CS 4104
APPLIED MACHINE LEARNING

Dr. Hashim Yasin

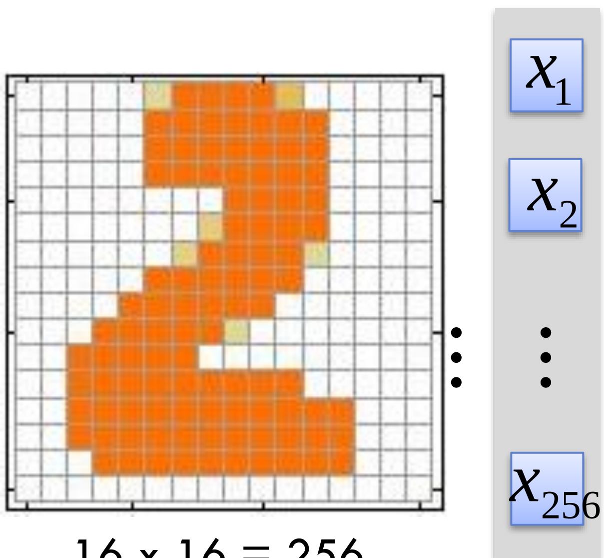
National University of Computer and Emerging Sciences,

Faisalabad, Pakistan.

DEEP LEARNING

Handwriting Digit Recognition

- Input

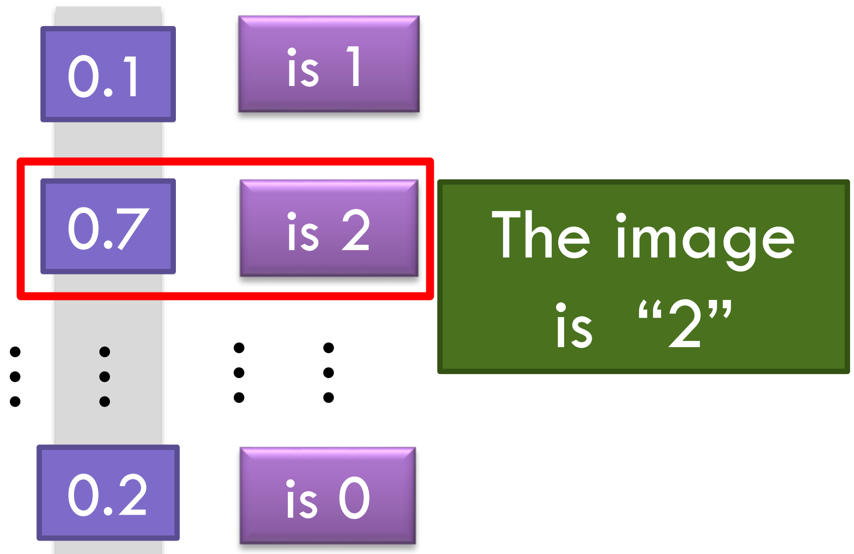


16 x 16 = 256

Ink → 1

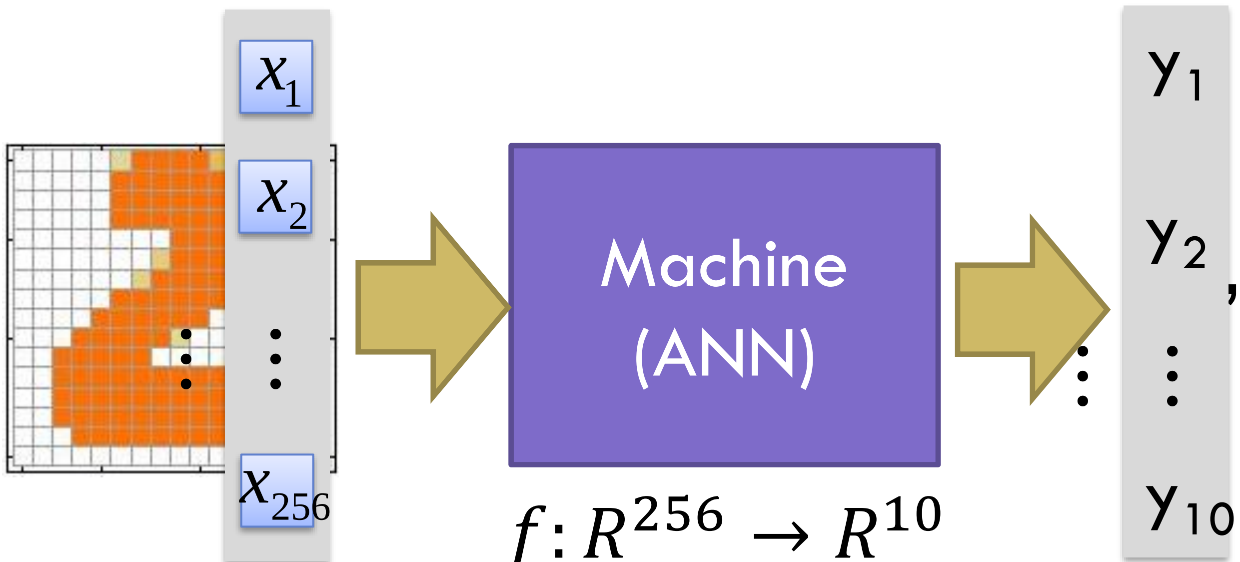
No ink → 0

- Output



Each dimension represents the confidence of a digit.

