

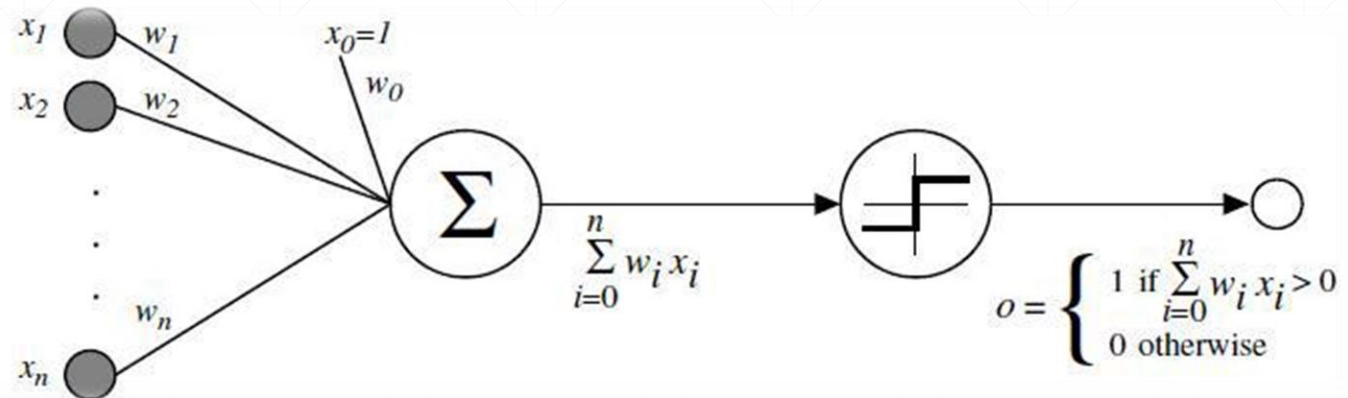


感知机

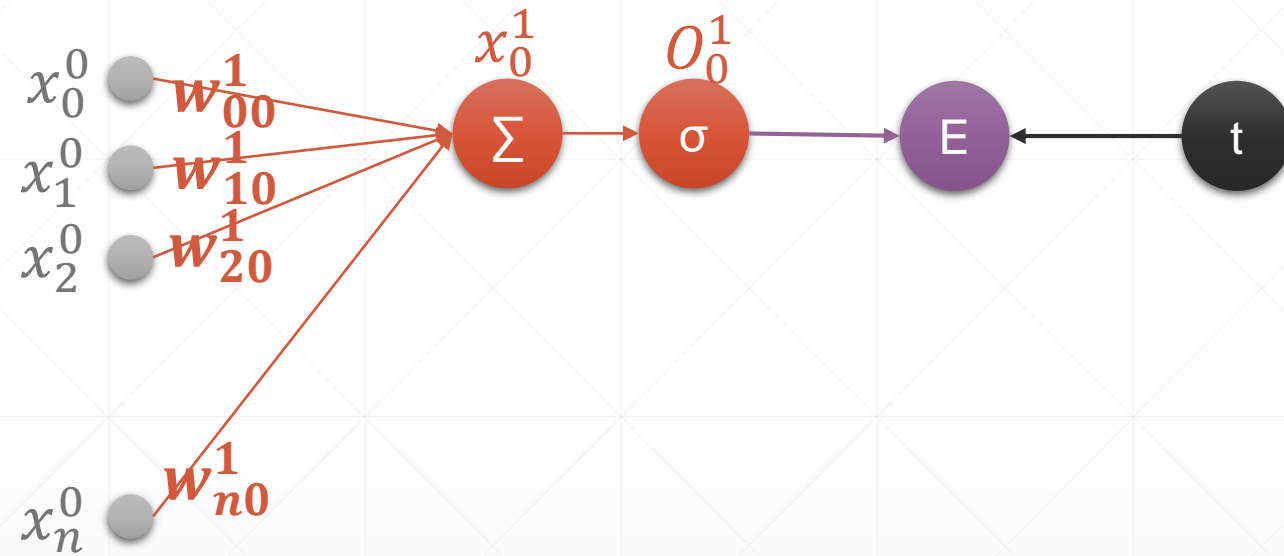
主讲人：龙良曲

recap

- $y = XW + b$
- $y = \sum x_i * w_i + b$



Perceptron



Derivative

$$E = \frac{1}{2} (O_0^1 - t)^2$$

