AI Coffe Club (09/01/2020)
Activation Functions

Last modification: January 9, 2020

Activation functions are used to modify or control the output of layers of a neural network. More precisely, they are applied to each neuron and they determine whether they should be activated or not and, in more complex cases, the magnitude of such activation.

Although activation functions were traditionally used as a mathematical gate, a simple step function, to turn the neuron on and off depending on a threshold to produce binary classifications, nowadays they are used for more complex purposes.

Modern neural networks use non-linear activation functions to create complex mappings between the inputs and the output and generate more complex decision boundaries to tackle difficult classification problems.

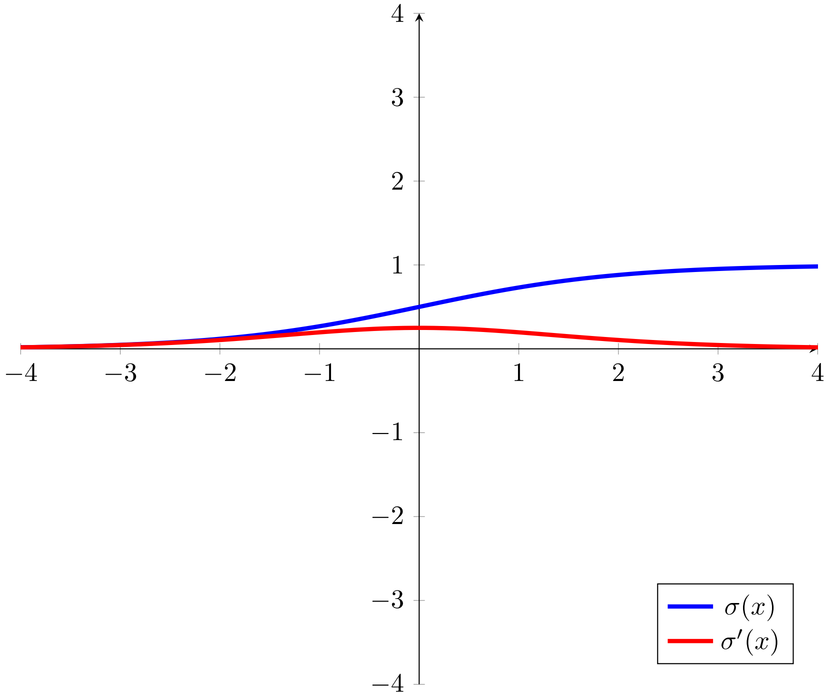
Many activation functions have been proposed to date, most of them trying to strike a balance between making the training easier, increasing representation capacity, avoiding common training stalemates, and computational efficiency. Table 1 shows a summarized overview.

Remark: we need activation functions to be differentiable so as to perform backpropagation optimization.

Name	Function	Derivative	Range
Sigmoid	$\sigma(x) = \frac{1}{1+e^{-x}}$	$\sigma'(x) = \sigma(x)(1-\sigma(x))$	$(0,1)$
Hyperbolic Tangent	$\sigma(x) = \frac{e^{2x}-1}{e^{2x}+1}$	$\sigma'(x) = 1-\sigma(x)^2$	$(-1,1)$
ReLU	$\sigma(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$	$\sigma(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases}$	$[0,\infty)$
Leaky ReLU	$\sigma(x) = \begin{cases} 0.01x & x \leq 0 \\ x & x > 0 \end{cases}$	$\sigma(x) = \begin{cases} 0.01 & x \leq 0 \\ x & x > 0 \end{cases}$	$(-\infty,\infty)$
Parametric ReLU	$\sigma(x) = \begin{cases} \alpha x & x \leq 0 \\ x & x > 0 \end{cases}$	$\sigma(x) = \begin{cases} \alpha & x \leq 0 \\ x & x > 0 \end{cases}$	$(-\infty,\infty)$
ELU	$\sigma(x) = \begin{cases} \alpha(e^x-1) & x \leq 0 \\ x & x > 0 \end{cases}$	$\sigma(x) = \begin{cases} \alpha+\sigma(x) & x \leq 0 \\ x & x > 0 \end{cases}$	$(-\alpha,\infty)$
Softplus	$\sigma(x) = \ln(1+e^x)$	$\sigma'(x) = \frac{1}{1+e^{-x}}$	$(0,\infty)$
Swish	$\sigma(x) = \frac{x}{1+e^{-\beta x}}$	$\sigma'(x) = \beta\sigma(x) + \frac{1}{1+e^{-\beta x}}(1-\beta\sigma(x))$	

Table 1: Common activation functions.