Handwriting Digit Recognition



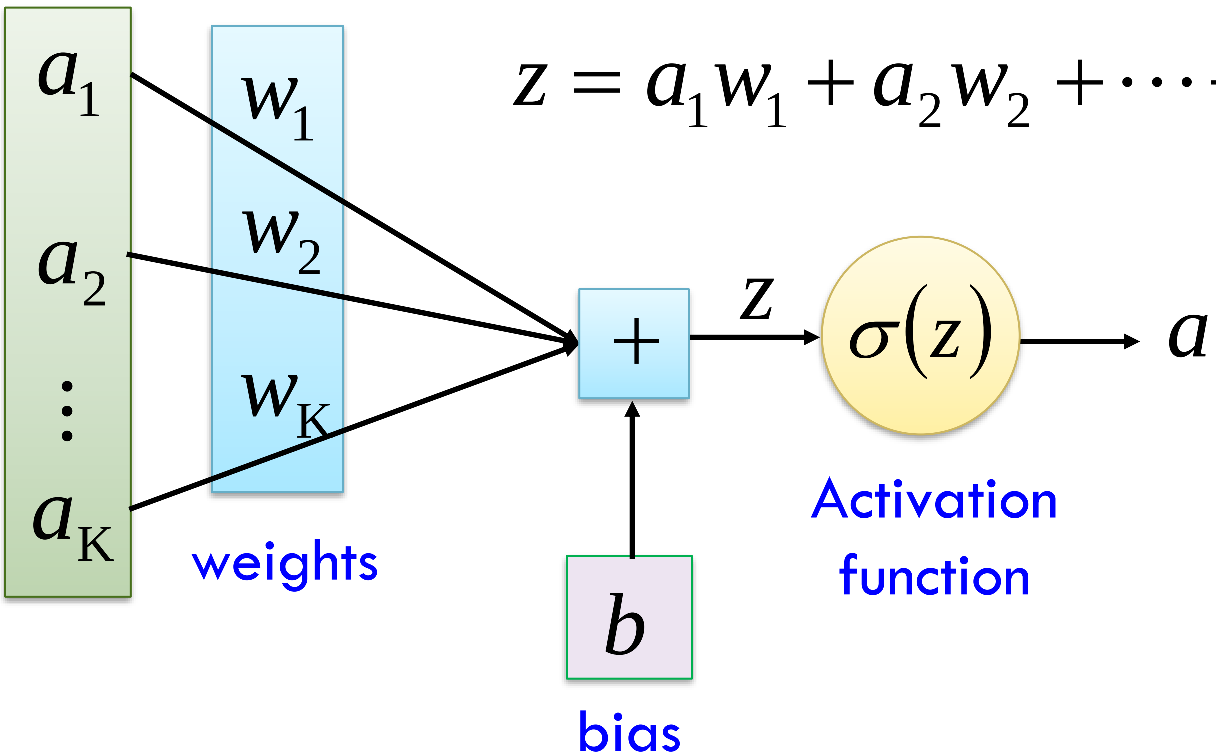
$$f: R^{256} \rightarrow R^{10}$$

In deep learning, the function f is represented by neural network

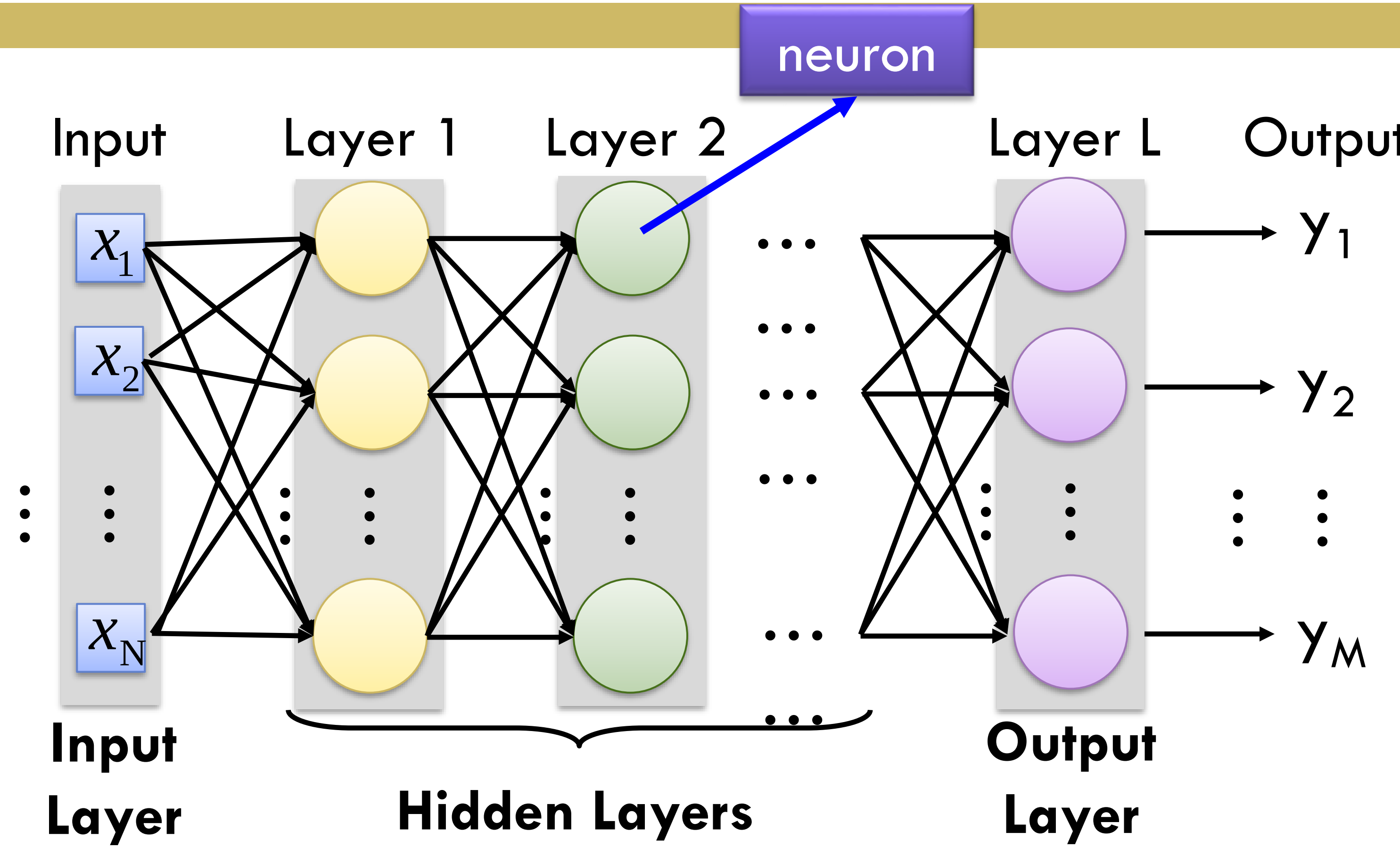
Neural Network Elements

Neuron $f: R^K \rightarrow R$

$$z = a_1 w_1 + a_2 w_2 + \cdots + a_K w_K + b$$

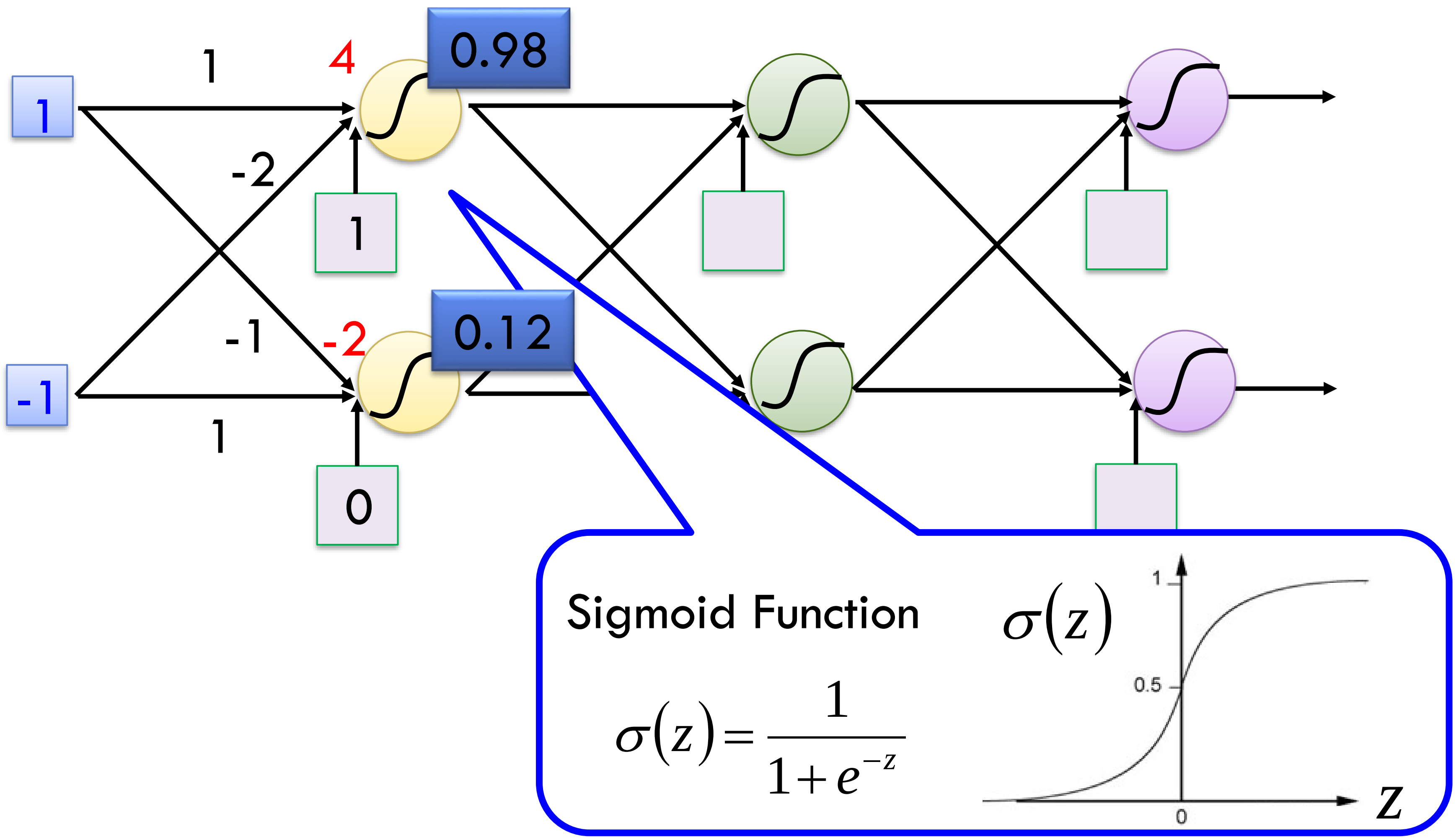


Deep Neural Network



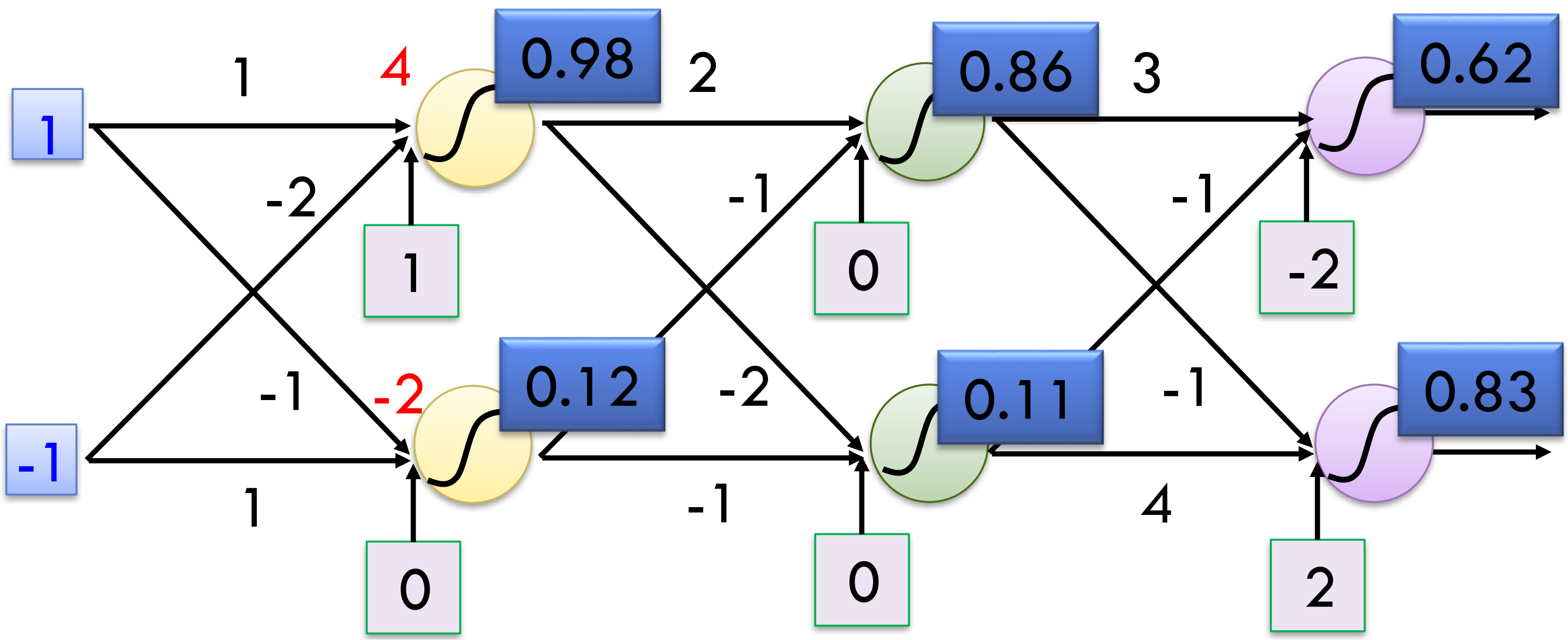
Deep means many hidden layers

Neural Network



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

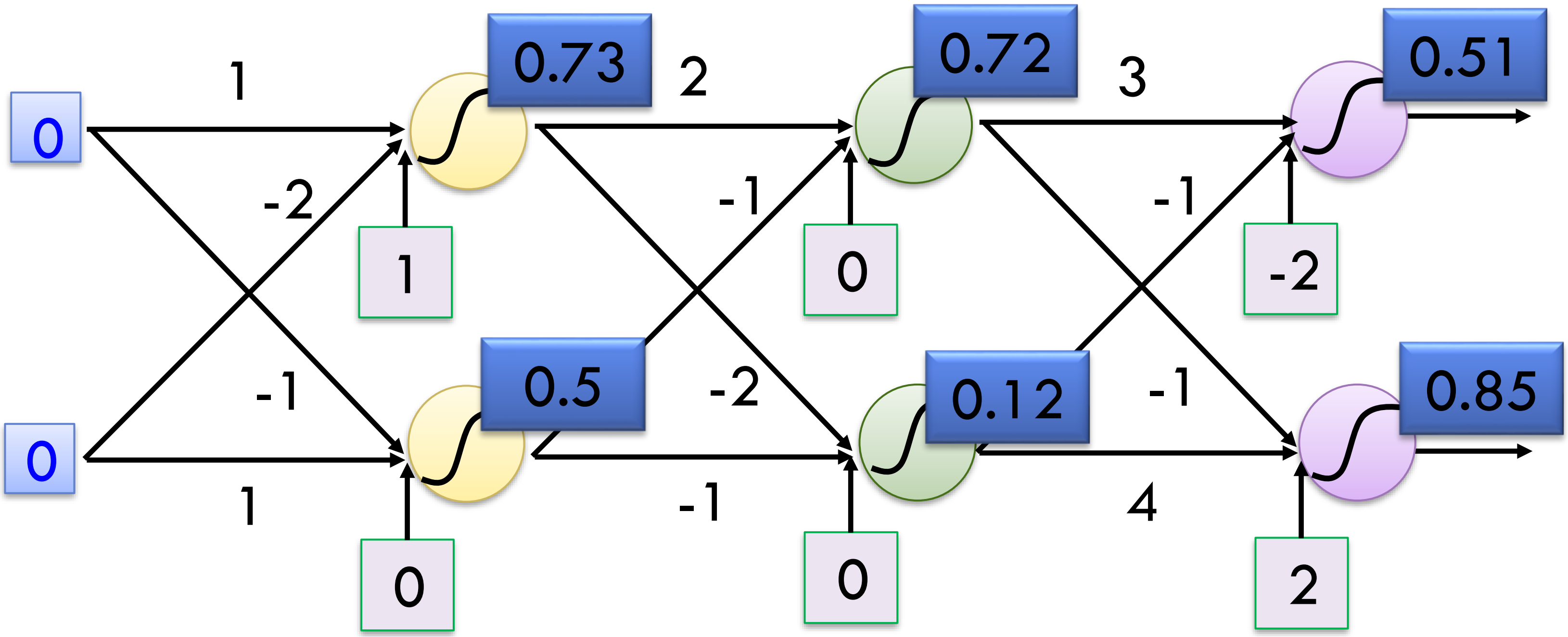
Neural Network



$f: R^2 \rightarrow R^2$