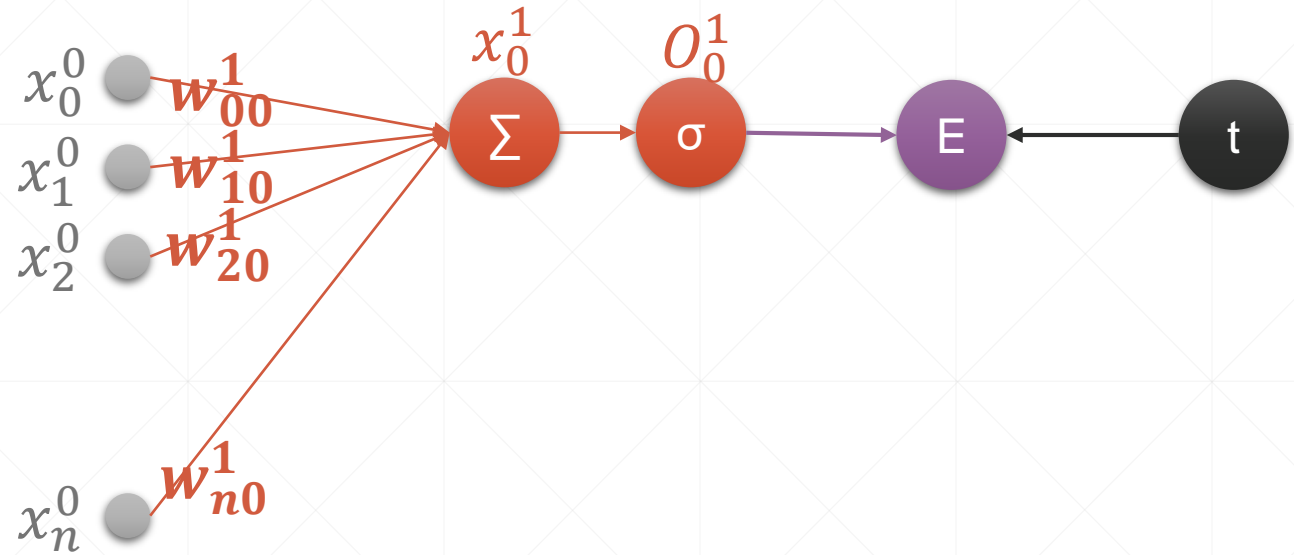
$$\frac{\partial E}{\partial w_{j0}} = (O_0^1 - t) \frac{\partial O_0^1}{\partial w_{j0}}$$

$$\frac{\partial E}{\partial w_{j0}} = (O_0^1 - t) \frac{\partial \sigma(x_0)}{\partial w_{j0}}$$

