

HIT391

MACHINE LEARNING: ADVANCEMENTS AND APPLICATIONS

- Lecturer: Dr. Yan Zhang
- Email: yan.zhang@cdu.edu.au
- Office: Purple 12.3.4



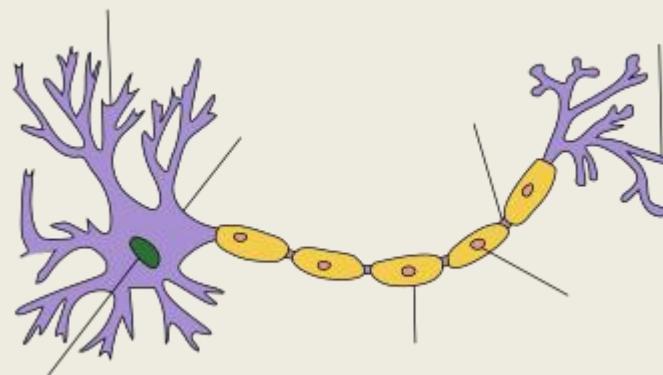
Week 9:

Deep Learning

- **Learning Outcomes**
 - Single Layer Perceptron
 - Activation Function
 - Loss Function
 - Multi-Layer Perceptron (MLP)
 - Gradient Decent

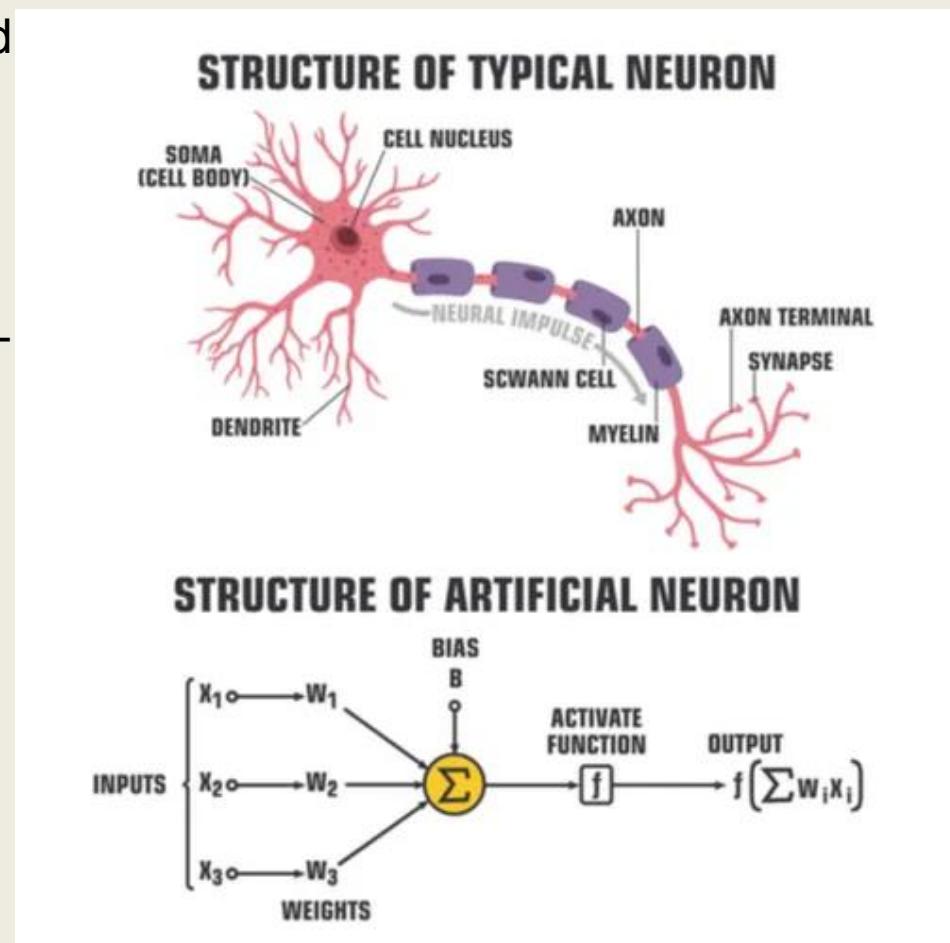
Biological Neuron

- In the biological neural network, each neuron is connected to other neurons. When it "fires," it sends chemical signals to the connected neurons, thereby altering the electrical potential within these neurons. If the electrical potential of a neuron surpasses a "threshold," it becomes activated, meaning it "fires" and sends chemical signals to other neurons.



