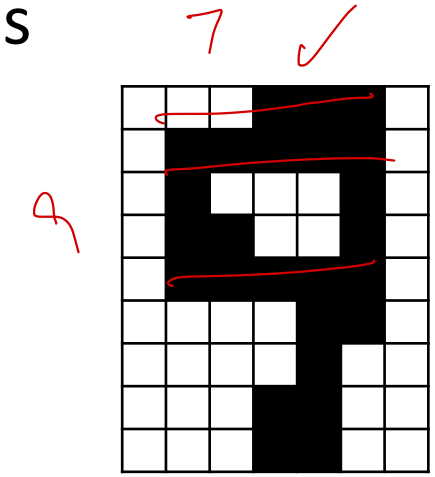
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

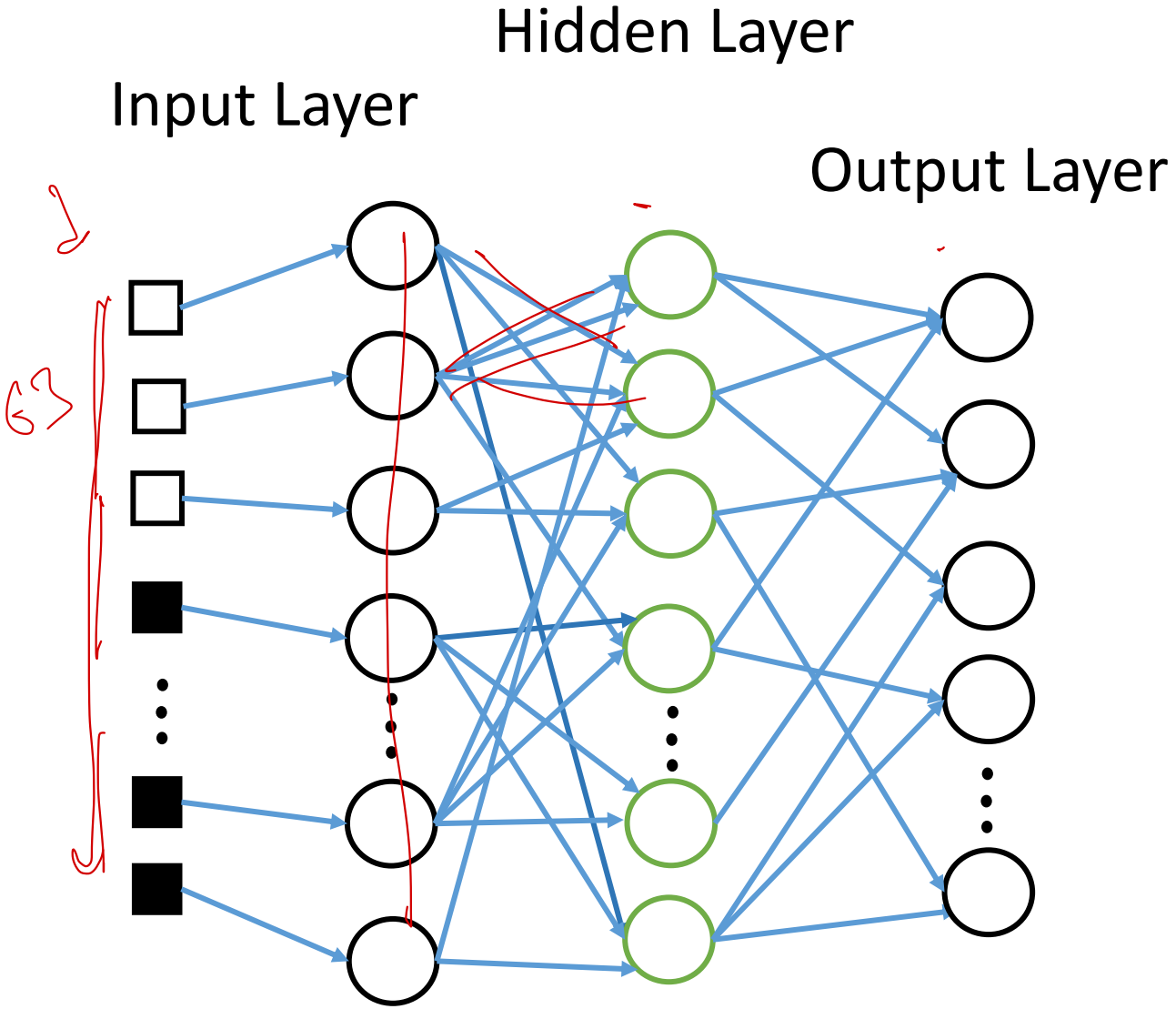
Neural Networks as Feature detectors

Handwritten images

■ +1

□ -1

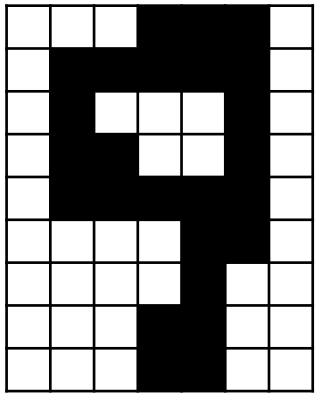


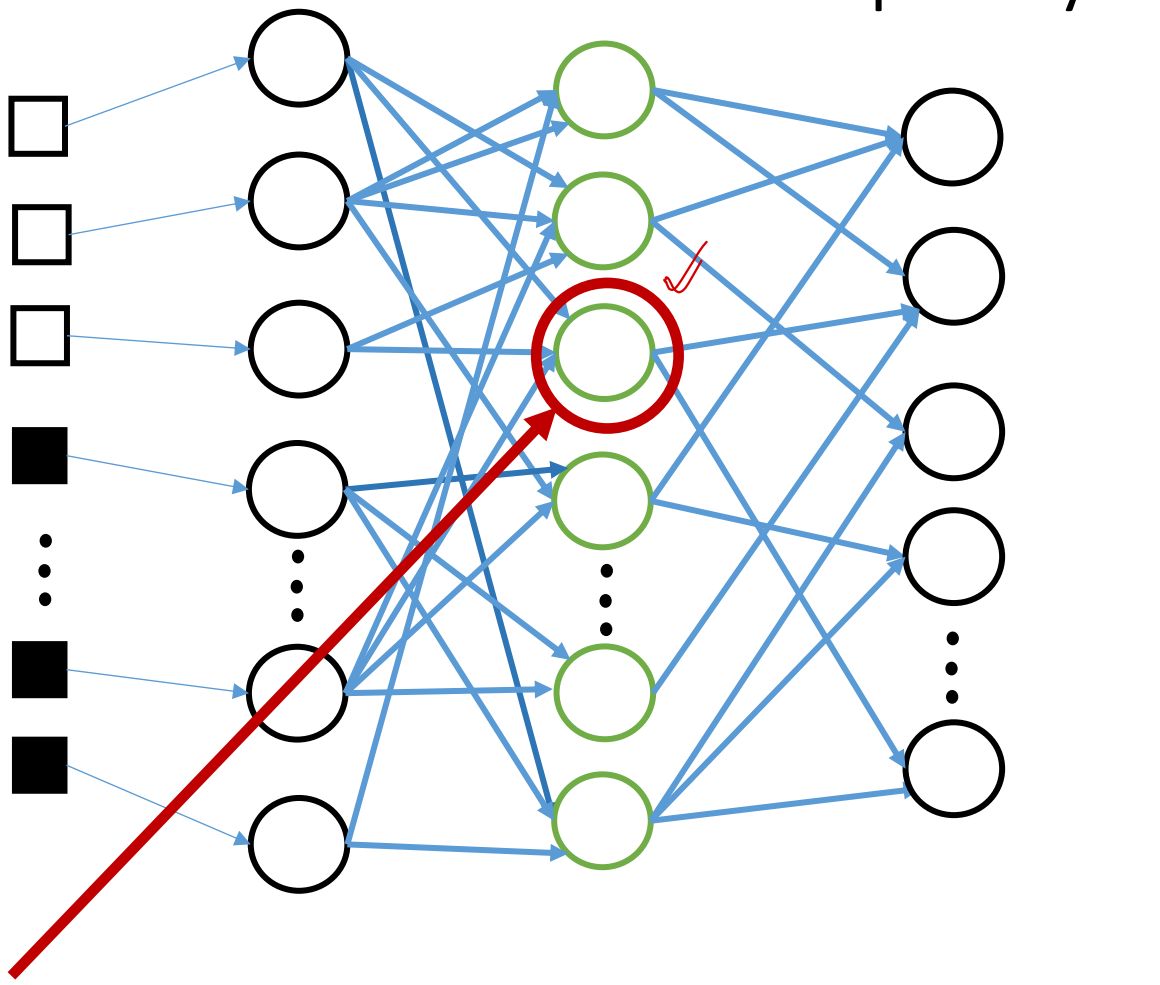


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What is this unit doing?

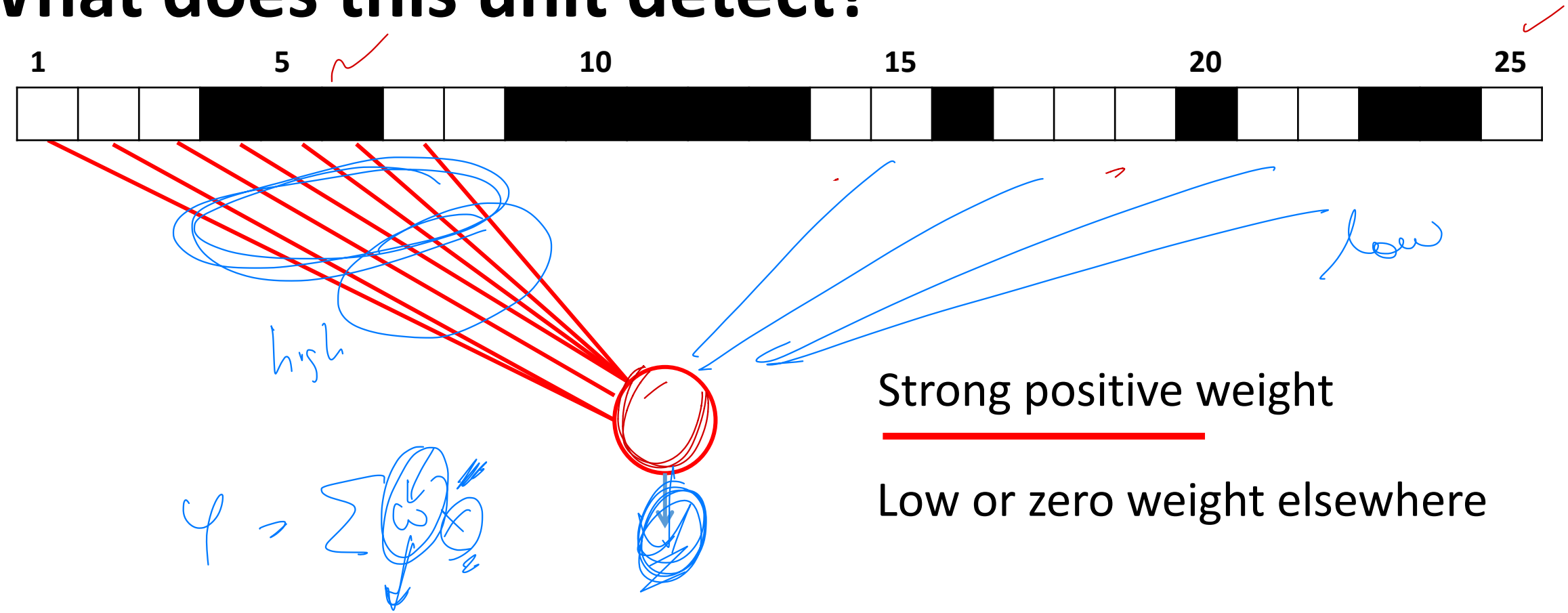
Handwritten images





Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?

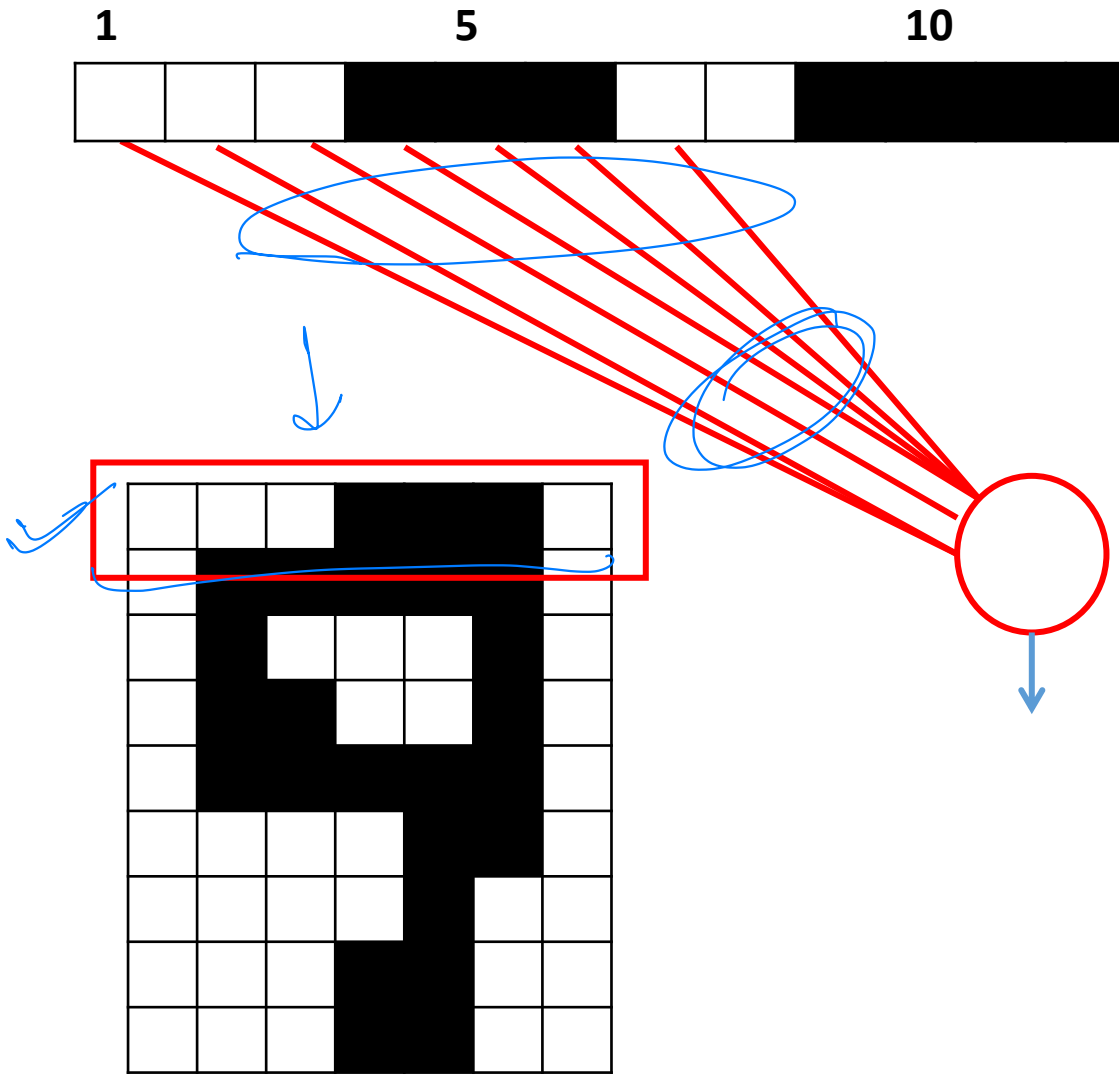


Strong positive weight

Low or zero weight elsewhere

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?

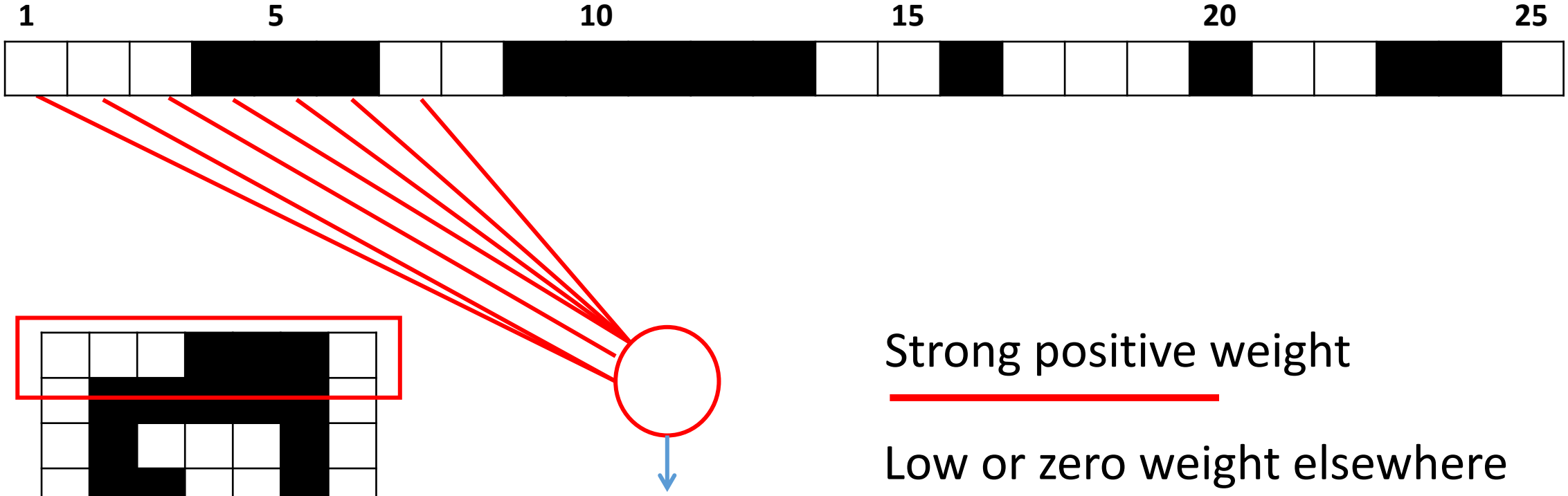


Strong positive weight

Low or zero weight elsewhere

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?



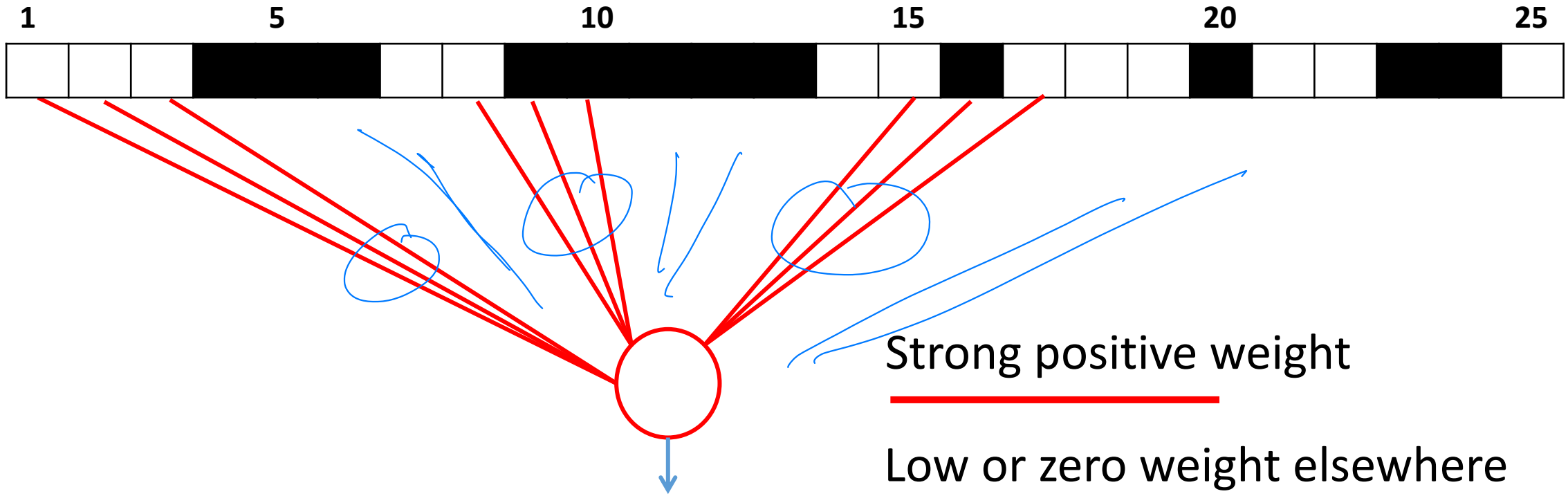
Strong positive weight

Low or zero weight elsewhere

it will send strong signal for a horizontal line in the top row, ignoring everywhere else

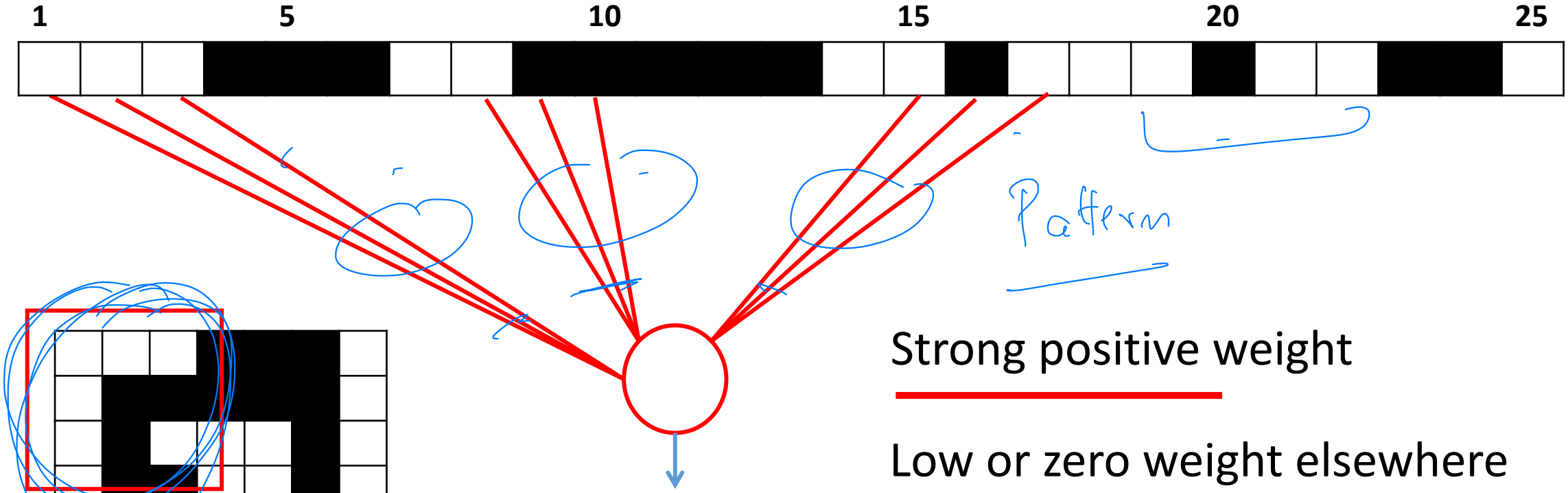
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?

