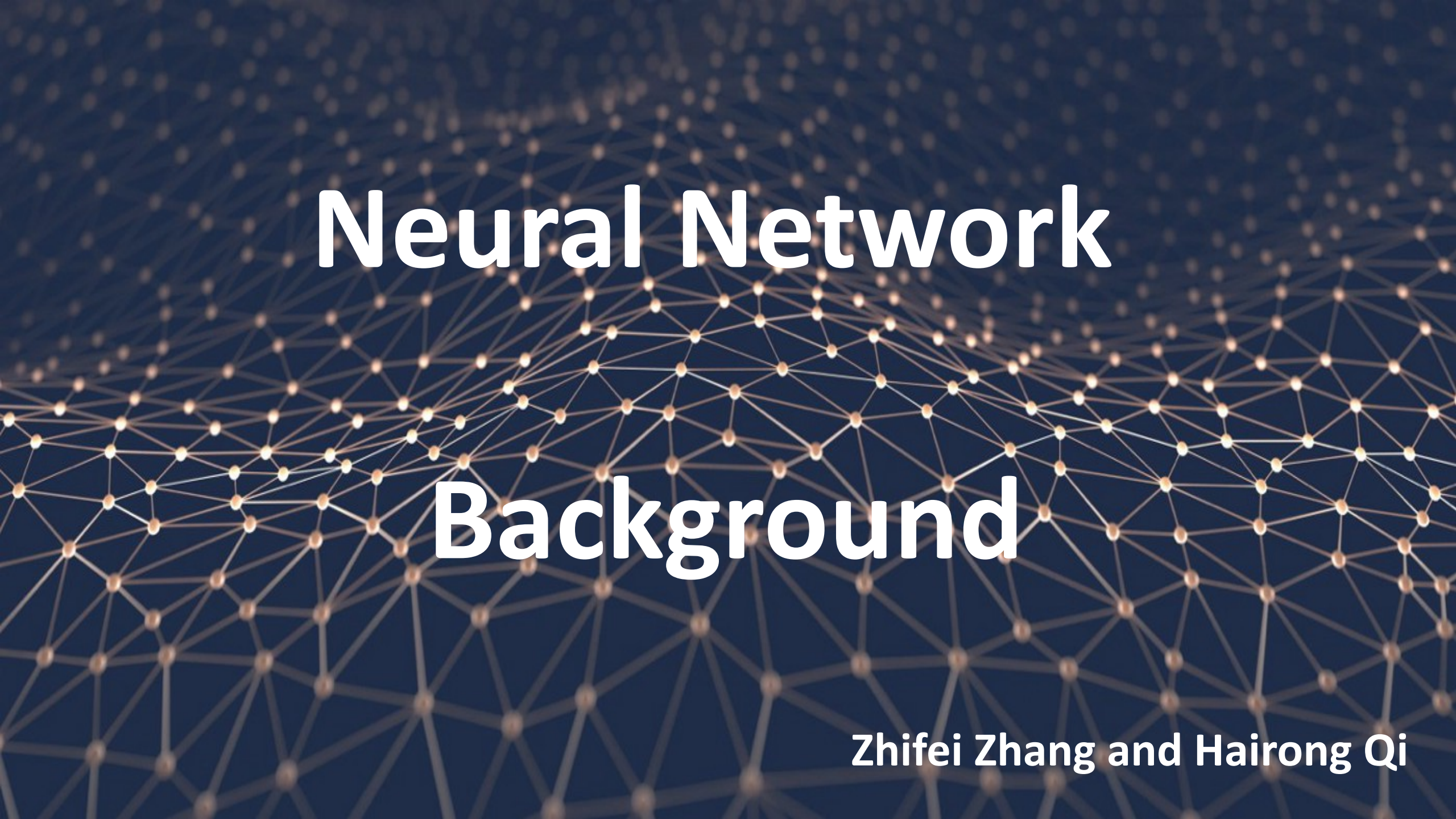
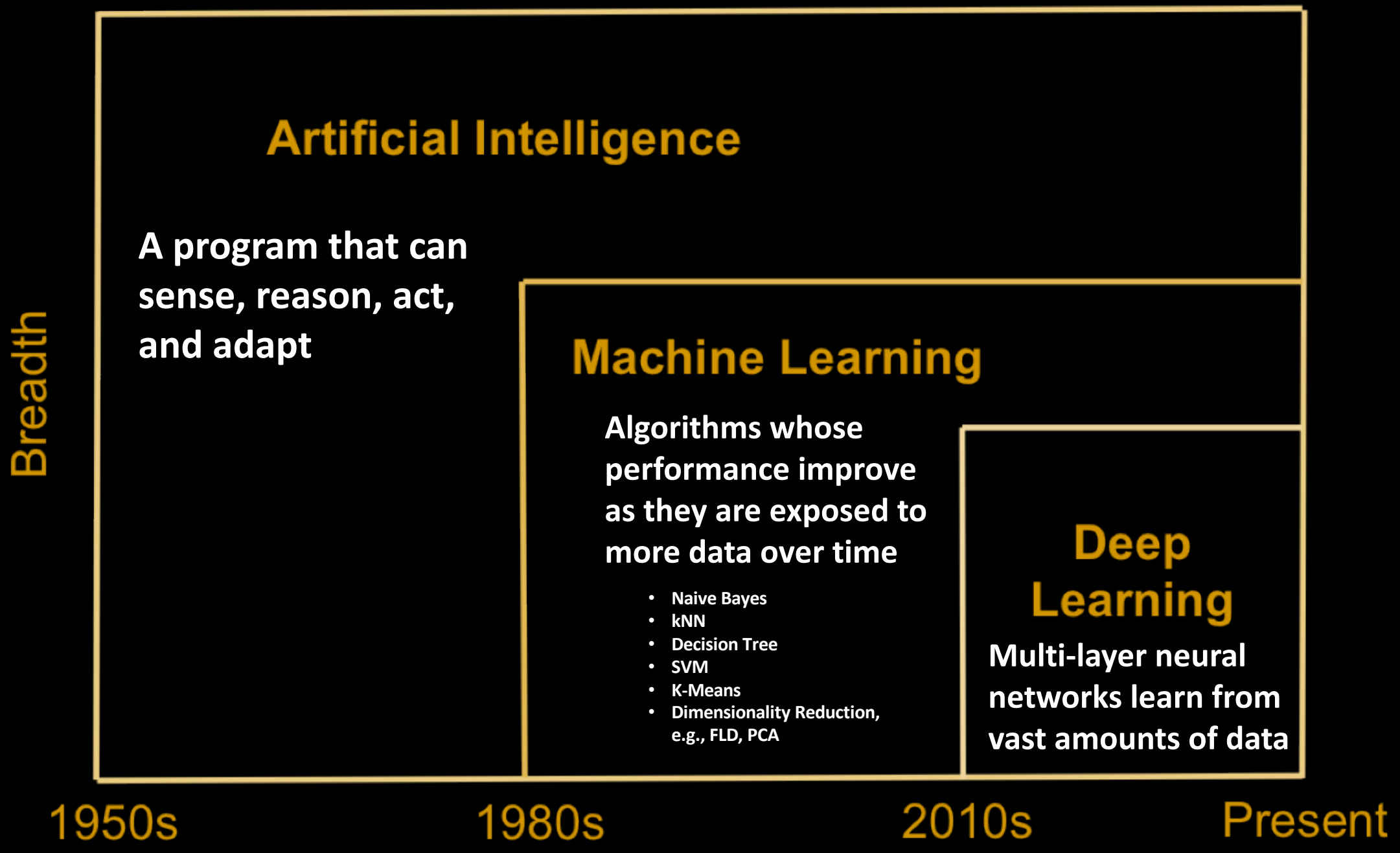

Neural Network
Background
Zhifei Zhang and Hairong Qi


Artificial Intelligence
A program that can sense, reason, act, and adapt
Machine Learning
Algorithms whose performance improve as they are exposed to more data over time
Naive Bayes
kNN
Decision Tree
SVM
K-Means
Dimensionality Reduction, e.g., FLD, PCA
Deep Learning
Multi-layer neural networks learn from vast amounts of data
Breadth
1950s
1980s
2010s
Present

> ...what we want is a machine that can learn from experience.
>
> Alan Turing, 1947

Electronic Brain — 1943

Perceptron — 1957

ADALINE — 1960

XOR Problem — 1969

Golden Age

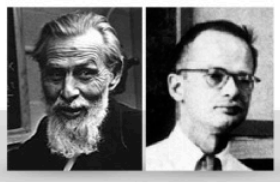
Dark Age ("AI Winter")

Multi-layered Perceptron (Backpropagation) — 1986

SVM — 1995

Deep Neural Network (Pretraining) — 2006

1940 1950 1960 1970 1980 1990 2000 2010



S. McCulloch – W. Pitts

F. Rosenblatt

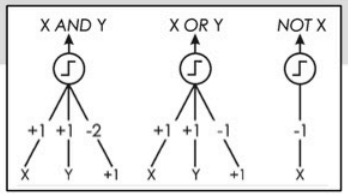
B. Widrow – M. Hoff

M. Minsky – S. Papert

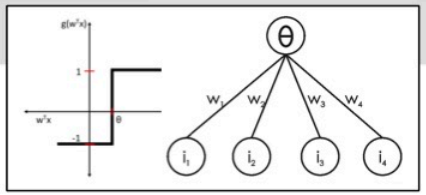
D. Rumelhart – G. Hinton – R. Wiliams

V. Vapnik – C. Cortes

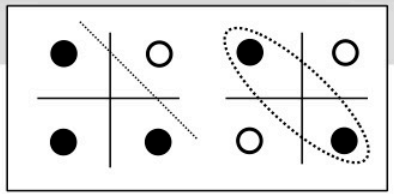
G. Hinton – S. Ruslan



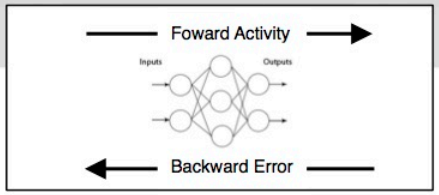
- Adjustable Weights
- Weights are not Learned



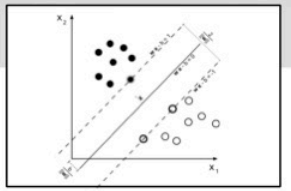
- Learnable Weights and Threshold



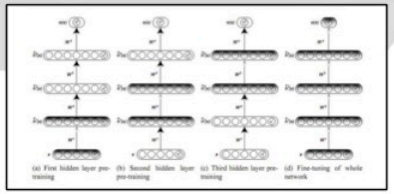
- XOR Problem



- Solution to nonlinearly separable problems
- Big computation, local optima and overfitting