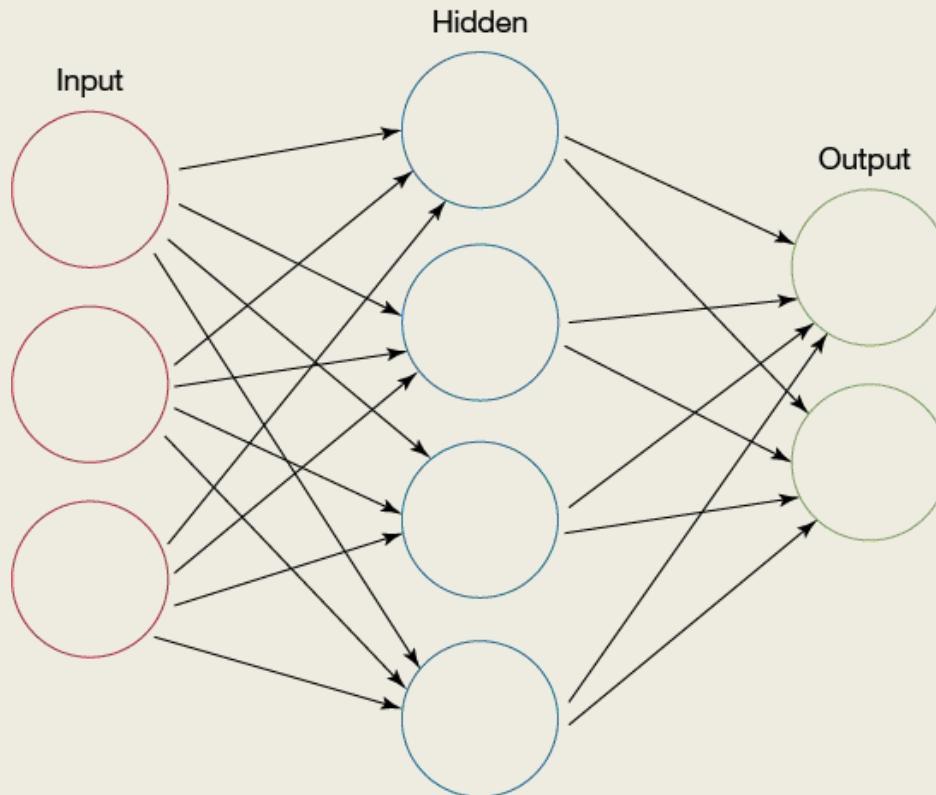
Biological Neuron vs Artificial Neuron

- A neural network is a widely interconnected network comprised of simple units with adaptability, capable of simulating the interactive responses of the **biological neuron** system to real-world objects.
- Artificial neuron aims to **mimic our brain's neural networks (biological neuron)**, which
- It gained massive attention in recent year by its performance in the field like **image recognition, auto driving cars, NLP etc**



