AN INTRODUCTION TO NEURAL NETWORKS

CMP 414 & 765 Artificial Intelligence

Liang Zhao

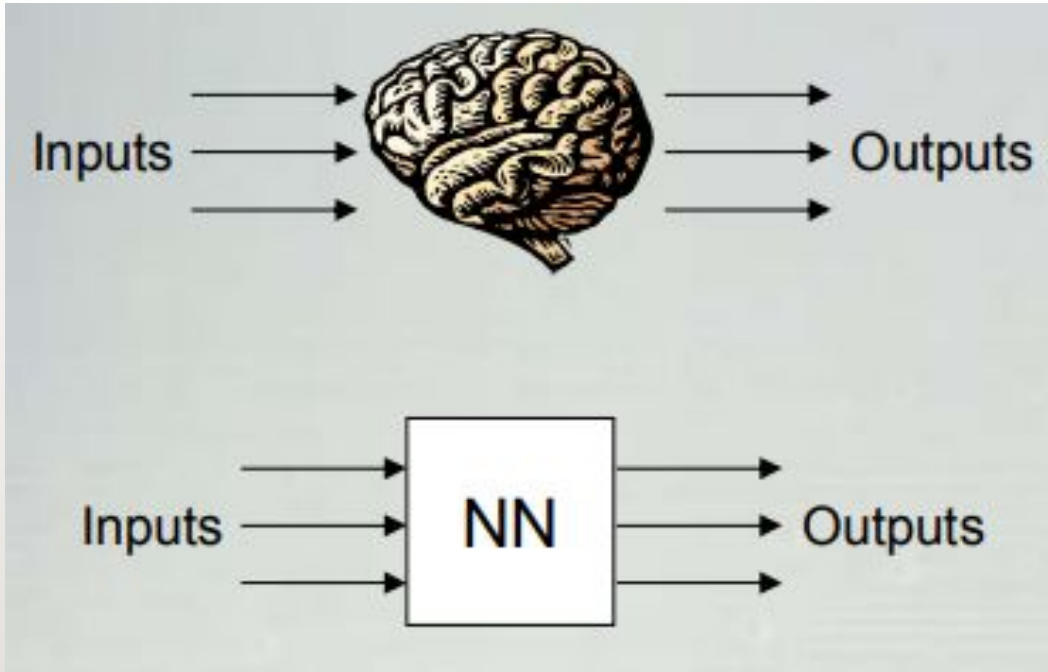
CUNY Lehman College

OUTLINE

- What are artificial neural networks?
- How to construct an artificial neural network?
- An XOR Example
- Training method: Backpropagation
- Discussion

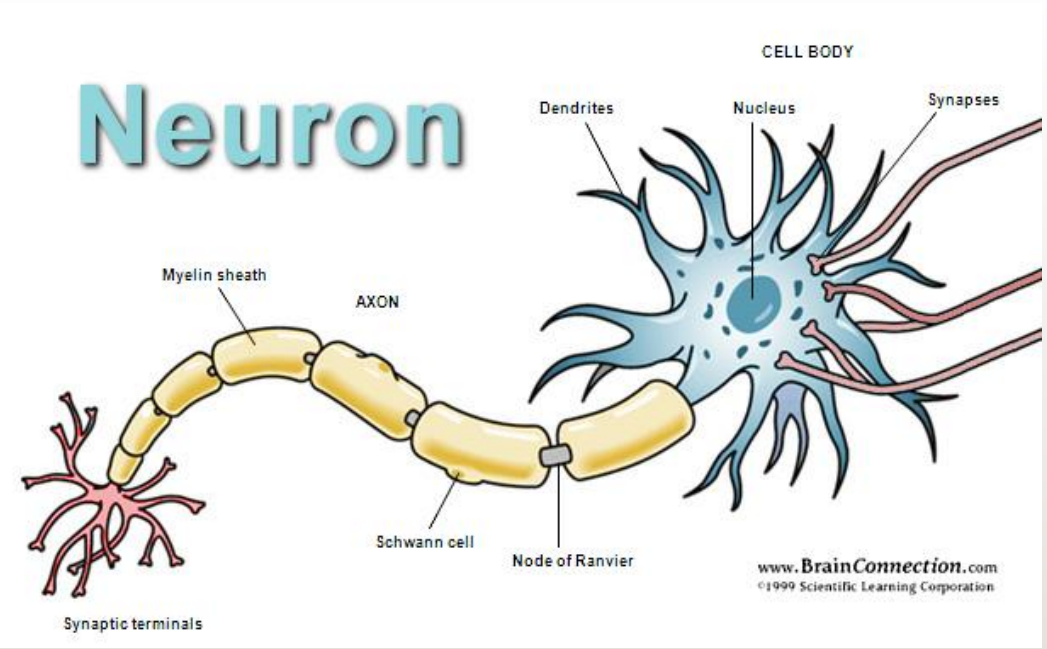
WHAT ARE ARTIFICIAL NEURAL NETWORKS?

