Indian Institute of Information Technology, Allahabad

Activation Functions

By

Dr. Shiv Ram Dubey
Assistant Professor
Computer Vision And Biometrics Lab (CVBL)
Department Of Information Technology
Indian Institute Of Information Technology, Allahabad

Email: srdubey@iiita.ac.in Web: https://profile.iiita.ac.in/srdubey/

DISCLAIMER

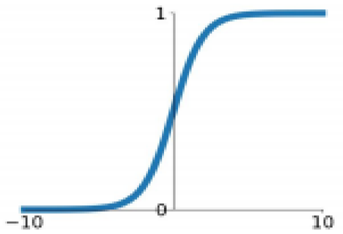
The content (text, image, and graphics) used in this slide are adopted from many sources for academic purposes. Broadly, the sources have been given due credit appropriately. However, there is a chance of missing out some original primary sources. The authors of this material do not claim any copyright of such material.

NON-LINEARITY LAYER

Activation Functions

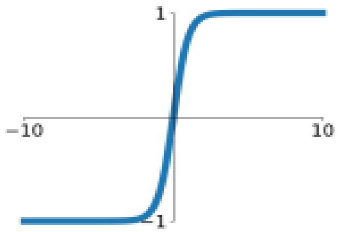
Sigmoid

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



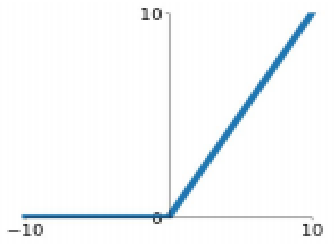
tanh

$\tanh(x)$



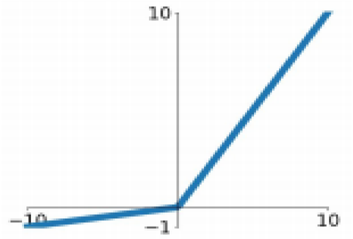
ReLU

$\max(0, x)$



Leaky ReLU

$\max(0.1x, x)$

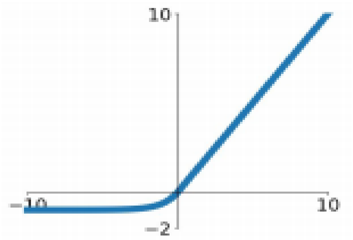


Maxout

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

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

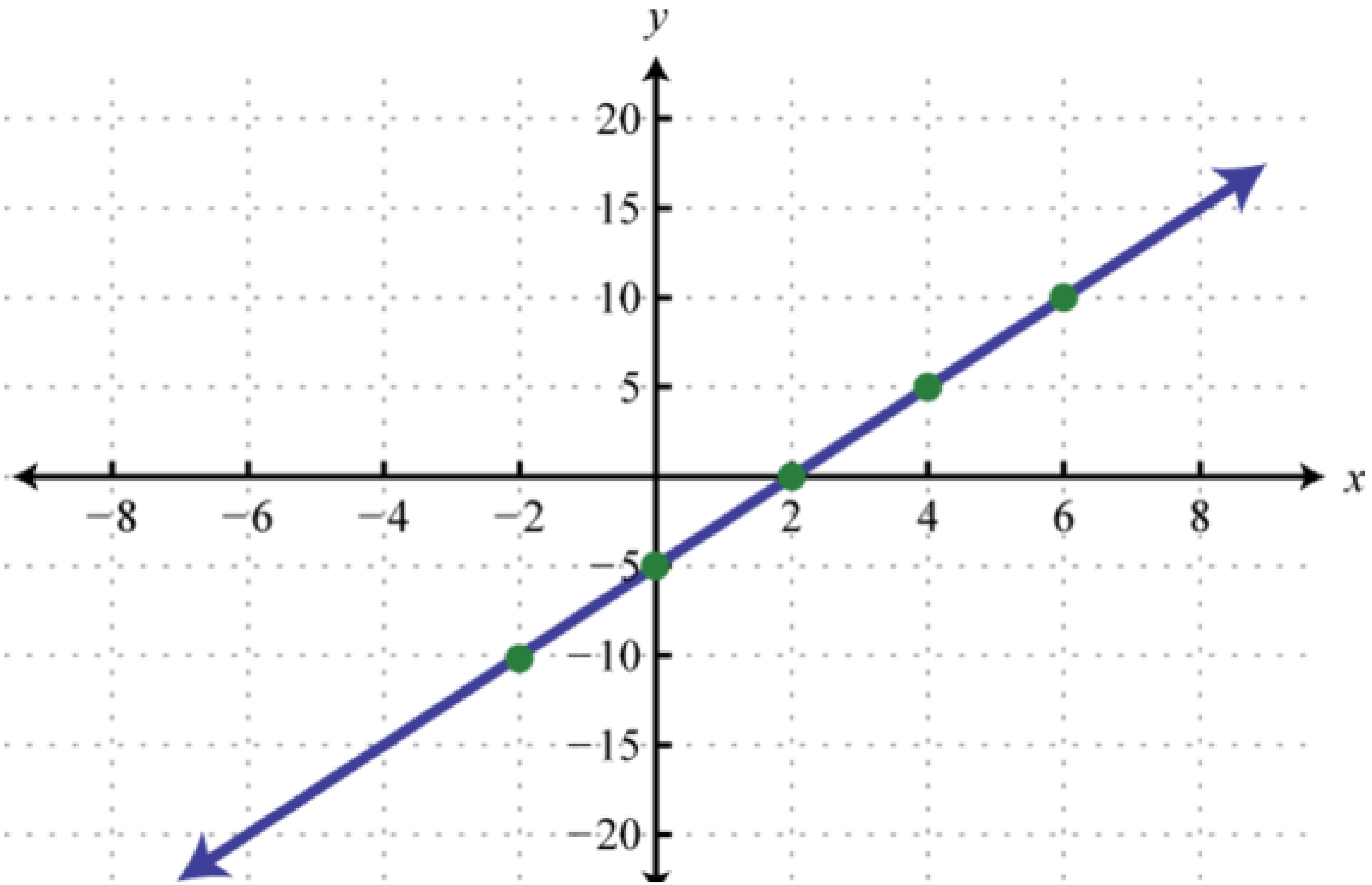


Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: LINEAR

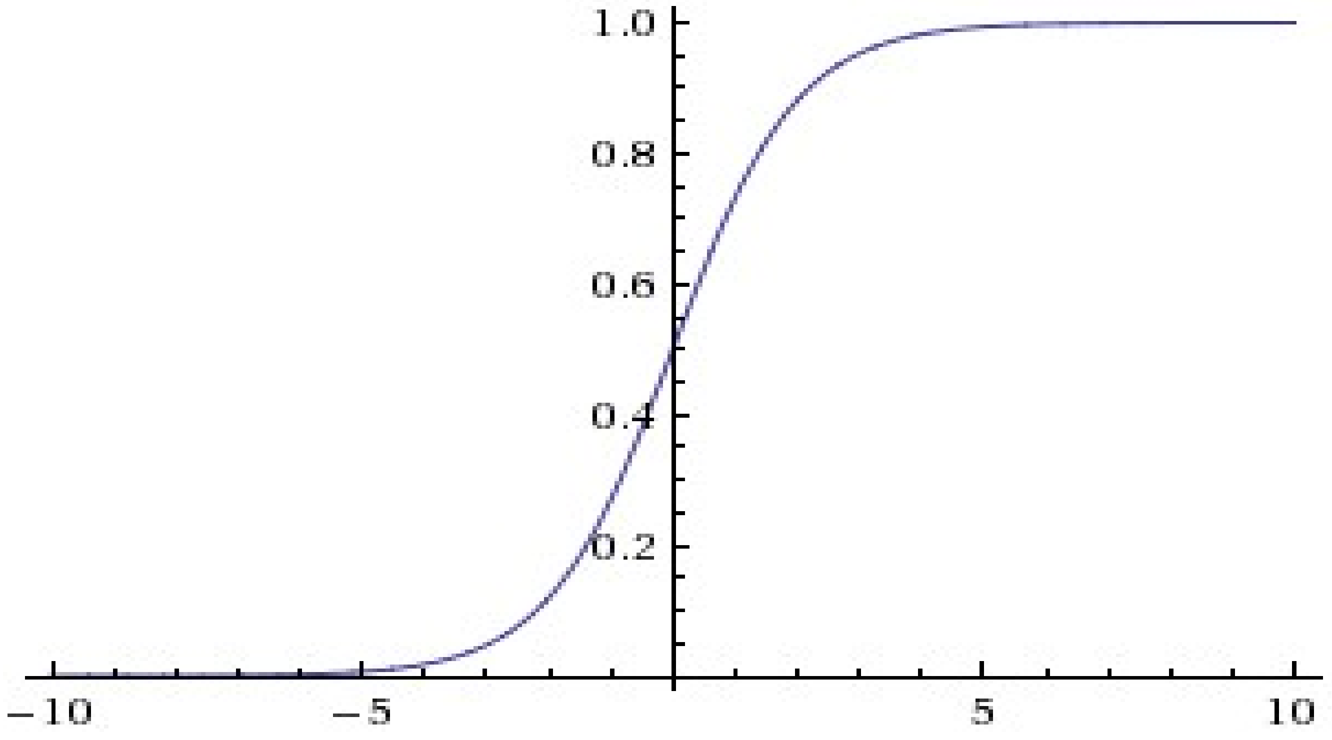
- Simplest activation function

- Does not include any non-linearity.



