Applied Machine Learning with Big Data
"EE 6973"

Topic:
Deep Learning

Paul Rad, Ph.D.
Chief Research Officer
UTSA Open Cloud Institute(OCI)
University of Texas at San Antonio

Logistics

Instructor: Paul Rad

- Email: paul.rad@utsa.edu
- Office: OCI - NPB 3rd floor
- Office hours: Tuesday &Thursday 10:15:11:00 (30 minutes after our class)

Course Web: https://github.com/ml6973/Course

Mailing list: TBD

Course Social Network: TBD

Outline

Linear Algebra Review

Neural Network History

Neural Network Model

Neural Network Forward Propagation

Linear Algebra Review

1) Matrices and Vectors

2) Addition and Scalar Multiplication

3) Matrix Vector Multiplication

4) Matrix Matrix Multiplication

5) Matrix Multiplication Properties

6) Inverse and Transpose

<https://github.com/ml6973/Course/blob/master/code/Deep%20Learning%20-%20Linear%20Algebra%20Review.ipynb>

Neural Network History

Algorithms that try to mimic the brain

It was very widely used in 80s and early 90s.

Recently very popular again due to State-of-the-art technique for many applications.

Human brain: Massively parallel network

The basic computational unit of the brain is a **neuron**. Approximately **86 billion neurons** can be found in the human nervous system and they are connected with approximately 10^{14} - 10^{15} synapses

The Brain as a Universal Learning Machine

Two viewpoints on the Mind:

1) Evolutionary psychologists propose that the mind is made up of genetically influenced and domain-specific mental algorithms or computational modules, designed to solve specific evolutionary problems of the past

An evolutionary perspective leads one to view the mind as a crowded zoo of evolved, domain-specific programs. Each is functionally specialized for solving a different adaptive problem that arose during hominid evolutionary history, such as face recognition, foraging, mate choice, heart rate regulation, sleep management, or predator vigilance, and each is activated by a different set of cues from the environment.

There is another viewpoint cluster, more popular in computational neuroscience (especially today), that is almost the *exact opposite* of the evolved modularity hypothesis.

2) Universal learner" hypothesis, aka the "one learning algorithm" hypothesis

The universal learning hypothesis proposes that *all* significant mental algorithms are learned; nothing is innate except for the learning and reward machinery itself (which is somewhat complicated, involving a number of systems and mechanisms), the initial rough architecture (equivalent to a prior over mindspace), and a small library of simple innate circuits (analogous to the operating system layer in a computer). In this view the mind (software) is distinct from the brain (hardware). The mind is a complex software system built out of a general learning mechanism.

https://youtu.be/r9mvRRwu5Gw

What is deep learning useful for?

Like all machine learning: making predictions

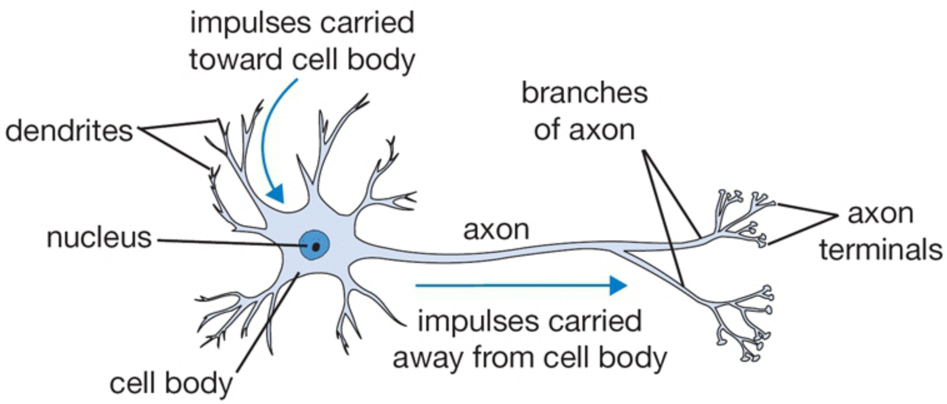
Deep learning does it with higher accuracy for some problems