1 Sigmoid

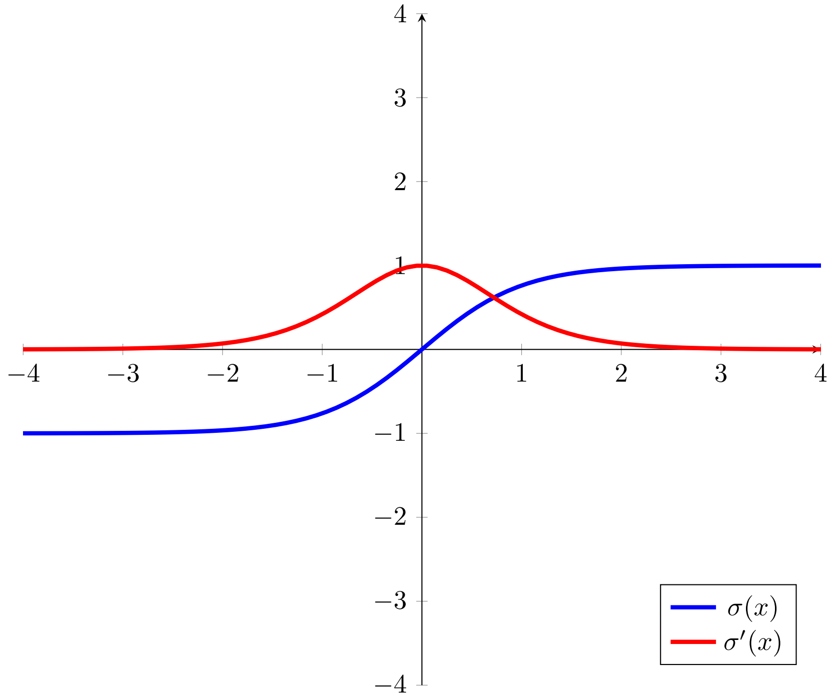


$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{1}$$

$$\sigma'(x) = \sigma(x)(1-\sigma(x)) \tag{2}$$

- Smooth gradient.
- Output values are bounded normalizing the output.
- Clear activations which easily saturate to the range extremes.
- Vanishing gradients.
- Not zero centered outputs.
- Computationally expensive.

2 Hyperbolic Tangent

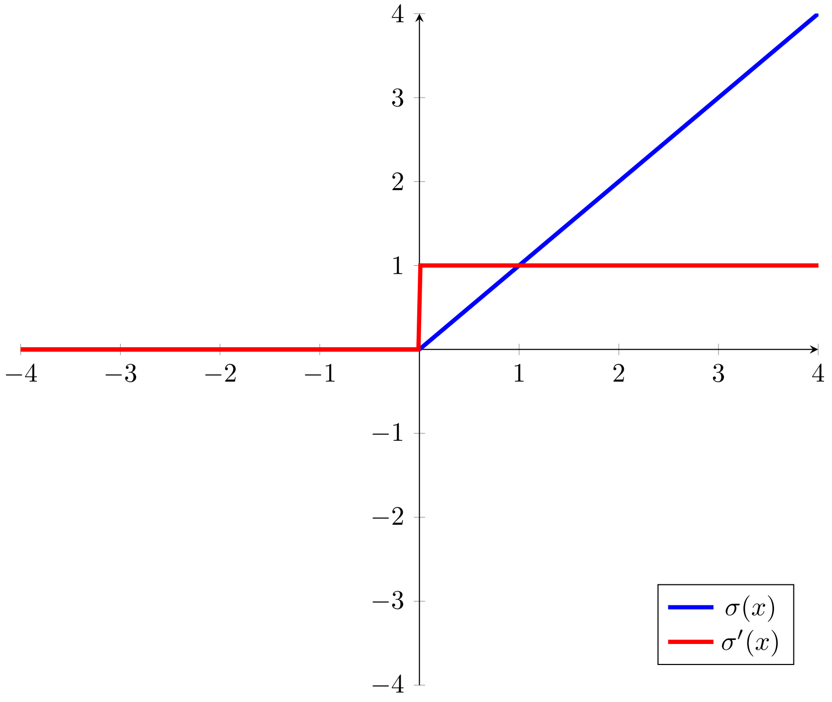


$$\sigma(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{3}$$

$$\sigma'(x) = 1 - \sigma(x)^2 \tag{4}$$

- Smooth gradient.
- Output values are bounded normalizing the output.
- Clear activations which easily saturate to the range extremes.
- Zero-centered outputs.
- Vanishing gradients.
- Computationally expensive.