Artificial Neuron

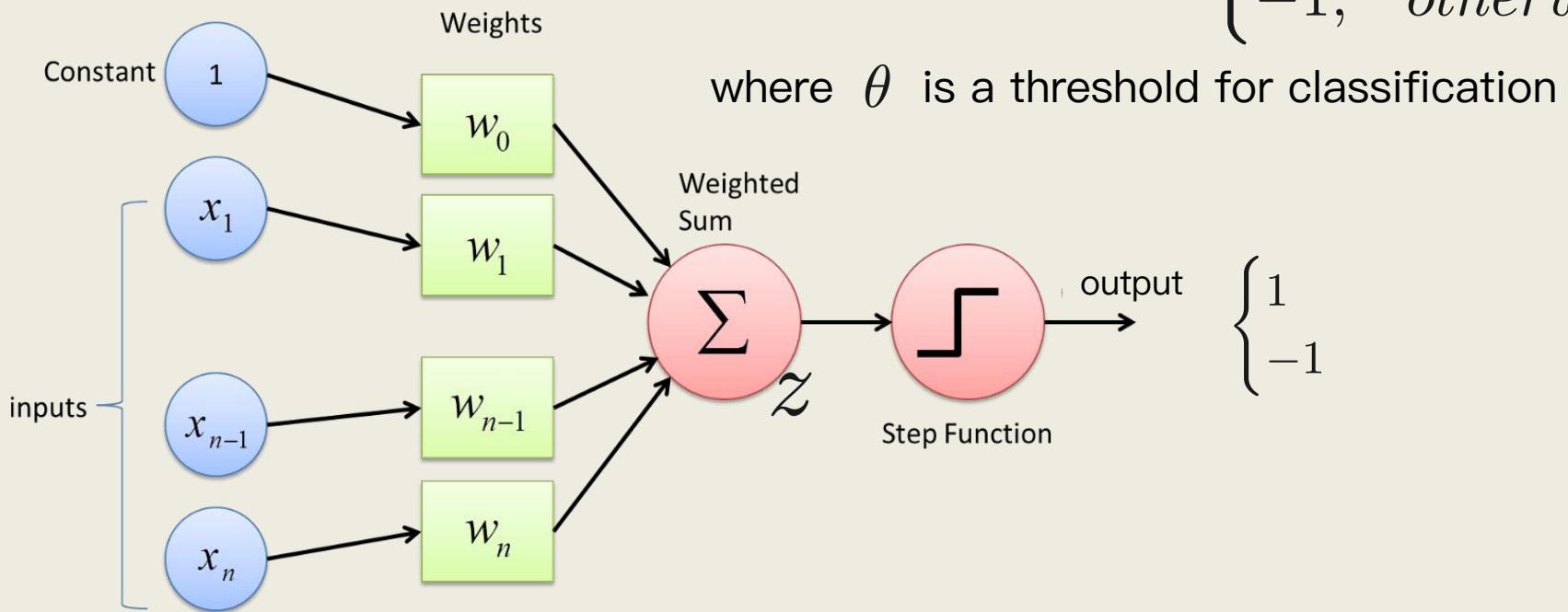
- The neural network is a widely interconnected network composed of adaptable simple units, capable of simulating the interactive responses of the biological nervous system to real-world objects.



Single Layer Perceptron

- A perceptron is a linear classifier
 - an algorithm that classifies input by separating two categories with a linear function z and a step function g .
 - Linear function: $z(\mathbf{x}) = \mathbf{w}\mathbf{x} + b$

– Step function: $g(z) = \begin{cases} 1, & z \geq \theta \\ -1, & \text{otherwise.} \end{cases}$

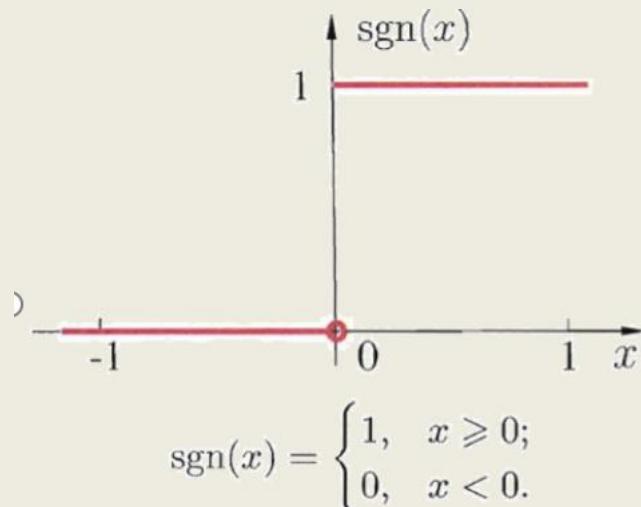


What is Activation Function?

- In a computational network, **an activation function** of a node defines the **output** of that node **given a specific input or set of inputs**.
- Standard computer chip circuits can be seen as digital circuit activation functions that produce either an 'on' (1) or 'off' (0) output based on inputs.
- In **artificial neural networks**, the functionality of activation function is also referred to as the transfer function.

Activation Function

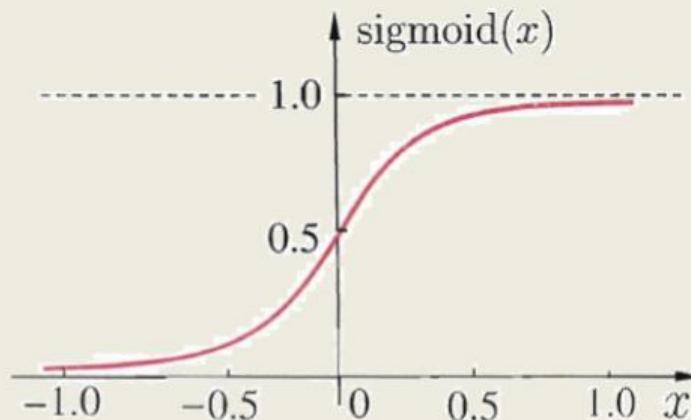
- The ideal activation function, as depicted in Figure (a), is the step function, which maps input values to output values '0' or '1', where '**1**' corresponds to **neuron excitation** and '**0**' corresponds to **neuron inhibition**.
 - However, the step function possesses undesirable properties such as **discontinuity and lack of smoothness**.



