DS4023 Machine Learning

Lecture 4:
Neural Networks – Activation & Loss Function

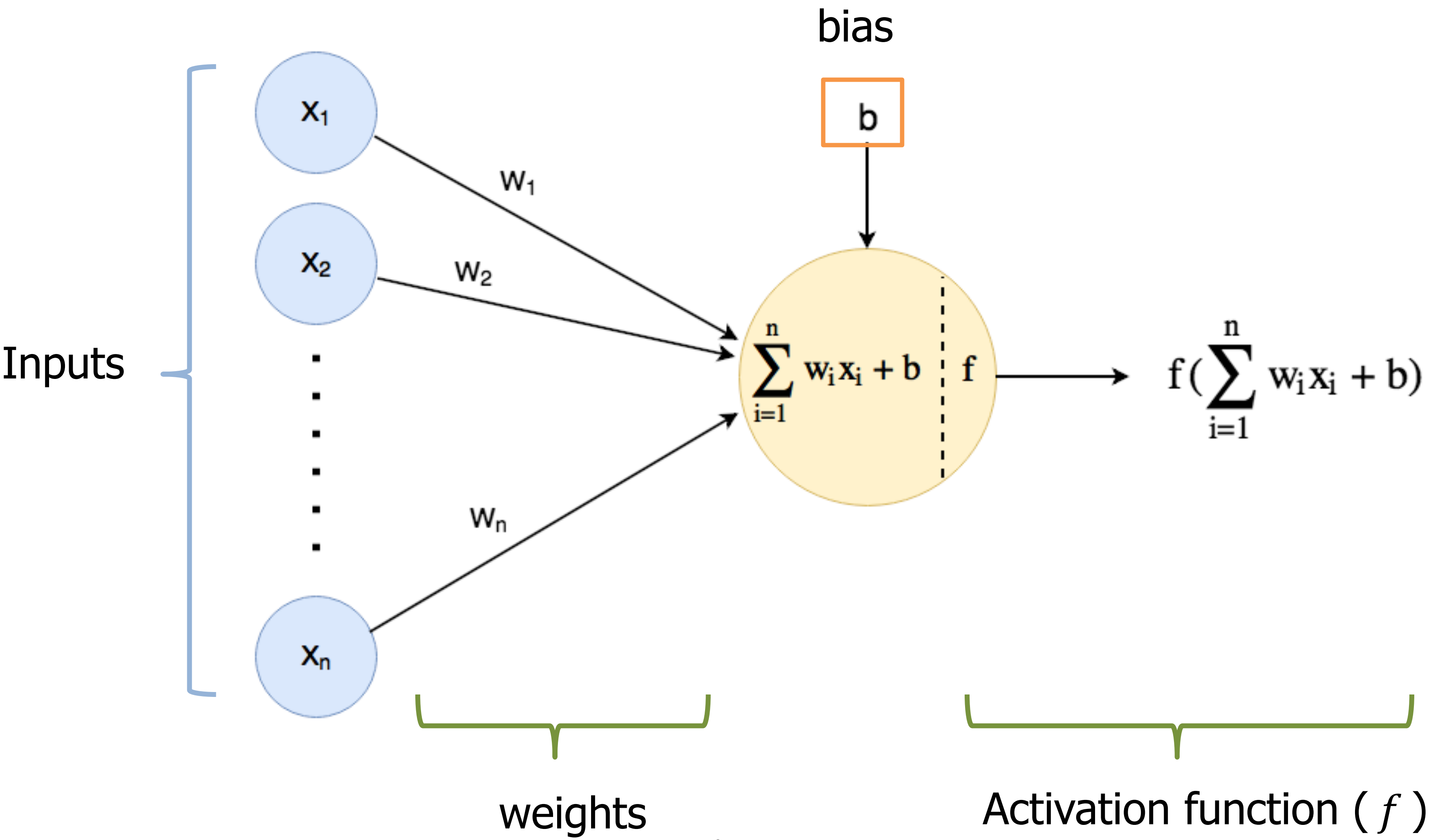
Mathematical Sciences
United International College

Reference: Stanford Course CS231n

Outline

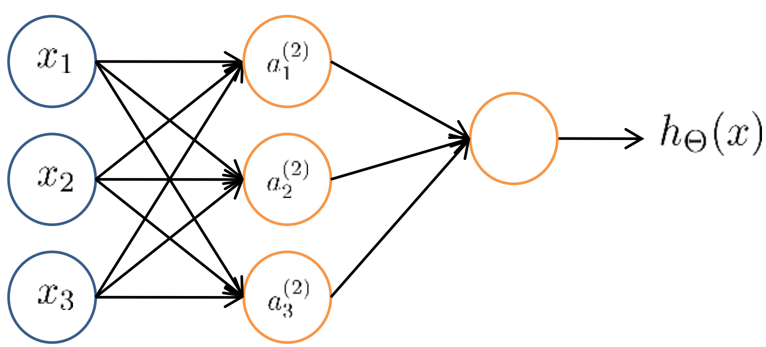
- Activation Functions
- Softmax Classifier
- Cross-entropy Loss

Activation Functions



Why Activation Function?

