Discussion 11

王欣奕，wangxy6@shanghaitech.edu.cn

Neural Network

Biological inspiration

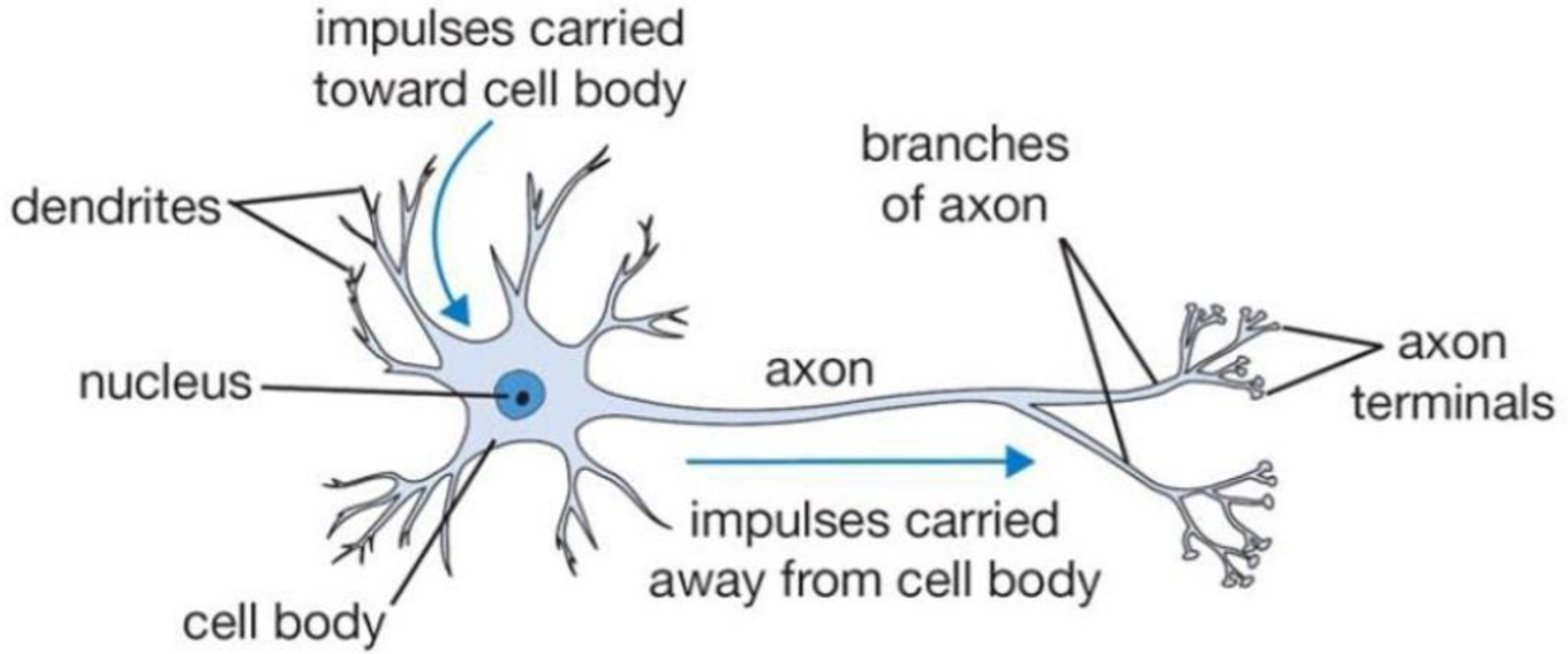
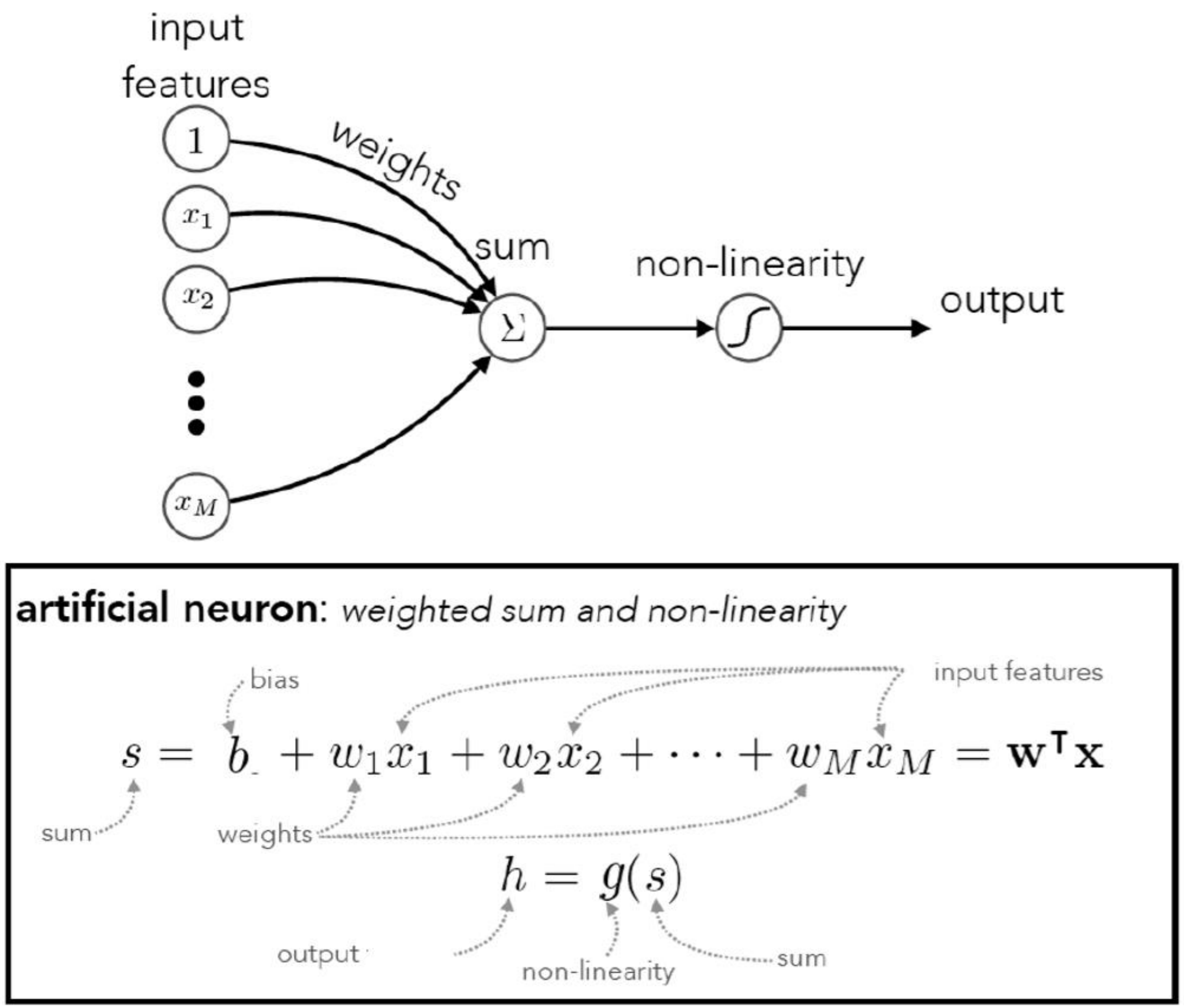


