Learning From Data
Lecture 20
Multilayer Perceptron

Multiple layers
Universal Approximation
The Neural Network

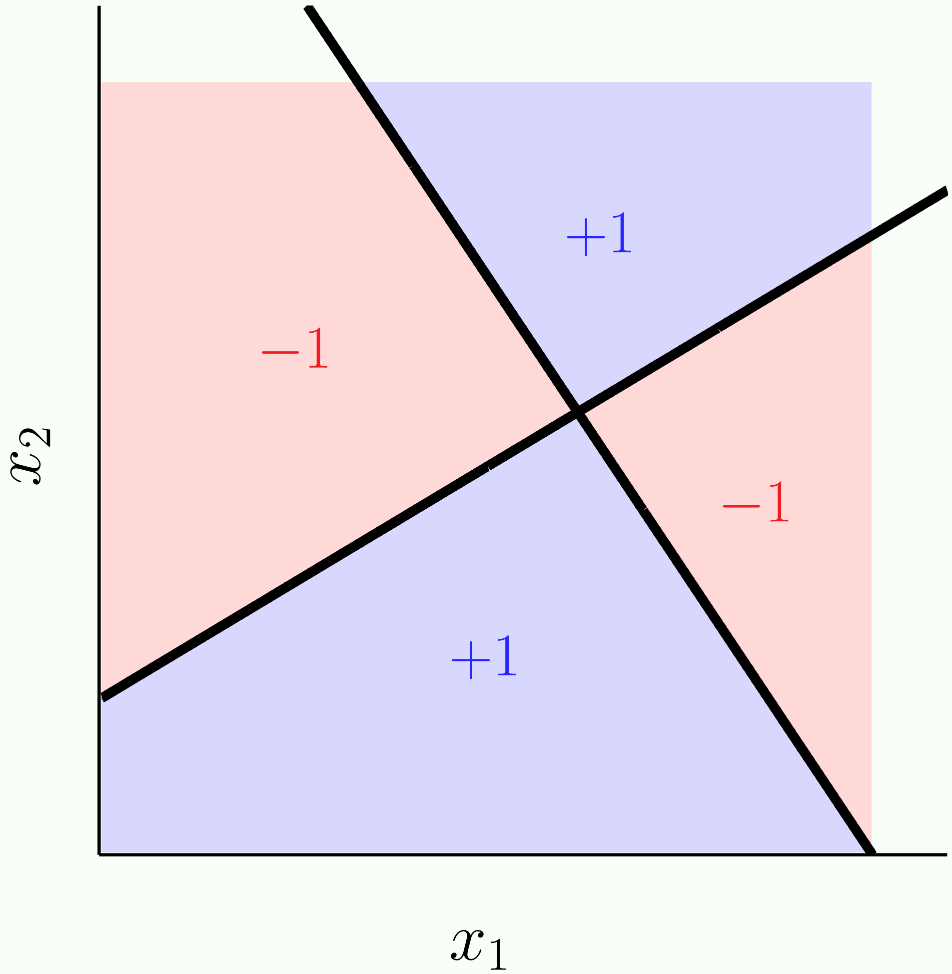
M. Magdon-Ismail
CSCI 4100/6100

The Neural Network - Biologically Inspired

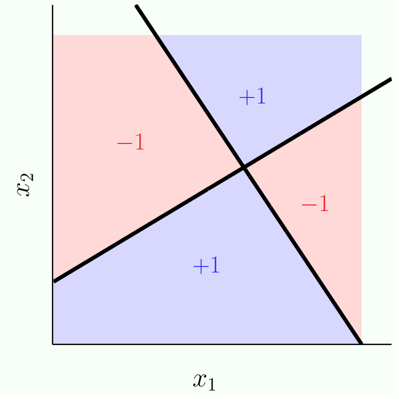
Planes Don't Flap Wings to Fly

Engineering success may start with biological inspiration, but then take a totally different path.

