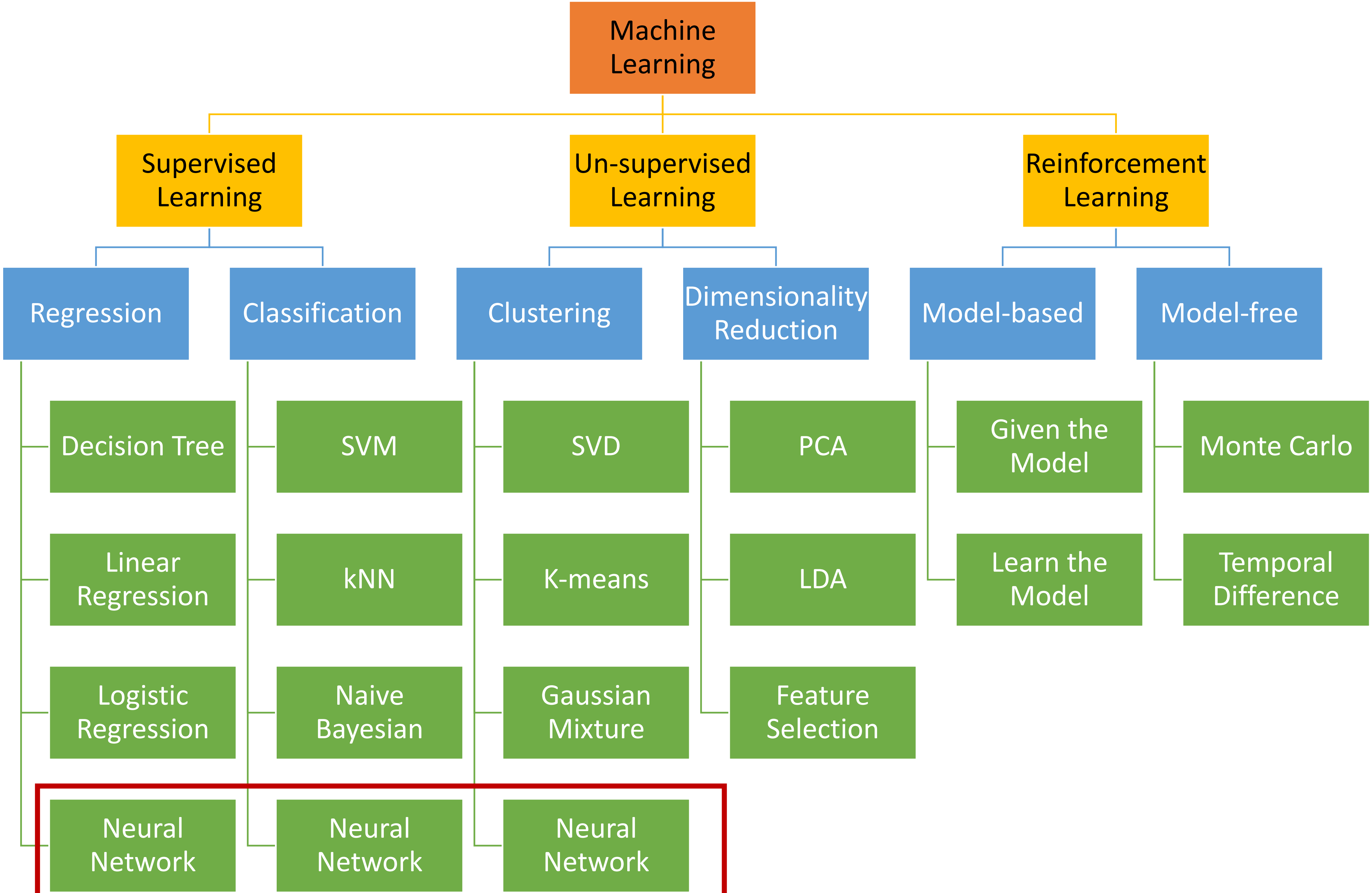
Summer Bootcamp 2021

Applied Machine Learning
Convolutional Neural Networks

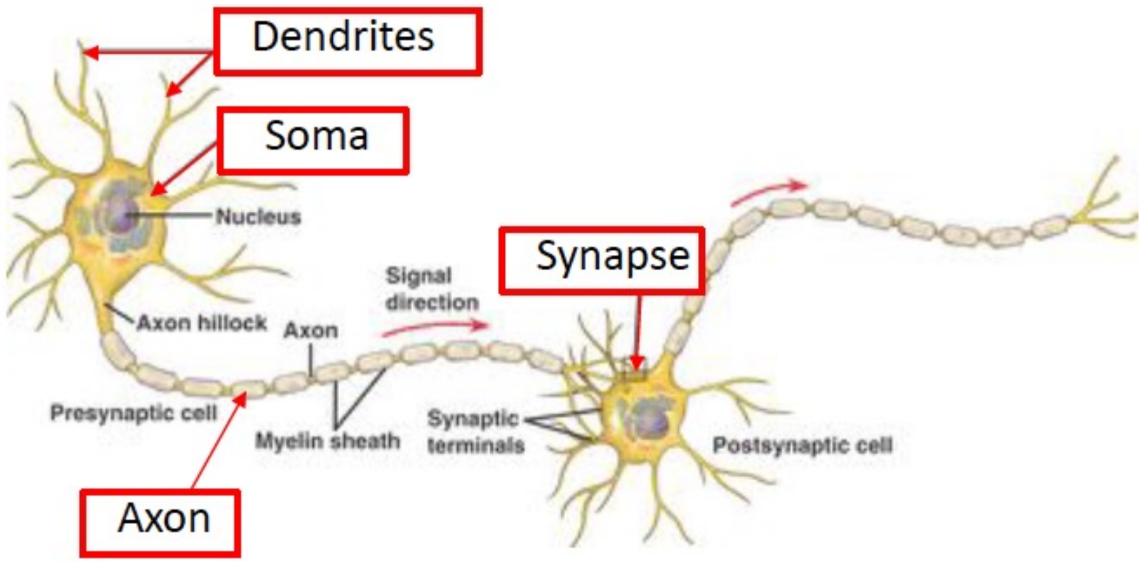
Ngan Le

thile@uark.edu


Machine Learning
Supervised Learning
Un-supervised Learning
Reinforcement Learning
Regression
Classification
Clustering
Dimensionality Reduction
Model-based
Model-free
Decision Tree
Linear Regression
Logistic Regression
Neural Network
SVM
kNN
Naive Bayesian
Neural Network
SVD
K-means
Gaussian Mixture
Neural Network
PCA
LDA
Feature Selection
Given the Model
Learn the Model
Monte Carlo
Temporal Difference

Perceptron

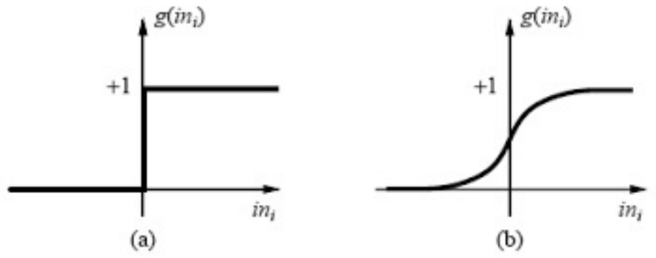
- A Neuron:



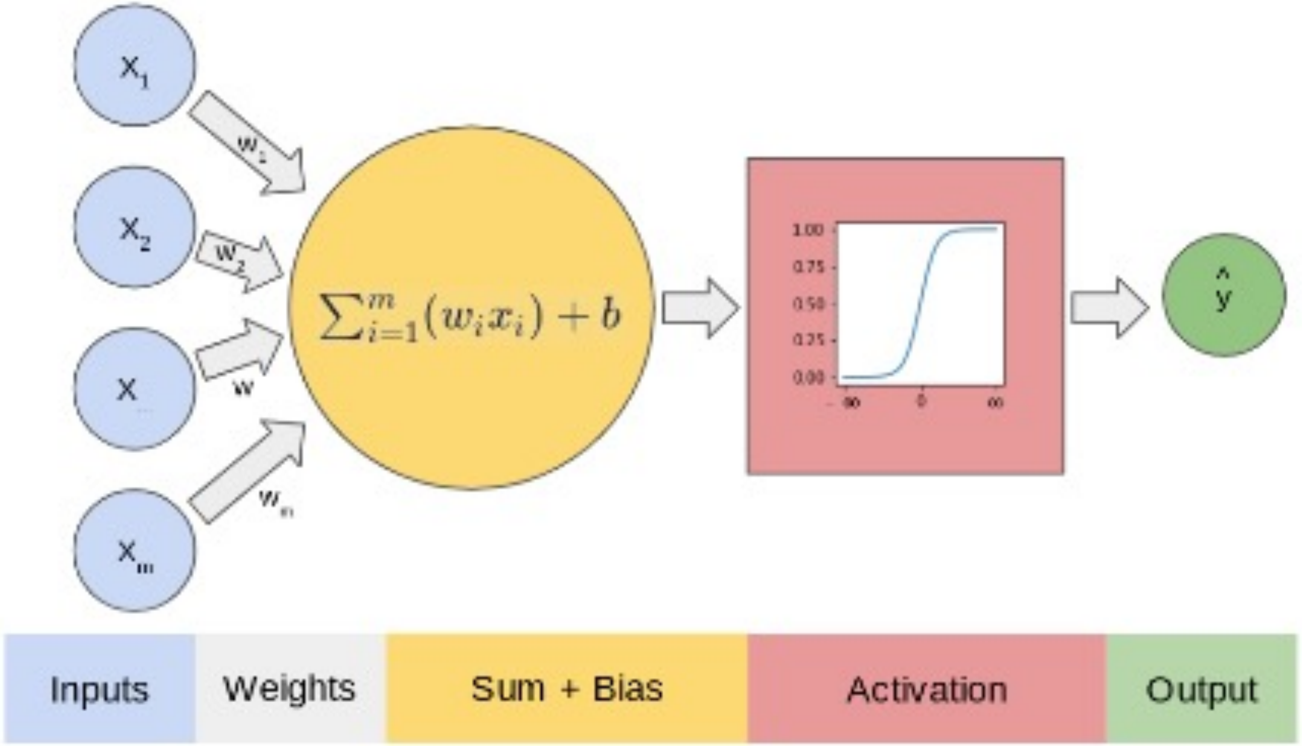
If two neurons consistently fire simultaneously, synaptic connection is increased, (if firing at different time, strength is reduced)

Cells that fire together, wire together

Perceptron

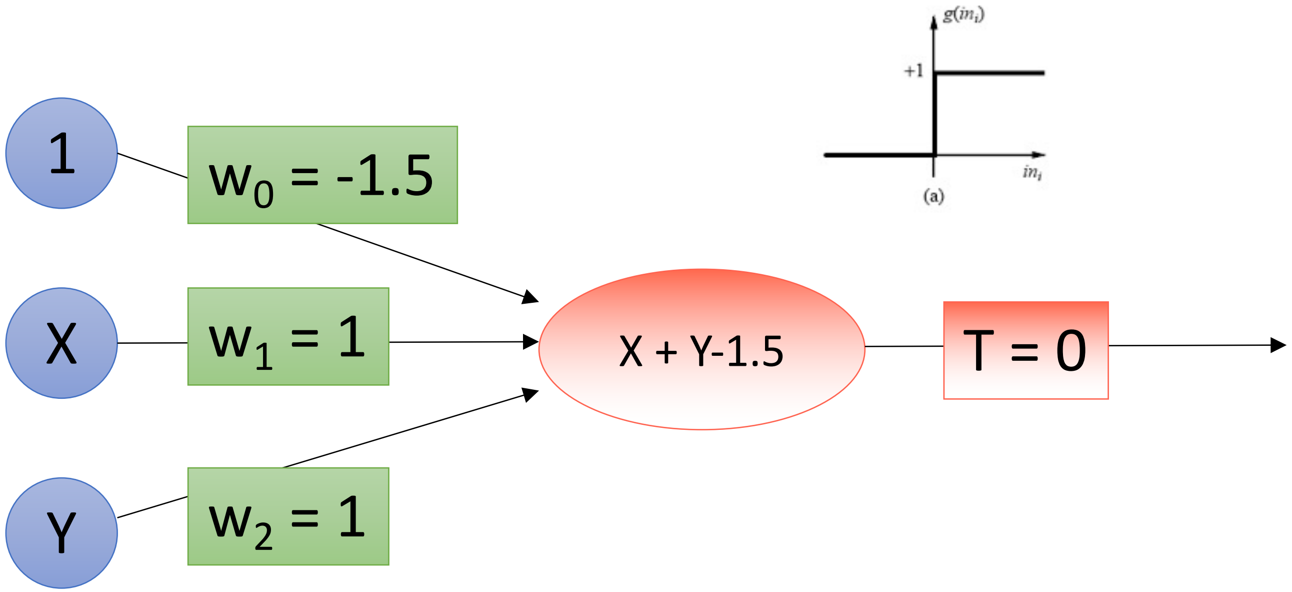


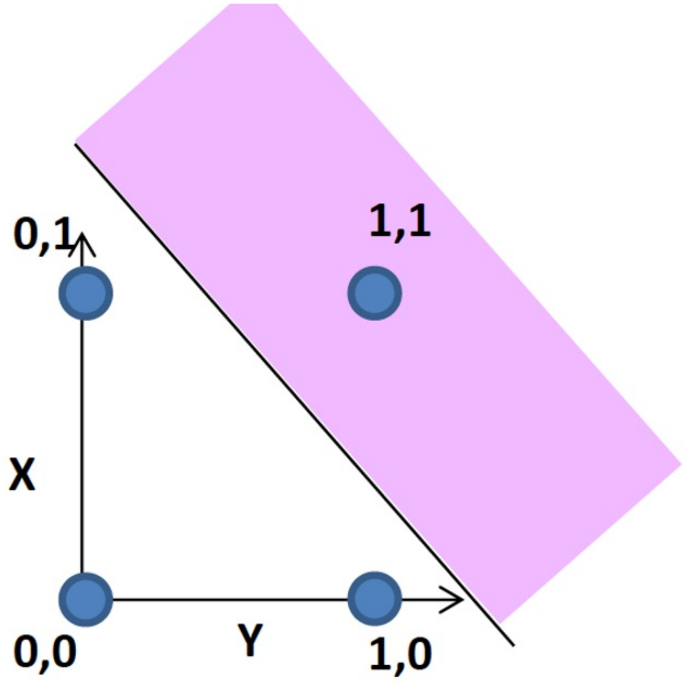
g(in_i)
+1
in_i
(a)
g(in_i)
+1
in_i
(b)



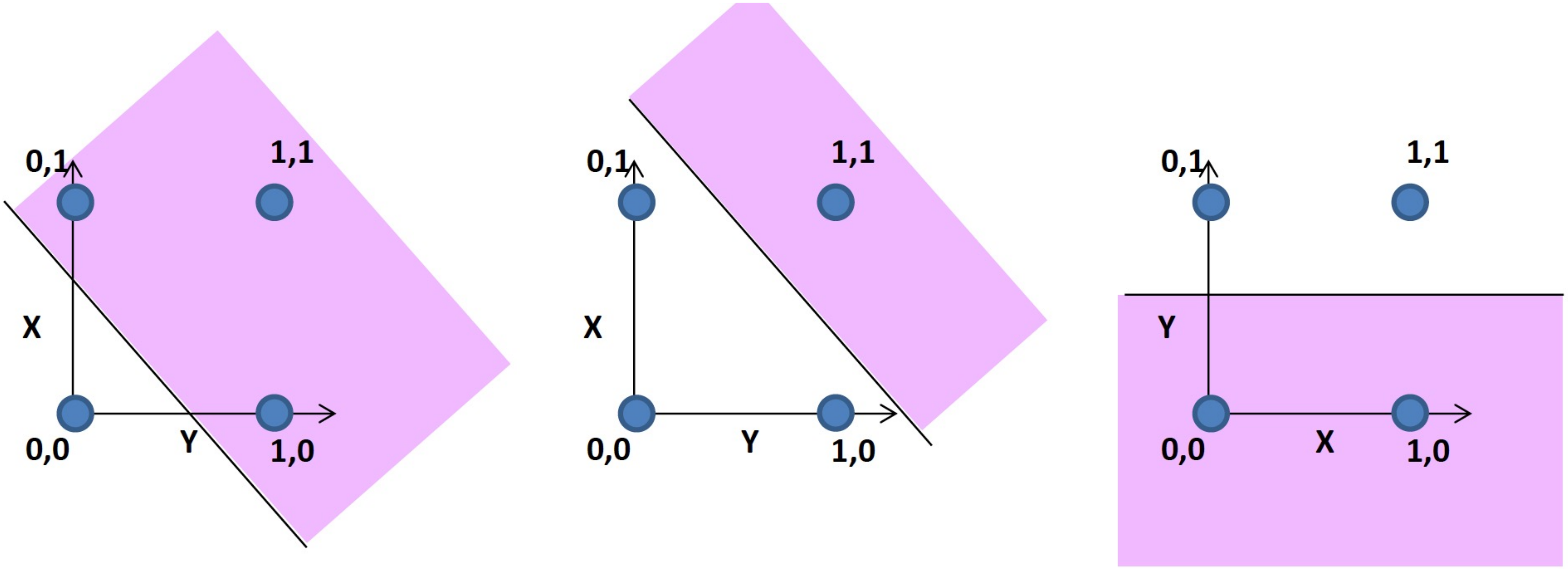
X1
X2
X
Xm
W1
W2
W
Wm
$\sum_{i=1}^{m}(w_i x_i) + b$
1.00
0.75
0.50
0.25
0.00
0
ŷ
Inputs
Weights
Sum + Bias
Activation
Output