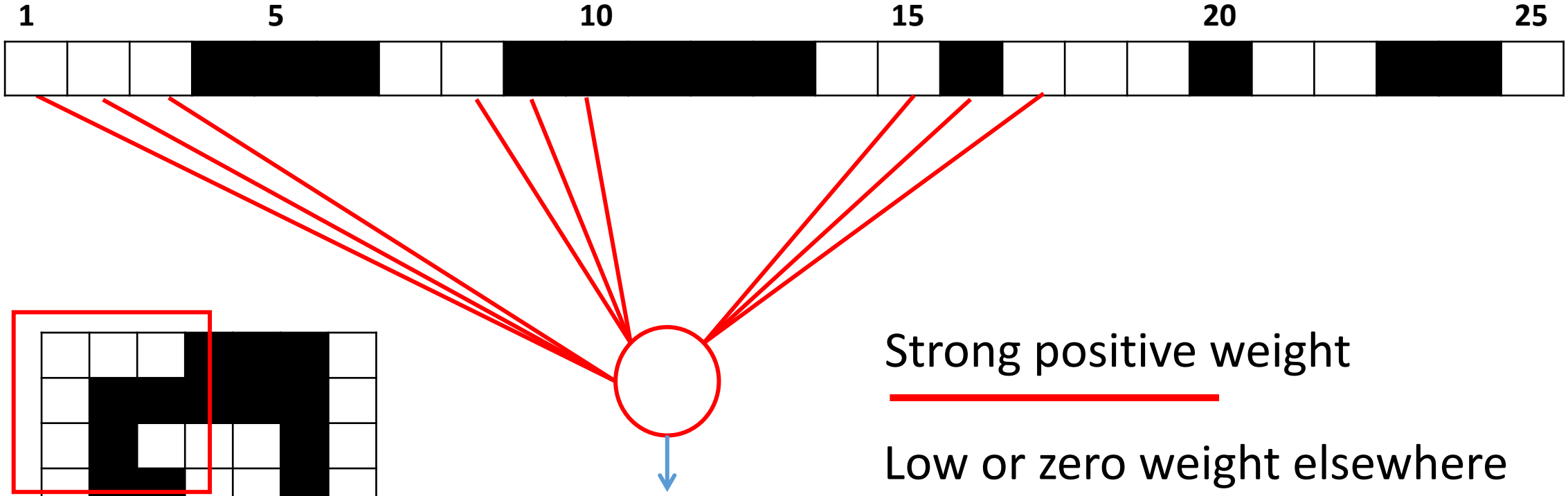


Strong positive weight

Low or zero weight elsewhere

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

What does this unit detect?



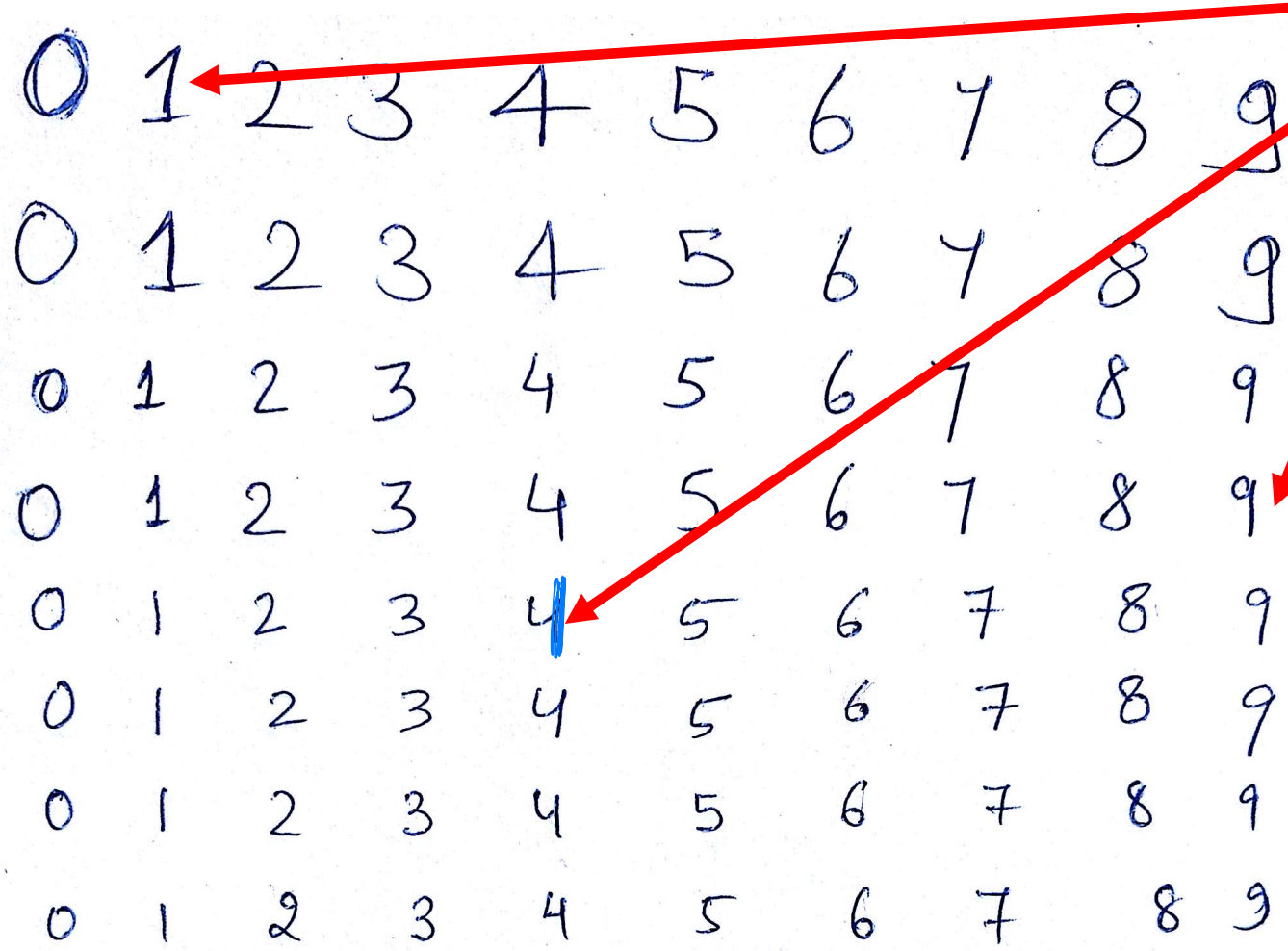
Strong positive weight

Low or zero weight elsewhere

Strong signal for a dark area in the top left corner

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

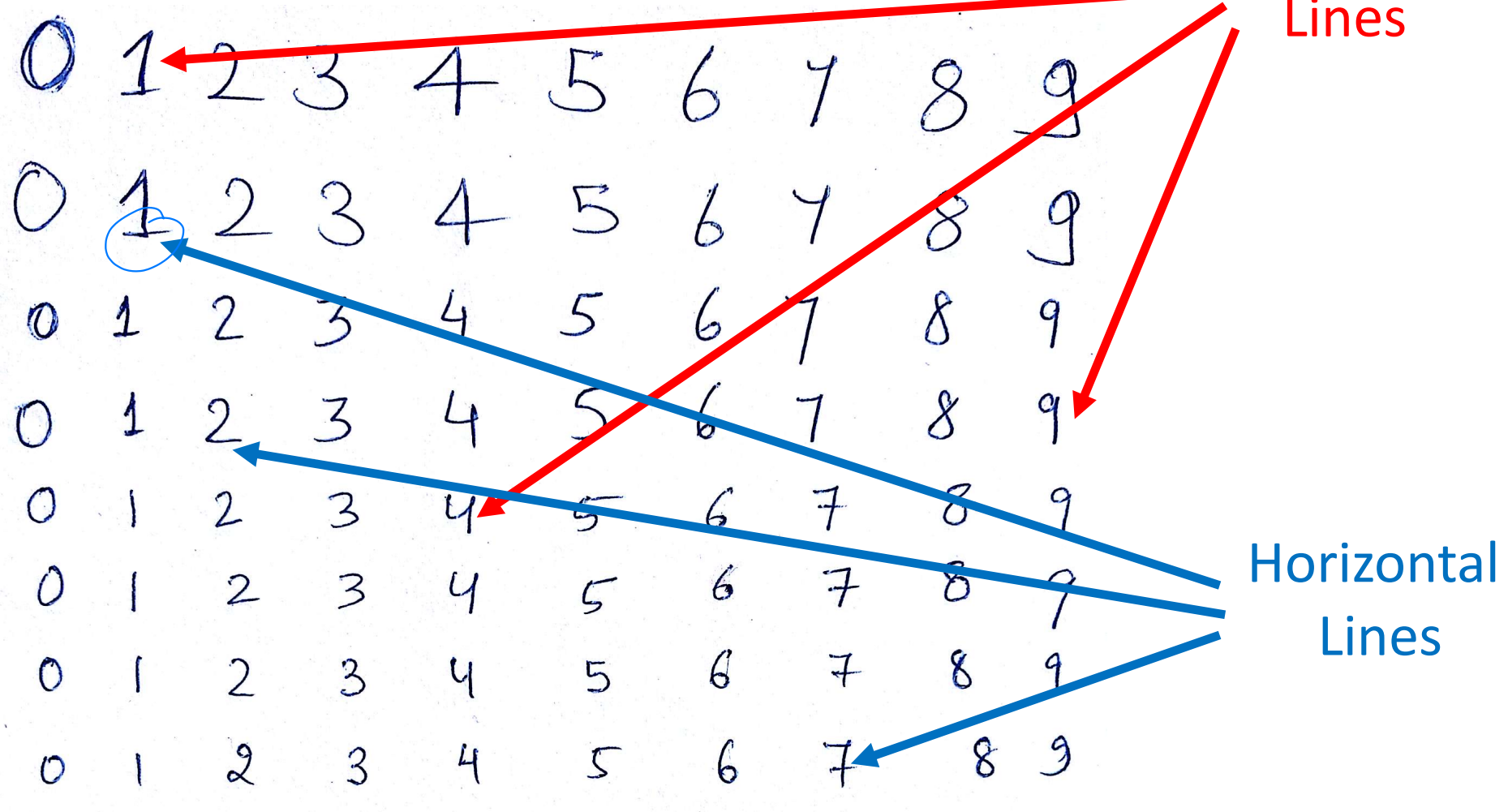
Perceptron as a Feature Detector



Handwritten images

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

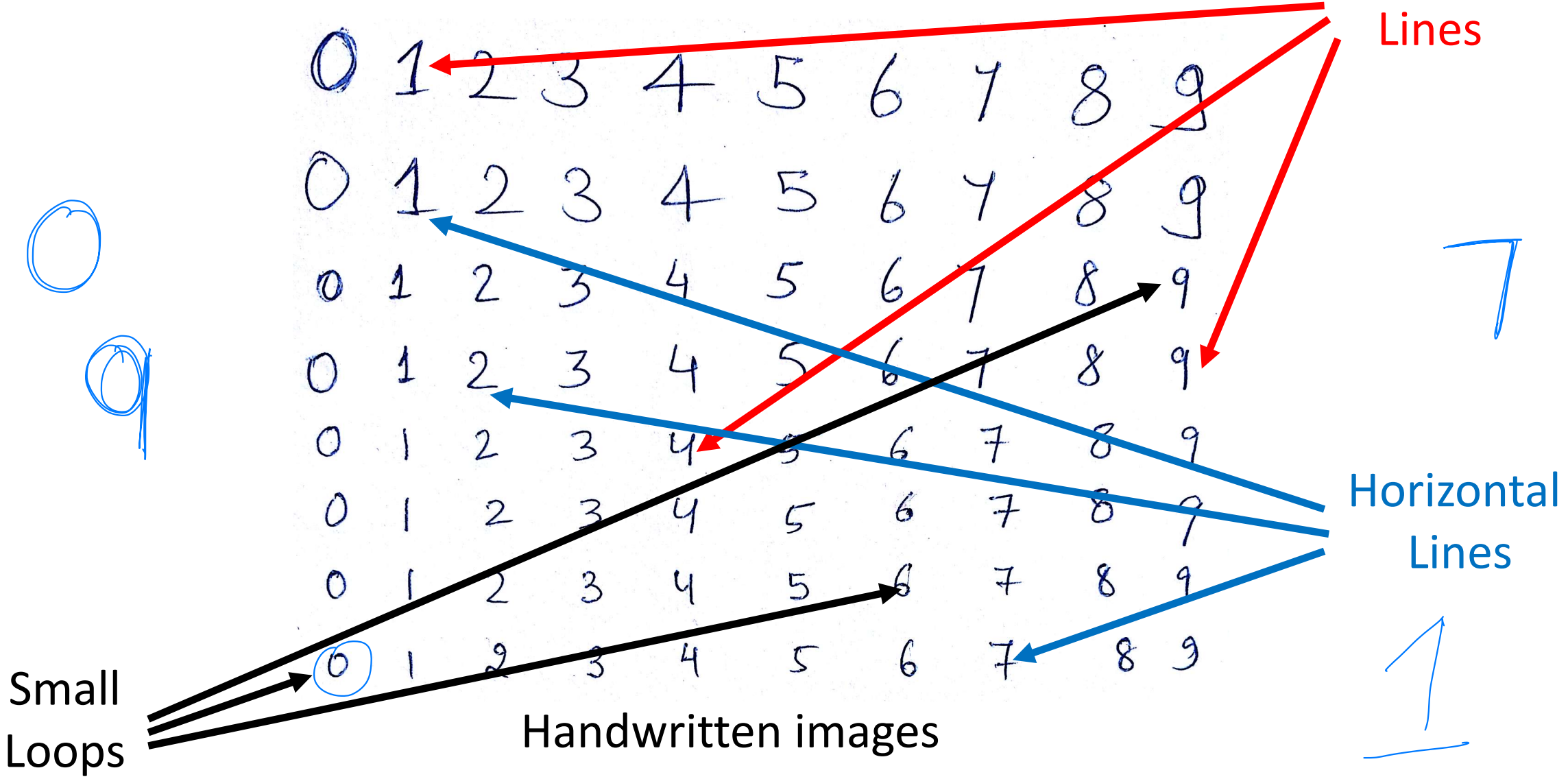
Perceptron as a Feature Detector



Handwritten images

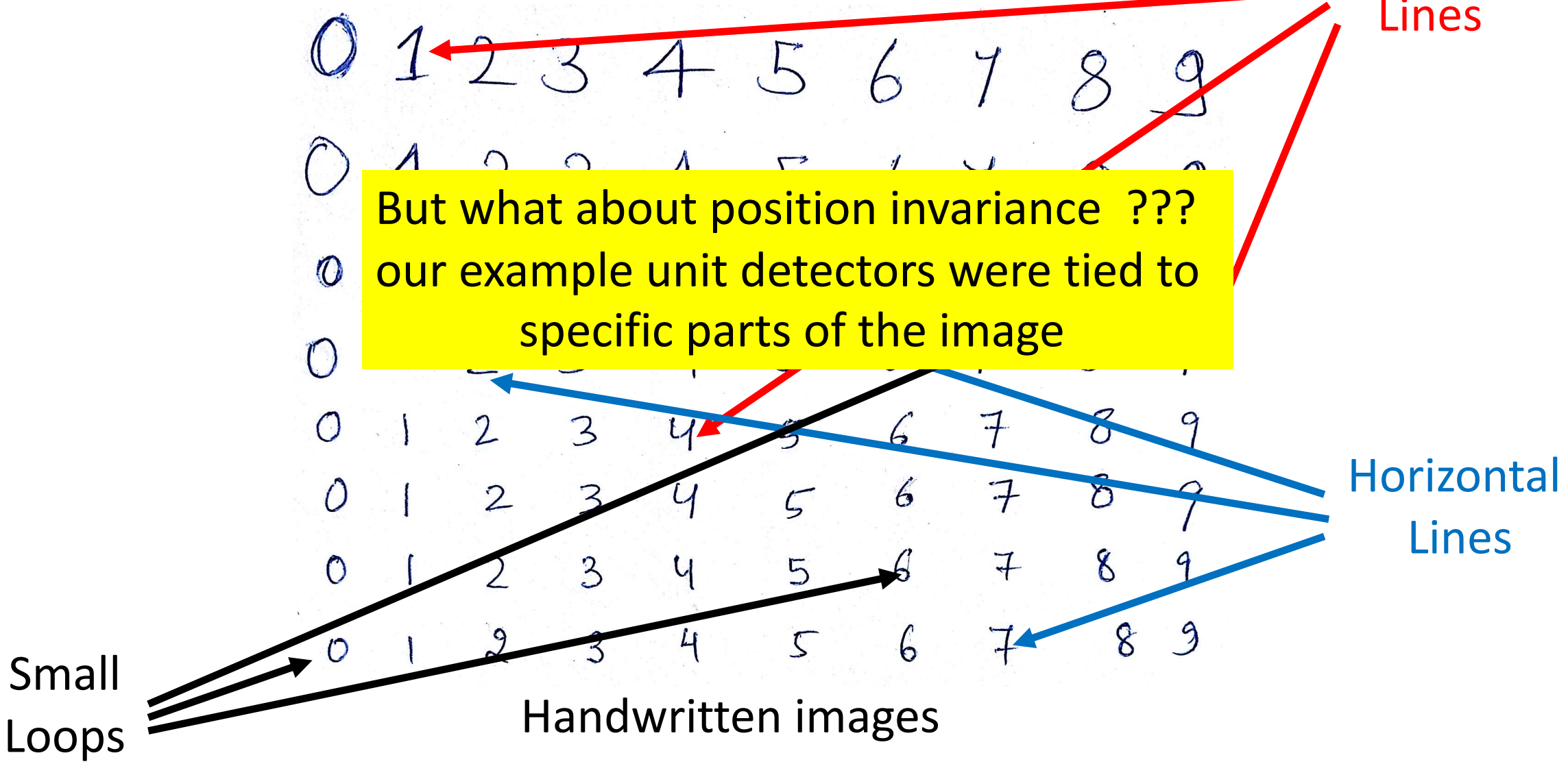
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Perceptron as a Feature Detector



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

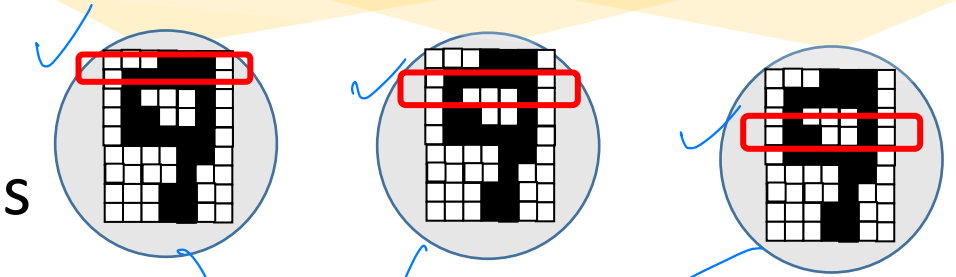
What does this unit detect?



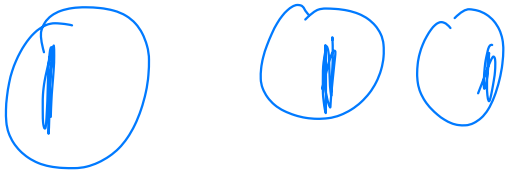
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

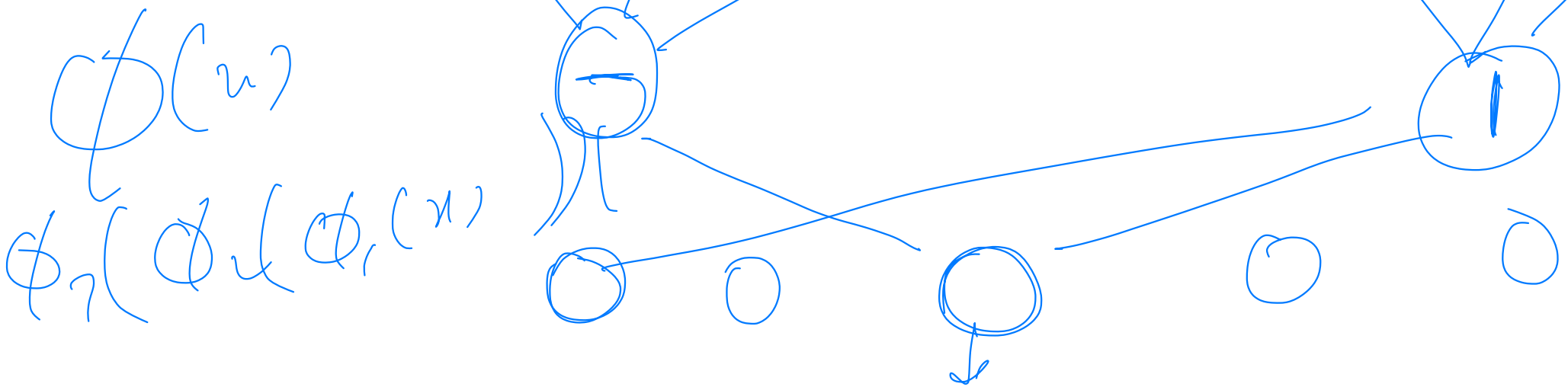
Successive layers can learn higher-level features

