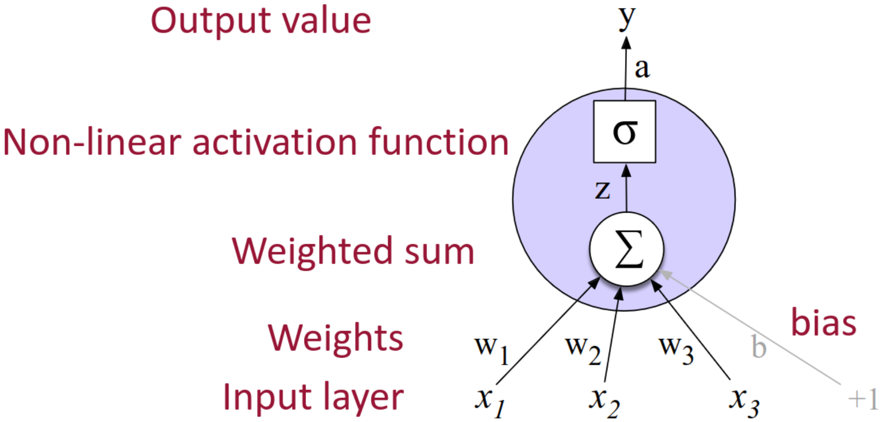
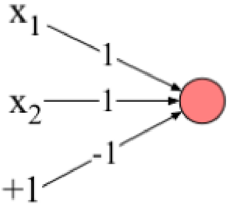
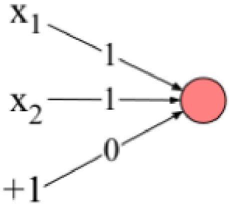
Neural network unit



This & many figures below are from
Speech & Language Processing, Jurafsky & Martin, 3rd ed (chap 7)

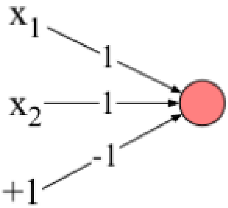


x_1 AND x_2

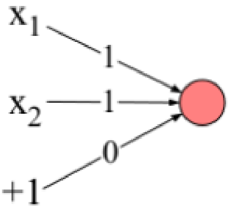


x_1 OR x_2

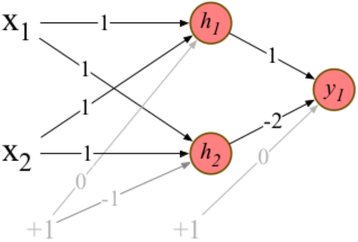
$$\text{step}(z) \ := \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases}$$



x_1 AND x_2



x_1 OR x_2

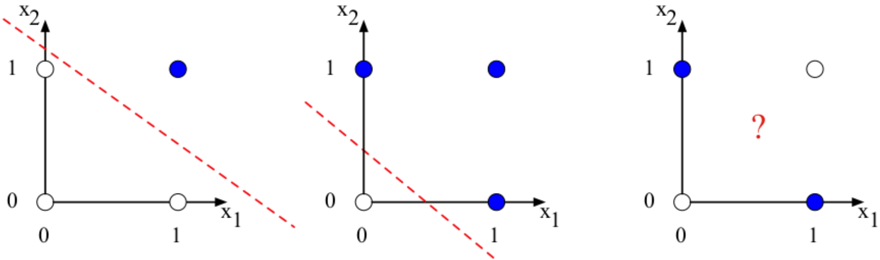


XOR $=$ OR_{h_1} but not AND_{h_2}

ReLU in h_1, h_2, y_1

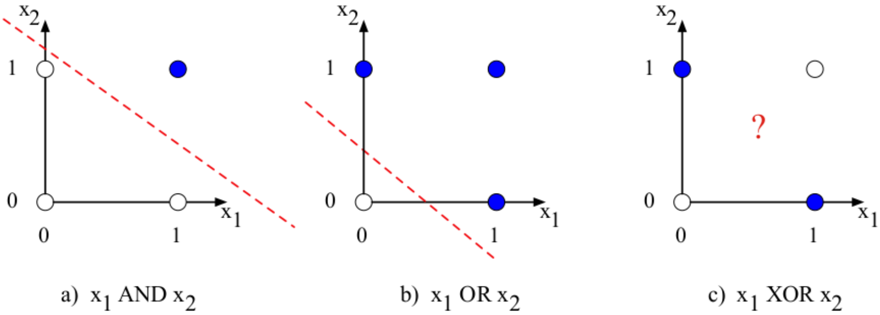
$$\text{step}(z) := \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{else} \end{cases}$$

$$\text{ReLU}(z) := \begin{cases} z & \text{if } z \geq 0 \\ 0 & \text{else} \end{cases}$$