$$f\left(\begin{bmatrix}1\\-1\end{bmatrix}\right) = \begin{bmatrix}0.62\\0.83\end{bmatrix}$$

Neural Network

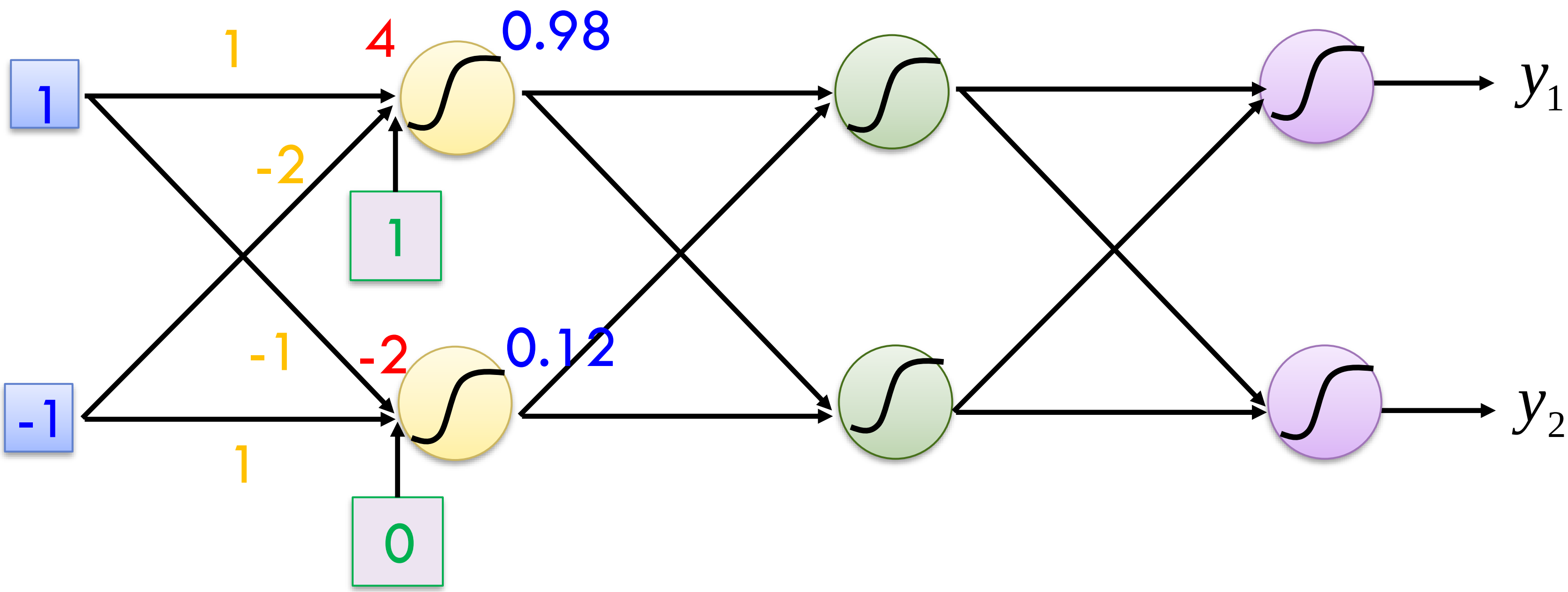


$$f: R^2 \rightarrow R^2$$

$$f\left(\begin{bmatrix}0\\0\end{bmatrix}\right) = \begin{bmatrix}0.51\\0.85\end{bmatrix}$$

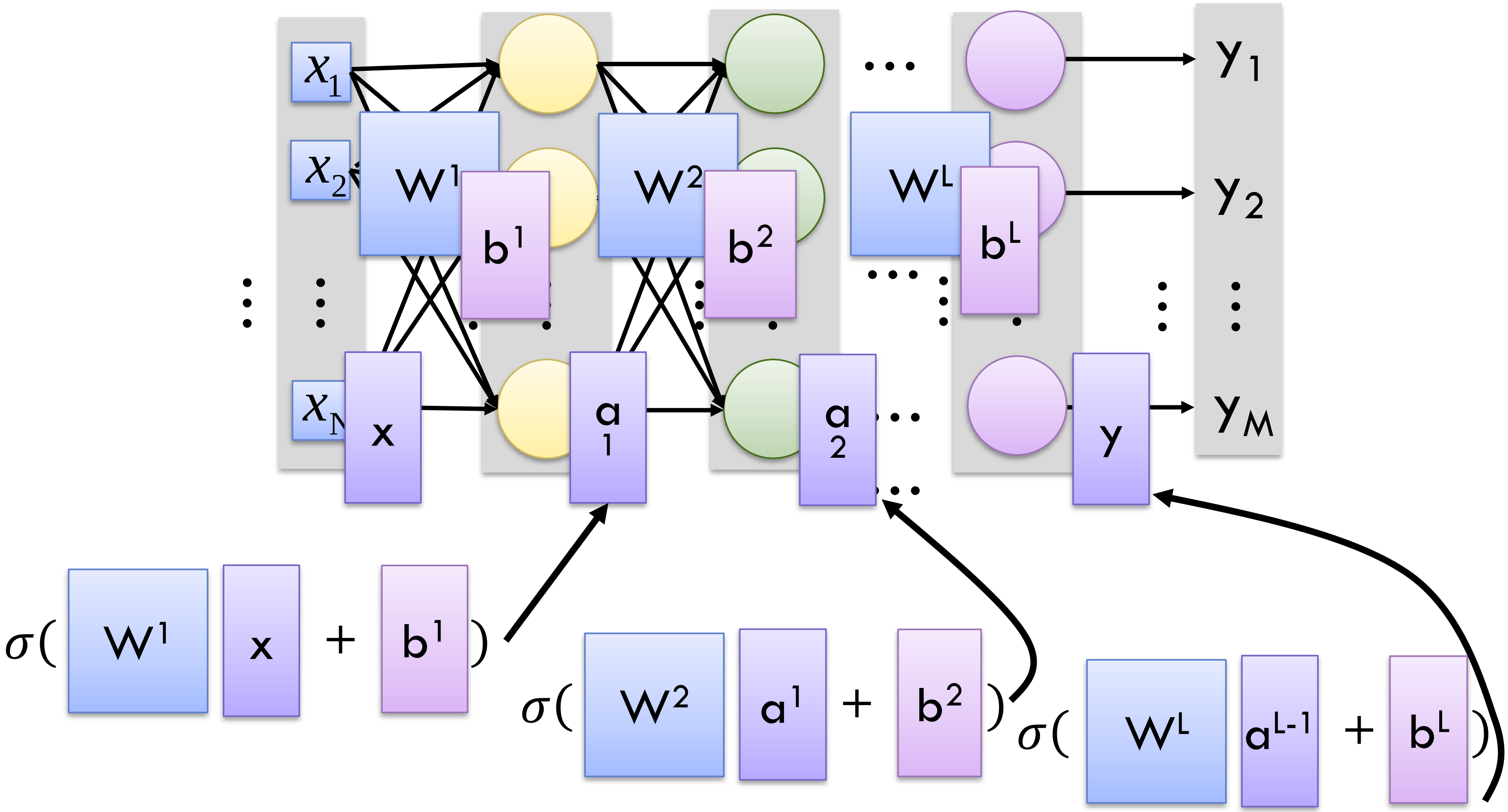
Different parameters define different function

Matrix Operation

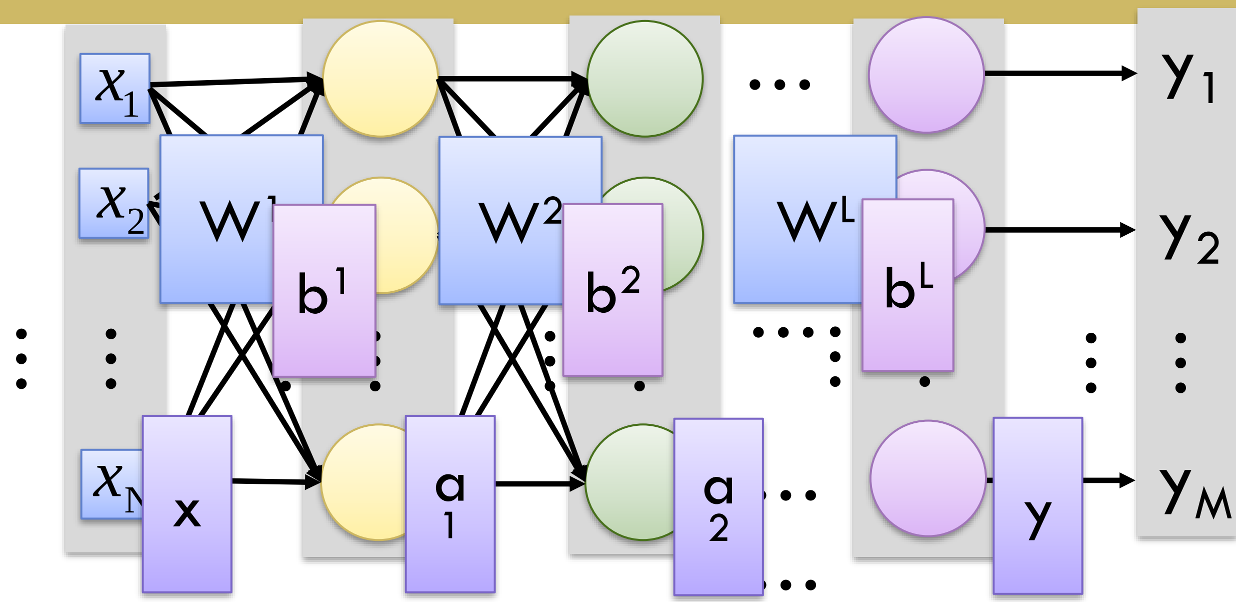


$$\sigma\left(\underbrace{\begin{bmatrix} 1 & -2 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}}_{\begin{bmatrix} 4 \\ -2 \end{bmatrix}} \right) = \begin{bmatrix} 0.98 \\ 0.12 \end{bmatrix}$$