detect lines in specific positions



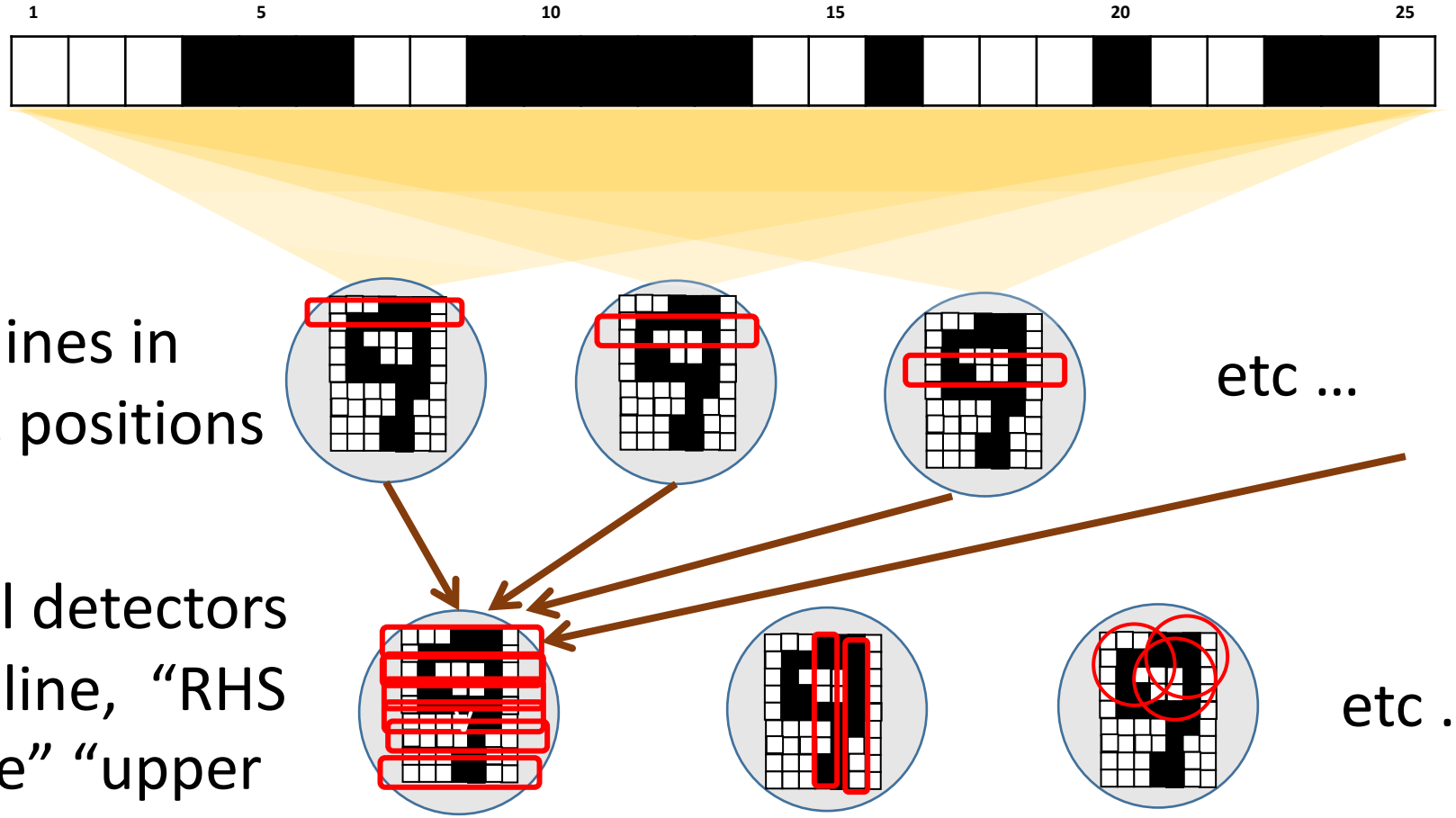
etc …





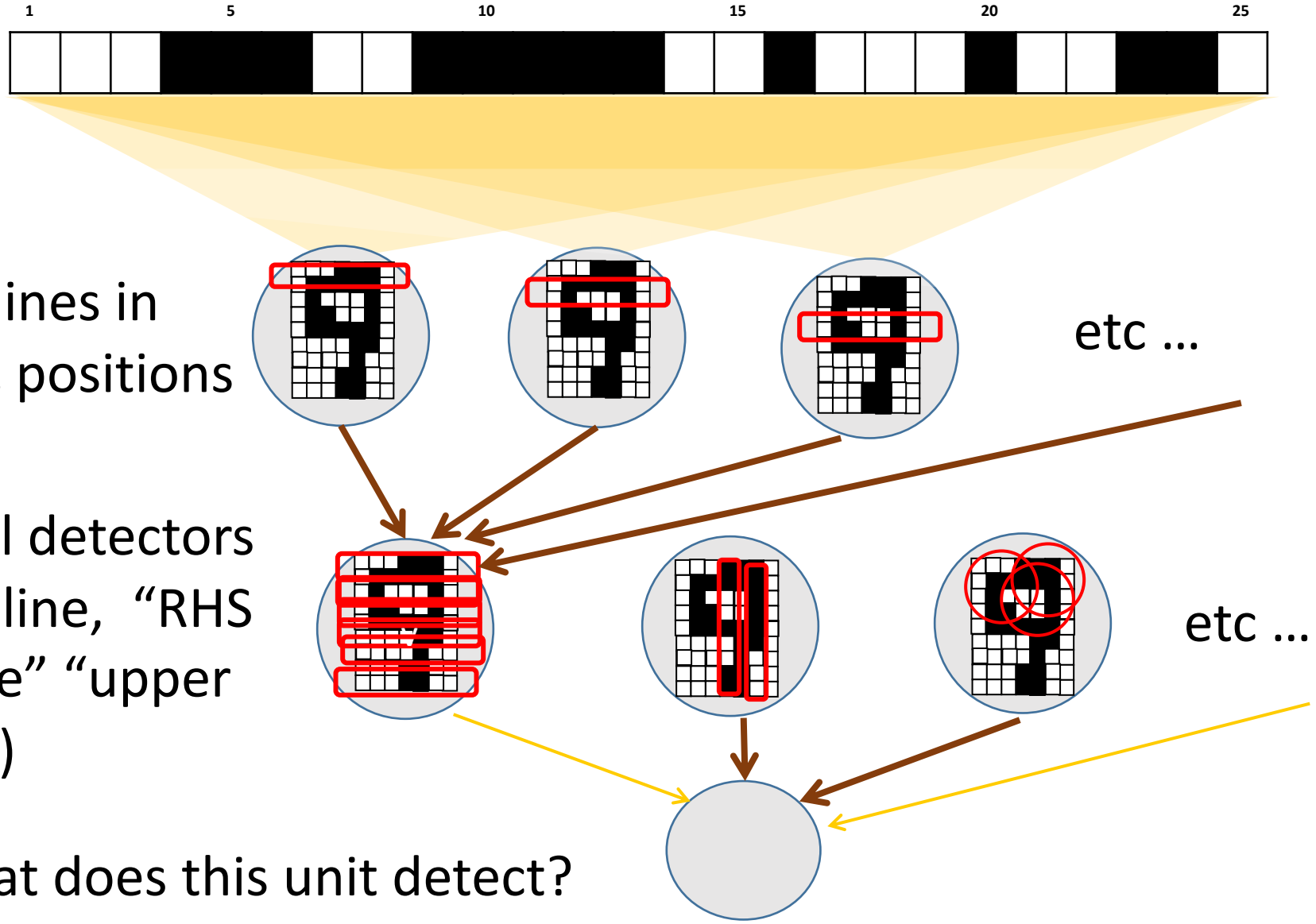
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Successive layers can learn higher-level features



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Successive layers can learn higher-level features

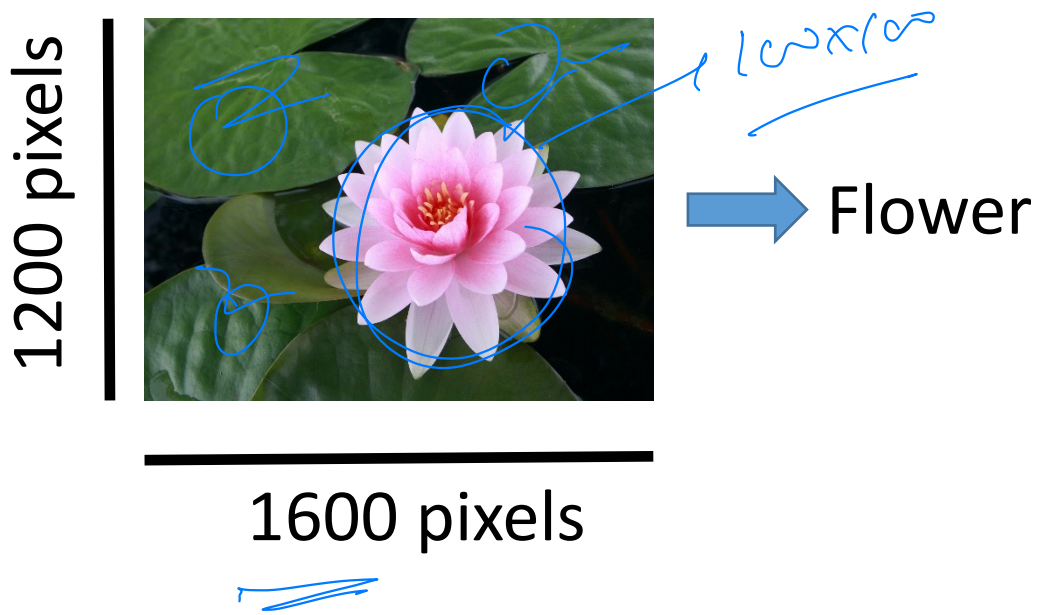


Convolutional Neural Networks

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Motivation: Object Detection in Vision

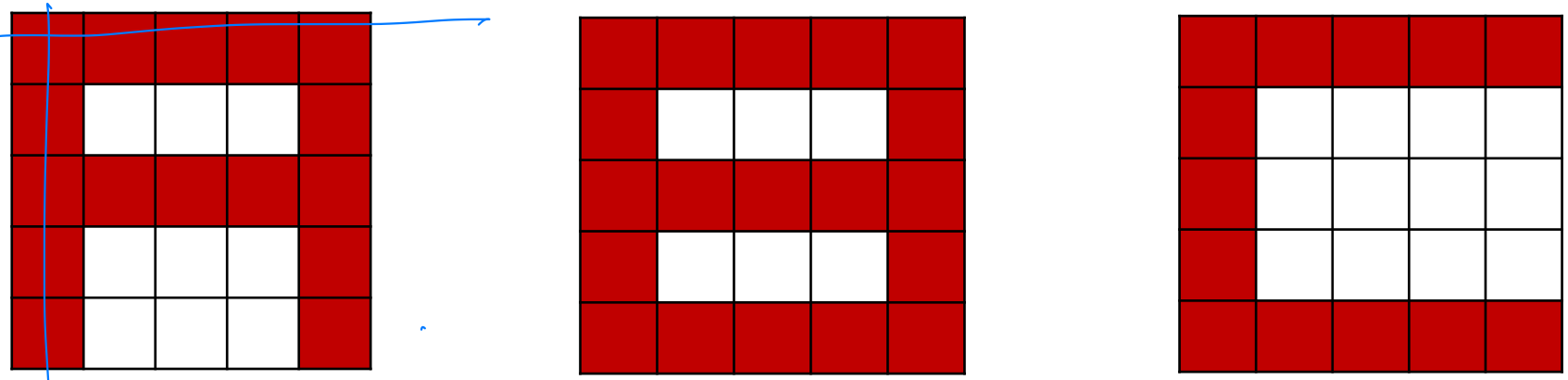
- Challenges
  - Must deal with very high-dimensional inputs
  - 1600 x 1200 pixels = 1.9m inputs, or 3 x 1.9m if RGB pixels
  - Can we exploit the 2D topology of pixels (or 3D for video data)
  - Can we build invariance to certain variations we can expect
    - Translations, illumination, etc



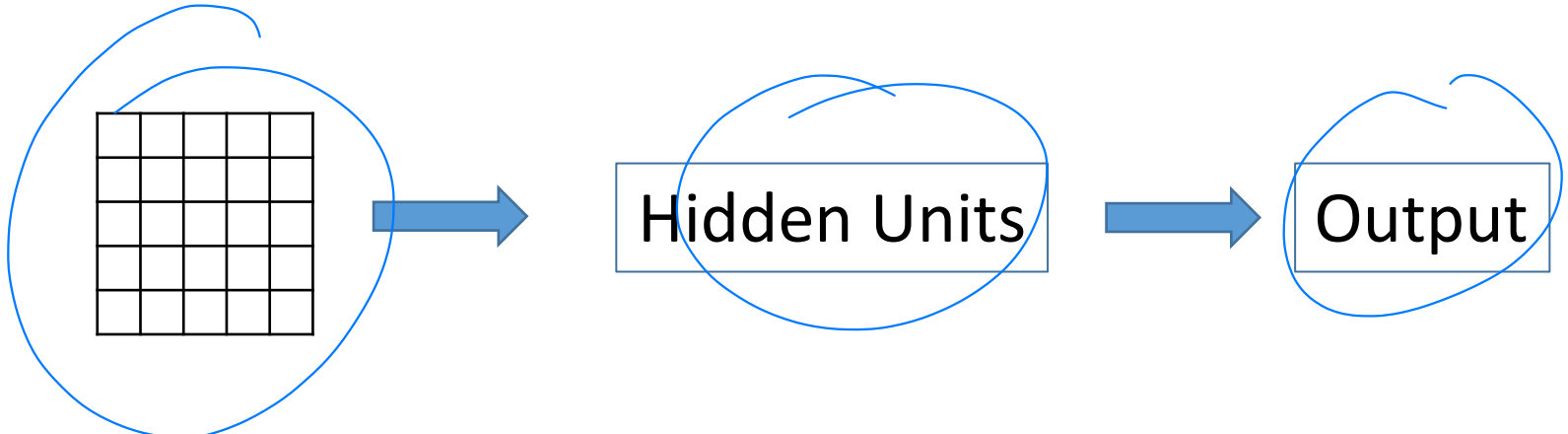
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Motivation: Recognizing an Image

- Input is 5x5 pixel array



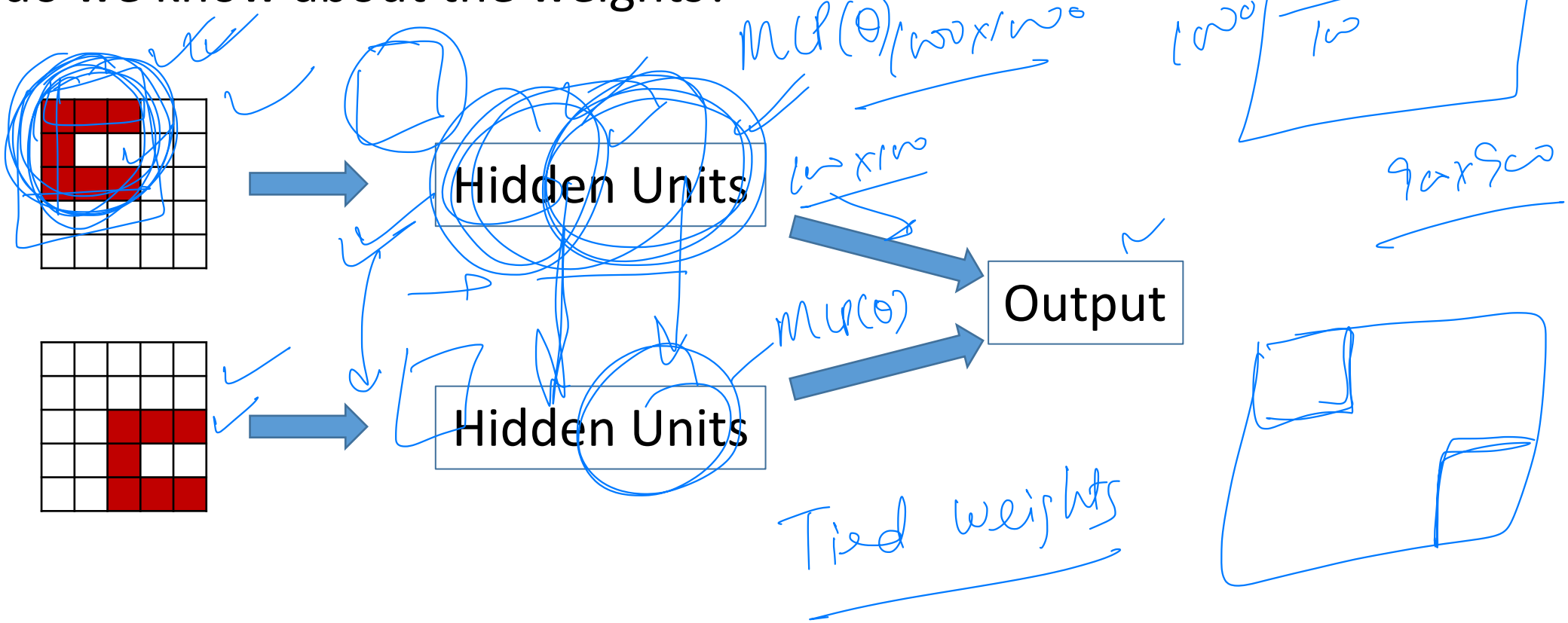
- Simple back propagation net



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Recognizing an Image with unknown location

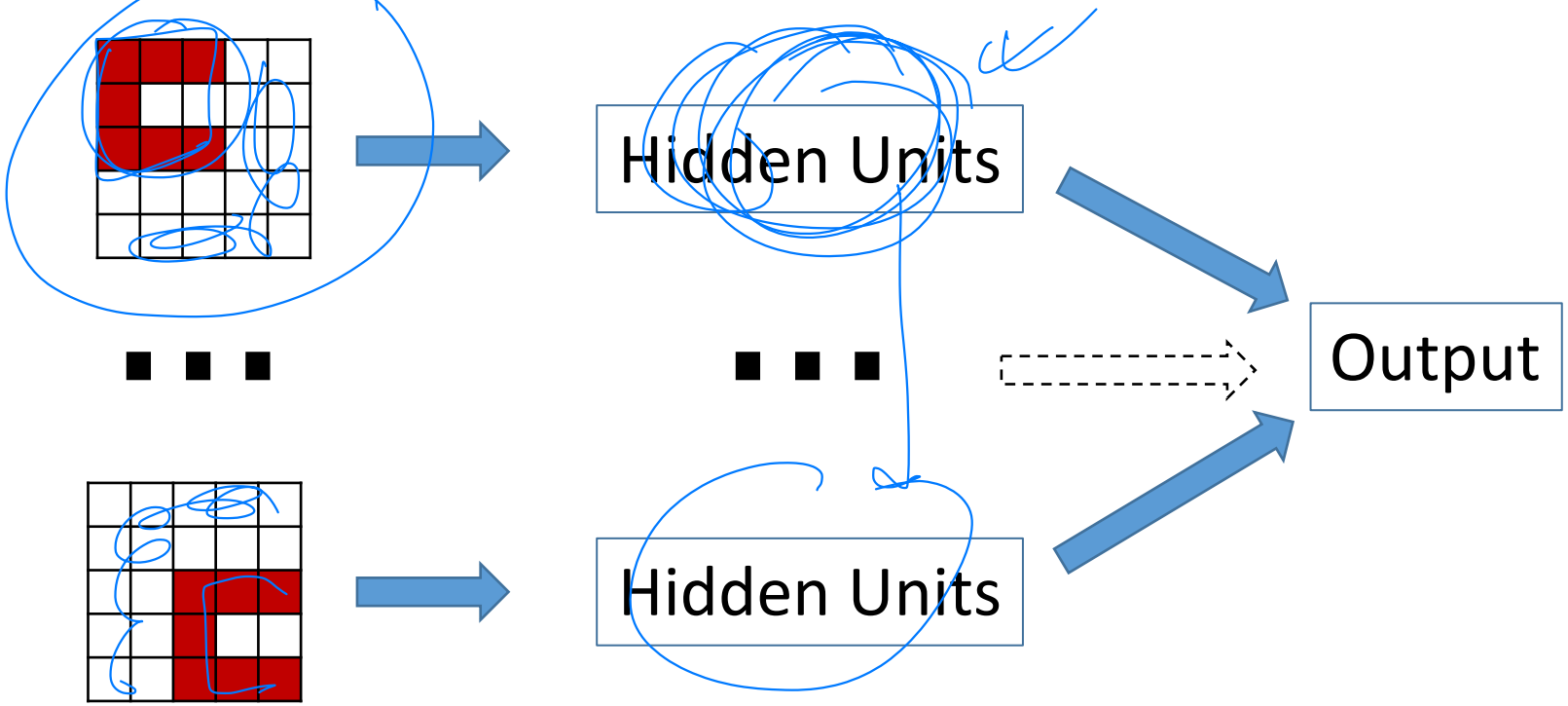
- Object can appear either in the top image or in the bottom image location
- Output indicates presence of object regardless of position
- What do we know about the weights?



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Recognizing an Image with any location

- Each possible location the object can appear in has its own set of hidden units
- Each set detects the same features except in a different location
- Locations can overlap



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

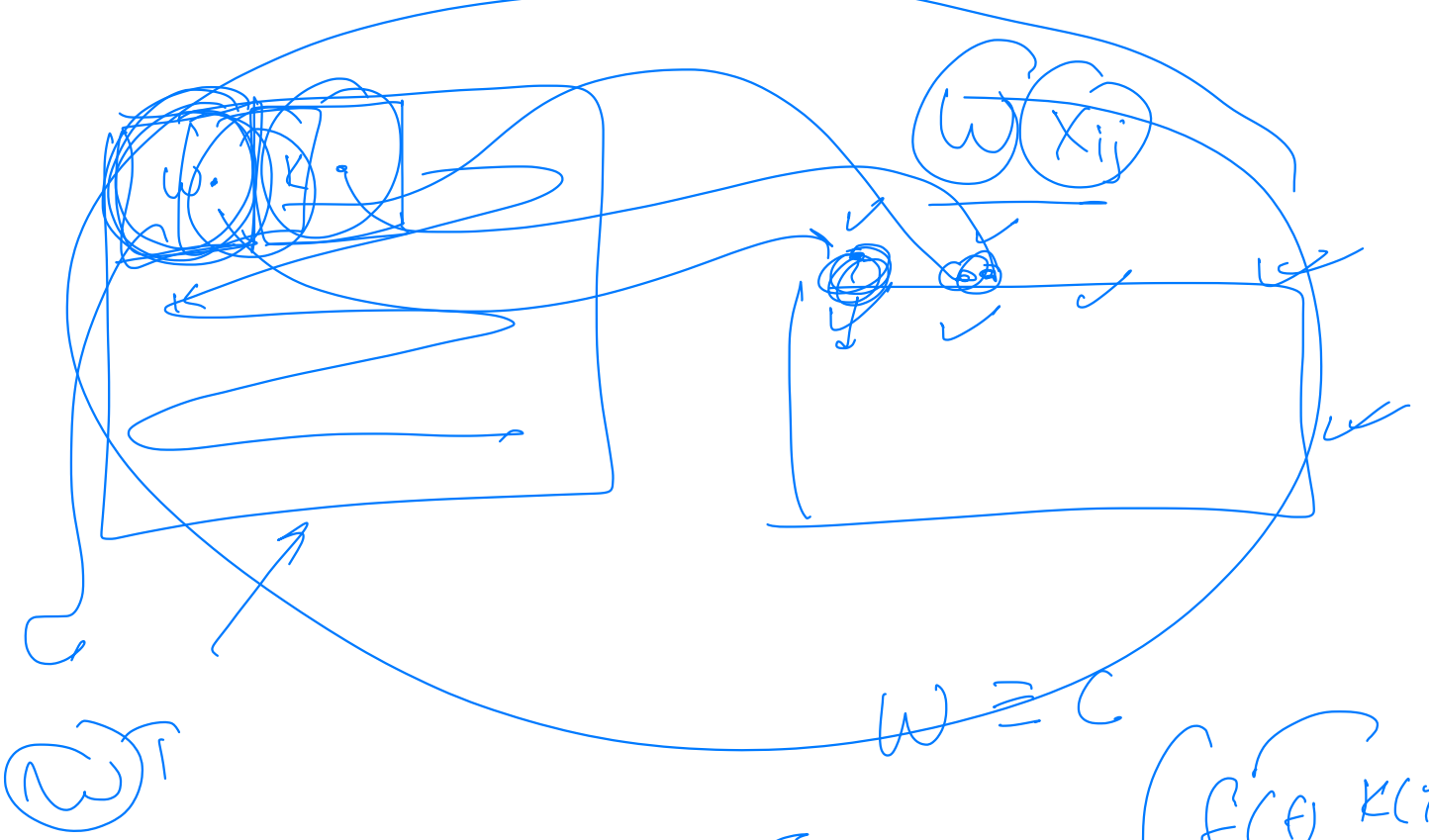
Convolutional Neural Network (CNN): Key Idea