3 Rectified Linear Unit

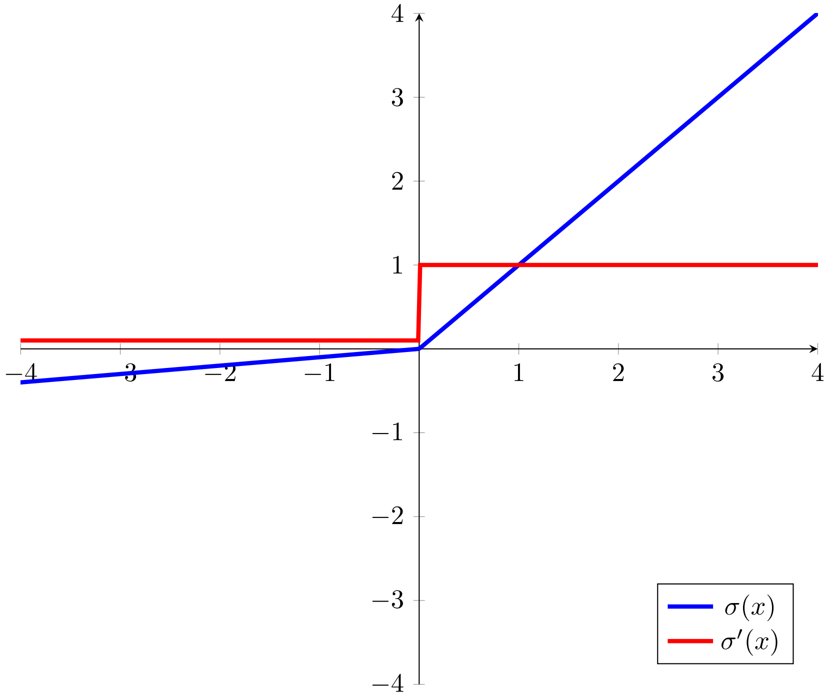


$$\sigma(x) = \begin{cases} 0 & x \leq 0 \\ x & x > 0 \end{cases} \tag{5}$$

$$\sigma'(x) = \begin{cases} 0 & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{6}$$

- Computationally efficient.
- Not affected by the vanishing gradient and converges faster.
- Sparse activations.
- Unbounded output values.
- Dying ReLU since it is zero for all negative values and the slope in the negative range is zero it can ”die”. A large gradient flowing through a ReLU can cause the weights to update in such a way that the neuron will never activate on any datapoint again. The gradient flowing through this unit will be forever zero.