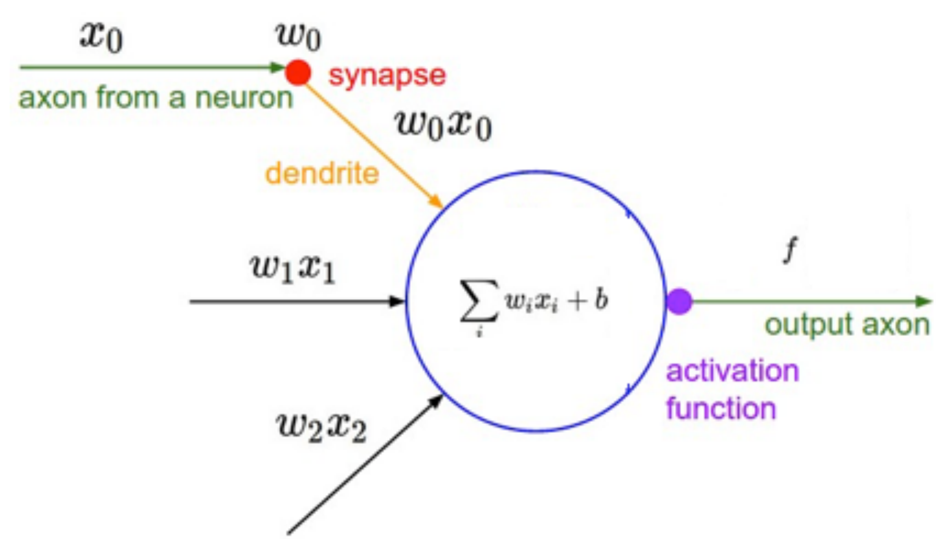
For example?

- Self-driving cars
- Stock market prediction
- Recognizing faces in an image

Under active development and research

Biological neuron and mathematical model



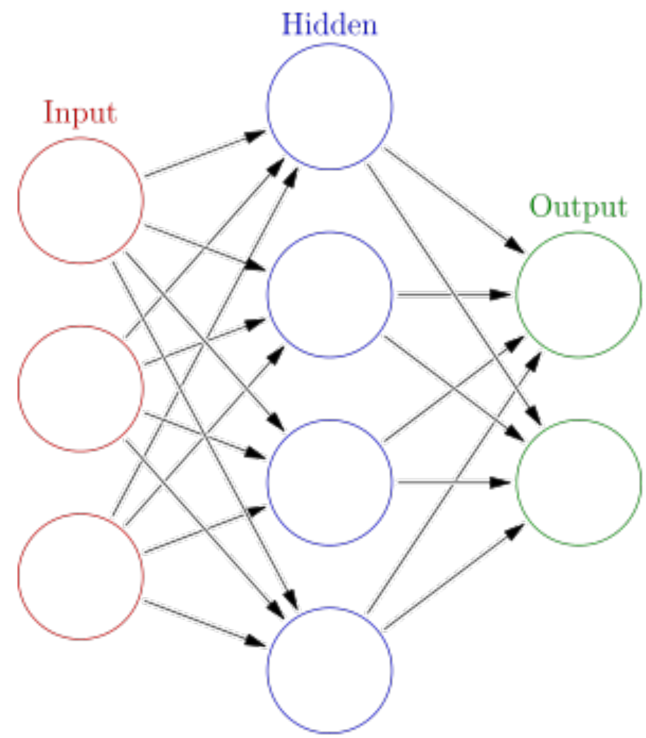


What Is A Neural Network?

The simplest definition of a neural network, more properly referred to as an 'artificial' neural network (ANN), is provided by the inventor of one of the first neurocomputers, Dr. Robert Hecht-Nielsen. He defines a neural network as:

"...a computing system made up of a number of simple, highly interconnected processing elements, which process information by their dynamic state response to external inputs.

In "Neural Network Primer: Part I" by Maureen Caudill, AI Expert, Feb. 1989



Every node in one layer is connected to every node in the next layer.