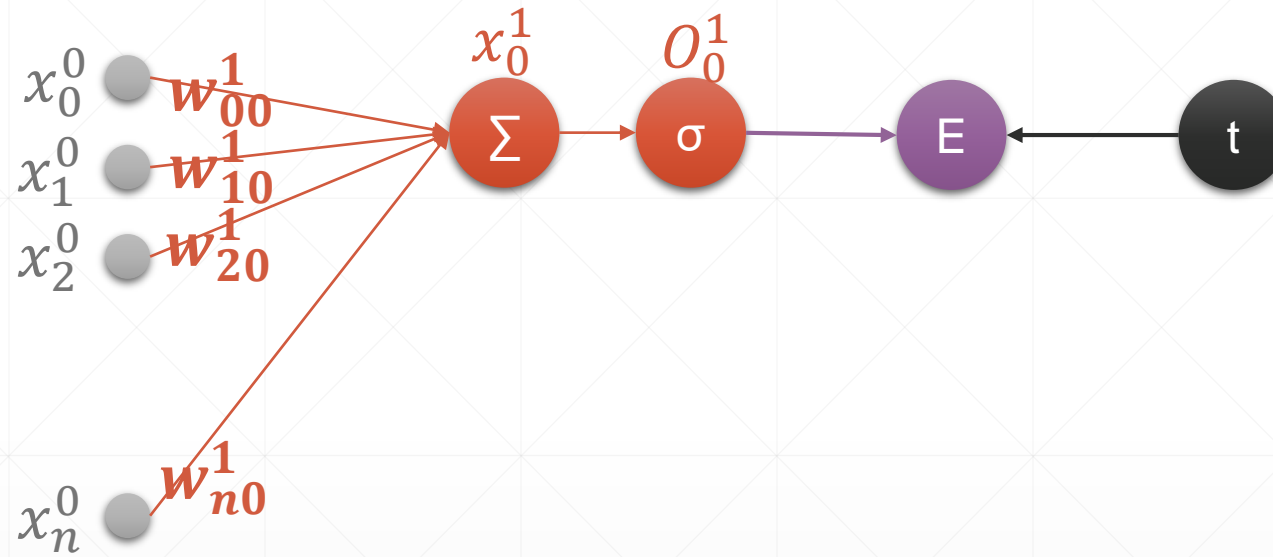
$$\frac{\partial E}{\partial w_{j0}} = (O_0^1 - t) \sigma(x_0)(1 - \sigma(x_0)) \frac{\partial x_0^1}{\partial w_{j0}}$$

$$\frac{\partial E}{\partial w_{j0}} = (O_0^1 - t) O_0^1 (1 - O_0^1) \frac{\partial x_0^1}{\partial w_{j0}}$$


$$\frac{\partial E}{\partial w_{j0}} = (O_0^1 - t) O_0^1 (1 - O_0^1) x_j^0$$



Perceptron



$$\frac{\partial E}{\partial w_{j0}} = (O_0 - t) O_0 (1 - O_0) x_j^0$$



```
In [41]: x=torch.randn(1,10)
```

```
In [48]: w=torch.randn(1,10,requires_grad=True)
```

```
In [49]: o=torch.sigmoid(x@w.t())
```

```
In [50]: o.shape
```

```
Out[50]: torch.Size([1, 1])
```

```
In [51]: loss=F.mse_loss(torch.ones(1,1),o)
```

```
In [52]: loss.shape
```

```
Out[52]: torch.Size([])
```

```
In [53]: loss.backward()
```

```
In [54]: w.grad
```

```
Out[54]:
```

```
tensor([[-0.0107, -0.0021,  0.0047,  0.0092, -0.0091, -0.0030,  0.0069, -0.0105,  
         -0.0061, -0.0051]])
```

下一课时

MLP及梯度

Thank You.