ACTIVATION FUNCTIONS: SIGMOID

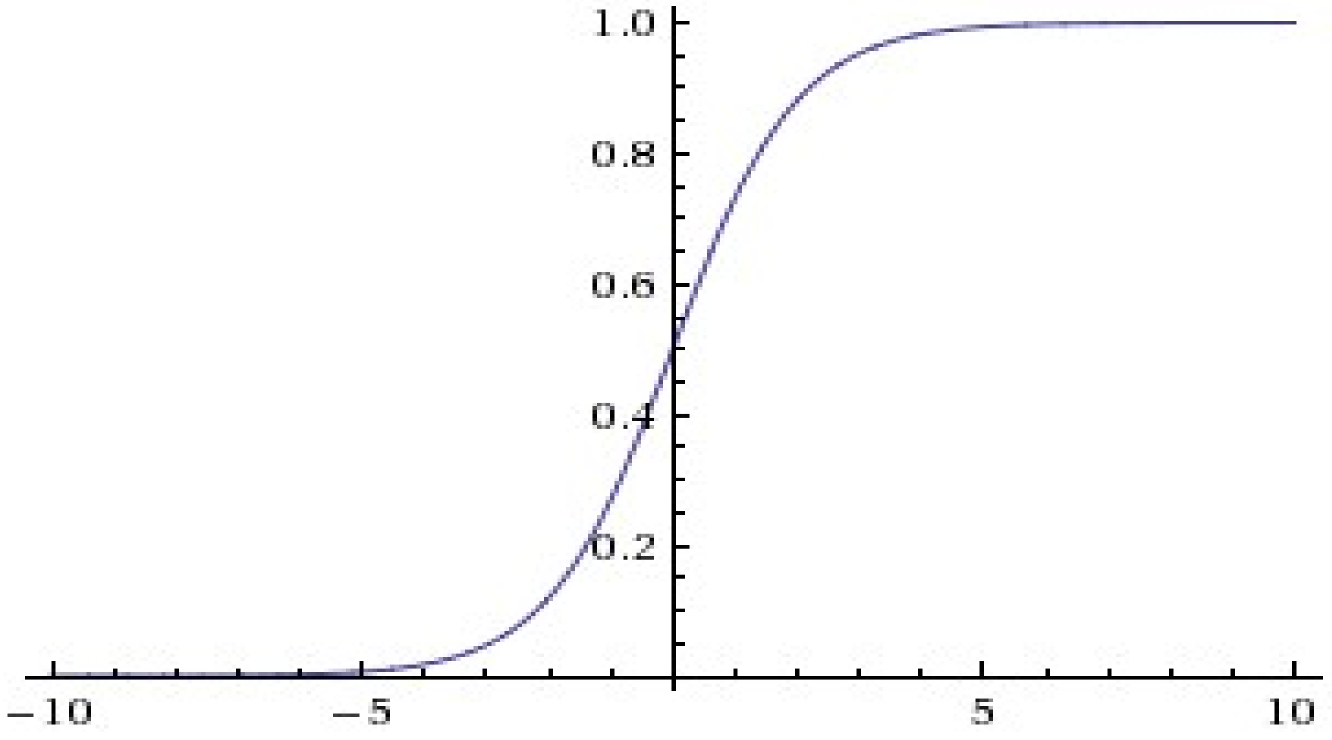
$$\sigma(x) = 1/(1 + e^{-x})$$



Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: SIGMOID

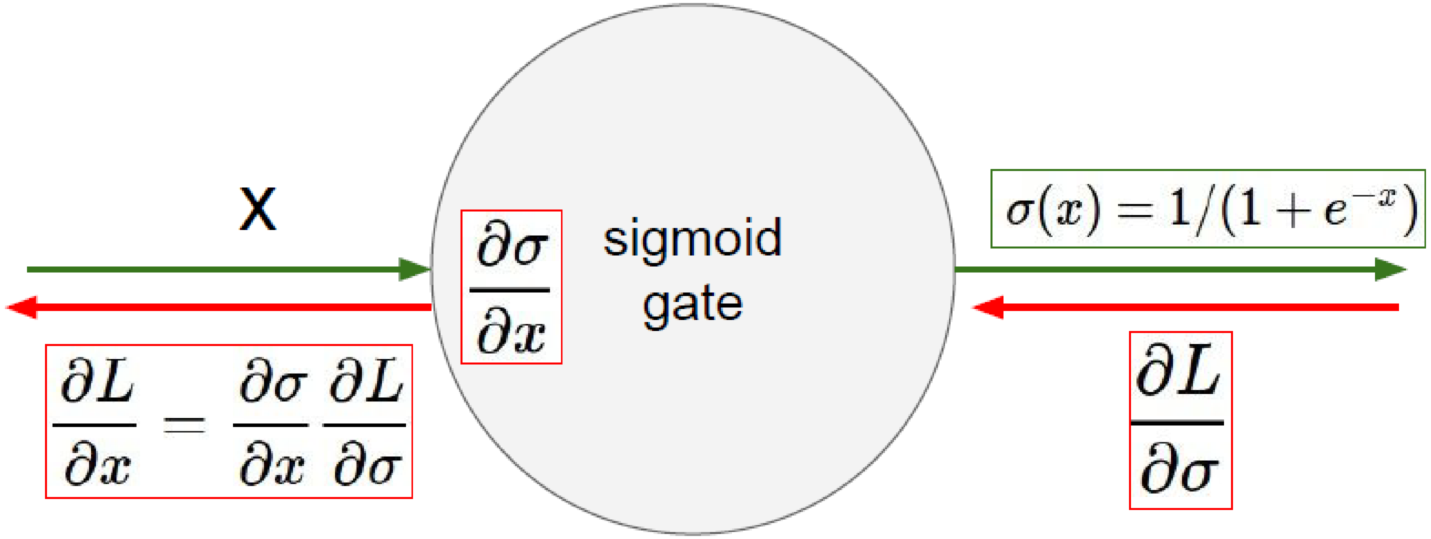
$$\sigma(x) = 1/(1 + e^{-x})$$

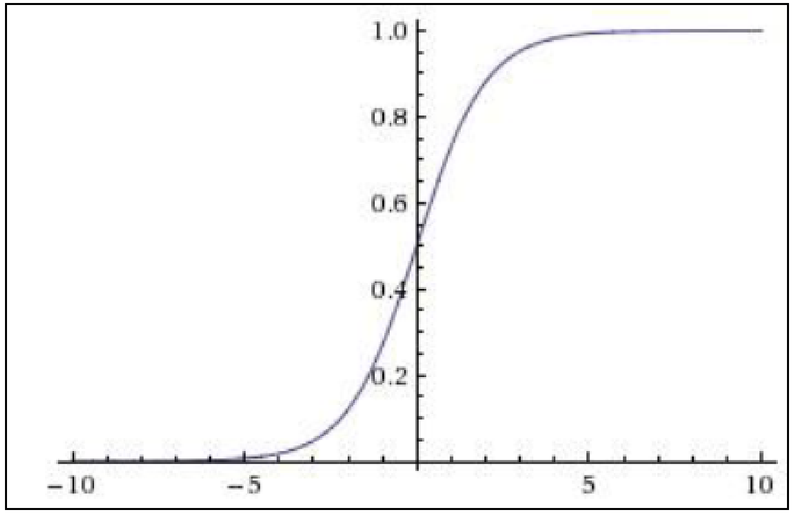


- Sigmoids saturate and kill gradients.

Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: SIGMOID



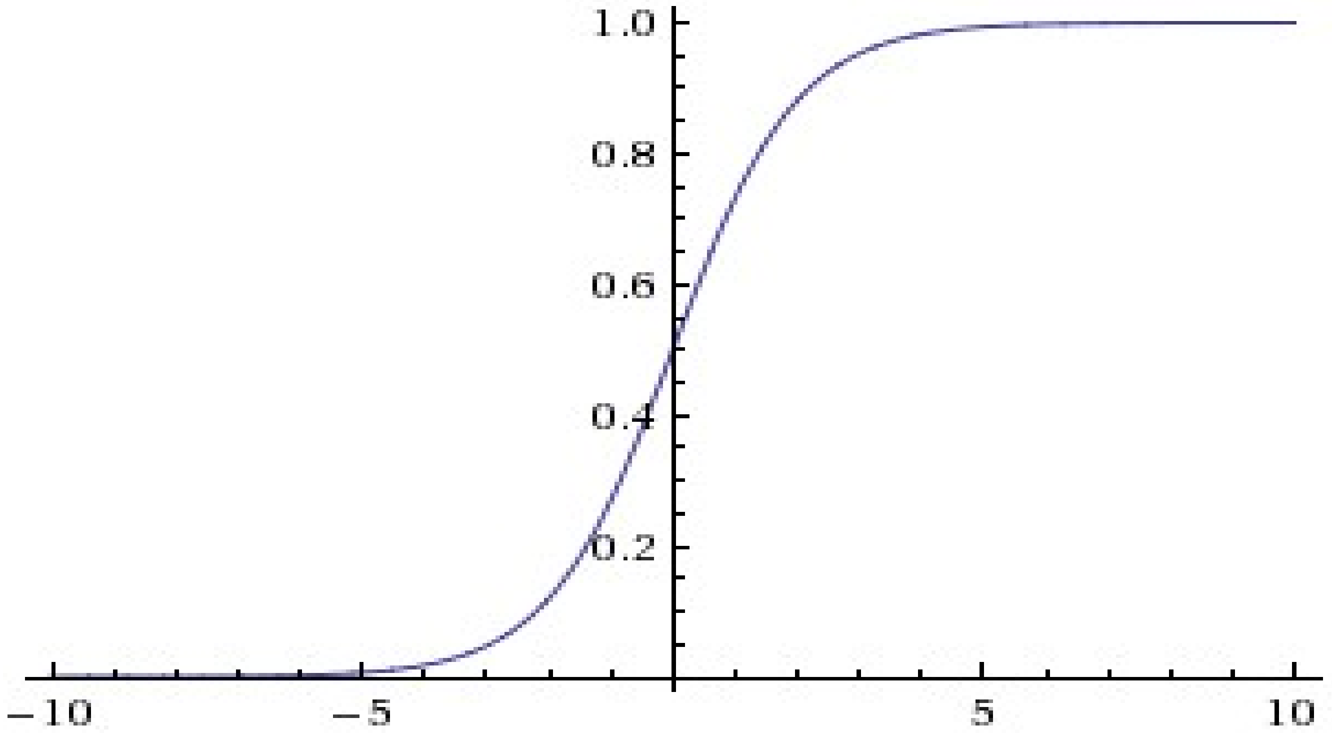


What happens when x = -10?
What happens when x = 0?
What happens when x = 10?

Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: SIGMOID

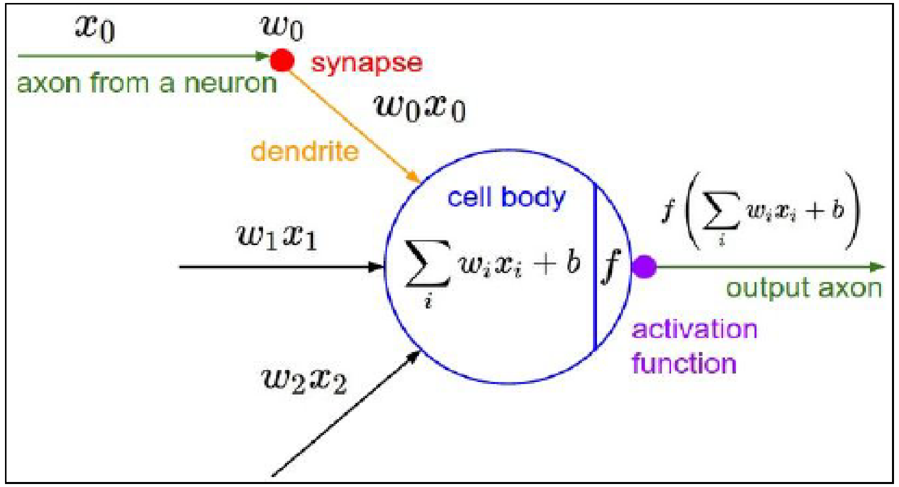
$$\sigma(x) = 1/(1 + e^{-x})$$



- Sigmoids saturate and kill gradients.
- Sigmoid outputs are not zero-centered.

Source: cs231n, Stanford University

Consider what happens when the input to a neuron (x) is always positive:



$$f\left(\sum_i w_i x_i + b\right)$$

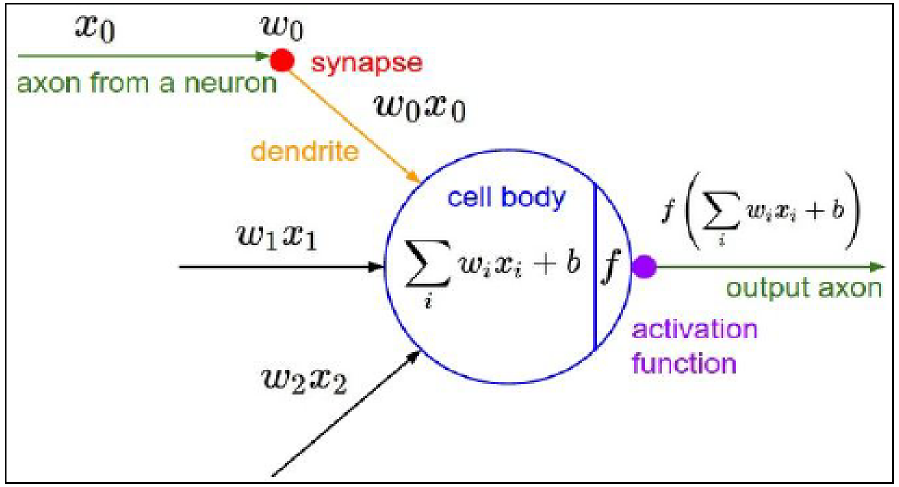
What can we say about the gradients on **w**?

Always all positive or all negative

(this is also why you want zero-mean data!)

Source: cs231n, Stanford University

Consider what happens when the input to a neuron (x) is always positive:



$$f\left(\sum_i w_i x_i + b\right)$$

What can we say about the gradients on **w**?

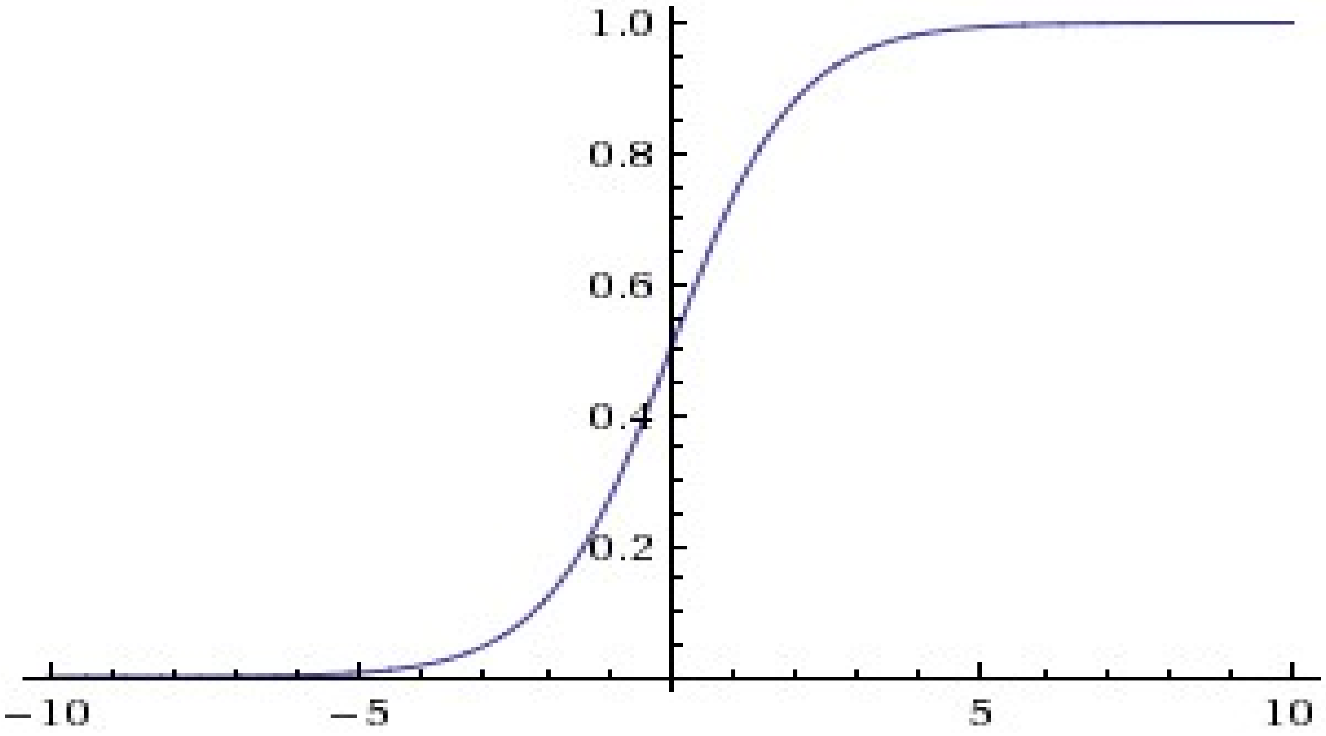
Always all positive or all negative

(this is also why you want zero-mean data!)

Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: SIGMOID

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

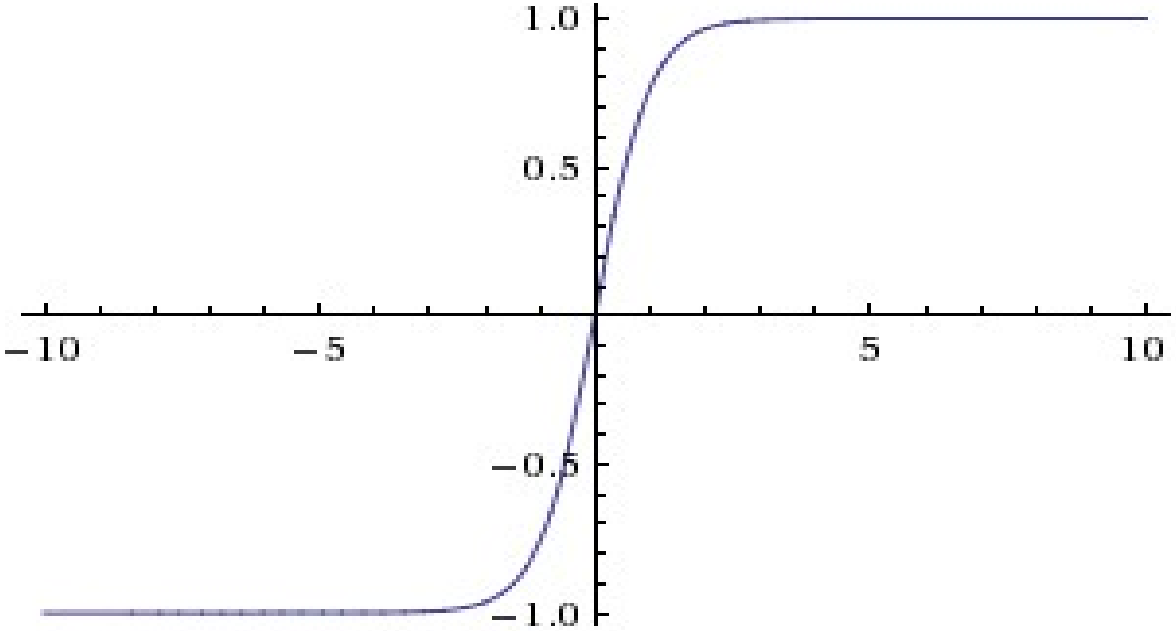


- Sigmoids saturate and kill gradients.
- Sigmoid outputs are not zero-centered.
- Exp() is a bit compute expensive.

Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: TANH

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



[LeCun et al., 1991]

Source: http://cs231n.github.io

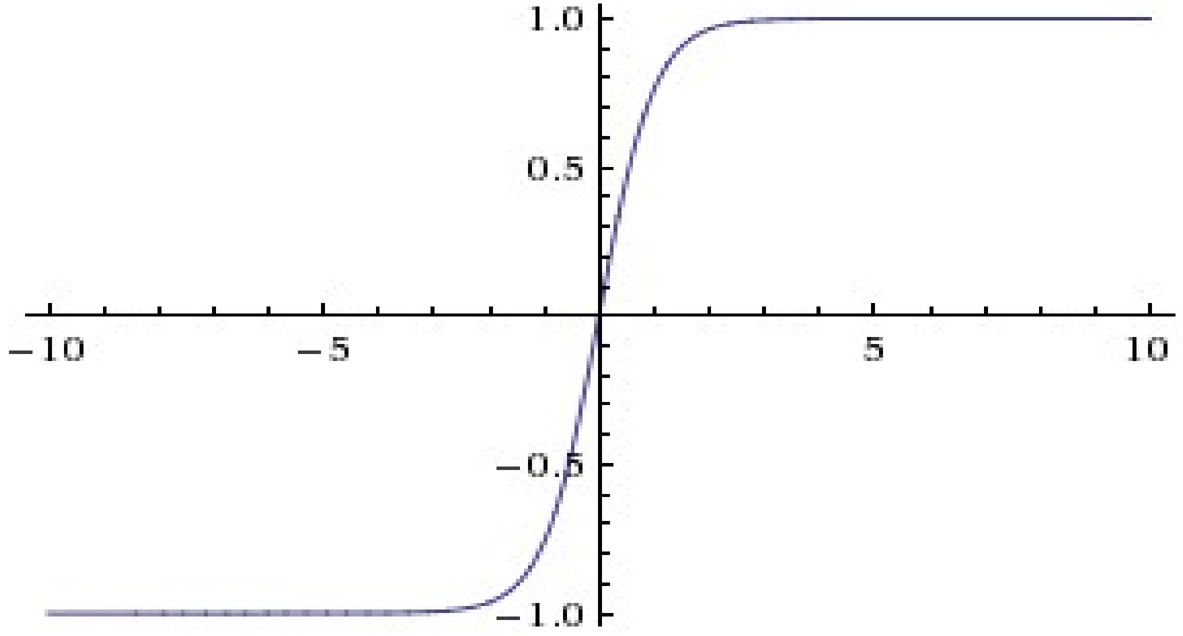
ACTIVATION FUNCTIONS: TANH

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

tanh neuron is simply a scaled sigmoid neuron

$$\tanh(x) = 2\sigma(2x) - 1.$$

Sigmoid



[LeCun et al., 1991]

Source: http://cs231n.github.io

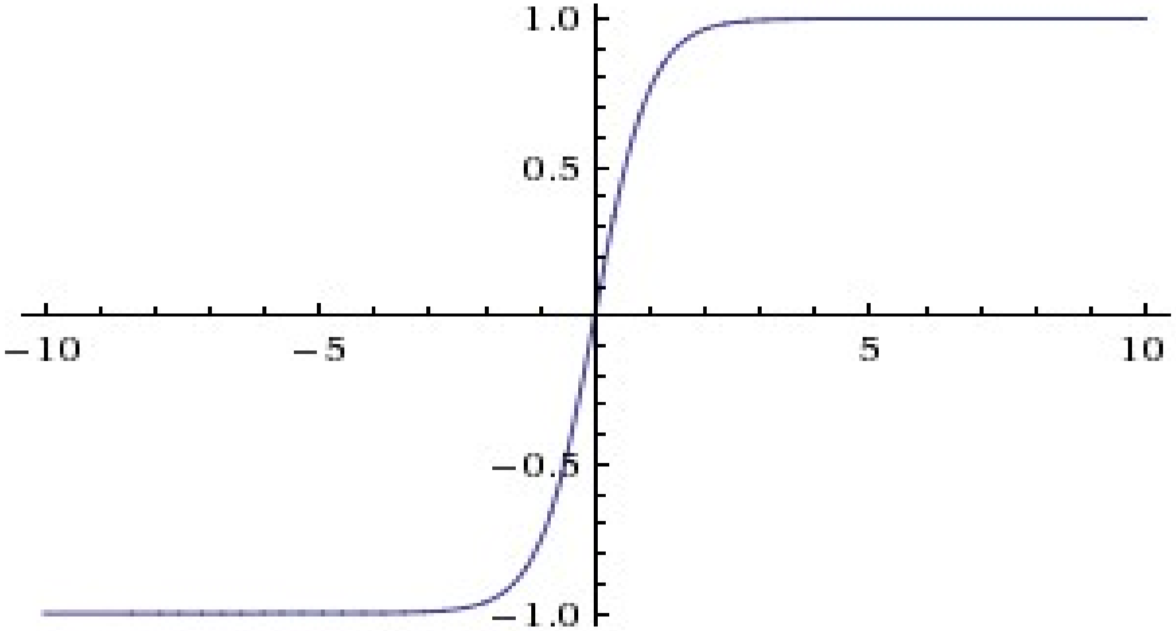
ACTIVATION FUNCTIONS: TANH

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

tanh neuron is simply a scaled sigmoid neuron

$$\tanh(x) = 2\sigma(2x) - 1.$$

Sigmoid



Like the sigmoid neuron, its activations saturate.

Unlike the sigmoid neuron its output is zero-centered.

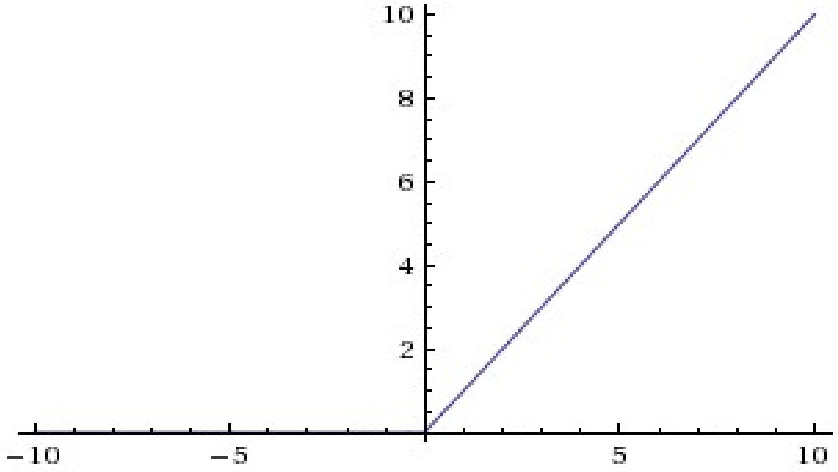
In practice the *tanh non-linearity is always preferred to the sigmoid nonlinearity.*

[LeCun et al., 1991]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: RELU

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

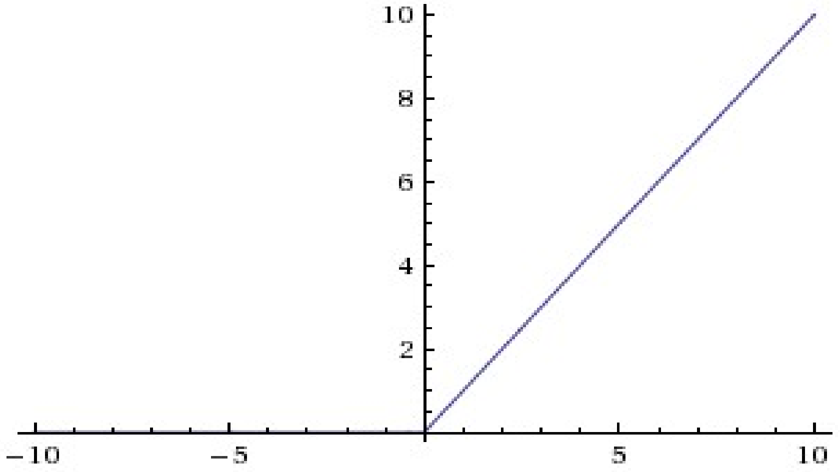


[Krizhevsky et al., 2012]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: RELU

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



ReLU is 6 times faster in the convergence of stochastic gradient descent compared to the sigmoid/tanh (Krizhevsky et al.).