- An extremely simplified model of the brain
- Transforms inputs into outputs to the best of its ability.



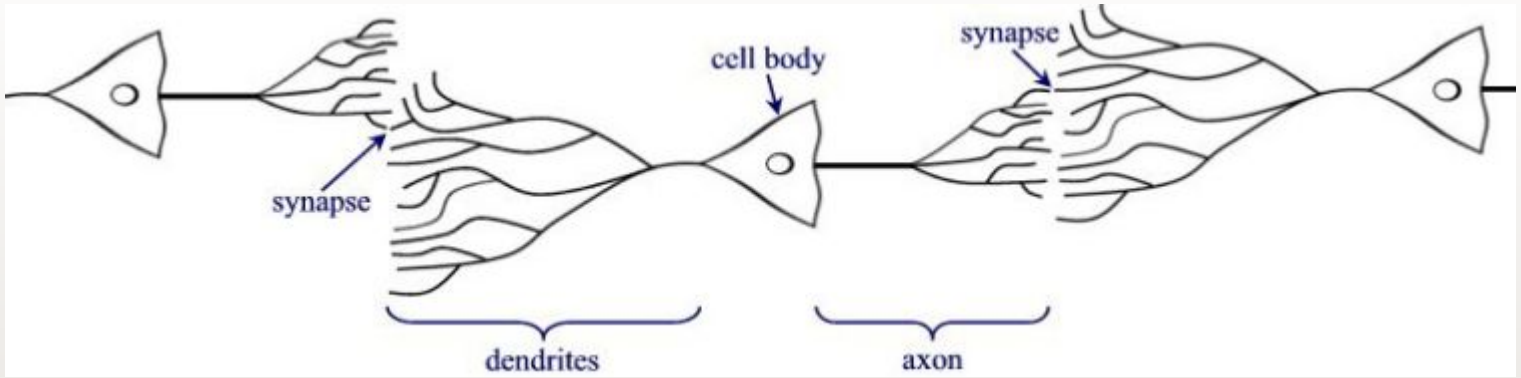
A NEURON

- Dendrites: receive impulses from other cells and transmit them to the cell body.
- Soma (or cell body): combines the impulses received and triggers action on the axons.
- Axons: the long thread-like part of a neuron that impulses are sent to other cells.



