Department of
Linguistics and Translation
香港城市大學
City University of Hong Kong

Computational Linguistics
LT3233

Jixing Li

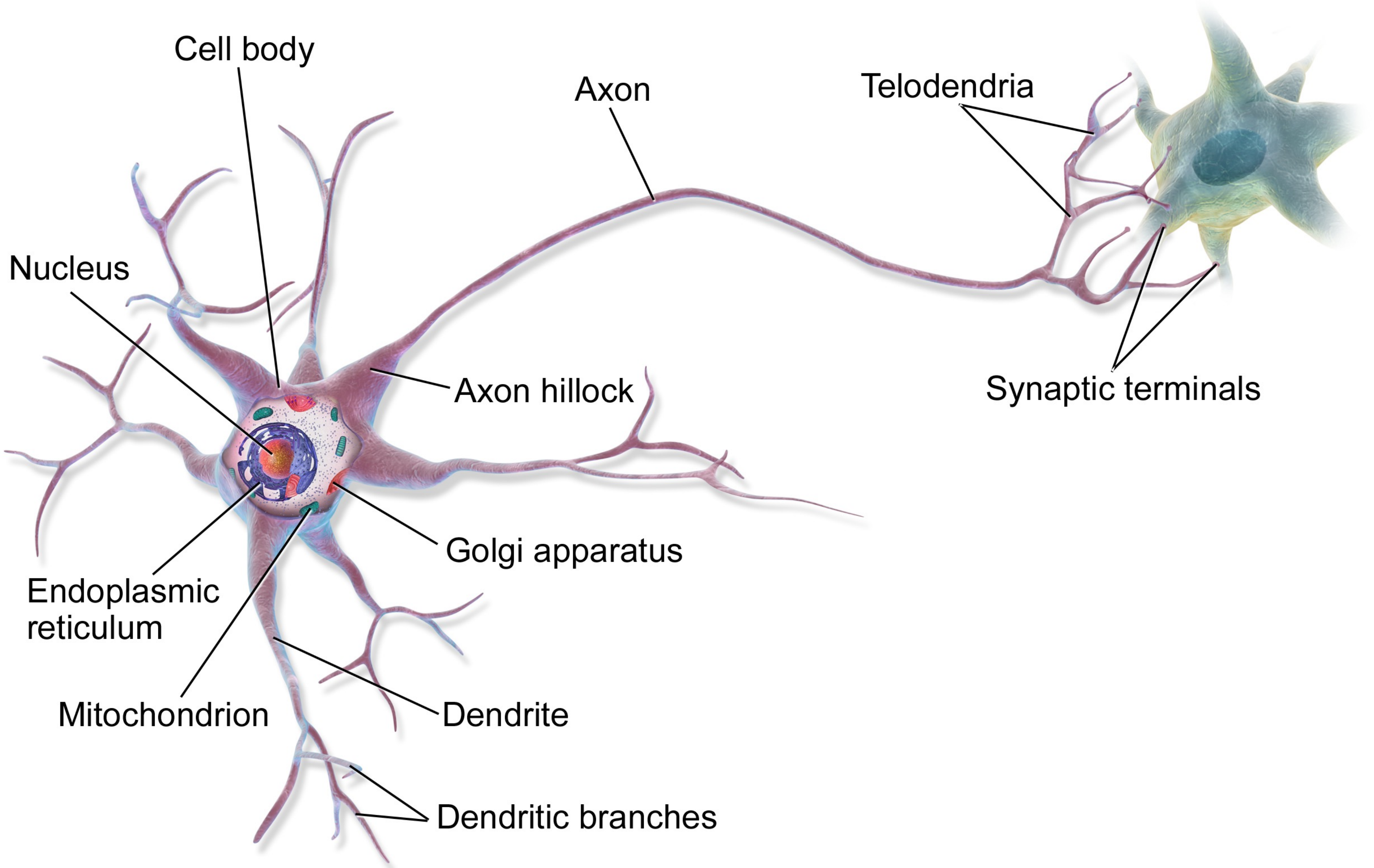
Lecture 8: Feedforward neural networks

Slides adapted from Dan Jurafsky

Lecture plan

- Neural network unit
- The XOR problem
- Feedforward neural networks
- Short break (15 mins)
- Hands-on exercises