Exploit

1. Structure
2. Local Connectivity
3. Share Parameter

To Give

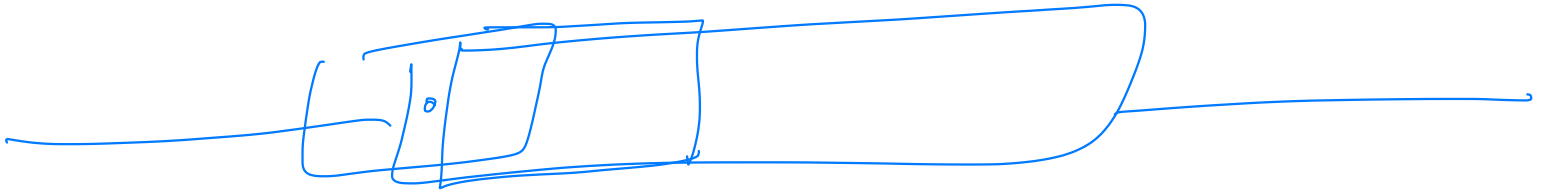
1. Translation Invariance



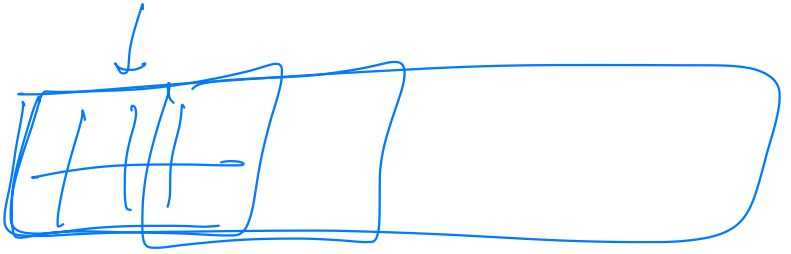
$$s(t) = \int f(t)\, K(t-\tau)\, d\tau$$

$K(t)$

$K(t-\tau)$



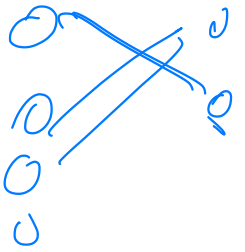
$f(t)$

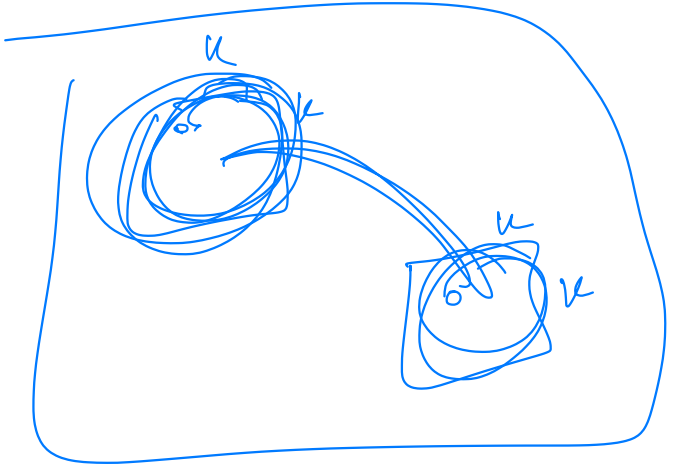


NN → Fully Connected NN

CNN → Locally connected & Tied weights

FC

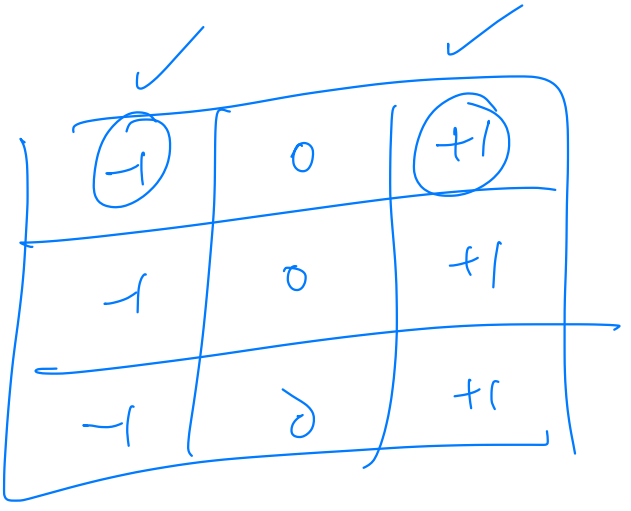


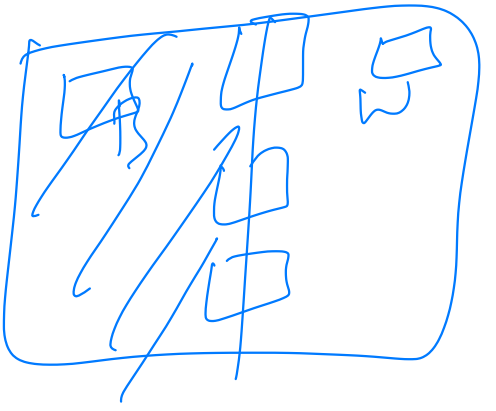


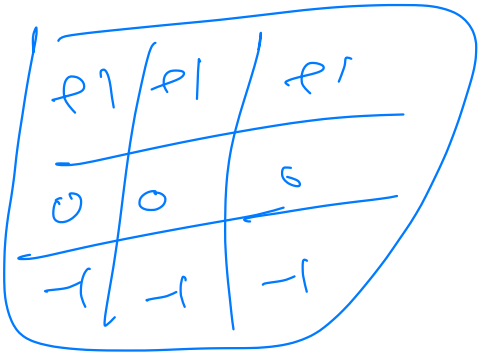
900 x 900

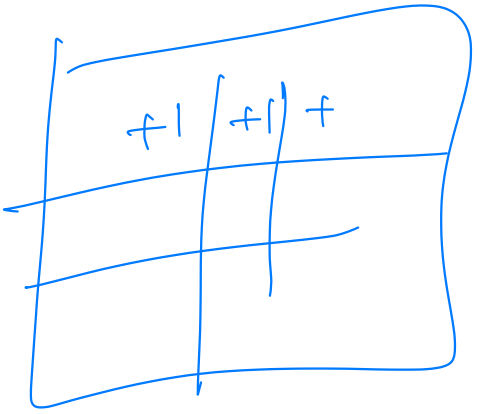
K = 16

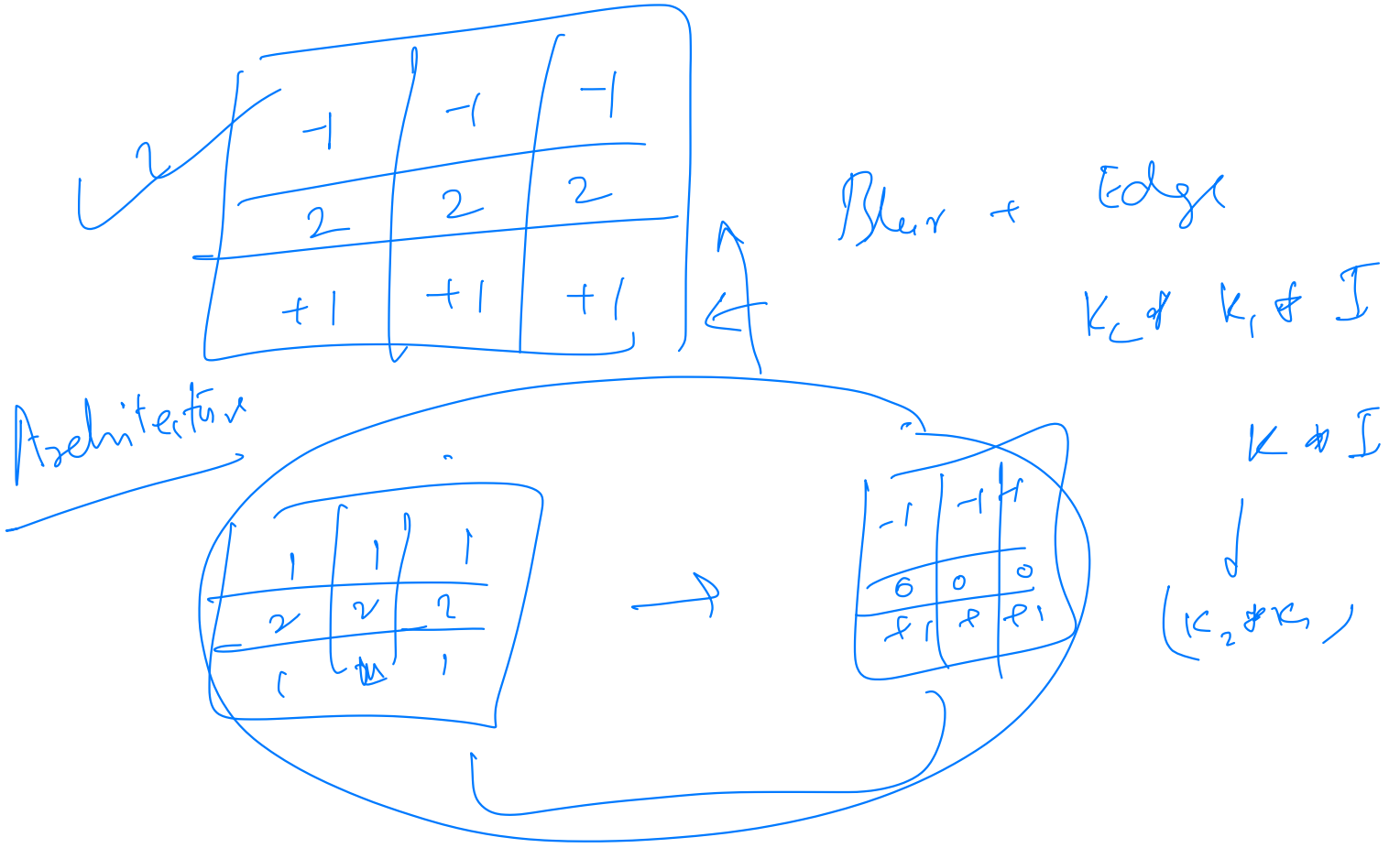
K = 7 x 3

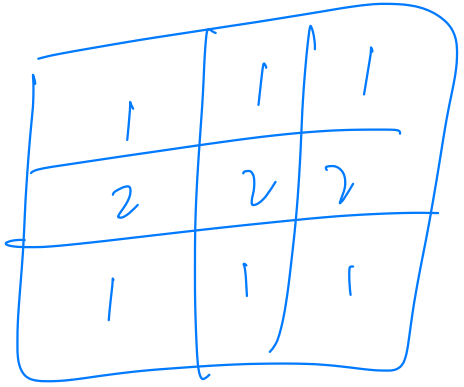

-1
0
+1
-1
0
+1
-1
0
+1





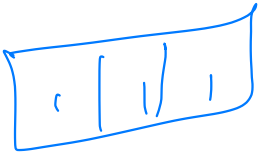

+1
+1
+


-1 -1 -1
2 2 2
+1 +1 +1
Blur + Edge
$K_2 * K_1 * I$
Architecture
$K * I$
$(K_2 * K_1)$



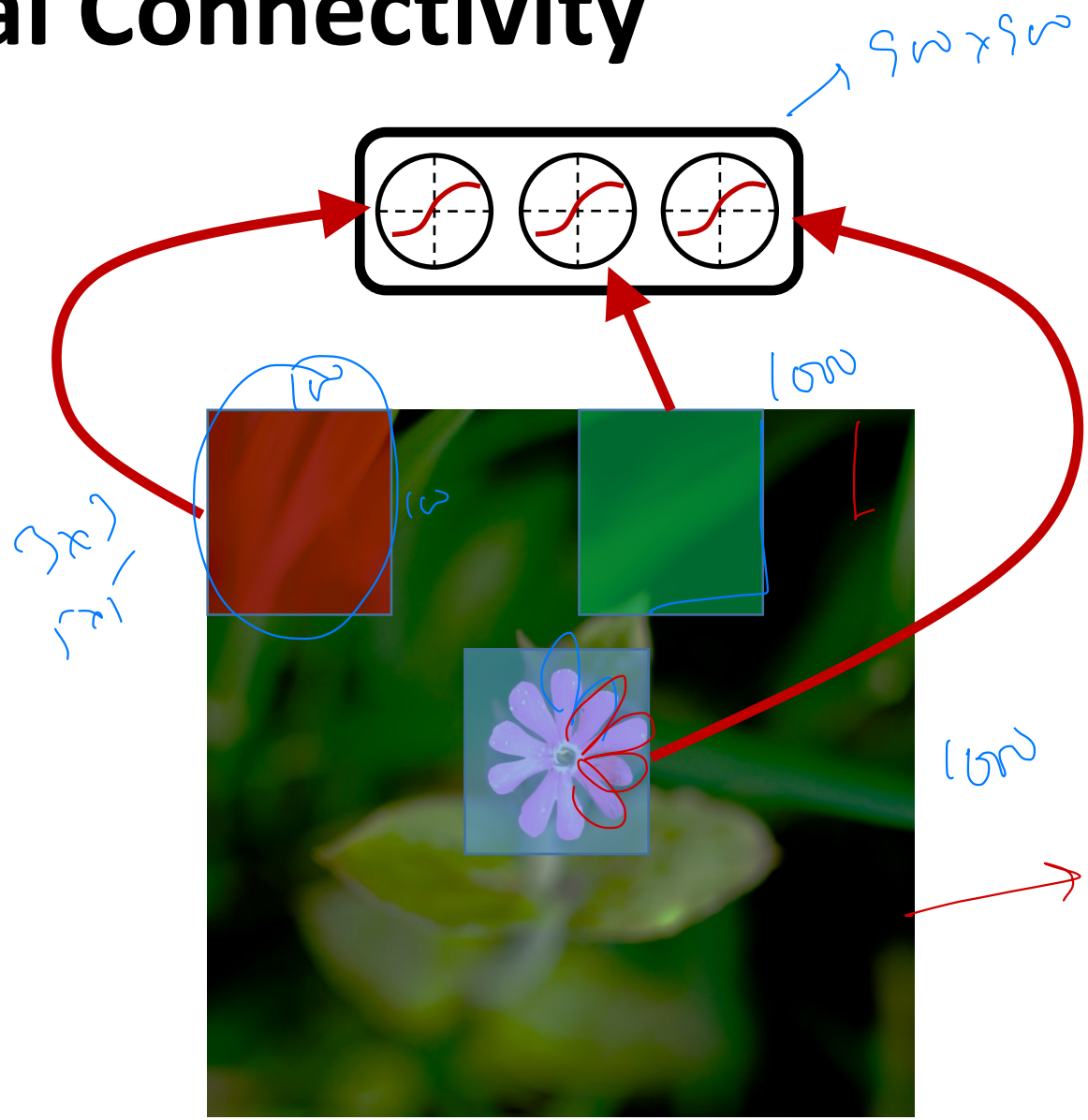
$=$

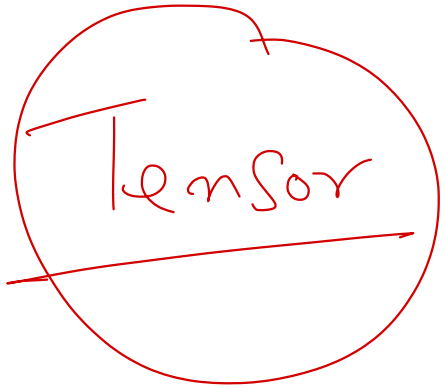
$\begin{matrix} 1 \\ 2 \\ 1 \end{matrix} \Rightarrow$

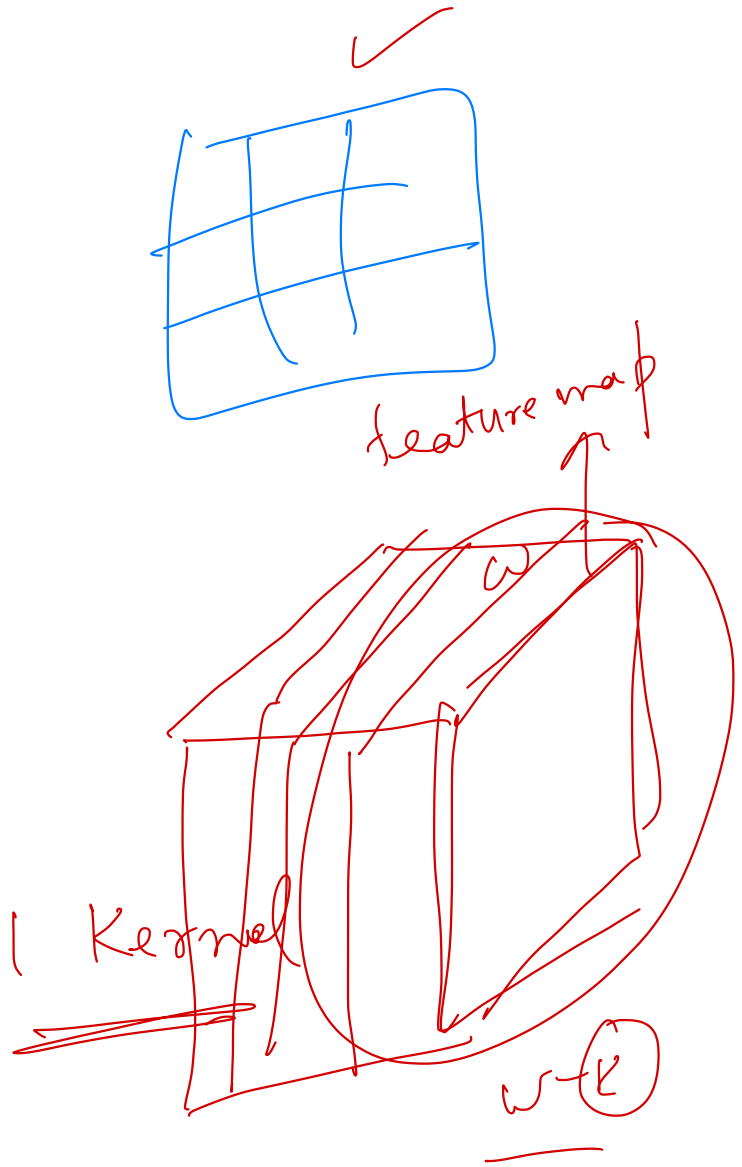


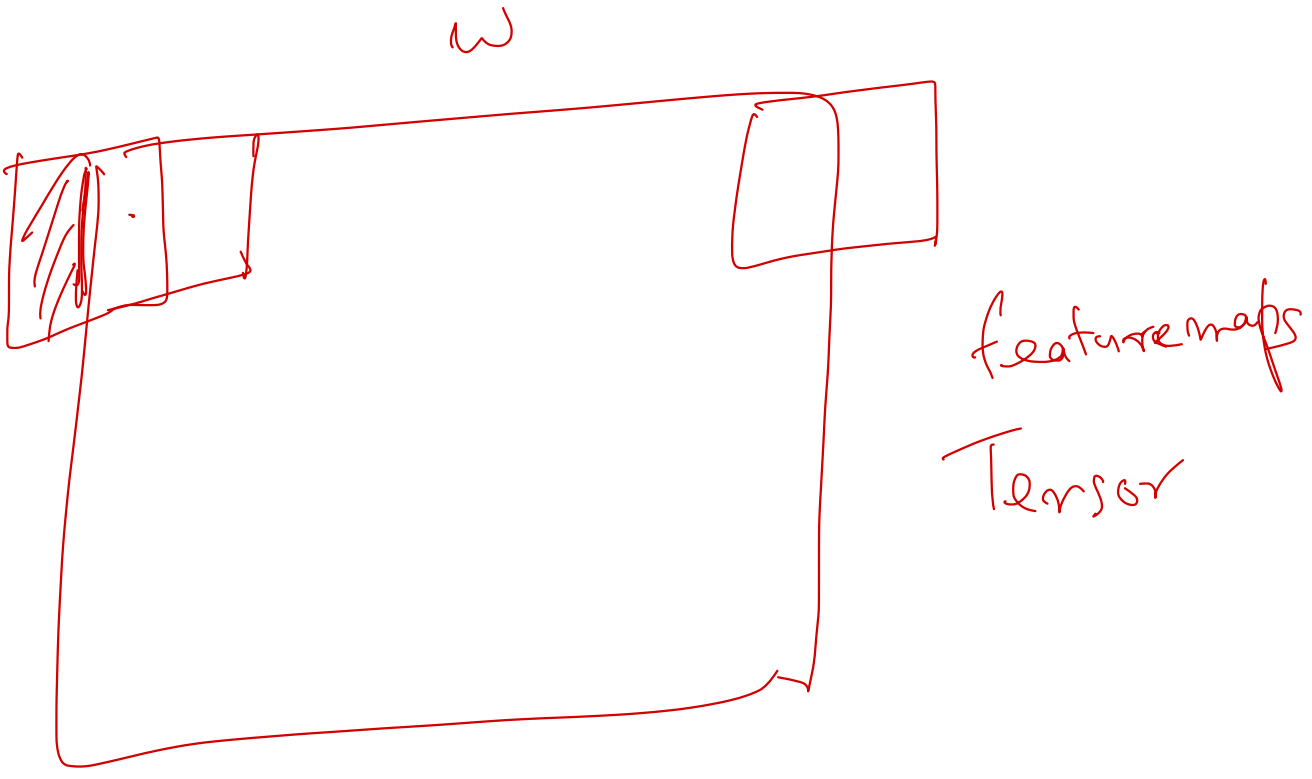
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