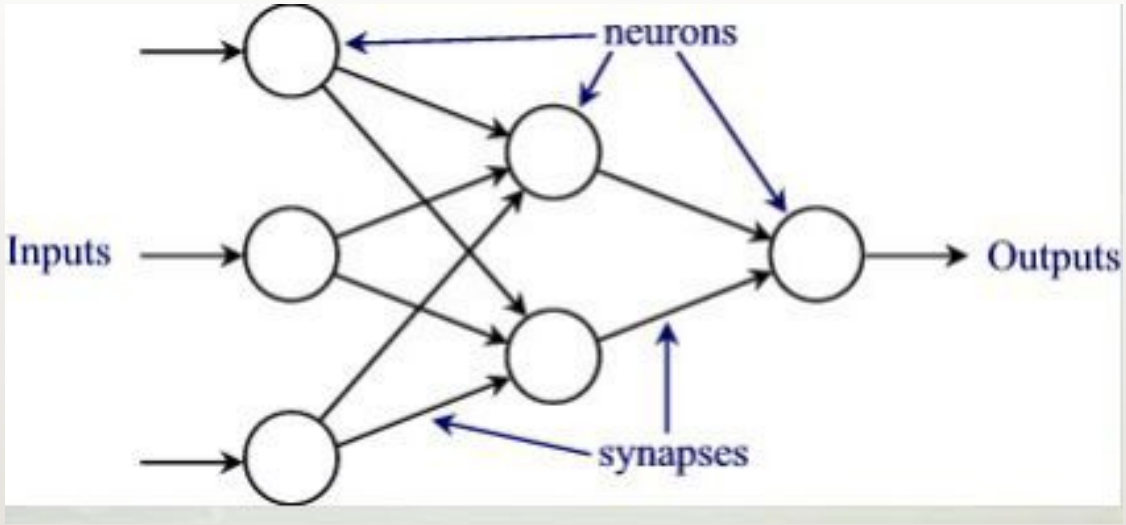
NEURON NETWORKS

- Composes of many neurons that cooperate to perform the desired function.



ARTIFICIAL NEURON NETWORKS (ANN)

- Composes of many neurons that cooperate to perform the desired function.



WHAT ARE NEURAL NETWORKS USED FOR?

- Image Processing
  - Image classification
  - Face recognition
  - Object detection
  - Image segmentation

Text to image

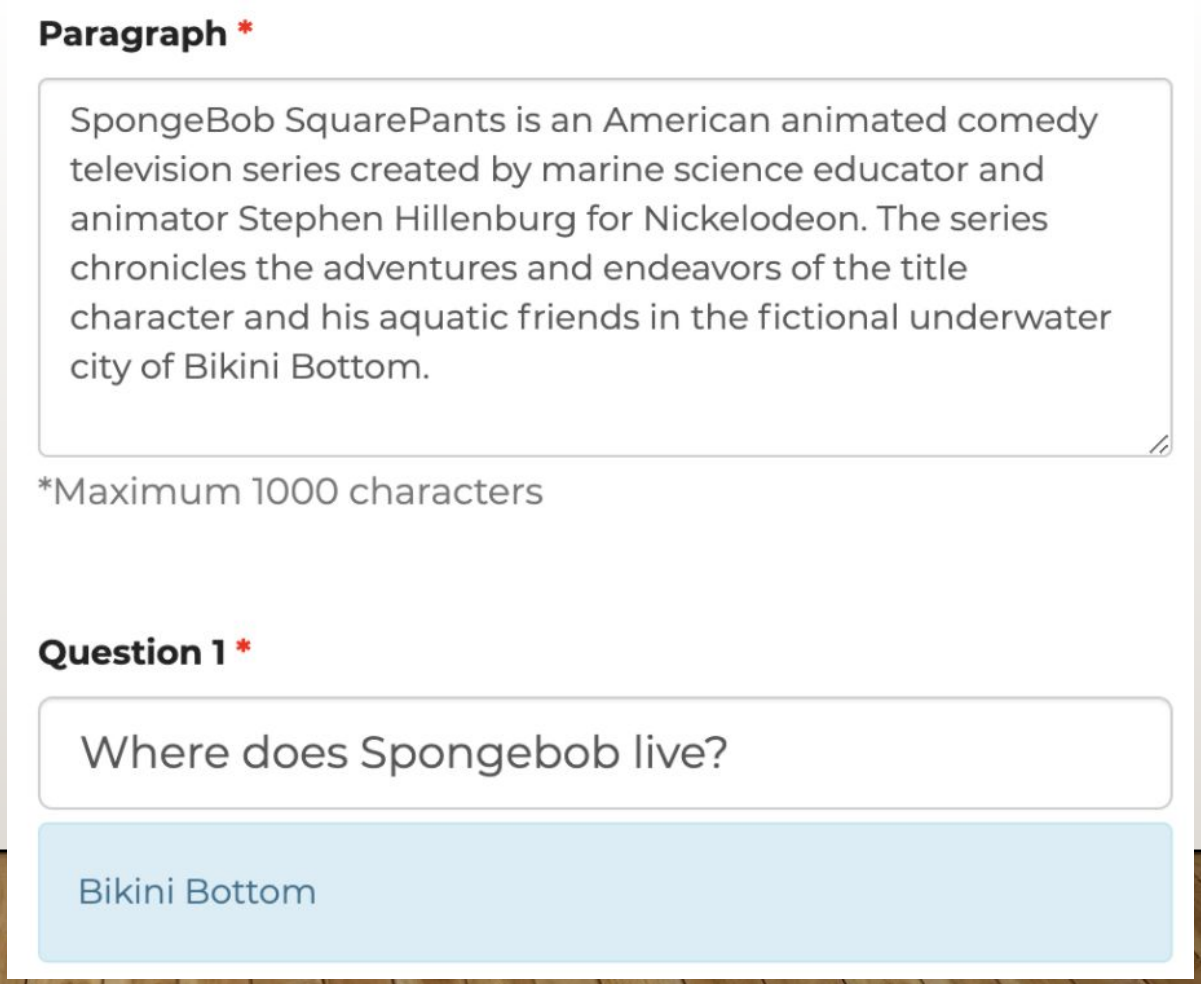
Artificially generated video

WHAT ARE NEURAL NETWORKS USED FOR?