This is in your brain



Neural network unit

This is not in your brain

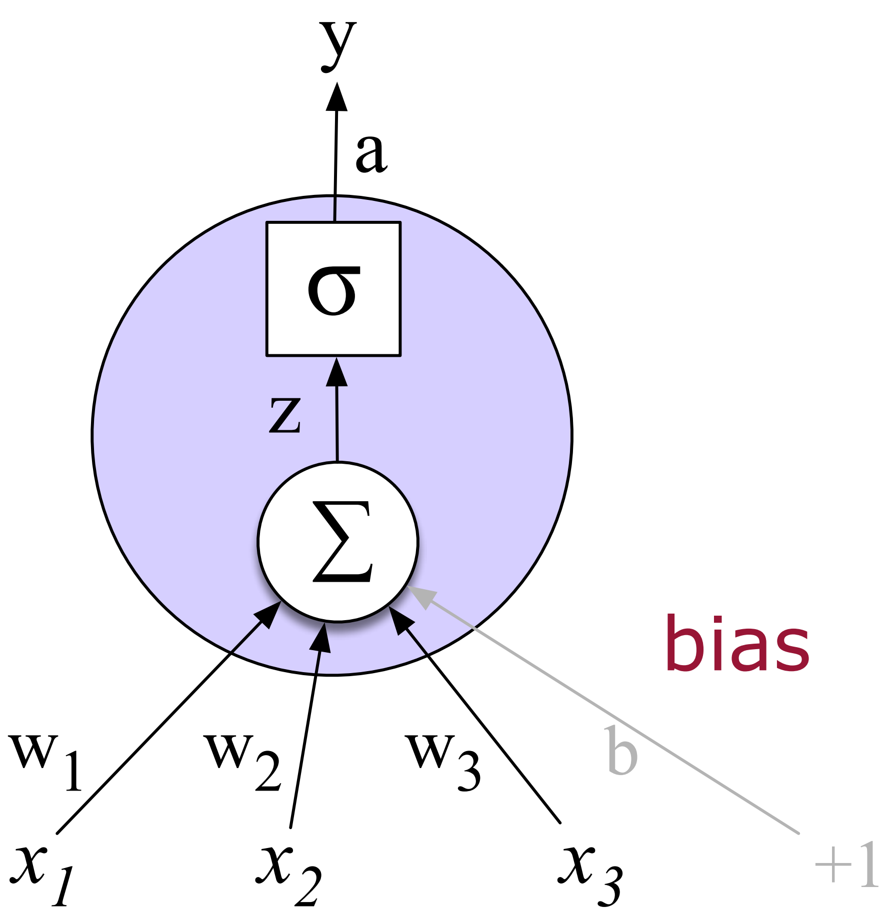
Output value

Non-linear transform

Weighted sum

Weights

Input layer



Neural unit

Take weighted sum of inputs, plus a bias

$$z = b + \sum_i w_i x_i$$

$$z = w \cdot x + b$$

Apply a nonlinear activation function f:

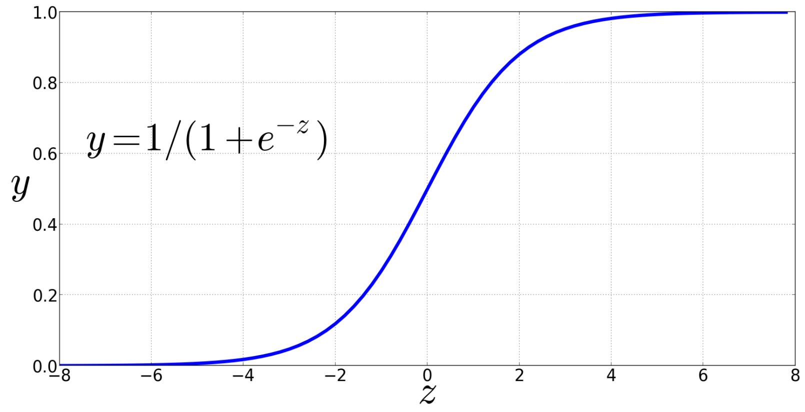
$$y = a = f(z)$$

Non-linear activation functions

We're already seen the sigmoid for logistic regression:

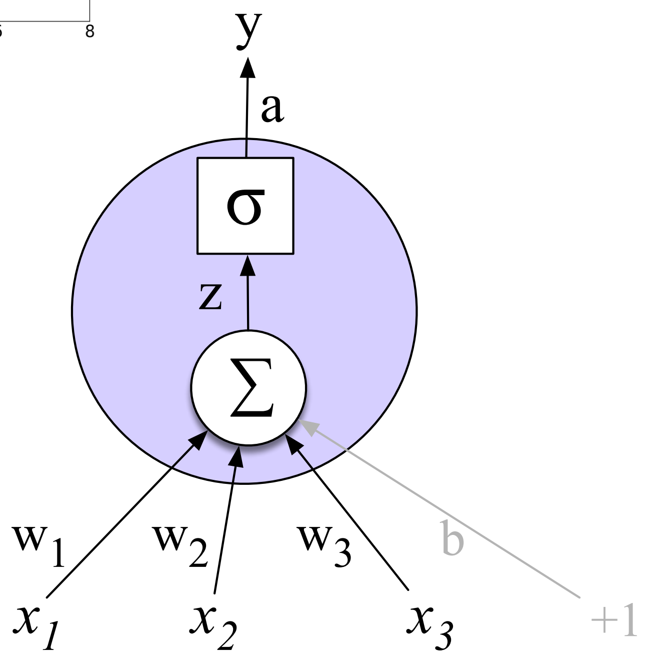
Sigmoid

$$y = \sigma(z) = \frac{1}{1+e^{-z}}$$



Final function the unit is computing:

$$y = \sigma(w \cdot x + b) = \frac{1}{1+\exp(-(w \cdot x + b))}$$



Example

Suppose a unit has:

w = [0.2,0.3,0.9]

b = 0.5

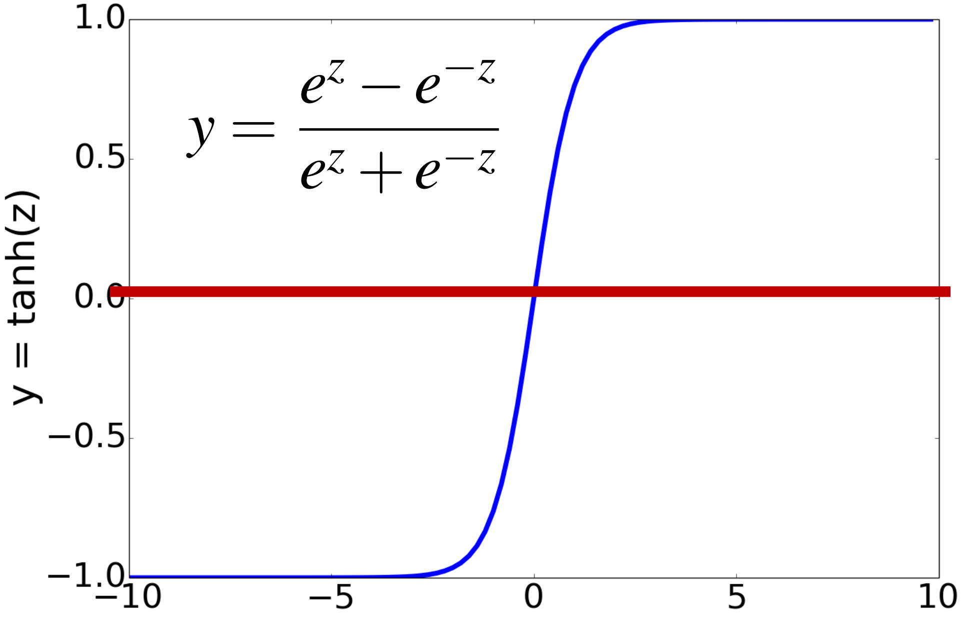
What is the output *y* for the input *x*:

x = [0.5,0.6,0.1]

$$y = \sigma(w \cdot x + b) = \frac{1}{1 + e^{-(w \cdot x + b)}}$$

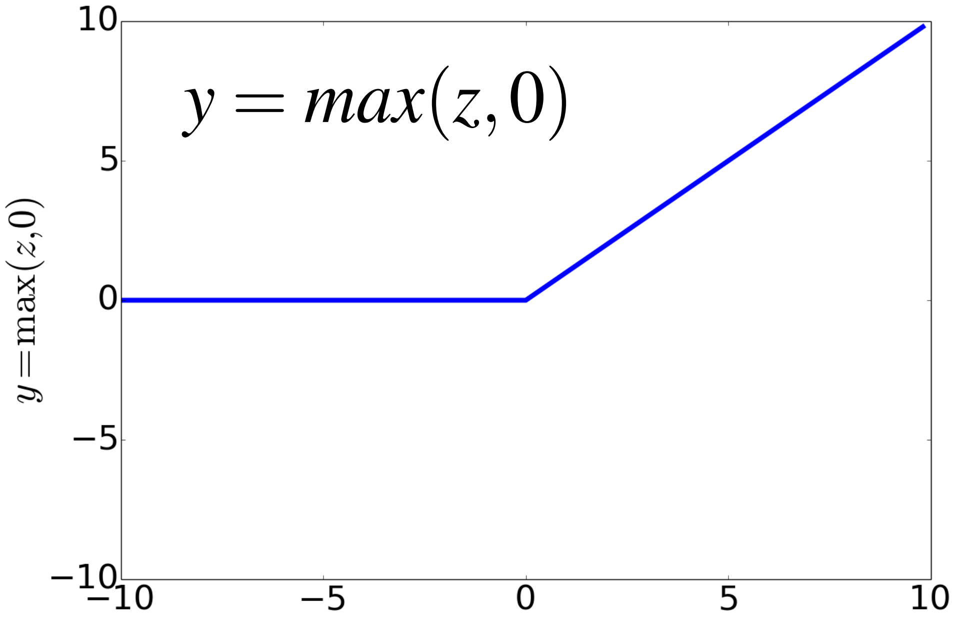
$$= \frac{1}{1 + e^{-(.5 * .2 + .6 * .3 + .1 * .9 + .5)}} = \frac{1}{1 + e^{-0.87}} = .70$$

Non-linear function besides sigmoid



tanh

hyperbolic tangent function



ReLU

Rectified Linear Unit

Most common

The XOR problem

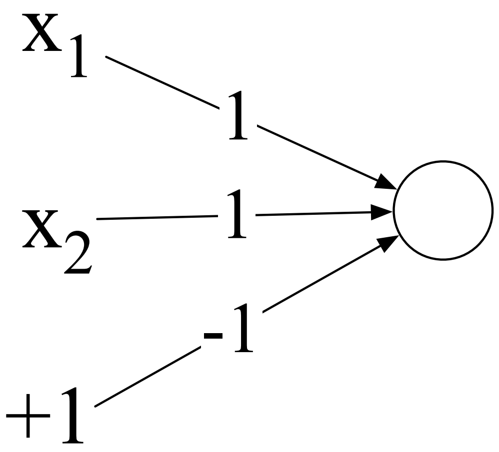
Can neural units compute simple functions of input?