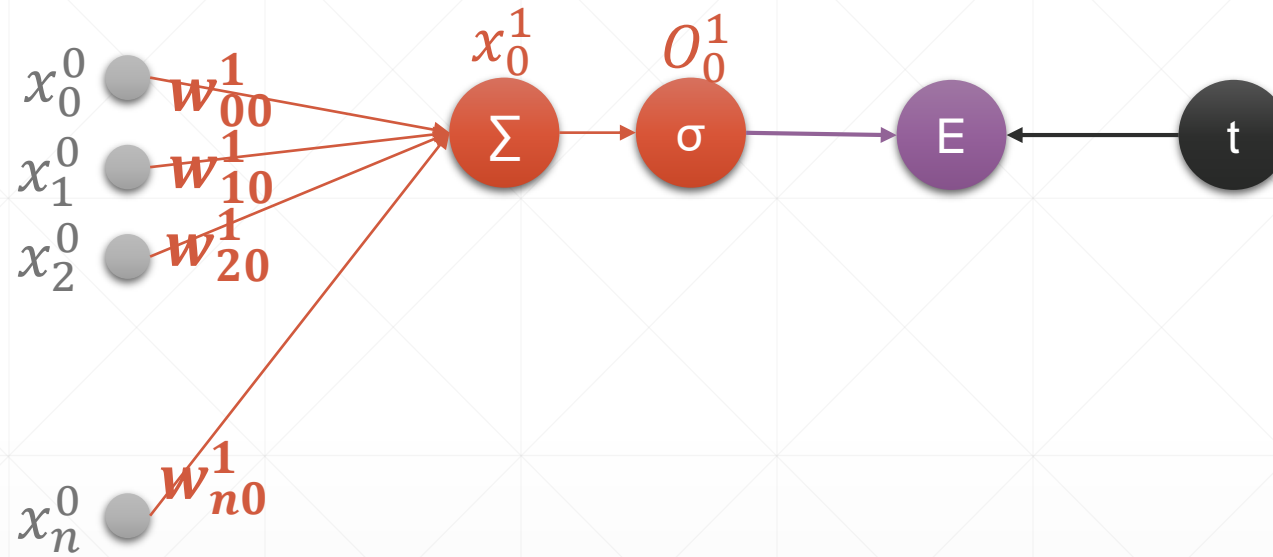


PyTorch

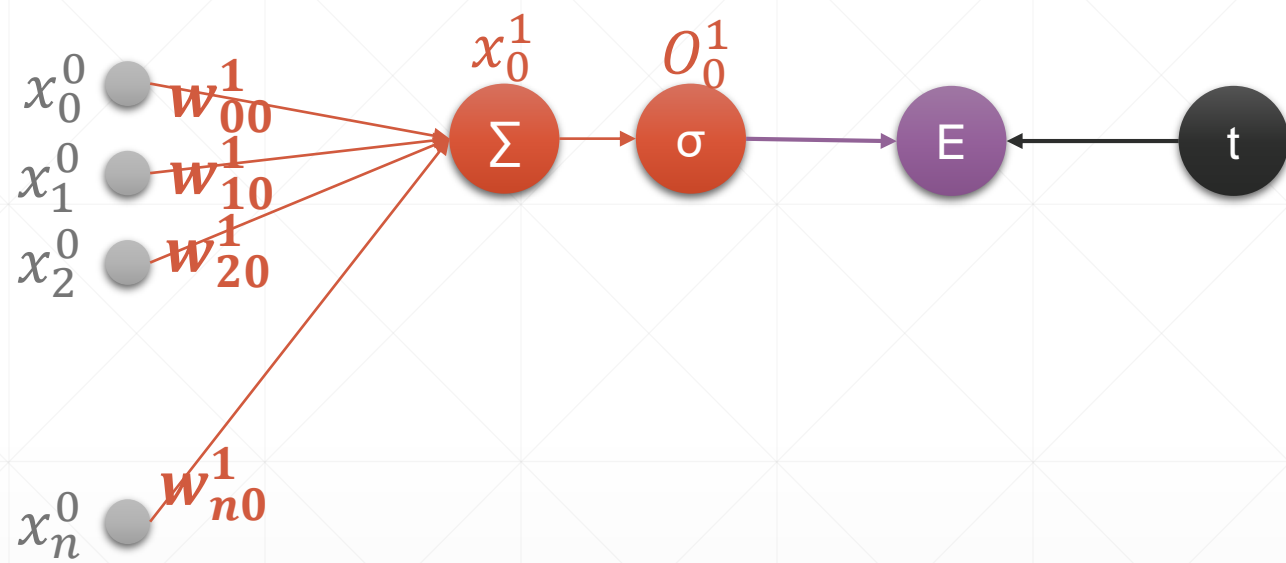
感知机-2

主讲人：龙良曲

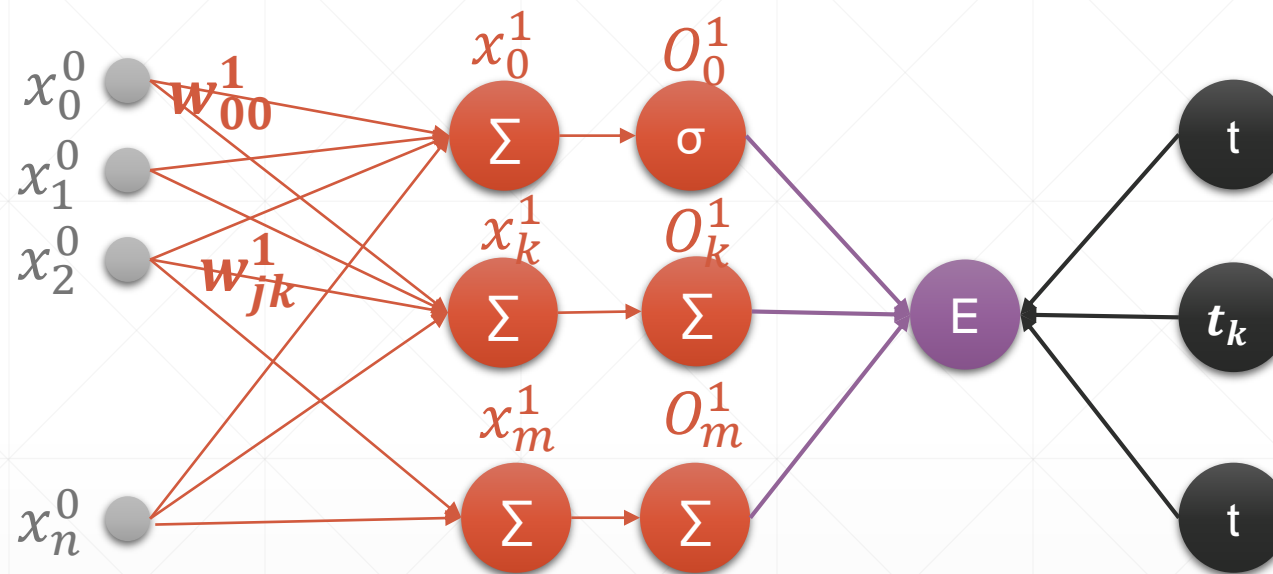
Perceptron



$$\frac{\partial E}{\partial w_{j0}} = (O_0 - t) O_0 (1 - O_0) x_j^0$$



Multi-output Perceptron