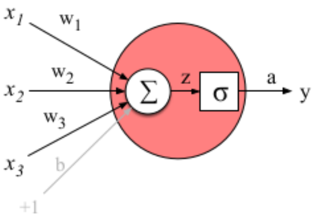


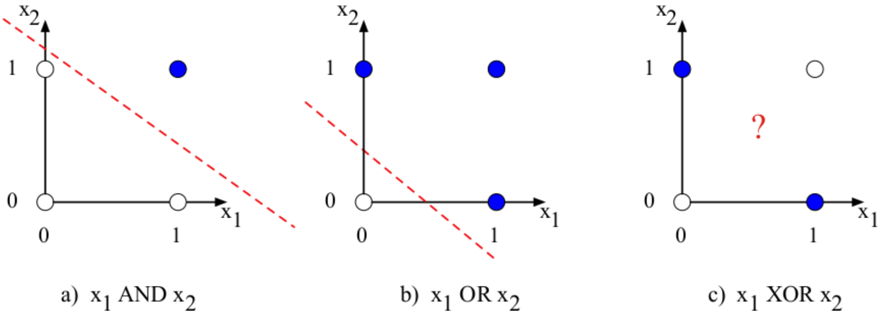
a) x_1 AND x_2 b) x_1 OR x_2 c) x_1 XOR x_2

Linear (In)separability

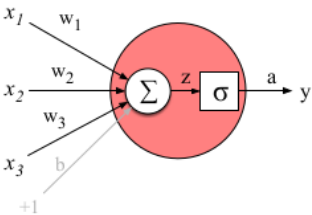


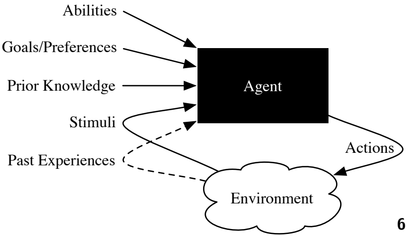
Linear (In)separability





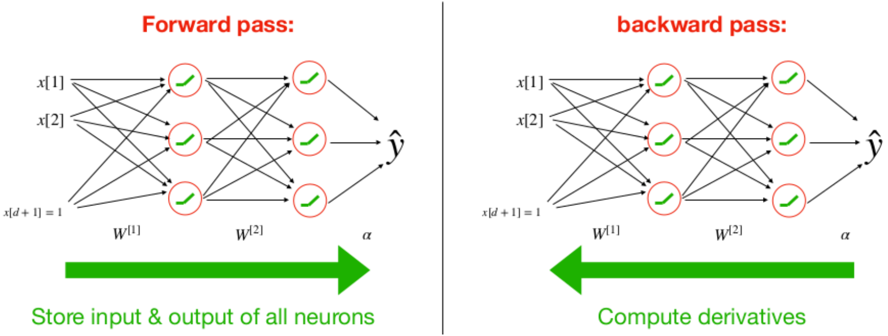
Linear (In)separability





Overview of backpropagation

Forward pass followed by a backward pass

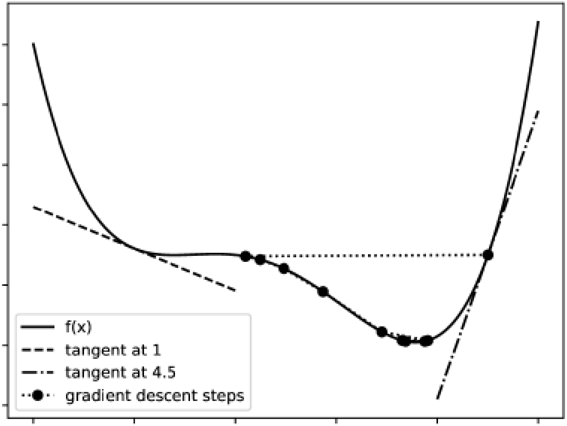


After a forward pass, propagate error measures backwards, starting with neurons directly connected to the outputs, and ending with weights on inputs.