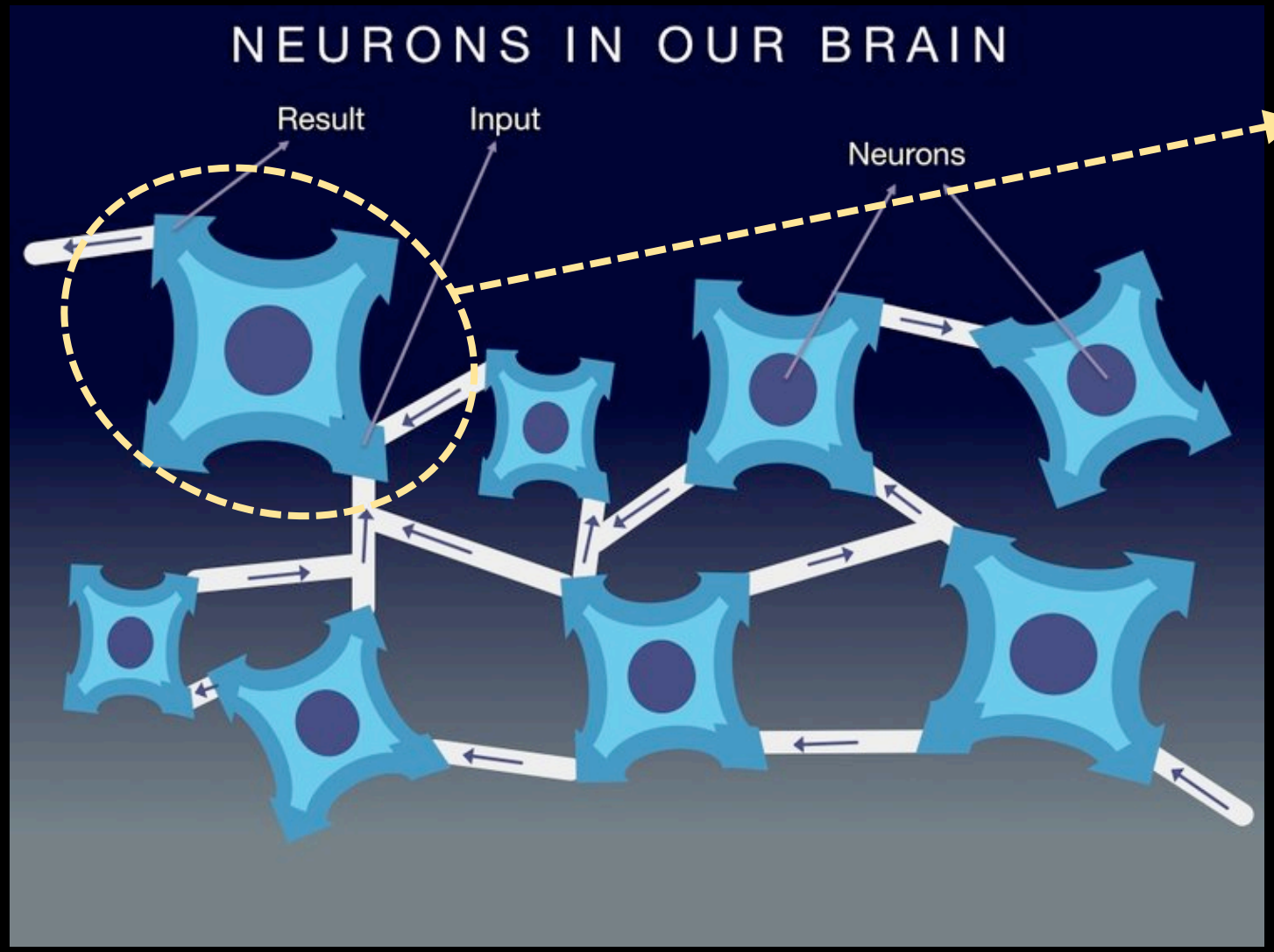
- Limitations of learning prior knowledge
- Kernel function: Human Intervention

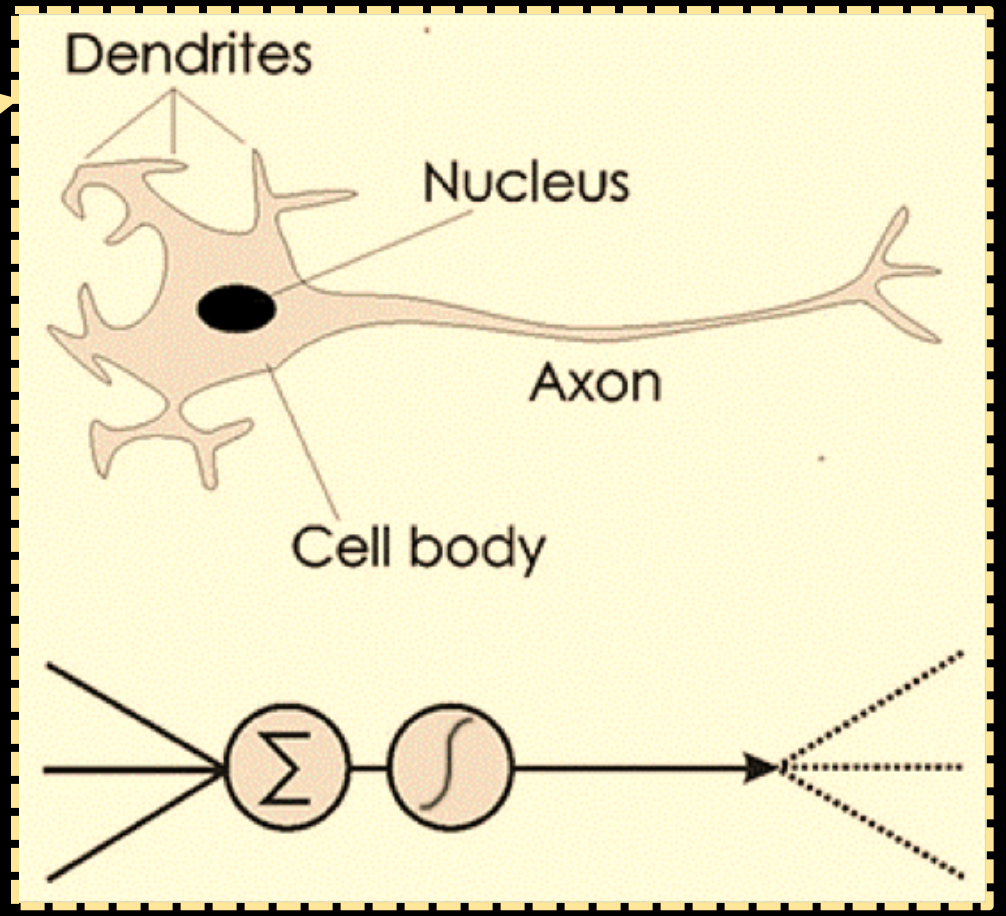


- Hierarchical feature Learning

For more detailed history of deep learning https://www.import.io/post/history-of-deep-learning/