Derivative

$$E = \frac{1}{2} \sum (O_i^1 - t_i)^2$$

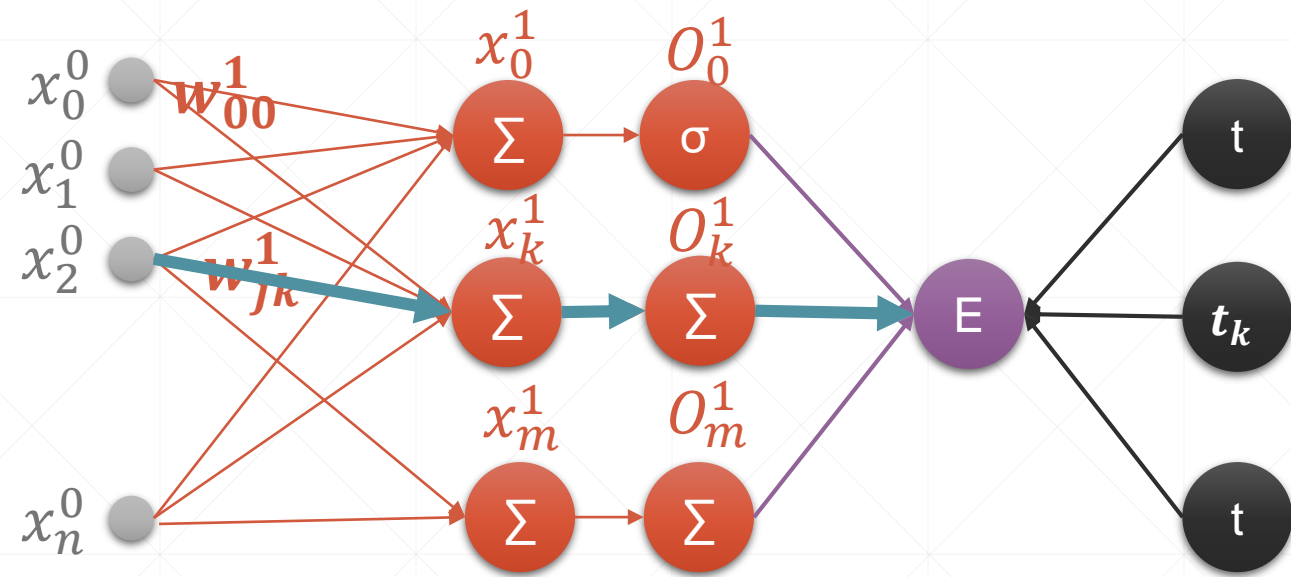
$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) \frac{\partial O_k}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) \frac{\partial \sigma(x_k)}{\partial w_{jk}}$$