Natural Language Processing

- Translation
- Sentiment analysis
- Text summarization

- Text Generation
- Machine Q & A



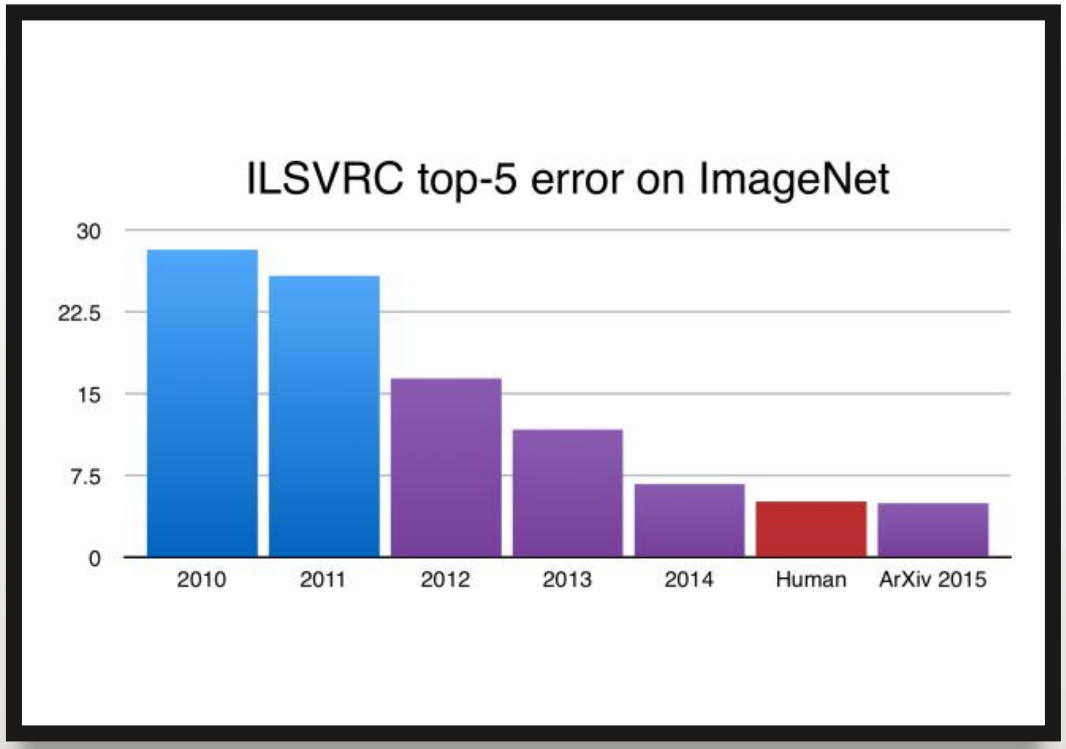
WHAT ARE NEURAL NETWORKS USED FOR?

Art
Music
Game

[AI Generated Rock Music]
[AI plays Mario]