Signals get transmitted from the input, to the hidden layer, to the output

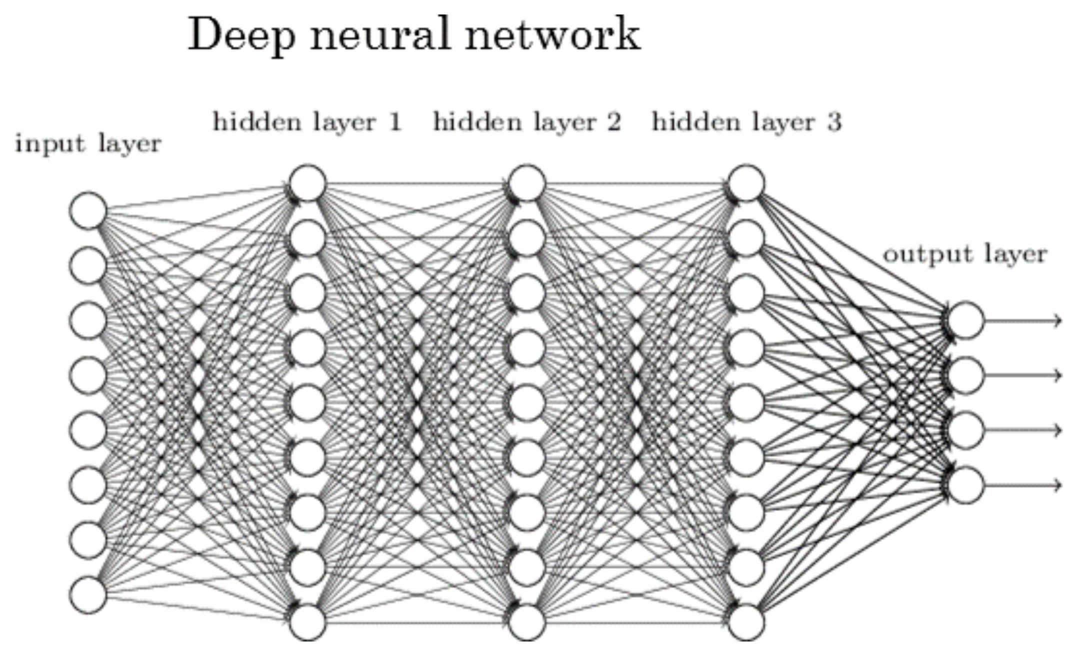
Deep Learning Architecture

An Artificial Neural Network with one or more hidden layers = Deep Learning

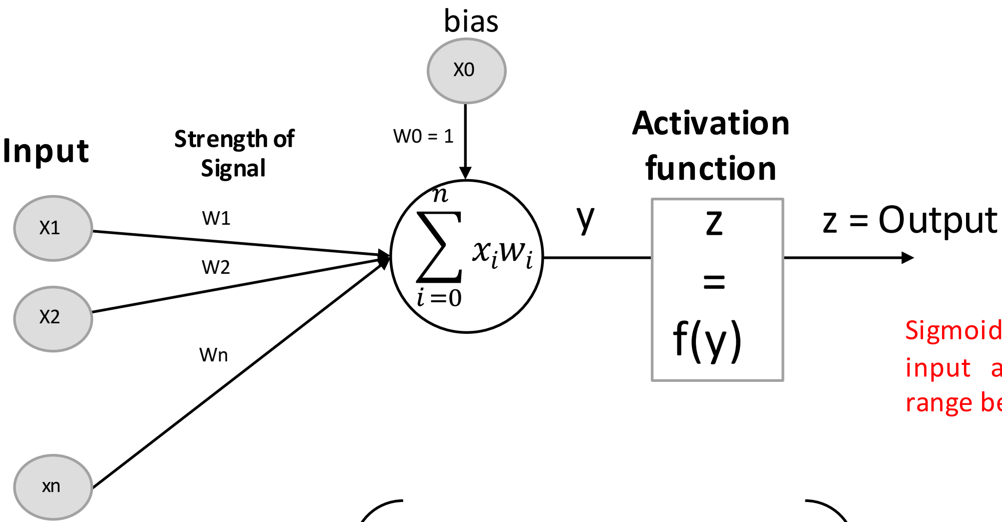
They typically consist of many hundreds of simple processing units which are wired together in a complex communication network.

Each unit or node is a simplified model of a real neuron which fires (sends off a new signal) if it receives a sufficiently strong input signal from the other nodes to which it is connected.

The output is aiming for a target.



Single Layer Neural Networks with Nonlinear Math Model

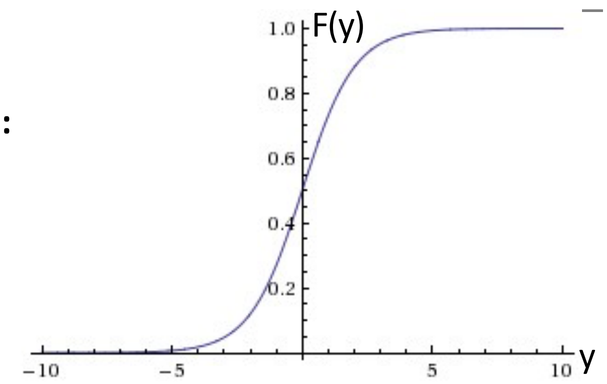


Sigmoid function:

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

$$(0, 1)$$

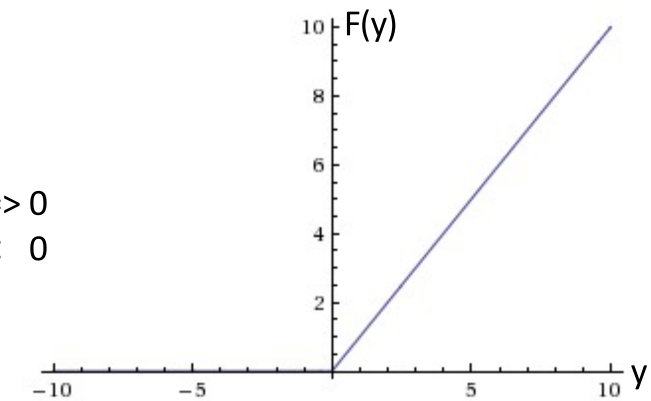
Sigmoid takes a real-valued input and squashes it to range between 0 and 1



$$\textbf{Output} = f\left(\begin{bmatrix} x0 & x1 & x2 & \dots & xn \end{bmatrix} \times \begin{bmatrix} 1 \\ w1 \\ w2 \\ \dots \\ wn \end{bmatrix} \right)$$

ReLU function:

$$f(y) = \begin{cases} y & y \geq 0 \\ 0 & y < 0 \end{cases}$$

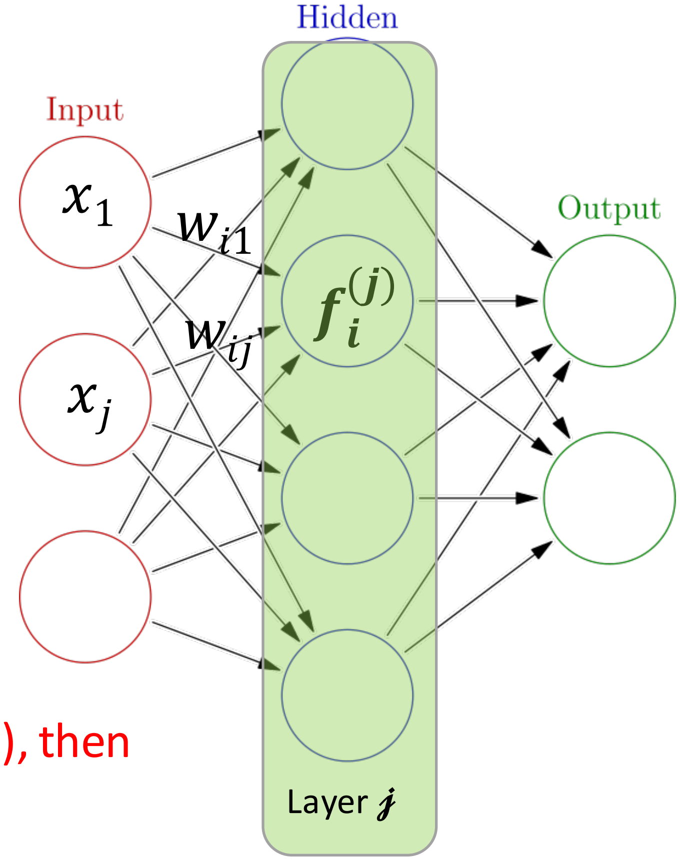


Multi Layer NN Nonlinear Math Model

$f_i^{(j)}$ = "activation function" of unit $\boldsymbol{i}$ in layer $\boldsymbol{j}$

W $^{(j)}$ = matrix of weights controlling function mapping from layer $\boldsymbol{j}$ to layer $\boldsymbol{j}$+1