AND			OR			XOR		
x1	x2	y	x1	x2	y	x1	x2	y
0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	1
1	0	0	1	0	1	1	0	1
1	1	1	1	1	1	1	1	0

Perceptrons

A very simple neural unit
Binary output (0 or 1)

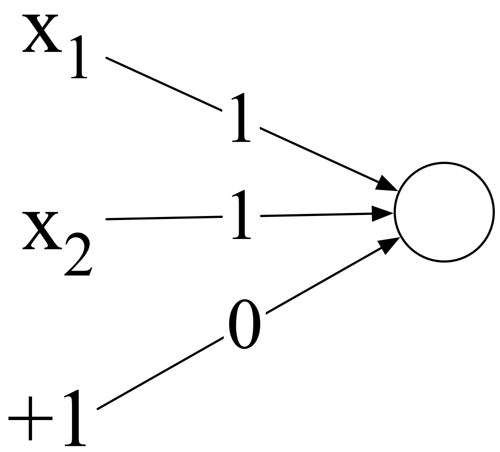
$$y = \begin{cases} 0, & \text{if } w \cdot x + b \leq 0 \\ 1, & \text{if } w \cdot x + b > 0 \end{cases}$$



AND

x1	x2	y
0	0	0
0	1	0
1	0	0
1	1	1

AND



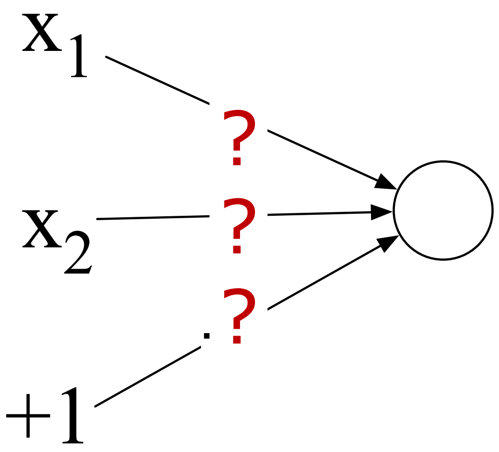
OR

x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	1

OR

XOR with perceptrons?

$$y = \begin{cases} 0, & \text{if } w \cdot x + b \leq 0 \\ 1, & \text{if } w \cdot x + b > 0 \end{cases}$$