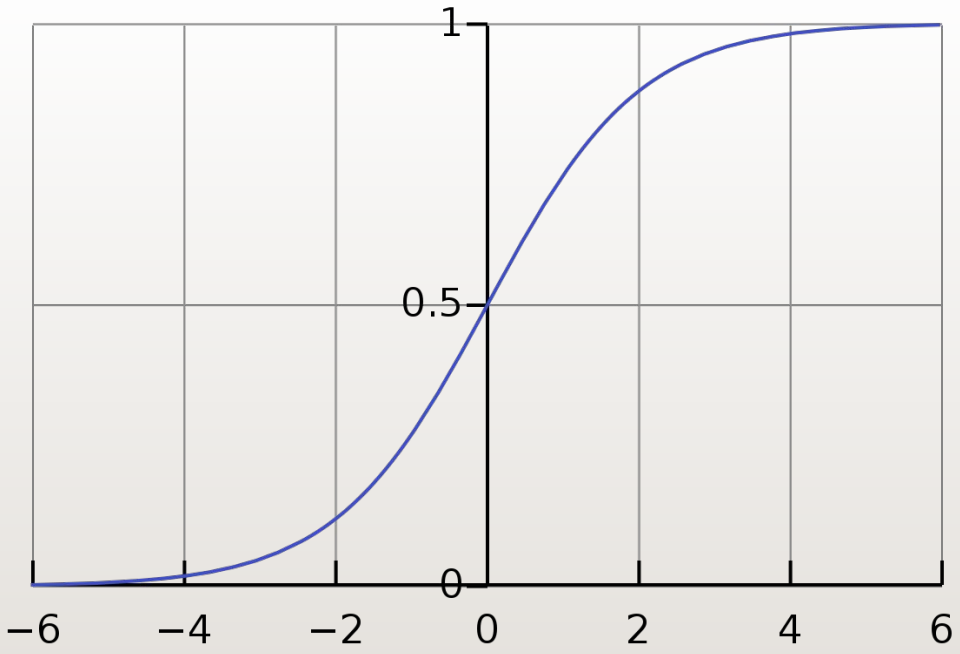
WHY USE NEURAL NETWORKS?

- Universal approximation property (Cybenko, 1989)
  - Any continuous function or indicator function can be approximated arbitrarily well by a neural network model.
- State-of-the-art results in many important applications



ARTIFICIAL NEURON: ACTIVATION FUNCTIONS

- Applied to the weighted sum of the inputs to produce the output.
- Choice 1: logistic function (sigmoid function)
  - Smooth, continuous, and monotonically increasing. (What's its derivative?)
  - Two asymptotes: y=0 and y=1.
  - Function values are very close to an asymptote for x << 0 and x >> 0.

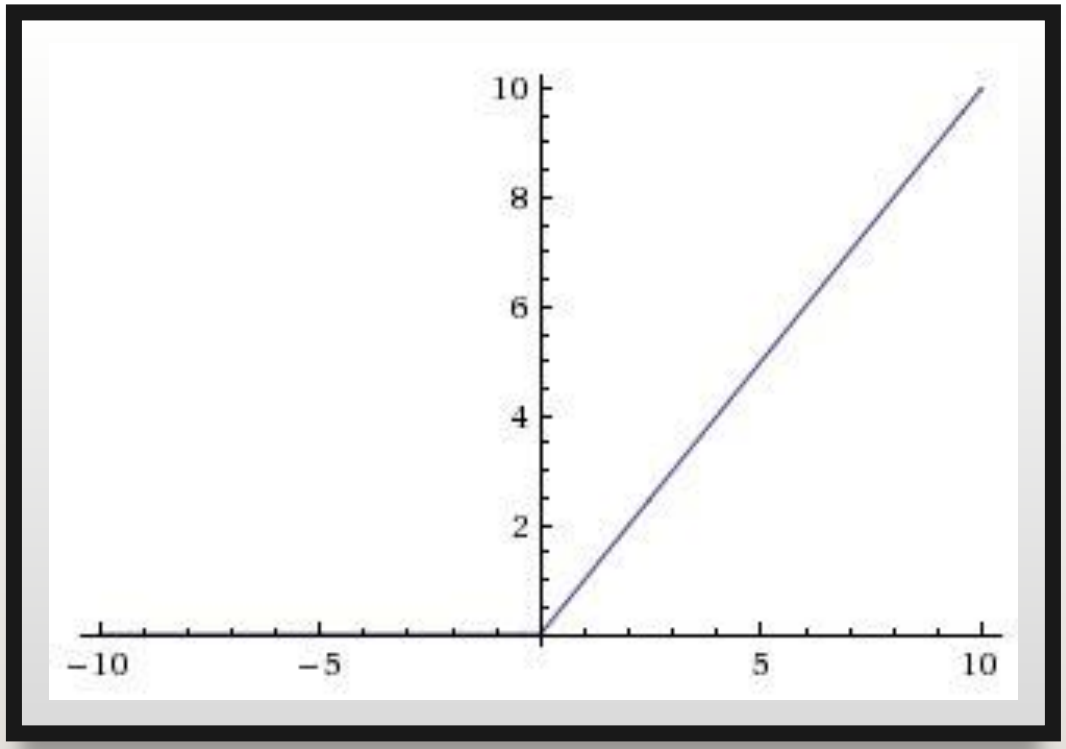


ACTIVATION FUNCTIONS

- Choice 2: Rectified Linear Unit (ReLU)