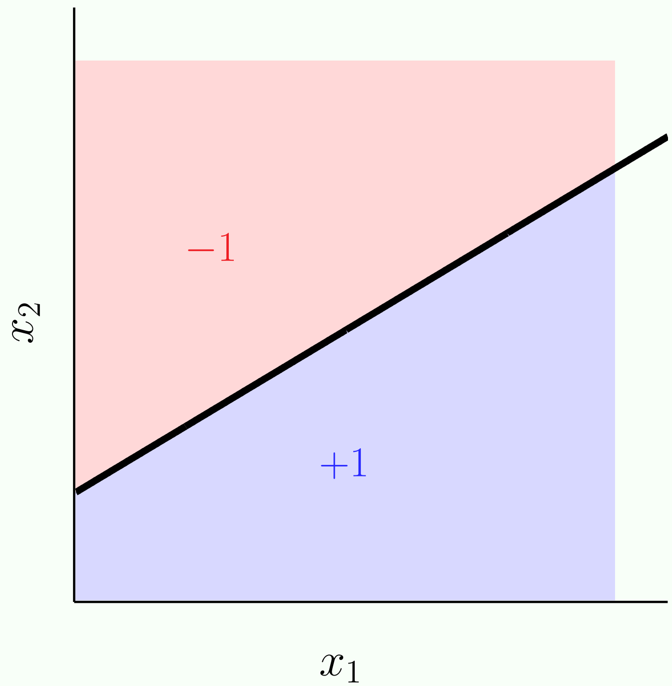
XOR: A Limitation of the Linear Model



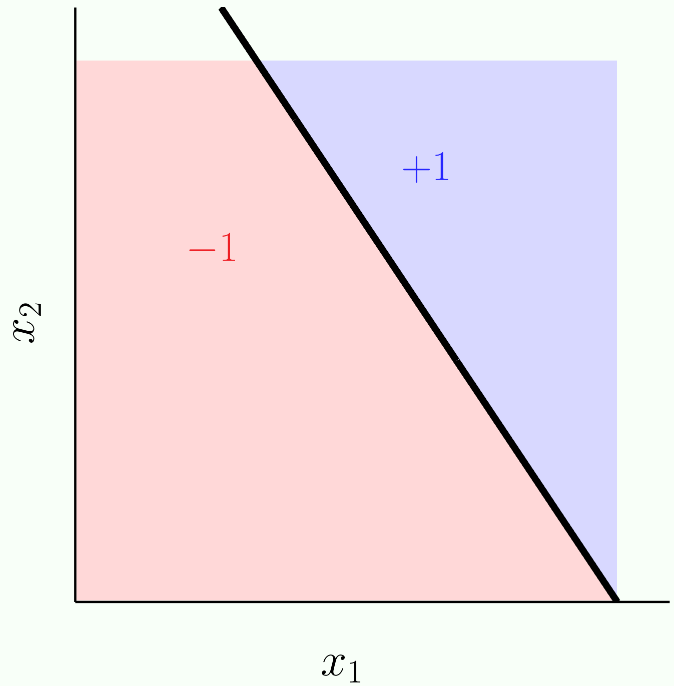
Decomposing XOR



$$f = h_1\overline{h_2} + \overline{h_1}h_2$$



$$h_1(\mathbf{x}) = \text{sign}(\mathbf{w}_1^{\text{T}}\mathbf{x})$$

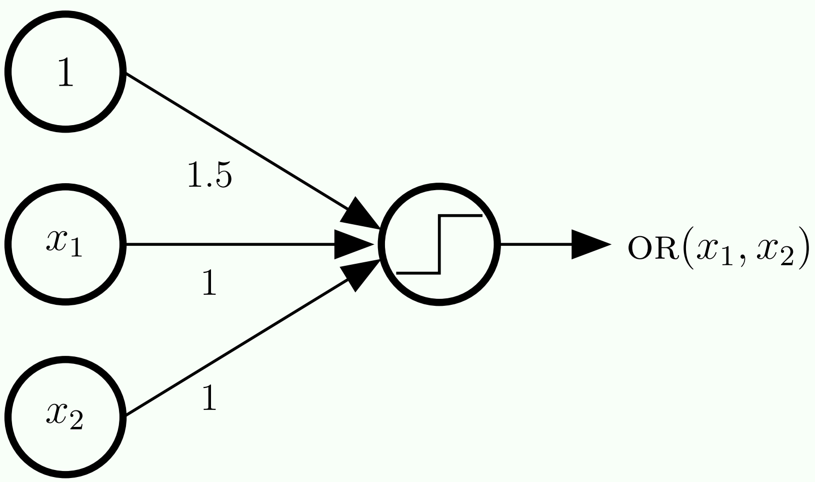