XOR

XOR

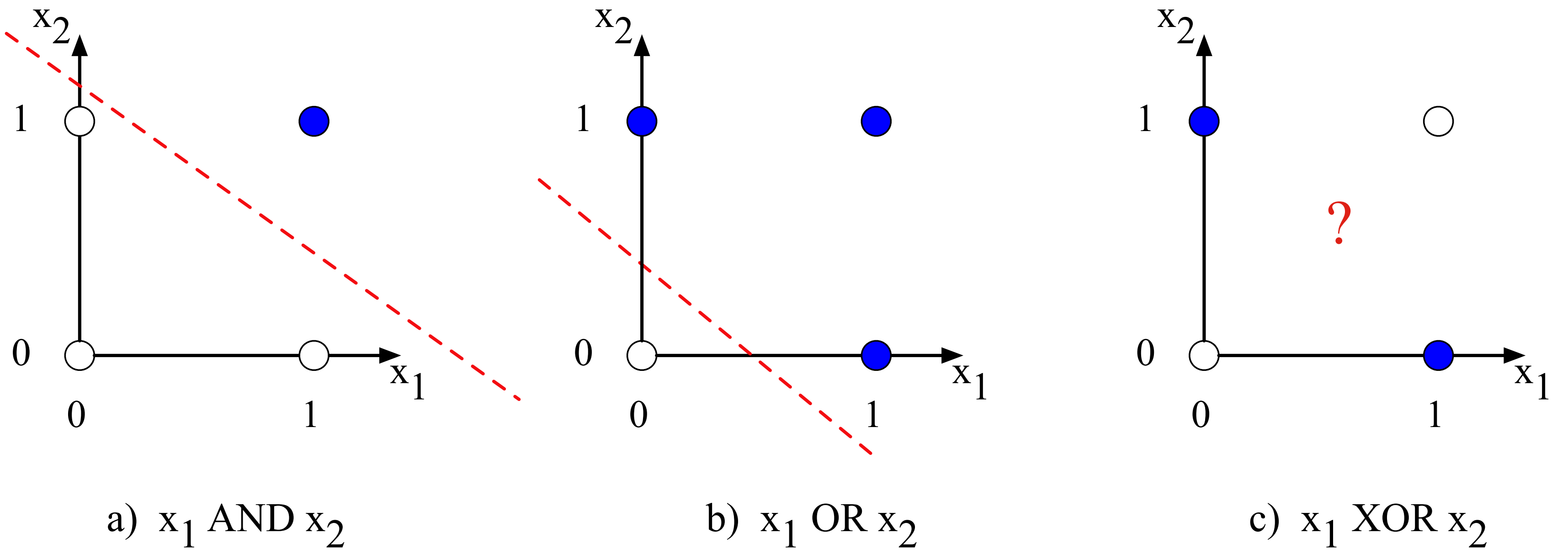
x1	x2	y
0	0	0
0	1	1
1	0	1
1	1	0

Not possible

Perceptrons are linear classifiers

Perceptron equation given x1 and x2, is the equation of a line

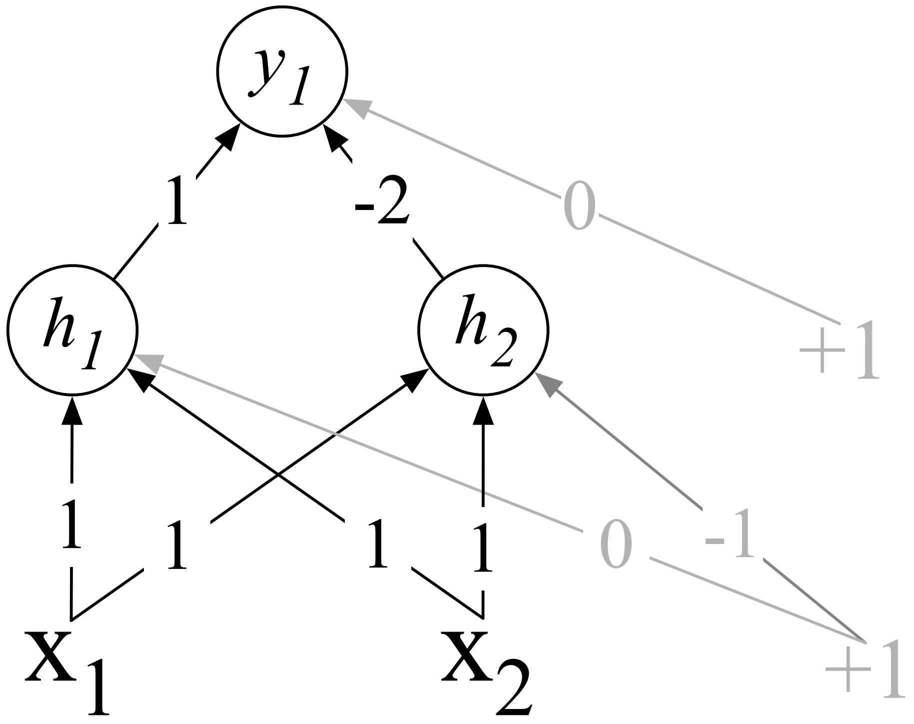
$$w_1x_1 + w_2x_2 + b = 0 \rightarrow x_2 = (-w_1/w_2)x_1 + (-b/w_2)$$



This line acts as a **decision boundary**

Solution to the XOR problem

A layered network of unit:



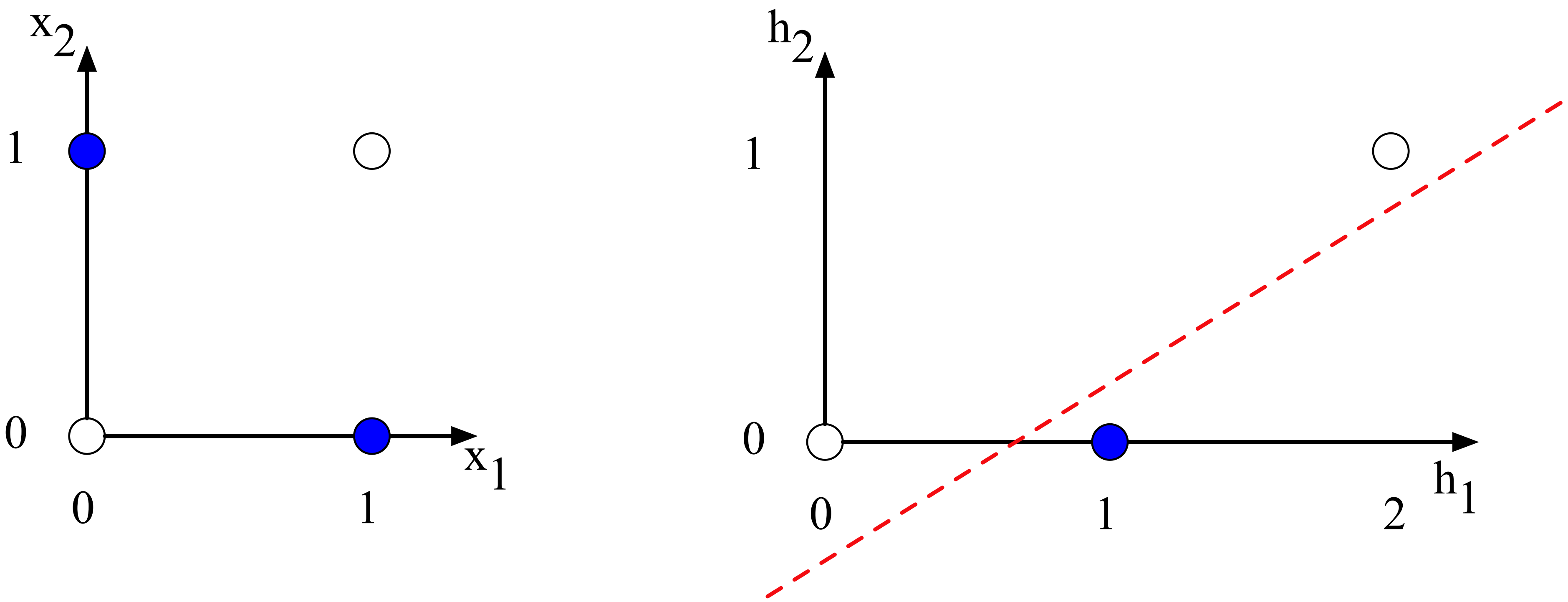
XOR				
x1	x2	y	h1	h2
0	0	0	0	0
0	1	1	1	0
1	0	1	1	0
1	1	0	2	1