Perceptron



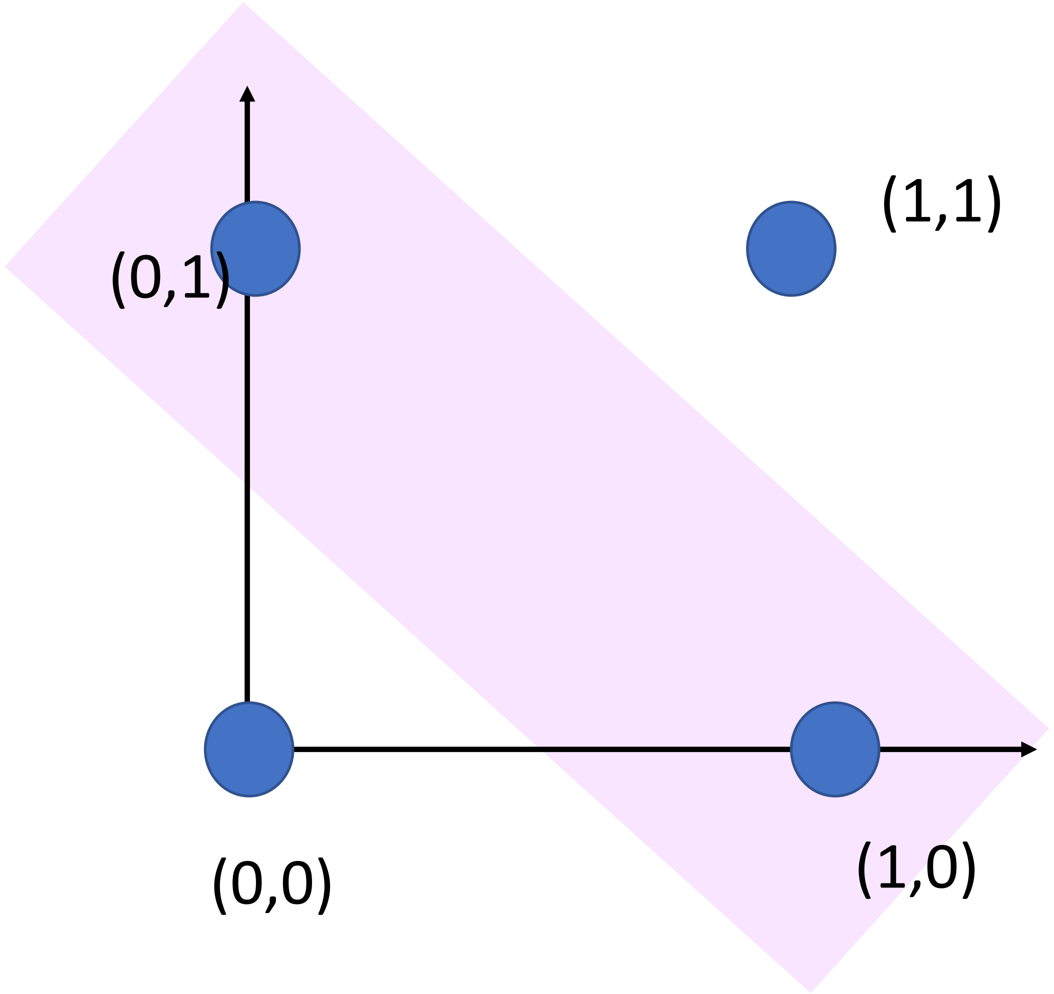


Perceptron



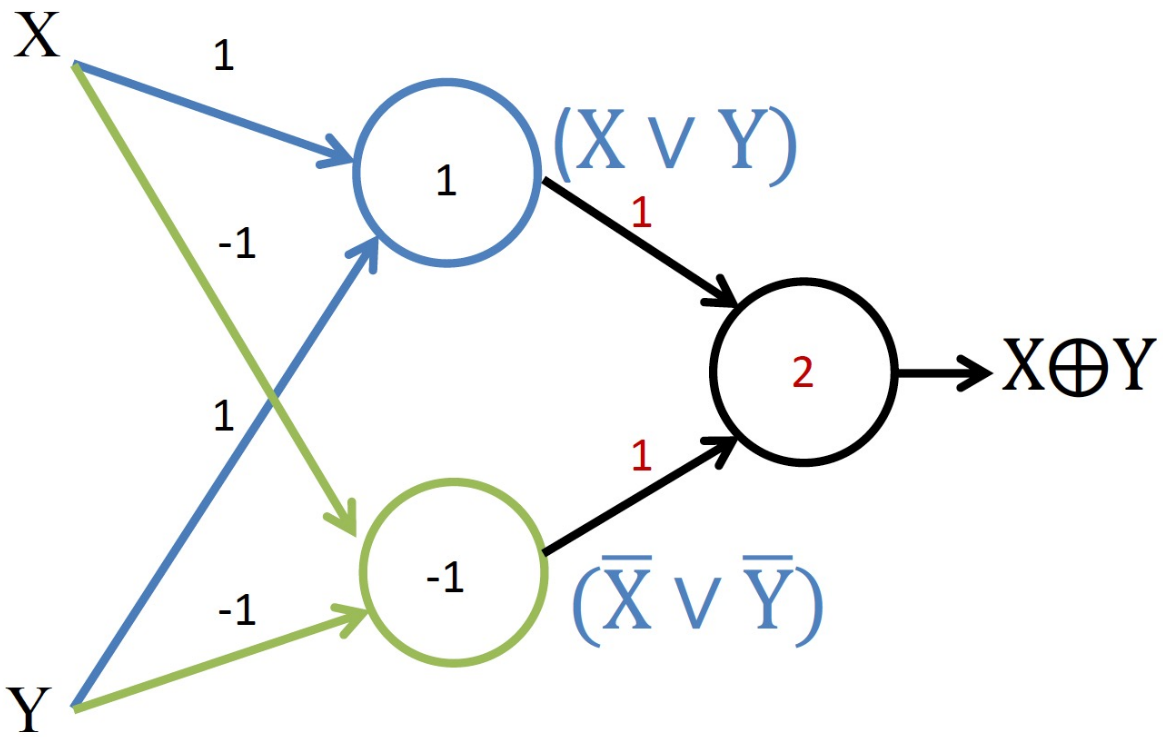
Neuron - Perceptron

x_1	x_2	x_1 XOR x_2
0	0	0
0	1	1
1	0	1
1	1	0

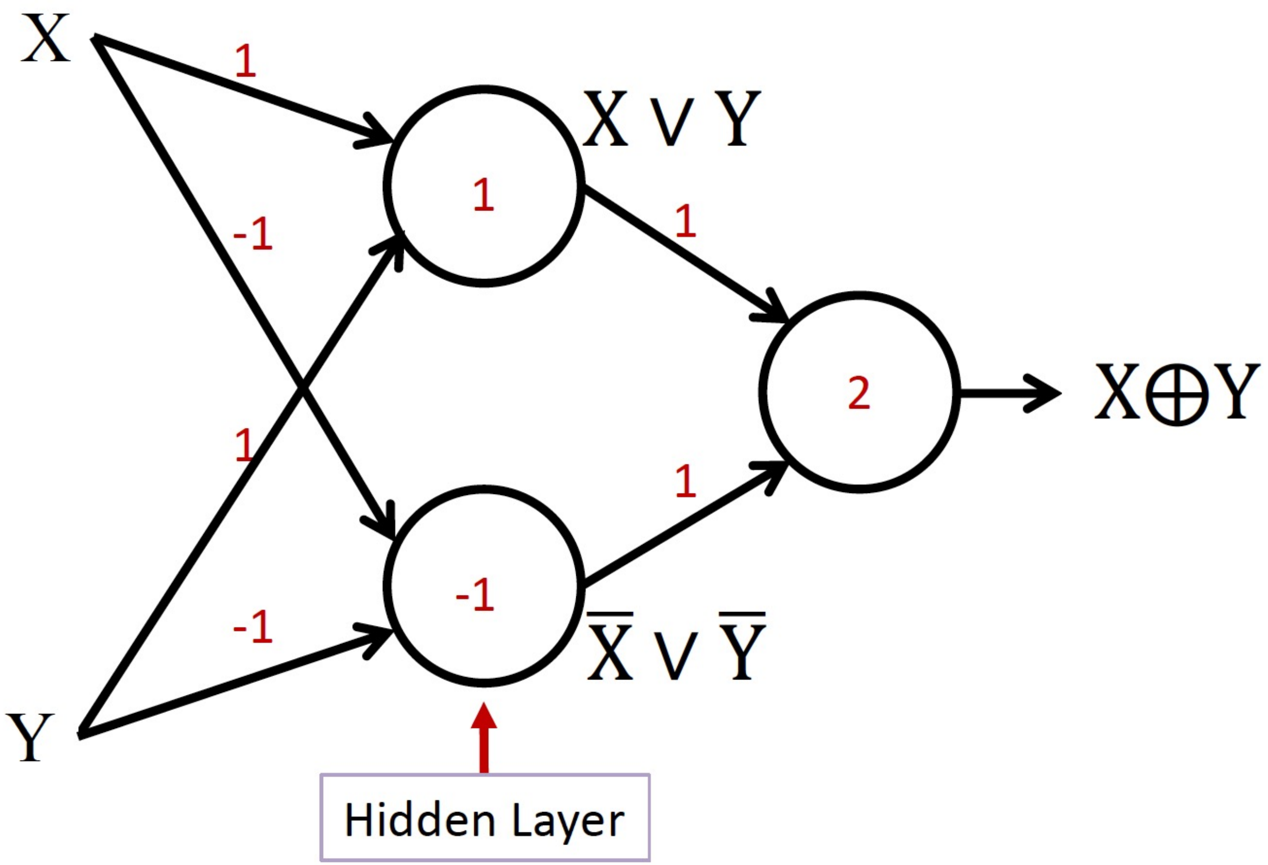


Multilayer Perceptron (MLP)

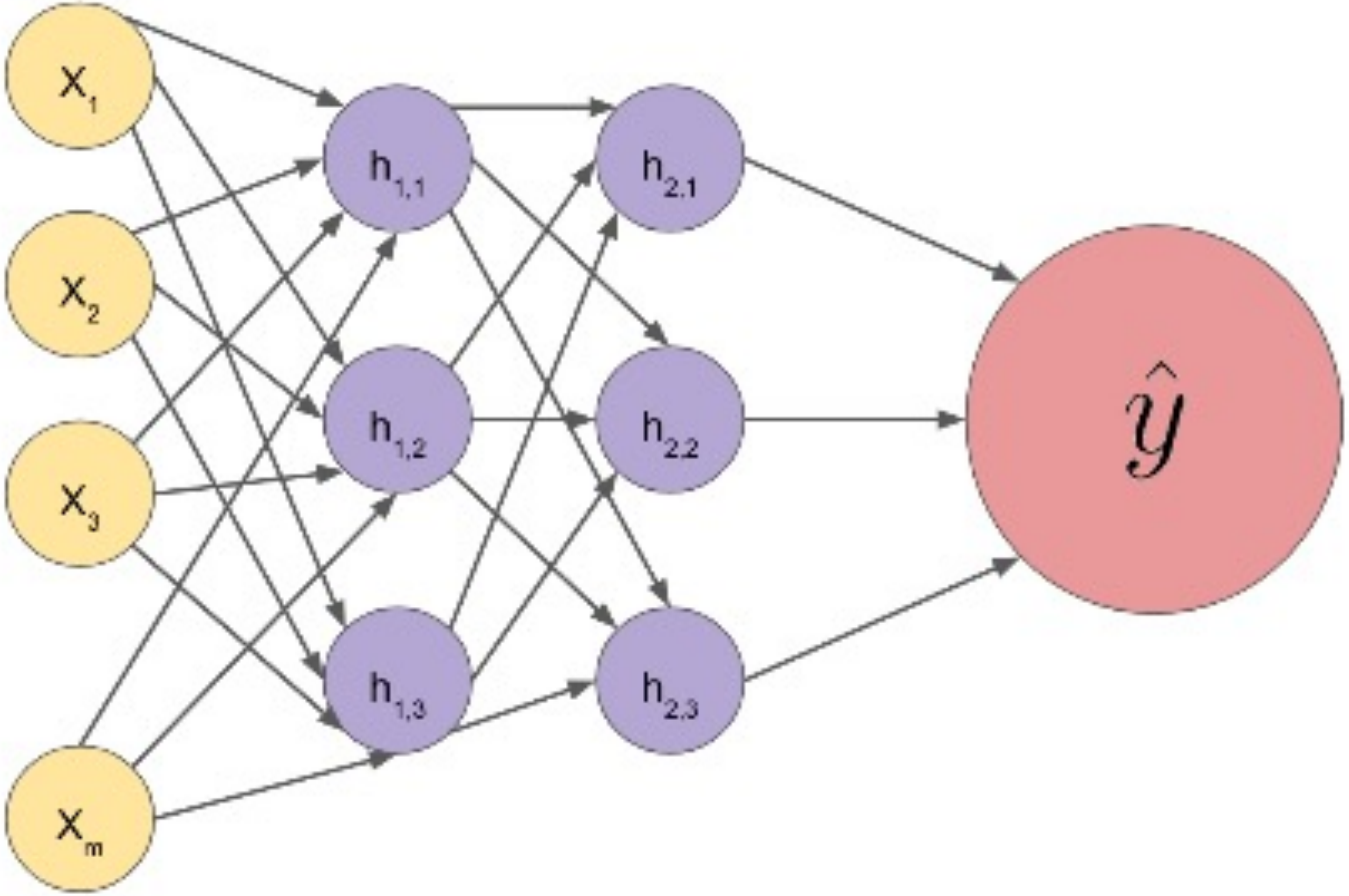
$$X \oplus Y = (X \vee Y) \wedge (\bar{X} \vee \bar{Y})$$



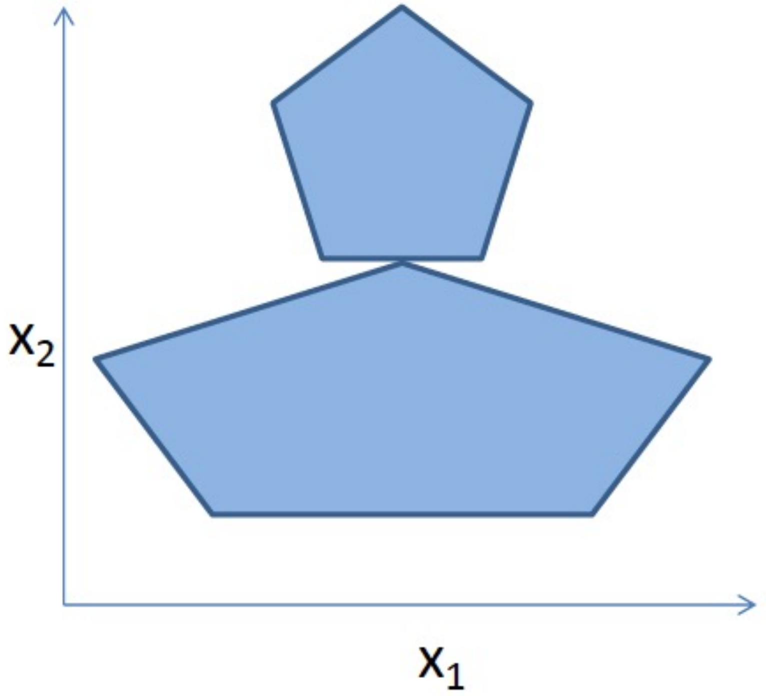
Multilayer Perceptron (MLP)