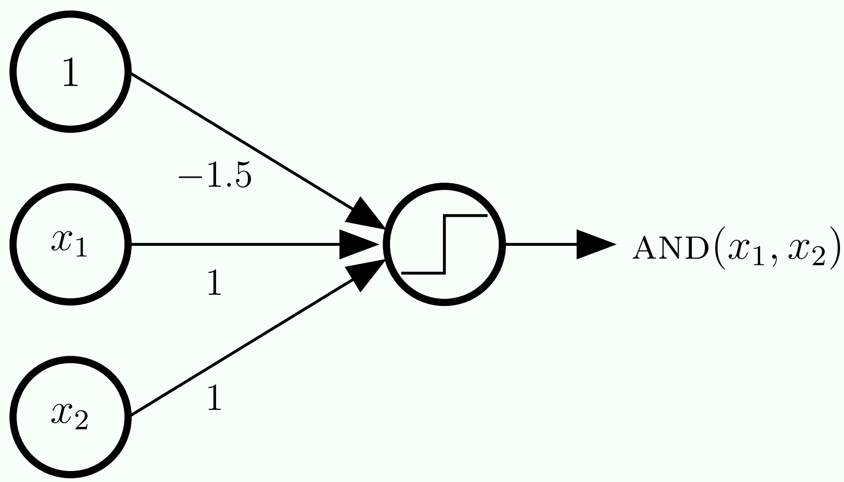
$$h_2(\mathbf{x}) = \text{sign}(\mathbf{w}_2^{\text{T}}\mathbf{x})$$

Perceptrons for OR and AND

$\text{OR}(x_1, x_2) = \text{sign}(x_1 + x_2 + 1.5)$

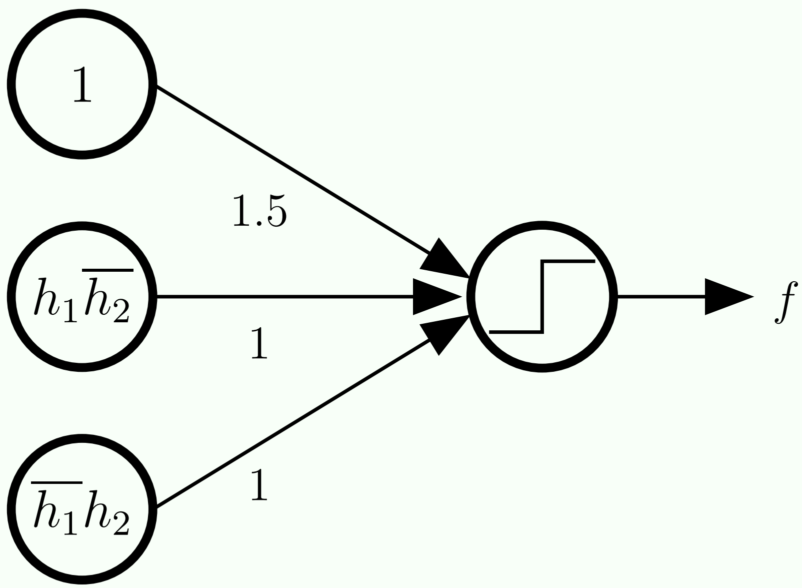
$\text{AND}(x_1, x_2) = \text{sign}(x_1 + x_2 - 1.5)$