Intuition of Neural Network

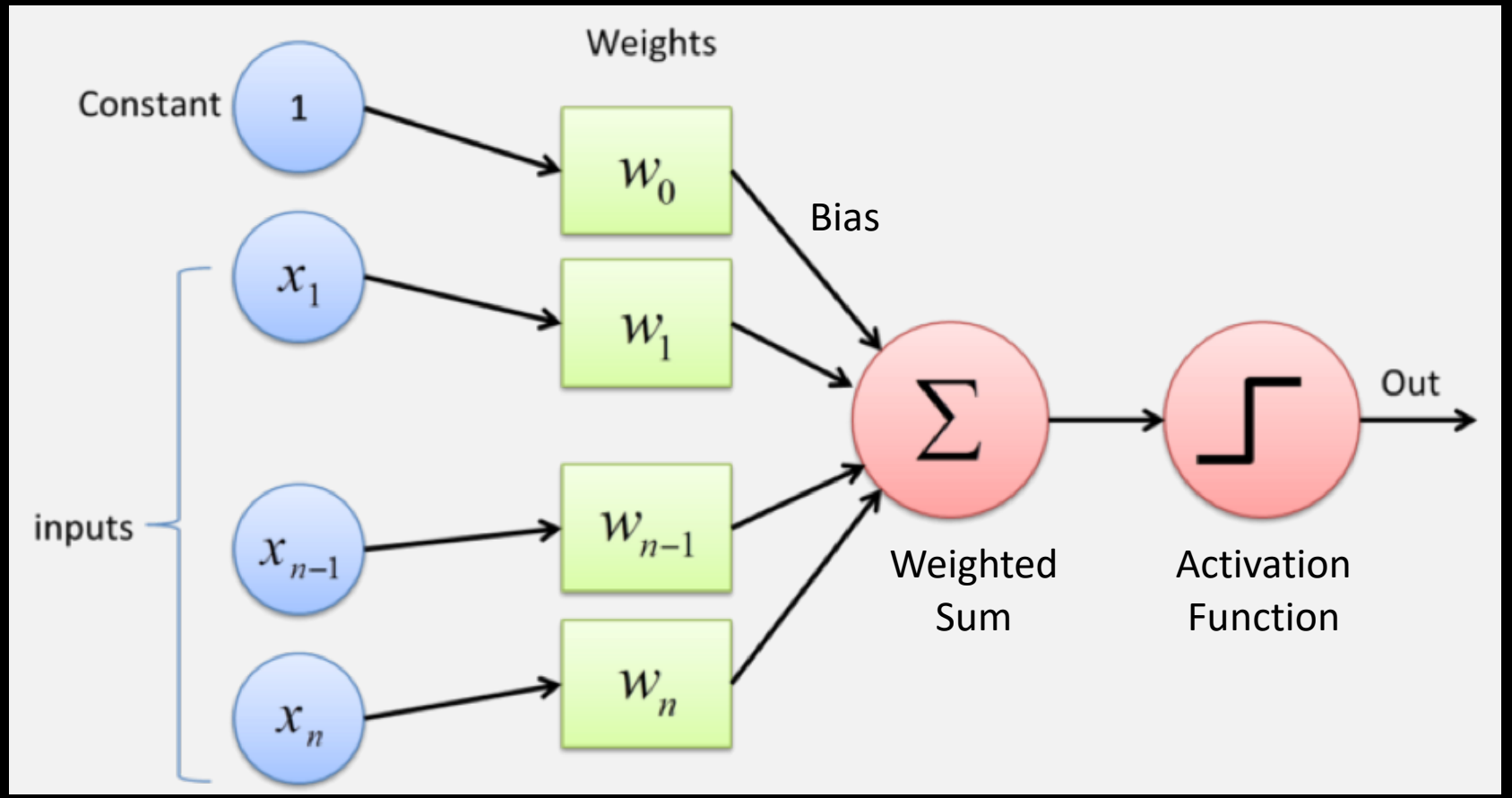




https://www.quora.com/What-is-an-intuitive-explanation-for-neural-networks

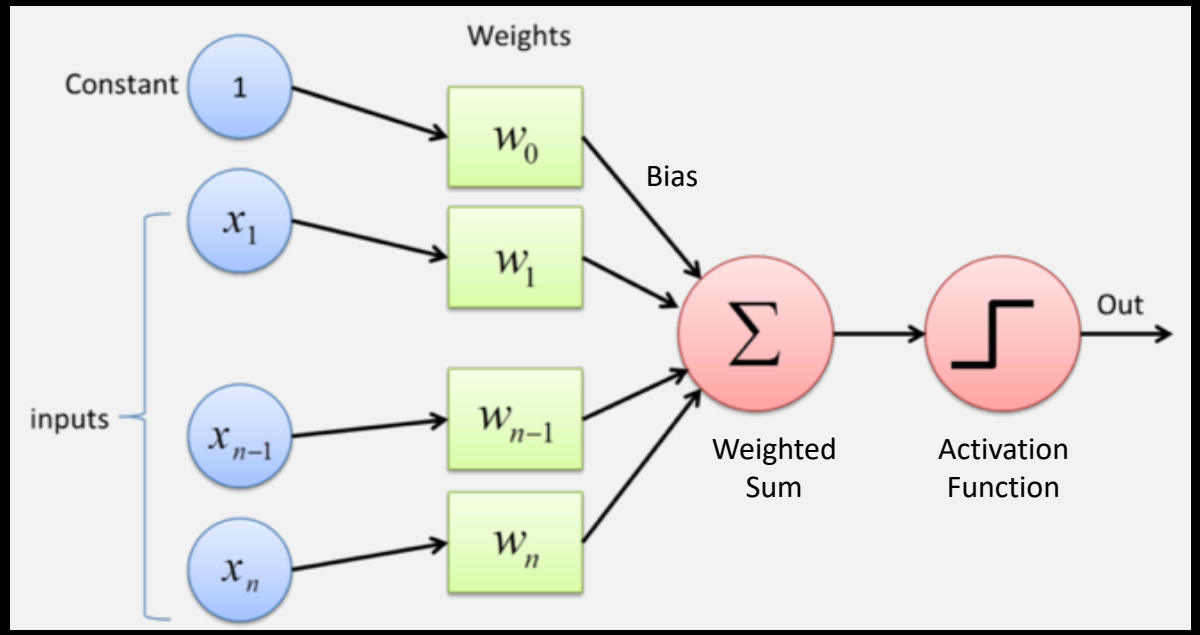
Perceptron

Perceptron is usually used to classify the data into two parts. Therefore, it is also known as a Linear Binary Classifier.



- **Weights shows the strength of the particular node.**
- **A bias value allows you to shift the activation function to the left or right.**
- **The activation function map the input between the required values, e.g., [0, 1].**

Perceptron



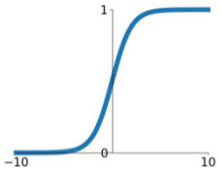
$$\begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}^T \times \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{n-1} \\ w_n \end{pmatrix} = s$$

$$F(s) = \hat{y}$$

Common activation functions:

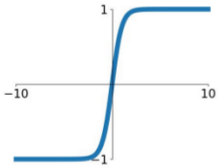
Sigmoid

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



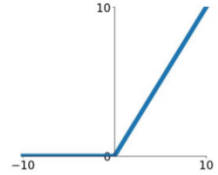
tanh

$\tanh(x)$



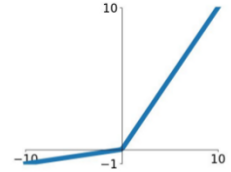
ReLU

$\max(0, x)$



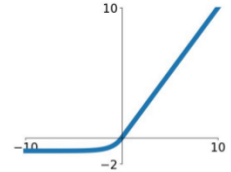
Leaky ReLU

$\max(0.1x, x)$

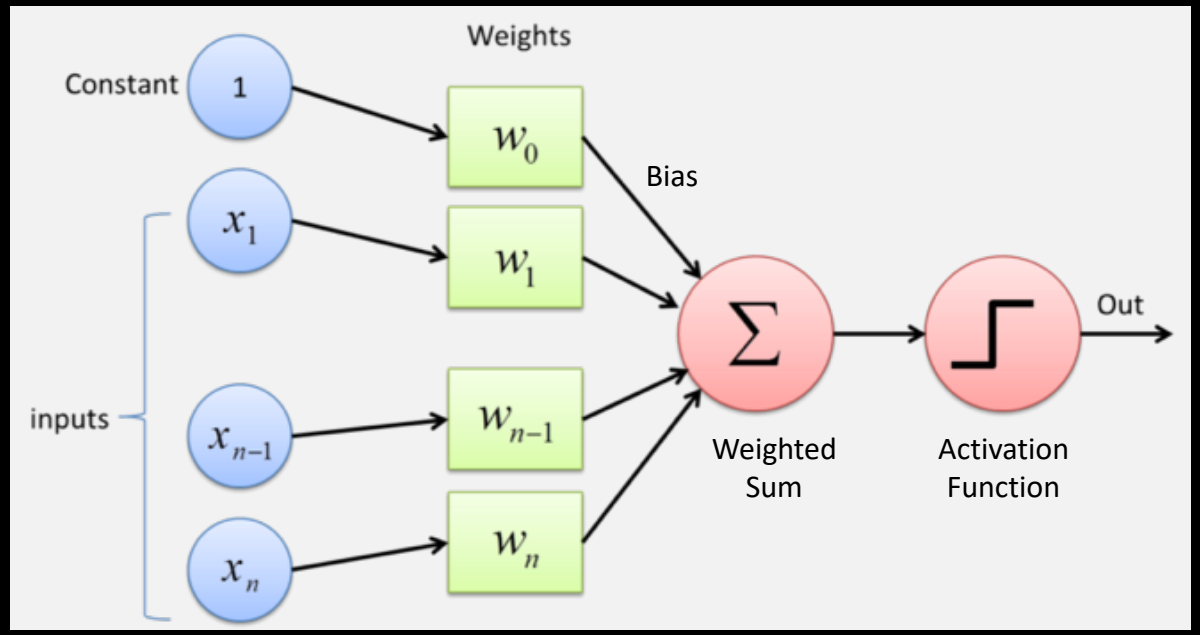


ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$



Perceptron Learning Rule



$$\begin{pmatrix} 1 \\ x_1 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}^T \times \begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_{n-1} \\ w_n \end{pmatrix} = s$$

$$F(s) = \hat{y}$$

Assume the input $x = (x_0, \ldots, x_n)$, ground truth y, and output of perceptron $\hat{y}$, the weight is updated as follow:

$$w_i := w_i - \alpha(\hat{y} - y)x_i$$

Where α is the learning rate that is a positive value in the range of (0, 1).

Objective: $E = \frac{1}{2}(y - \hat{y})^2$

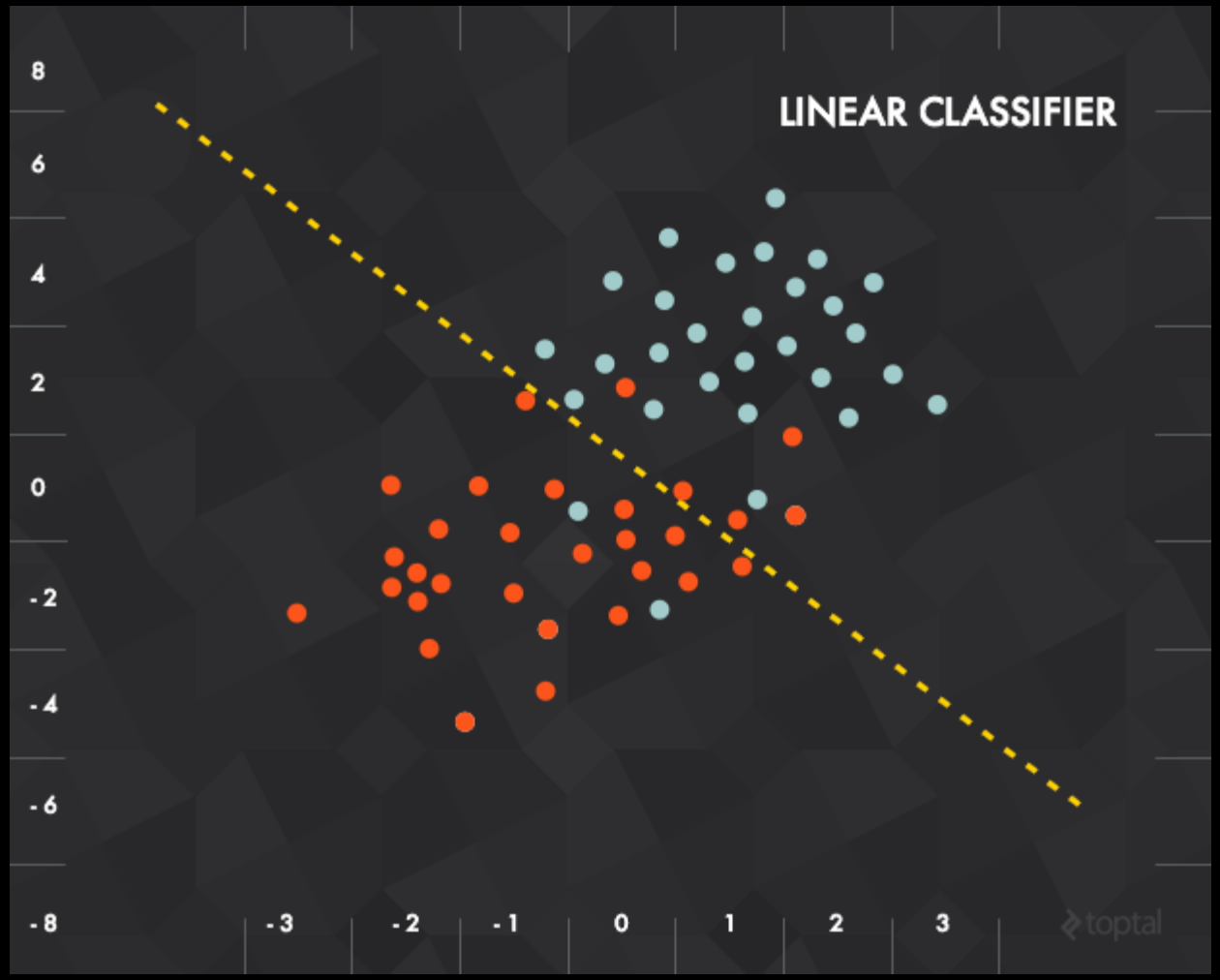
Partial derivative of E w.r.t. w_i:

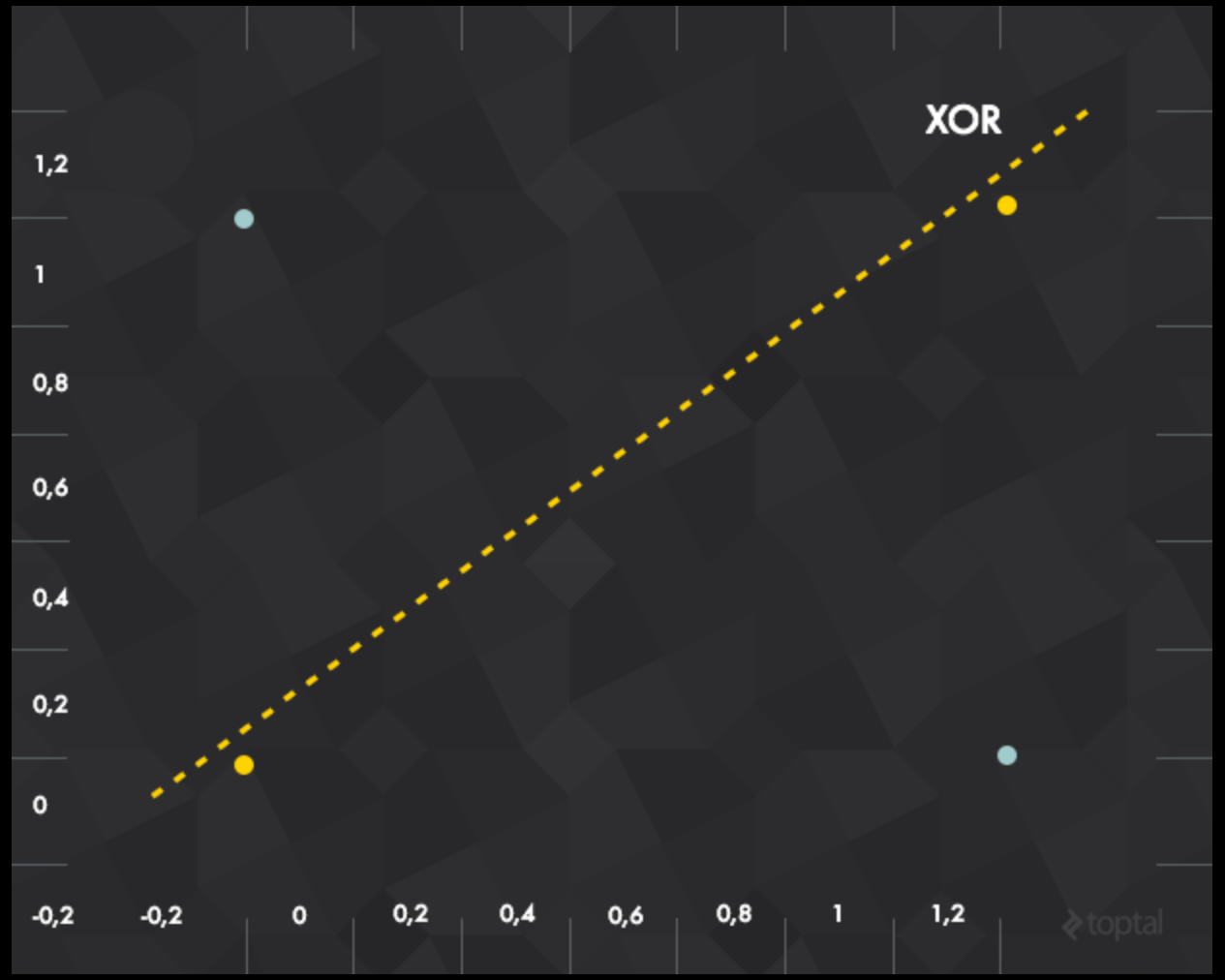
$$\Delta w_i = \frac{\partial E}{\partial w_i} = \frac{\partial E}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s} \frac{\partial s}{\partial w_i}$$

$$\Delta w_i = (\hat{y} - y) \cdot \underbrace{F'(s)} \cdot x_i$$

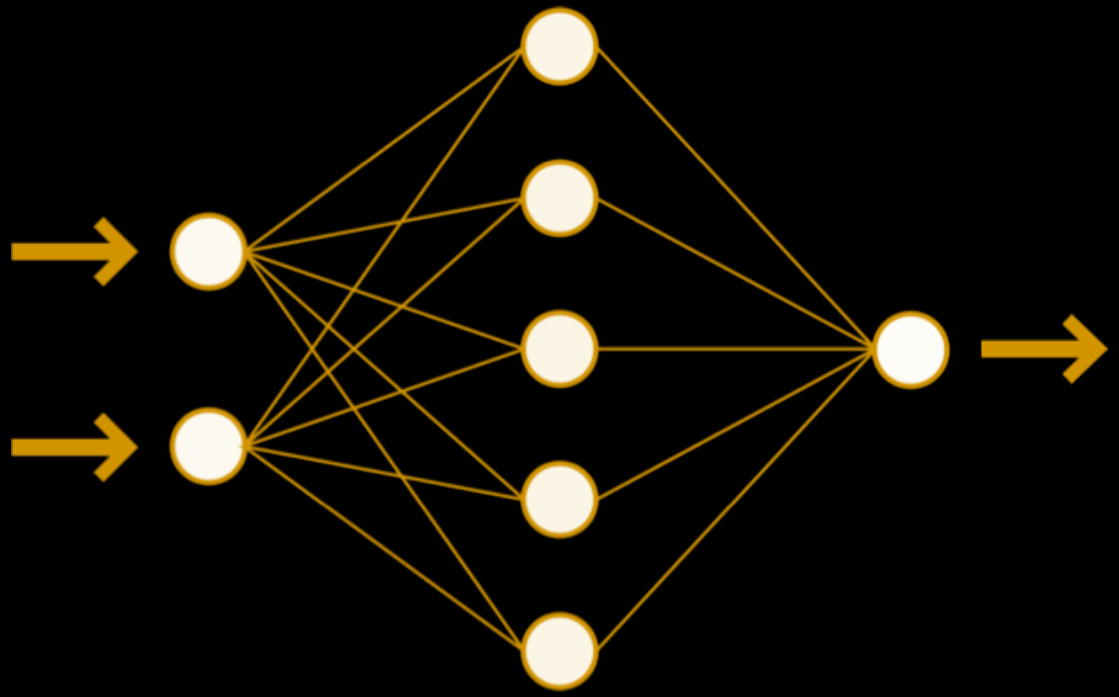
Assume 1, directly pass the error backward

XOR Problem

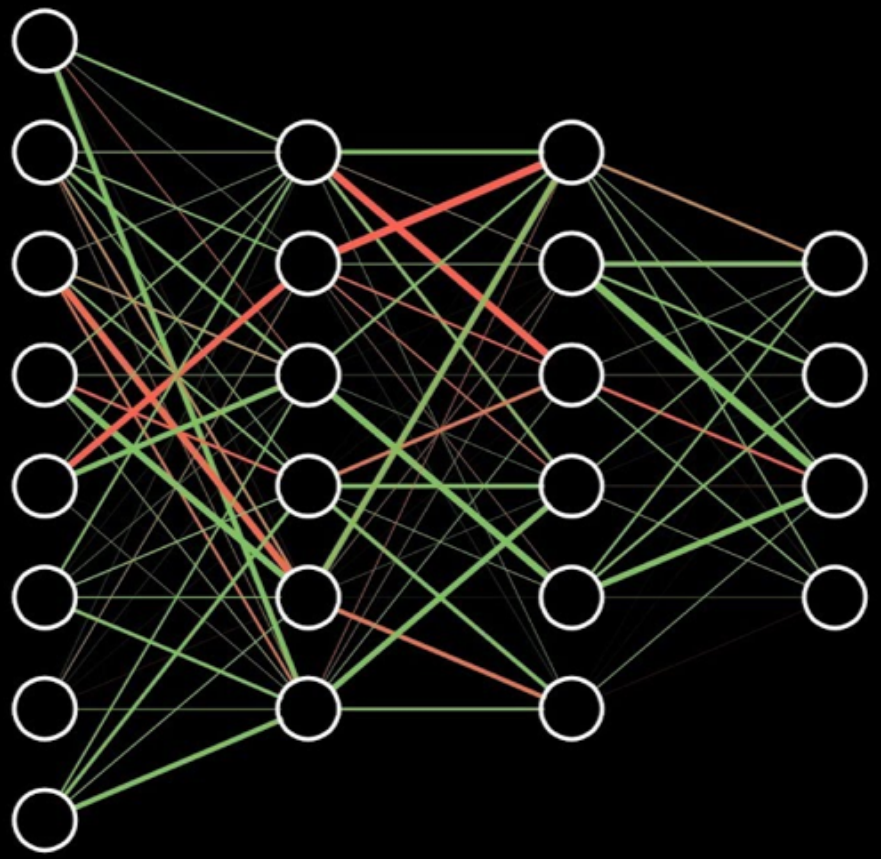

LINEAR CLASSIFIER
8
6
4
2
0
-2
-4
-6
-8
-3
-2
-1
0
1
2
3
toptal


XOR
1,2
1
0,8
0,6
0,4
0,2
0
-0,2
-0,2
0
0,2
0,4
0,6
0,8
1
1,2
toptal

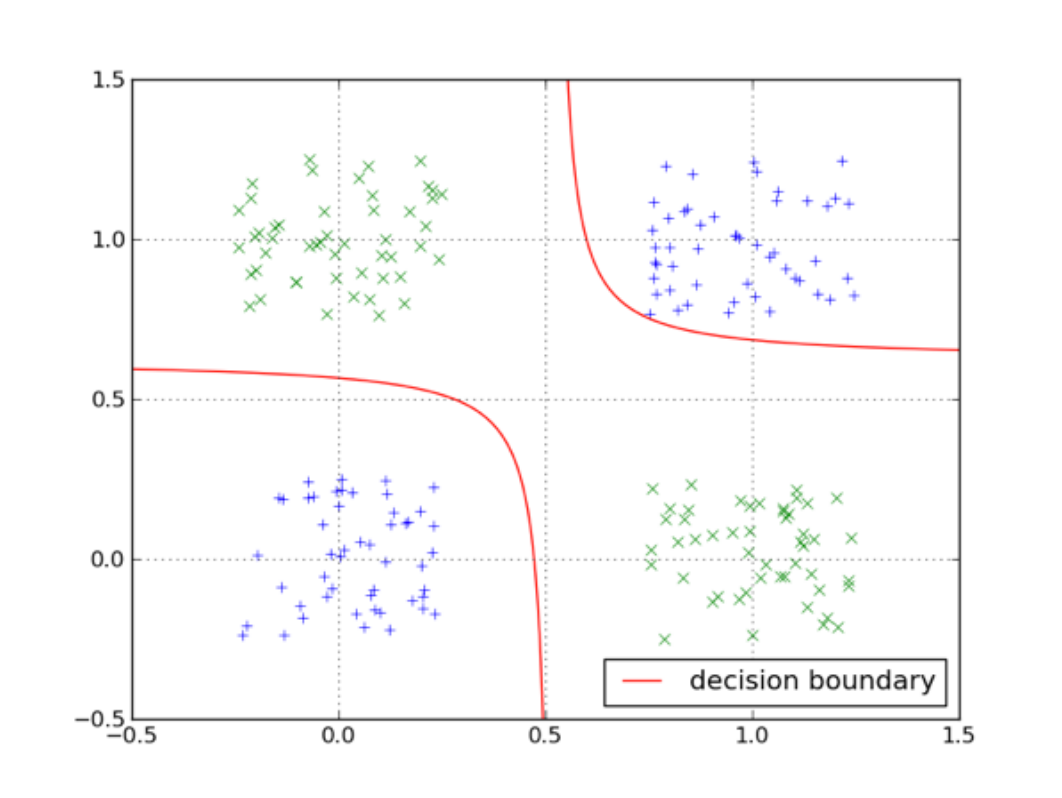
Multi-layer Perceptron --- Neural Network