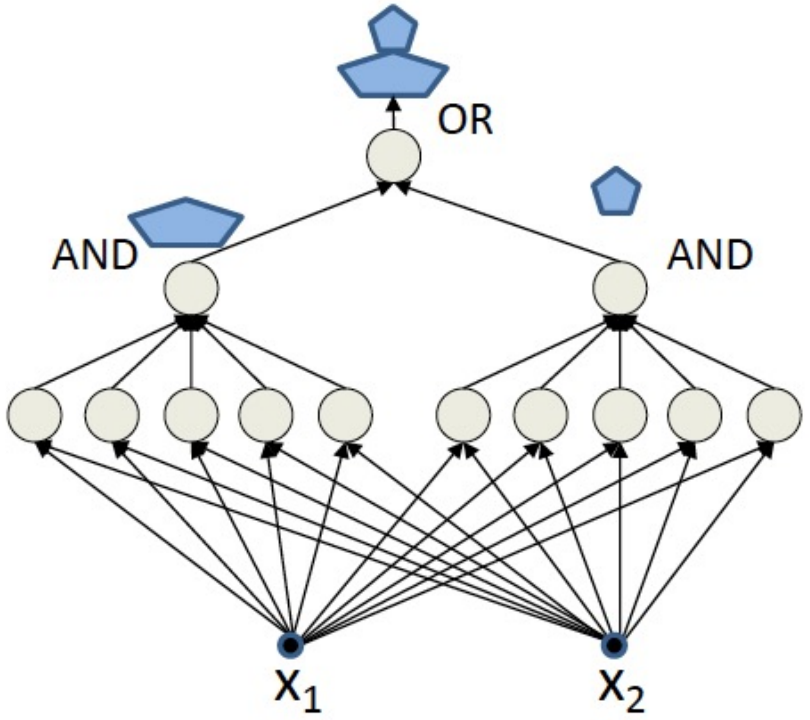


Neuron with Hidden Layers

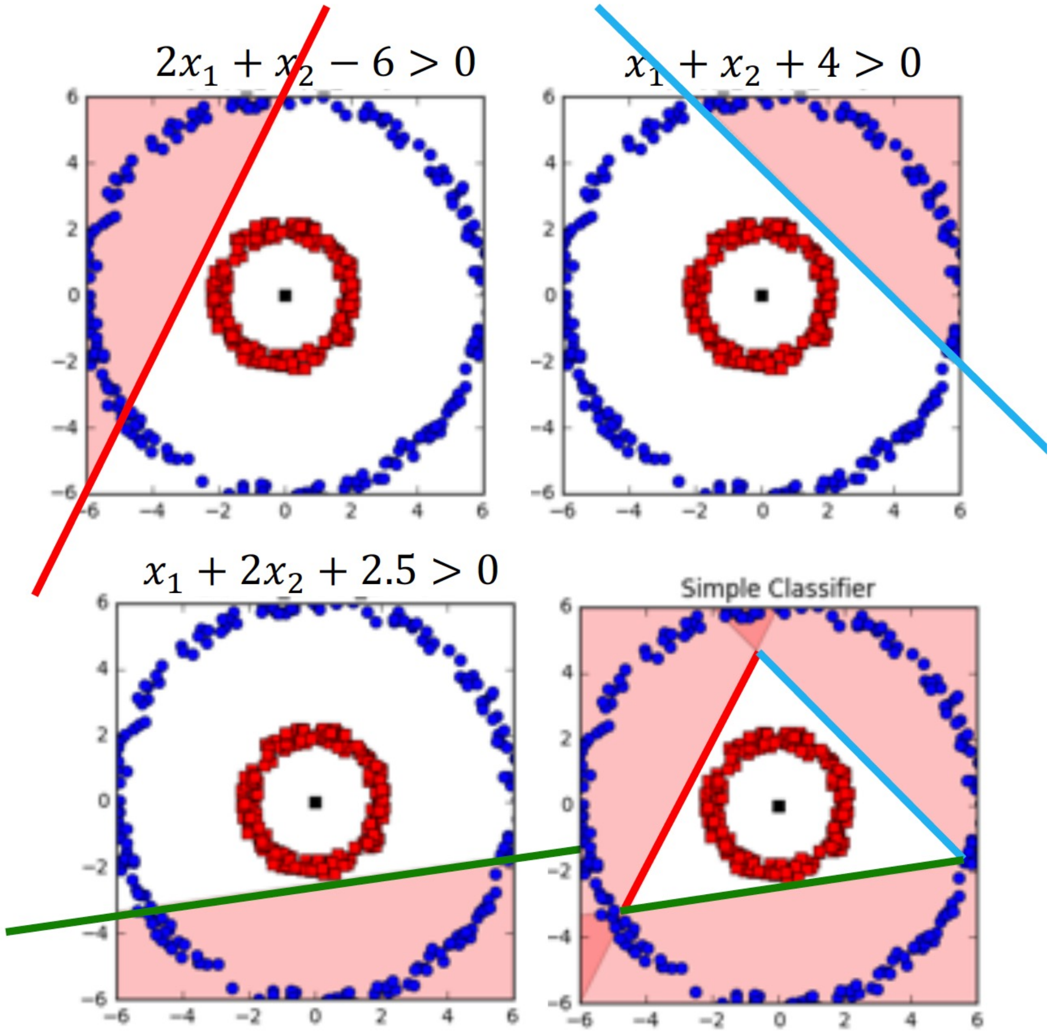


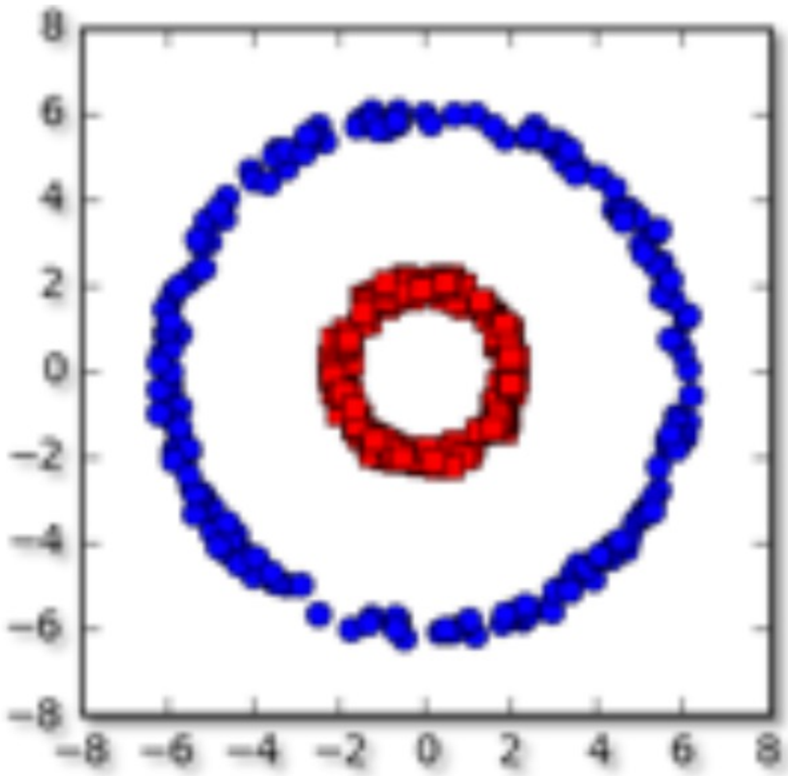
Neuron with Hidden Layers

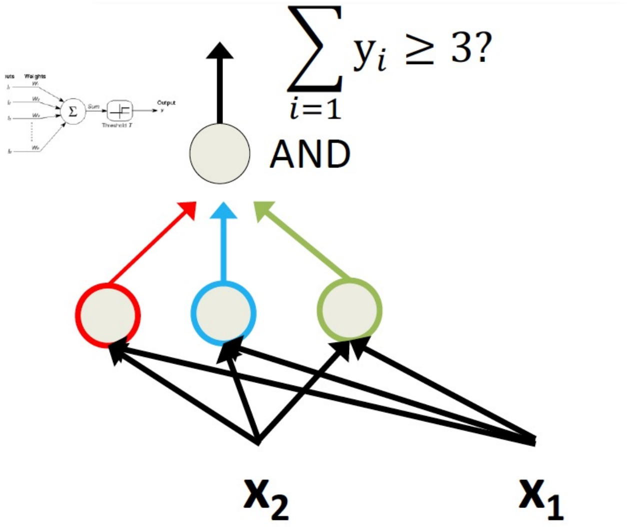




MLP for Classification

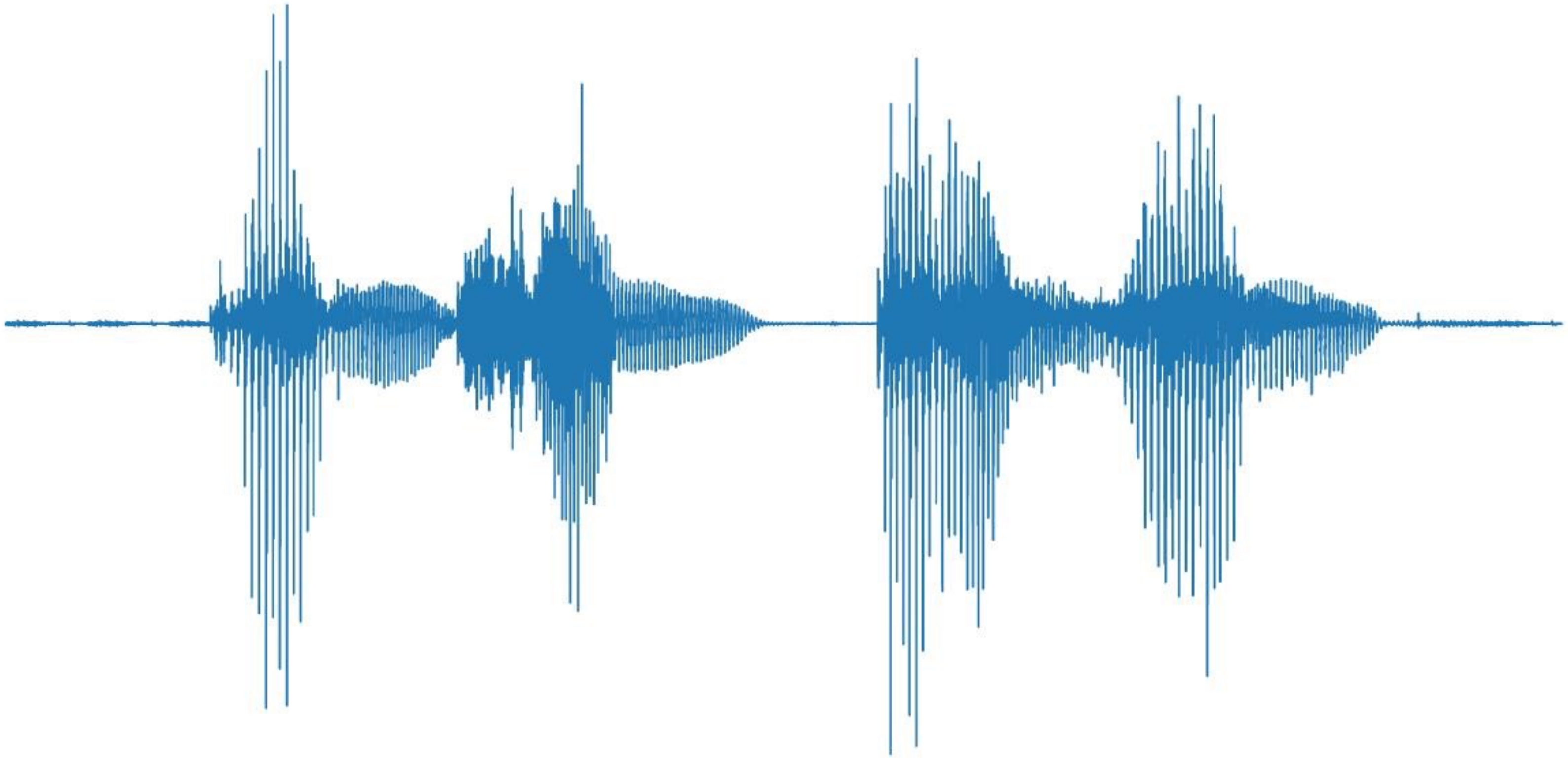






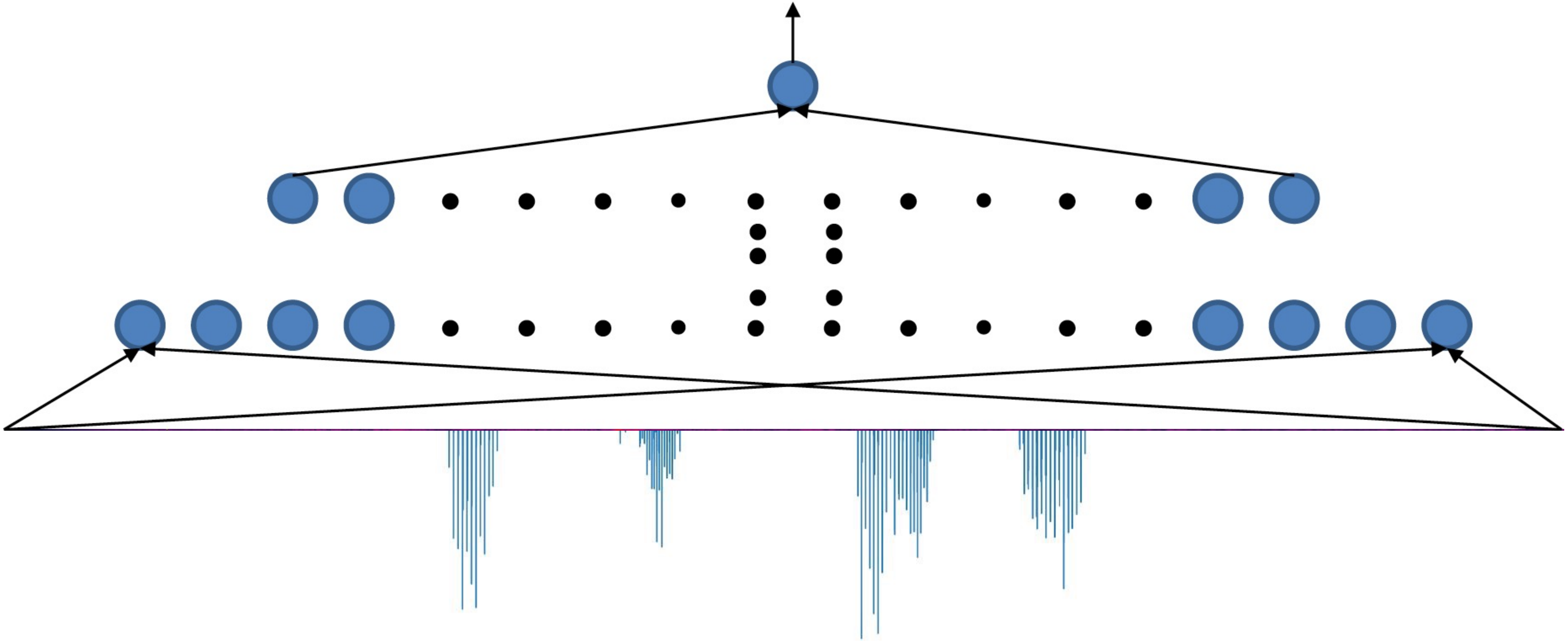
Search Pattern

Search Pattern

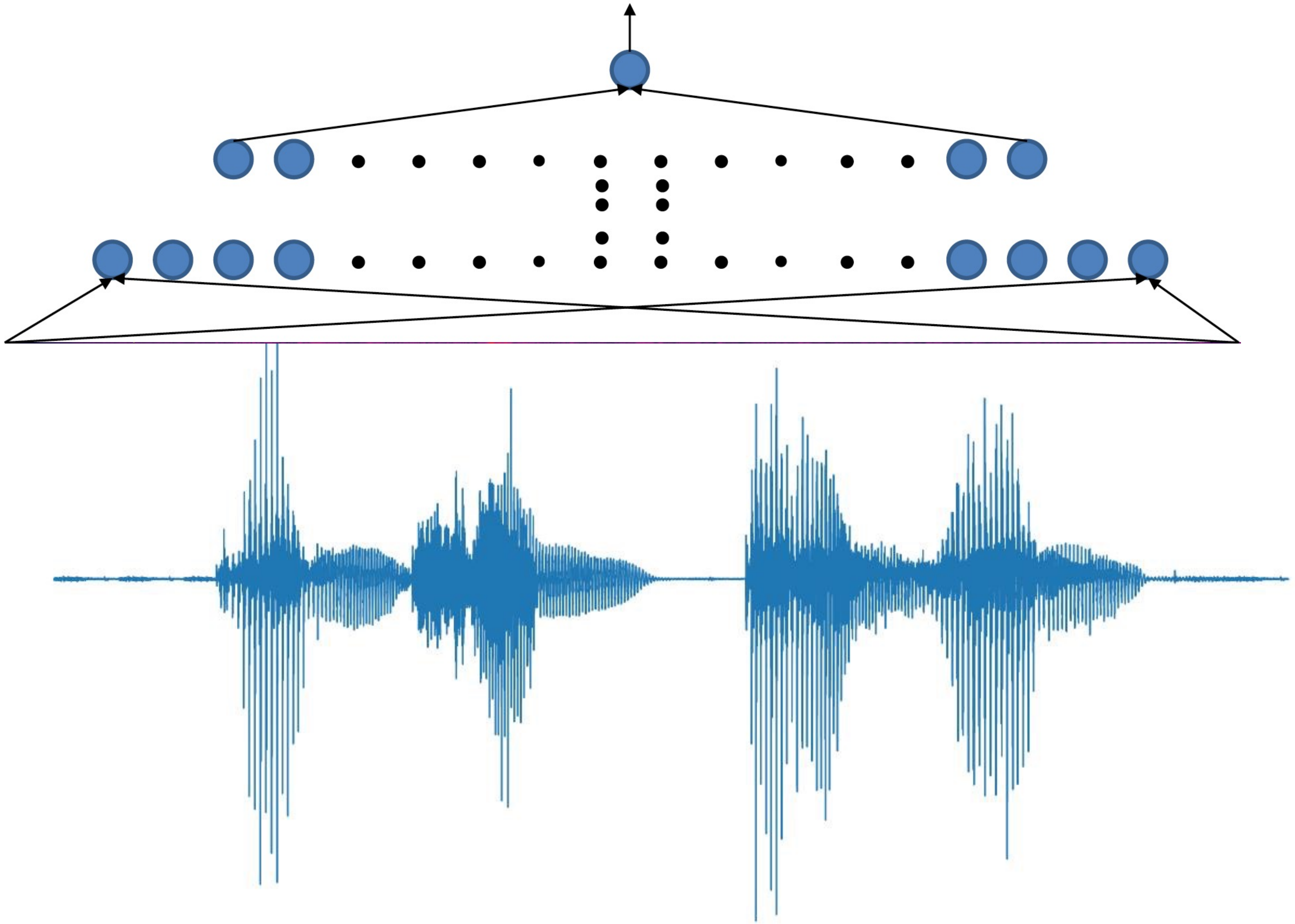


Using MLP to answer:
– Does this signal contain the word “Machine Learning”?
– If yes, where is the word “Machine Learning” ?