Neural Network



Neural Network



$$\mathbf{y} = f(\mathbf{x})$$

Using parallel computing techniques to speed up matrix operation

$$= \sigma\left(W^L \cdots \sigma\left(W^2 \, \sigma\left(W^1 \mathbf{x} + b^1\right) + b^2\right) \cdots + b^L\right)$$

SOFTMAX LAYER

Softmax

Properties

- The calculated probabilities will be in the range of 0 to 1.
- The sum of all the probabilities equals 1.

Softmax Function Usage

- Used in *multiple classification logistic regression model*.
- In building neural networks, softmax functions are used in different layer levels, mostly in the output layer.

Softmax

- The formulation of softmax function is,

$$S(z_i) = \frac{e^{z_i}}{\sum_j^n e^{z_j}}$$

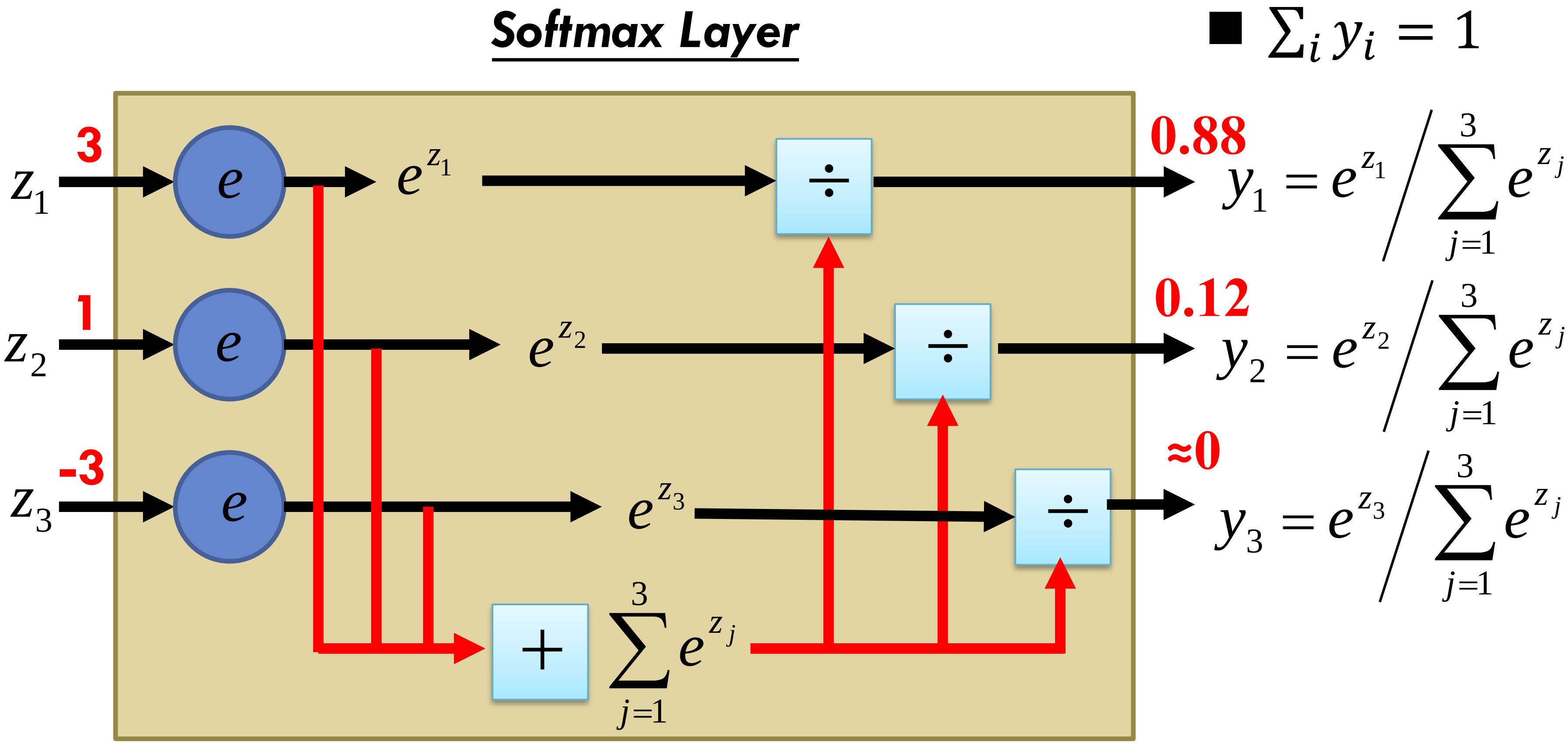
- The bigger the z, the higher its probability.

Softmax

- Softmax layer as the output layer

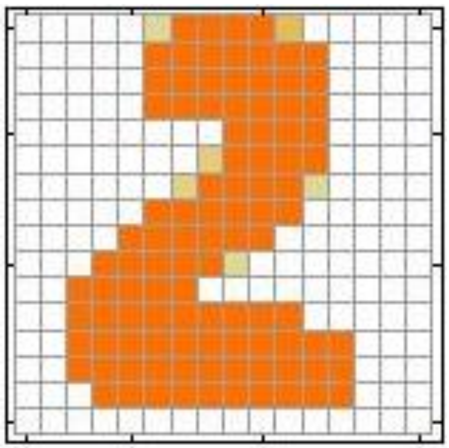
Probability:

- $1 > y_i > 0$
- $\sum_i y_i = 1$



Network Parameters

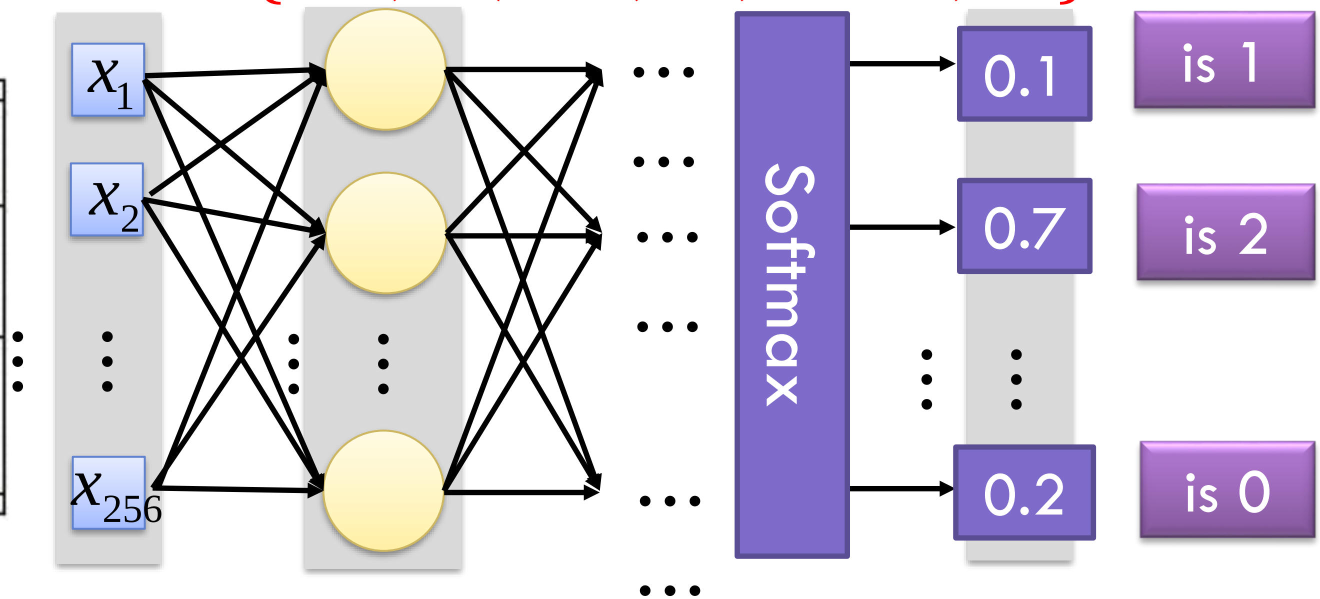
$$\theta = \{W^1, b^1, W^2, b^2, \cdots W^L, b^L\}$$



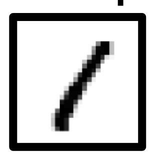
16 x 16 = 256

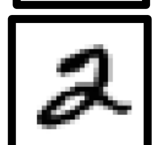
Ink → 1

No ink → 0



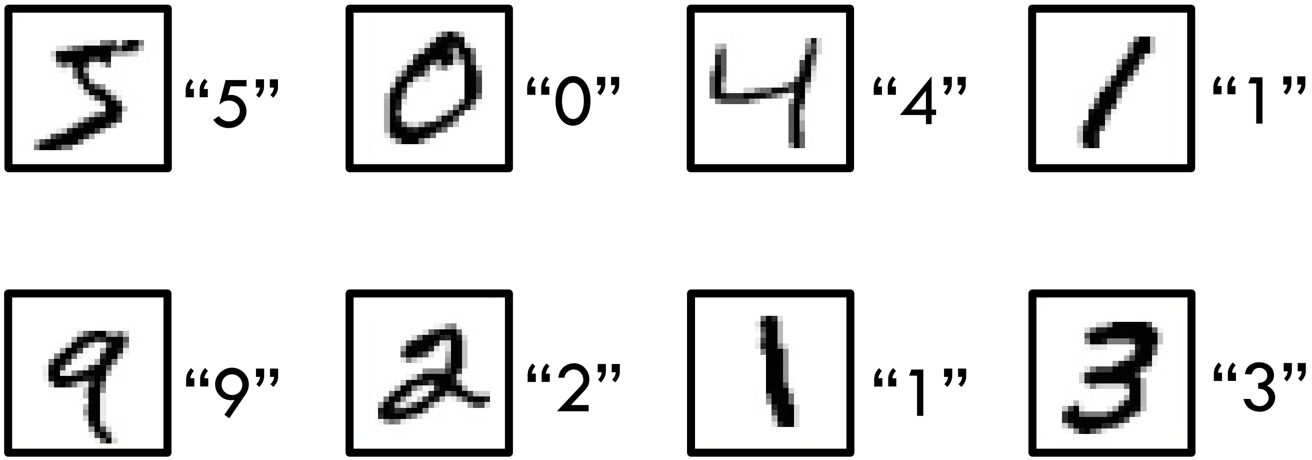
Set the network parameters θ such that ……