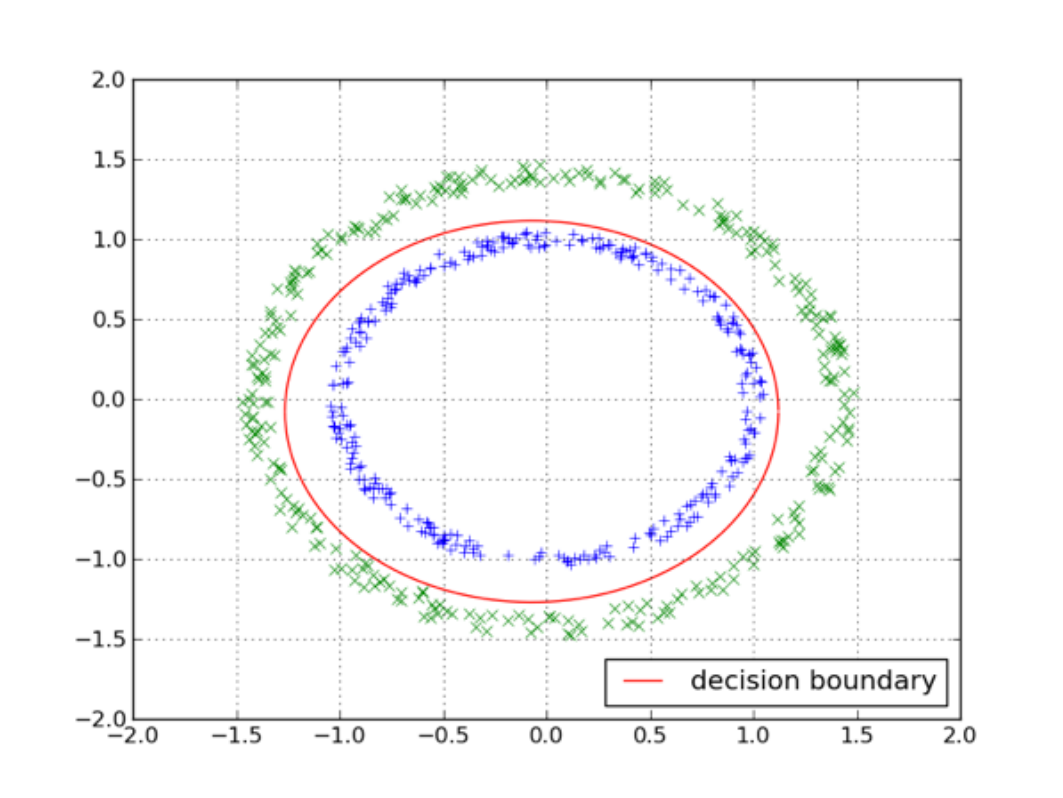


Neural Network in more practice



Multi-layer Perceptron --- Solve the XOR Problem



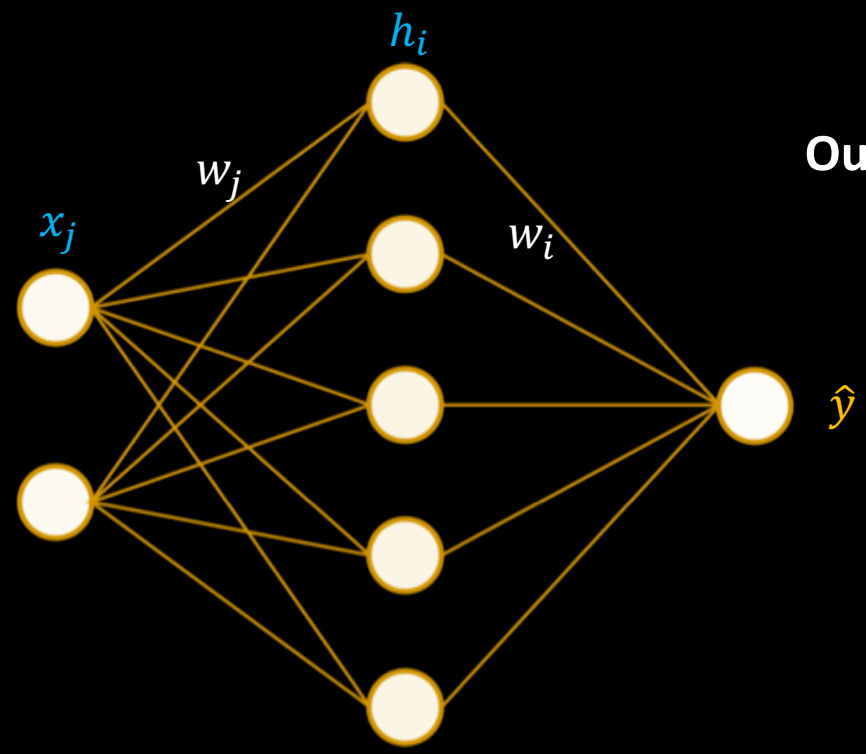


Online demo: https://lecture-demo.ira.uka.de/neural-network-demo/

Backpropagation

How the multi-layer perceptron (neural network) is learned?

Back propagate the error layer-by-layer



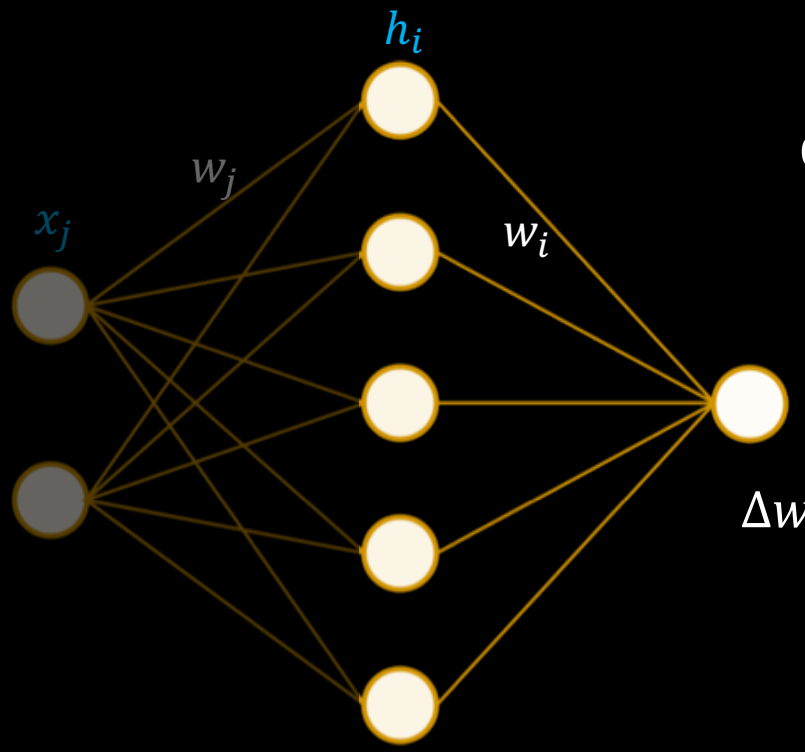
Output error:

$$E = \frac{1}{2}(y - \hat{y})^2$$

Backpropagation

How those multi-layer perceptron (neural network) is learned?

Back propagate the error layer-by-layer



Output error:

$$E = \frac{1}{2}(y - \hat{y})^2$$

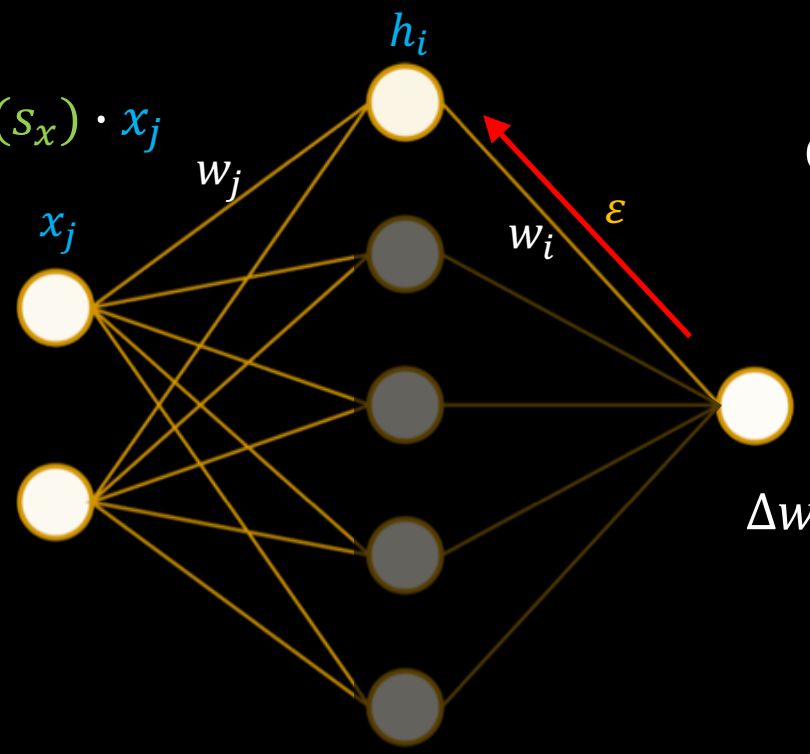
$$\Delta w_i = (\hat{y} - y) \cdot F'(s_h) \cdot h_i$$

Backpropagation

How those multi-layer perceptron (neural network) is learned?

Back propagate the error layer-by-layer, and

$$\Delta w_j = \varepsilon \cdot F'(s_x) \cdot x_j$$



Output error:

$$E = \frac{1}{2}(y - \hat{y})^2$$

Back propagated error:

$$\varepsilon = \frac{\partial E}{\partial h_i} = \frac{\partial E}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s_h} \frac{\partial s}{\partial h_i}$$

