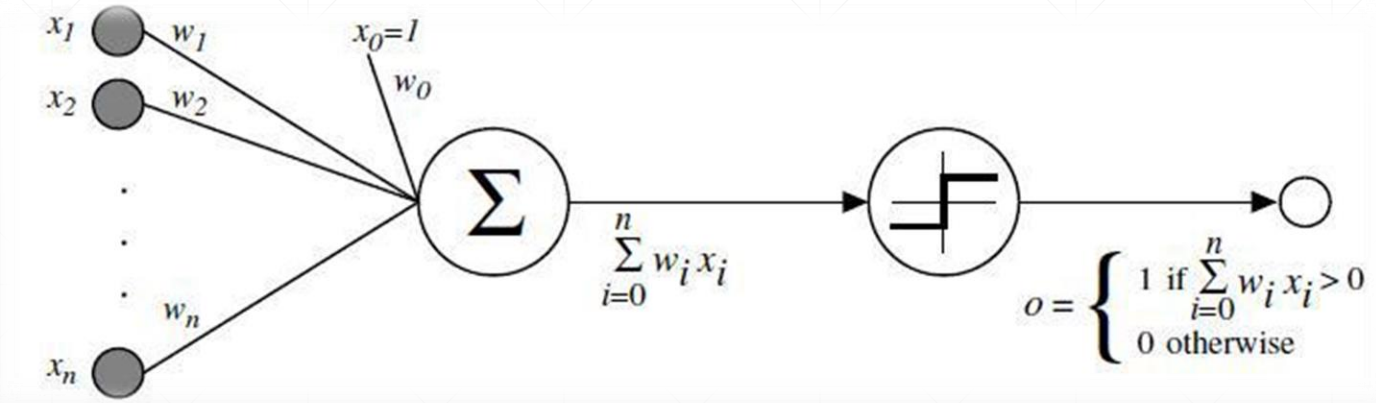
PyTorch

感知机

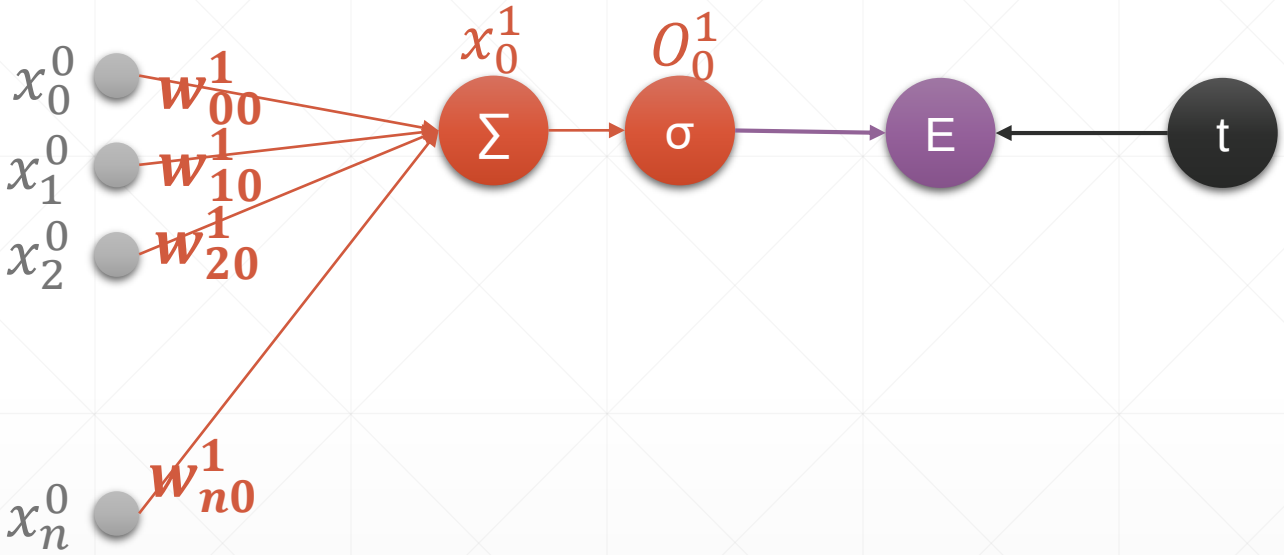
主讲人：龙良曲

recap

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



Perceptron



Derivative

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

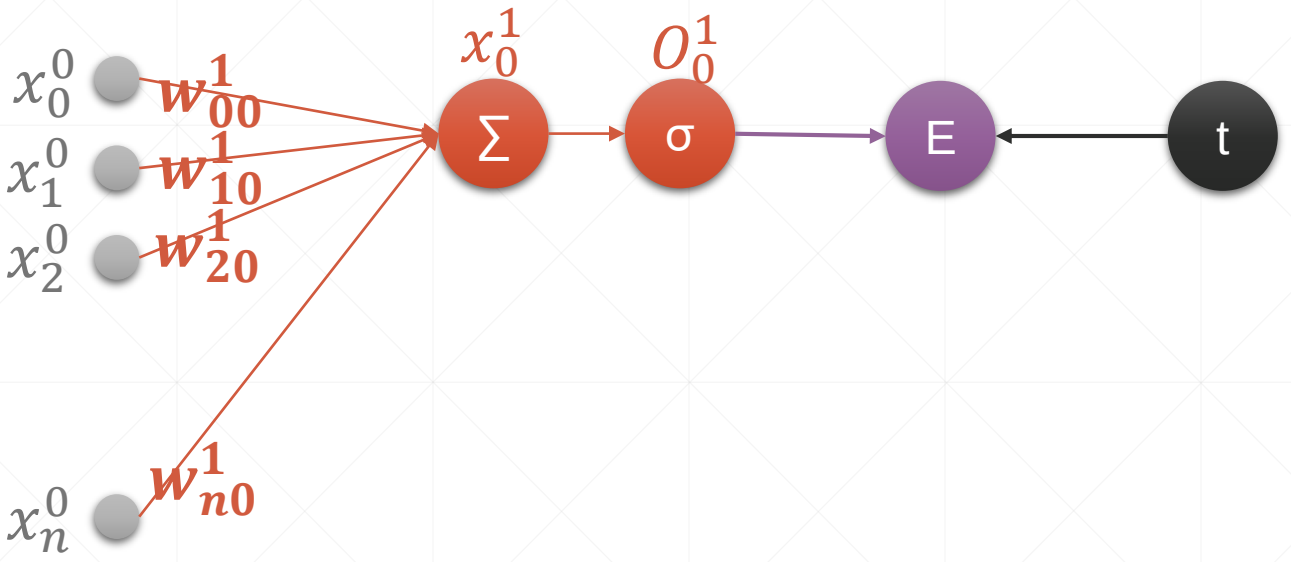
$$\frac{\partial E}{\partial w_{j0}} = (O_0 - t)\frac{\partial O_0}{\partial w_{j0}}$$