Input:  → y_1 has the maximum value

Input:  → y_2 has the maximum value

Training Data

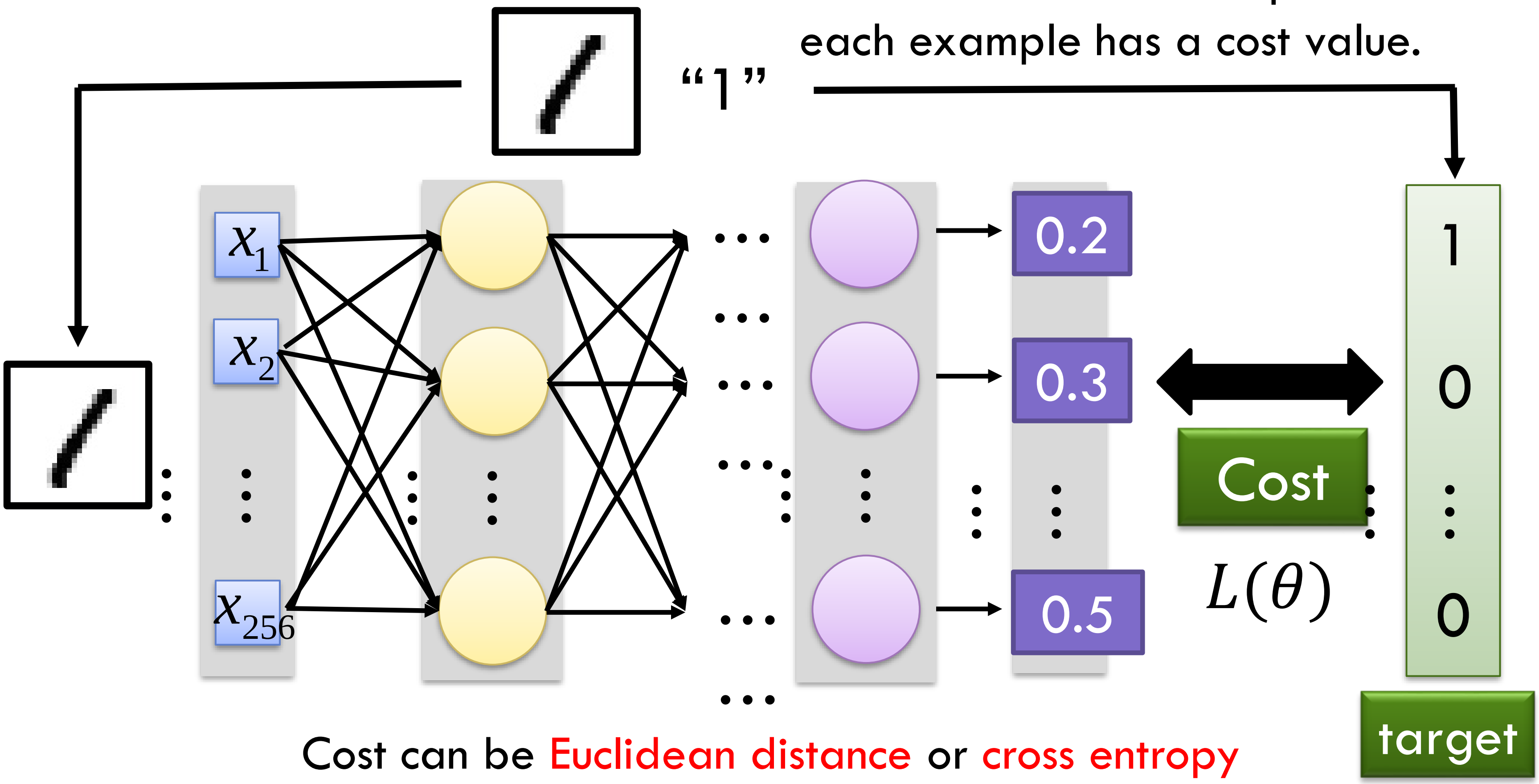
- Preparing training data: images and their labels



Using the training data to find the network parameters.

Cost

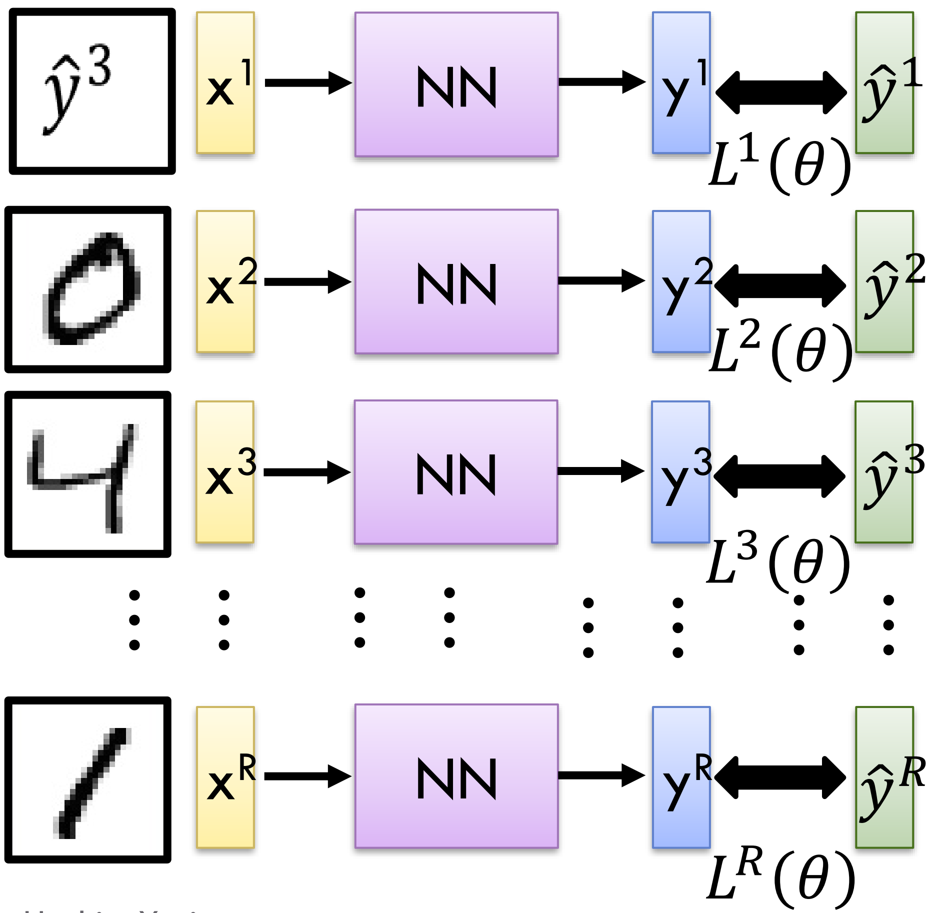
Given a set of network parameters θ, each example has a cost value.



Cost can be Euclidean distance or cross entropy of the network output and target value

Total Cost

For all training data …



Total Cost:

$$C(\theta) = \sum_{r=1}^{R} L^r(\theta)$$

How bad the network parameters θ is on this task

Find the network parameters θ^* that minimize this value

WHY DEEP LEARNING

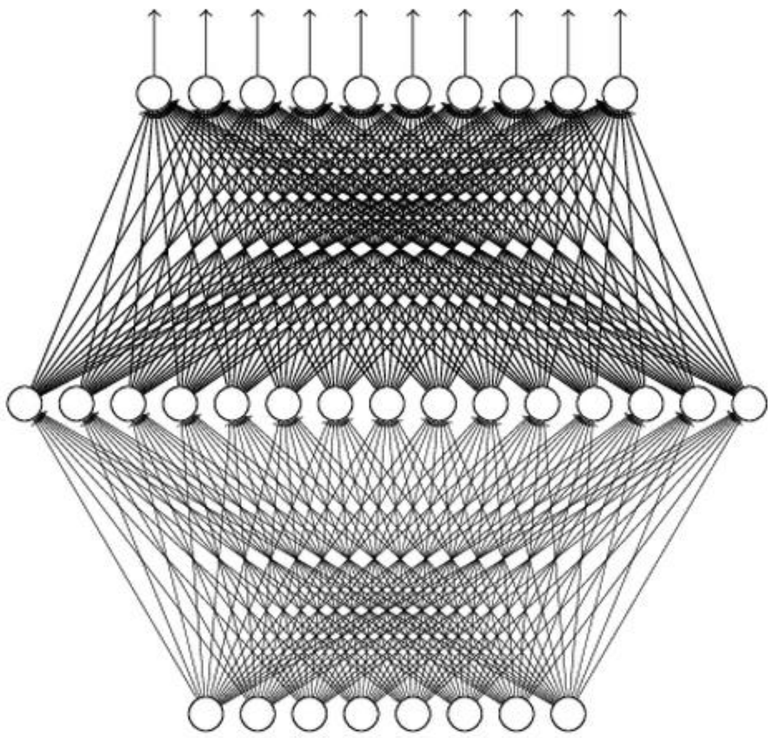
Why Deep Learning

Any continuous function f

$$f : R^N \rightarrow R^M$$

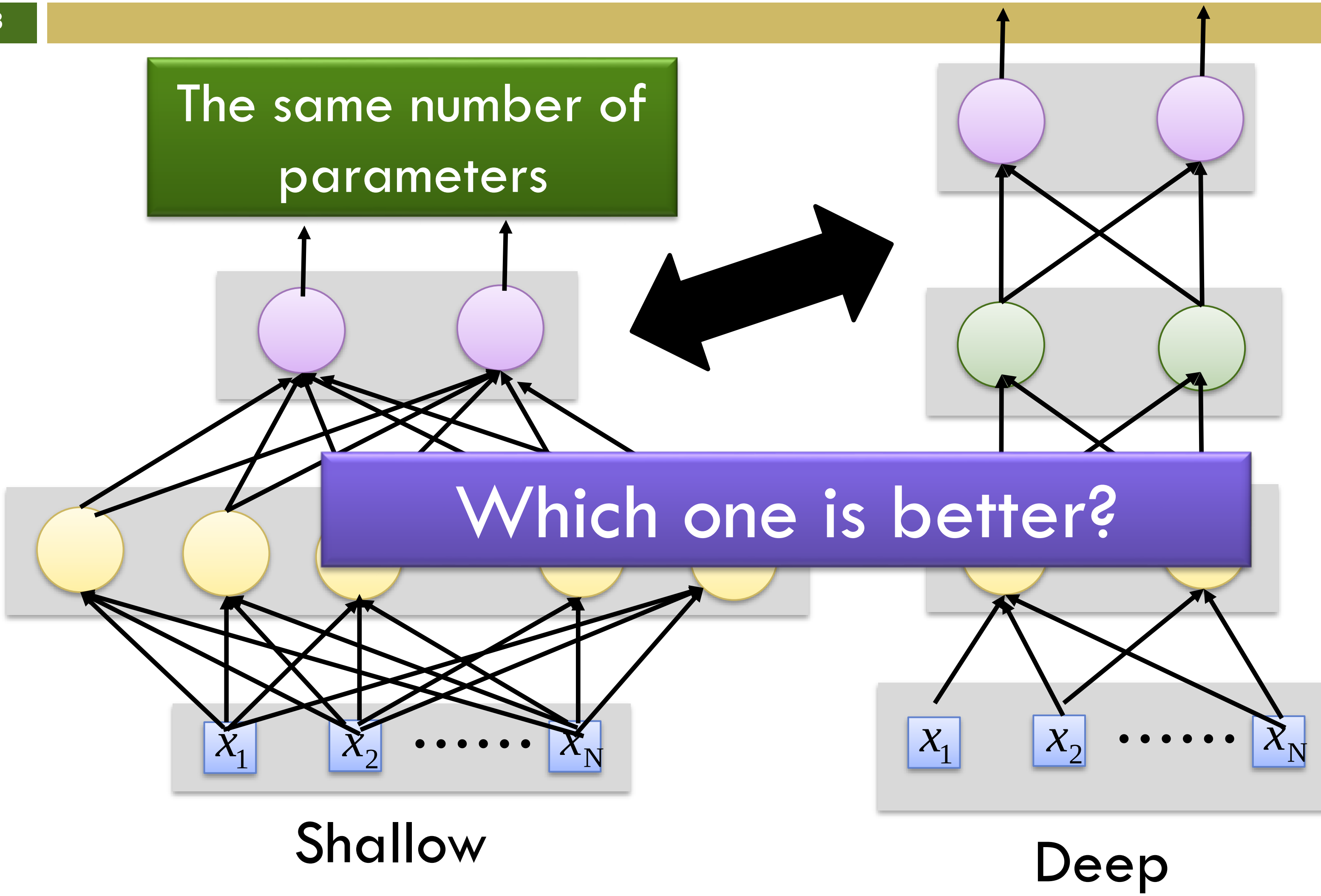
Can be realized by a network with one hidden layer

(given **enough** hidden neurons)



Why "Deep" neural network not "Fat" neural network?

Shallow vs Deep



Why Deep Learning

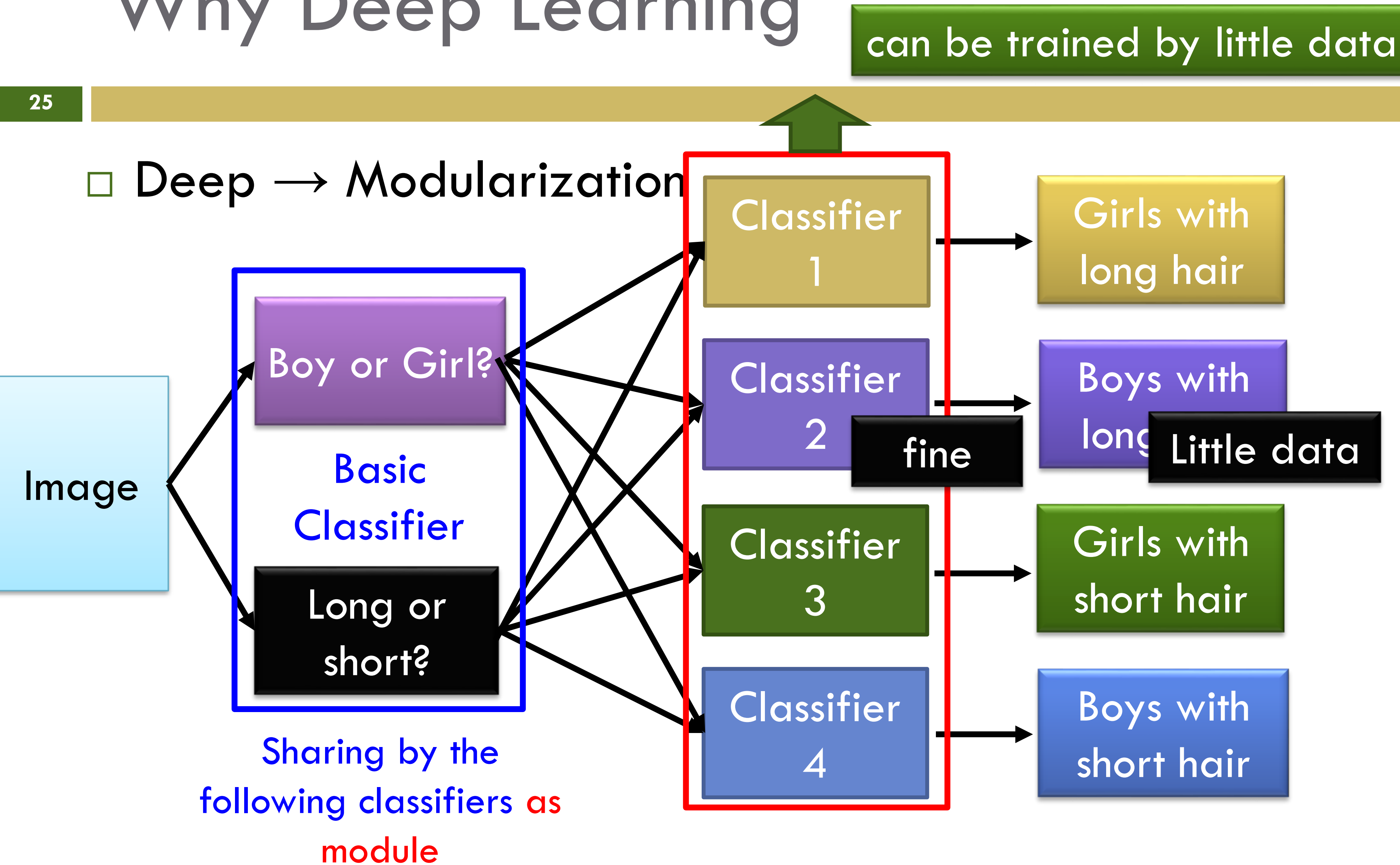
Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1
9 X 2k	17.0

Not surprised, more parameters, better performance

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

Why Deep Learning

- Deep → Modularization

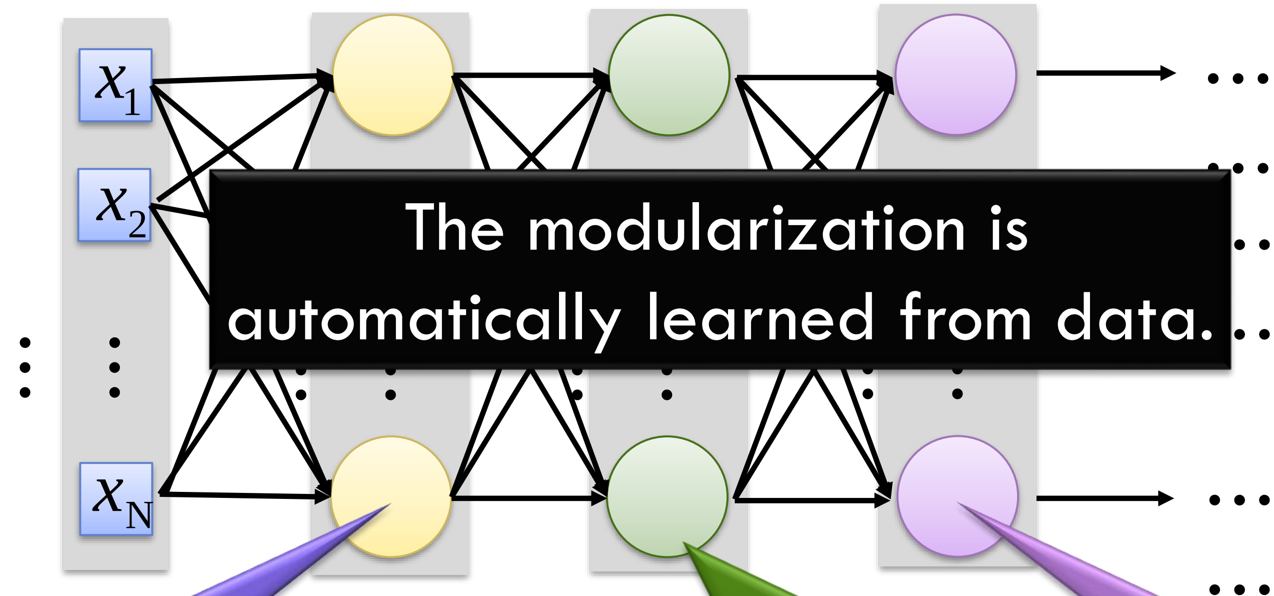


Why Deep Learning

Deep Learning also works on small data set

- Deep → Modularizatio

→ Less training data?



The modularization is automatically learned from data.