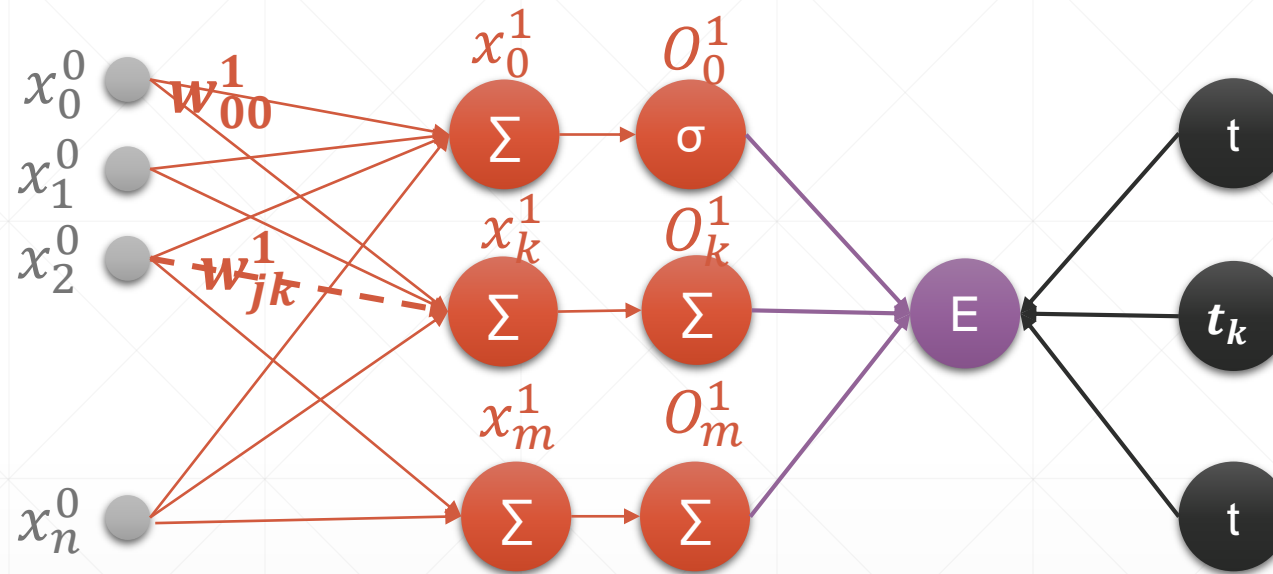
$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) \sigma(x_k)(1 - \sigma(x_k)) \frac{\partial x_k^1}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) O_k (1 - O_k) \frac{\partial x_k^1}{\partial w_{jk}}$$

$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) O_k (1 - O_k) x_j^0$$



Multi-output Perceptron



$$\frac{\partial E}{\partial w_{jk}} = (O_k - t_k) O_k (1 - O_k) x_j^0$$

In [55]: `x=torch.randn(1,10)`

In [56]: `w=torch.randn(2,10,requires_grad=True)`

In [57]: `o=torch.sigmoid(x@w.t())`

In [58]: `o.shape`

Out[58]: `torch.Size([1, 2])`

In [59]: `loss=F.mse_loss(torch.ones(1,1),o)`

In [60]: `loss`

Out[60]: `tensor(0.2443, grad_fn=<MeanBackward1>)`

In [61]: `loss.backward()`

In [62]: `w.grad`

Out[62]:

`tensor([[-0.0040, 0.0004, 0.0041, -0.0088, 0.0017, -0.0103, 0.0067, -0.0005,
 -0.0035, -0.0043],
 [-0.1182, 0.0114, 0.1221, -0.2628, 0.0495, -0.3060, 0.2002, -0.0149,
 -0.1027, -0.1266]])`

下一课时

链式法则

Thank You.