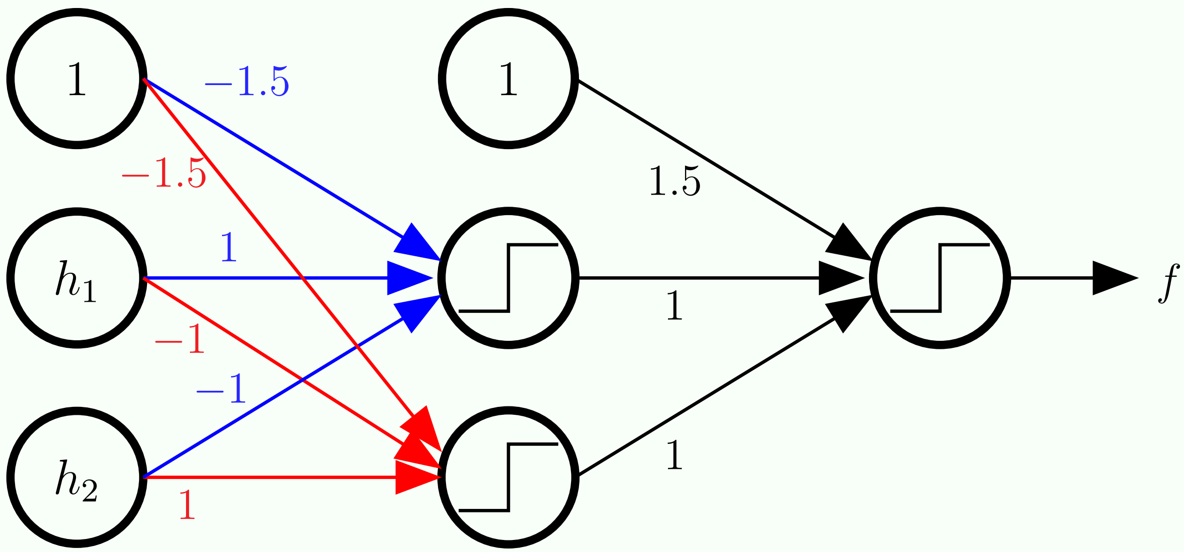
Representing f Using OR and AND

$$f = h_1\overline{h_2} + \overline{h_1}h_2$$



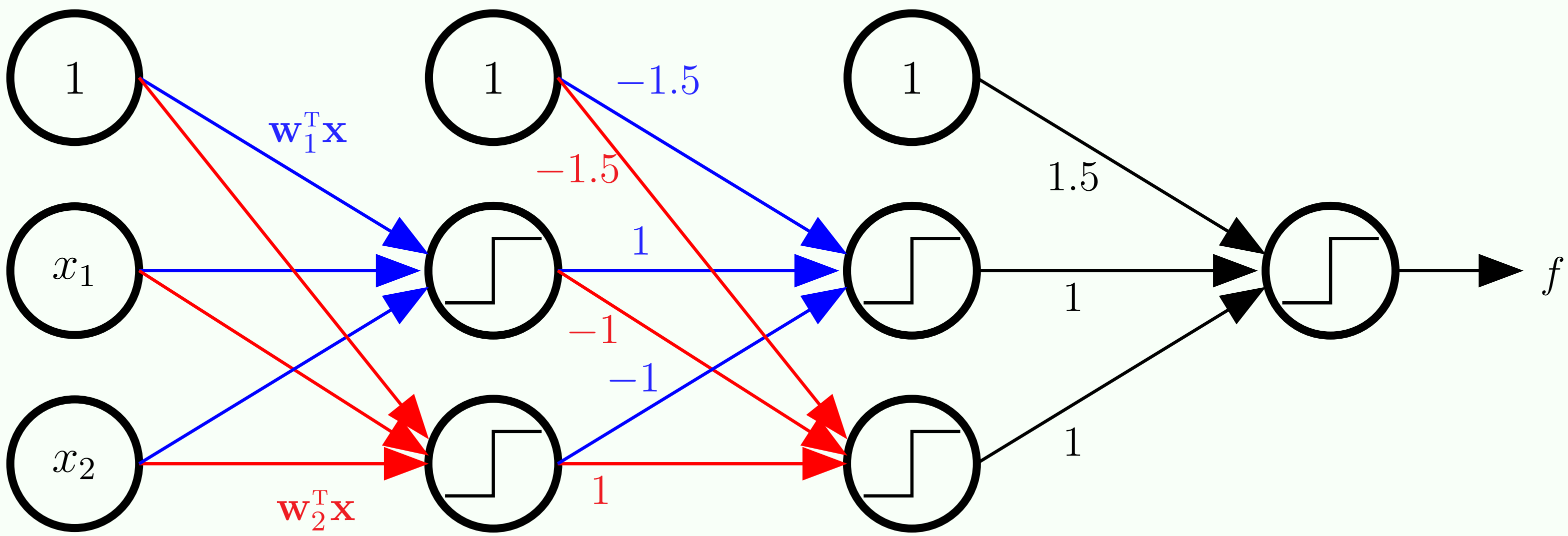
Representing f Using OR and AND

$$f = h_1\overline{h_2} + \overline{h_1}h_2$$