The most basic classifiers

Use 1st layer as module to build classifiers

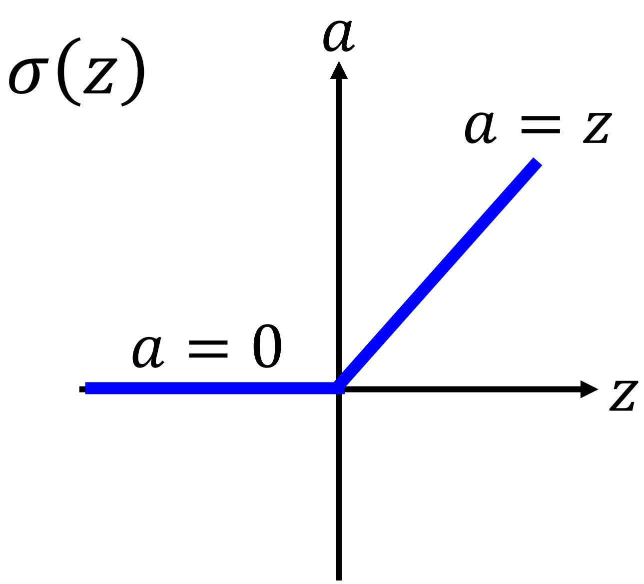
Use 2nd layer as module ……

MORE ACTIVATION FUNCTIONS

ReLU

- ***Rectified Linear Unit* (ReLU)**

$$f(x) = \max(0, x)$$



[Xavier Glorot, AISTATS'11]
[Andrew L. Maas, ICML'13]
[Kaiming He, arXiv'15]

Reason:

1. Fast to compute
2. one sided
3. Efficient gradient propagation *(accelerate (e.g. a factor of 6) the convergence of stochastic gradient descent compared to the sigmoid/tanh functions.)*
4. Scale-invariant
5. Sparse activation

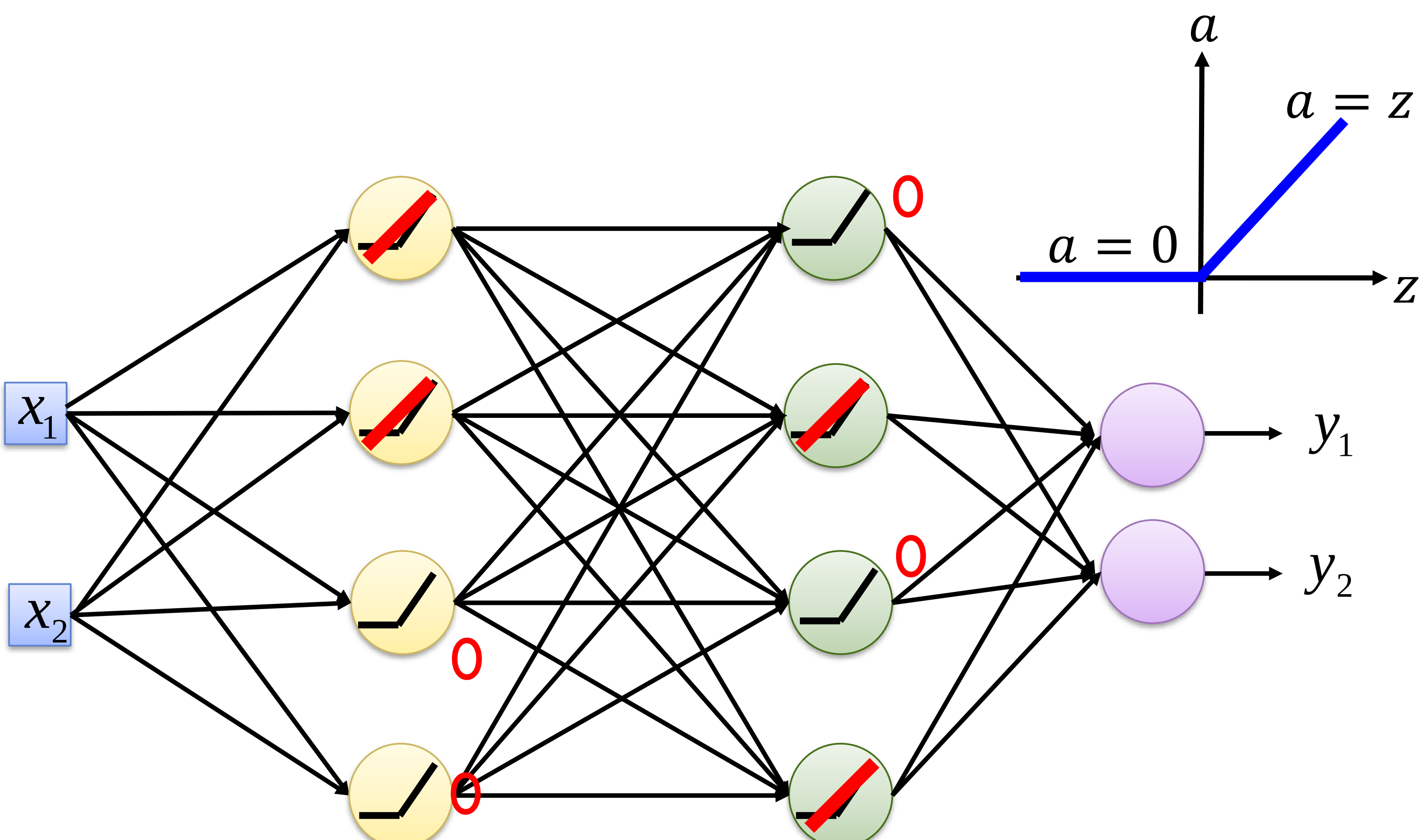
ReLU

- Non-linear in nature

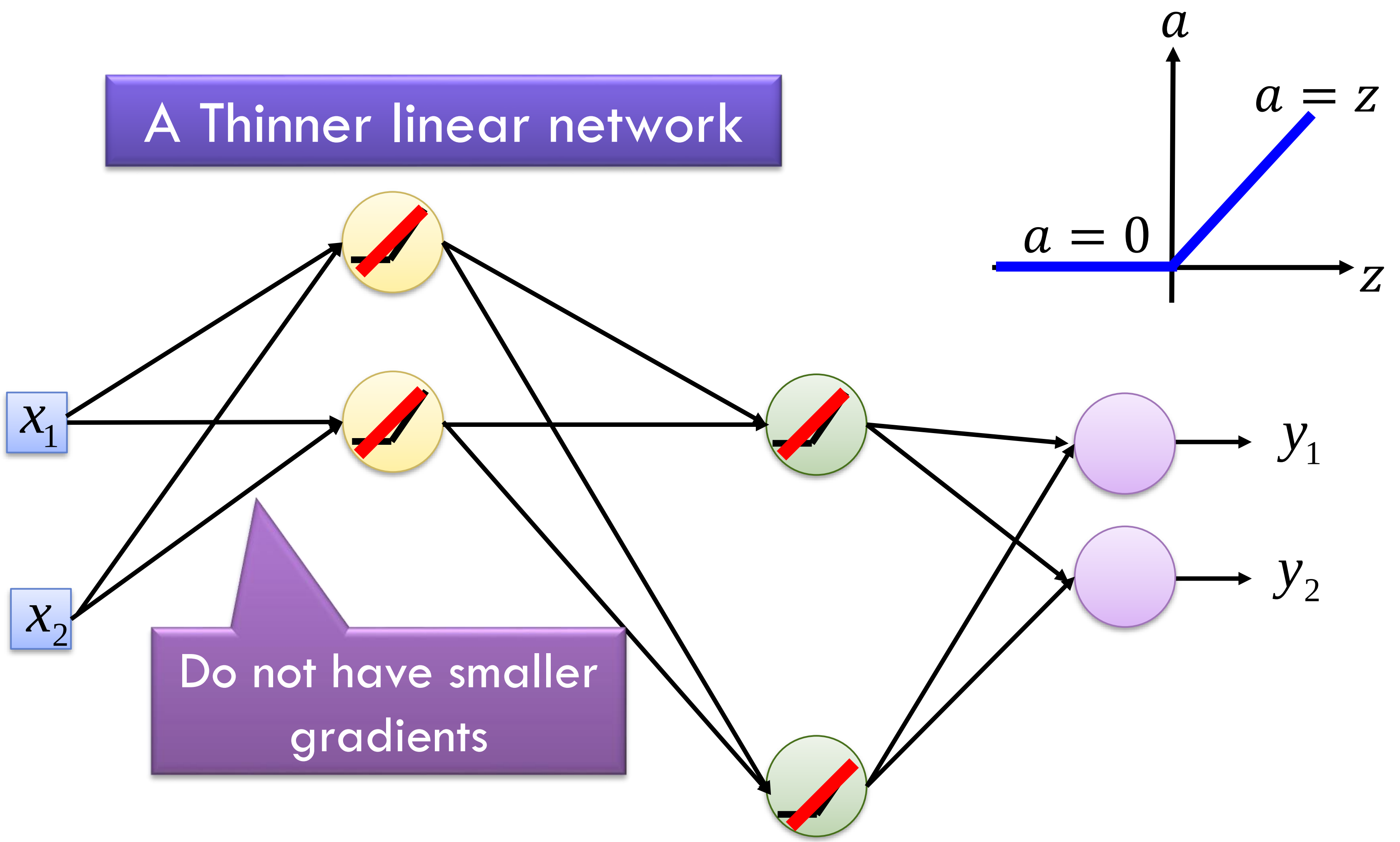
- The main advantage of using the ReLU function over other activation functions is that it does not activate all the neurons at the same time.

- If the *input is negative, it will convert it to zero and the neuron does not get activated*.
  - This means that at a time only a few neurons are activated making the network sparse, making it efficient and easy for computation.