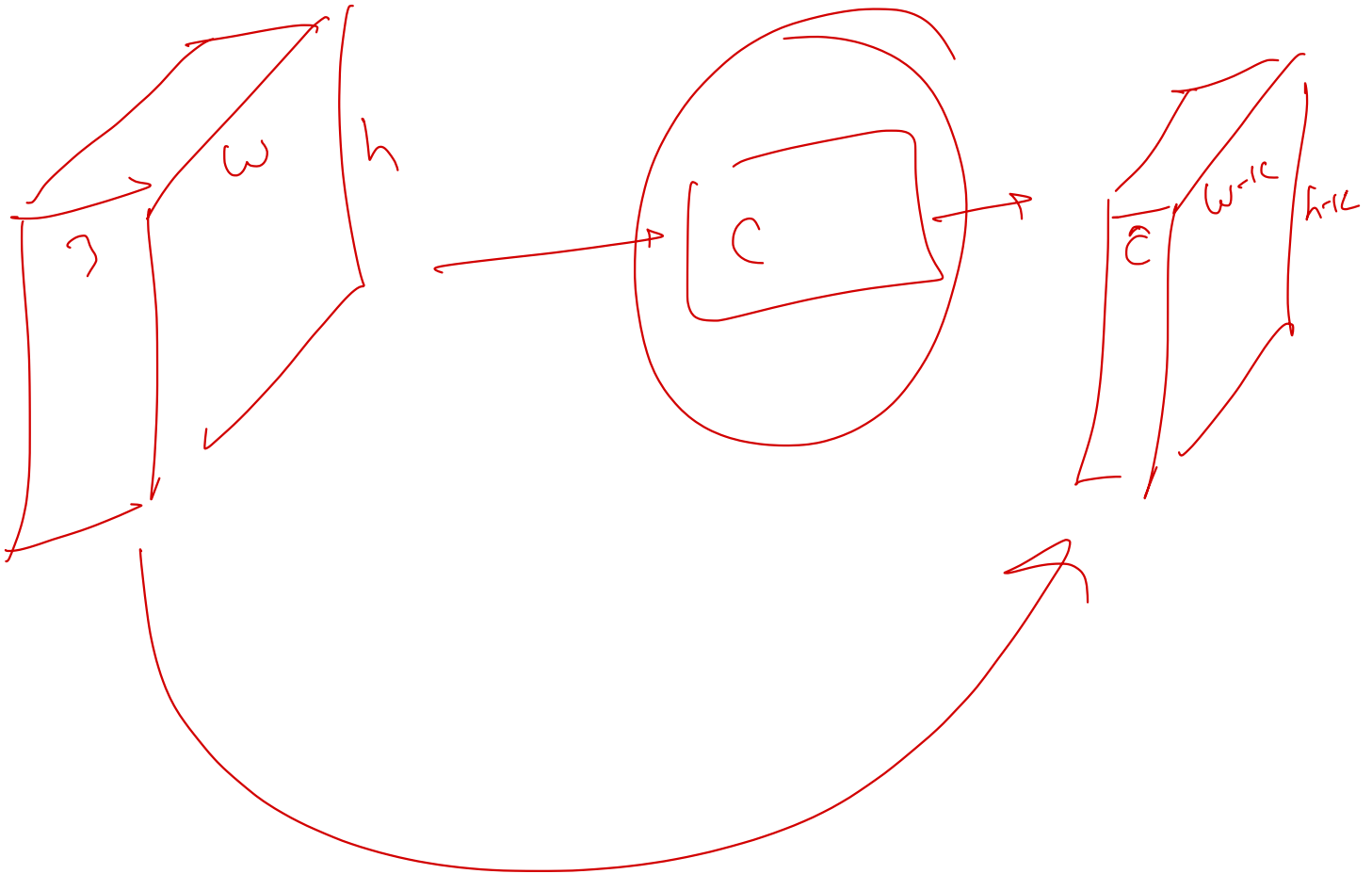
CNN: Local Connectivity

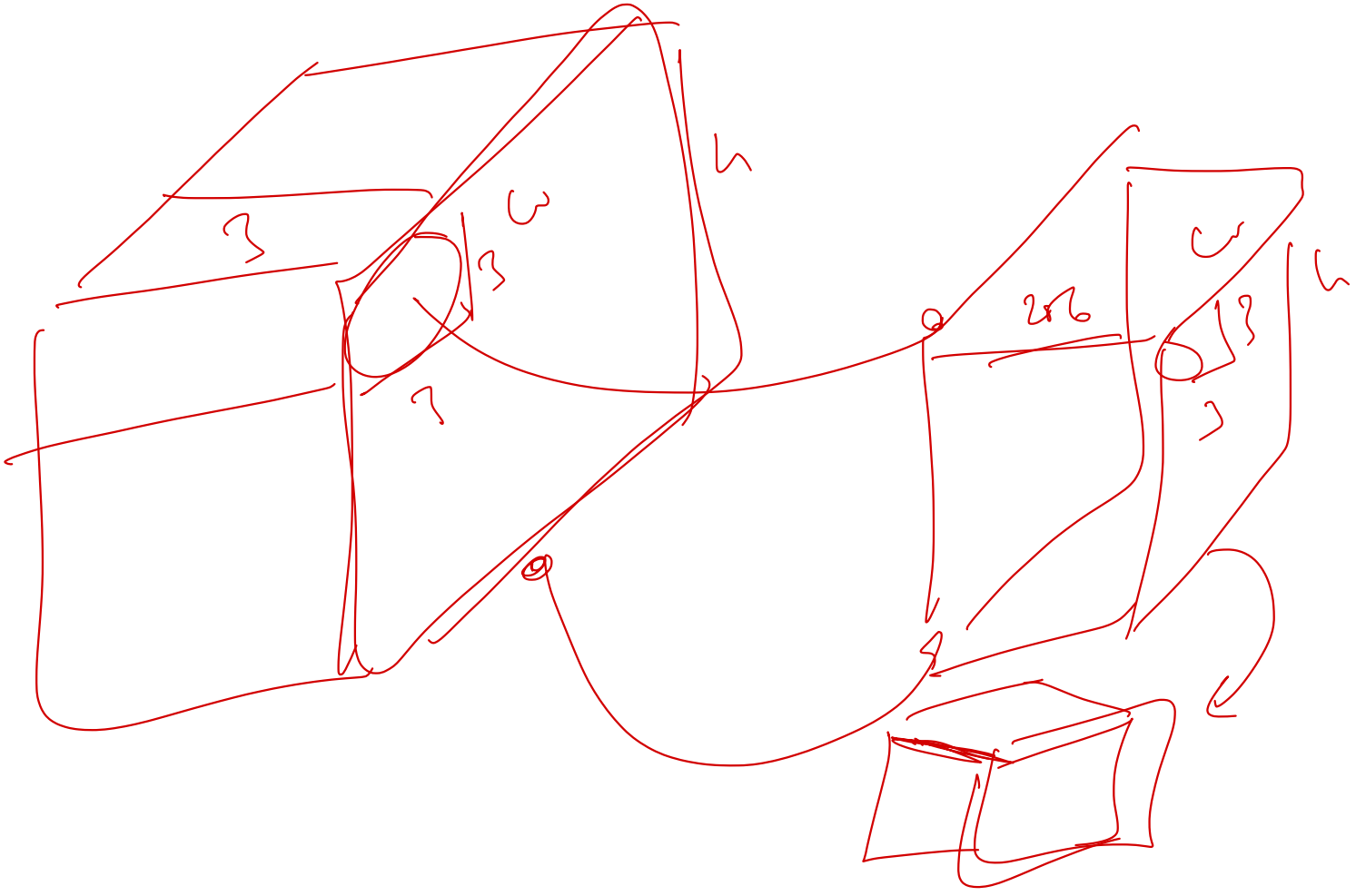


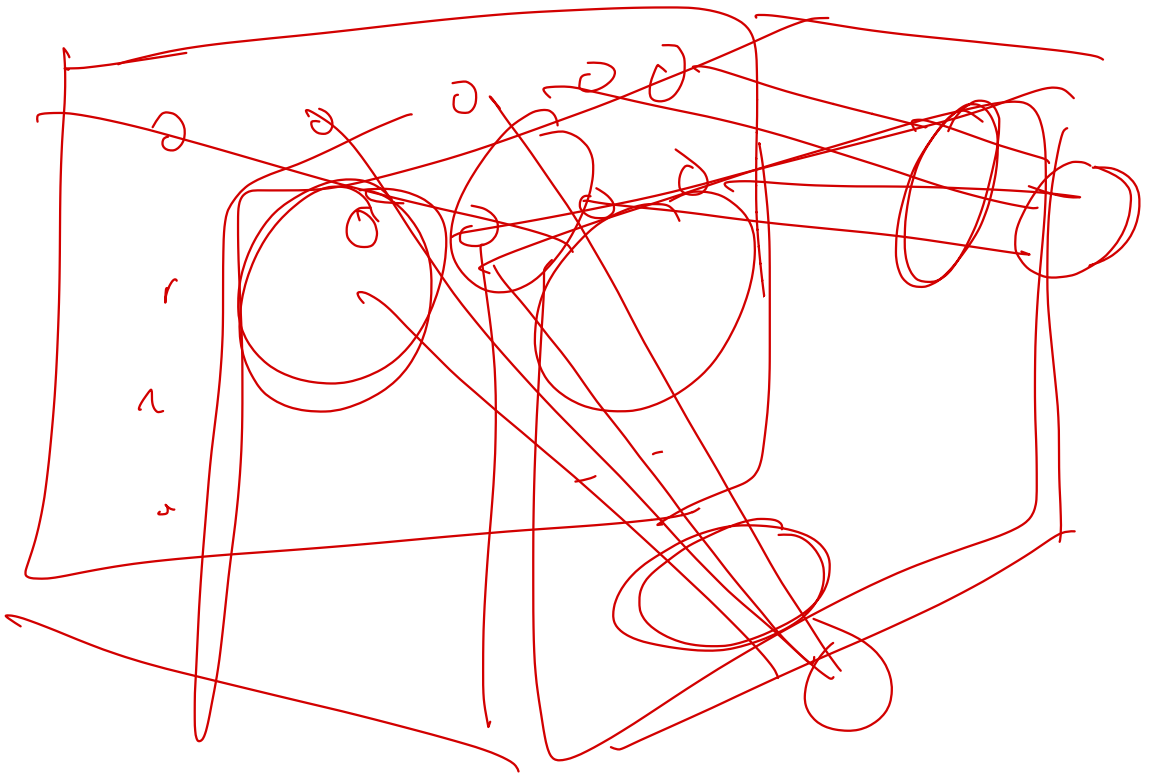


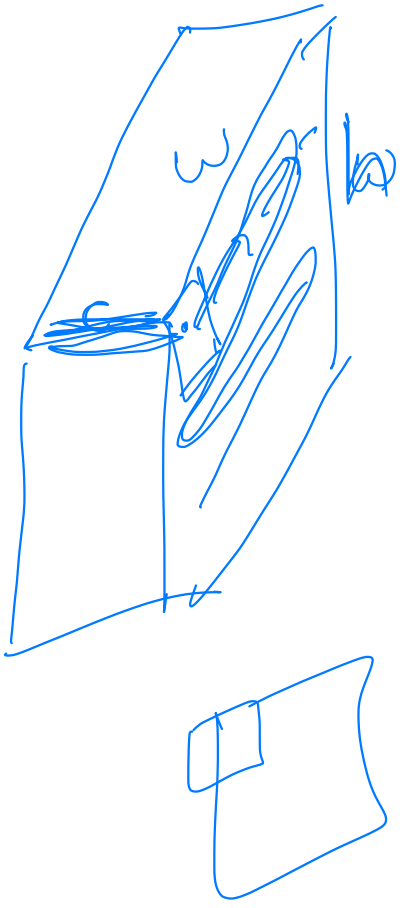



W
featuremaps
Tensor


w
h
c
c
w-1c
h-1c



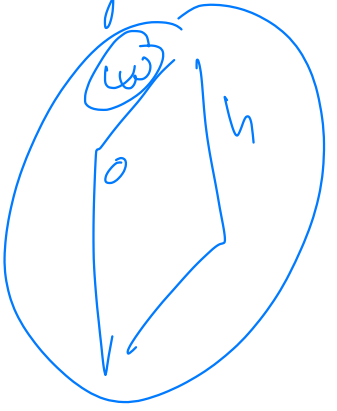




$\sum w_i x_i$

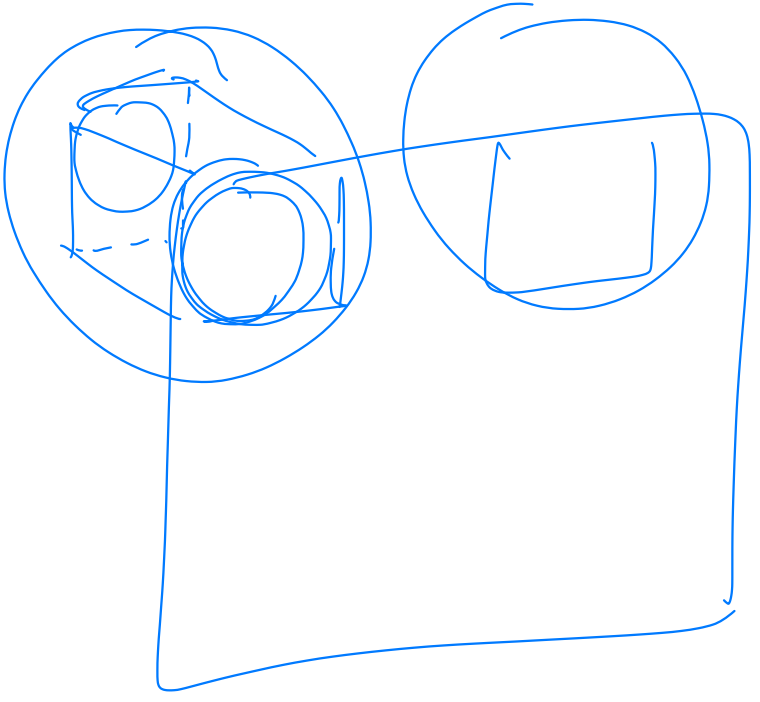
CNN

Convolution Layer



X 3

27




Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

CNN: Local Connectivity

- Each hidden unit is connected only to a sub-region (patch) of the input image
- It is connected to all channels
  - 1 if greyscale image
  - 3 (R, G, B) for color image

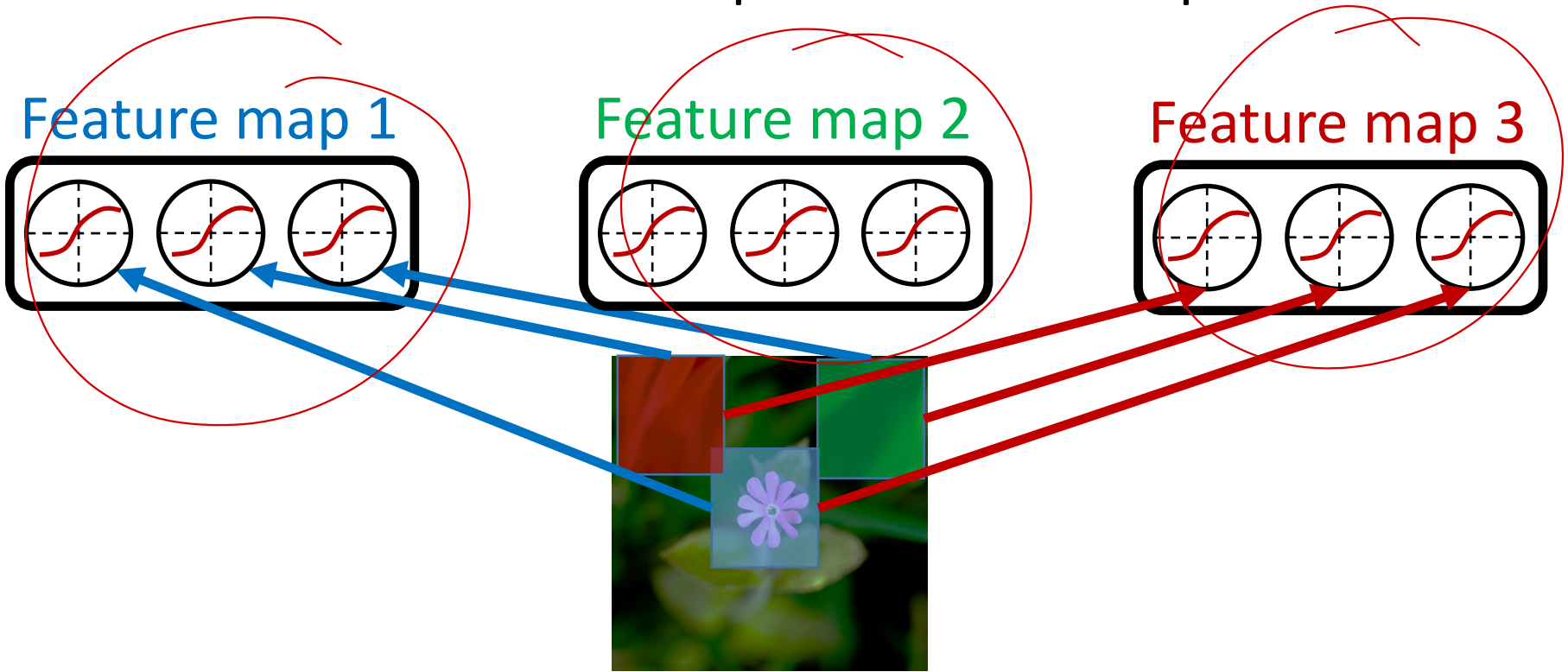
Solves the following problems:

- Fully connected hidden layer would have an unmanageable number of parameters
- Computing the linear activations of the hidden units would be very expensive

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

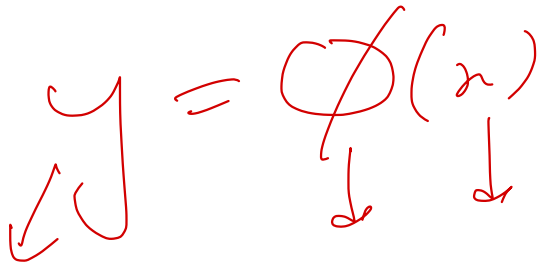
CNN: Parameter Sharing

- Units organized into the same “feature map/template” share parameters
- In this way all neurons detect the same feature/template at different positions in the input image
- Hidden units within a feature map cover different positions in the image



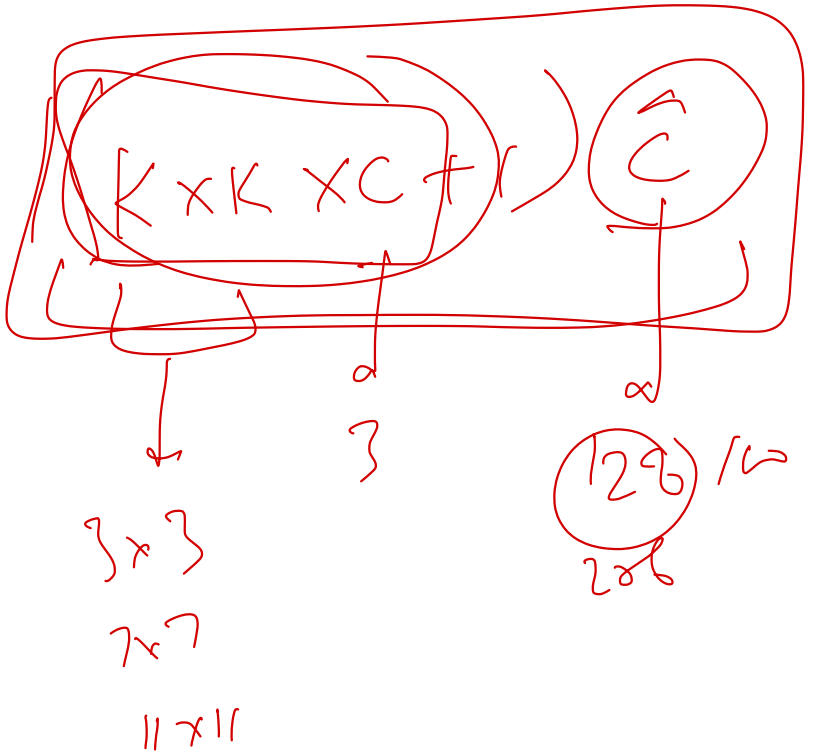
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

CNN: Parameter Sharing



How does it help?

- Reduces even more number of parameters (compared to local connectivity)
- Will extract the same features at every position
  - features are “equi-variant”



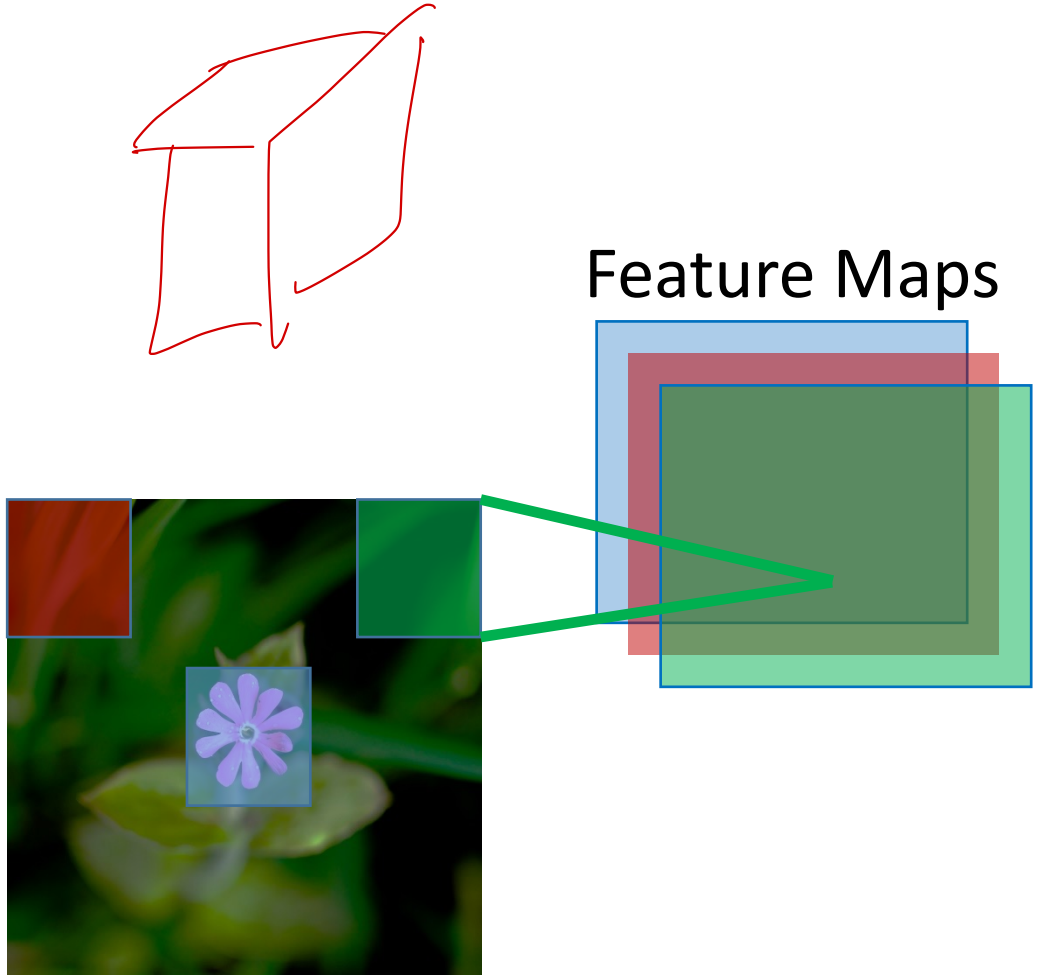
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