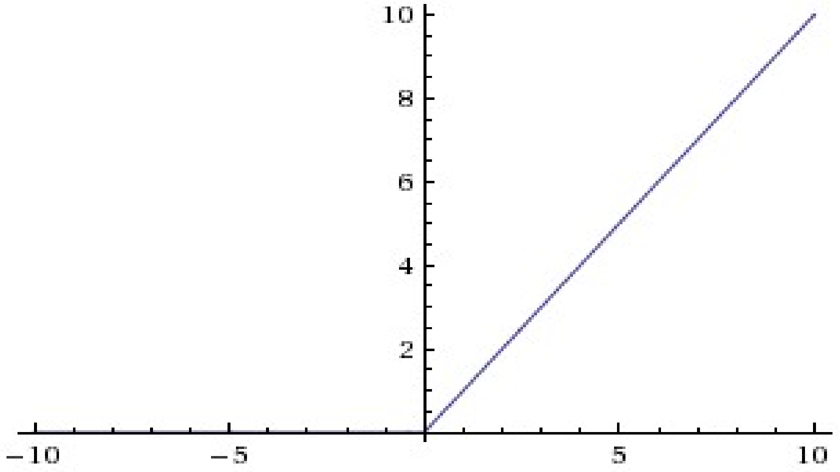
ReLU is simple as compared to tanh/sigmoid that involve expensive operations (exponentials, etc.)

[Krizhevsky et al., 2012]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: RELU

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



ReLU is 6 times faster in the convergence of stochastic gradient descent compared to the sigmoid/tanh (Krizhevsky et al.).

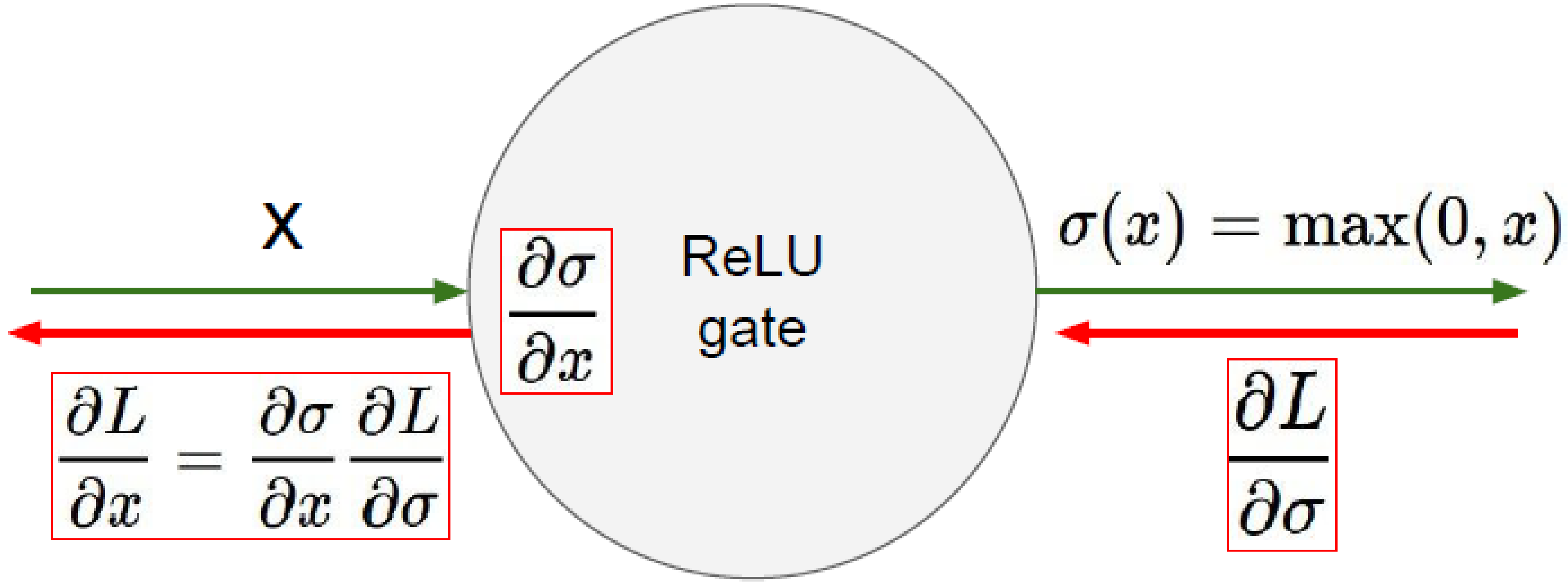
ReLU is simple as compared to tanh/sigmoid that involve expensive operations (exponentials, etc.)

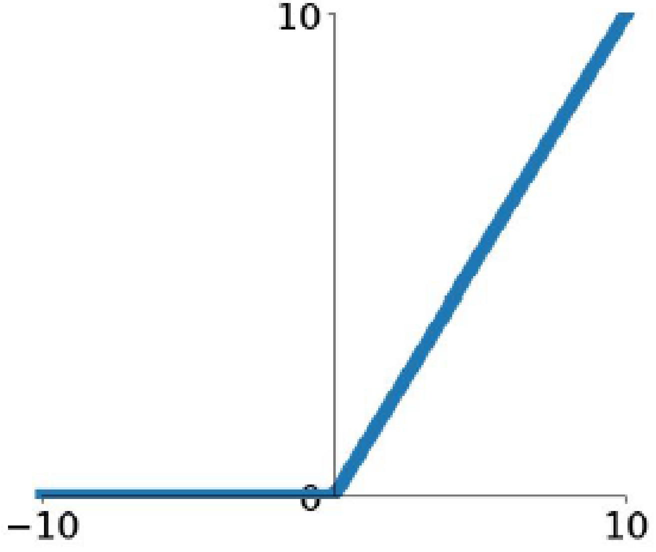
Dying ReLU problem: a large gradient flowing through a ReLU neuron could cause the weights to update in such a way that the neuron will never activate on any datapoint again.

[Krizhevsky et al., 2012]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: RELU





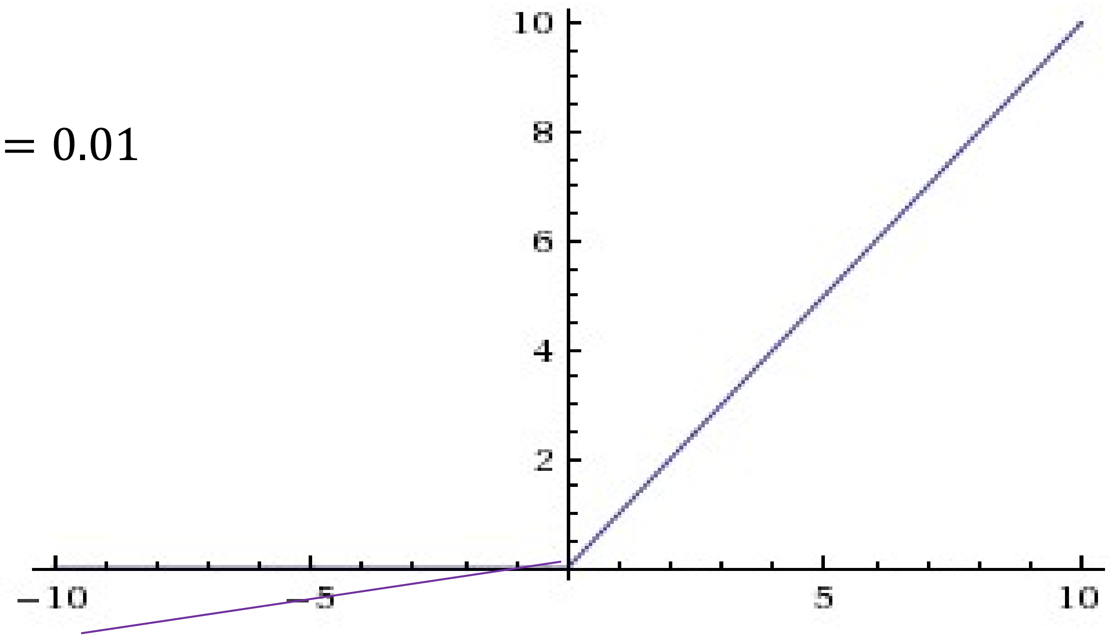
What happens when x = -10?
What happens when x = 0?
What happens when x = 10?

Source: cs231n, Stanford University

ACTIVATION FUNCTIONS: LEAKY RELU

$$f(x) = \begin{cases} \alpha x, & x < 0 \\ x, & x \geq 0 \end{cases}$$

$\alpha = 0.01$



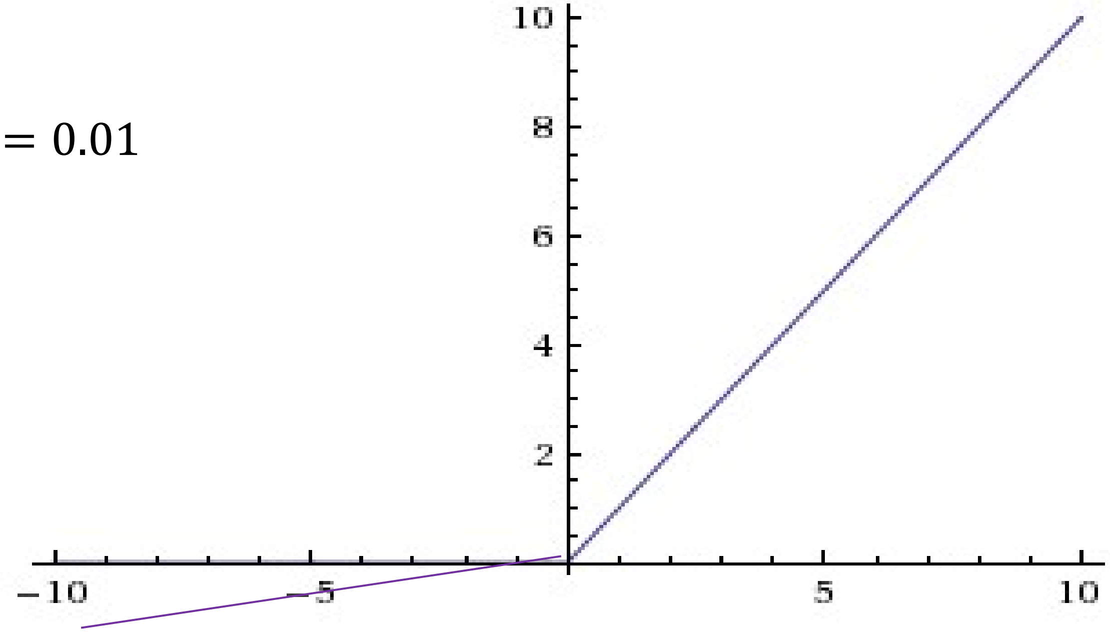
[Mass et al., 2013]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: LEAKY RELU