Figure : The basic computational unit of the brain: Neuron

Single neuron



Activation function

Sigmoid: $\sigma(z) = \frac{1}{1+\exp(-z)}$

Tanh: $\tanh(z) = \frac{\exp(z)-\exp(-z)}{\exp(z)+\exp(-z)}$

ReLU (Rectified Linear Unit): $\mathrm{ReLU}(z) = \max(0, z)$

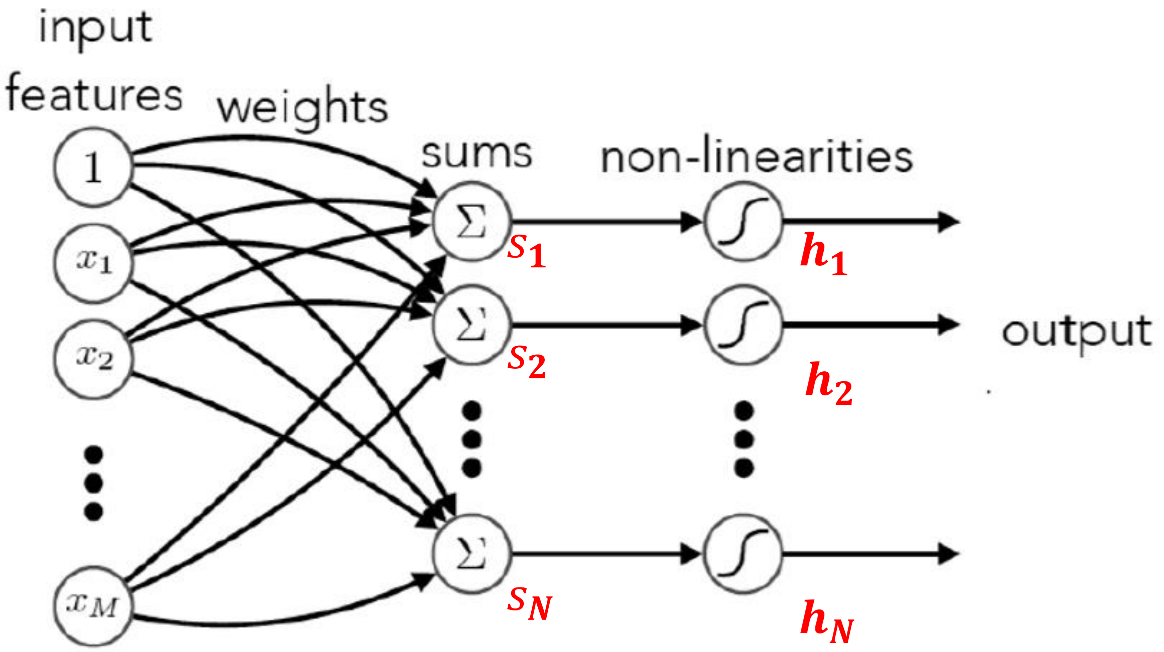
Nice property: $\boldsymbol{\sigma' = \sigma(1 - \sigma)}$

Loss function

MLE : $L(W) = \frac{1}{N} \sum_{i=1}^{N} L_i(f(x_i, W), y_i)$

MAP : $L(W) = \frac{1}{N} \sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$

