

# Convolutional Neural Networks I: Activation Function and FC Layer

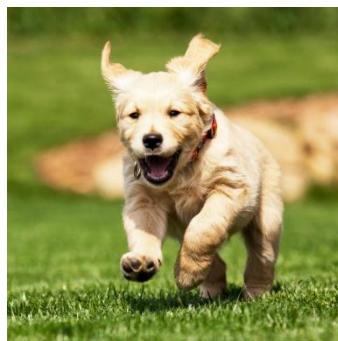
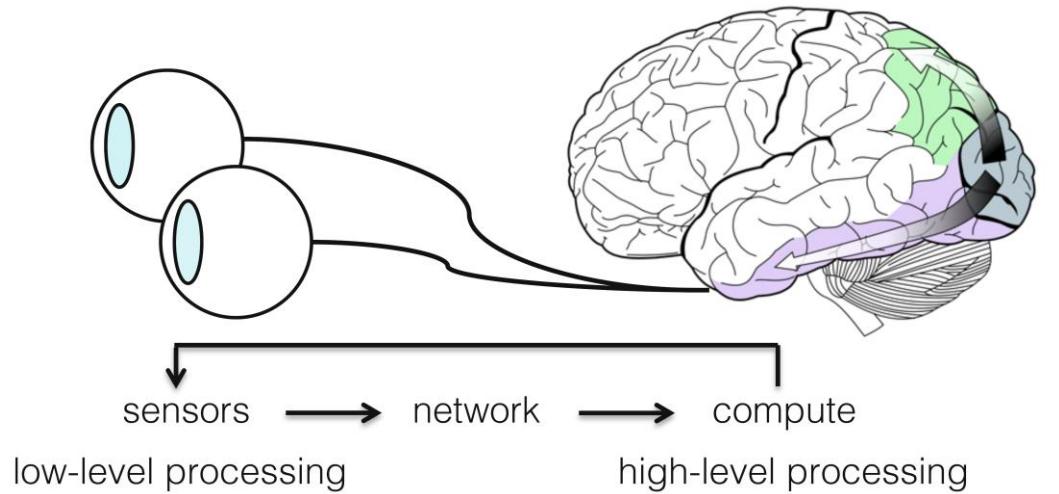
CS 4391 Introduction Computer Vision