$$f(x) = \begin{cases} \alpha x, & x < 0 \\ x, & x \geq 0 \end{cases}$$

$\alpha = 0.01$

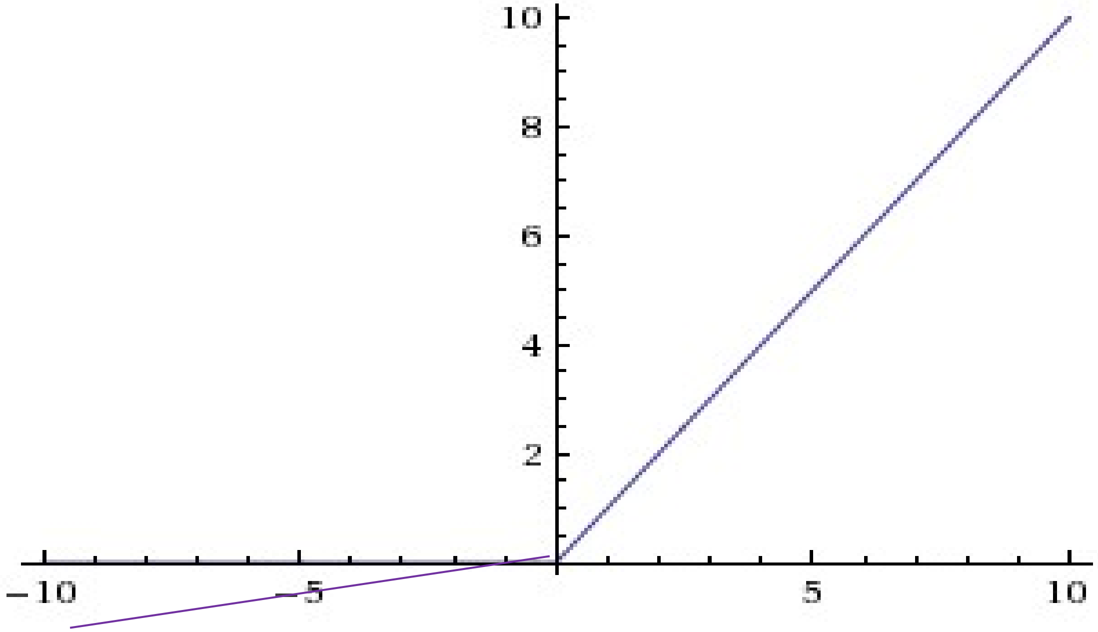


Succeeded in some cases, but the results are not always consistent.

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: PARAMETRIC RELU

$$f(x) = \begin{cases} \alpha x, & x < 0 \\ x, & x \geq 0 \end{cases}$$



In PReLU, the slope in the negative region is considered as a parameter of each neuron and learnt from data.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *IEEE international conference on computer vision* (CVPR).

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: MAXOUT

Maxout neuron (introduced by Goodfellow et al.) generalizes the ReLU and its leaky version.

The Maxout neuron computes the function:

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

[Goodfellow et al., 2013]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: MAXOUT

Maxout neuron (introduced by Goodfellow et al.) generalizes the ReLU and its leaky version.

The Maxout neuron computes the function:

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

Both ReLU and Leaky ReLU are a special case of this form (for example, for ReLU, we have w1=0,b1=0, w2=identity, and b2=0).

[Goodfellow et al., 2013]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: MAXOUT

Maxout neuron (introduced by Goodfellow et al.) generalizes the ReLU and its leaky version.

The Maxout neuron computes the function:

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

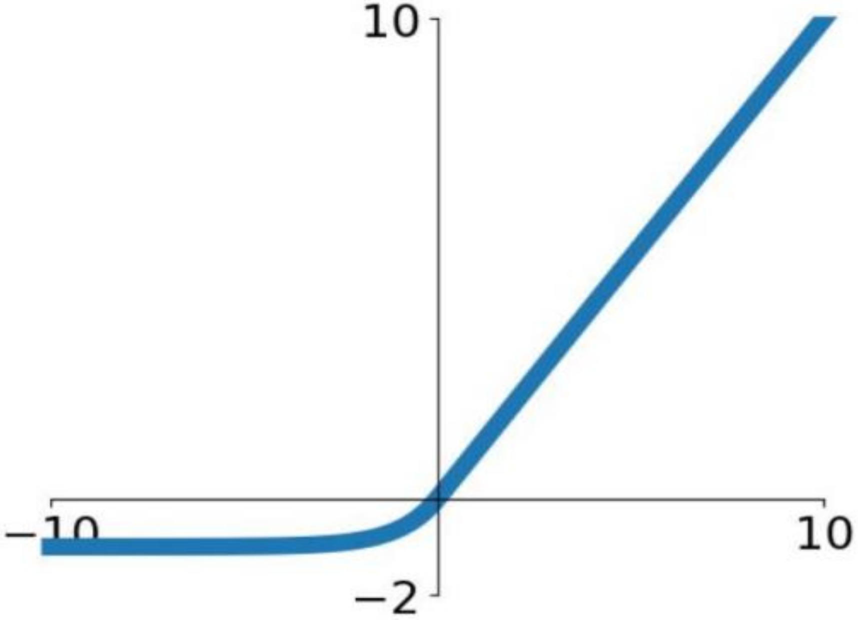
Both ReLU and Leaky ReLU are a special case of this form (for example, for ReLU, we have w1=0,b1=0, w2=identity, and b2=0).

Unlike the ReLU neurons it doubles the number of parameters.

[Goodfellow et al., 2013]

Source: http://cs231n.github.io

ACTIVATION FUNCTIONS: ELU

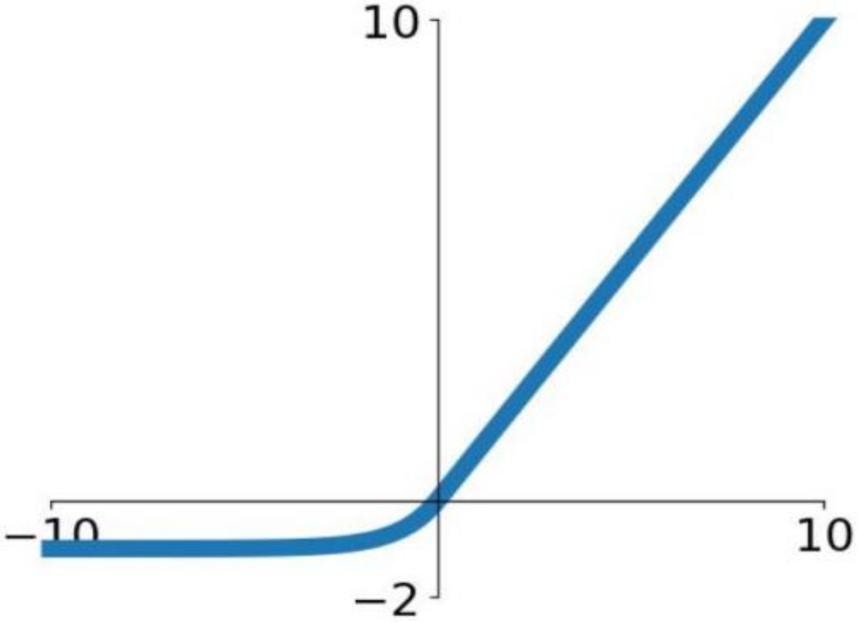


$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \le 0 \end{cases}$$

- Exponential Linear Unit

Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)." International Conference on Learning Representations (ICLR) *2016*.

ACTIVATION FUNCTIONS: ELU



$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \leq 0 \end{cases}$$

- Exponential Linear Unit
- All benefits of ReLU
- Negative saturation regime compared with Leaky ReLU adds some robustness to noise

- Computation requires exp()

Clevert, Djork-Arné, Thomas Unterthiner, and Sepp Hochreiter. "Fast and accurate deep network learning by exponential linear units (elus)." International Conference on Learning Representations (ICLR) *2016*.