Search Pattern



Search Pattern

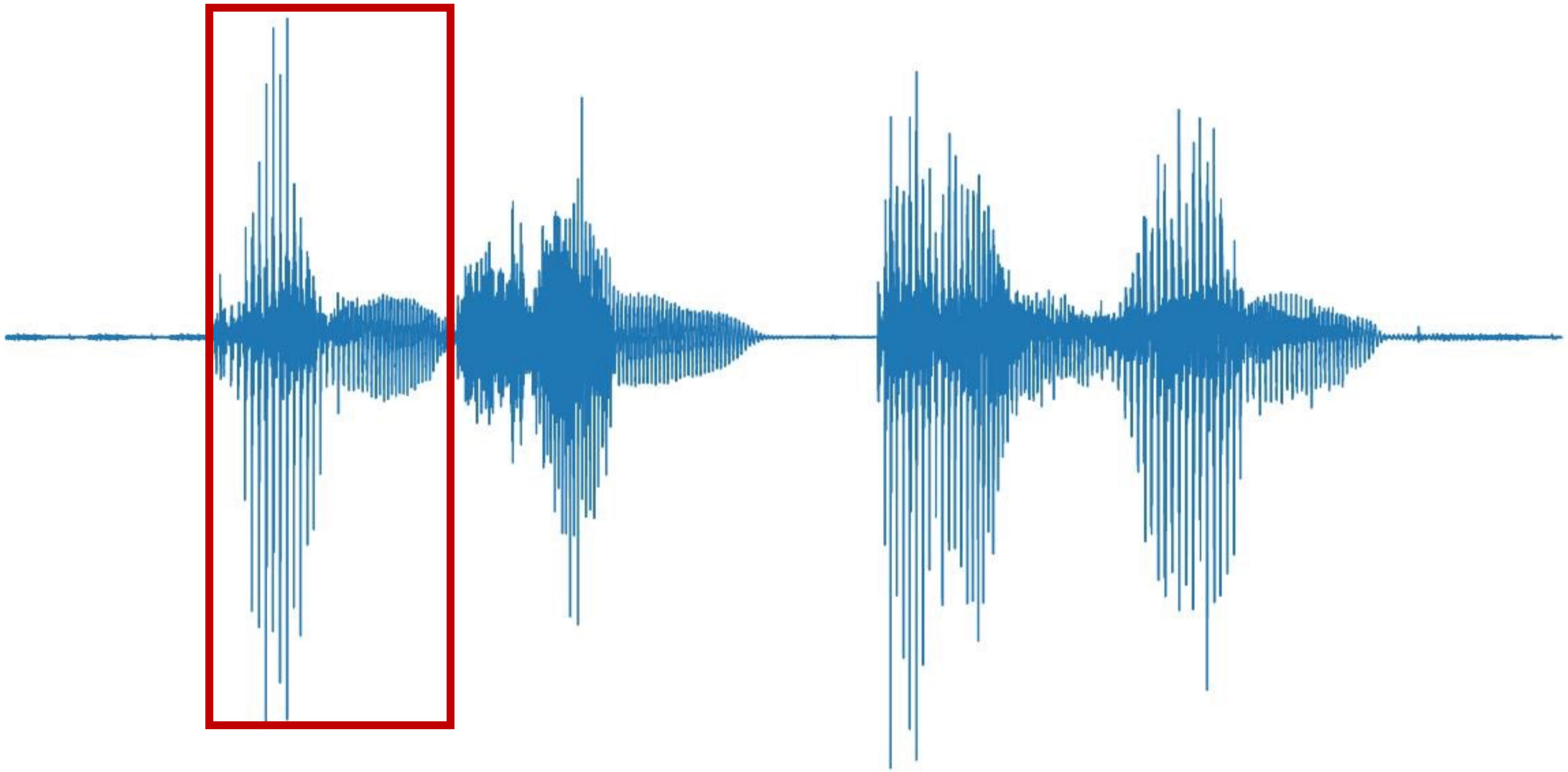


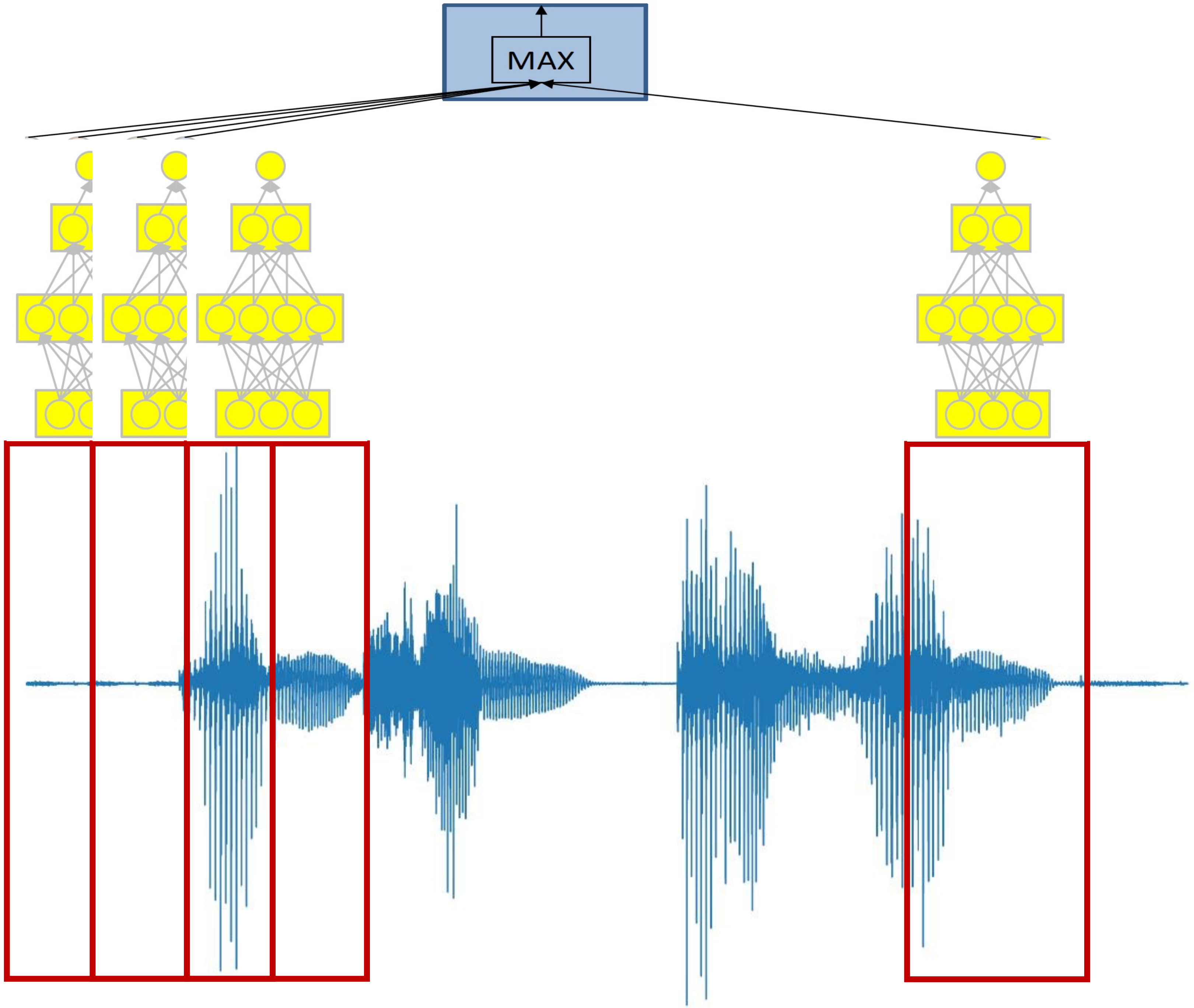
Search Pattern with MLP

Problems?

Search Pattern

Scanning



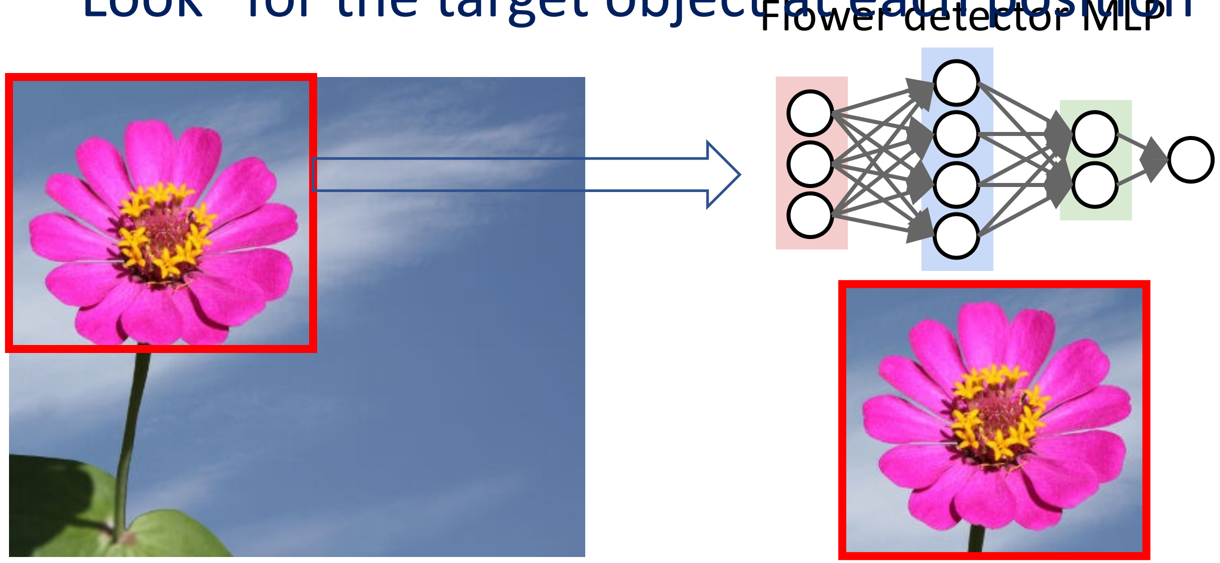

MAX

Flower Detection by scanning

"Look" for the target object at each position

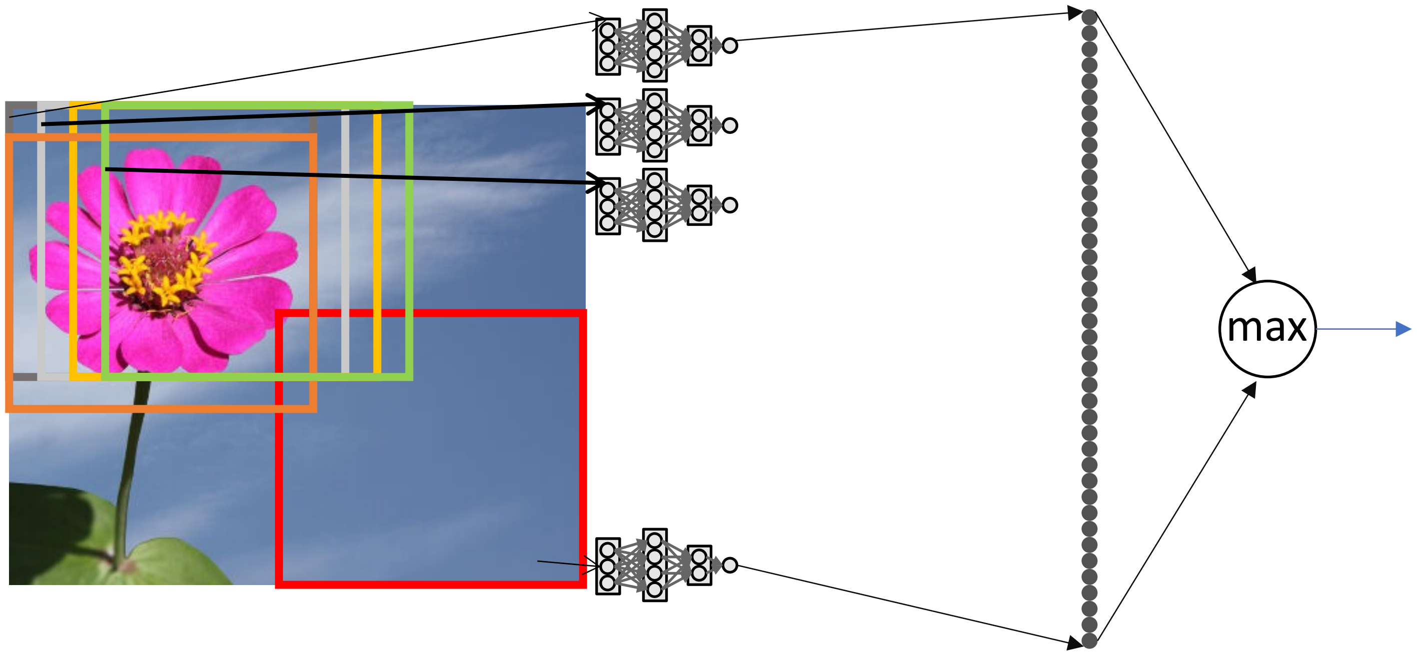
Flower Detection by scanning

"Look" for the target object at each position

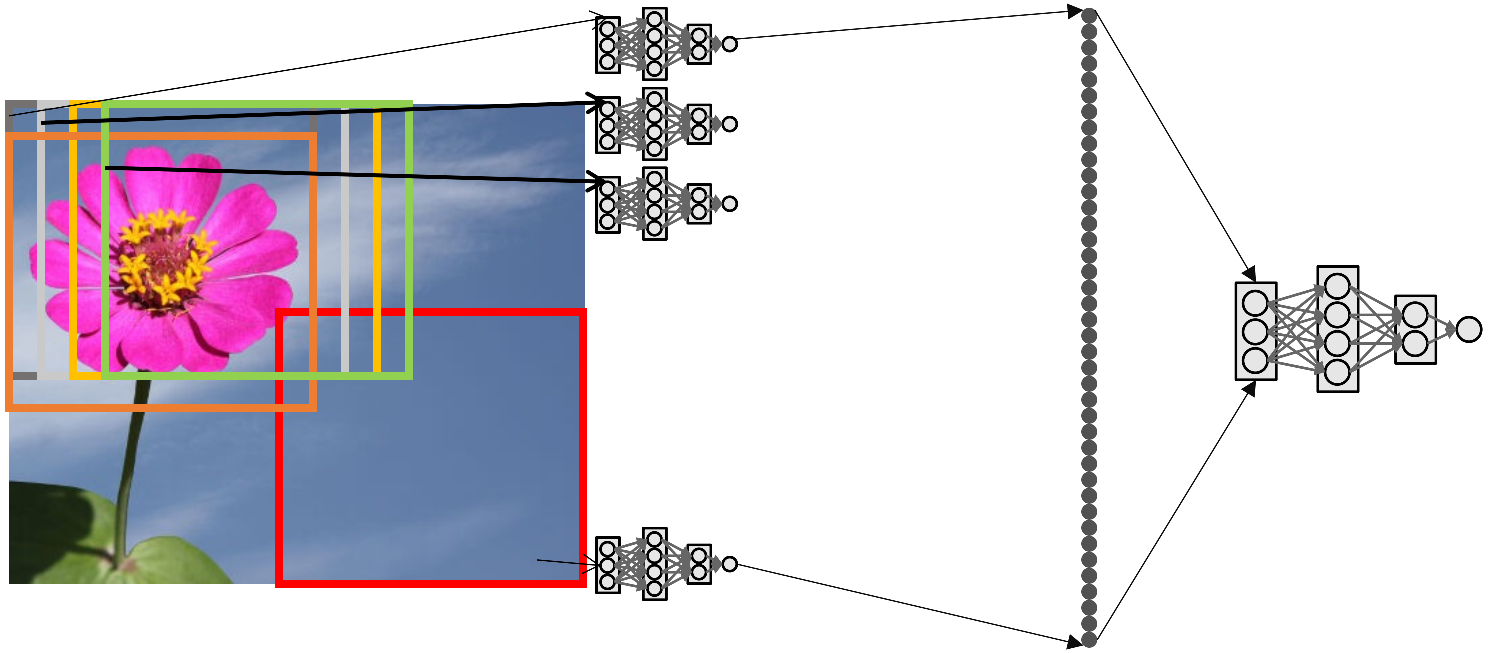


Flower Detection by scanning

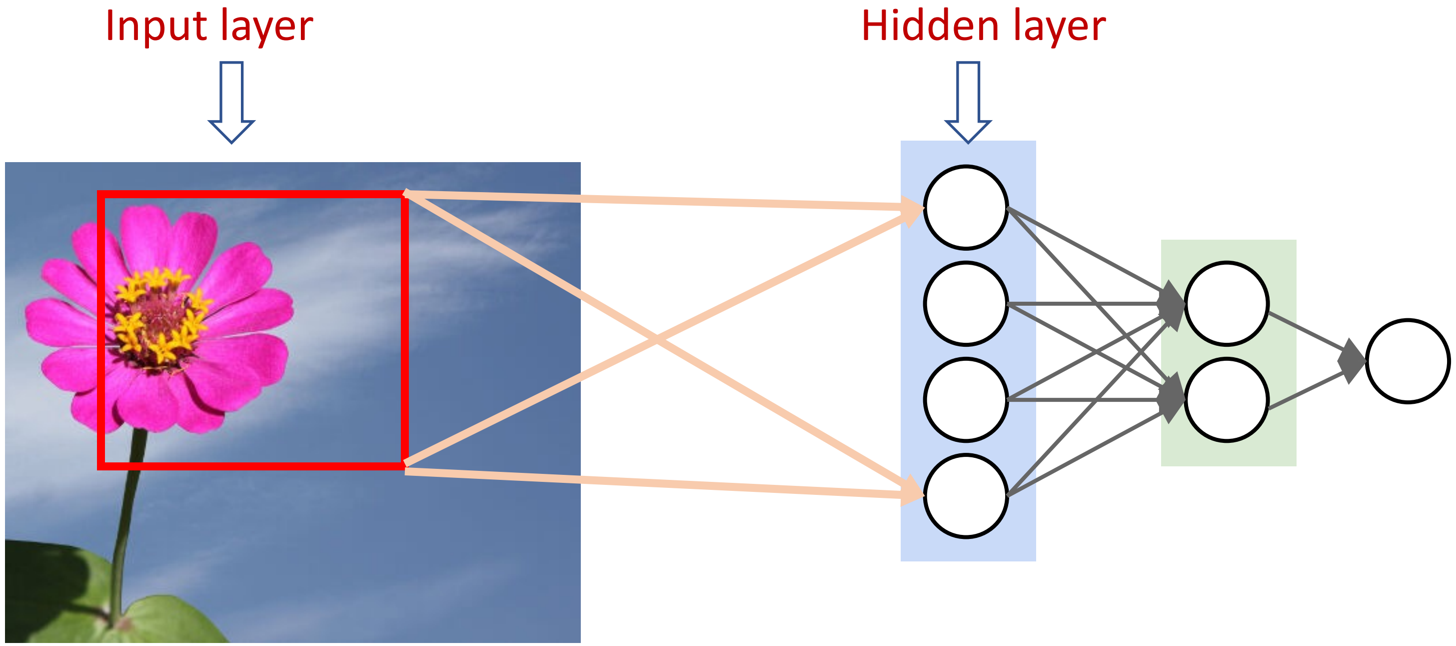
"Look" for the target object at each position



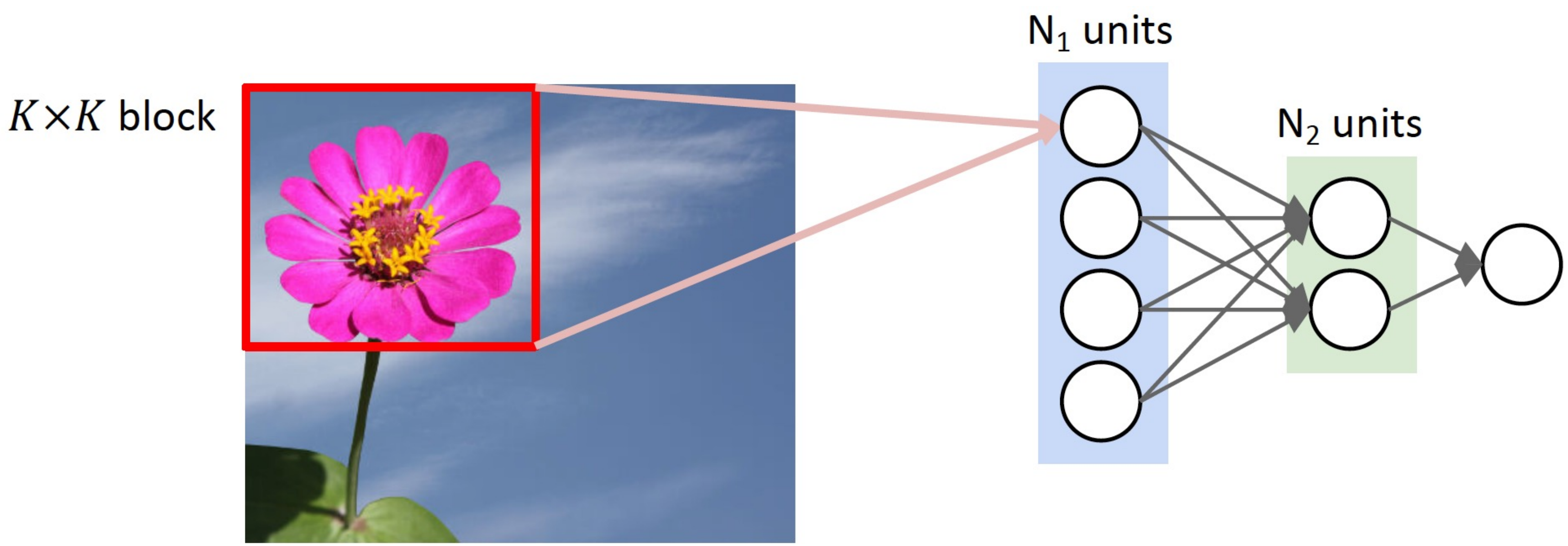
Its just a giant network with common subnets