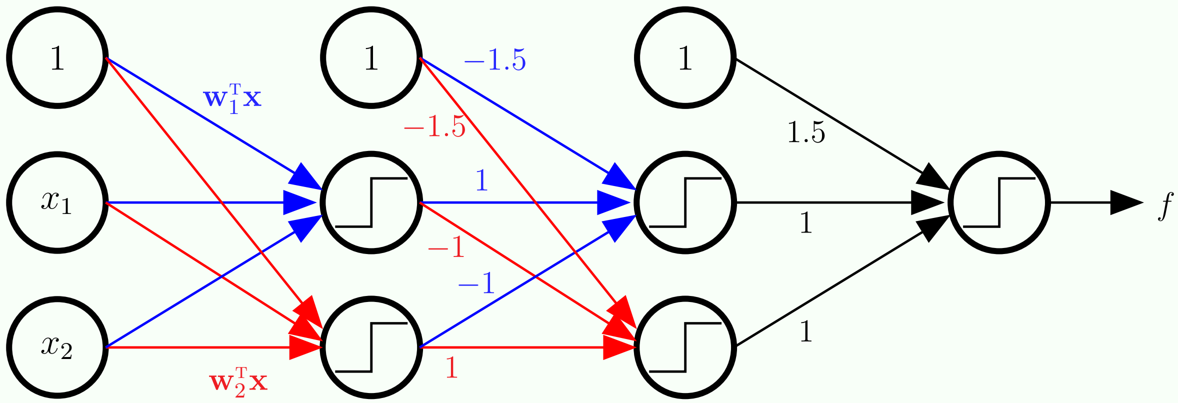


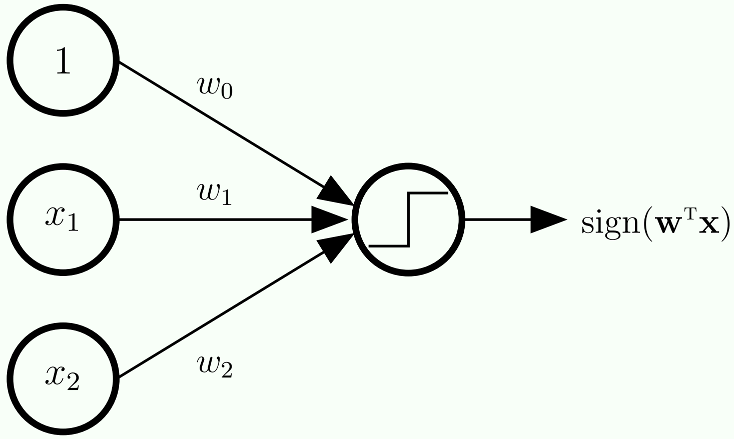
Representing f Using OR and AND

$$f = h_1\overline{h_2} + \overline{h_1}h_2$$



The Multilayer Perceptron (MLP)