Activation function: ReLU

$$y = max(z, 0)$$

The hidden representation h

A layered network of unit:



a) The original *x* space

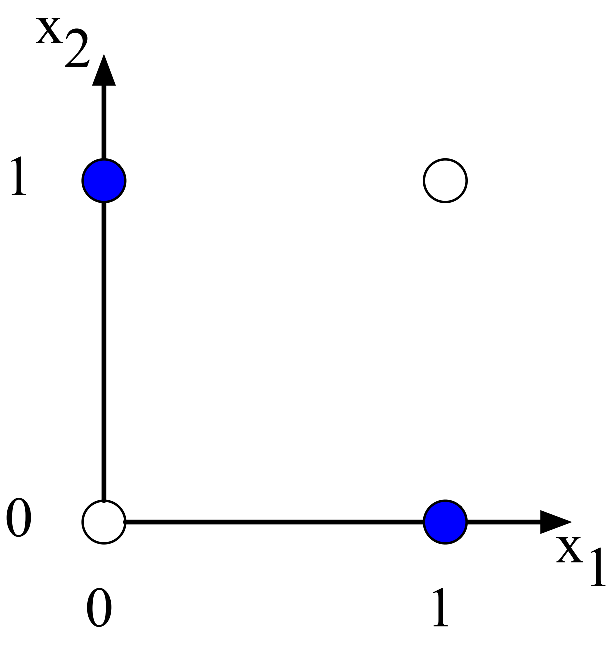
b) The new (linearly separable) *h* space

h1	h2
0	0
1	0
1	0
2	1

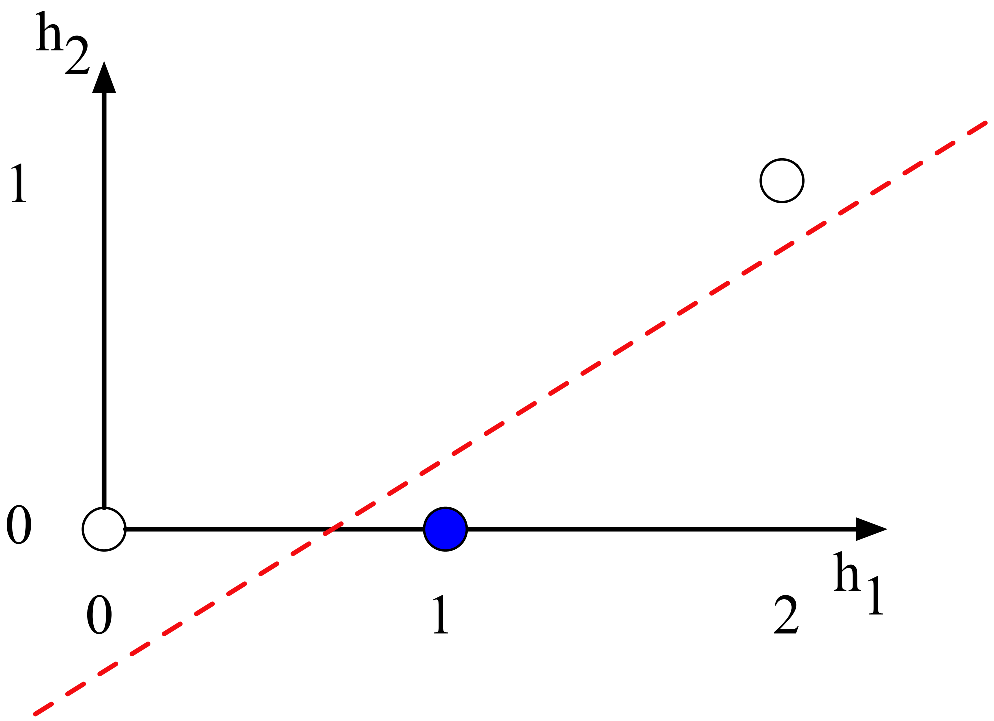
hidden layers learn to form useful representations