- Activation function is one of the building blocks on neural network, which brings non-linearity.
- A neural network without an activation function is essentially just a linear model.



Without non-linear activation:

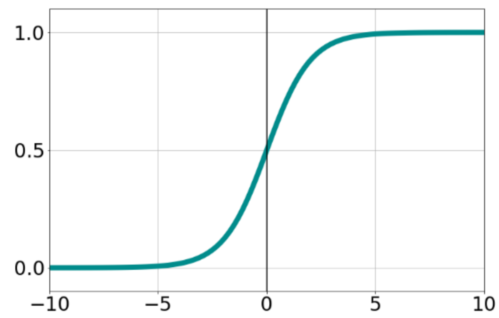
$$a^{(2)} = z^{(2)} = \Theta^{(1)}x$$

$$h_\Theta(x) = z^{(3)} = \Theta^{(2)}z^{(2)}$$
$$= \Theta^{(1)}\Theta^{(2)}x$$

Activation Functions

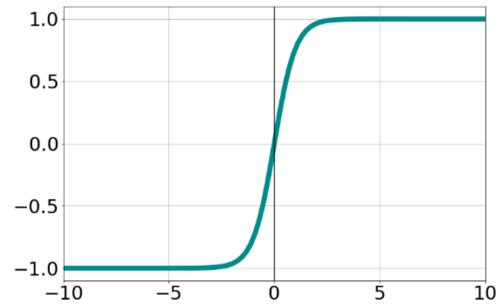
Sigmoid:

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



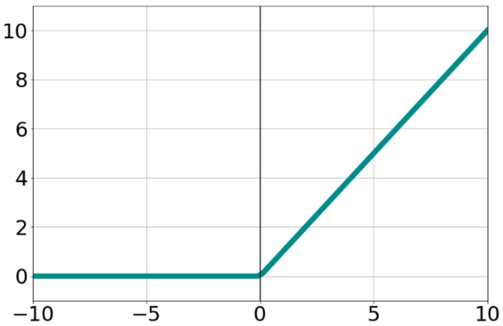
tanh:

$tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$



ReLU:

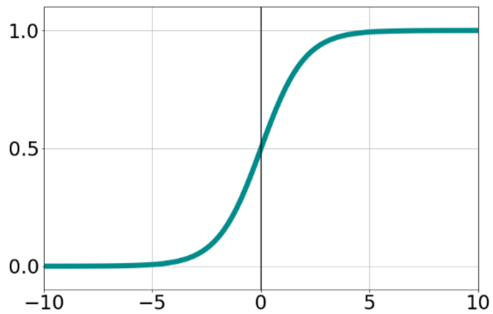
$ReLU(x) = \max(0, x)$



Sigmoid Function

Sigmoid:

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

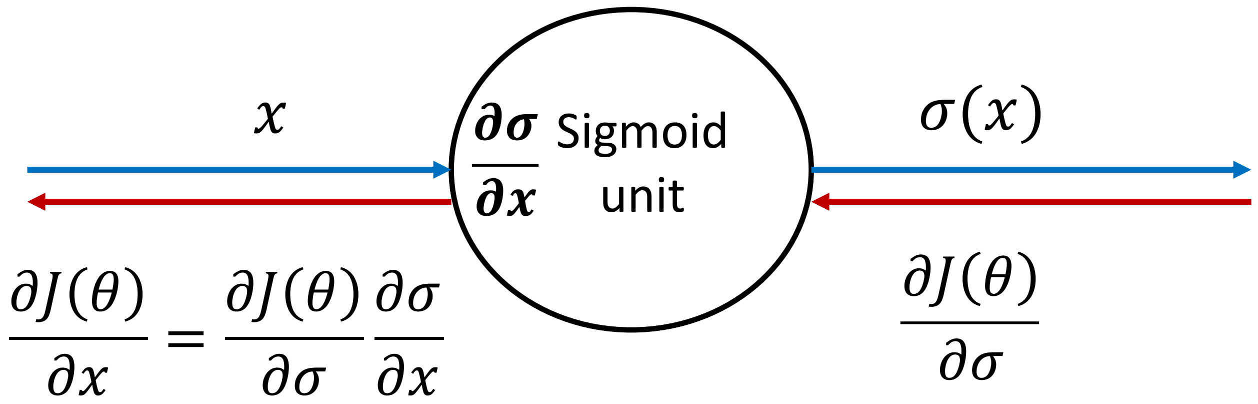


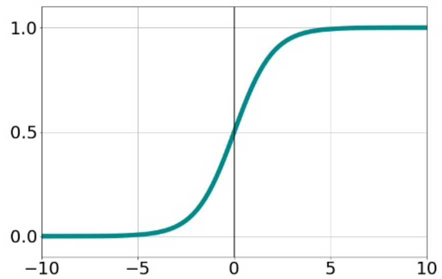
- Squashes numbers to range [0,1]
- Historically popular since they have nice interpretation as the “firing rate” of a neuron
  - Not firing at all (0)
  - Fully saturated firing at an assumed maximum frequency (1)

Drawback:

- Saturated neurons kill gradients.

Sigmoid Function





Gradient of sigmoid function: $\frac{\partial \sigma}{\partial x} = \sigma(x)(1-\sigma(x))$

What happens when x =-10?
What happens when x =10?
What happens when x =0?

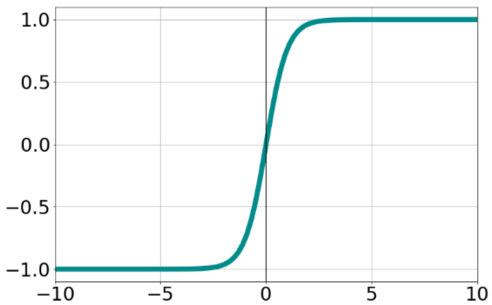
Sigmoid Function

- The output of sigmoid saturates at either 1 or 0, for a large positive or large negative number.
  - Thus, the local gradient at these regions is almost zero.
- During backpropagation, this local gradient is multiplied with upstream gradient, which "kills" the gradient, almost no signal flow through the neuron to its weights.

Tanh Function

tanh:

$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



- Hyperbolic tangent function.
- Squashes numbers to range [-1,1].
- Tanh outputs are zero-centered.
- Note that the Tanh function is simply a scaled sigmoid function:
$$\tanh(x) = 2\sigma(2x) - 1$$
- $\tanh'(x) = 1 - \tanh^2(x)$

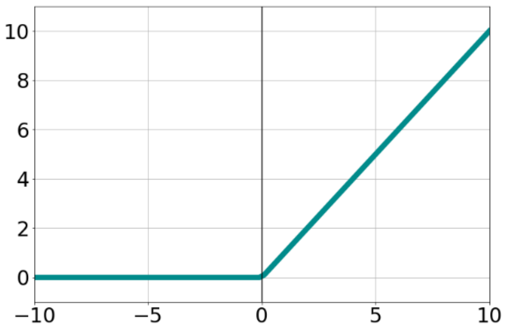
Drawback:

- Saturated neurons kill gradients.

ReLU Function

ReLU:

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

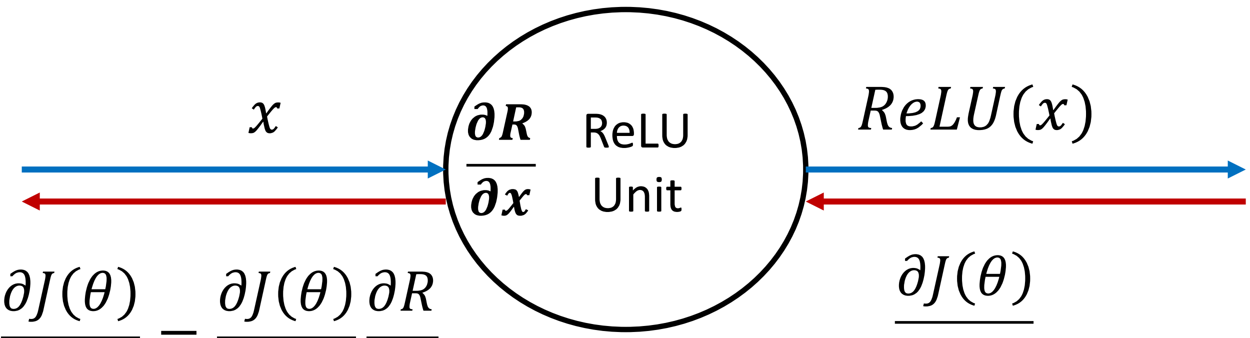


- Rectified Linear Unit
- The activation is simply thresholded at zero.
- Does not saturate (in +region)
- Very computationally efficient (no exponentials)
- Converges much faster than sigmoid/tanh in practice

Drawback:

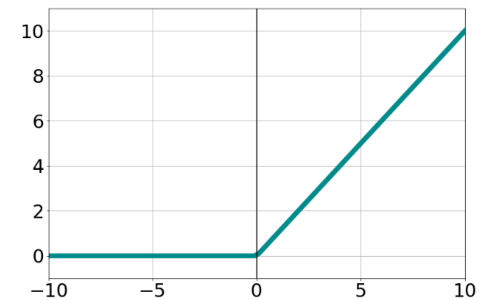
- Killing the gradient in half of the regime.