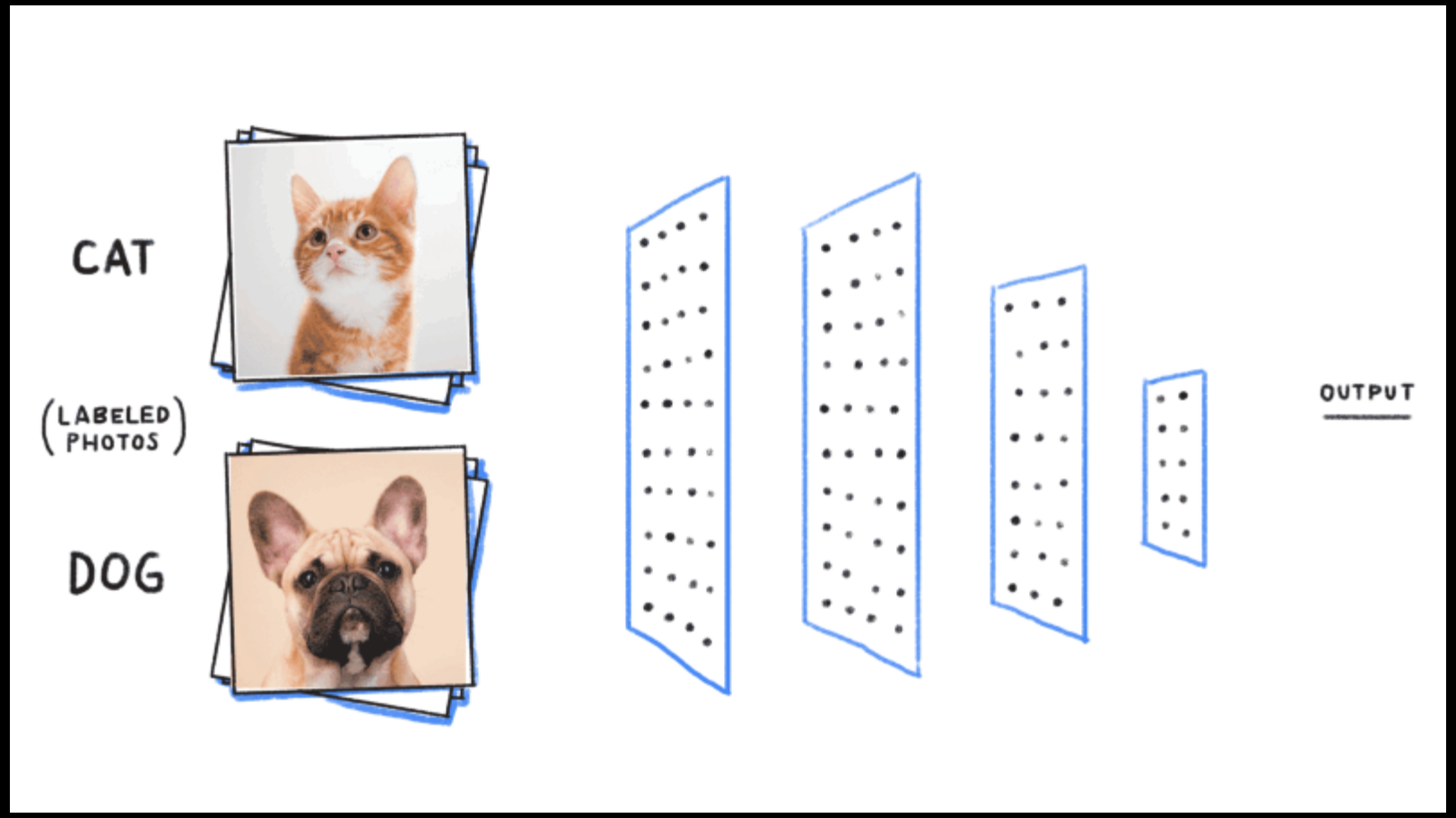
$$\varepsilon = (\hat{y} - y) \cdot F'(s_h) \cdot w_i$$

$$\Delta w_i = (\hat{y} - y) \cdot F'(s_h) \cdot h_i$$

Hidden nodes are independent from each other

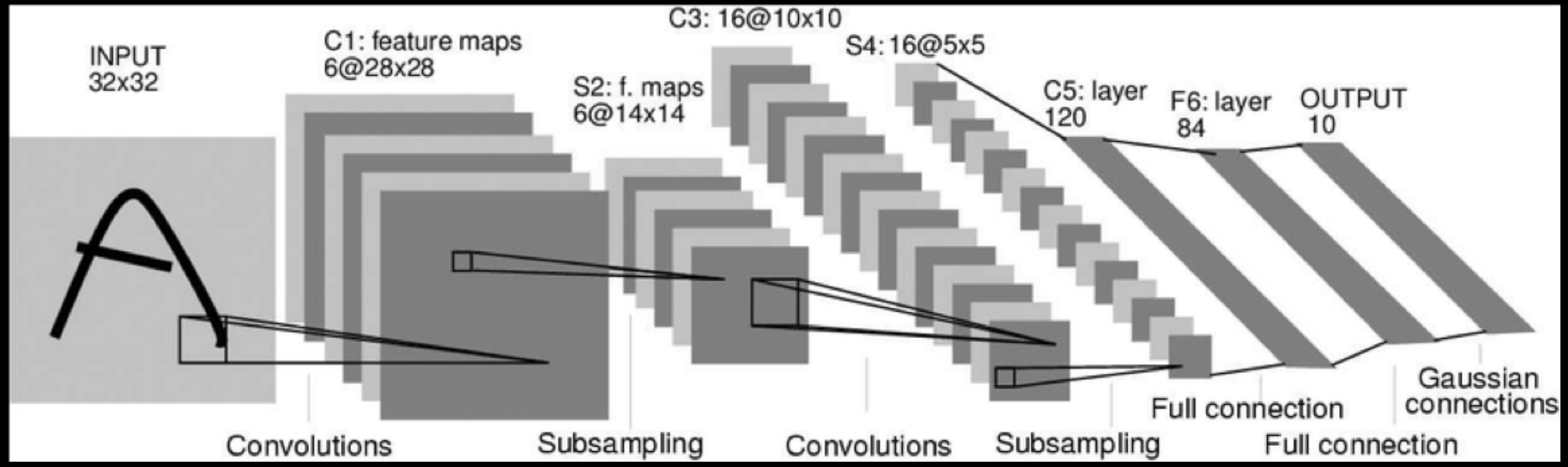
Convolutional Neural Network (CNN)

Handle images which need to preserve the special structure



Convolutional Neural Network (CNN)

LeNet-5 by Yann LeCun et al., 1998



A demo code in Matlab: https://github.com/ZZUTK/An-Example-of-CNN-on-MNIST-dataset

Convolutional Neural Network (CNN)

Appearance of CNN is early (back to 1998) but it booms around 2010s, WHY?

Large dataset (over 10 million)

From 2009

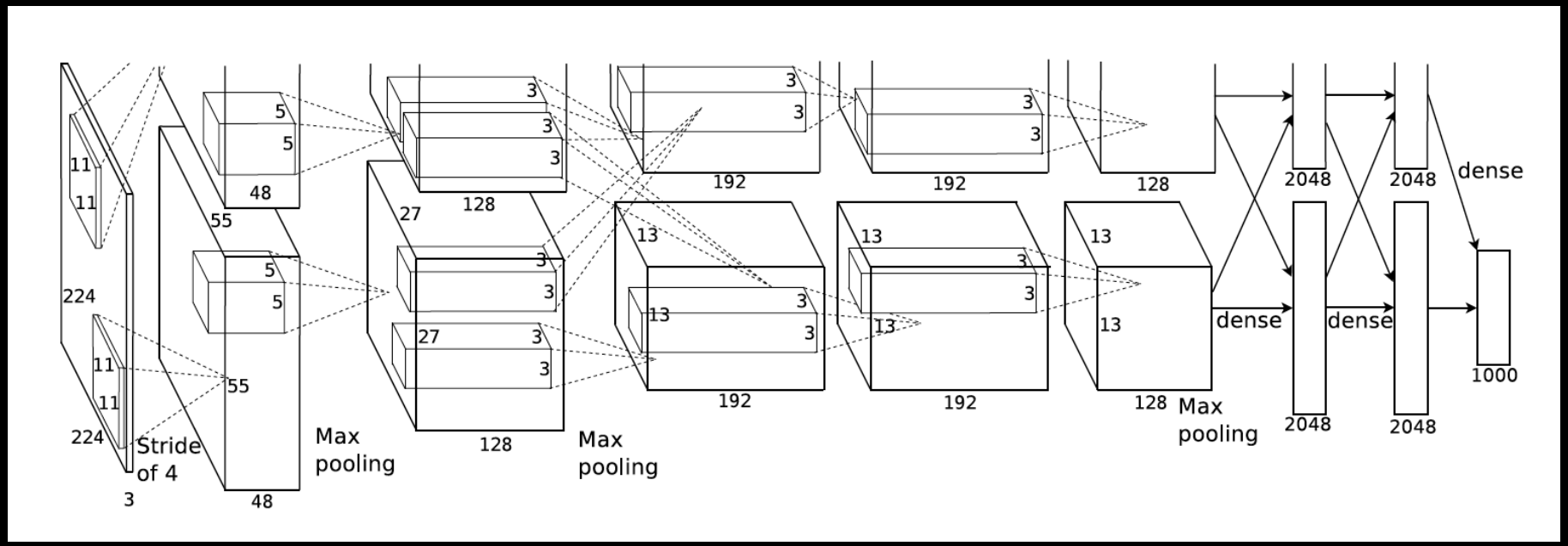
Powerful computing unit

From 2010

Convolutional Neural Network (CNN)

Appearance of CNN is early (back to 1998) but it booms around 2010s, WHY?

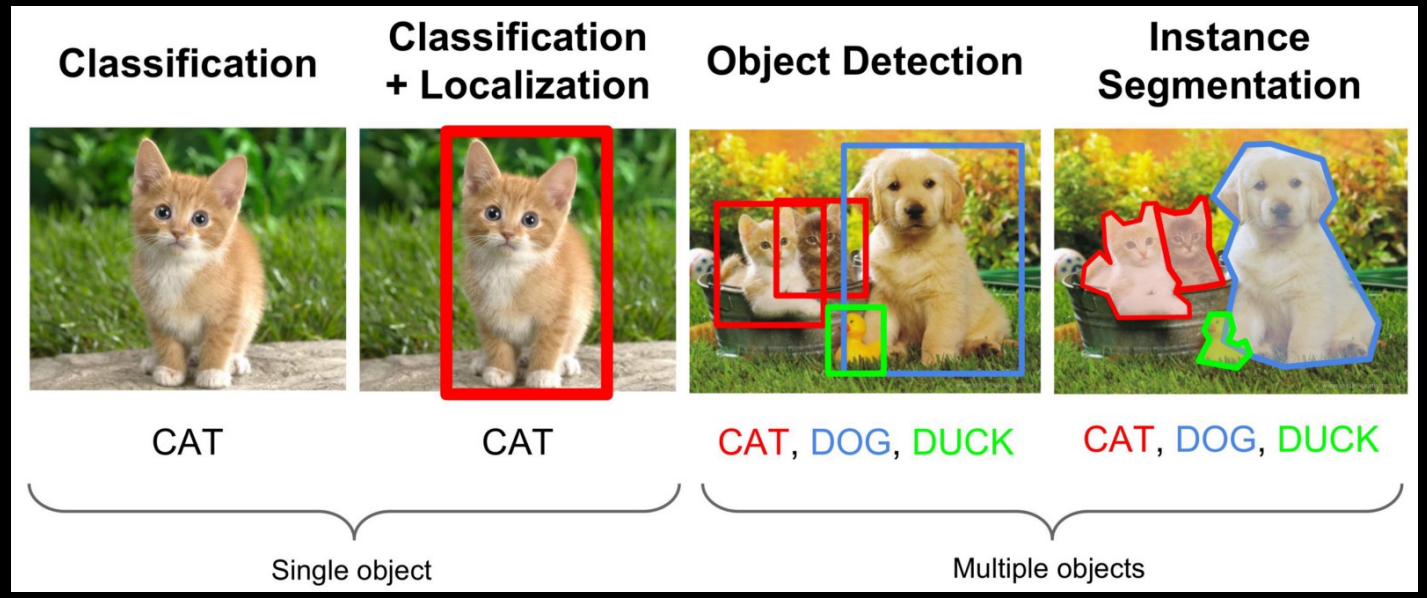
Deep convolutional neural networks with competitive performance to human