(a) step function

Activation Function

- Therefore, the sigmoid function is commonly used as an activation function in practice. A typical sigmoid function, as shown in Figure (b), compresses input values that may vary over a large range into an output value range of (0, 1), hence sometimes referred to as a 'squashing function' (S-shaped function).

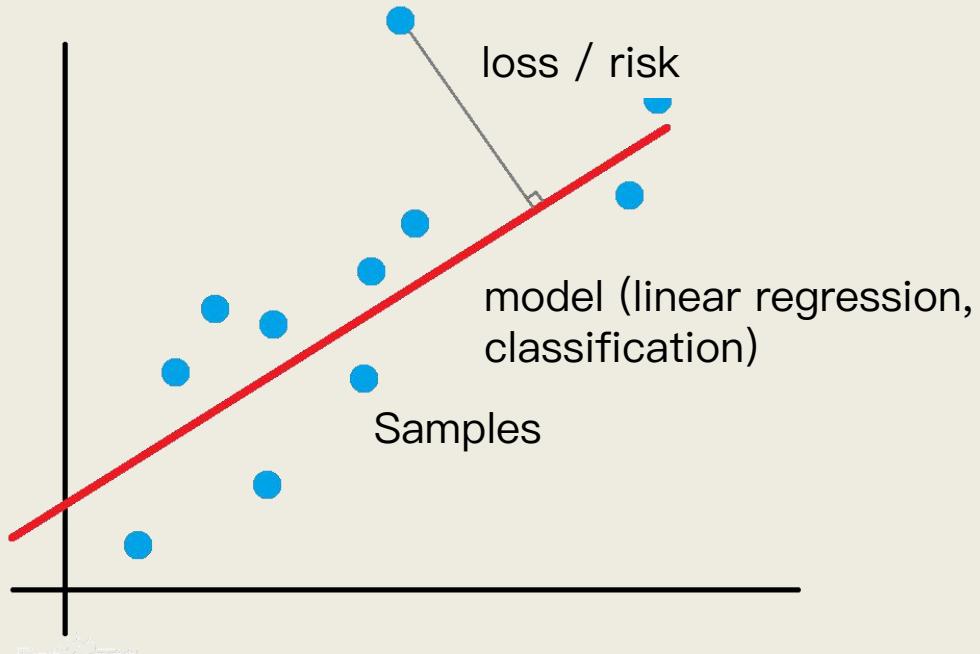


$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

(b) sigmoid function

Loss Function

- The **loss function** or cost function is a function that maps the values of random events or their related random variables to non-negative real numbers, representing the '**risk**' or '**loss**' of that random event



Loss Function – MSE

- The Mean Squared Error (MSE) measures the average squared difference between predicted values and actual observed values. It only considers the average magnitude of errors, without considering their direction. However, because of the squaring operation, predicted values that deviate more from the true values are penalized more severely than those that deviate less.
Additionally, the favorable mathematical properties (easy to get the derivative) of MSE make gradient computation easier

$$MSE = \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}$$

Loss Function – MAE

- The Mean Absolute Error (MAE) measures the average of the absolute differences between predicted values and actual observed values. Similar to MSE, this metric assesses the magnitude of errors without considering their direction.
 - However, unlike MSE, MAE requires more complex tools like linear programming to compute gradients. Additionally, MAE is **more robust to outliers** because it does not involve squaring