Single layer network



$s_1 = w_1{}^T x$ $\quad$ $h_1 = \sigma(s_1)$

$s_2 = w_2{}^T x$ $\quad$ $h_2 = \sigma(s_2)$

$\vdots$ $\quad$ $\vdots$

$s_N = w_N{}^T x$ $\quad$ $h_N = \sigma(s_N)$

$\longrightarrow$ $\quad s = W^T x \qquad h = \sigma(s)$

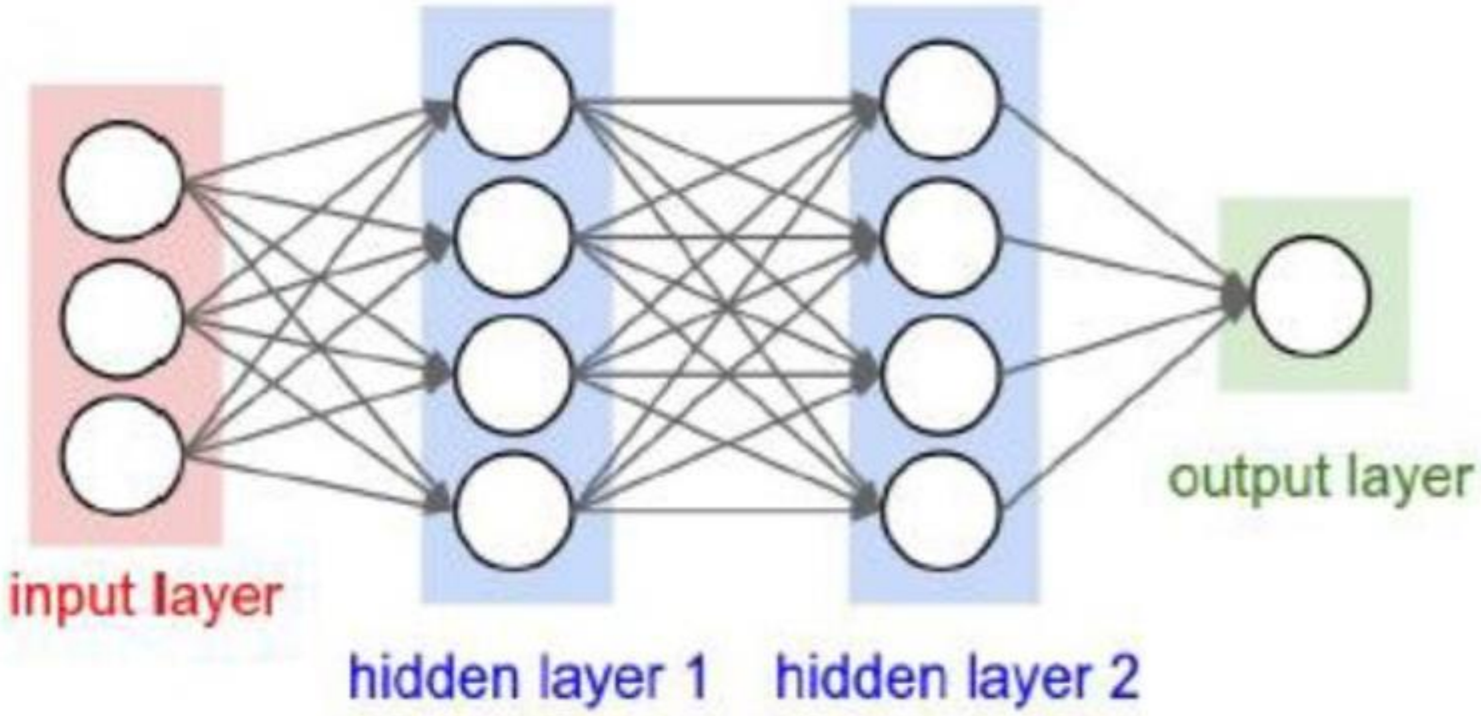
Single layer network

$$\boldsymbol{s} = \boldsymbol{W}^{T}\boldsymbol{x} \qquad \boldsymbol{h} = \boldsymbol{\sigma}(\boldsymbol{s})$$

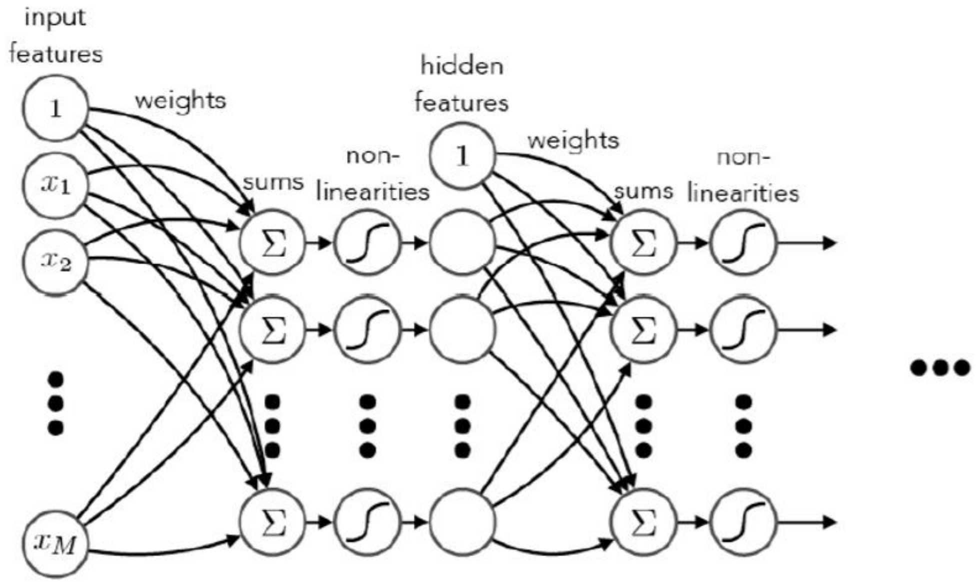
- It can be used in multiclass classification
- h is a N*1 vector which represents class scores
- Can also apply softmax function to get normalized probabilities

Multi-layer network

- A 3-layer neural network



Multi-layer network



network: *sequence of parallelized weighted sums and non-linearities*

DEFINE $\mathbf{x}^{(0)} \equiv \mathbf{x},\ \mathbf{x}^{(1)} \equiv \mathbf{h},$ ETC.