Gradient descent

To minimize error, update weight w to

$$w - \alpha \frac{\partial \text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$



Poole & Mackworth 2023, Figure 4.13

Gradient descent

To minimize error, update weight w to

$$w - \alpha \frac{\partial \text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$

where for example,

$$\text{error} = \sum_i (t_i - y_i)^2 \qquad (\text{output } y_1 \ldots y_n,\ \text{target } t_1 \ldots t_n)$$

$$\frac{\partial \text{error}}{\partial y_j} = \sum_i \frac{\partial (t_i - y_i)^2}{\partial y_j}$$

Gradient descent & chain rule $\frac{df(g(x))}{dx} = \frac{df(g(x))}{dg(x)} \frac{dg(x)}{dx}$

To minimize error, update weight w to

$$w - \alpha \frac{\partial \text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$

where for example,

$$\text{error} = \sum_i (t_i - y_i)^2 \qquad (\text{output } y_1 \ldots y_n, \ \text{target } t_1 \ldots t_n)$$

$$\frac{\partial \text{error}}{\partial y_j} = \sum_i \frac{\partial (t_i - y_i)^2}{\partial y_j} = \sum_i \frac{\partial (t_i - y_i)^2}{\partial (t_i - y_i)} \frac{\partial (t_i - y_i)}{\partial y_j}$$

$$= 2(t_j - y_j)(-1) = 2(y_j - t_j)$$

Gradient descent & chain rule $\frac{df(g(x))}{dx} = \frac{df(g(x))}{dg(x)}\frac{dg(x)}{dx}$

To minimize error, update weight w to

$$w - \alpha\frac{\partial\text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$

where for example,

$$\text{error} \;=\; \sum_i (t_i - y_i)^2 \qquad (\text{output } y_1 \ldots y_n,\;\; \text{target } t_1 \ldots t_n)$$

$$\frac{\partial\text{error}}{\partial y_j} \;=\; \sum_i \frac{\partial(t_i - y_i)^2}{\partial y_j} \;=\; \sum_i \frac{\partial(t_i - y_i)^2}{\partial(t_i - y_i)}\frac{\partial(t_i - y_i)}{\partial y_j}$$

$$=\; 2(t_j - y_j)(-1) \;=\; 2(y_j - t_j)$$

and for $y_j = \sigma(z_j)$,

$$\frac{\partial\text{error}}{\partial z_j} \;=\; \frac{\partial y_j}{\partial z_j}\frac{\partial\text{error}}{\partial y_j} \qquad (\dagger)$$

Gradient descent & chain rule $\frac{df(g(x))}{dx} = \frac{df(g(x))}{dg(x)} \frac{dg(x)}{dx}$

To minimize error, update weight w to

$$w - \alpha \frac{\partial \text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$

where for example,

$$\text{error} = \sum_i (t_i - y_i)^2 \qquad (\text{output } y_1 \ldots y_n, \text{ target } t_1 \ldots t_n)$$

$$\frac{\partial \text{error}}{\partial y_j} = \sum_i \frac{\partial (t_i - y_i)^2}{\partial y_j} = \sum_i \frac{\partial (t_i - y_i)^2}{\partial (t_i - y_i)} \frac{\partial (t_i - y_i)}{\partial y_j}$$

$$= 2(t_j - y_j)(-1) = 2(y_j - t_j)$$

and for $y_j = \sigma(z_j)$,

$$\frac{\partial \text{error}}{\partial z_j} = \frac{\partial y_j}{\partial z_j} \frac{\partial \text{error}}{\partial y_j} = \sigma'(z_j) \frac{\partial \text{error}}{\partial y_j} \qquad (\dagger)$$

Gradient descent & chain rule $\frac{df(g(x))}{dx} = \frac{df(g(x))}{dg(x)} \frac{dg(x)}{dx}$