ACTIVATION FUNCTIONS: SELU

Scaled Exponential Linear Unit (SELU)

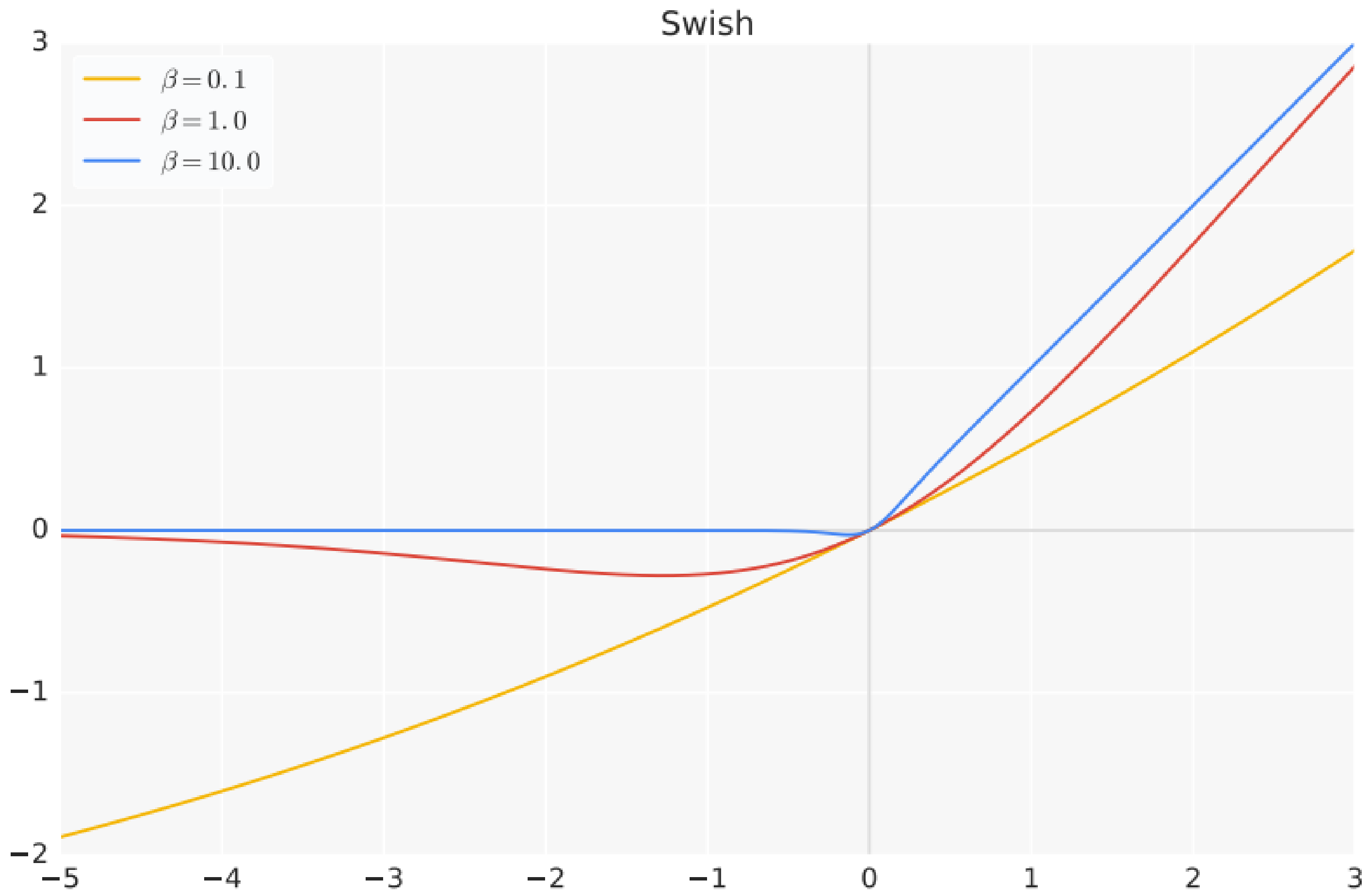
$$f(x) = \lambda \begin{cases} x & \text{if } x \geq 0 \\ \alpha(\exp(x) - 1) & \text{if } x < 0 \end{cases}$$

with $\alpha \approx 1.6733$ and $\lambda \approx 1.0507$.

SELU induces self-normalization to automatically converge towards zero mean and unit variance

Klambauer, Günter, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. "Self-normalizing neural networks." In *Advances in Neural Information Processing Systems* (NIPS), 2017.

ACTIVATION FUNCTIONS: SWISH

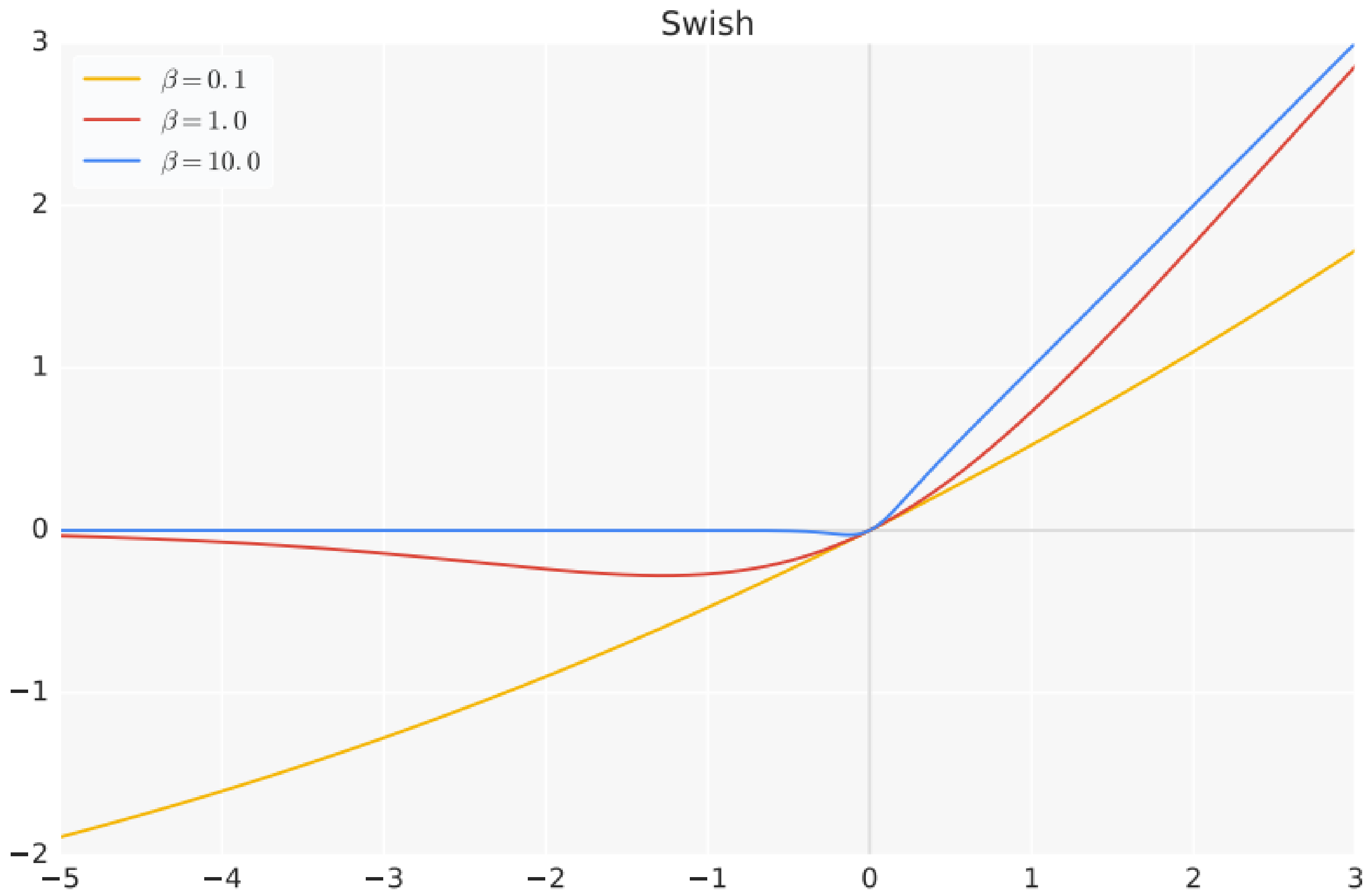


$$f(x) = x \cdot \text{sigmoid}(\beta x)$$

- ReLU is special case of Swish

Ramachandran et al. "Swish: a self-gated activation function." *ICLR Workshops*, 2018.

ACTIVATION FUNCTIONS: SWISH



$$f(x) = x \cdot \text{sigmoid}(\beta x)$$

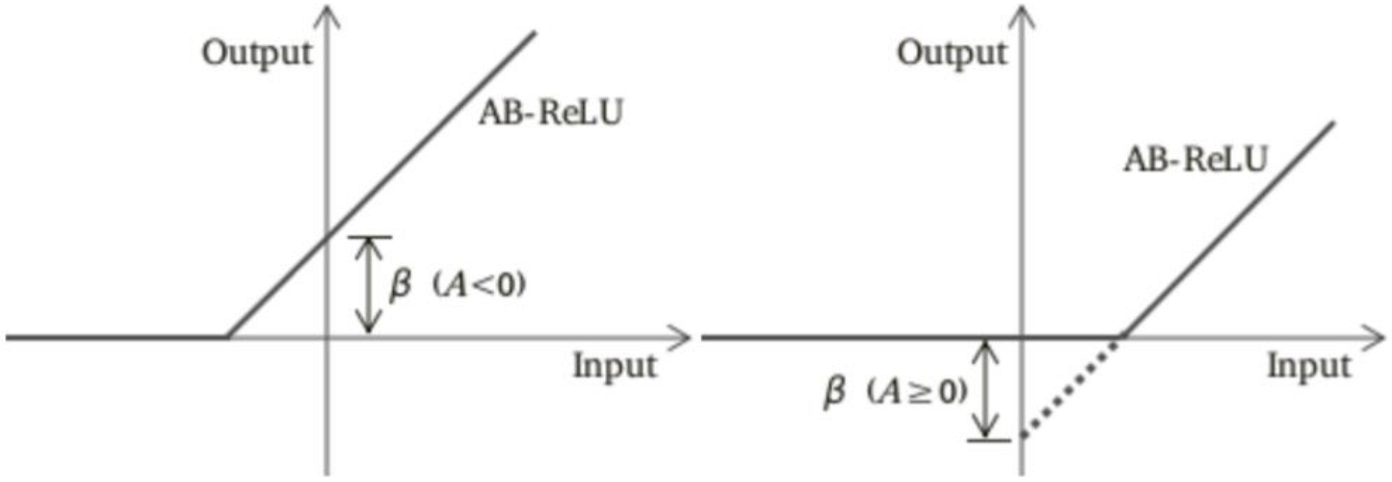
- ReLU is special case of Swish

CIFAR-10 accuracy

Model	ResNet	WRN	DenseNet
LReLU	94.2	95.6	94.7
PReLU	94.1	95.1	94.5
Softplus	94.6	94.9	94.7
ELU	94.1	94.1	94.4
SELU	93.0	93.2	93.9
GELU	94.3	95.5	94.8
ReLU	93.8	95.3	94.8
Swish-1	94.7	95.5	94.8
Swish	94.5	95.5	94.8

Ramachandran et al. "Swish: a self-gated activation function." *ICLR Workshops*, 2018.