Flower Detection by scanning

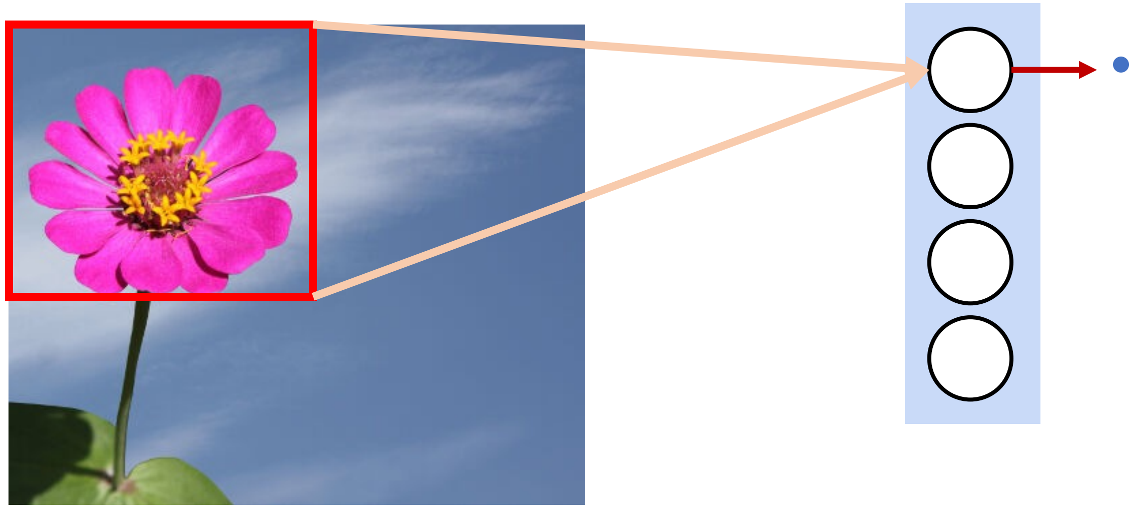


Flower Detection by scanning

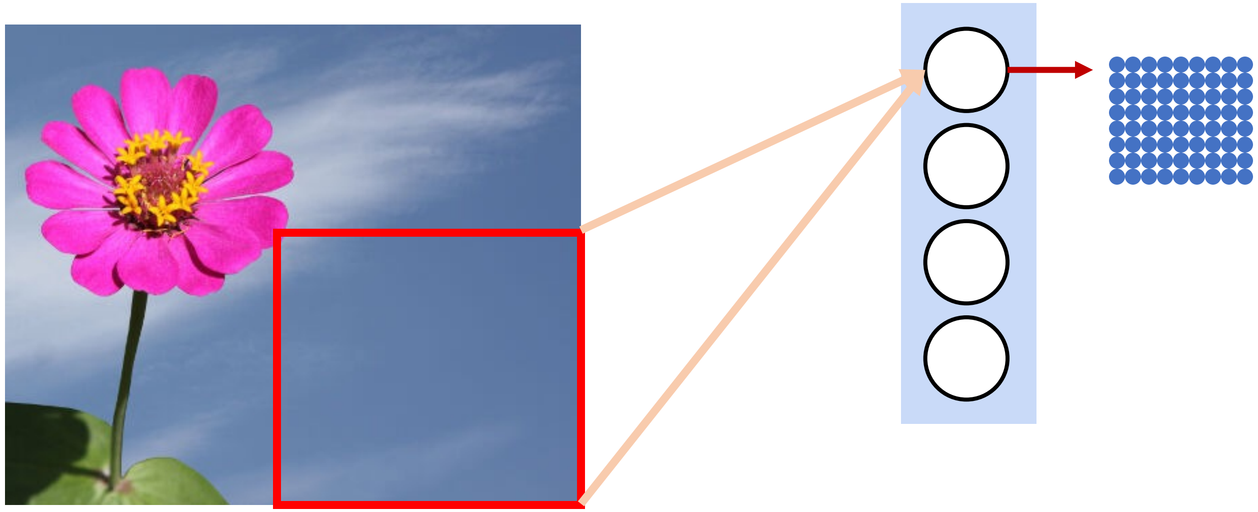


Flower Detection by scanning

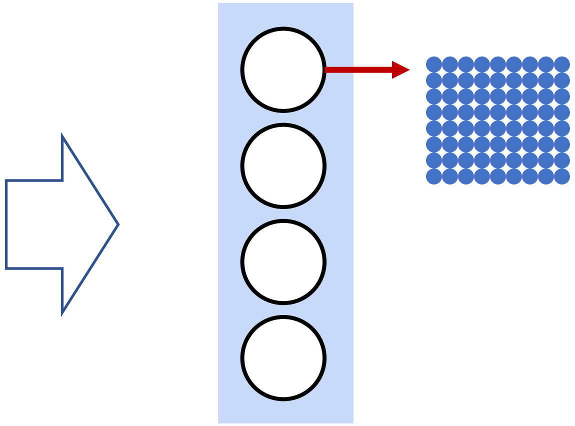
$$activation\left(\sum_{i,j} w_{ij}p_{ij} + b\right)$$



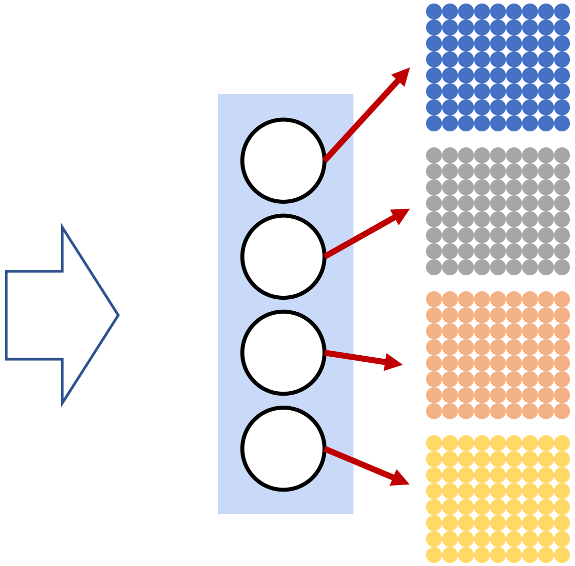
Flower Detection by scanning



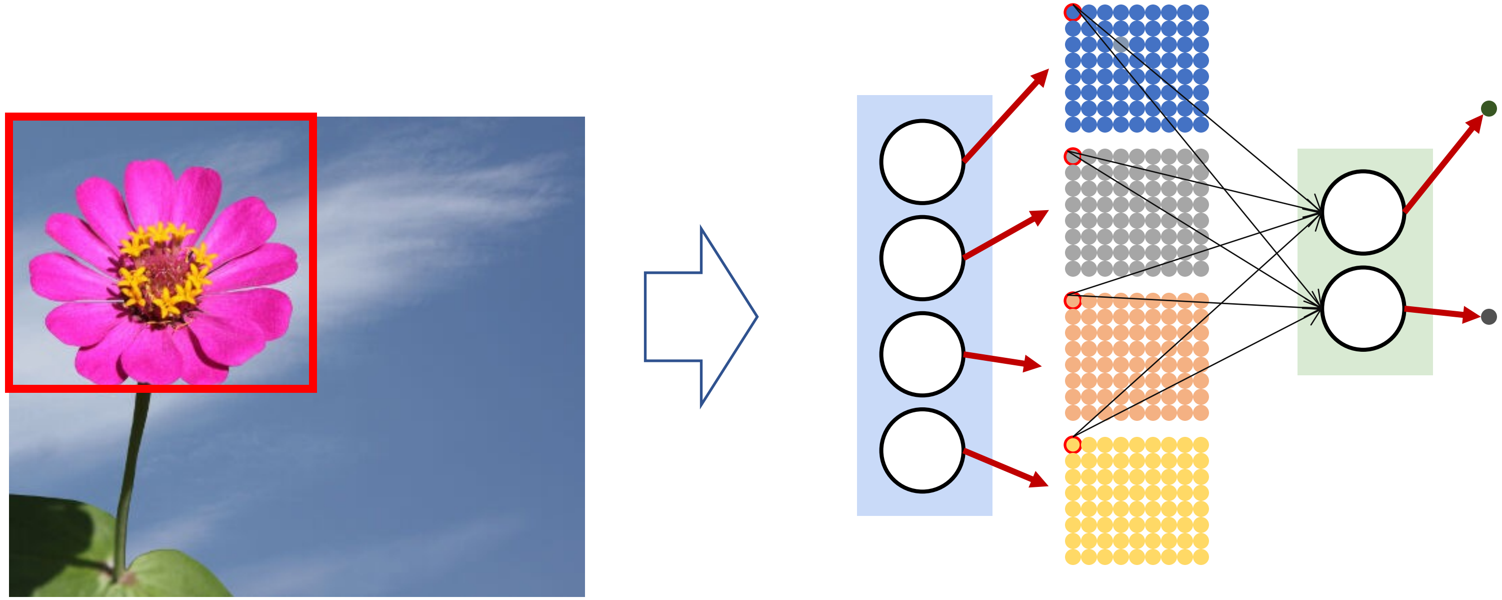
Flower Detection by scanning



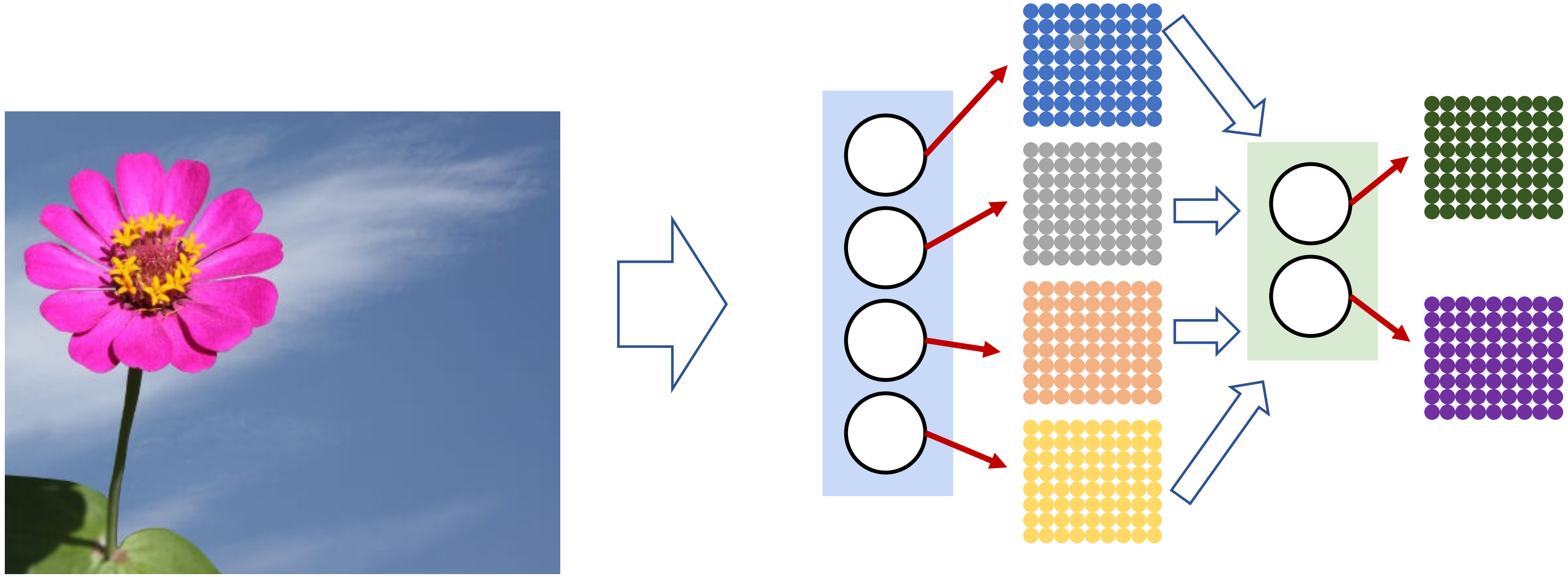
Flower Detection by scanning



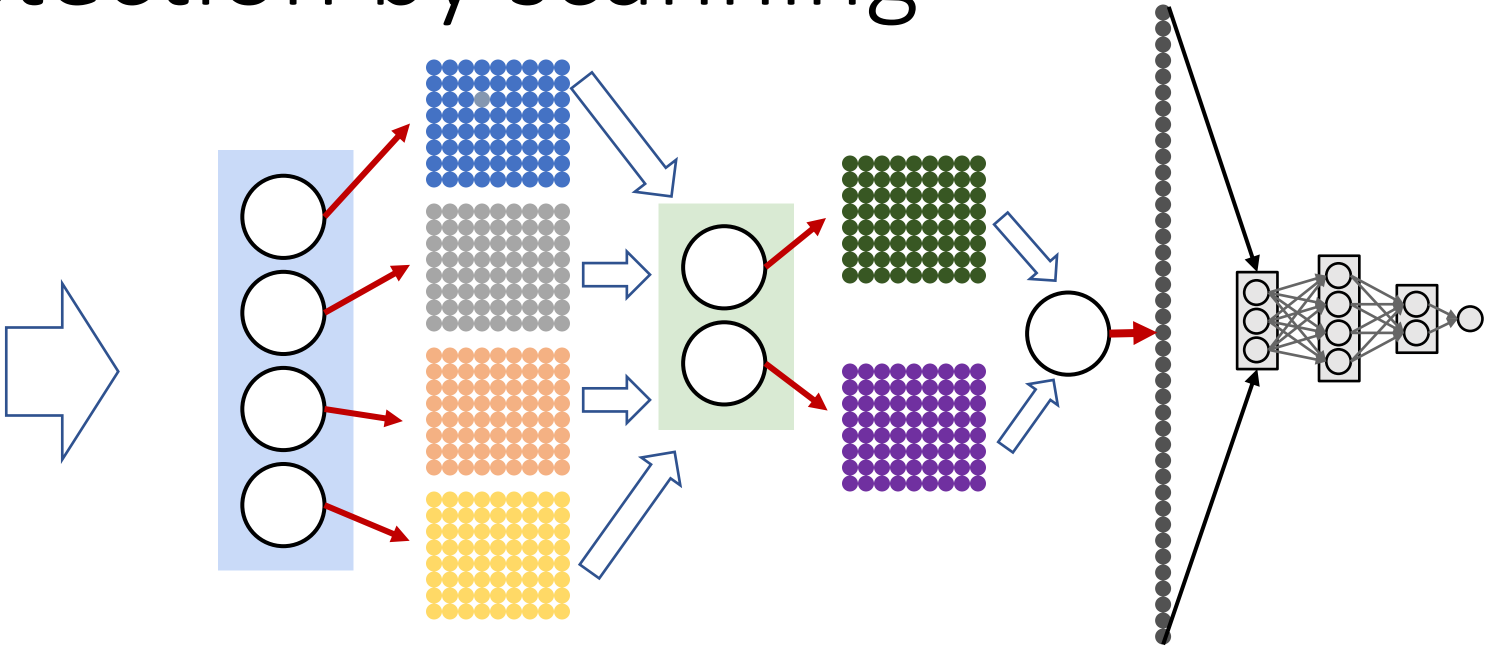
Flower Detection by scanning



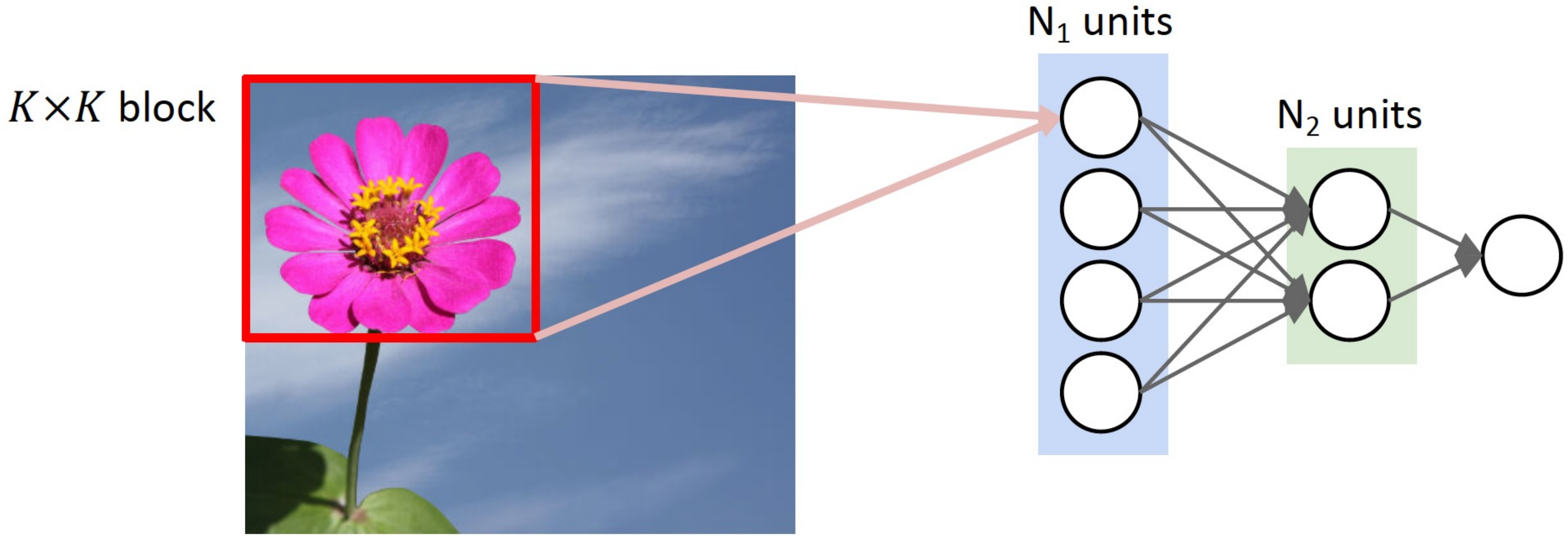
Flower Detection by scanning



Flower Detection by scanning



Flower Detection by scanning



$(K^2 + 1)N_1$ weights in first layer

$(N_1 + 1)N_2$ weights in second layer

$(N_{i-1} + 1)N_i$ weights in subsequent ith layer

Total parameters:

$$\mathcal{O}(K^2 N_1 + N_1 N_2 + N_2 N_3 \ldots)$$

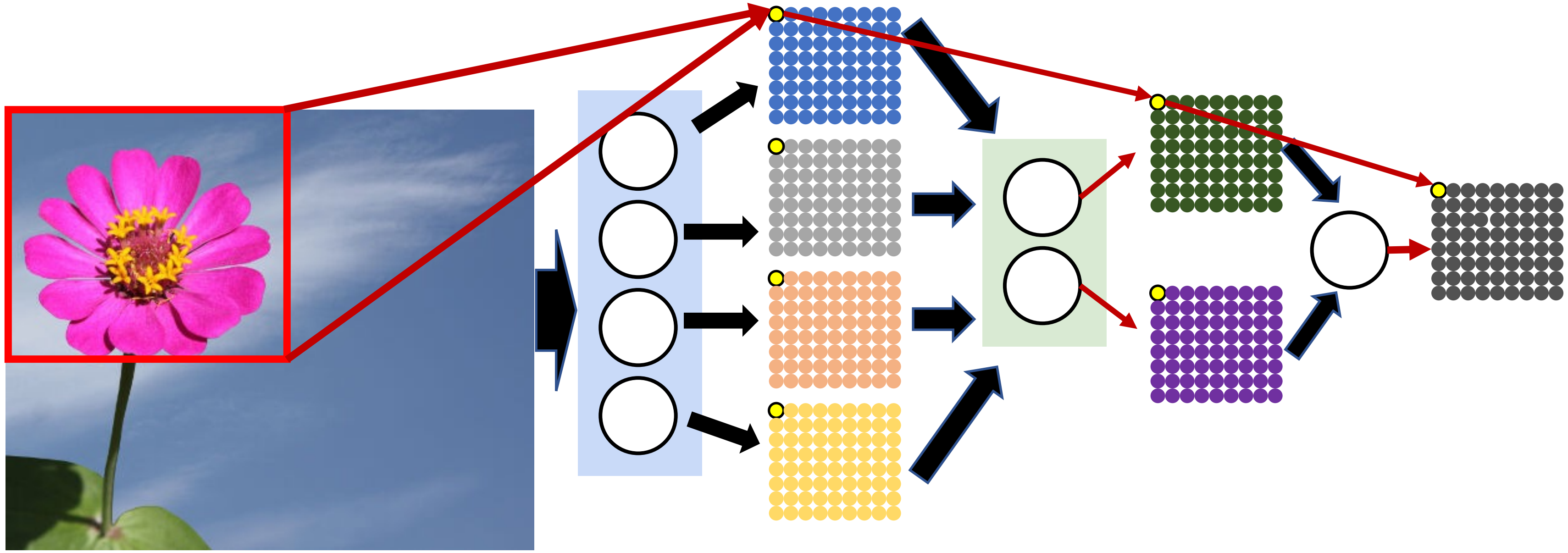
- Example

For this example, let $K = 16, N_1 = 4, N_2 = 2, N_3 = 1$

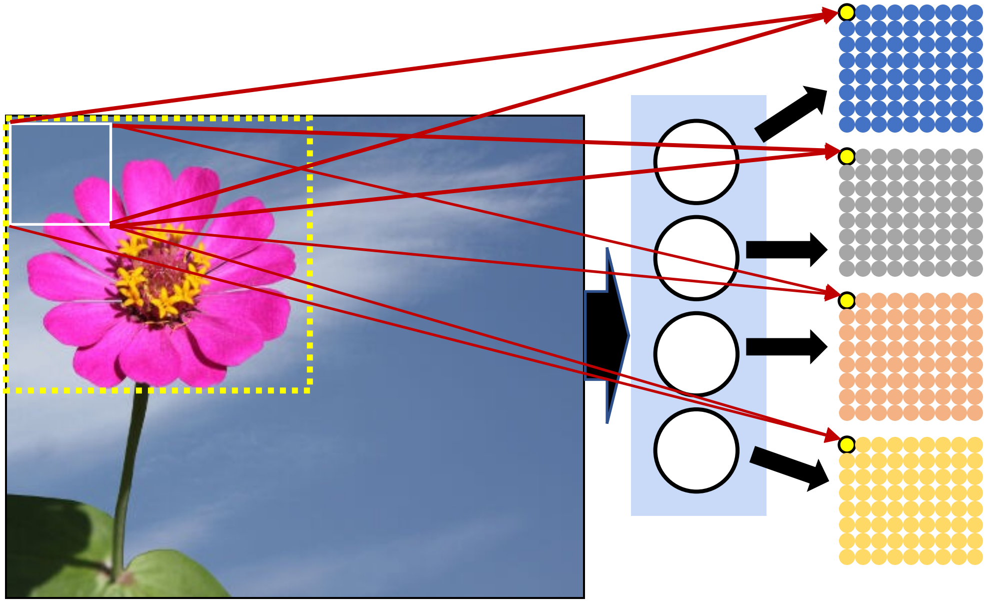
Total 1034 weights

Scanning with Distribution

Flower Detection by scanning



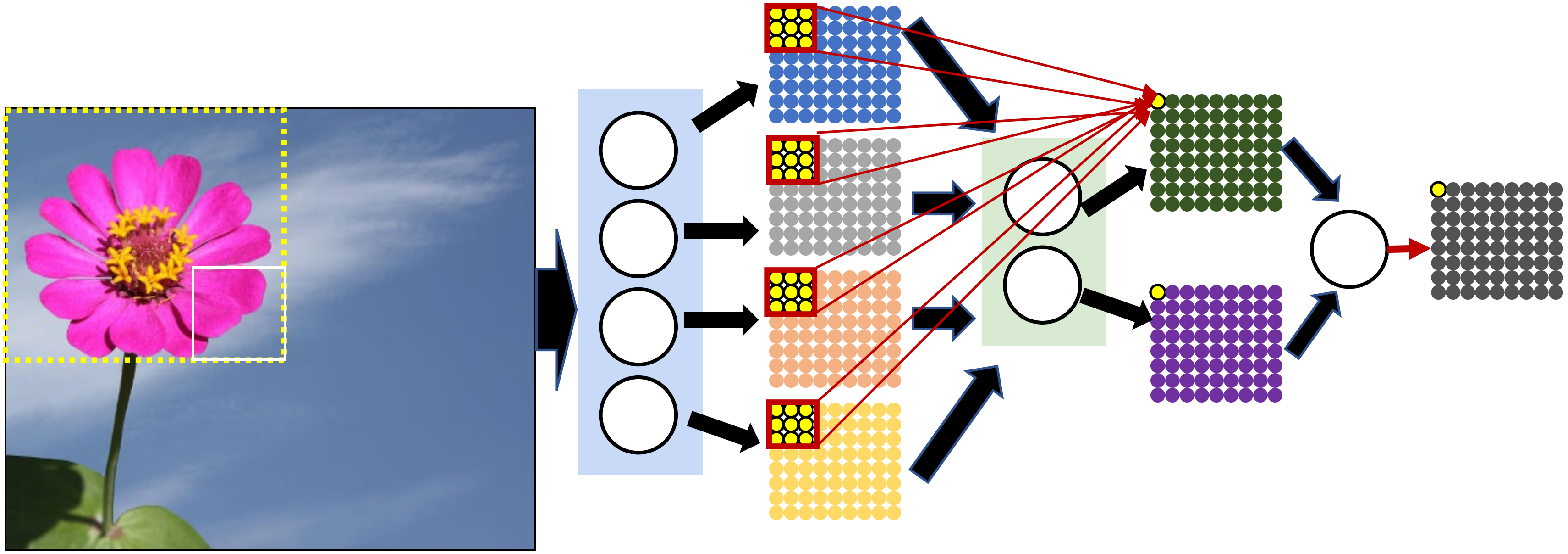
Flower Detection: Distributing the scan



The first layer evaluates smaller blocks of pixels

We can distribute the pattern matching over more (two) layers and still achieve the same block analysis at the second layer

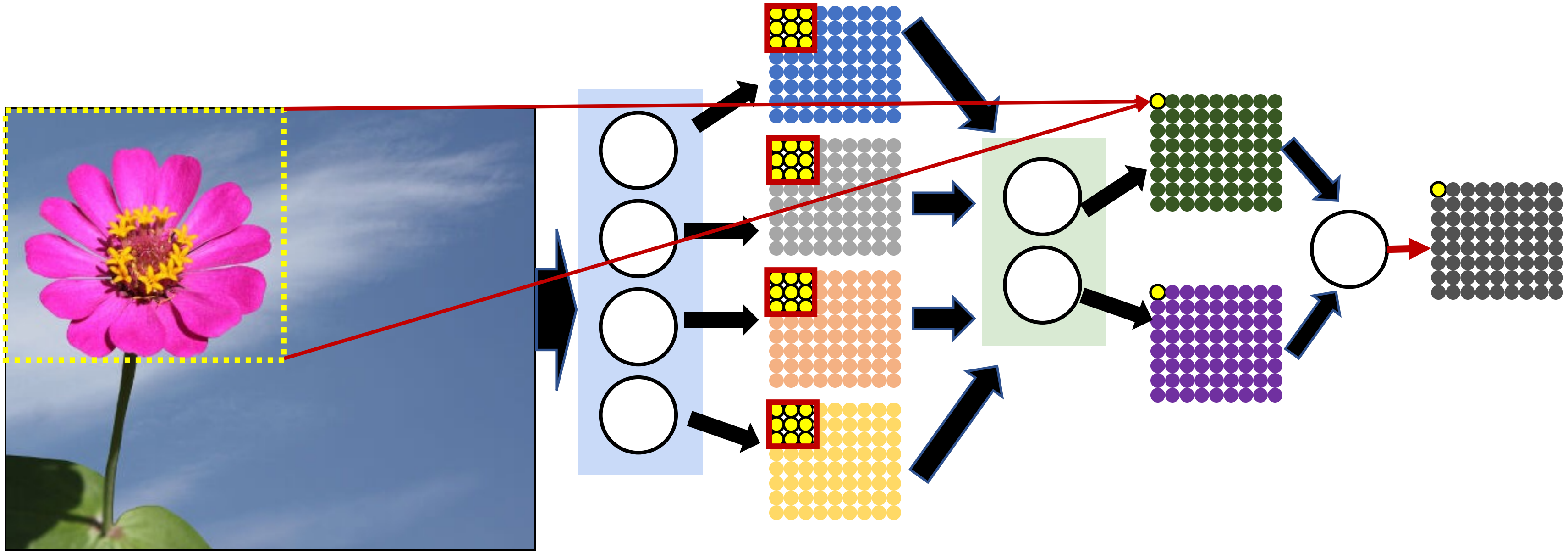
Flower Detection: Distributing the scan



The first layer evaluates smaller blocks of pixels

The next layer evaluates blocks of outputs from the first layer

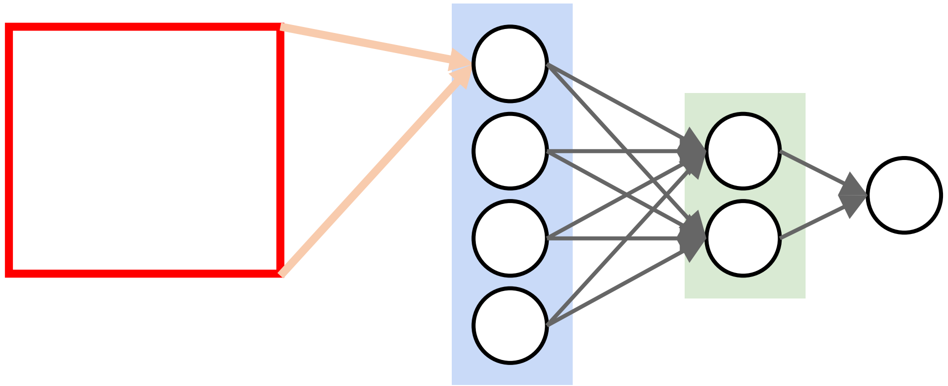
Flower Detection: Distributing the scan



The first layer evaluates smaller blocks of pixels

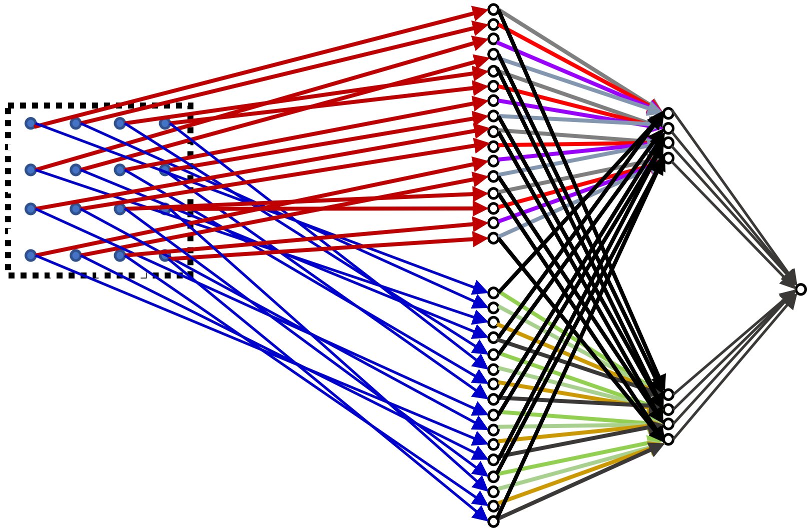
The next layer evaluates blocks of outputs from the first layer

Conventional MLP, not distributed

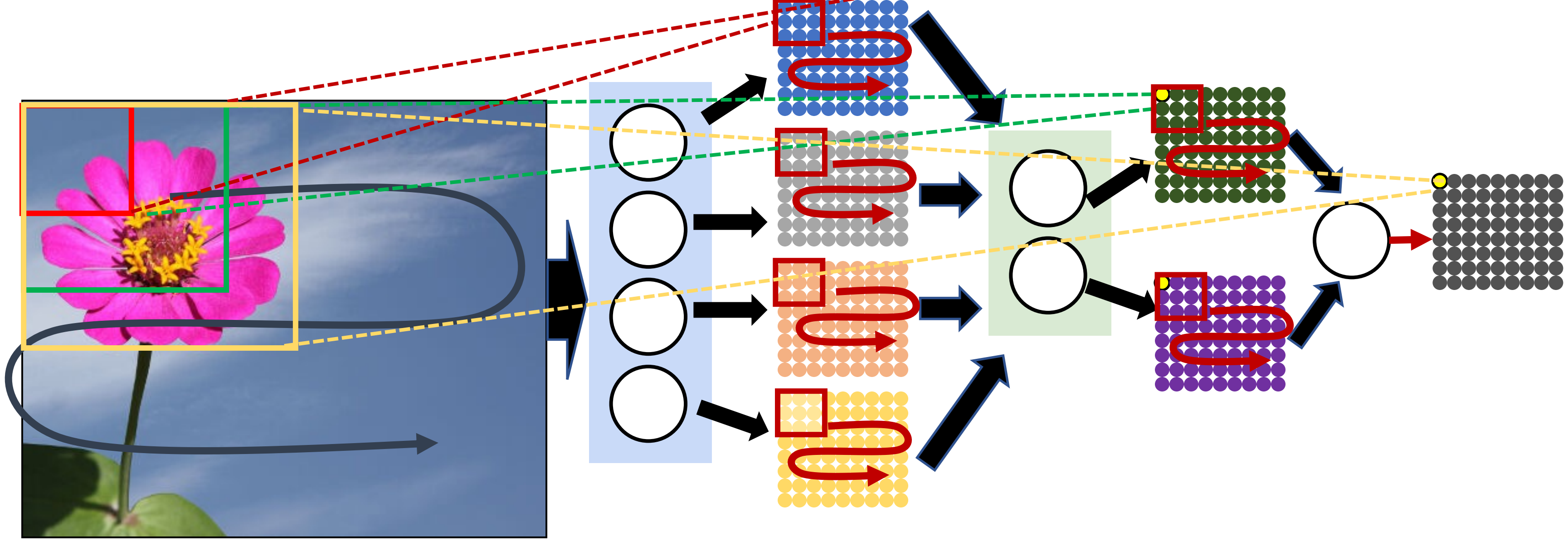


- $\mathcal{O}(K^2N_1 + N_1N_2 + N_2N_3 \ldots)$
- For this example, let $K = 16, N_1 = 4, N_2 = 2, N_3 = 1$
- Total 1034 weights

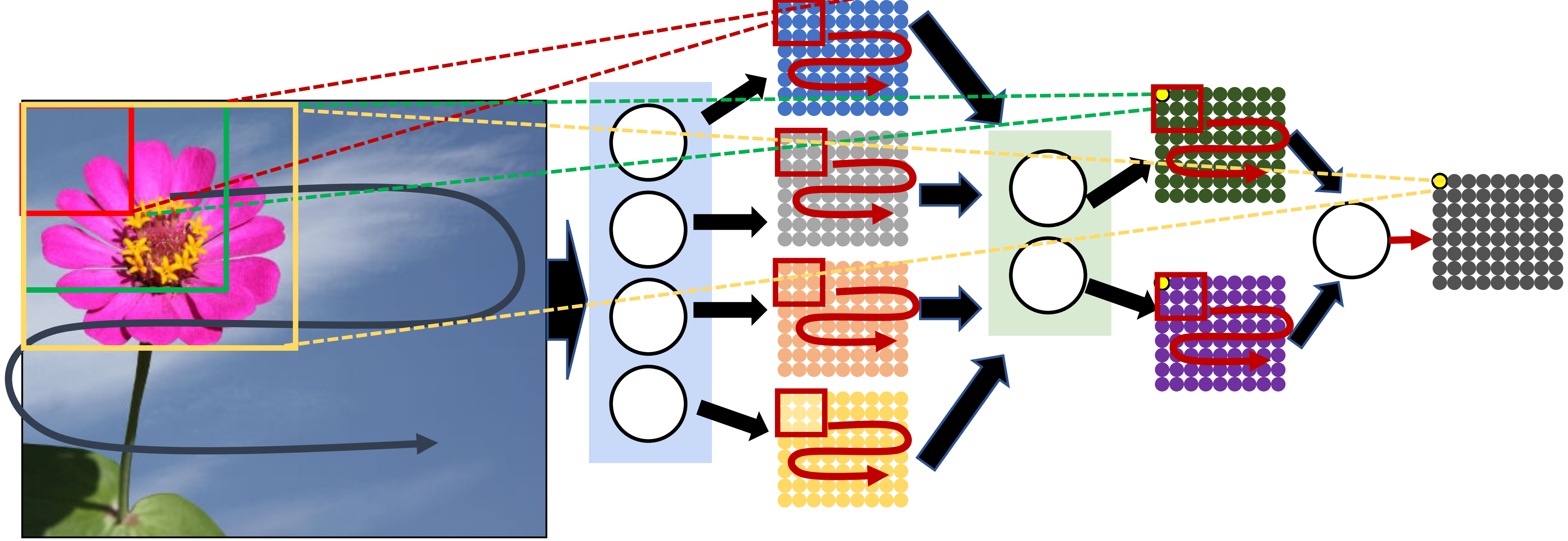
Distributed (3 layers)



- $\mathcal{O}\left(L_1^2N_1 + L_2^2N_1N_2 + \left(\frac{K}{L_1L_2}\right)^2 N_2N_3 + \cdots\right.$
- Here, let $K = 16,\ L_1 = 4,\ L_2 = 4,\ N_1 = 4, N_2 = 2, N_3 = 1$
- Total 64+128+8 = 160 weights