Instructor Yu Xiang

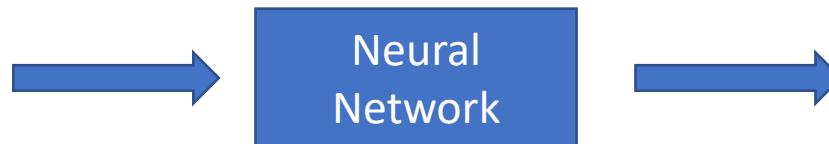
The University of Texas at Dallas

Some slides of this lecture are courtesy Stanford CS231n

# Visual Perception vs. Computational Perception



Image

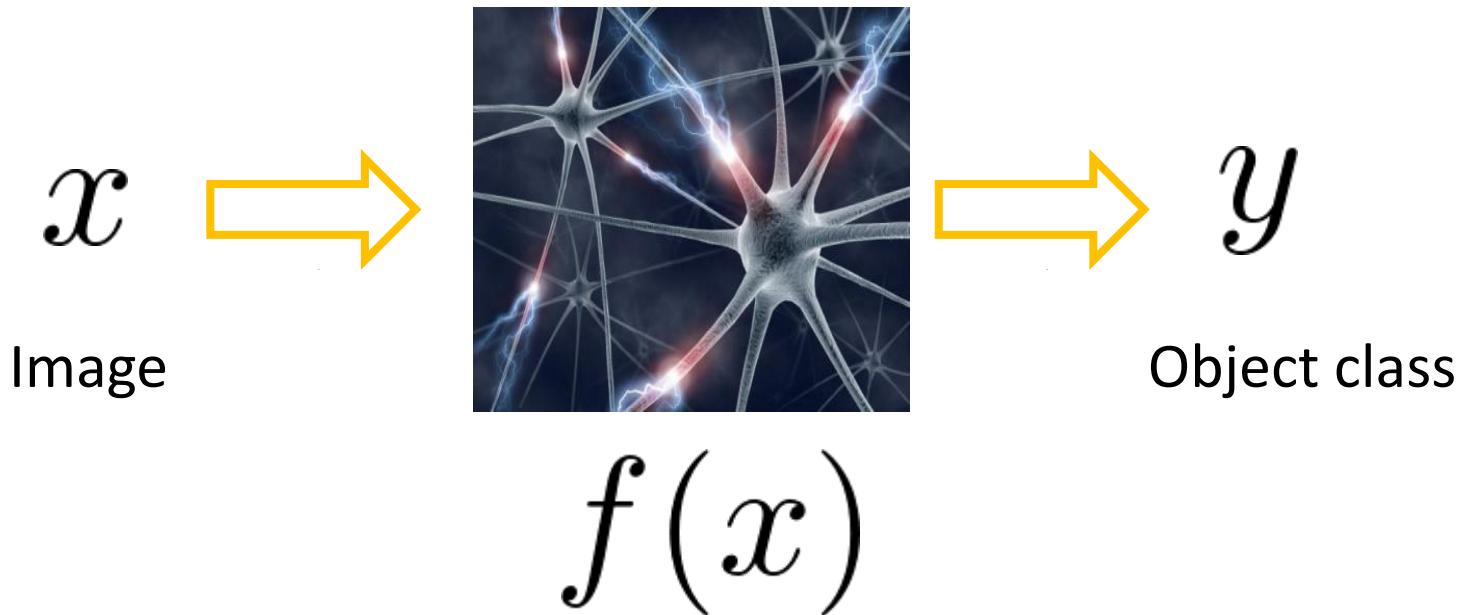


High-level information

- Depth
- Motion
- Object classes
- Object poses
- Etc.

# Mathematic Model

- Try to model the human brain with computational models, e.g., neural networks



# Mathematic Model

- What is the form of the function  $f(x)$ ?
  - No idea!
  - Concatenate simple functions (neurons)



$x$



$$f(x)$$



$$y \in \{+1, -1\}$$

Dog

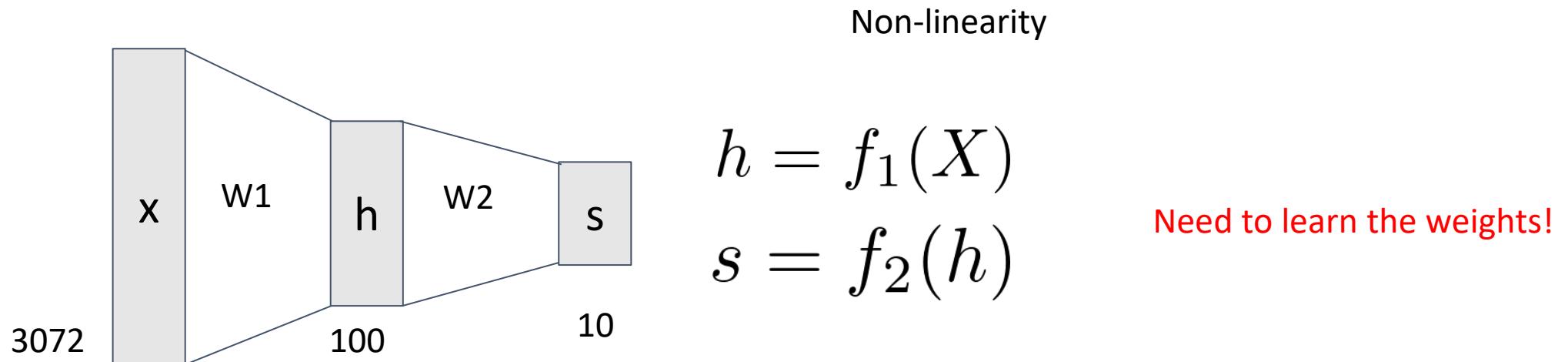
# Neural Network: Concatenation of functions

Linear score function:

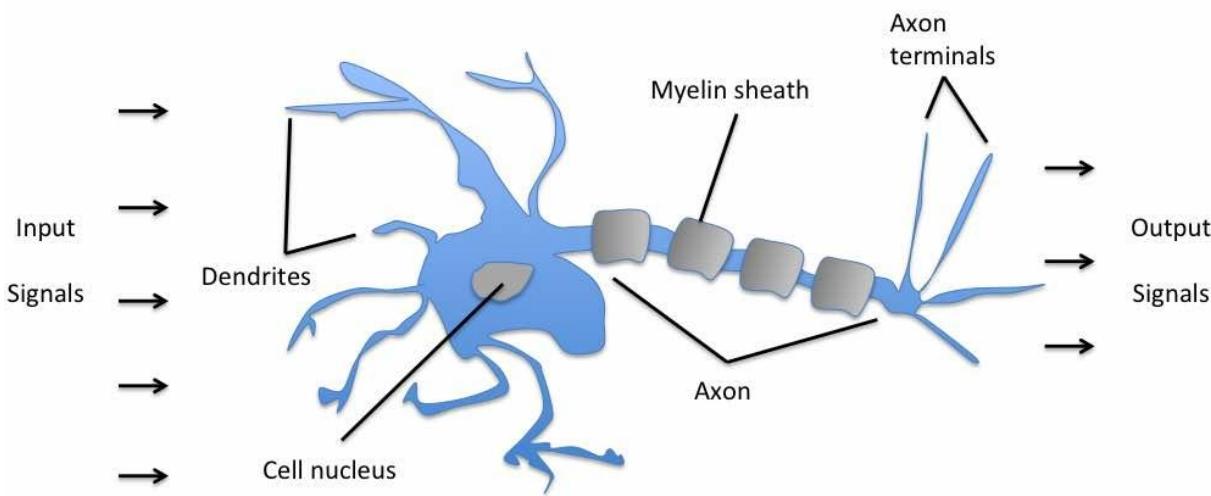
$$f = Wx$$

2-layer Neural Network

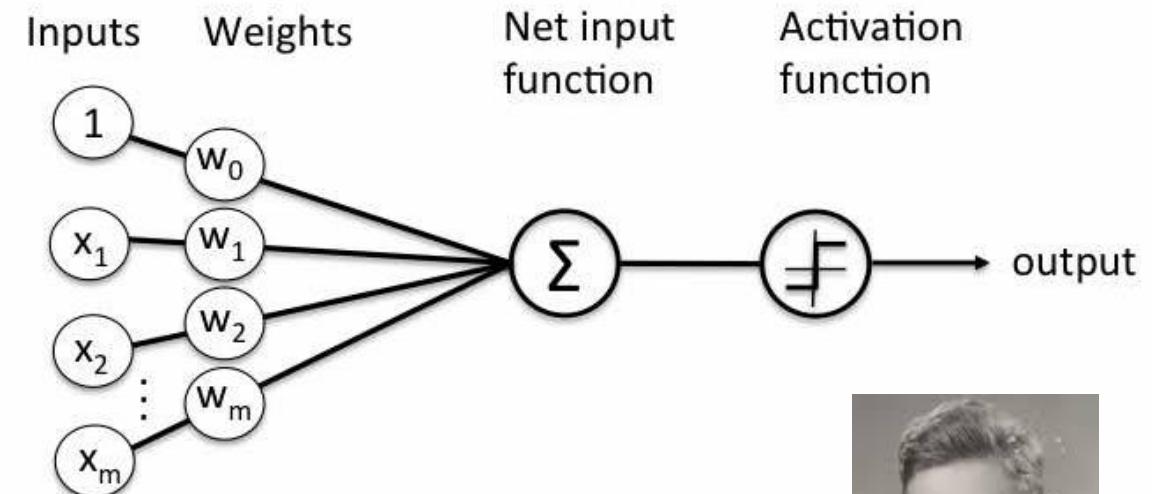
$$f = f_2(f_1(x)) = W_2 \max(0, W_1 x)$$



# Frank Rosenblatt's Perceptron



Schematic of a biological neuron.



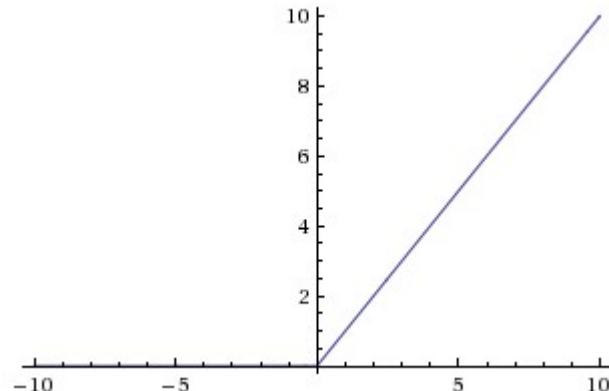
Frank Rosenblatt  
(1928-1971)