1st layer

$$\mathbf{s}^{(1)} = \mathbf{W}^{(1)\intercal}\mathbf{x}^{(0)}$$

$$\mathbf{x}^{(1)} = \sigma(\mathbf{s}^{(1)})$$

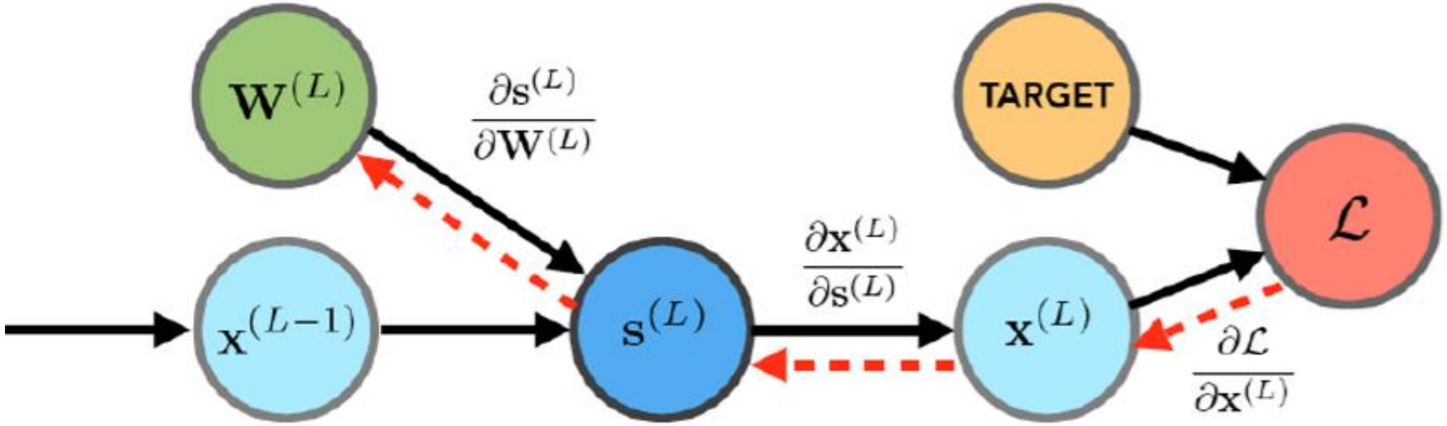
2nd layer

$$\mathbf{s}^{(2)} = \mathbf{W}^{(2)\intercal}\mathbf{x}^{(1)}$$

$$\mathbf{x}^{(2)} = \sigma(\mathbf{s}^{(2)})$$

•••

Multi-layer network(Back-propagation)



$$L = \frac{1}{2}(x^{(L)} - t)^2$$

$$x^{(L)} = \sigma(s^{(L)})$$

$$s^{(L)} = (W^{(L)})^T x^{(L-1)}$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}^{(L)}} = \frac{\partial \mathcal{L}}{\partial \mathbf{x}^{(L)}} \frac{\partial \mathbf{x}^{(L)}}{\partial \mathbf{s}^{(L)}} \frac{\partial \mathbf{s}^{(L)}}{\partial \mathbf{W}^{(L)}}$$

$$x^{(L)} - t$$

$$(x^{(L-1)})^T$$

$$\sigma(s^{(L)})\left(1 - \sigma(s^{(L)})\right) = x^{(L)}(1 - x^{(L)})$$

Multi-layer network(Gradient-based learning)

- **Model initialization**
- **Forward propagate**
- **Loss function**
- **Differentiation**
- **Back-propagation**
- **Weight update:** $\boldsymbol{W}^{(\boldsymbol{L})} = \boldsymbol{W}^{(\boldsymbol{L})} - \eta \frac{\partial \boldsymbol{Loss}}{\partial \boldsymbol{W}^{(\boldsymbol{L})}}$

Batch GD; Mini-batch GD; SGD