ReLU(t) = max(0, t)

- Very simple to evaluate
- Derivative exists everywhere except for 0
- Derivative is either 0 or 1

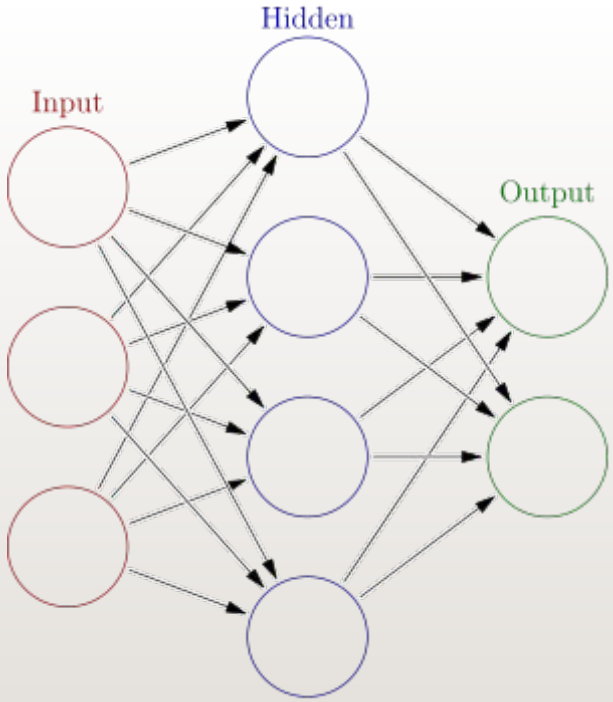


HOW TO CONSTRUCT A NEURAL NETWORK?

- Input layer: for each input feature, create a node to read its value.
- First hidden layer: create n_1 nodes in this layer. Number of nodes is determined by the researcher.
  - Input of each node: a weighted sum of previous nodes plus a bias.
  - Activation function: based on the weighted sum, decide whether to trigger reaction.
  - Output: evaluation of the activation function
- Second hidden layer, ...
- Output layer: create n2 nodes in this layers, one node for each output feature.
  - Input: a weighted sum of previous node outputs plus a bias.
  - Transformation (optional): convert the values to a proper model output.

HOW TO CONSTRUCT A NEURAL NETWORK?

- Circles: nodes or neurons that represent activation functions
- Columns of circles: layers of a neural network
- Arrows: inputs of each node (a weight is attached to each arrow)



EXAMPLE: XOR FUNCTION

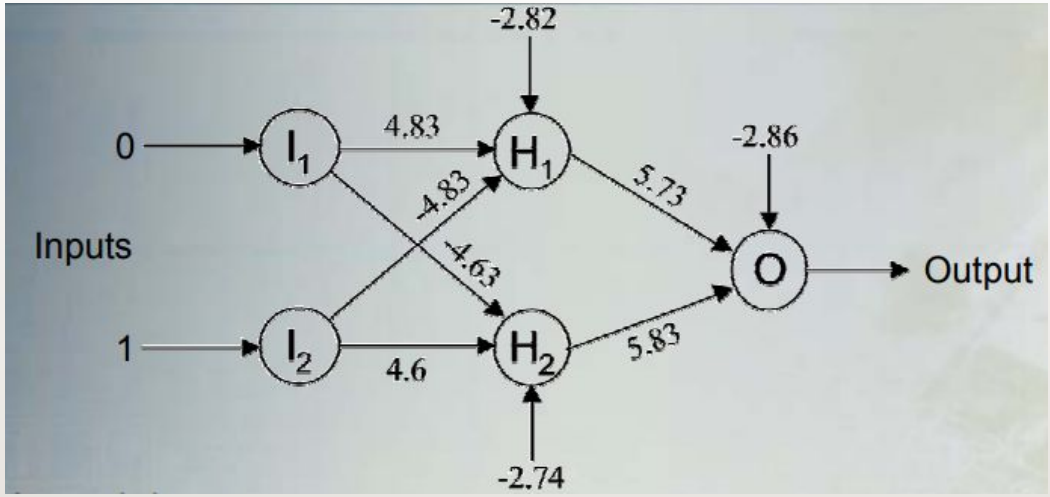
- Exclusive or (XOR) function is a logical operation that outputs true only when inputs differ (one is ture, other is false)