ReLU Function



$$\frac{\partial J(\theta)}{\partial x} = \frac{\partial J(\theta)}{\partial R}\frac{\partial R}{\partial x}$$

$$\frac{\partial J(\theta)}{\partial R}$$



Gradient of RuLU function: $\frac{\partial R}{\partial x} = \begin{cases} 1\ , & \text{if } x > 0 \\ 0\ , & \text{if } x < 0 \end{cases}$

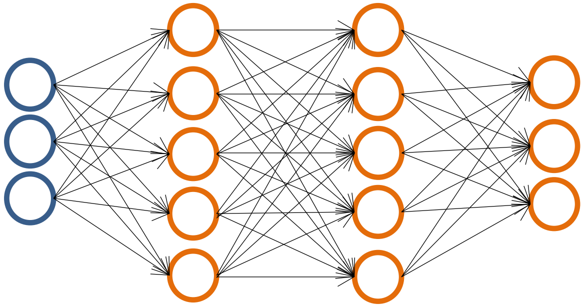
What happens when x =-10?
What happens when x =10?
What happens when x =0?

Multiclass Classification

Pedestrian

Car

Motorcycle



$$h_\Theta(x) \in \mathbb{R}^3$$

$$h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$$

when pedestrian

$$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$$

when car

$$h_\Theta(x) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

when motorcycle

Multiclass Classification

- In previous loss function, treat multiclass classification as multiple binary classifiers.
- Each classifier generates a score.

Input		**Output Score**	**Classes**
	Neural Network	0.8	Pedestrian
		0.6	Car
		0.2	Motorcycle

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log\left((h_\theta(x^{(i)}))_k\right) - \left(1 - y_k^{(i)}\right)\log(1 - (h_\theta(x^{(i)}))_k)\right]$$

Softmax Classifier

- Extension of binary classifiers that allows for more than two categories of outcome variable.
  - Get its name from the softmax function.
- Interpret raw classifier scores as probabilities.

Input		Output Score	Probability	Classes
	Neural Network	0.8	0.42	Pedestrian
		0.6	0.34	Car
		0.2	0.23	Motorcycle

- Softmax Function: $f(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$
  - $\vec{z}$ is input vector with K entries.
  - Exponential (monotonic, output positive value) and normalization.

Cross-Entropy Loss

- Output probability close to true probability.
- The most common loss function in this case is the cross-entropy loss.
- KL-divergence. Given two probability distributions p and q, the use of KL-divergence is to measure the difference between the two distributions.

$$KL(p||q) = \sum_{i=1}^{K} p_i \log \frac{p_i}{q_i}$$

  – More similar, smaller KL-divergence
  – Range $[0, +\infty]$

Cross-Entropy Loss

- KL-Divergence:

$$KL(p||q) = \sum_{i=1}^{K} p_i \log \frac{p_i}{q_i} = \sum_{i=1}^{K} [p_i \log p_i - p_i \log q_i]$$

$$= \sum_{i=1}^{K} p_i \log p_i - \sum_{i=1}^{K} p_i \log q_i$$

$$= -Entroy(p) - \sum_{i=1}^{K} p_i \log q_i$$

- The cross-entropy between two probability distributions, such as q from p, can be stated formally as: $H(p, q) = -\sum_{i=1}^{K} p_i \log q_i$
  - Given p, the distribution q with larger KL-divergence, also have larger cross-entropy.

Cross-Entropy Loss

- Let p be true class distribution, q be our prediction distribution, we use cross-entropy as our loss function.
- For true class distribution, only one entry is 1, other entries are 0.

Input		Output Probability	True Probability	Classes
	Neural Network	0.42	1	Pedestrian
		0.34	0	Car
		0.23	0	Motorcycle

Cross-Entropy Loss

- Given the data instance (x, y), let p be true class distribution, q be our prediction distribution, we use cross-entropy as our loss function.

$$L = -\sum_{i=1}^{K} p_i \log q_i$$

  - For p, only one entry is 1, other entries are 0.
- Therefore, cross-entropy loss can be re-write as:

$$L = -\log q_t, \quad \text{where } p_t = 1$$

Output Probability	**True Probability**
0.42	1
0.34	0
0.23	0

cross-entropy loss:
$-\log 0.42 = 0.87$