$$MAE = \frac{\sum_{i=1}^n | y_i - \hat{y}_i |}{n}$$

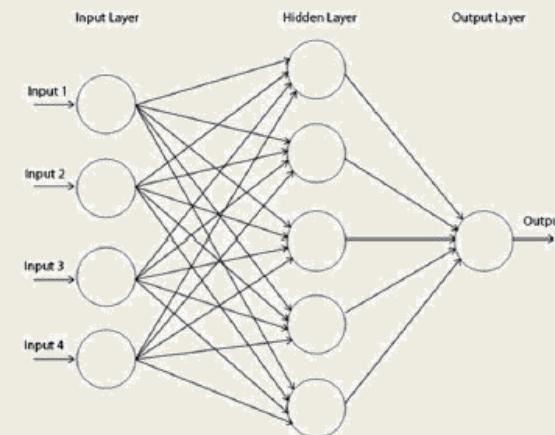
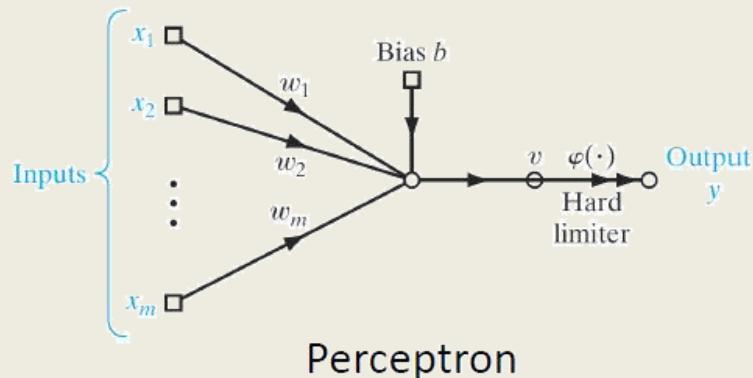
Loss Function – Cross-entropy loss

- **Cross-entropy loss(CE)** is the most common setup in classification problems. As the predicted probabilities deviate from the actual labels, the cross-entropy loss gradually increases.
 - when the actual label is 1 ($y(i)=1$), the latter part of the function disappears, and when the actual label is 0 ($y(i)=0$), the former part of the function disappears.
 - cross-entropy loss heavily penalizes predictions that are confidently wrong.

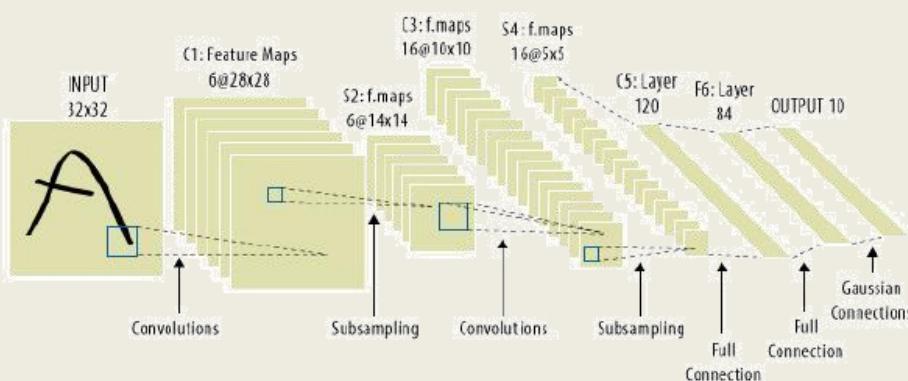
$$CE = -y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i)$$

Deep Learning

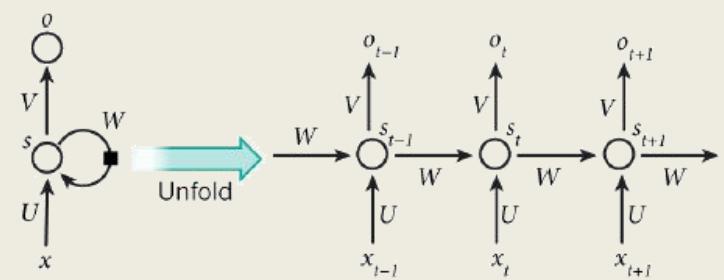
- The artificial neural network is the foundation of deep learning