XOR truth table

Input		Output
A	B	
0	0	0
0	I	I
I	0	I
I	I	0

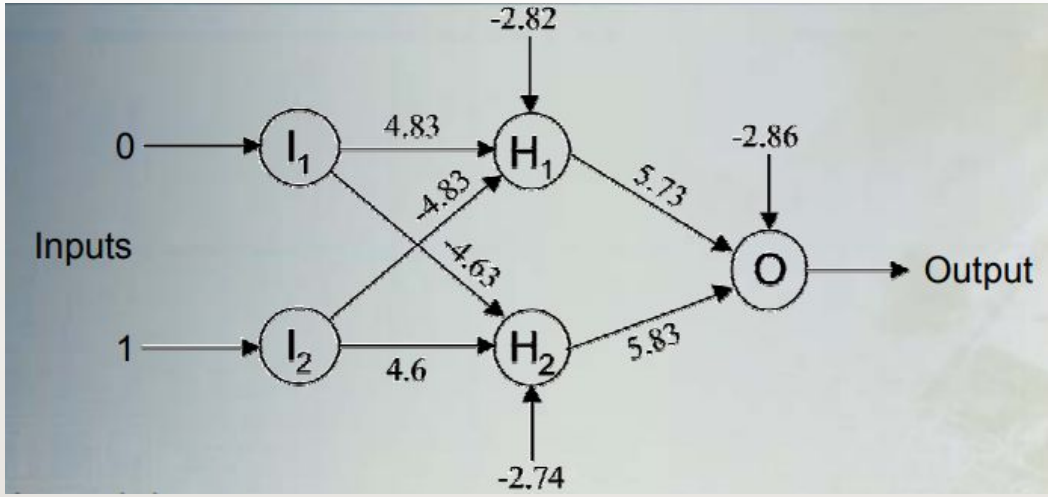
EXAMPLE: XOR FUNCTION

- Inputs: 0, 1
- Node H_1:
- $Weighted\ sum\ = 0 \times 4.83 + 1 \times (-4.83) - 2.82 = -7.65$
- $Output = \frac{1}{1+e^{7.65}} = 4.758 \times 10^{-4}$



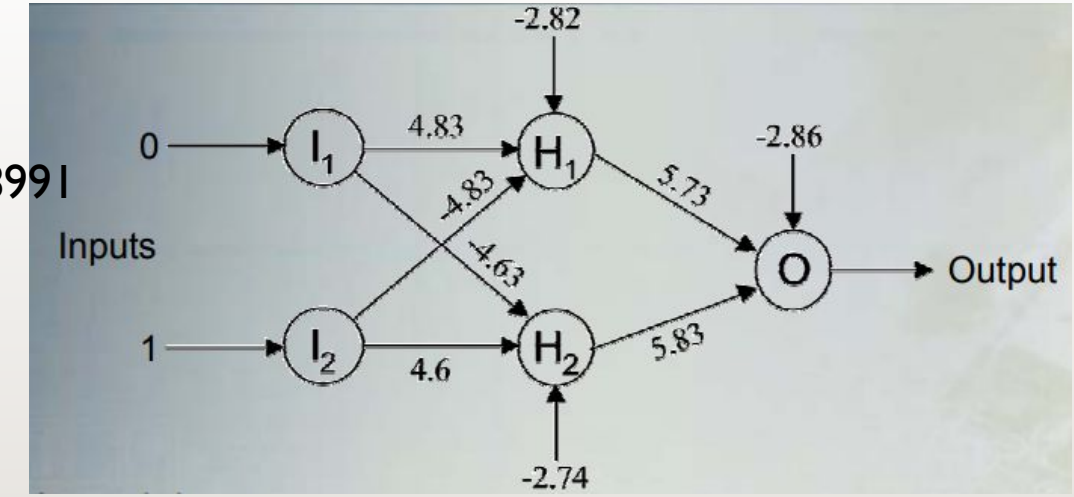
EXAMPLE: XOR FUNCTION

- Inputs: 0, 1
- Node H_2:
- $Weighted\ sum\ = 0 \times (-4.63) + 1 \times 4.6\ - 2.74 = 1.86$
- $Output = \frac{1}{1+e^{-1.86}} = 0.8652$