4 Leaky Rectified Linear Unit

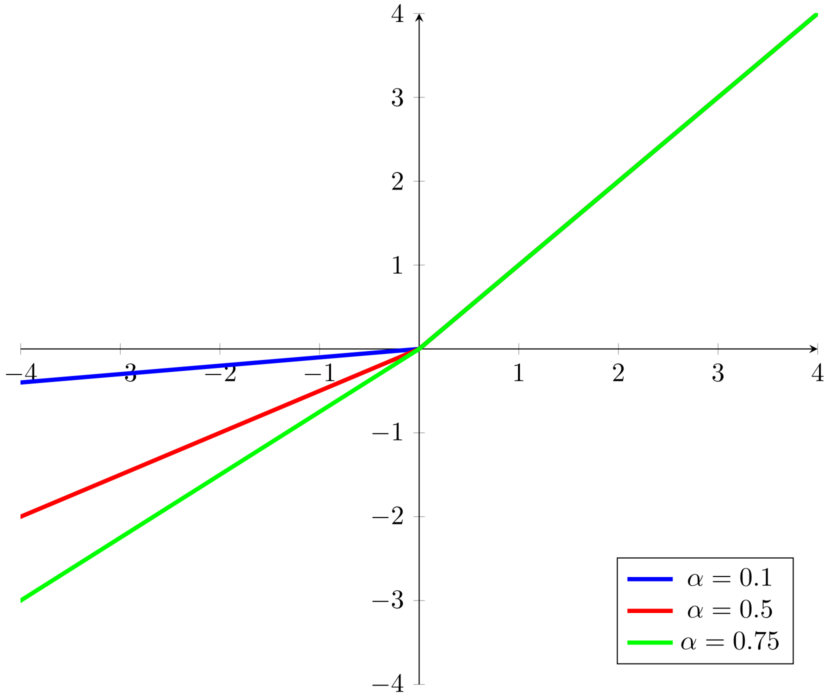


$$\sigma(x) = \begin{cases} 0.1x & x \leq 0 \\ x & x > 0 \end{cases} \tag{7}$$

$$\sigma'(x) = \begin{cases} 0.1 & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{8}$$

- Computationally efficient.
- Not affected by the vanishing gradient and converges faster.
- Prevents the dying ReLU problem!
- Sparse activations.
- Not zero-centered.
- Unbounded output values.
- Inconsistent results due to inconsistent predictions for negative values.

5 Parametric Rectified Linear Unit

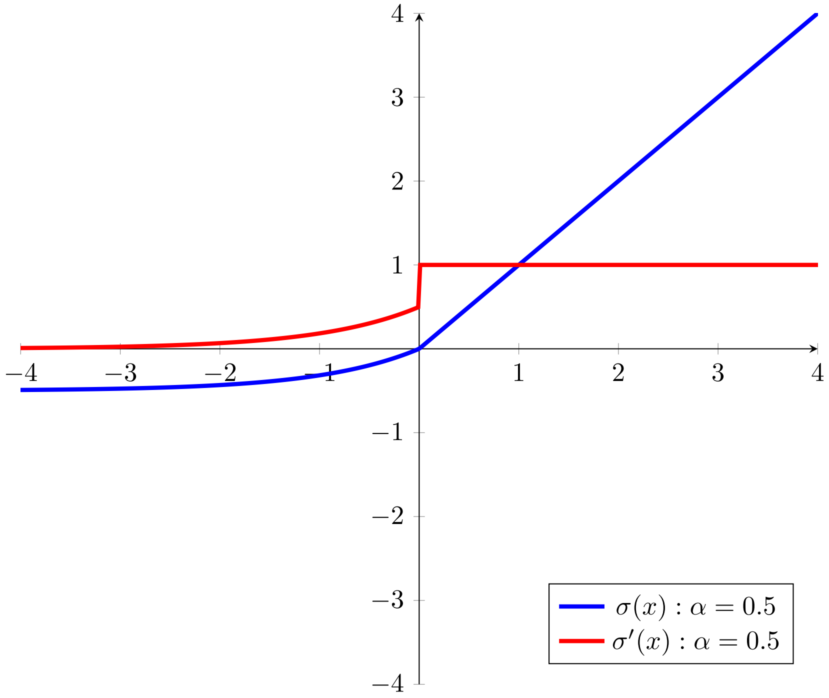


$$\sigma(x) = \begin{cases} \alpha x & x \leq 0 \\ x & x > 0 \end{cases} \tag{9}$$

$$\sigma'(x) = \begin{cases} \alpha & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{10}$$

- Computationally efficient.
- Not affected by the vanishing gradient and converges faster.
- Prevents the dying ReLU problem!
- Sparse activations.
- Negative slope is learned and backpropagation is performed with the most appropriate value.
- Not zero-centered.
- Unbounded output values.