To minimize error, update weight w to

$$w - \alpha \frac{\partial \text{error}}{\partial w} \qquad \text{step size } \alpha \text{ (learning rate)}$$

where for example,

$$\text{error} \;=\; \sum_i (t_i - y_i)^2 \qquad (\text{output } y_1 \ldots y_n,\ \ \text{target } t_1 \ldots t_n)$$

$$\frac{\partial \text{error}}{\partial y_j} \;=\; \sum_i \frac{\partial (t_i - y_i)^2}{\partial y_j} \;=\; \sum_i \frac{\partial (t_i - y_i)^2}{\partial (t_i - y_i)} \frac{\partial (t_i - y_i)}{\partial y_j}$$

$$=\; 2(t_j - y_j)(-1) \;=\; 2(y_j - t_j)$$

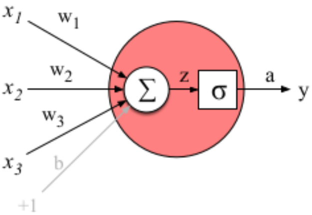
and for $y_j = \sigma(z_j)$,

$$\frac{\partial \text{error}}{\partial z_j} \;=\; \frac{\partial y_j}{\partial z_j} \frac{\partial \text{error}}{\partial y_j} \;=\; \sigma'(z_j)\, \frac{\partial \text{error}}{\partial y_j} \qquad (\dagger)$$

where for $\sigma(z) = \text{ReLU}(z) := \max(z, 0)$,

$$\sigma'(z) \;=\; \begin{cases} 1 & \text{if } z \geq 0 \\ 0 & \text{otherwise} \end{cases} \qquad (\text{contra step}'(z) = 0)$$

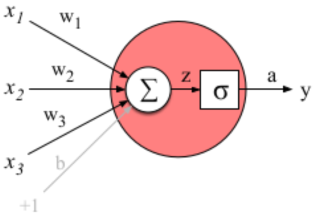
Backpropagation



w_{ij} connected to output via $z_j = \sum_i x_i w_{ij} \quad (i \to j)$

$$\frac{\partial \text{error}}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial \text{error}}{\partial z_j}$$

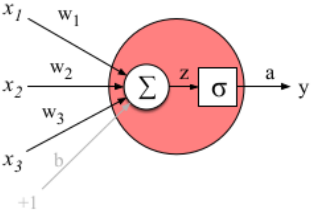
Backpropagation



w_{ij} connected to output via $z_j = \sum_i x_i w_{ij} \quad (i \to j)$

$$\frac{\partial \text{error}}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial \text{error}}{\partial z_j} = x_i \frac{\partial \text{error}}{\partial z_j}$$

Backpropagation



$$y = \sigma(z)$$

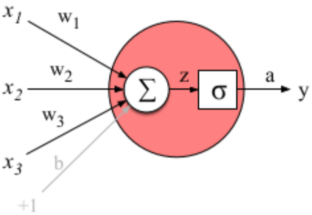
$$\frac{\partial \text{error}}{\partial z} = \sigma'(z) \frac{\partial \text{error}}{\partial y} \qquad (\dagger)$$

w_{ij} connected to output via $z_j = \sum_i x_i w_{ij} \quad (i \to j)$

$$\frac{\partial \text{error}}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial \text{error}}{\partial z_j} = x_i \frac{\partial \text{error}}{\partial z_j}$$

$$= x_i \ \sigma'(z_j) \frac{\partial \text{error}}{\partial y_j} \qquad \text{for } y_j = \sigma(z_j) \text{ and by } (\dagger)$$

Backpropagation



$$y = \sigma(z)$$

$$\frac{\partial \text{error}}{\partial z} = \sigma'(z) \frac{\partial \text{error}}{\partial y} \qquad (\dagger)$$

w_{ij} connected to output via $z_j = \sum_i x_i w_{ij} \quad (i \to j)$

$$\frac{\partial \text{error}}{\partial w_{ij}} = \frac{\partial z_j}{\partial w_{ij}} \frac{\partial \text{error}}{\partial z_j} = x_i \frac{\partial \text{error}}{\partial z_j}$$

$$= x_i \; \sigma'(z_j) \frac{\partial \text{error}}{\partial y_j} \qquad \text{for } y_j = \sigma(z_j) \text{ and by } (\dagger)$$

$\frac{\partial \text{error}}{\partial y_j}$ is directly calculable for y_j at output layer

$$\text{e.g.,} \quad \frac{\partial \sum_i (t_i - y_i)^2}{\partial y_j} = 2(y_j - t_j)$$

Otherwise, y_j feeds forward to a layer

$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

Otherwise, y_j feeds forward to a layer

$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

and by the multivariable chain rule,[1]

$$\frac{\partial \text{error}}{\partial y_j} \;=\; \sum_k \frac{\partial z_k}{\partial y_j}\frac{\partial \text{error}}{\partial z_k}$$