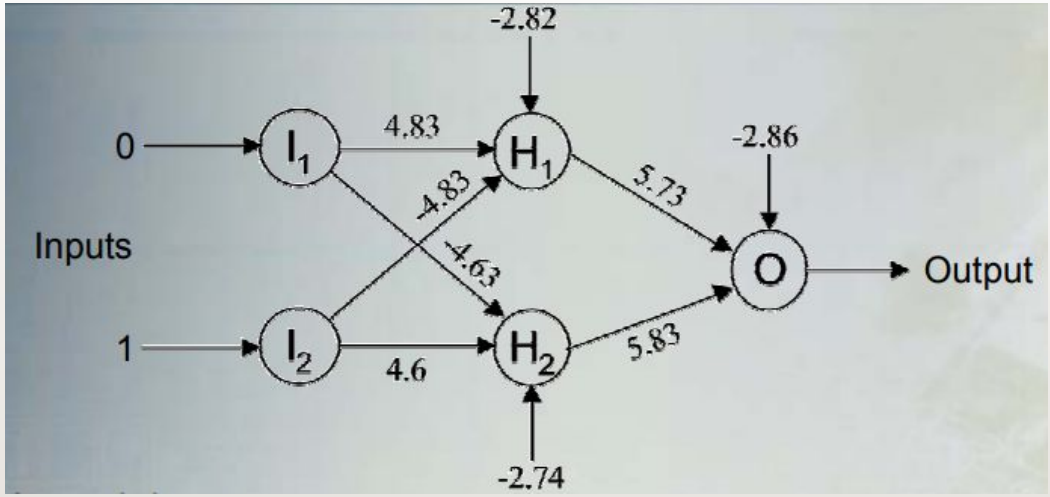
EXAMPLE: XOR FUNCTION

- Inputs: 0, 1
- Node O:
- $Weighted\ sum\ = 4.758 \times 10^{-4} \times$ 0.8991
  $5.73 + 0.8652 \times 5.83\ - 2.86 = 2.187$
- $Output = \frac{1}{1+e^{-2.187}} = 0.8991 \rightarrow$ "1"



EXAMPLE: XOR FUNCTION

- Inputs: 1, 0 ?



TRAINING A NEURAL NETWORK

- Training set $\{(x^{(i)}, y^{(i)}) | i = 1,2, \dots, m\}$
- Model prediction for $x^{(i)}$ is $\tilde{y}^{(i)}$.
- Put a (square) cost to each prediction error: $cost = (\tilde{y}^{(i)} - y^{(i)})^2$
- Goal: find proper weights that minimize the average cost: $\mathrm{L}(\text{weights}) = \frac{1}{m}\sum_i (\tilde{y}^{(i)} - y^{(i)})^2$
- Optimization method: (Stochastic) Gradient Descent
- Challenge: How to compute all the gradients?

BACKPROPAGATION

- Short for "backward propagation of errors"
- An algorithm for gradient descent of artificial neural networks
- Calculates the gradient (all partial derivatives) of the cost function with respect to the weights.
- Based on the chain rule for differentiating composition of functions, the algorithm computes partial derivatives of weights in the last layer first, then it computes partial derivatives for the second-to-last layer, and so on. Hence it is called "backpropagation".

DISCUSSION: WHY USE NEURAL NETWORKS?

- NNs outperforms other ML techniques on large and complex problems.
- The tremendous increase in computing power makes it possible to train large neural networks in a reasonable amount of time.
- The training algorithms have been improved.
- Some theoretical limitations of NNs have turned out to be benign in practice.

WHY NOT USE NEURAL NETWORKS?

- Huge number of parameters: slow to use, prone to overfitting.
- Black box model: hard to understand the learned function
- High Variance and lack of performance guarantee

Neural Network Playground

Please go to the neural network playground, and use the tools to construct a simple neural network.

Example: RELU Activation

The figure on the right shows a neural network with one hidden layer with ReLU as activation function.

Given input (1, 1) and weights of the network, compute the input and output of each neuron in the hidden layer and the output layer (assume that all biases are zero).