ACTIVATION FUNCTIONS: ABRELU



(a) AB-ReLU if $A_v^n < 0$ (b) AB-ReLU if $A_v^n \geq 0$

$$I_v^{n+1}(\rho) = \begin{cases} I_v^n(\rho) - \beta, & \text{if } I_v^n(\rho) - \beta > 0 \\ 0, & \text{otherwise} \end{cases}$$

$$\beta = \alpha \times A_v^n$$

average of input volume

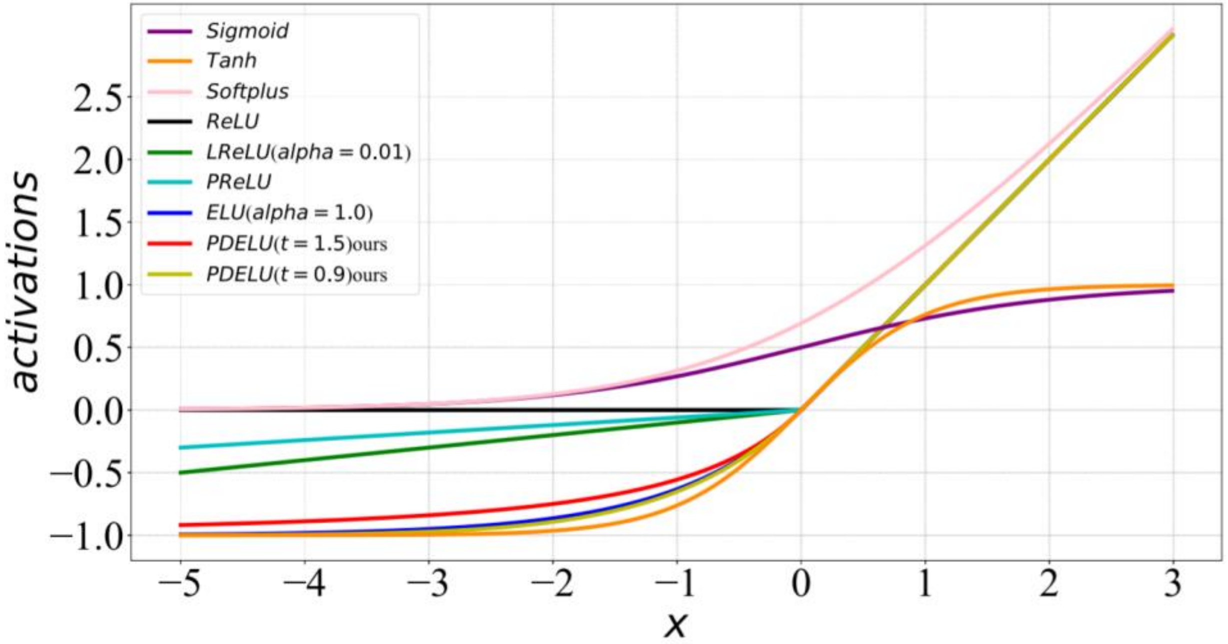
Average Biased ReLU (ABReLU)

S.R. Dubey and S. Chakraborty (2020). Average Biased ReLU Based CNN Descriptor for Improved Face Retrieval. Multimedia Tools and Applications. (Springer)

ACTIVATION FUNCTIONS: PDELU

$$f(x_i) = \begin{cases} x_i & \text{if } x_i > 0 \\ \alpha_i \cdot \left([1 + (1 - t)x_i]^{\frac{1}{1-t}} - 1\right) & \text{if } x_i \le 0 \end{cases}$$

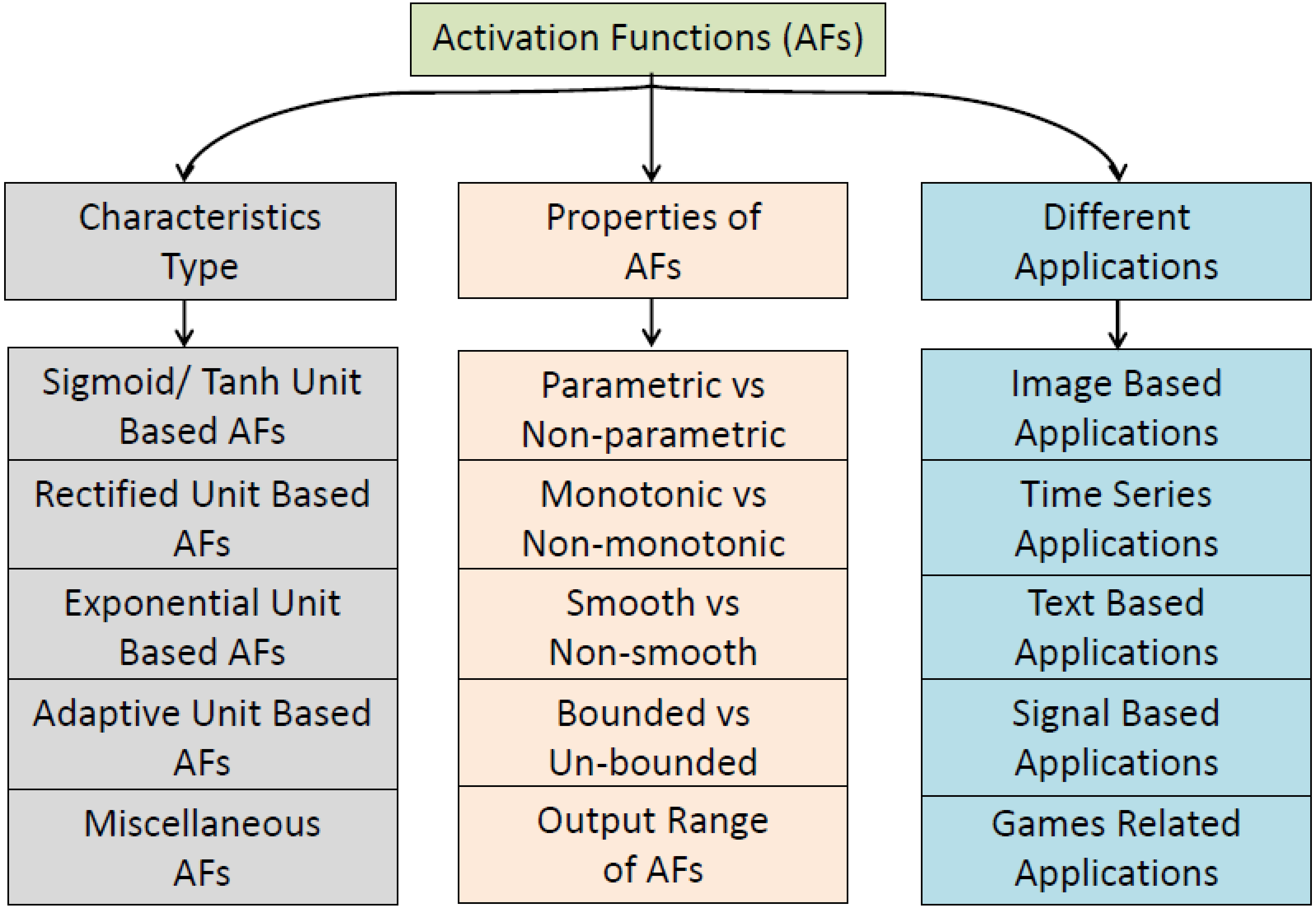
1. When $x_i \geqslant 0$, $f(x_i) = x_i$, so, $f(x_i) \in [0, +\infty]$.
2. When $x_i < 0$ and $\lim t \to -\infty$, $f(x_i) = \alpha \cdot ([1+(1-t)x_i]^{\frac{1}{1-t}} - 1)$ and $f(x_i)$ is monotonically increasing exponentially. So, $f(x_i) \in (-\alpha, 0]$.



Parametric Deformable Exponential Linear Units (PDELU)

Cheng, Q., Li, H., Wu, Q., Ma, L., & King, N. N. (2020). Parametric Deformable Exponential Linear Units for deep neural networks. *Neural Networks*.

ACTIVATION FUNCTIONS: CLASSIFICATION



Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark, Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri, *Neurocomputing*, 503:92-108, Sept 2022. (Elsevier)

ACTIVATION FUNCTIONS: IN PRACTICE

- Use ReLU. Be careful with your learning rates
- Try out PDELU/ABReLU/Swish/

- Try out Leaky ReLU but performance might not be stable
- Try out tanh but don't expect much
- Don't use sigmoid

ACKNOWLEDGEMENT

- Deep Learning, Stanford University
- Introduction to Deep Learning, University of Illinois at Urbana-Champaign
- Introduction to Deep Learning, Carnegie Mellon University
- Convolutional Neural Networks for Visual Recognition, Stanford University
- Natural Language Processing with Deep Learning, Stanford University
- NVDIEA Deep Learning Teaching Kit