AlexNet, 2012

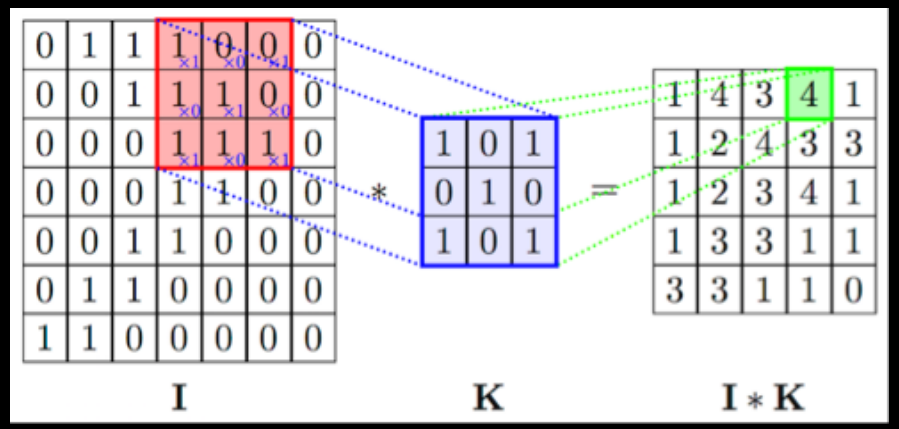
Convolutional Neural Network (CNN)

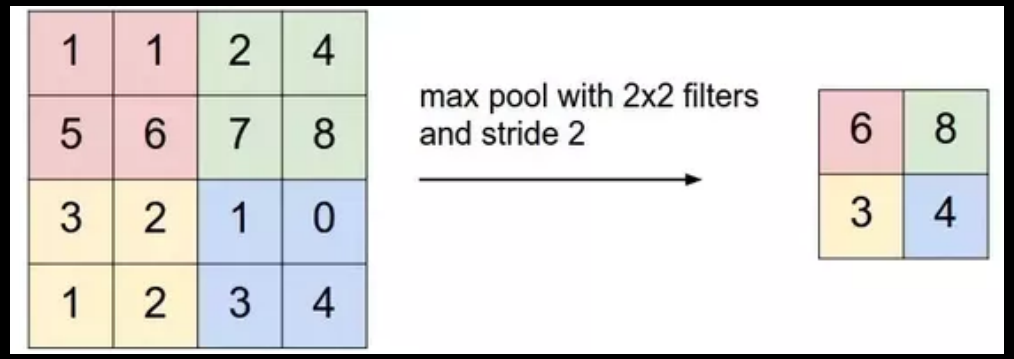
Since 2010s, the deep convolutional neural networks are mostly referred to as deep learning, and it flourishes in computer vision area until today.

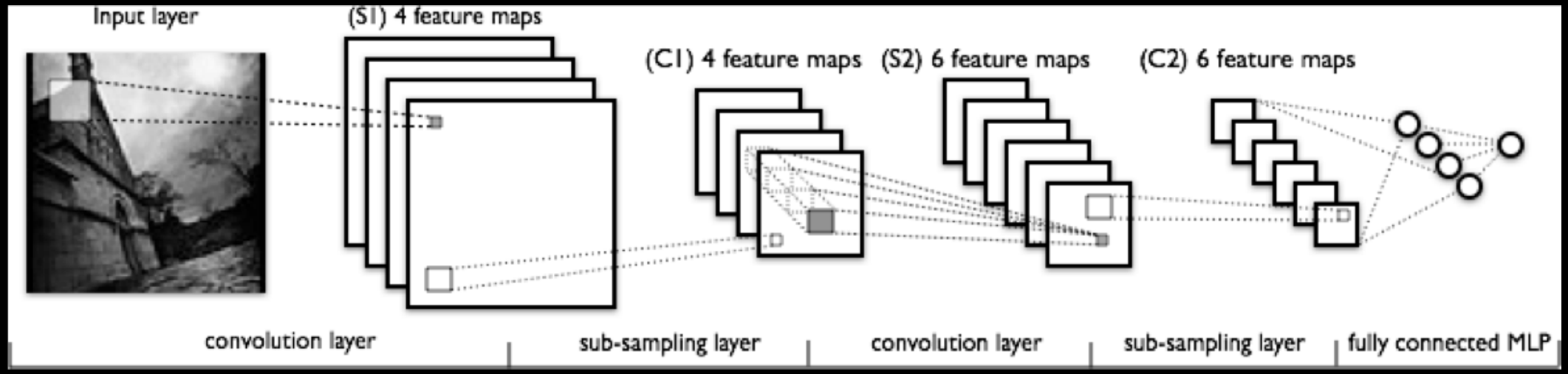


Convolutional Neural Network (CNN)

Convolution and Pooling



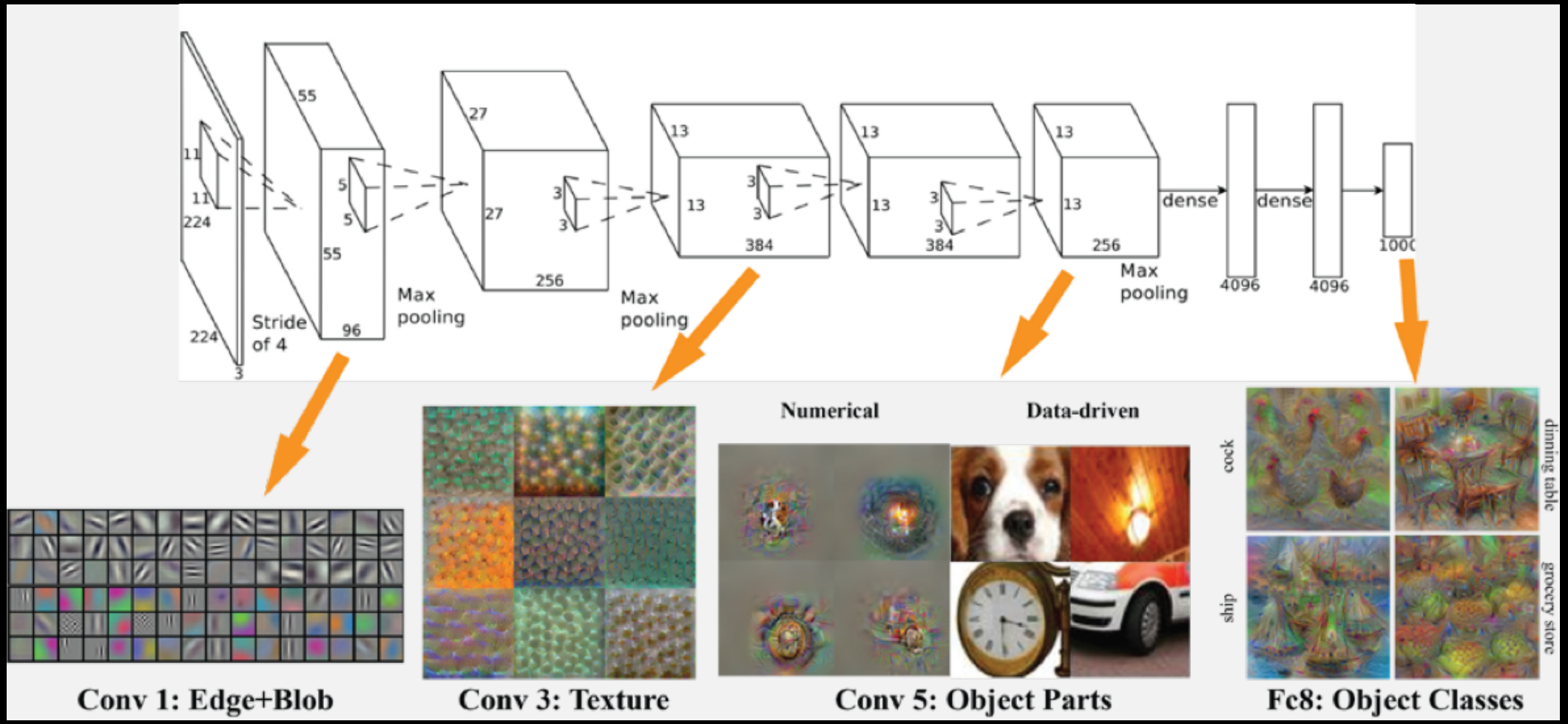




Where is the activation function?

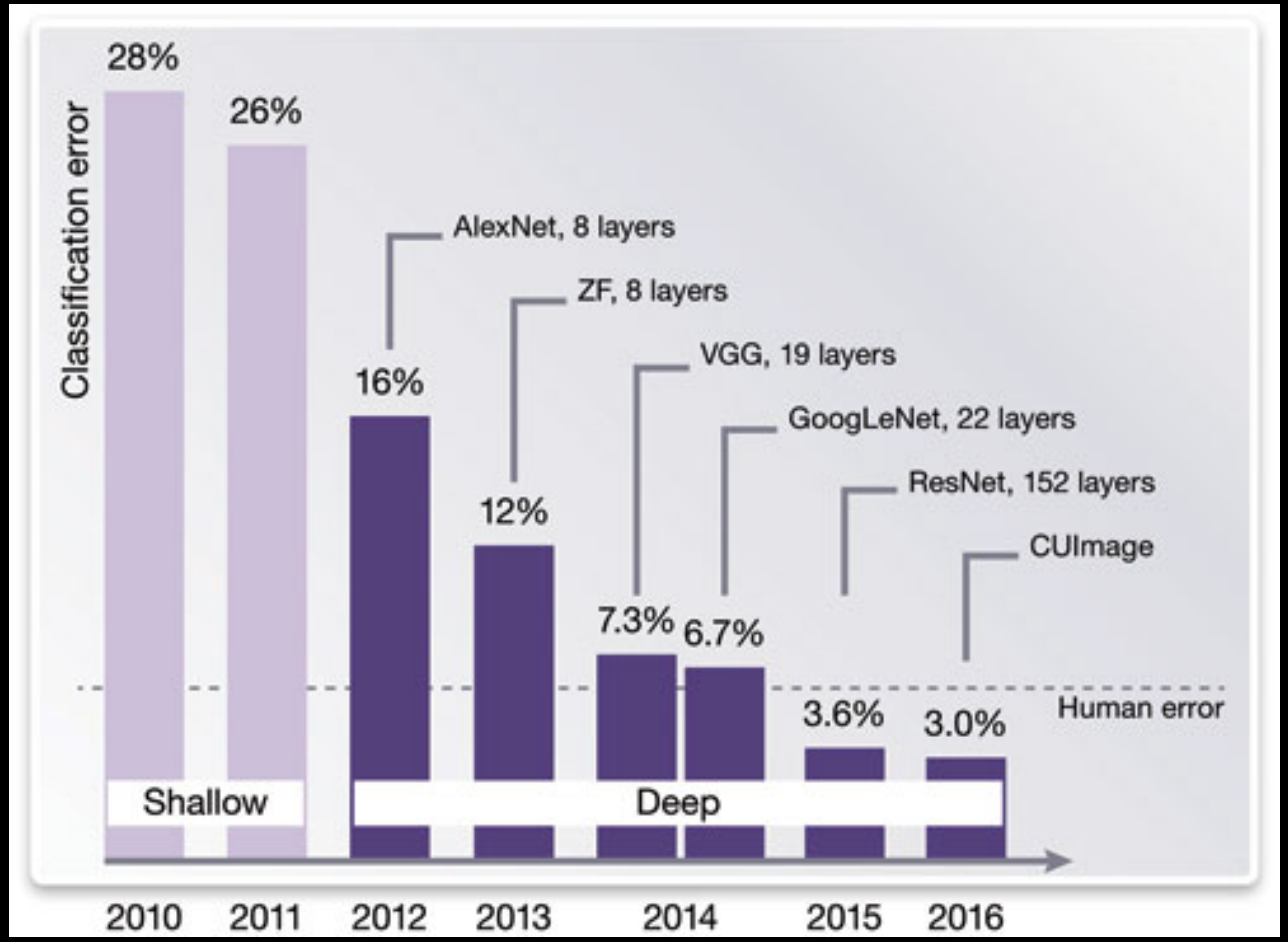
Convolutional Neural Network (CNN)

Why deep? What does each layer learn?