6 Exponential Linear Unit

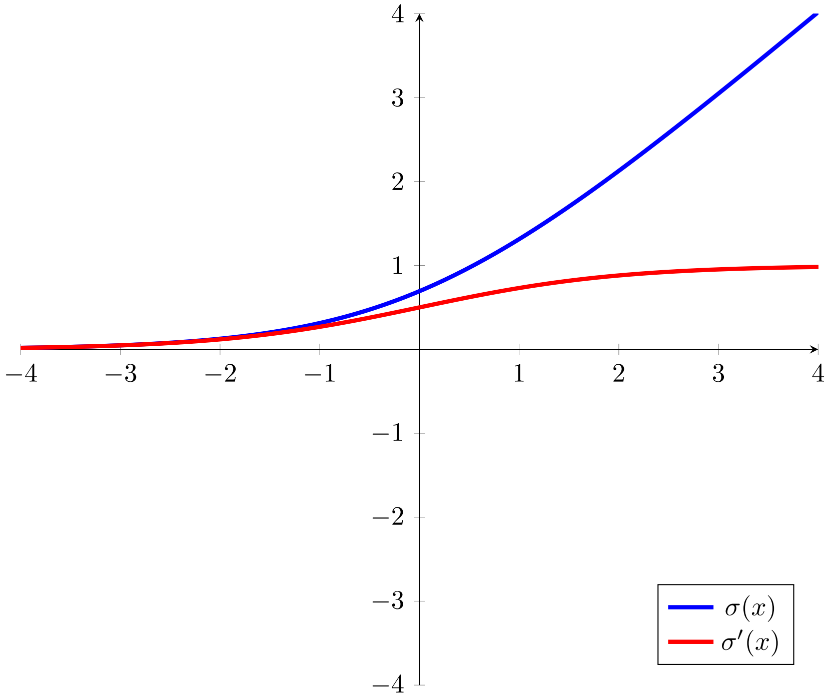


$$\sigma(x) = \begin{cases} \alpha(e^x - 1) & x \leq 0 \\ x & x > 0 \end{cases} \tag{11}$$

$$\sigma'(x) = \begin{cases} \alpha + \sigma(x) & x \leq 0 \\ 1 & x > 0 \end{cases} \tag{12}$$

- Computationally efficient.
- Not affected by the vanishing gradient and converges faster.
- Prevents the dying ReLU problem!
- Sparse activations.
- Negative slope is learned and backpropagation is performed with the most appropriate value.
- Smooth negative slope.
- Not zero-centered.
- Unbounded output values.

7 Softplus

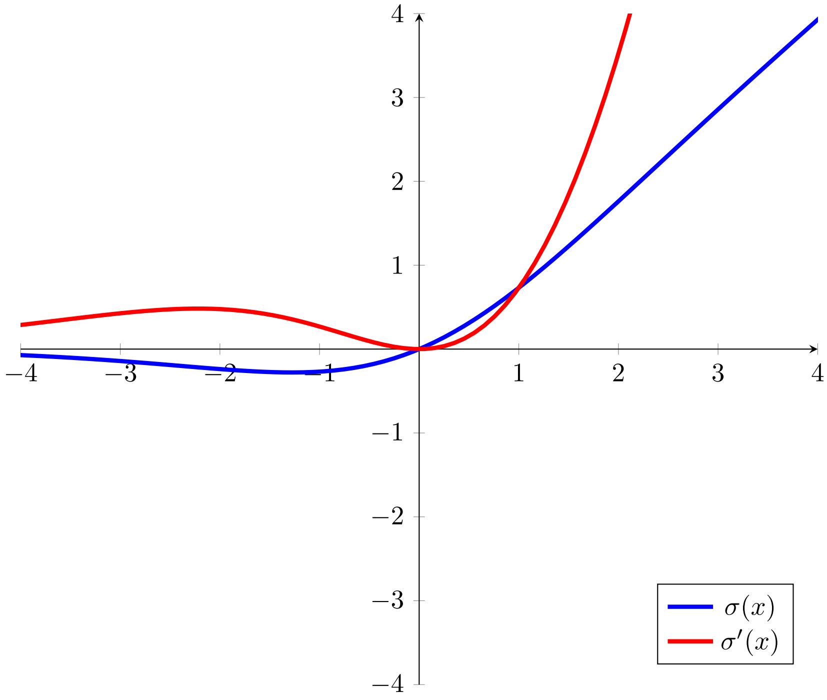


$$\sigma(x) = \ln(1 + e^x) \tag{13}$$

$$\sigma'(x) = \frac{1}{1 + e^{-x}} \tag{14}$$

- All the advantages of sigmoid/tanh.
- No vanishing gradients.
- Computationally inefficient.

8 Swish



$$\sigma(x) = \frac{x}{1 + e^{-x}} \tag{15}$$

$$\sigma'(x) = x\sigma(x) \tag{16}$$

- All the advantages of sigmoid/tanh.
- Similar behaviour to ReLU but smooth.
- Computationally inefficient.