[1] If $y = f(u_1, \ldots, u_n)$ where $u_i = u_i(x_1, \ldots, x_k)$ for $1 \leq i \leq n$, then

$$\frac{\partial y}{\partial x_j} \;=\; \sum_{i=1}^{n} \frac{\partial y}{\partial u_i}\frac{\partial u_i}{\partial x_j} \quad \text{for } 1 \leq j \leq k.$$

Otherwise, y_j feeds forward to a layer

$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

and by the multivariable chain rule,

$$\begin{aligned} \frac{\partial \text{error}}{\partial y_j} &= \sum_k \frac{\partial z_k}{\partial y_j} \frac{\partial \text{error}}{\partial z_k} \\ &= \sum_k w_{jk} \frac{\partial \text{error}}{\partial z_k} \qquad\qquad \text{as } z_k = \sum_j y_j w_{jk} \end{aligned}$$

Otherwise, y_j feeds forward to a layer

$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

and by the multivariable chain rule,

$$\begin{aligned}
\frac{\partial \text{error}}{\partial y_j} &= \sum_k \frac{\partial z_k}{\partial y_j} \frac{\partial \text{error}}{\partial z_k} \\
&= \sum_k w_{jk} \frac{\partial \text{error}}{\partial z_k} && \text{as } z_k = \sum_j y_j w_{jk} \\
&= \sum_k w_{jk} \, \sigma'(z_k) \, \frac{\partial \text{error}}{\partial y_k} && \text{for } y_k = \sigma(z_k) \text{ and by } (\dagger)
\end{aligned}$$

Otherwise, y_j feeds forward to a layer

$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

and by the multivariable chain rule,

$$\begin{aligned}
\frac{\partial \text{error}}{\partial y_j} &= \sum_k \frac{\partial z_k}{\partial y_j} \frac{\partial \text{error}}{\partial z_k} \\
&= \sum_k w_{jk} \frac{\partial \text{error}}{\partial z_k} && \text{as } z_k = \sum_j y_j w_{jk} \\
&= \sum_k w_{jk}\ \sigma'(z_k)\ \frac{\partial \text{error}}{\partial y_k} && \text{for } y_k = \sigma(z_k) \text{ and by } (\dagger)
\end{aligned}$$

propagating the error measure

$$\frac{\partial \text{error}}{\partial y_k}$$

Otherwise, y_j feeds forward to a layer

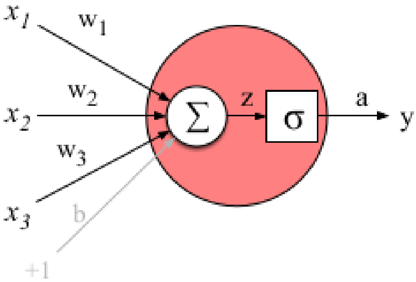
$$z_k = \sum_j y_j w_{jk} \qquad \text{(with possibly more than one } k\text{)}$$

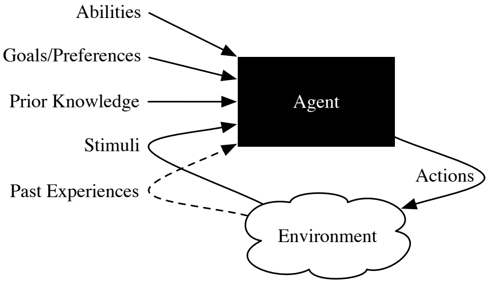
and by the multivariable chain rule,

$$\begin{aligned}
\frac{\partial \text{error}}{\partial y_j} &= \sum_k \frac{\partial z_k}{\partial y_j} \frac{\partial \text{error}}{\partial z_k} \\
&= \sum_k w_{jk} \frac{\partial \text{error}}{\partial z_k} && \text{as } z_k = \sum_j y_j w_{jk} \\
&= \sum_k w_{jk}\ \sigma'(z_k)\ \frac{\partial \text{error}}{\partial y_k} && \text{for } y_k = \sigma(z_k) \text{ and by } (\dagger)
\end{aligned}$$

propagating the error measure

$$\frac{\partial \text{error}}{\partial y_k} \text{ back to } \frac{\partial \text{error}}{\partial y_j} \quad \text{(towards the input).}$$





There's tons more to learn . . .