The hidden representation h

A layered network of unit:



a) The original *x* space



b) The new (linearly separable) *h* space

h1	h2
0	0
1	0
1	0
2	1

hidden layers learn to form useful representations

Feedforward neural networks

Two-layer network with scalar output

output layer

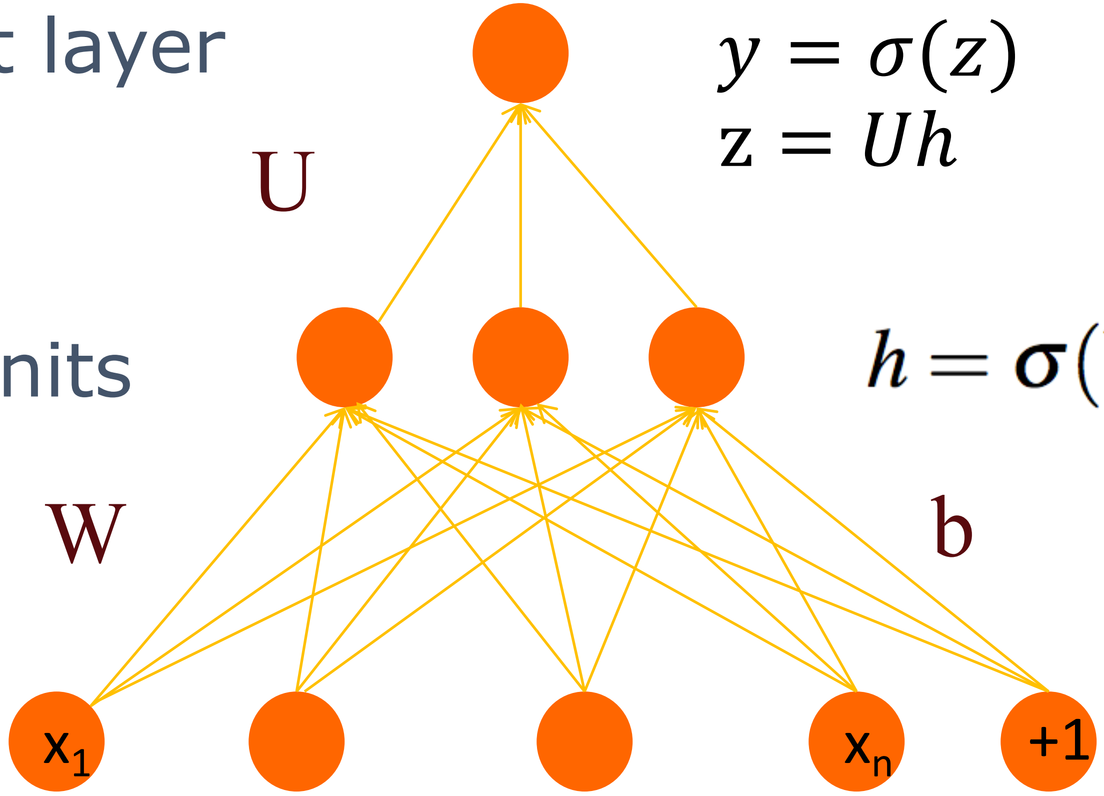
$y = \sigma(z)$
$z = Uh$

y is a scalar

hidden units

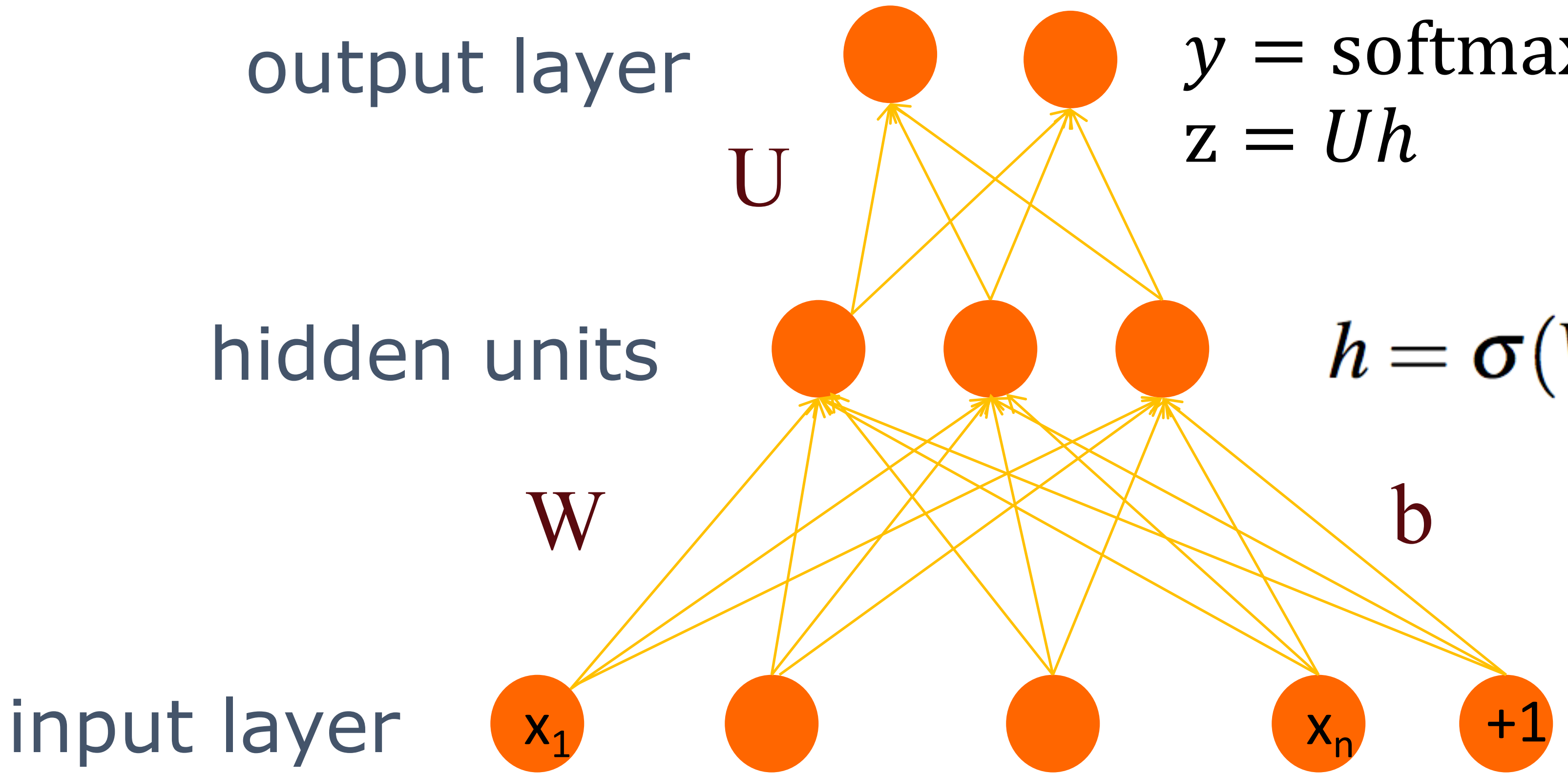
$h = \sigma(Wx + b)$

input layer



Feedforward neural networks

Two-layer network with softmax output



$y = \text{softmax}(z)$
$z = Uh$

$h = \sigma(Wx + b)$

The softmax function

Turns a vector $z = [z_1, z_2, \ldots, z_k]$ of k arbitrary values into probabilities