$$y_i = f_i(\sum_{j=0}^{n} w_{ij} x_j)$$



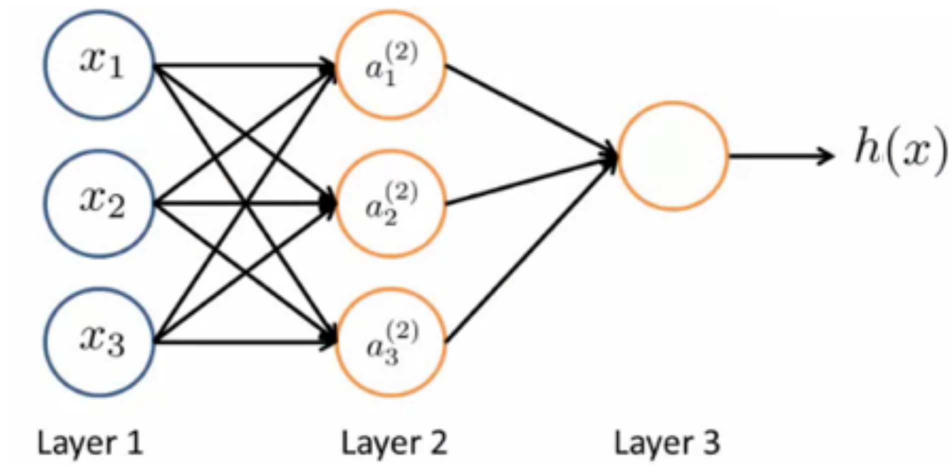
If network has N units in layer $\boldsymbol{j}$, and M units in layer ($\boldsymbol{j}$-1), then W $^{(j)}$ will be of dimension of (M+1) x N

Forward Propagation to Calculate h(x)

$$a_1{}^{(2)} = f\left(\sum_{j=0}^{n} w_{1j}\, x_j\right) =$$

$$a_2{}^{(2)} = f\left(\sum_{j=0}^{n} w_{2j}\, x_j\right) =$$

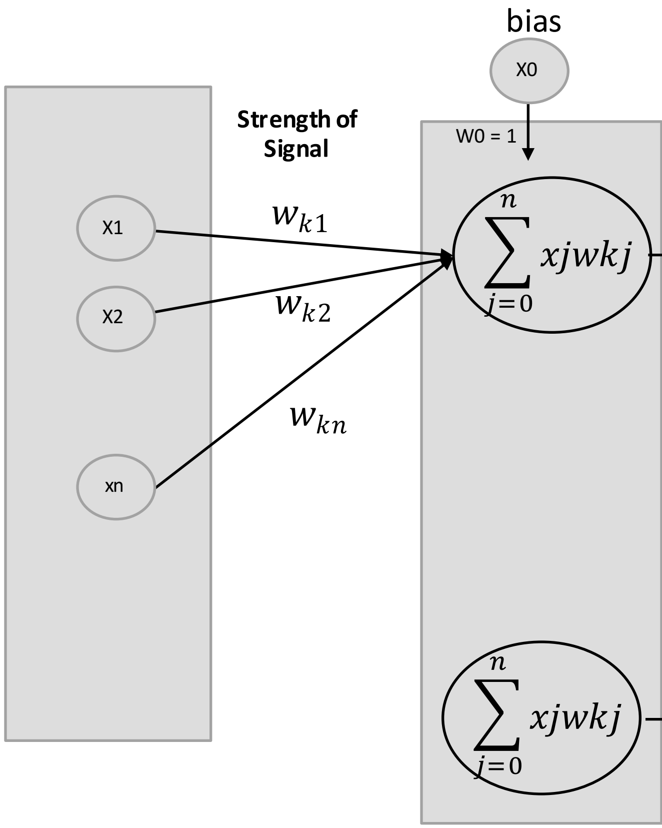
$$a_3{}^{(2)} = f\left(\sum_{j=0}^{n} w_{3j}\, x_j\right) =$$



$$a_i{}^{(2)} = f\left(\sum_{j=0}^{n} w_{ij}\, x_j\right)$$

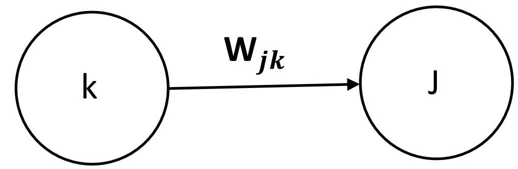
$$\boldsymbol{h\ (x) = f\ (a_3{}^{(2)} + a_2{}^{(2)} + a_1{}^{(2)} + a_0)}$$