Multi-layer Perceptron

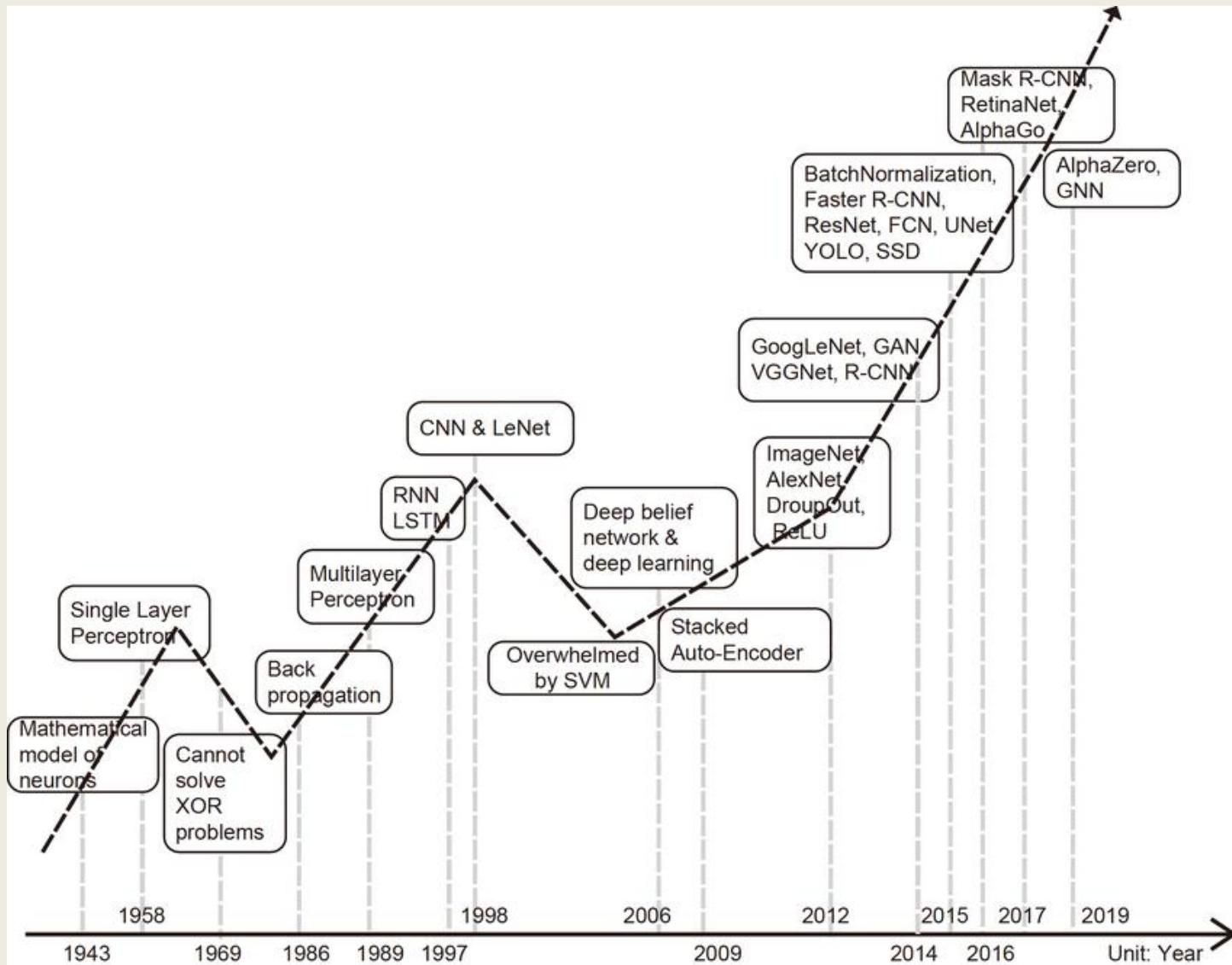


Convolutional Neural Network



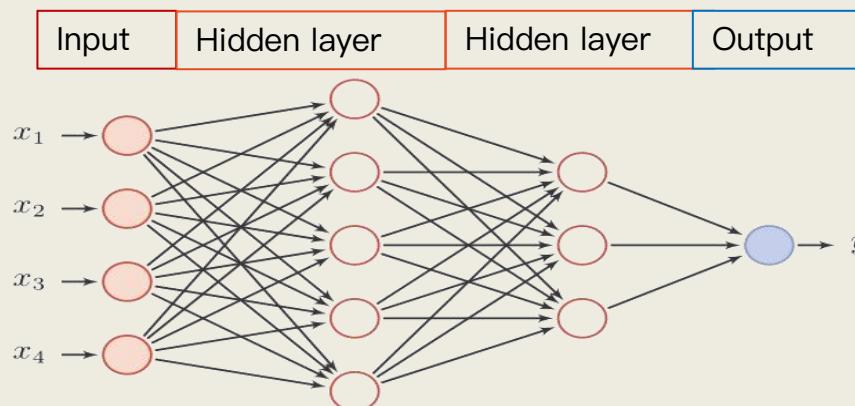
Recurrent Neural Network

The History of Deep Learning



Multi-Layer Perceptron (MLP)

- Multi-layer Perceptron(also known as a **Feedforward neural network**), is a kind of Deep Learning model.
 - Each neuron belongs to a different layer, with no connections within a layer.
 - All neurons between adjacent layers are fully connected.
 - There is no feedback in the entire network; **signals propagate unidirectionally from the input layer to the output layer**, which can be represented by a directed acyclic graph.