More layers allow us to implement f

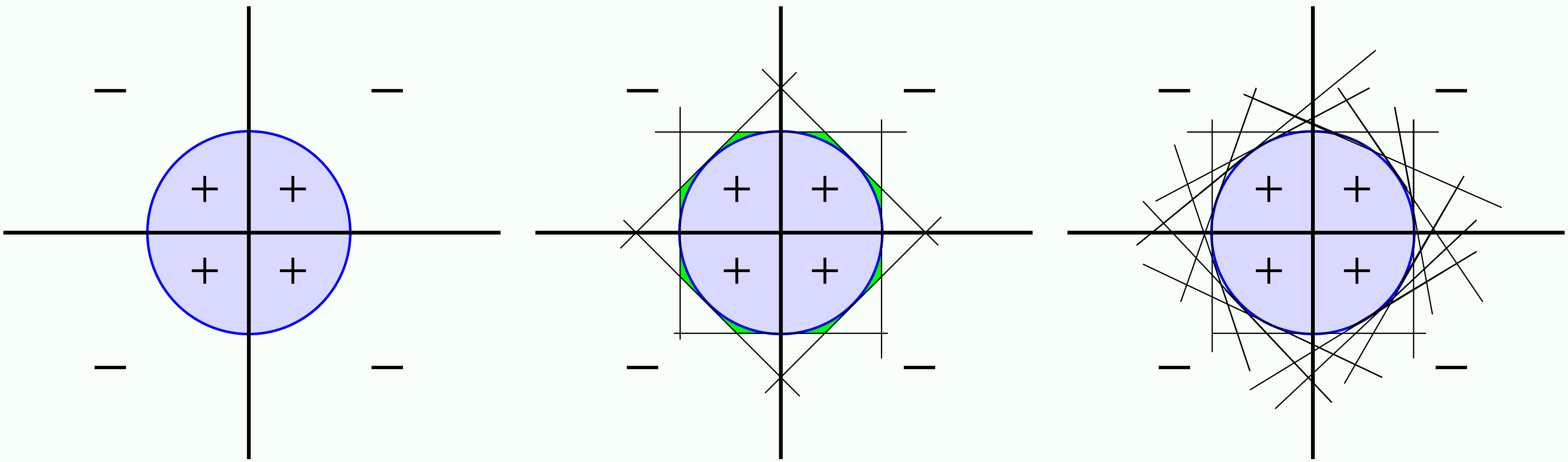
These additional layers are called *hidden layers*

Universal Approximation

Any target function f that can be decomposed into linear separators can be implemented by a 3-layer MLP.

Universal Approximation

A sufficiently smooth separator can “essentially” be decomposed into linear separators.