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

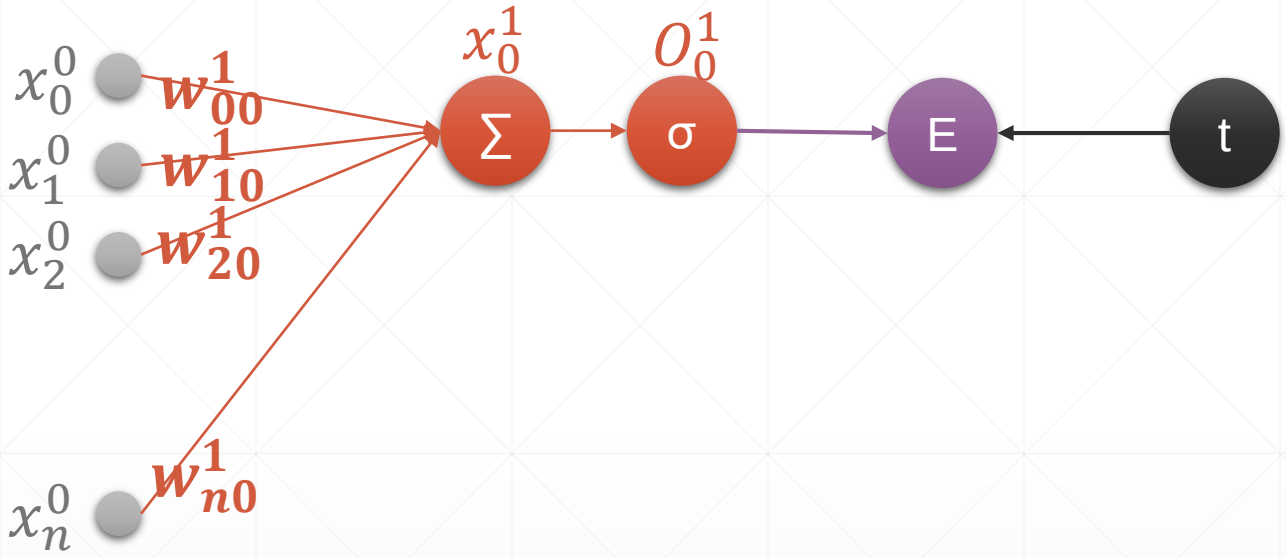
$$\frac{\partial E}{\partial w_{j0}} = (O_0 - t)\,\sigma(x_0)(1 - \sigma(x_0))\frac{\partial x_0^1}{\partial w_{j0}}$$

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

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



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.