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \quad 1 \le i \le k$$

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)}, \ldots, \frac{\exp(z_k)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

$$z = [0.6, 1.1, -1.5, 1.2, 3.2, -1.1]$$

$$\text{softmax}(z) = [0.055, 0.090, 0.0067, 0.10, 0.74, 0.010]$$

Example

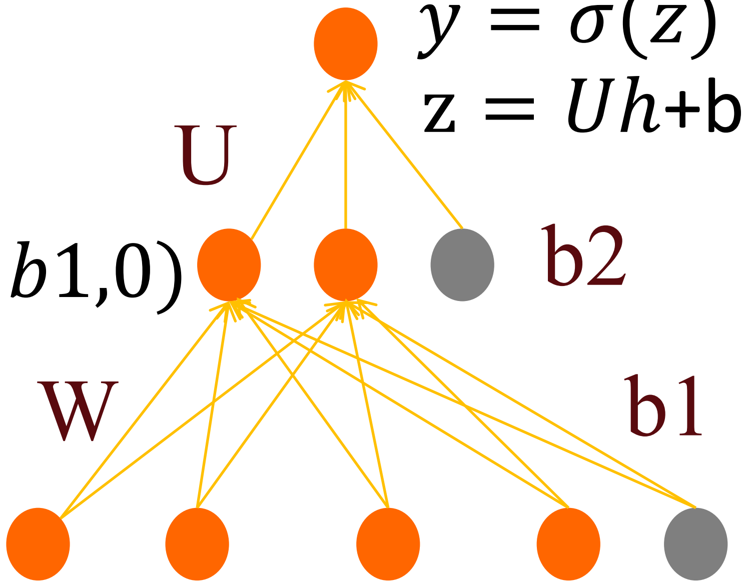
chinese_name	major	gender	n1_male	n2_male	n1_uniqueness	n2_uniqueness
林加敏	LLA	F	0.442	-0.562	2.795	2.087

x = [0.442,-0.562,2.795,2.087]
W = [[1,1,1,1],
[1,1,1,1]]
b1 = [1,1], b2 = 1
U = [1,1]
h = ReLU(W*x+b1)
y = sigmoid(U*h+b2)

$y = \sigma(z)$

z $= Uh$+b2

h $= \max(Wx + b1, 0)$



To do

- Optional reading: **SLP** Ch7.1-7.3