Target | 8 perceptrons | 16 perceptrons

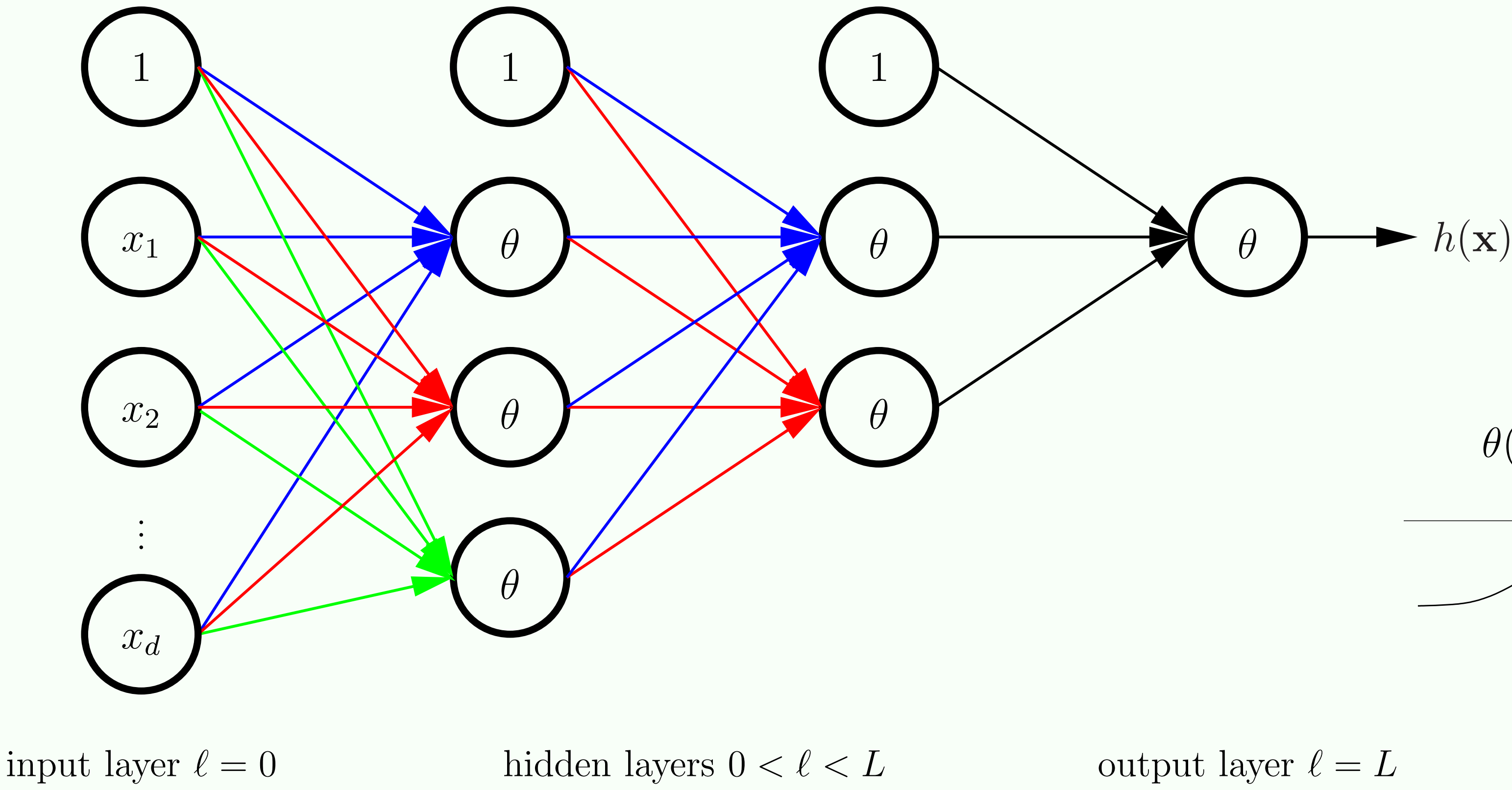
Minimizing E_{in}

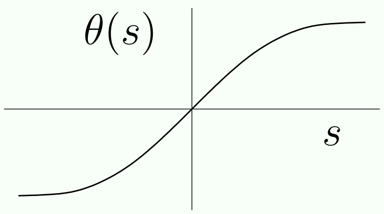
A combinatorial problem even harder with the MLP than the Perceptron.

E_{in} is not smooth (due to sign function), so cannot use gradient descent.

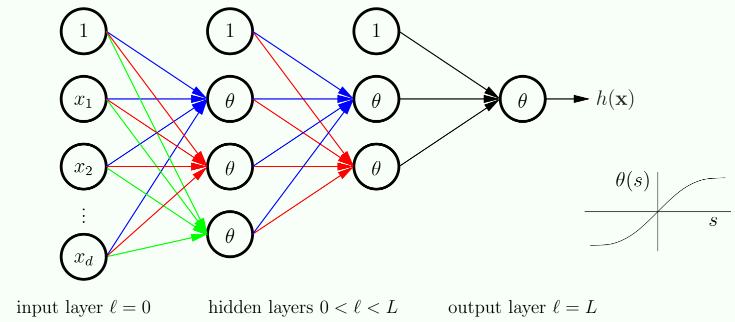
$\text{sign}(x) \approx \tan(x) \longrightarrow$ gradient descent to minimize E_{in}.