ReLU



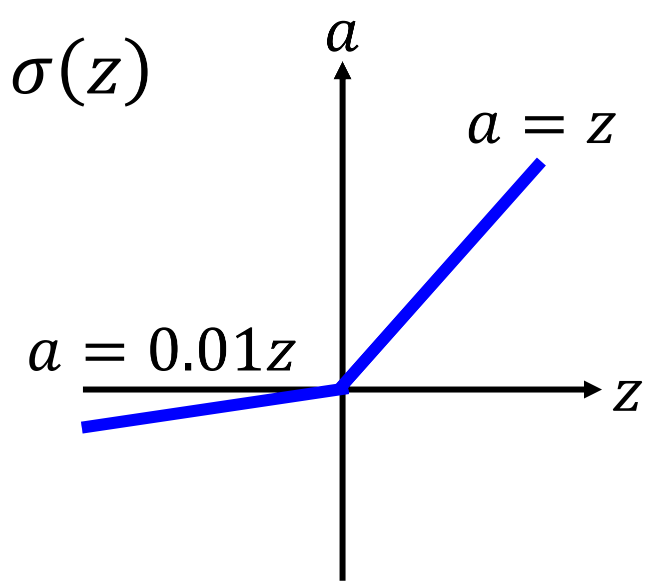
ReLU



A Thinner linear network
a
$a = z$
$a = 0$
z
x_1
x_2
y_1
y_2
Do not have smaller gradients

Leaky ReLU

- ***Leaky Rectified Linear Unit* (LReLU)**

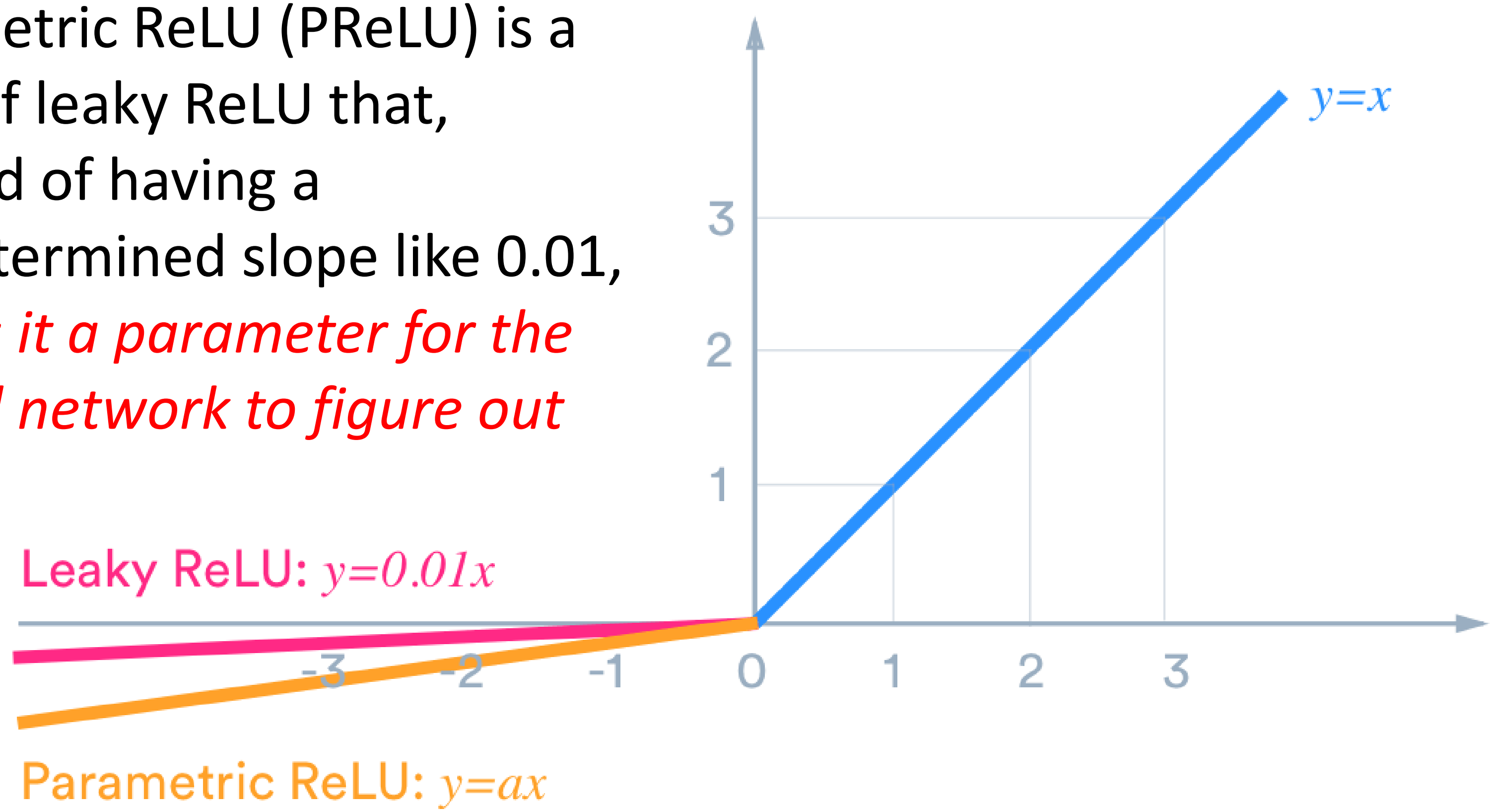


It fixes the *"dying ReLU"* problem, as it doesn't have zero-slope parts.

Parameterized ReLU

- ***Parameterized Rectified Linear Unit* (PReLU)**

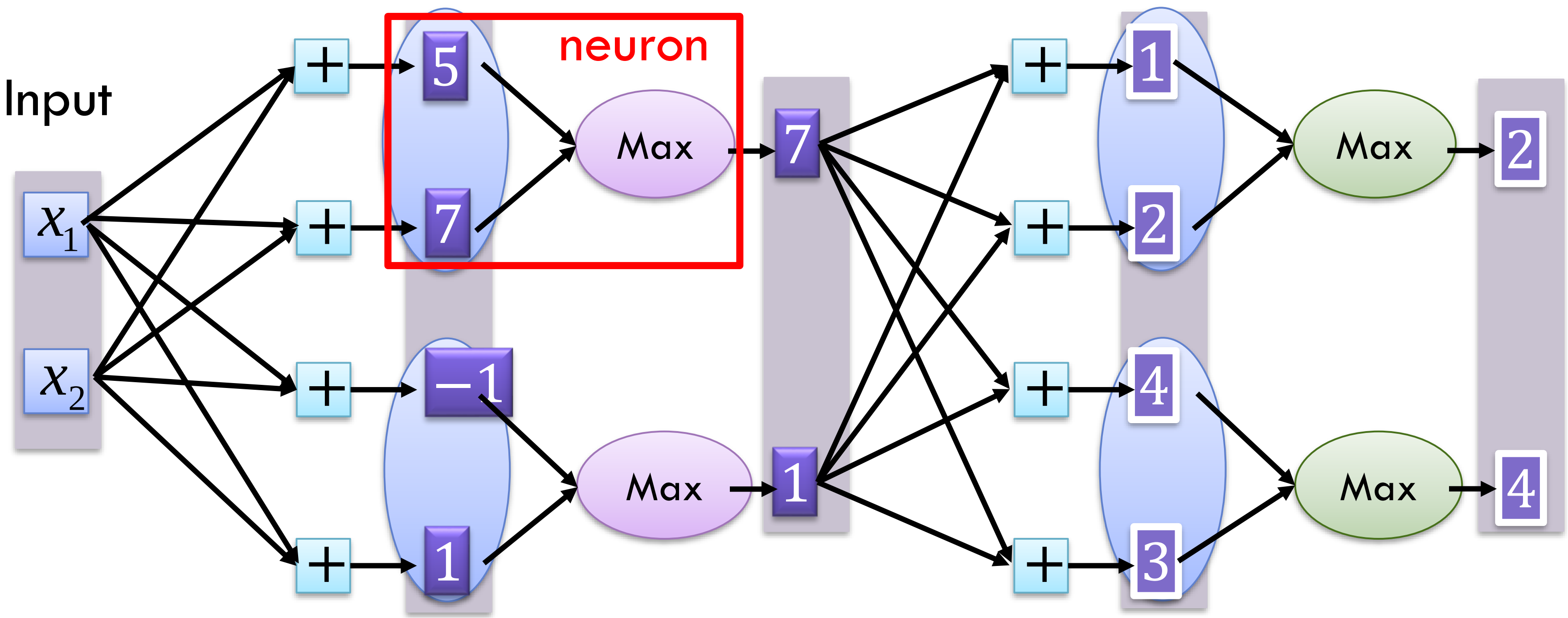
Parametric ReLU (PReLU) is a type of leaky ReLU that, instead of having a predetermined slope like 0.01, *makes it a parameter for the neural network to figure out itself*.



Maxout

ReLU is a special cases of Maxout

- Learnable activation function [Ian J. Goodfellow, ICML'13]



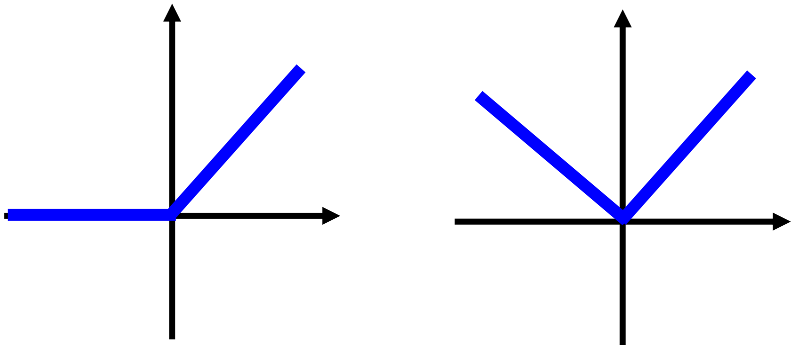
You can have more than 2 elements in a group.

Maxout

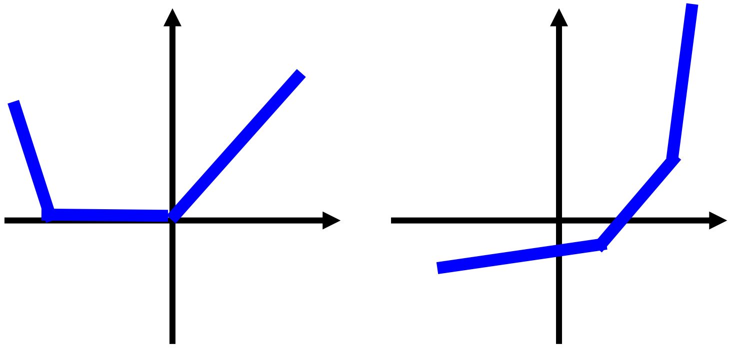
ReLU is a special cases of Maxout

- Learnable activation function
- Activation function in maxout network can be any ***piecewise linear convex function***
  - How many pieces depending on how many elements in a group

2 elements in a group



3 elements in a group



Choosing Activation Function

- Sigmoid functions and their combinations generally work better in the case of classifiers
- Sigmoids and tanh functions are sometimes avoided due to the vanishing gradient problem
- ReLU function is a general activation function and is used in most cases
- If we encounter a case of dead neurons in our networks the leaky ReLU function is the best choice

Choosing Activation Function

- ReLU function should only be used in the hidden layers
- As a rule of thumb, you can begin with using ReLU function and then move over to other activation functions in case ReLU doesn't provide optimum results
- Softmax function is used generally on output layer and for multi-label classification.

Acknowledgement

Stuart J. Russell and Peter Norvig, Tom M. Mitchell, Jiwon Jeong, Floydhub, Danqing Liu Andrej Karpathy, Hung-yi Lee