Convolutional Neural Networks

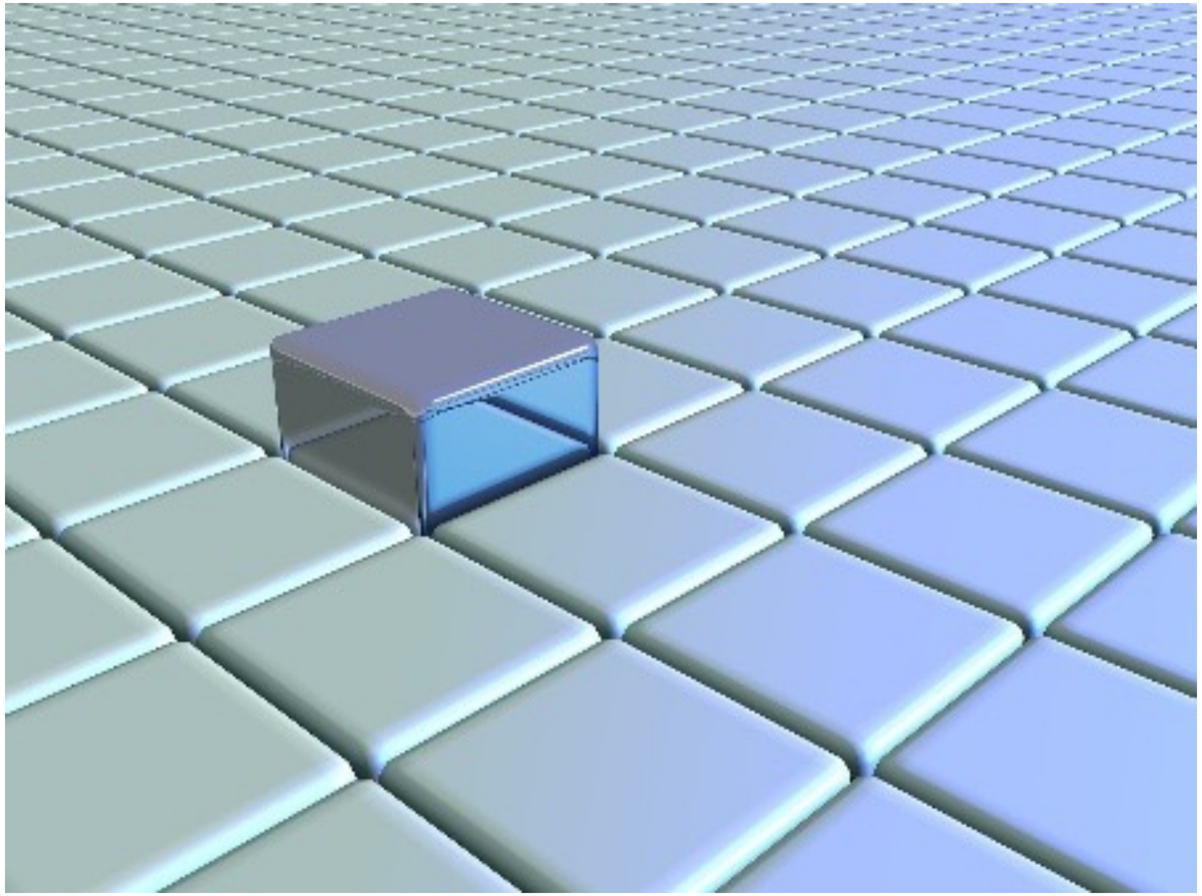


Each of the scanning neurons is generally called a "**filter**"

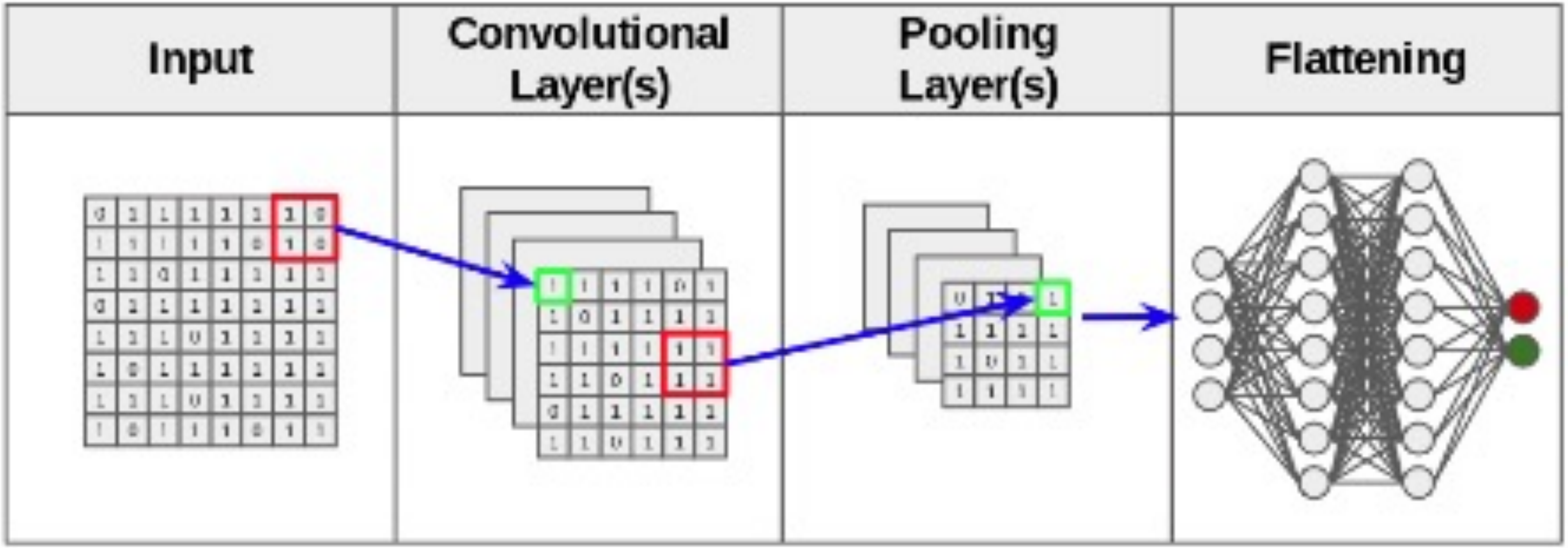
Its really a **correlation filter** as we saw earlier

Each filter scans for a pattern in the map it operates on

For a given neuron, the visual space that affects whether or not that neuron will fire is known as its "**receptive field**."



Convolutional Neural Networks



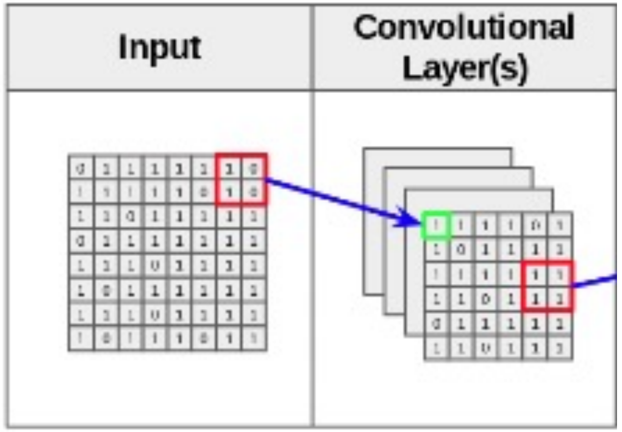
Convolutional Neural Networks: Convolution

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input

1	0	1
0	1	0
1	0	1

Filter

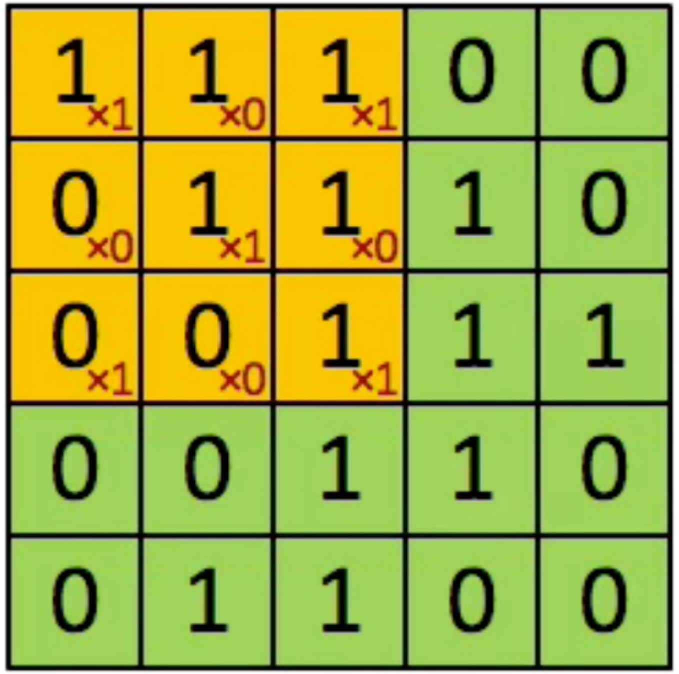


Bias: 0

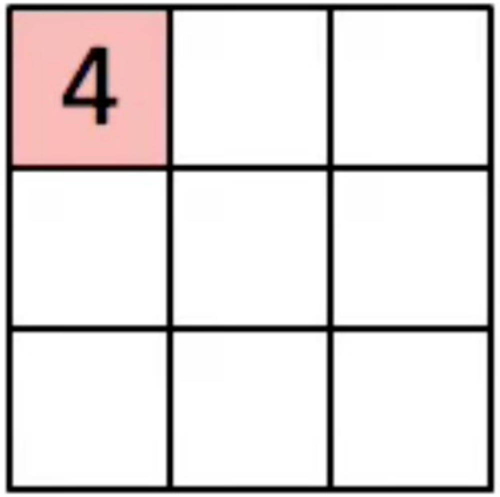
Convolutional Neural Networks: <u>Convolution</u>

1	0	1
0	1	0
1	0	1

Filter



Input Map

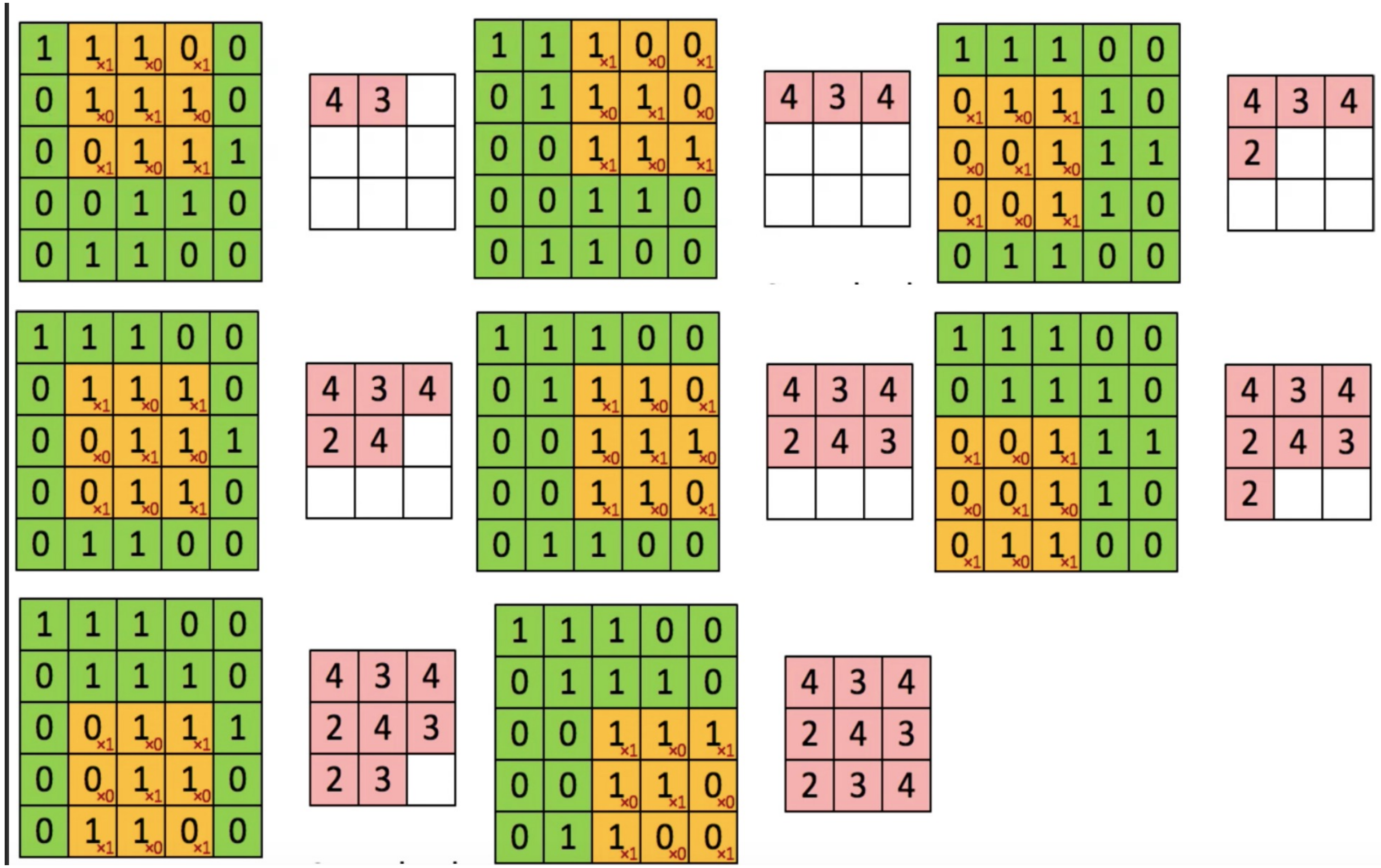


Input

1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Filter

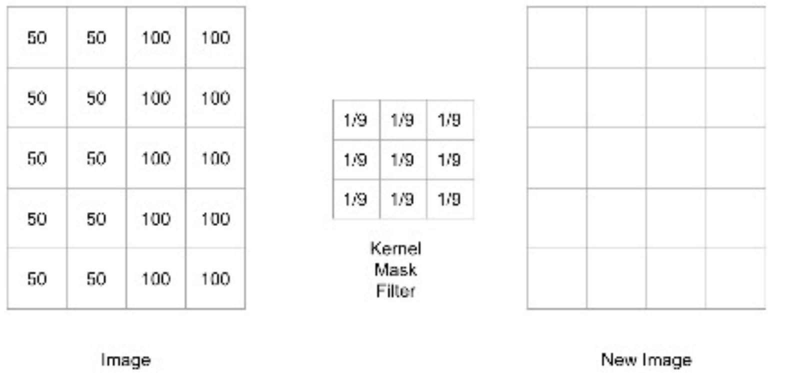
1	0	1
0	1	0
1	0	1

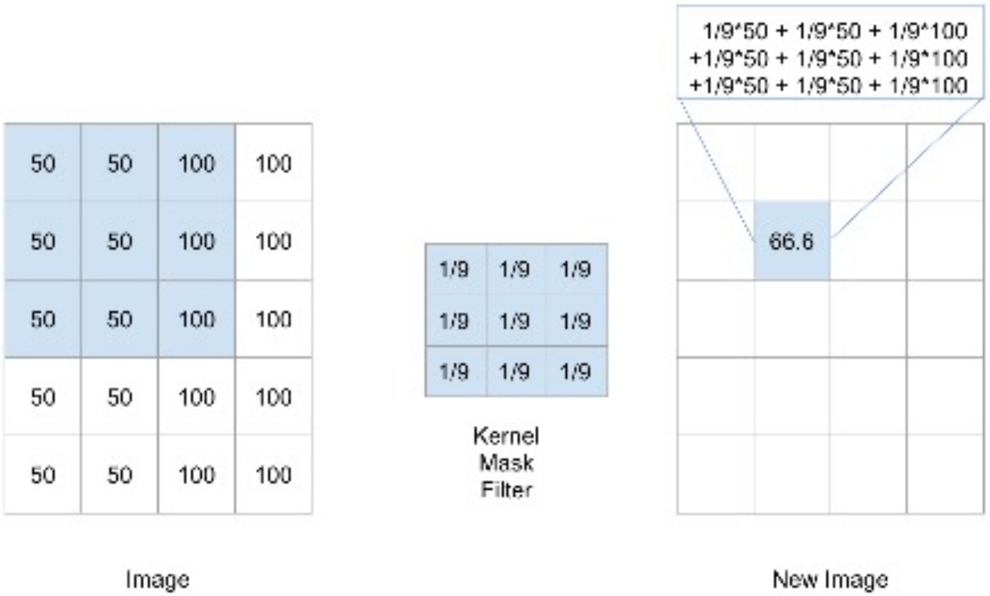


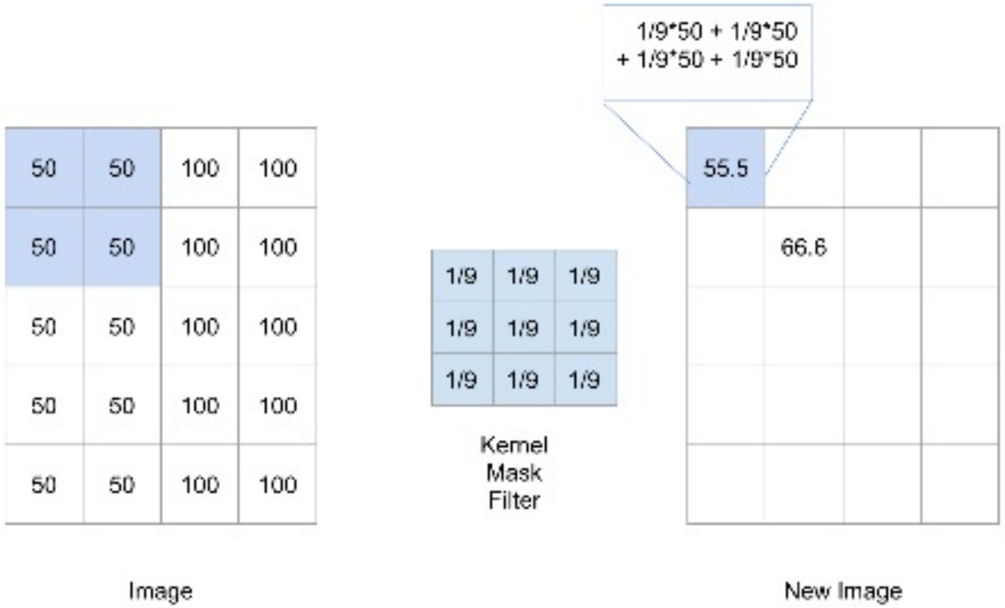
Blurring

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

Blurring


50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
Kernel
Mask
Filter
Image
New Image


1/9*50 + 1/9*50 + 1/9*100
+1/9*50 + 1/9*50 + 1/9*100
+1/9*50 + 1/9*50 + 1/9*100
50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
Kernel
Mask
Filter
66.6
Image
New Image


1/9*50 + 1/9*50
+ 1/9*50 + 1/9*50
50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
50 50 100 100
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
Kernel
Mask
Filter
55.5
66.6
Image
New Image

Edge Detection

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Measures changes in x-direction
(i.e. detects vertical lines)

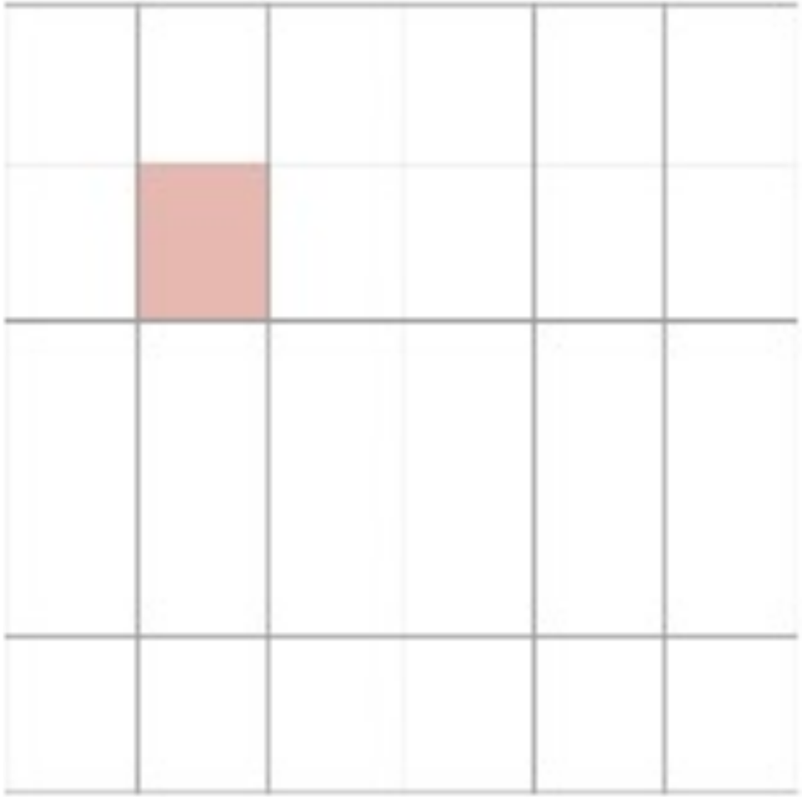
$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