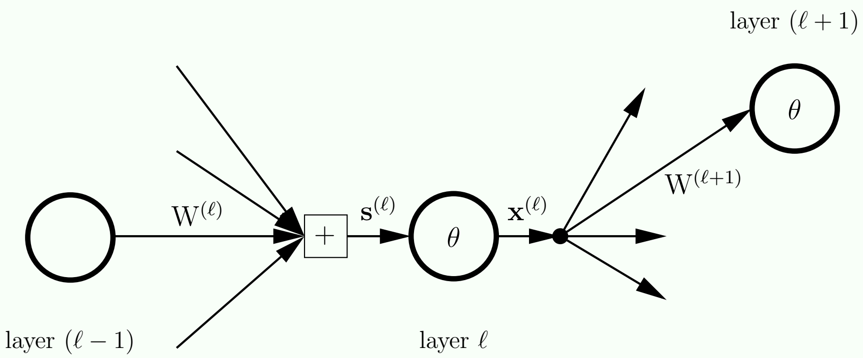
The Neural Network





Zooming into a Hidden Node





layer ℓ parameters

signals in	$\mathbf{s}^{(\ell)}$	$d^{(\ell)}$ dimensional input vector
outputs	$\mathbf{x}^{(\ell)}$	$d^{(\ell)} + 1$ dimensional output vector
weights in	$\mathrm{W}^{(\ell)}$	$(d^{(\ell-1)} + 1) \times d^{(\ell)}$ dimensional matrix
weights out	$\mathrm{W}^{(\ell+1)}$	$(d^{(\ell)} + 1) \times d^{(\ell+1)}$ dimensional matrix

layers $\ell = 0, 1, 2, \ldots, L$

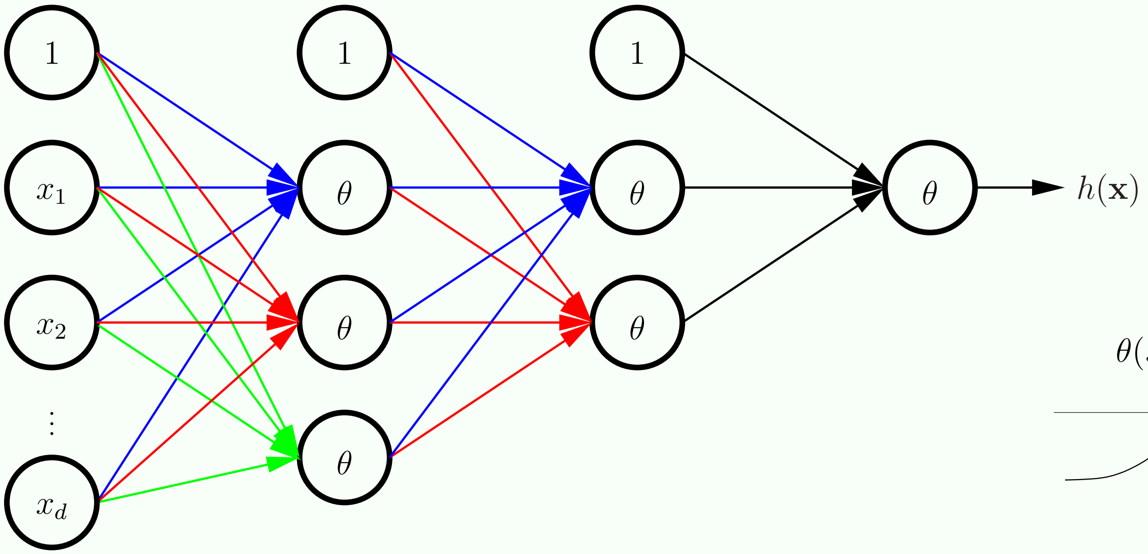
layer ℓ has "dimension" $d^{(\ell)} \implies d^{(\ell)} + 1$ nodes

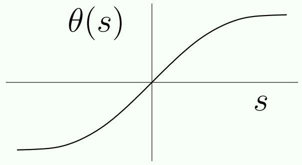
$$\mathrm{W}^{(\ell)} = \begin{bmatrix} \mathbf{w}_1^{(\ell)} & \mathbf{w}_2^{(\ell)} & \cdots & \mathbf{w}_{d^{(\ell)}}^{(\ell)} \\ \Bigg\vert & \Bigg\vert & \vdots & \Bigg\vert \end{bmatrix}$$

The Neural Network

Biology

Engineering





input layer $\ell = 0$ hidden layers $0 < \ell < L$ output layer $\ell = L$