Multi-Layer Perceptron (MLP)

- Multi-layer Perceptron propagates information using the following formula

$$\mathbf{z}^{(l)} = W^{(l)} \cdot \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}$$

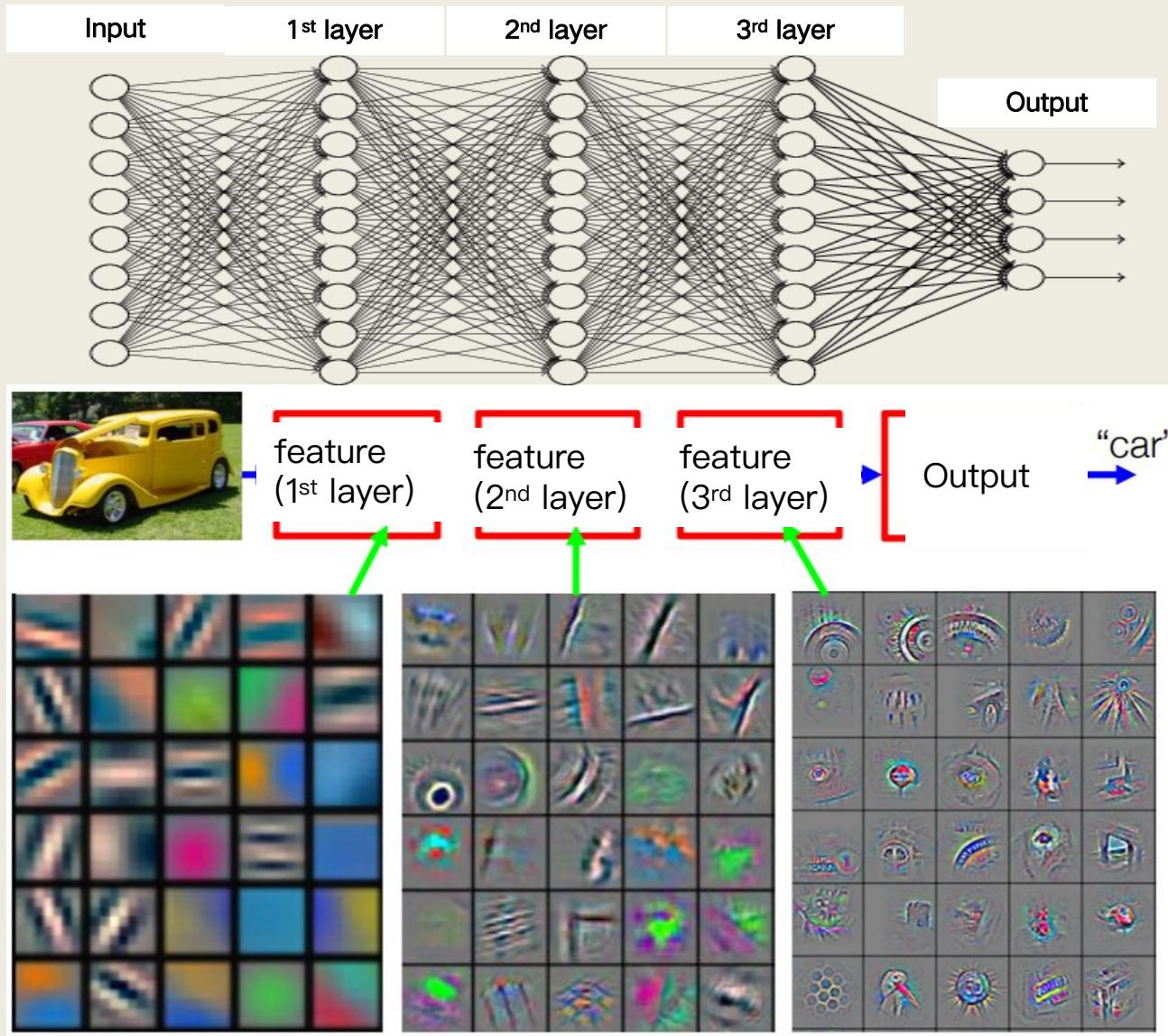
$$\mathbf{a}^{(l)} = f_l(\mathbf{z}^{(l)})$$

activation
function

- Forward propagation (computation)

$$\mathbf{x} = \mathbf{a}^{(0)} \rightarrow \mathbf{z}^{(1)} \rightarrow \mathbf{a}^{(1)} \rightarrow \mathbf{z}^{(2)} \rightarrow \dots \rightarrow \mathbf{a}^{(L-1)} \rightarrow \mathbf{z}^{(L)} \rightarrow \mathbf{a}^{(L)} = f(\mathbf{x}; W, \mathbf{b})$$

An Example



How to Learn?