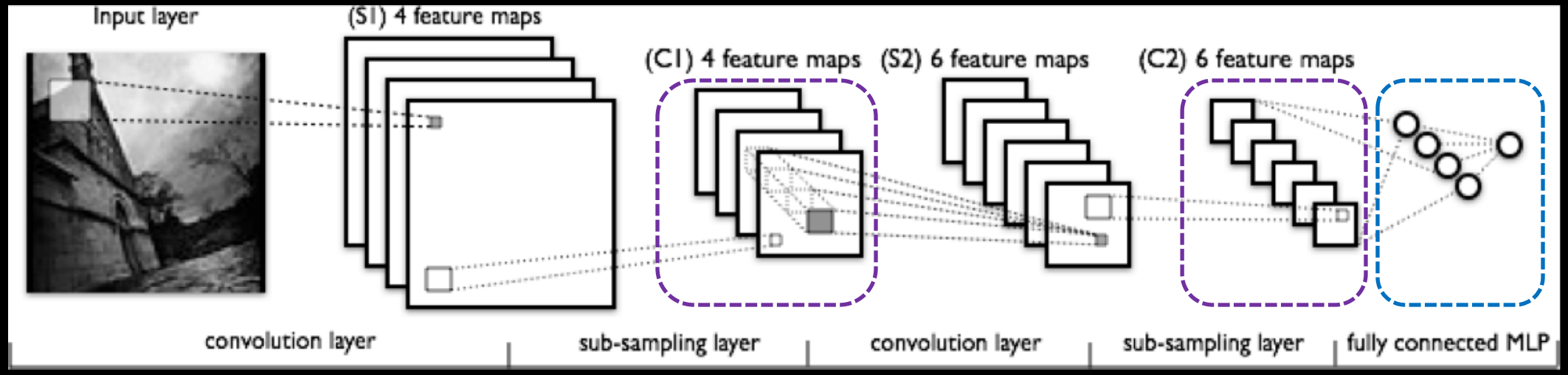
Convolutional Neural Network (CNN)

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) results show deep learning surpassing human levels of accuracy



Convolutional Neural Network (CNN)

How to learn a CNN? Again, **Backpropagation**.



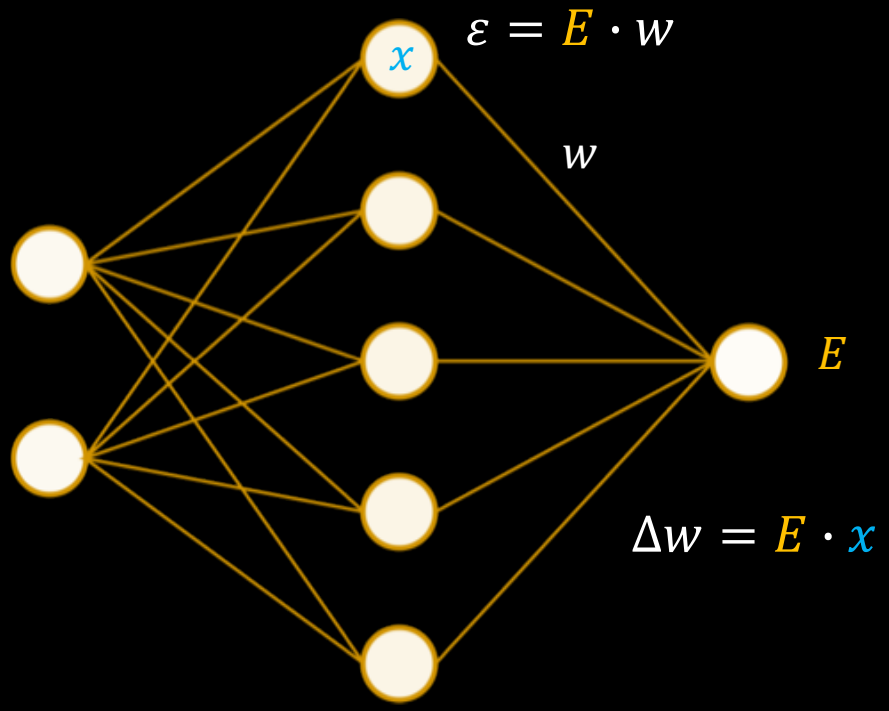
No parameters, do not need update.
Directly pass the error backward.

The same as multi-layer perceptron

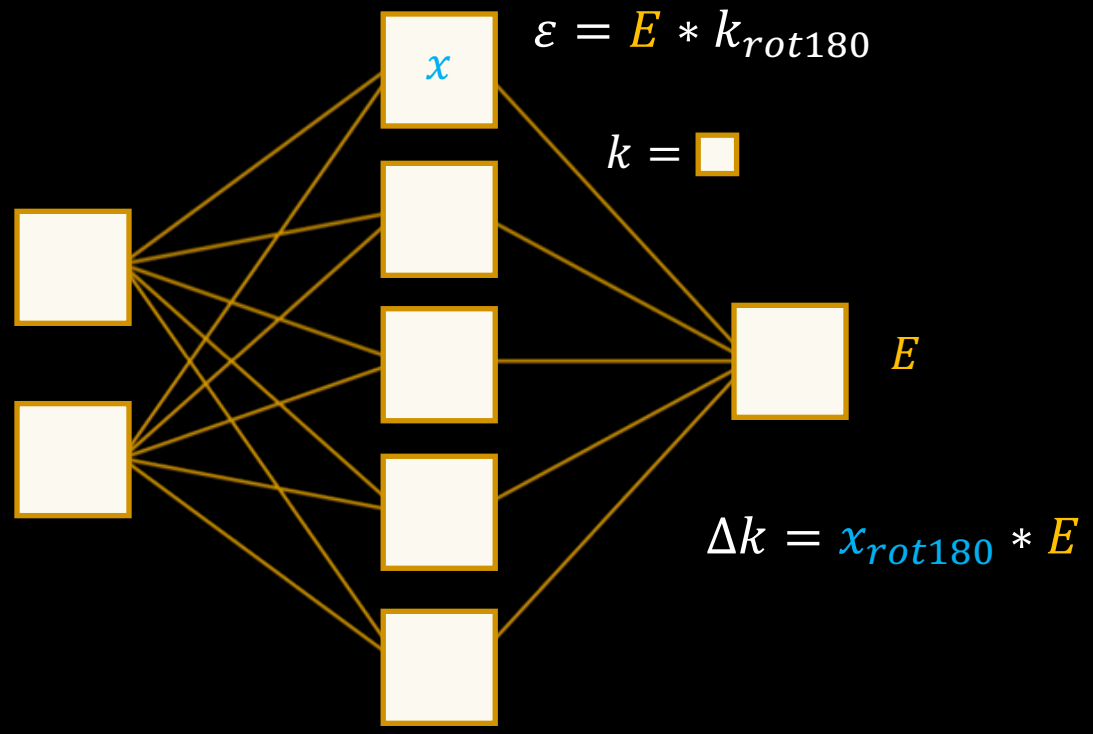
Convolutional Neural Network (CNN)

How to learn a CNN? Again, Backpropagation.

Multi-layer Perceptron

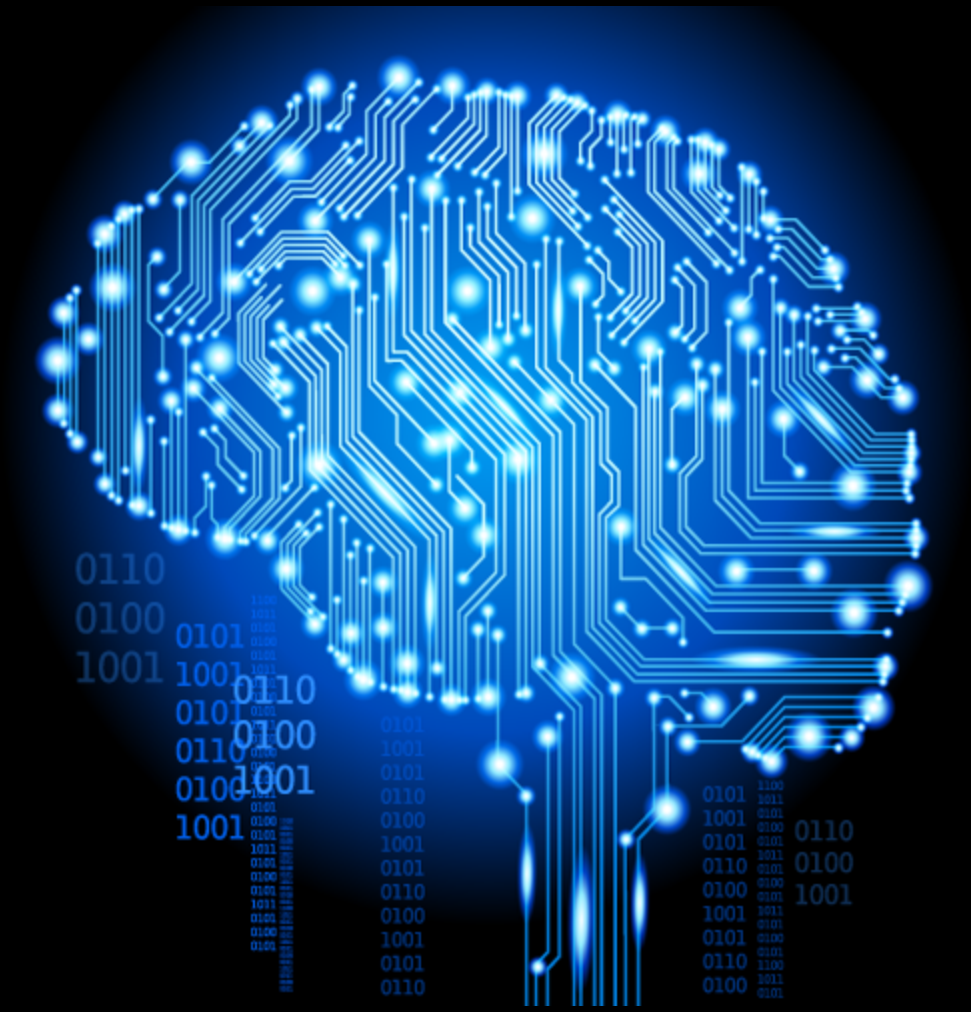


Convolutional Neural Network



For simplicity, assume $F'(s) = 1$ here.

https://github.com/ZZUTK/An-Example-of-CNN-on-MNIST-dataset/blob/master/doc/Derivation_of_Backpropagation_in_CNN.pdf


0110
0100
1001
0101
1001
0110
0100
1001
0101
1001
0110
0100
1001
0101
0110
0100
1001
0101
0110
0100
1001
0101
1001
0110
0100
1001
0101
0110
0100
1001
0101
0110