CNN: Parameter Sharing

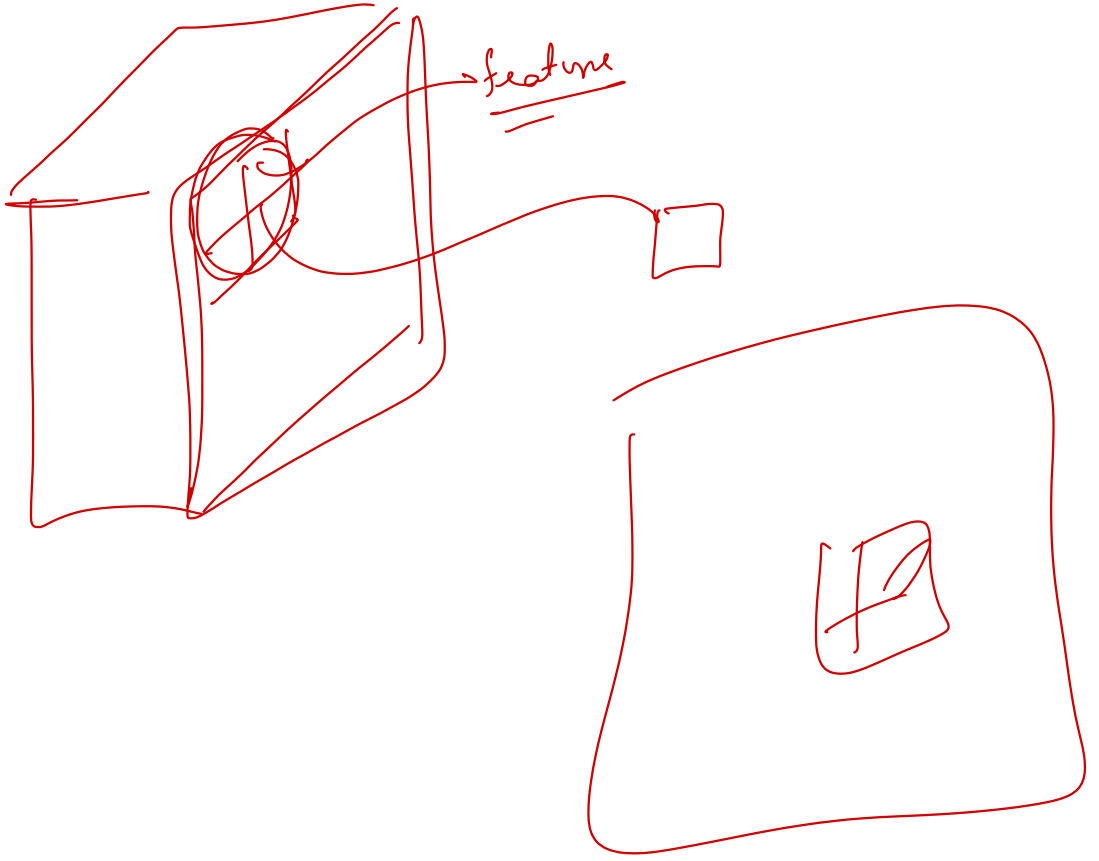
- Each feature map forms a 2D grid of features
- Can be computed with a discrete convolution ($*$) of a kernel matrix k_{ij} which is the hidden weights matrix W_{ij} with its rows and columns flipped:

$$y_j = \tanh \sum_i k_{ij} x_i$$

- x_i is the i^{th} channel of input
- k_{ij} is the convolution kernel
- y_j is the hidden layer

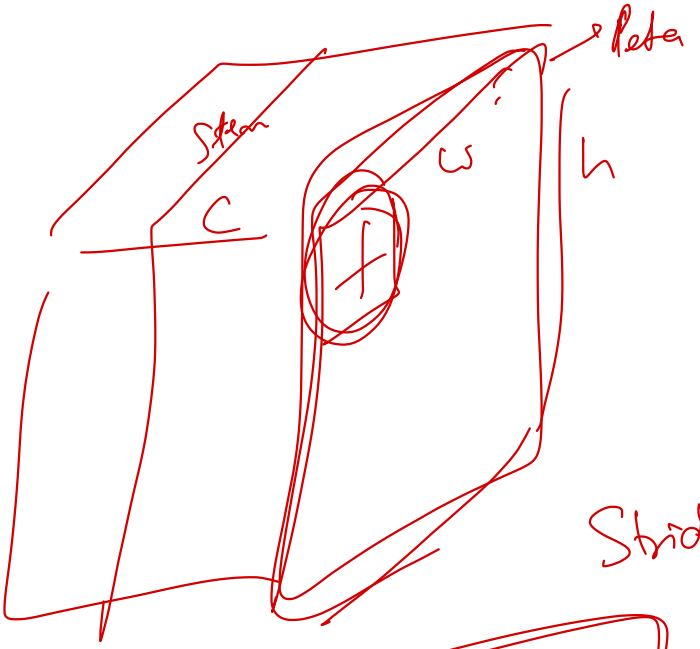


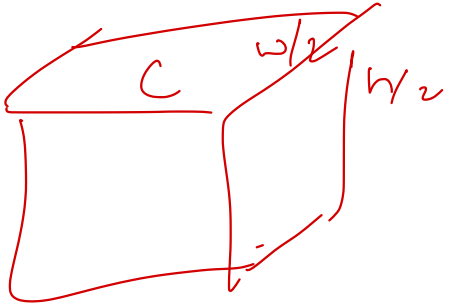
Pooling



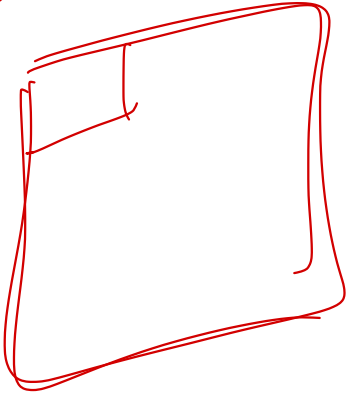
Pooling → max
→ average

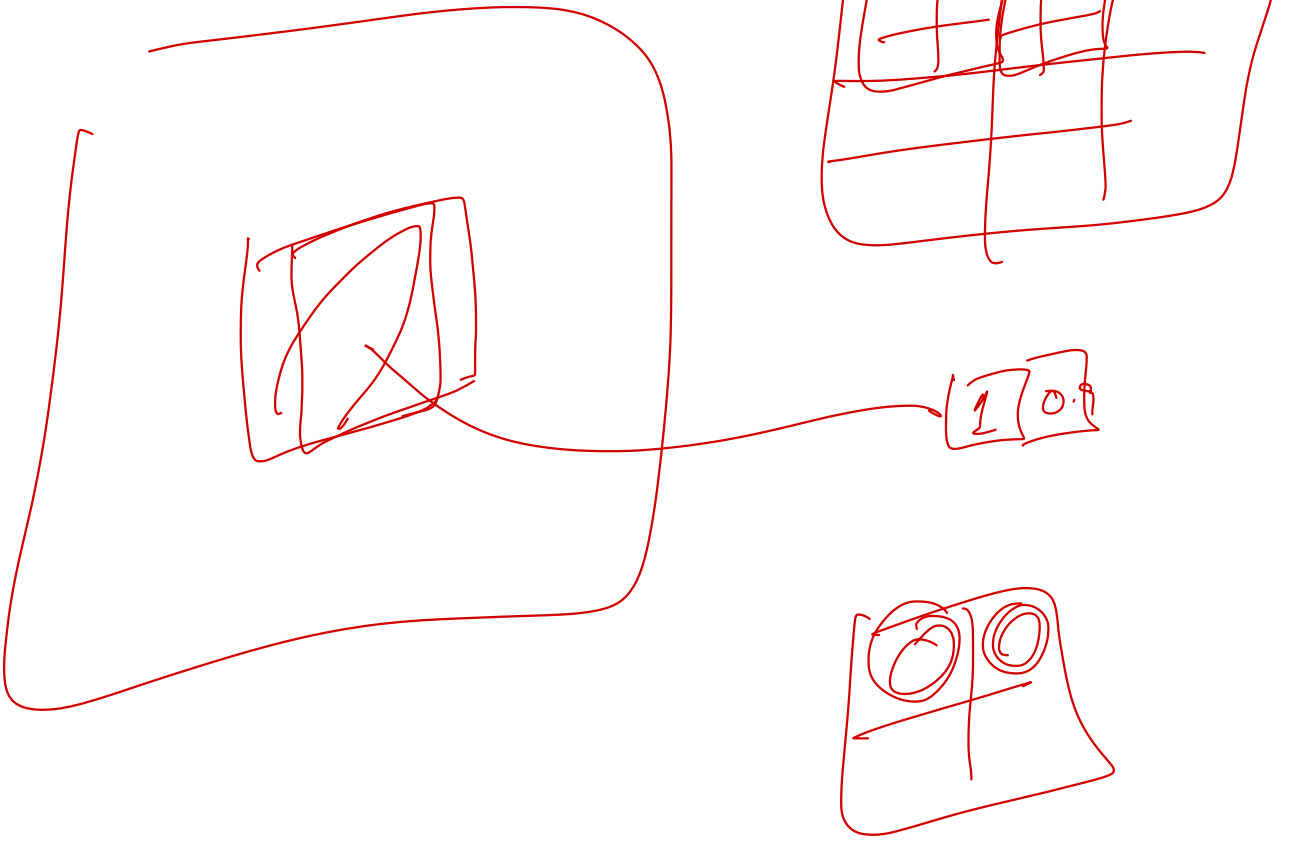
Pooling
Convolution + Stride

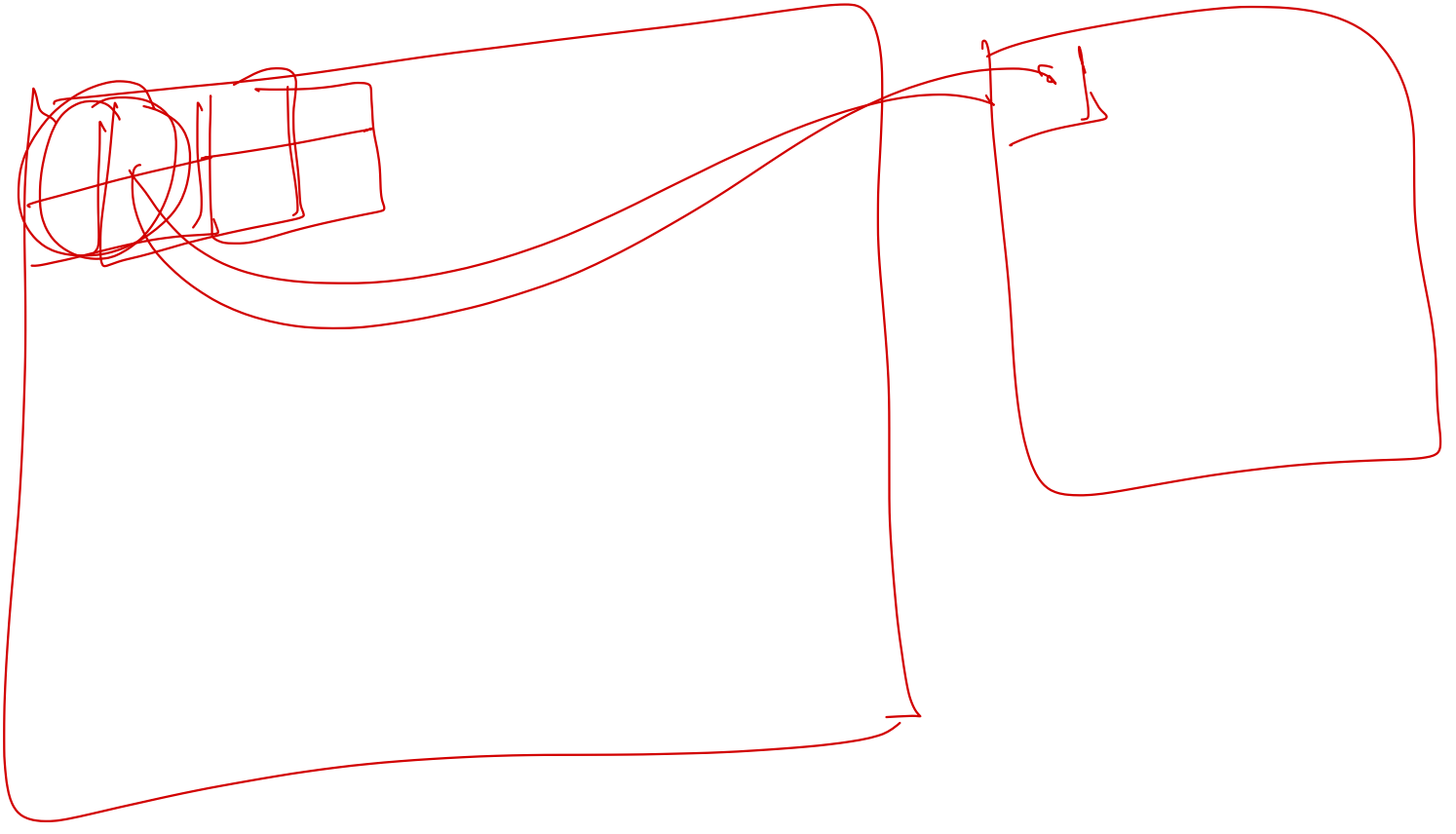


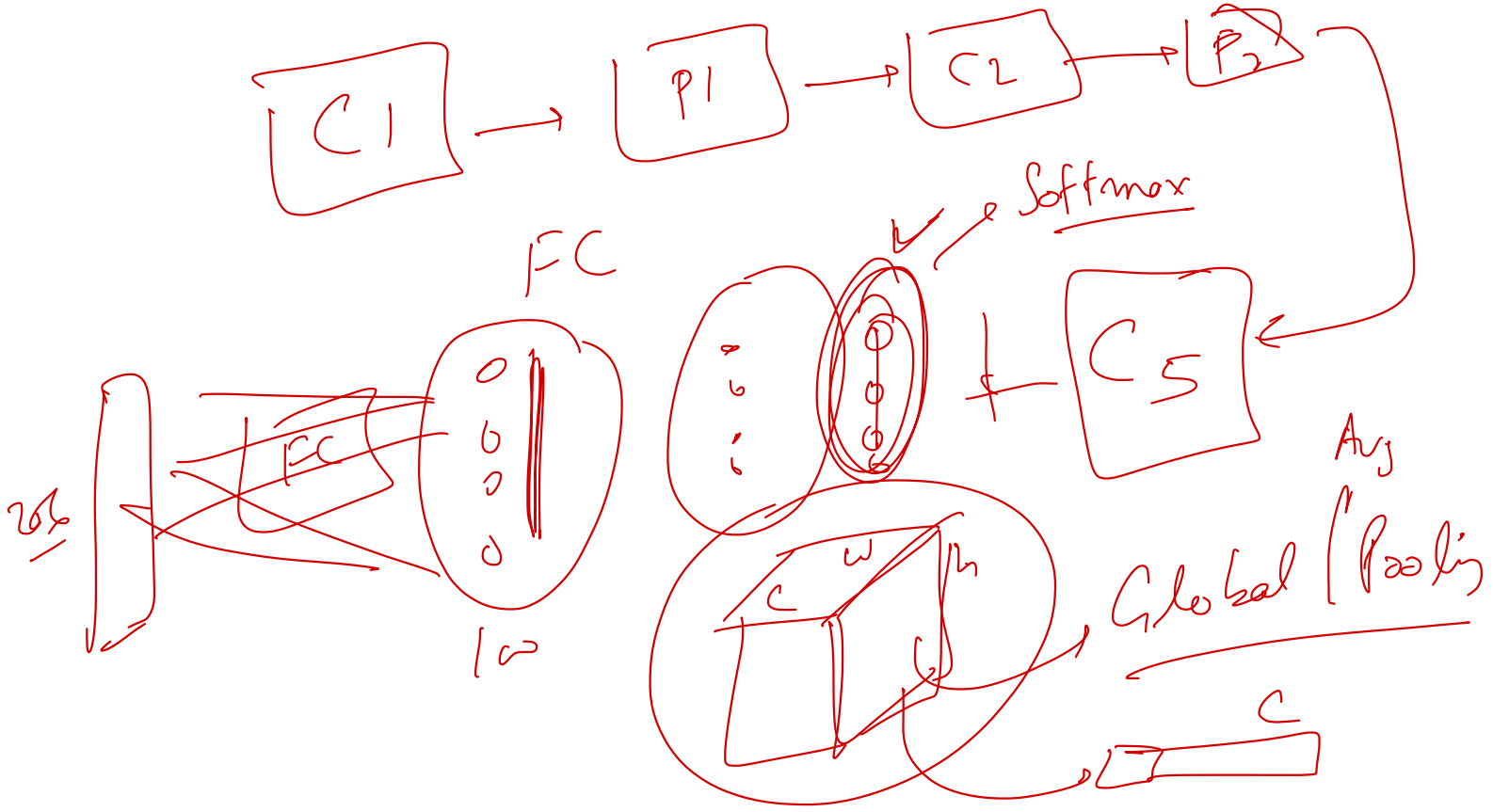


Stride




10.9




C1
P1
C2
P2
Softmax
FC
FC
C5
Avg
Global Pooling
C


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

CNN: Parameter Sharing

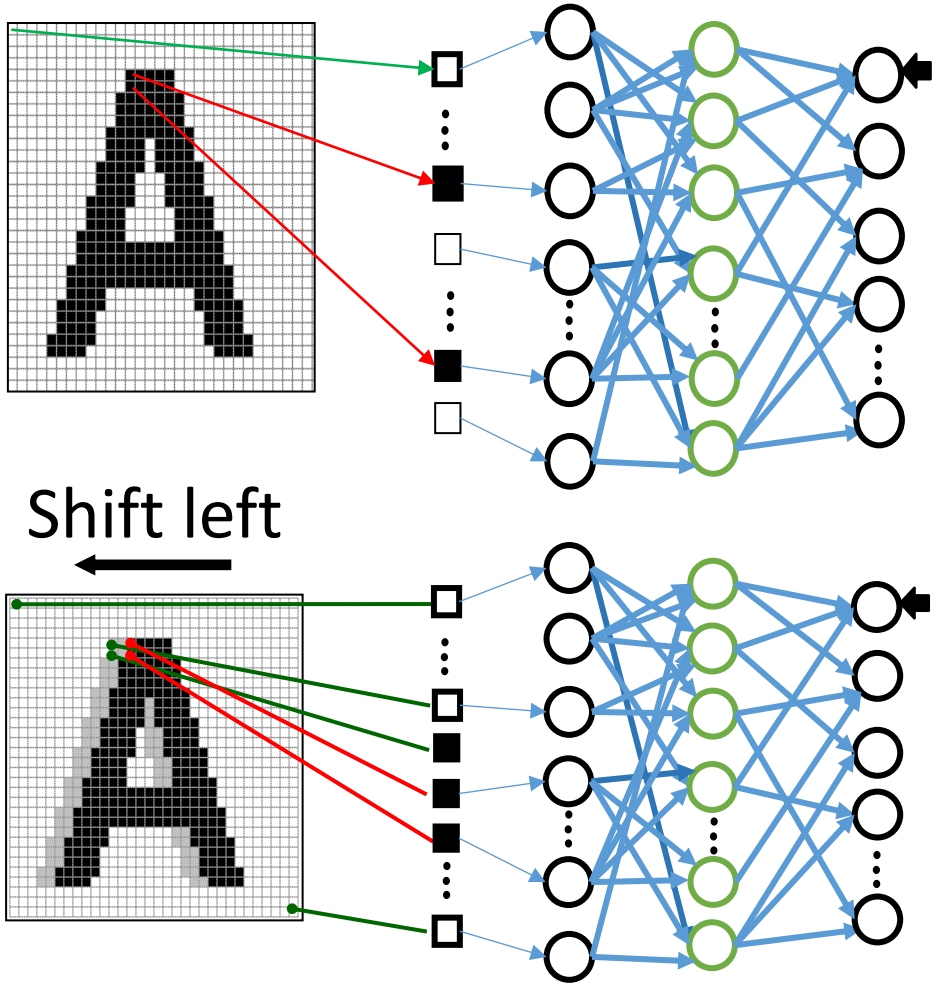
- **Stride** (typically denoted by **s** in CNNs) decides the spacing between overlapping convolutions

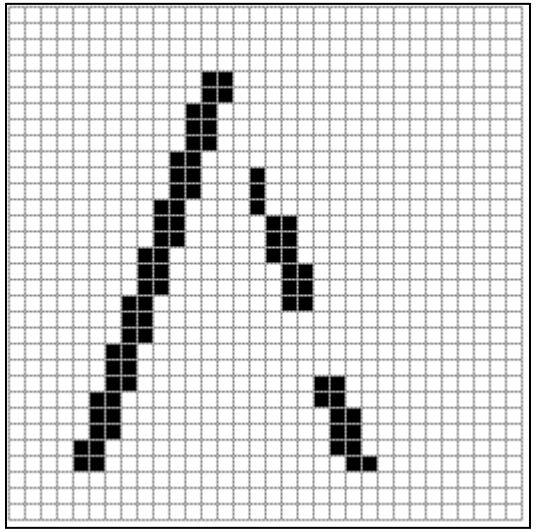
- Helps in reducing parameters further

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Drawbacks of Neural Networks

- Little or no invariance to shifting





154 input change from 2 shift left

77 : black to white

77 : white to black

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Drawbacks of Neural Networks

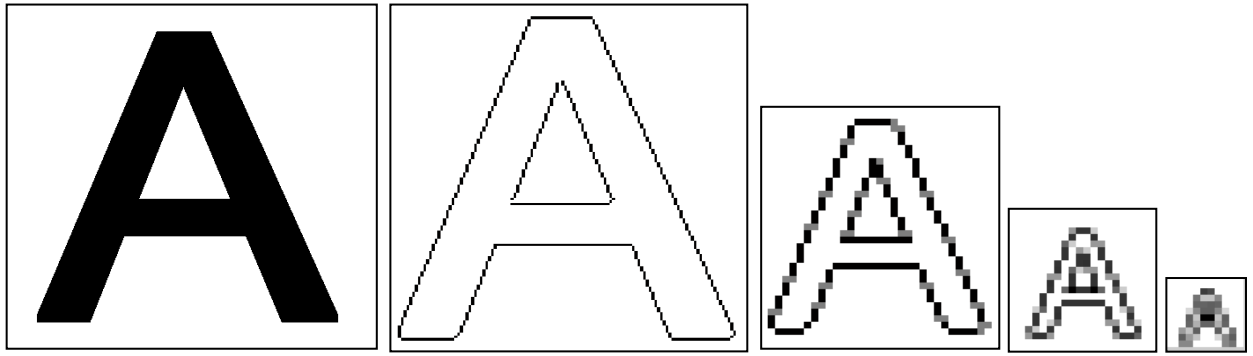
- Scaling, or other forms of distortion

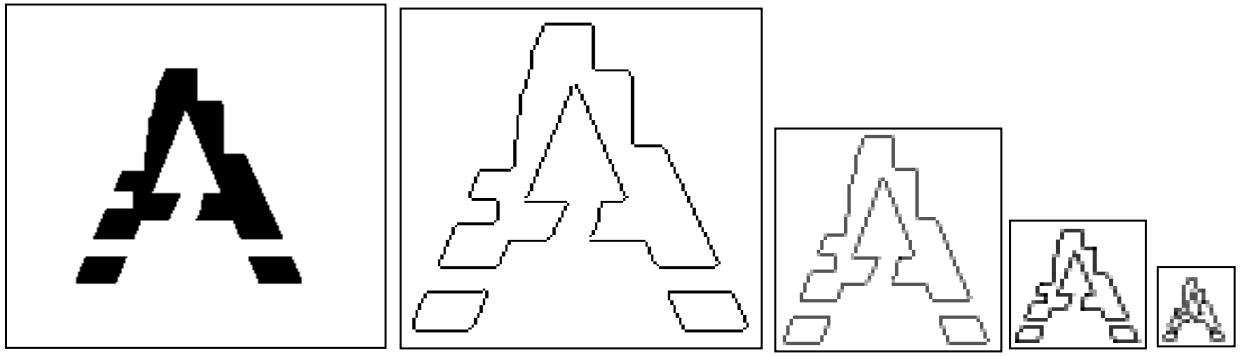
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Pooling or Subsampling Layer

- The pooling layers reduce the spatial resolution of each feature map. Helps to collate the features in a region

- By reducing the spatial resolution of the feature map, a certain degree of noise, shift and distortion invariance is also achieved.






Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

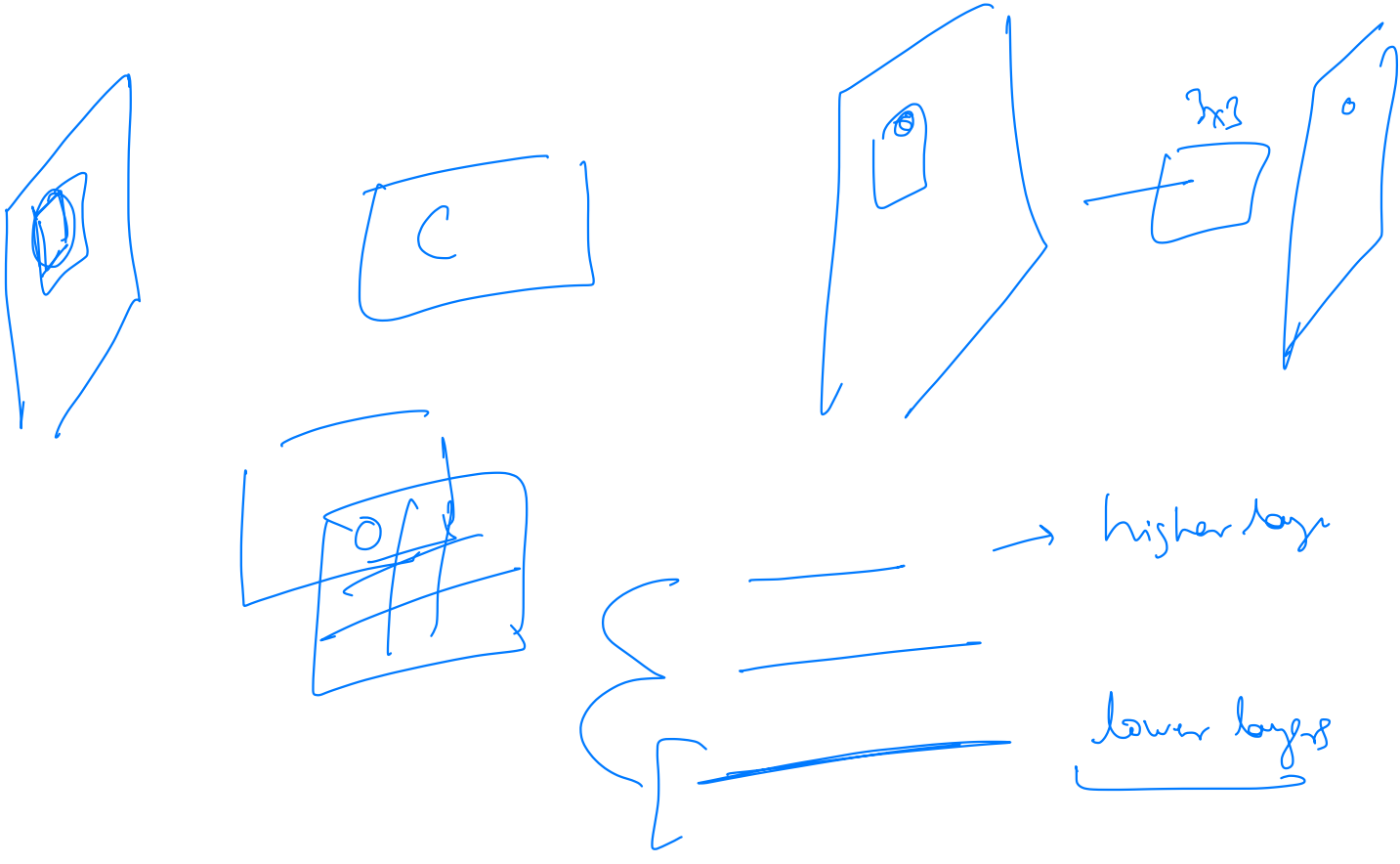
Jargon

- Convolutional Neural Networks
  - also called CNNs, Conv Nets etc.

- Each hidden unit channel
  - also called **map, feature, feature type, dimension**

- Weights for each channel
  - also called **kernels**

- Input patch to a hidden unit at (x,y)
  - also called **receptive field**


C
3x3
higher layer
lower layers


Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Typical CNN

- Alternates convolutional and pooling layers
- Output layer is a regular, fully connected layer with softmax non-linearity
  - Output provides an estimate of the conditional probability of each class
- The network is trained by stochastic gradient descent

Backpropagation

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Backpropagation in General Cases

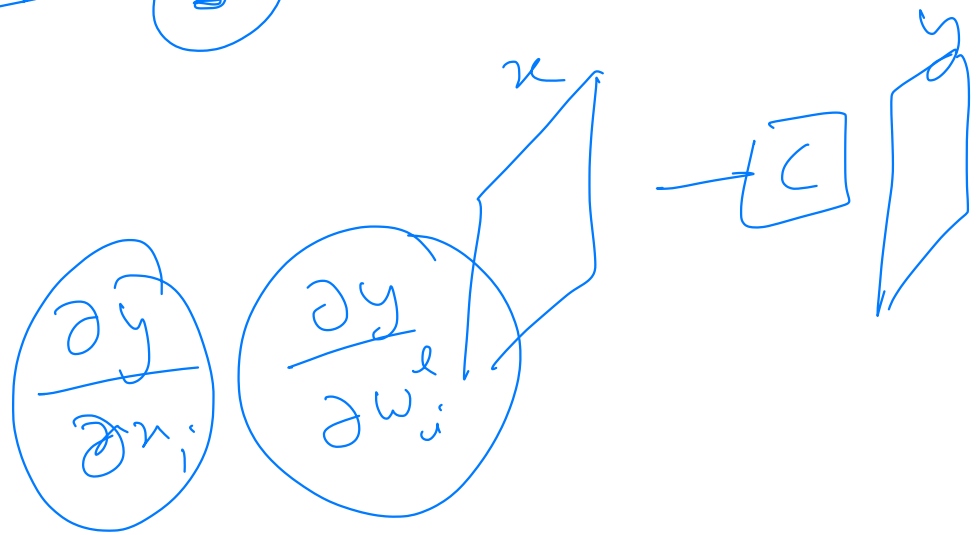
1. Decompose operations in layers of a neural network into function elements whose derivatives w.r.t. inputs are known by symbolic computation.

$$h_\theta(x) = \left(f^{(l_{max})} \circ \cdots \circ f^{(l)}_{\theta^{(l)}} \circ \cdots \circ f^{(2)}_{\theta^{(2)}} \circ f^{(1)} \right)(x)$$

where $f^{(1)} = x$ $f^{(l_{max})} = h_\theta$

Cost function / Sigmoid Softmax

and $\forall l: \frac{\partial f^{(l)}}{\partial f^{(l-1)}}$ is known



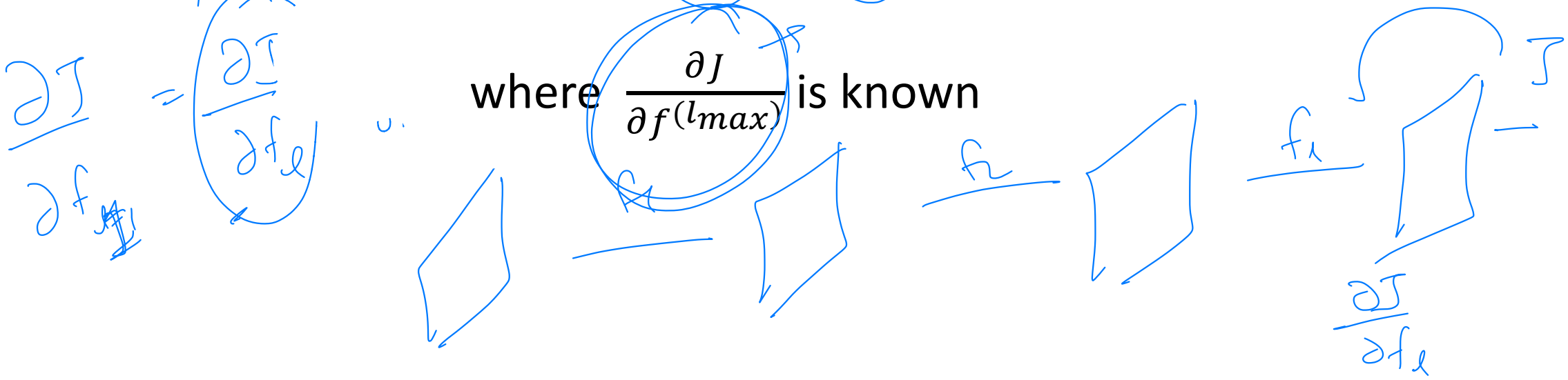
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

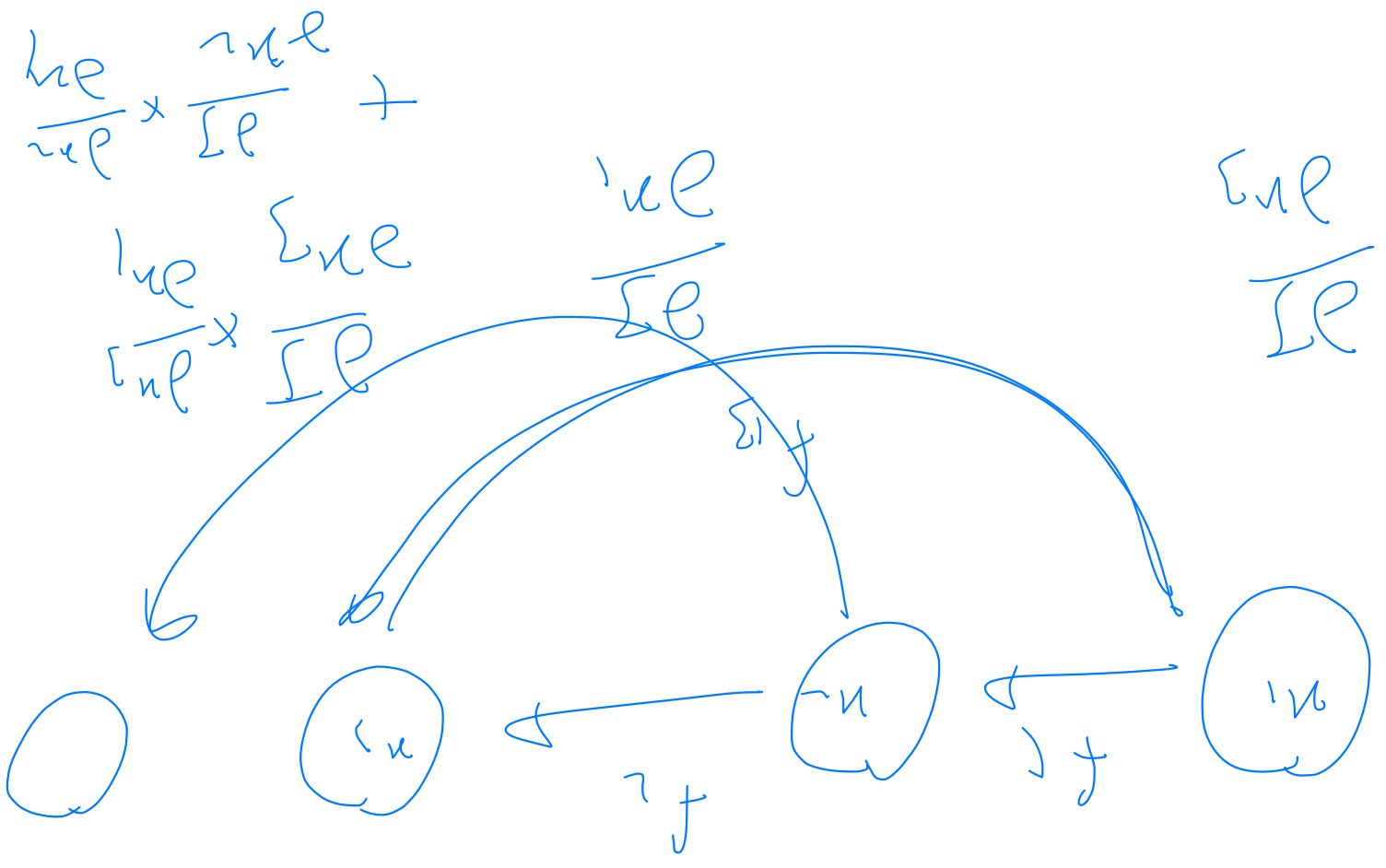
Backpropagation in General Cases

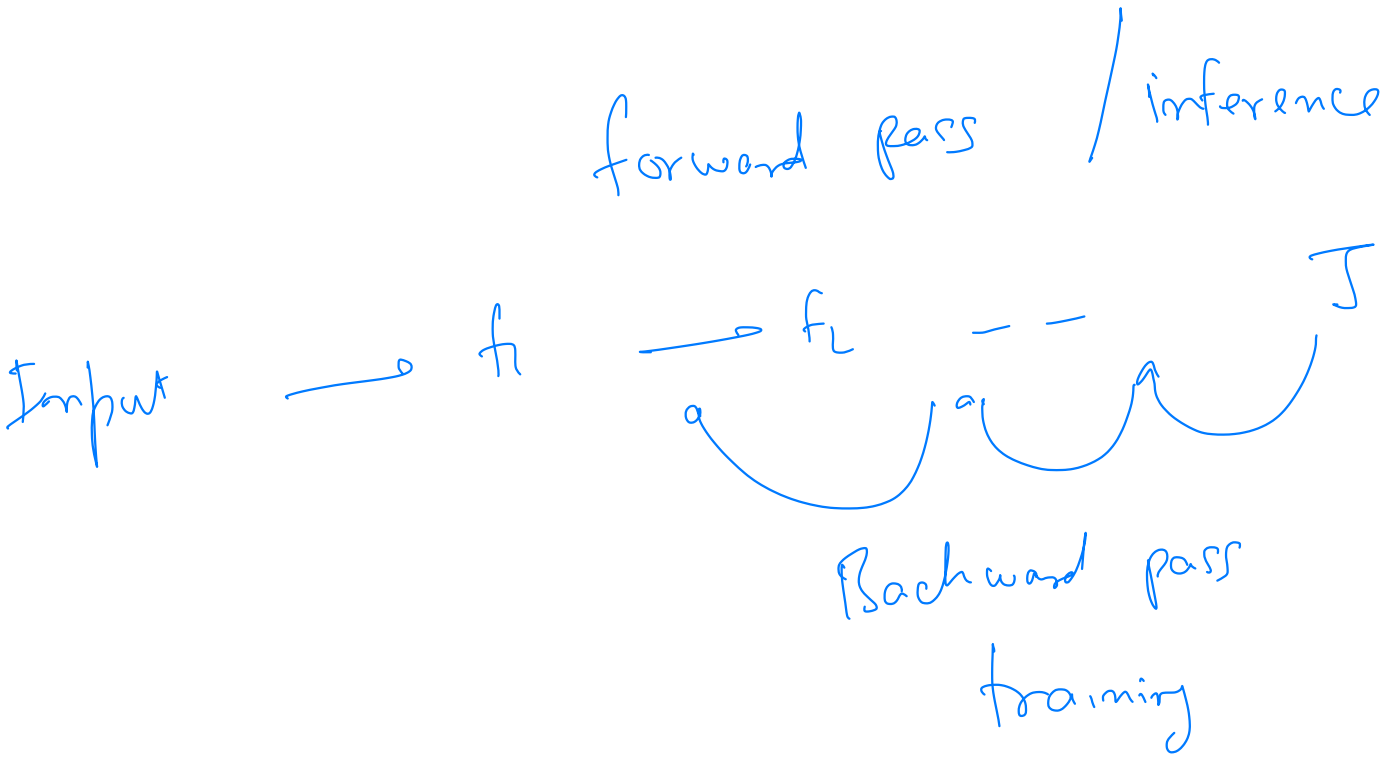
2. Backpropagate error signals corresponding to a differentiable cost function by numerical computation (Starting from cost function, plug in error signals backward).

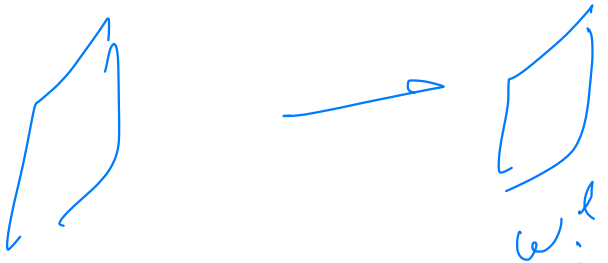
$$\delta^{(l)} = \frac{\partial}{\partial f^{(l)}} J(\theta; x, y) = \frac{\partial J}{\partial f^{(l+1)}} \frac{\partial f^{(l+1)}}{\partial f^{(l)}} = \delta^{(l+1)} \frac{\partial f^{(l+1)}}{\partial f^{(l)}}$$

where $\frac{\partial J}{\partial f^{(l_{max})}}$ is known






forward pass / inference
Input
f_1
f_L
J
Backward pass
training

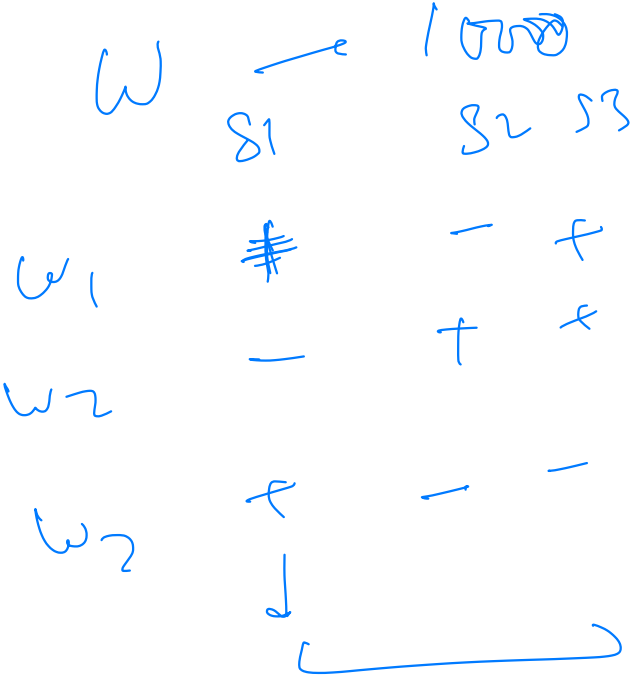


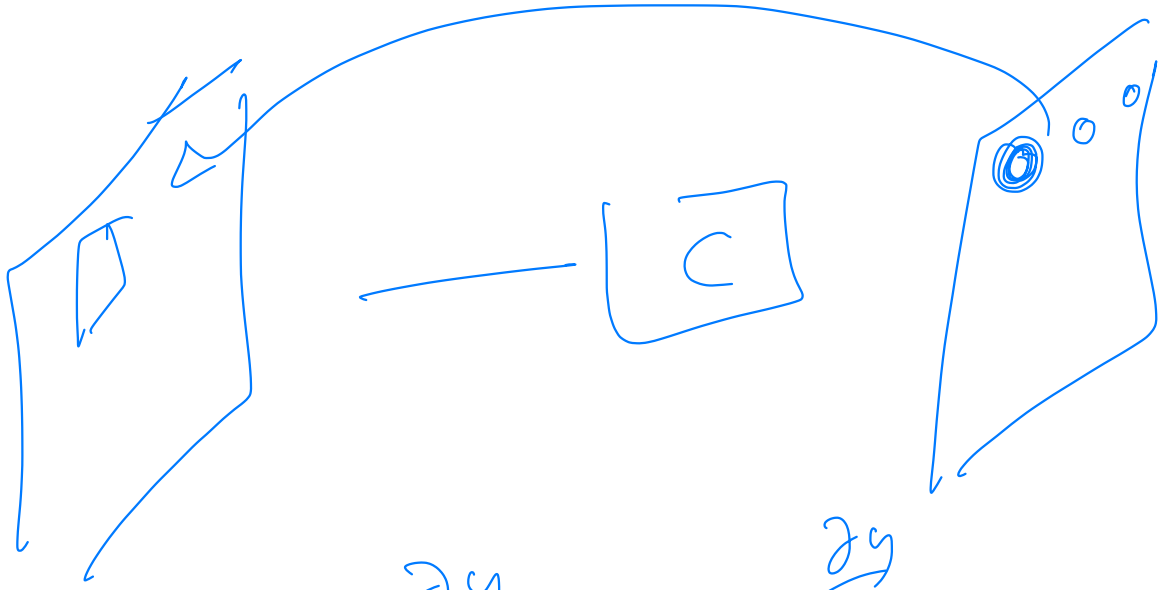
$$\sum_{k} \left(\frac{\partial I}{\partial w_i^{\ell}} \right)^{k}$$

mini-batches

Batch mode GD

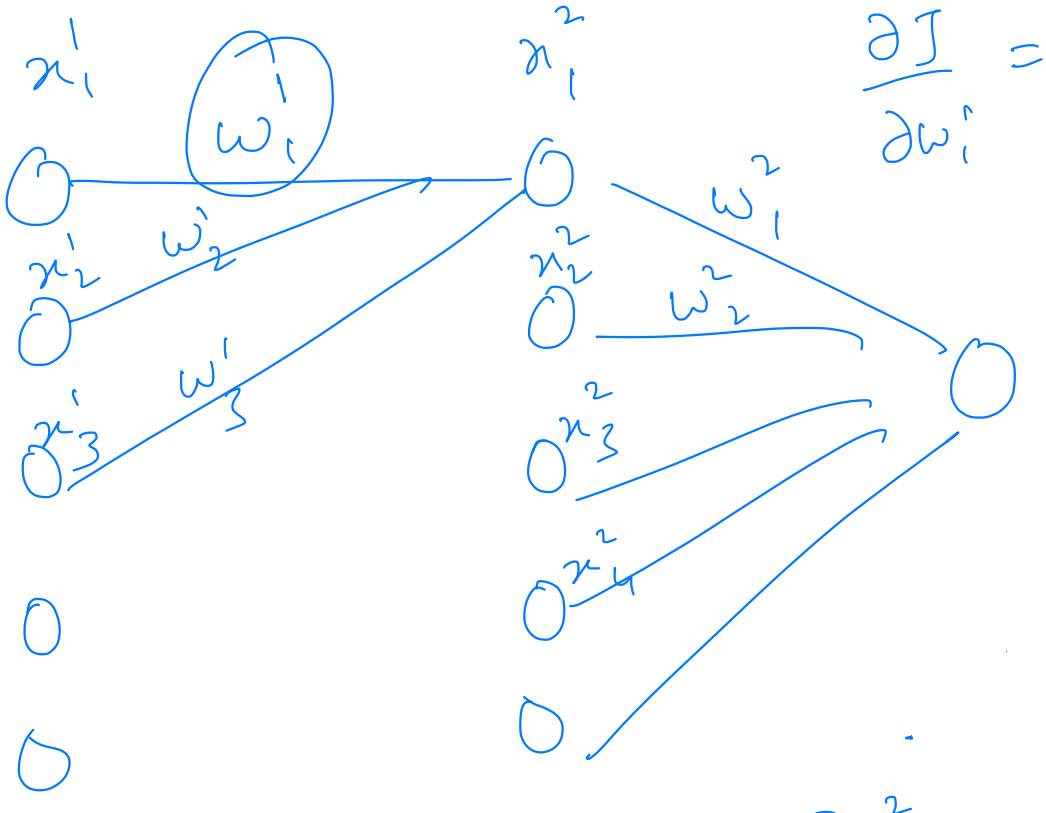
SGD





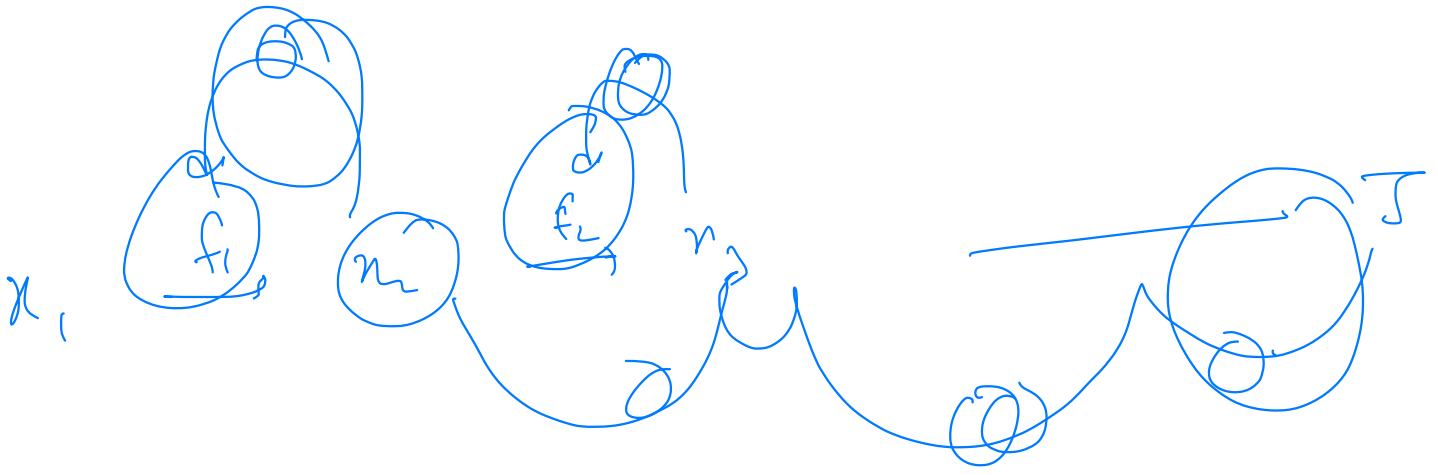
$\frac{\partial y}{\partial x_i}$ $\frac{\partial y}{\partial w_j}$