- To minimize a **loss function** with a regularizer term.

$$\mathcal{R}(W, \mathbf{b}) = \frac{1}{N} \sum_{n=1}^N \mathcal{L}(\mathbf{y}^{(n)}, \hat{\mathbf{y}}^{(n)}) + \frac{1}{2} \lambda \|W\|_F^2$$

Diagram annotations:

- Model parameters**: Points to the term $\frac{1}{2} \lambda \|W\|_F^2$.
- True label**: Points to the term $\mathcal{L}(\mathbf{y}^{(n)}, \hat{\mathbf{y}}^{(n)})$.
- Predicted label**: Points to the term $\hat{\mathbf{y}}^{(n)}$.
- Hyper parameter**: Points to the term λ .

- Update rule for each iteration (l -th) with learning rate α :

$$W^{(l)} \leftarrow W^{(l)} - \alpha \frac{\partial \mathcal{R}(W, \mathbf{b})}{\partial W^{(l)}}$$

$$\mathbf{b}^{(l)} \leftarrow \mathbf{b}^{(l)} - \alpha \frac{\partial \mathcal{R}(W, \mathbf{b})}{\partial \mathbf{b}^{(l)}}$$

How to Learn?

- How to train a classifier?
 - Gradient Decent (GD)
 - Stochastic Gradient Descent (SGD)
 - Backpropagation (BP)

Gradient Decent (GD)

- Gradient Decent (GD)

- Initialize w, b;
- Repeat;
 - Compute the gradient $\partial L / \partial w$;
 - Update parameters $w \leftarrow w - \alpha \partial L / \partial w$;

$$b \leftarrow b - \alpha \partial L / \partial b;$$

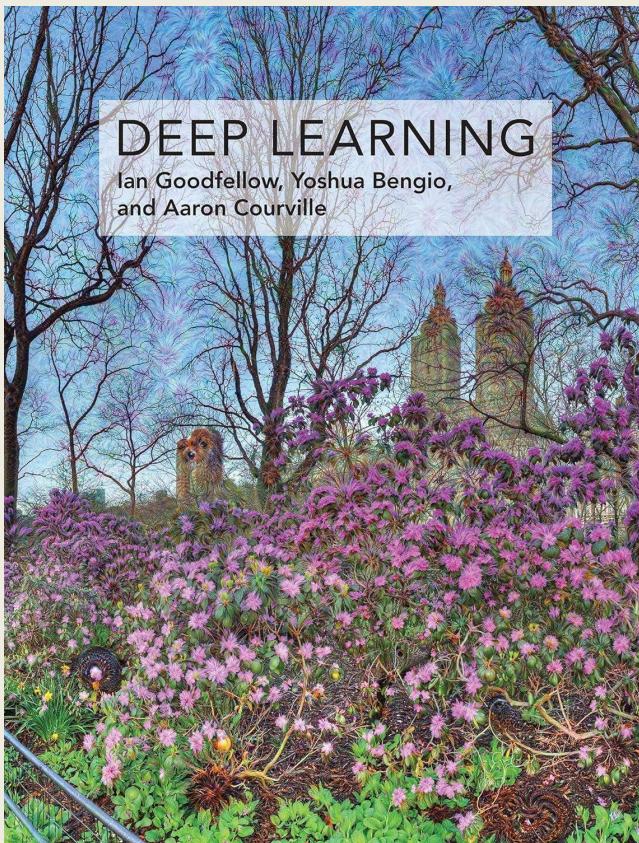
Note: calculate the gradient with the chain rule for more than 2 layers neural network.

$$y = f^5(f^4(f^3(f^2(f^1(x))))) \rightarrow \frac{\partial y}{\partial x} = \frac{\partial f^1}{\partial x} \frac{\partial f^2}{\partial f^1} \frac{\partial f^3}{\partial f^2} \frac{\partial f^4}{\partial f^3} \frac{\partial f^5}{\partial f^4}$$

Summary

- Single Layer Perceptron
 - Activation Function
 - Loss Function
- Multi-Layer Perceptron (MLP)
 - Gradient Decent

References



- <https://www.baeldung.com/cs/sgd-vs-backpropagation>
 - Ian Goodfellow, etl., 'Deep Learning', 2016, MIT Press. <https://www.deeplearningbook.org/>
-