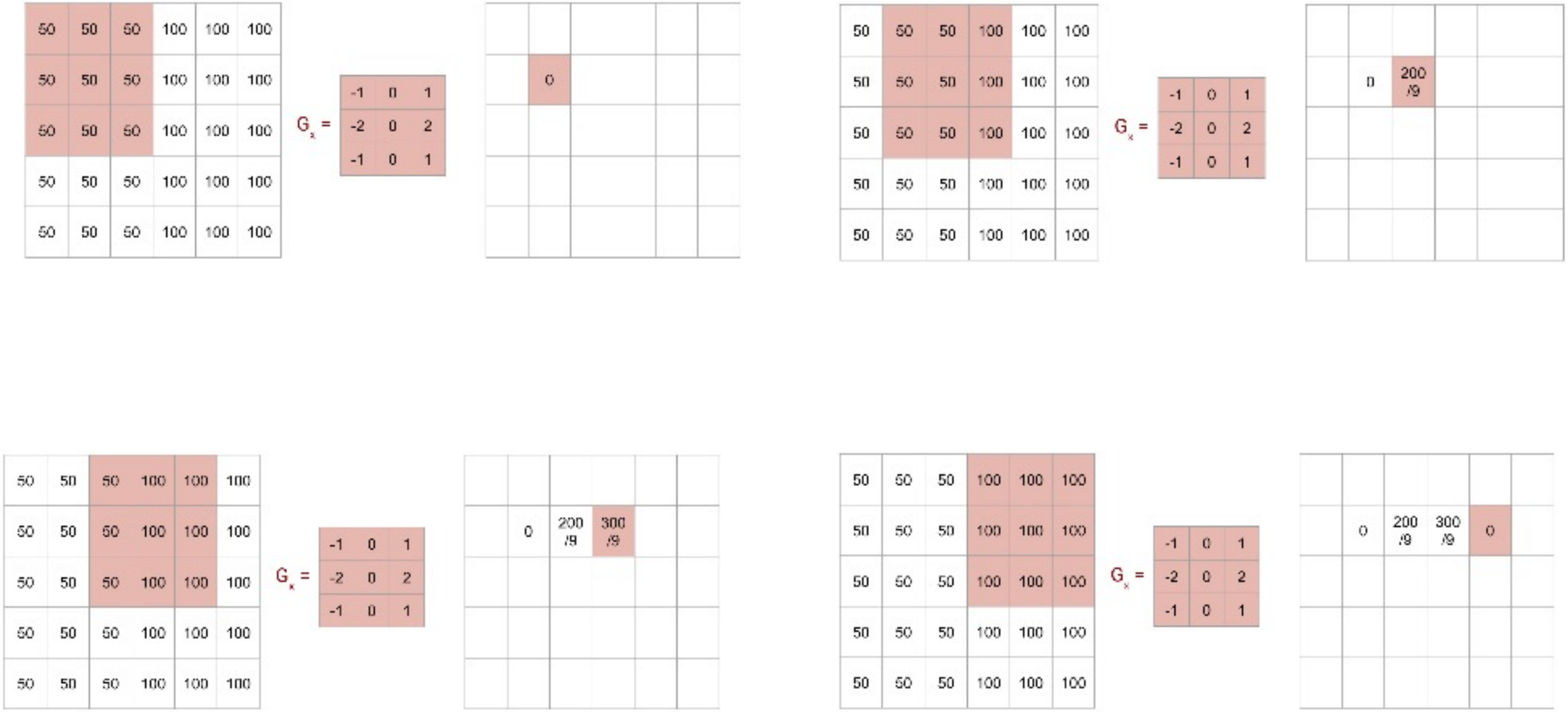
Measures changes in y-direction
(i.e. detects horizontal lines)

Edge Detection

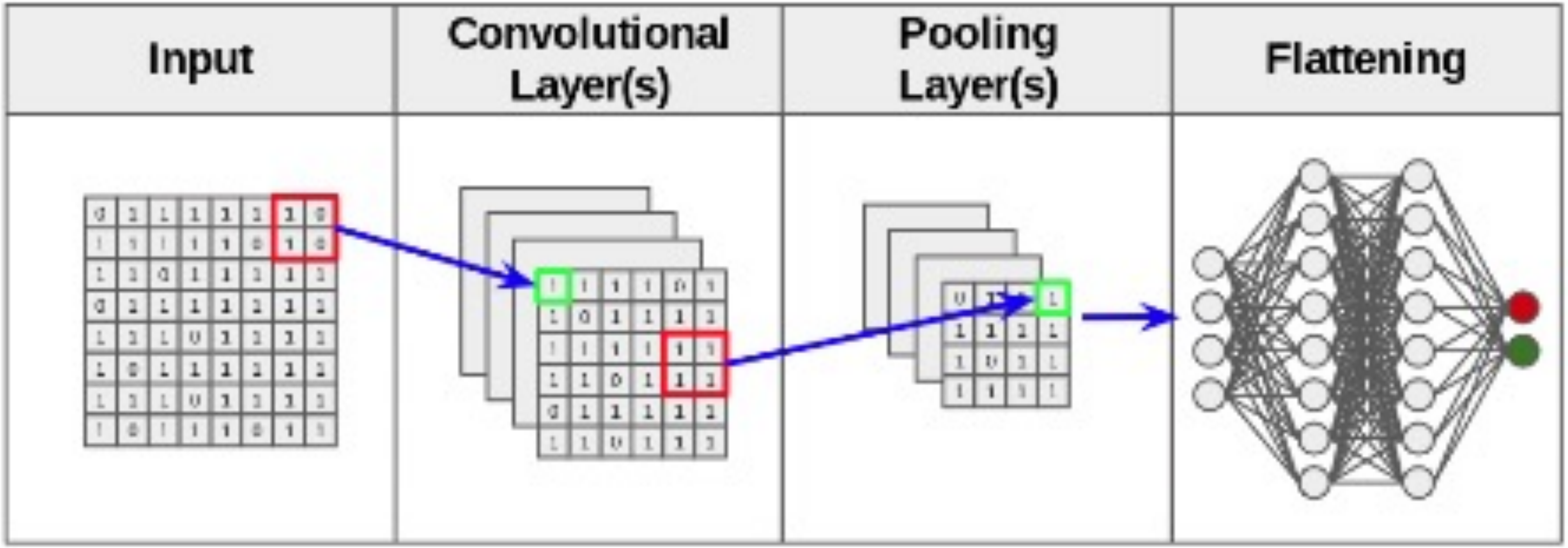
50	50	50	100	100	100
50	50	50	100	100	100
50	50	50	100	100	100
50	50	50	100	100	100
50	50	50	100	100	100

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

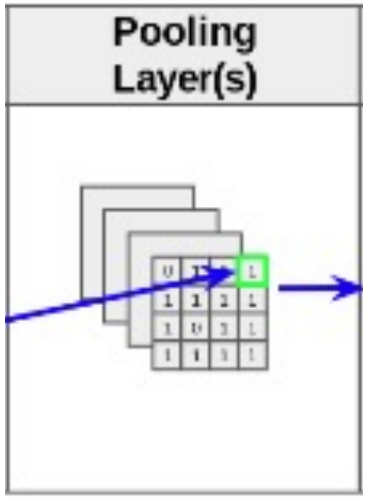



50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
G_x =
-1 0 1
-2 0 2
-1 0 1
0
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
G_x =
-1 0 1
-2 0 2
-1 0 1
0 200/9
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
G_x =
-1 0 1
-2 0 2
-1 0 1
0 200/9 300/9
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
50 50 50 100 100 100
G_x =
-1 0 1
-2 0 2
-1 0 1
0 200/9 300/9 0

Convolutional Neural Networks

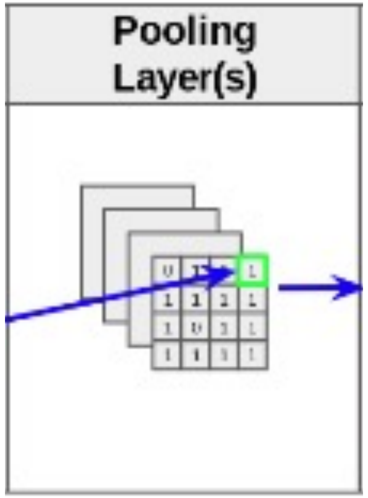


Convolutional Neural Networks: **Pooling**



Pooling is a type of downsampling that often occurs after convolution. The goal is, without losing much information, to reduce the size of the training data before it goes into the fully connected network.

Convolutional Neural Networks: **Pooling**

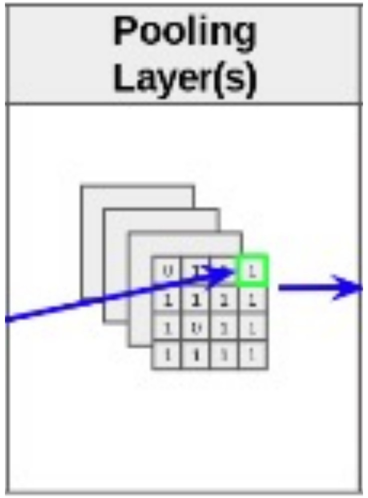


1. Pick a window size
2. Pick a strike
3. Move your window across each of the filtered image
4. Take maximum/minimum, average value in each window

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

Convolutional Neural Networks: <u>**Pooling**</u>

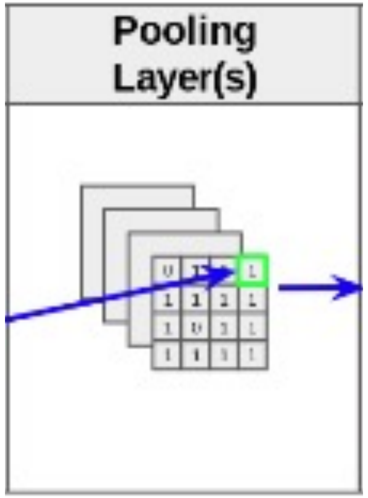


1. Pick a window size
2. Pick a strike
3. Move your window across each of the filtered image
4. Take maximum/minimum, average value in each window

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

Convolutional Neural Networks: **Pooling**



1. Pick a window size
2. Pick a strike
3. Move your window across each of the filtered image
4. Take maximum/minimum, average value in each window

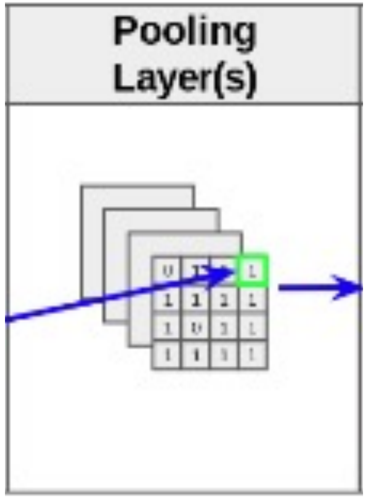
29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

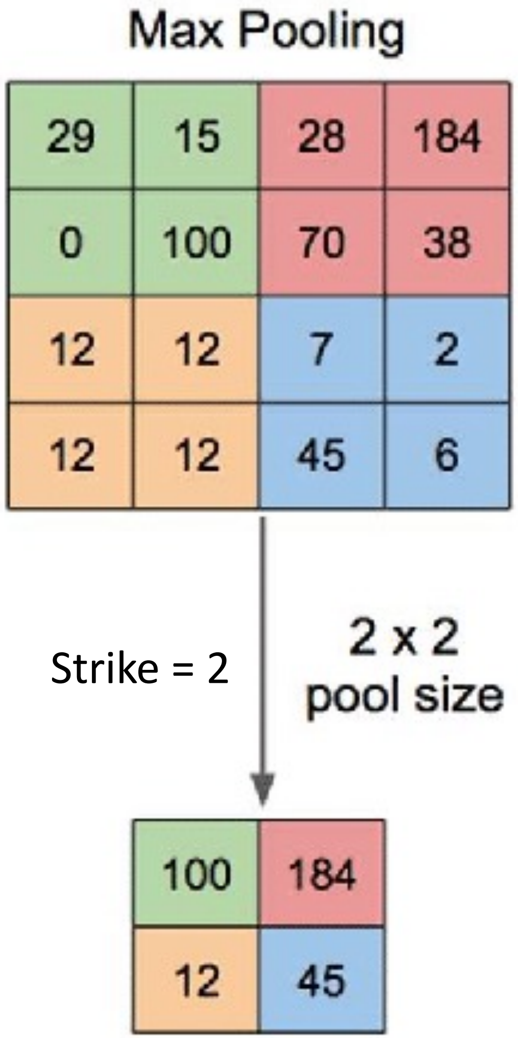
29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

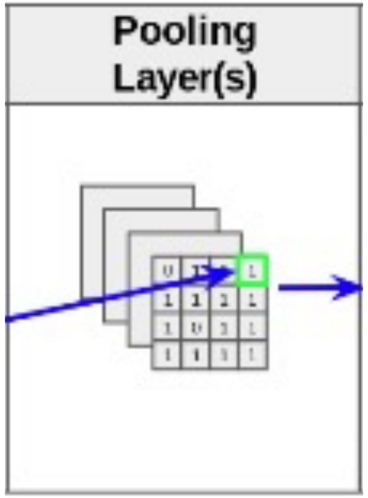
29	15	28	184
0	100	70	38
12	12	7	2
12	12	45	6

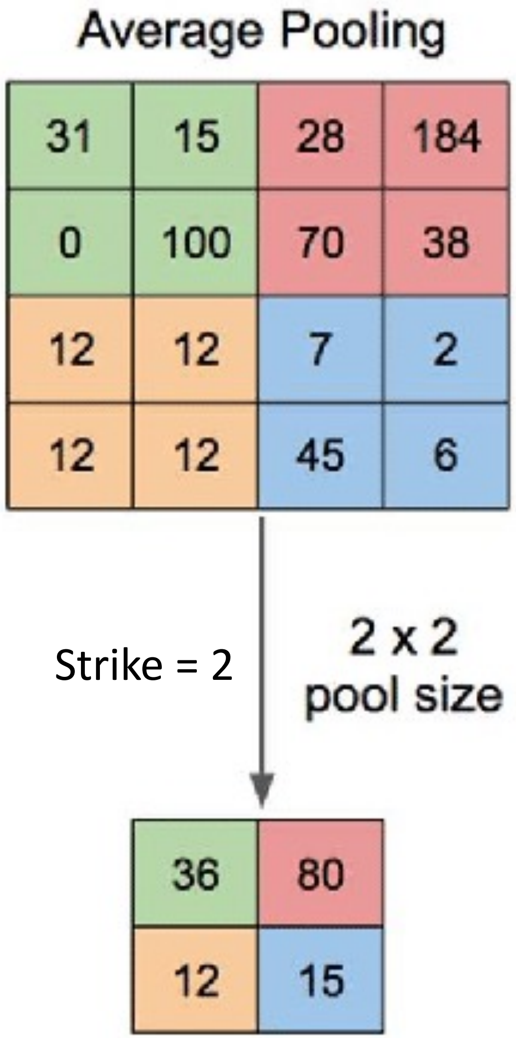
Convolutional Neural Networks: **Pooling**



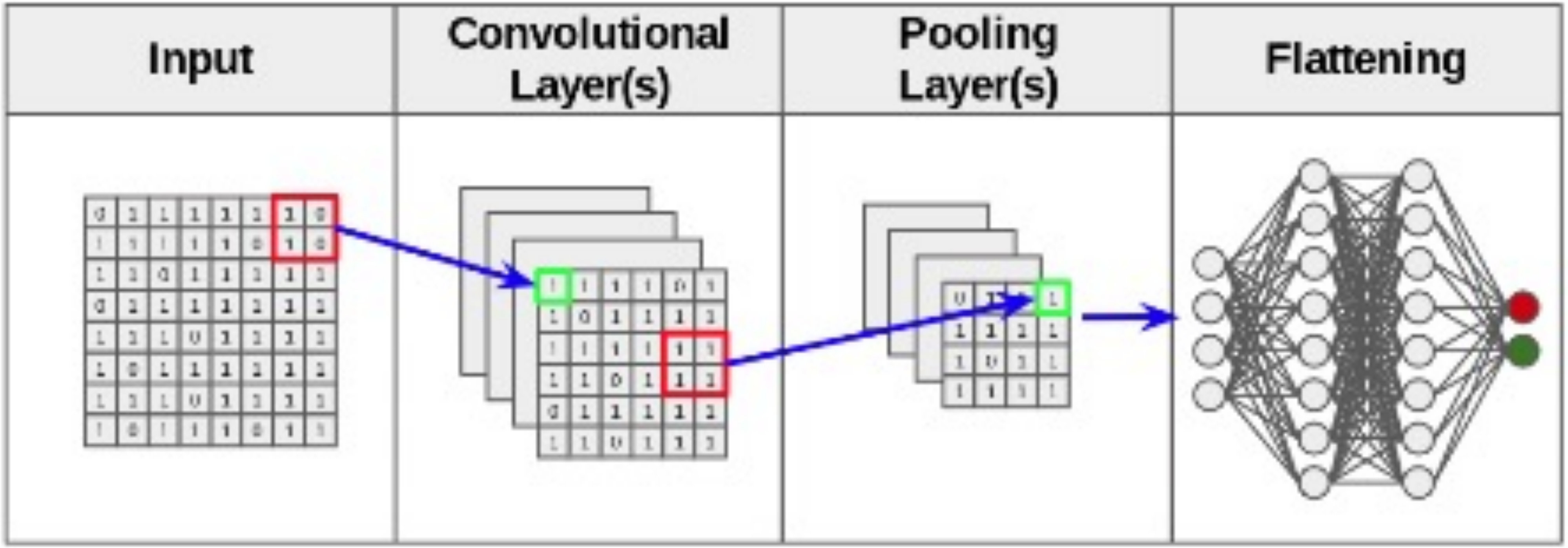


Convolutional Neural Networks: <u>Pooling</u>





Hyperparameters



How many of each layers and in what orders

Hyperparameters

- Convolution Layer
  - Number of filters
  - Size of filters
- Pooling Layers
  - Window size
  - Strike
- Fully Connected Layers
  - Number of nodes