# Activation Functions

2-layer Neural Network

$$f = f_2(f_1(x)) = W_2 \max(0, W_1 x)$$

**Rectified Linear Unit (ReLU)**  
 $\max(0, x)$

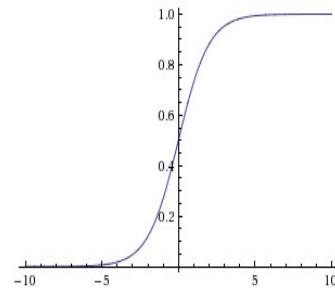


Introduce non-linearity to the network

# Activation Functions

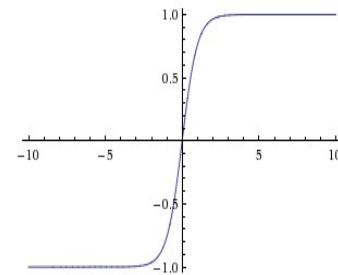
**Sigmoid**

$$\sigma(x) = 1/(1 + e^{-x})$$

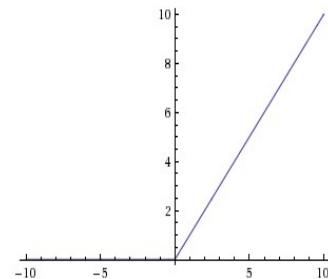


**tanh**  $\tanh(x)$

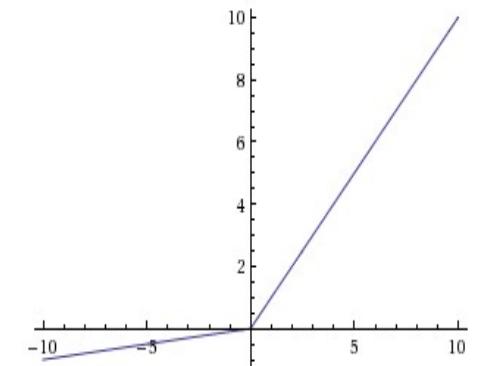
$$\frac{e^{2x} - 1}{e^{2x} + 1}$$



**ReLU**  $\max(0, x)$

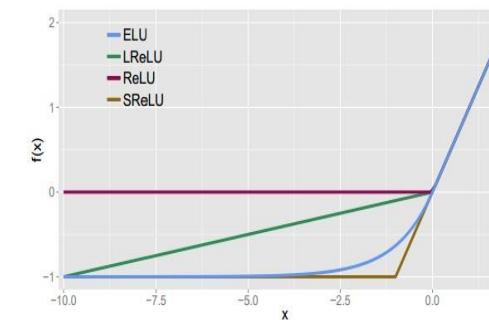


**Leaky ReLU**  
 $\max(0.1x, x)$



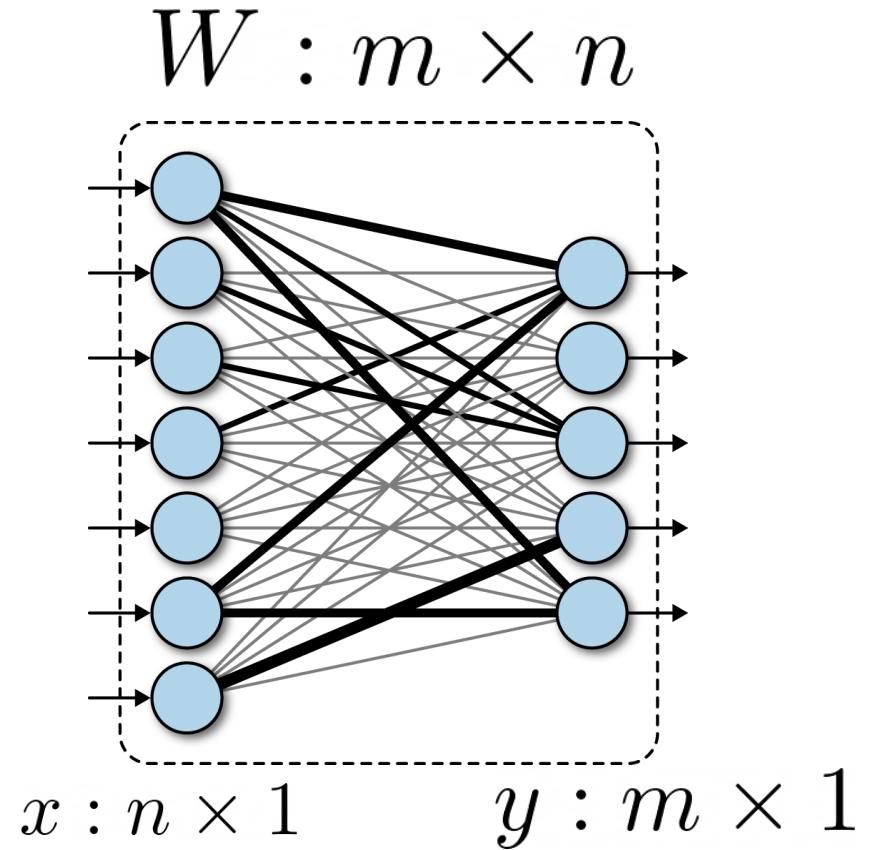
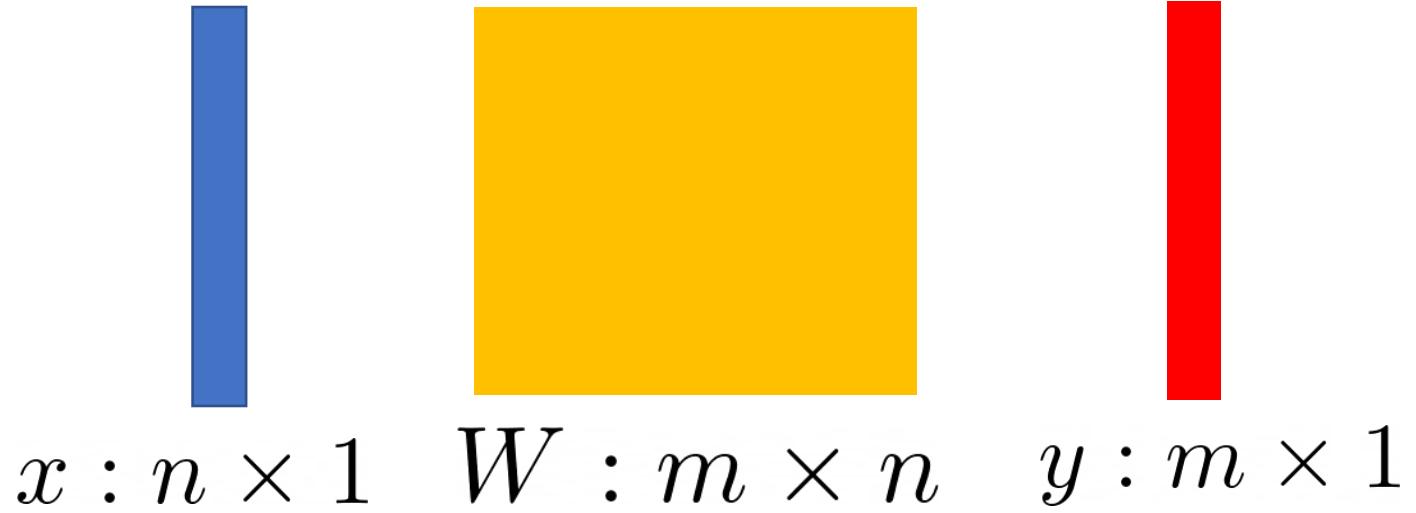
**Maxout**  $\max(w_1^T x + b_1, w_2^T x + b_2)$

**ELU** Exponential Linear Unit  $f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha (\exp(x) - 1) & \text{if } x \leq 0 \end{cases}$



# Fully Connected Layer

$$y = Wx$$



# Fully Connected Layer

- What is the drawback of only using fully connected layers?

$$y = Wx$$

- Consider an image with 640 x 480
  - $x$  is with dimension 307,200
  - The weight matrix of the fully connect layer is too large

# Further Reading

- Stanford CS231n, lecture 5, Convolutional Neural Networks  
<http://cs231n.stanford.edu/schedule.html>
- Deep learning with PyTorch  
[https://pytorch.org/tutorials/beginner/deep\\_learning\\_60min\\_blitz.html](https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html)
- AlexNet (2012):  
<https://papers.nips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>
- Vgg16 (2014): <https://arxiv.org/abs/1409.1556>
- GoogleNet (2014): <https://arxiv.org/abs/1409.4842>
- ResNet (2015): <https://arxiv.org/abs/1512.03385>