$y = \sum f(x)$ $S\left(\sum_{j \neq w} x_j\right)$



$$\frac{\partial J}{\partial w_1^1} = \frac{\partial J}{\partial x_1^2} \times \frac{\partial x_1^2}{\partial w_1^1}$$

$$\frac{\partial y}{\partial x_1^1} = \frac{\partial y}{\partial x_1^2} \times \frac{\partial x_1^2}{\partial x_1^1} + \frac{\partial y}{\partial x_2^2} \frac{\partial x_2^2}{\partial x_1^1}$$


x_1
f_1
n_2
d
f_2
n_3
J

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Backpropagation in General Cases

3. Use back-propagated error signals to compute gradients w.r.t. parameters only for the function elements with parameters where their derivatives w.r.t. parameters are known by symbolic computation.

$$\nabla_{\theta^{(l)}} J(\theta; x, y) = \frac{\partial}{\partial \theta^{(l)}} J(\theta; x, y) = \frac{\partial J}{\partial f^{(l)}} \frac{\partial f^{(l)}_{\theta^{(l)}}}{\partial \theta^{(l)}} = \delta^{(l)} \frac{\partial f^{(l)}_{\theta^{(l)}}}{\partial \theta^{(l)}}$$

where $\frac{\partial f^{(l)}_{\theta^{(l)}}}{\partial \theta^{(l)}}$ is known

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Backpropagation in General Cases

4. Sum gradients over all example to get overall gradient.

$$\nabla_{\theta^{(l)}} J(\theta) = \sum_{i=1}^{m} \nabla_{\theta^{(l)}} J\left(\theta; x^{(i)}, y^{(i)}\right)$$

$w \leftarrow w - \eta \nabla J$

Learning Rate

Output of penultimate layer = $[a_1, \ldots a_i, \ldots a_k]$ k = # classes

Also called "logits" (we will soon describe why)

Softmax $P_\Lambda[y = i \mid a] = \dfrac{e^{a_i}}{\sum_{j=0}^{k} e^{a_j}}$ denoted as "Σ"

Sigmoid / Logistic function $= \dfrac{1}{1 + e^{-z}}$

$=$ Softmax for 2 classes.

$P_\Lambda[y = 0 \mid a] = \dfrac{e^{a_0}}{e^{a_0} + e^{a_1}} = \dfrac{1}{1 + e^{-(a_0 - a_1)}} = \dfrac{1}{1 + e^{-z}}$

$e^{a_0} + e^{a_1}$: Softmax

Output of penultimate layer = $[a_1, \ldots a_i, \ldots a_k]$ k = # classes

Also called "logits" (we will soon describe why)

Softmax $P_\Lambda[y = i \mid a] = \dfrac{e^{a_i}}{\sum_{j=0}^{k} e^{a_j}}$ denoted as "Σ"

Sigmoid / Logistic function $= \dfrac{1}{1 + e^{-z}}$

$=$ Softmax for 2 classes.

$$P_\Lambda[y = 0 \mid a] = \underbrace{\frac{e^{a_0}}{e^{a_0} + e^{a_1}}}_{\text{Softmax}} = \frac{1}{1 + e^{-(a_0 - a_1)}} = \frac{1}{1 + e^{-z}}$$

Alternate form of softmax

$$Pr[y=i|a] = \frac{e^{+a_i}}{\sum_j e^{+a_j}}$$

If we add a constant c to all the elements of "a" vector

$$Pr[y=i|a] = \frac{e^{+(a_i+c)}}{\sum e^{(a_j+c)}} = \frac{e^c \times e^{a_i}}{e^c \sum_j e^{a_j}} = \frac{e^{a_i}}{\sum_j e^{a_j}}$$

Hence we may choose the K^{th} logit as anchor and choose c such that $e^{a_k+c} = e^0 = 1$ which makes the other $K-1$ logits as independent.

$$Pr[y=i|a] = \frac{e^{a_i}}{1 + \sum_{j=1}^{K-1} e^{a_j}}$$

Logits = Log odds

- Imagine running $K-1$ independent binary logistic regression models
- Penultimate layer output = $[a_1 \dots a_i, \dots a_K]$
- Choose K^{th} output as the pivot

$$\text{log odds} = \log\left(\frac{P_r(y=i)}{P_r(y=K)}\right) = a_i$$

$$P_r(y=i) = e^{a_i} P(y=K)$$

We know that

$$P_n(y=K) = 1 - \sum_{i=1}^{K-1} P_n(y=i) = 1 - \sum_{i=1}^{K-1} P_n(y=K)\, e^{a_i}$$

$$\Rightarrow \quad P_n(y=K) = \frac{1}{1 + \sum_{i=1}^{K-1} e^{a_i}}$$

$$\Rightarrow \quad P_n(y=i) = \frac{e^{a_i}}{1 + \sum_{j=1}^{K-1} e^{a_j}}$$

Gradient of Softmax with respect to logits

$$\frac{\partial S_i}{\partial a_j} = \frac{\partial \frac{e^{a_i}}{\Sigma e^{a_j}}}{\partial a_j} = \frac{\partial \frac{e^{a_i}}{\Sigma}}{\partial a_j}$$

Let $g = e^{a_i}$, $h = \Sigma$

if $f = \frac{g}{h} \Rightarrow f' = \frac{g'h - h'g}{h^2}$

$$\frac{\partial S_i}{\partial a_i} = \frac{e^{a_i}\Sigma - e^{a_i}e^{a_i}}{\Sigma \times \Sigma} = \frac{e^{a_i}}{\Sigma} - \frac{e^{a_i}}{\Sigma} \times \frac{e^{a_i}}{\Sigma}$$

$$\frac{\partial s_i}{\partial a_i} = \frac{e^{a_i}}{\Sigma}\left(1 - \frac{e^{a_i}}{\Sigma}\right) = s_i(1 - s_i)$$

$g' = 0$

for $\frac{\partial s_i}{\partial a_j}$

hence $$\frac{\partial s_i}{\partial a_j} = \frac{0 - e^{a_j} e^{a_i}}{\Sigma^2} = \frac{-e^{a_j}}{\Sigma} \times \frac{e^{a_i}}{\Sigma}$$

$$= -s_j s_i$$

Cross Entropy

If Q is the target probability & P is the predicted probability

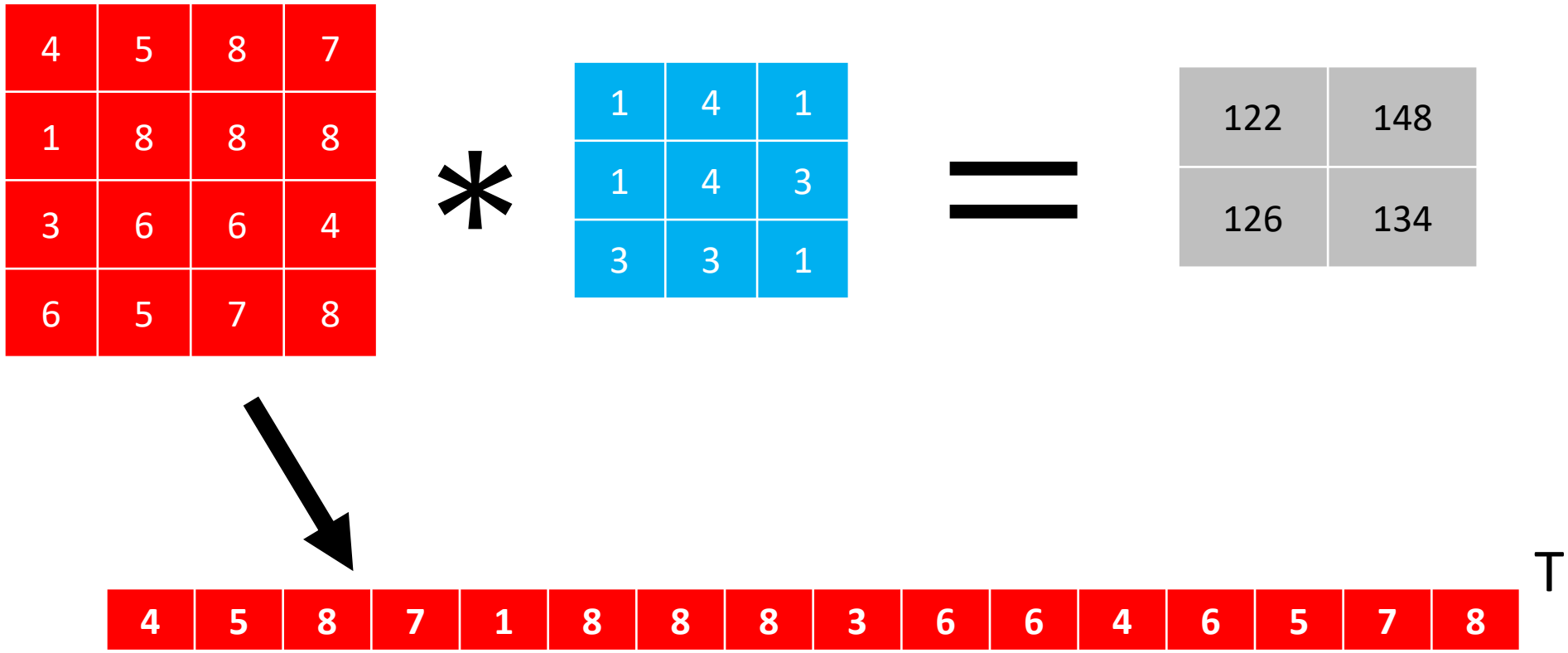
$$CE(P, Q) = -\sum_{i=1}^{n} q_i \log(p_i)$$

Recall that $\lim_{x \to 0} x \log x = \lim_{x \to 0} \frac{\log x}{1/x}$

Using L'Hospital Rule $\lim_{x \to 0} \frac{1/x}{-1/x^2} = 0$

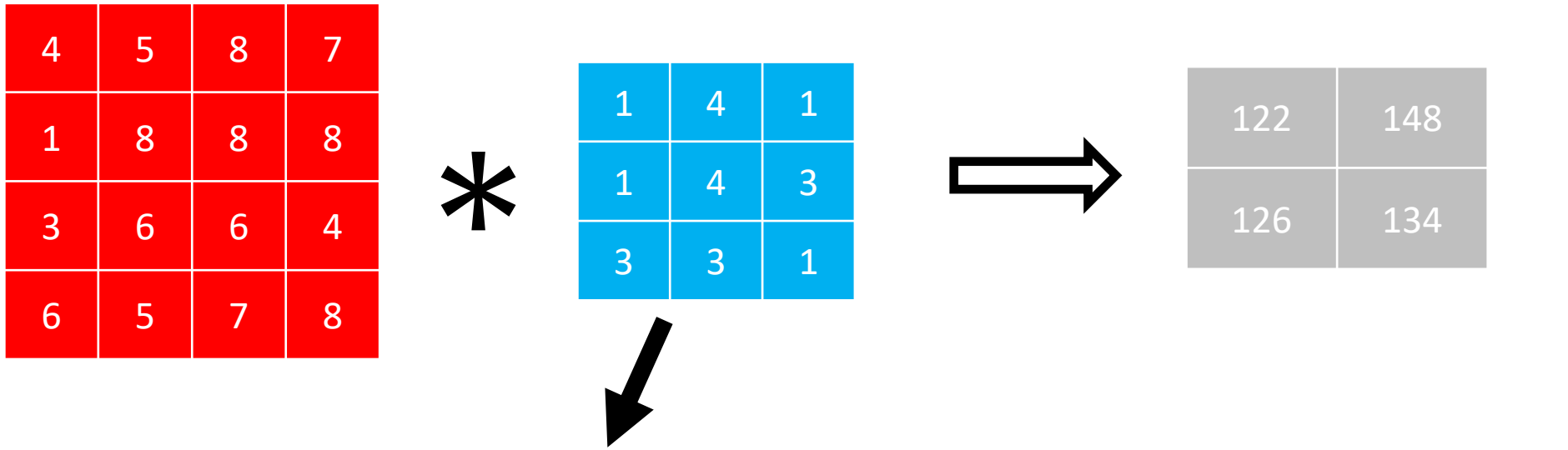
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Convolution as Matrix Multiplication



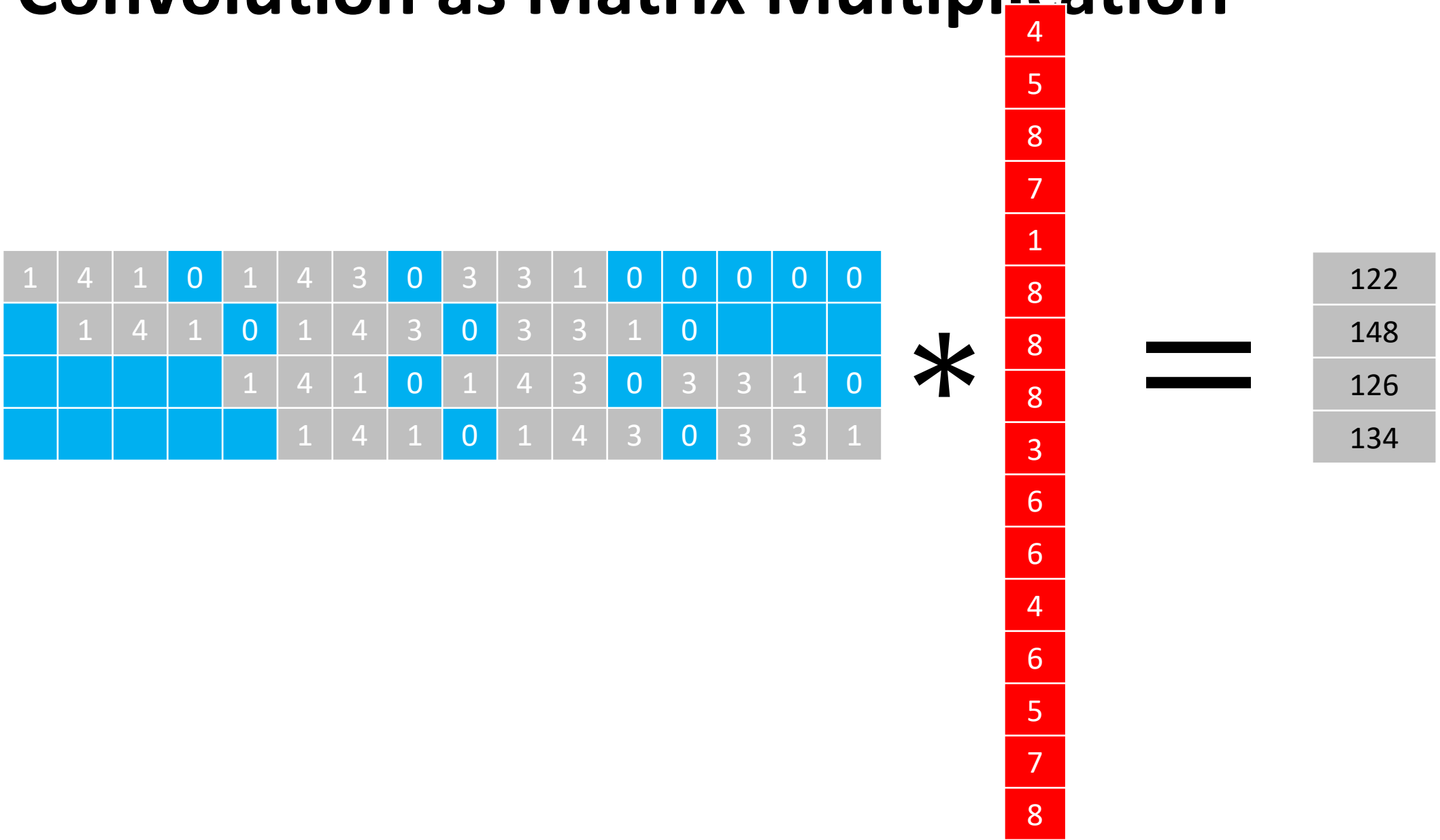
Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Convolution as Matrix Multiplication



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Convolution as Matrix Multiplication



Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Backpropagation in Convolution Layer

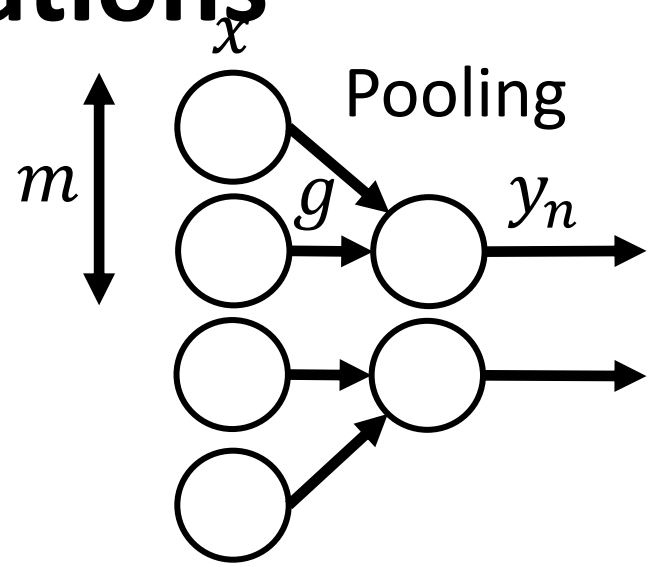
Forward propagation $x * w = y$

Backward propagation $\frac{\partial J}{\partial x} = flip(w) * \frac{\partial J}{\partial y}$

Gradient computation $x * \frac{\partial J}{\partial y} = \frac{\partial J}{\partial w}$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Pooling Layer: Subsampling Equations



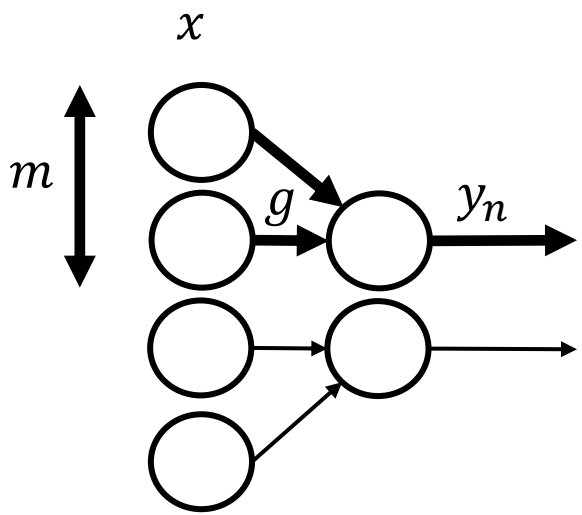
Mean Pooling $g(x) = \frac{\sum_{k=1}^{m} x_k}{m}$ $\frac{\partial g}{\partial x} = \frac{1}{m}$

Max Pooling $\mathrm{g(x)} = \max(x)$ $\frac{\partial g}{\partial x_i} = \begin{cases} 1 & if\ x_i = \max(x) \\ 0 & otherwise \end{cases}$

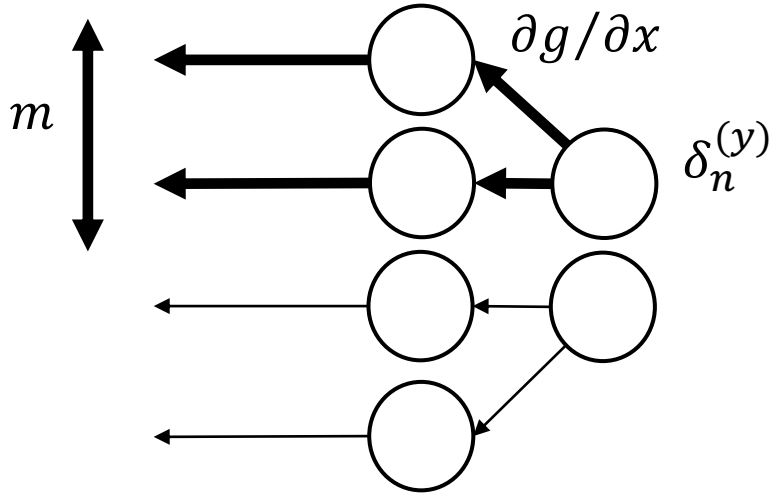
L^p Pooling $g(x) = \|x\|_p = \left(\sum_{k=1}^{m} |x_k|^P \right)^{\frac{1}{P}}$ $\frac{\partial g}{\partial x_i} = \left(\sum_{k=1}^{m} |x_k|^P \right)^{\frac{1}{P}-1} . |x_i|^{p-1}$

Chetan Arora
Department of Computer Science and Engineering, IIT Delhi

Backpropagation in the Pooling Layer



Forward propagation (subsampling)



Backward propagation (upsampling)