Multi Layer NN Nonlinear Math Model



$$y_k = f_k\left(\sum_{j=0}^{n} x_j w_{kj}\right)$$

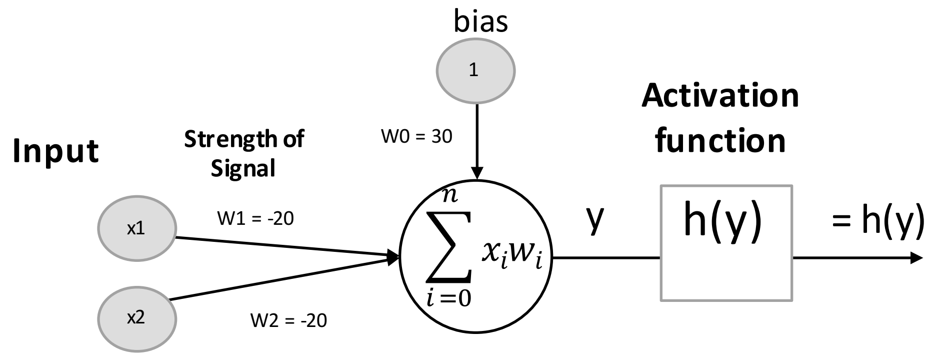
Connection weight from neuron in layer k to neuron in layer j = $\mathbf{W}_{jk}$



$$\begin{bmatrix} y_1 \\ y_2 \\ \\ y_k \end{bmatrix} = f\left(\begin{bmatrix} w_{k1} & w_{k2} & & w_{kj} \end{bmatrix} \right) \begin{bmatrix} x_1 \\ x_2 \\ \\ xj \end{bmatrix}$$

Simple example: OR

INPUTS		OUTPUTS					
A	B	AND	NAND	OR	NOR	EXOR	EXNOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1



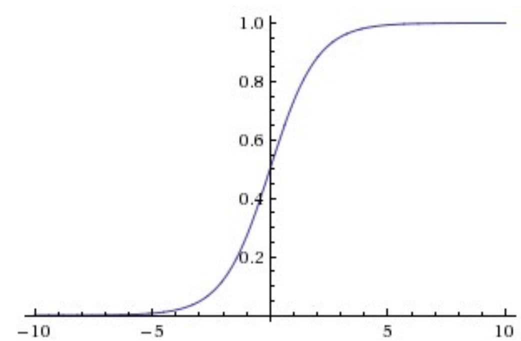
h(x) = Sigmoid (30 x 1 -20 x A - 20 x B)
h(x) = Sigmoid (30-20A-20B)

Verification:

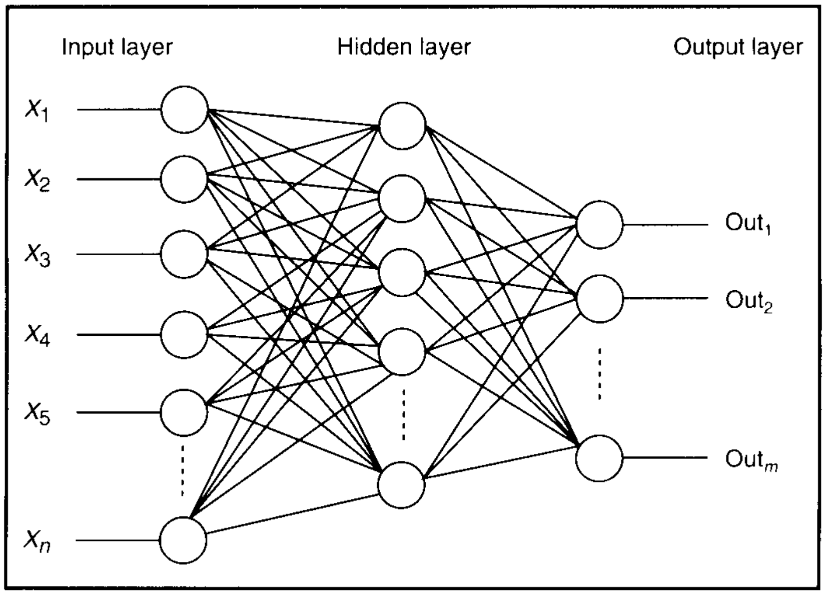
x = (A,B) = (0,0) → h(x) = Sigmoid (30) = 1
x = (A,B) = (0,1) → h(x) = Sigmoid (10) = 1
x = (A,B) = (1,0) → h(x) = Sigmoid (10) = 1
x = (A,B) = (1,1) → h(x) = Sigmoid (-10) = 0

Sigmoid function:

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



Example: Multi output units



0

1

…

9

$$h(x) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$h(x) = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

$$h(x) = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

References

How do I start using Deep Learning <http://deeplearning4j.org/deeplearningforbeginners.html>

High-level view of ANY supervised learning problem

All models (logistics regression, k-nearest neighbor, Naïve Bayes, SVM, decision tree, neural nets) have the same 2 functions:

Train () – learn model parameters (W) from the data

Predict () – make accurate predictions using the parameters (W) learned during training