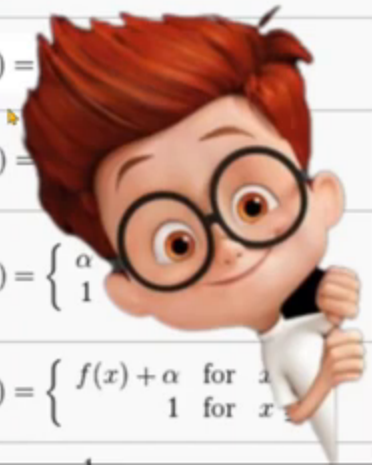
Binary step	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)	$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH	$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan	$f(x) = \tan^{-1}(x)$	$f'(x) =$
Rectified Linear Unit (ReLU)	$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) =$
Parameteric Rectified Linear Unit (PReLU)[2]	$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha \\ 1 \end{cases}$
Exponential Linear Unit (ELU)[3]	$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x \\ 1 & \text{for } x \end{cases}$

10 activations in 10 mins



Functions Covered

- Step
- Sigmoid
- Tanh
- ReLU
- Softplus
- Maxout
- GeLU
- Swish
- Mish
- Softmax

Functions Covered

- Step
- Sigmoid
- Tanh
- ReLU
- Softplus
- Maxout
- GeLU
- Swish
- Mish
- Softmax

Topics Covered

- Motivation
- Definition
- Graph
- Derivative
- Properties
- Performance
- Use cases

Topics Covered

- Motivation
- Definition
- Graph
- Derivative
- Properties
- Performance
- Use cases

Step Activation Function

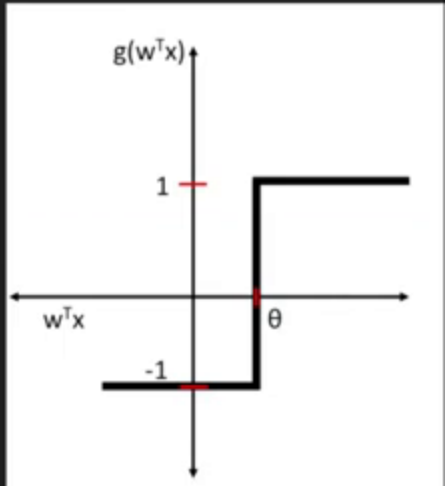
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



Unit step function.

Step Activation Function

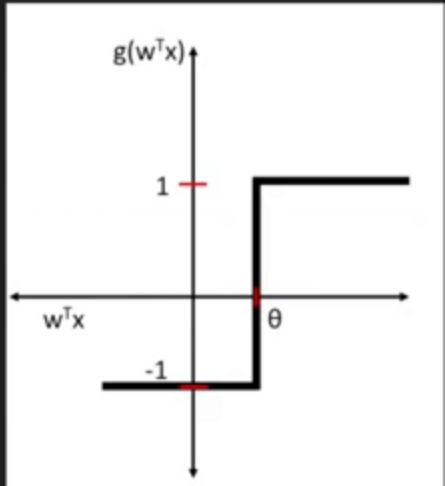
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



Unit step function.

Step Activation Function

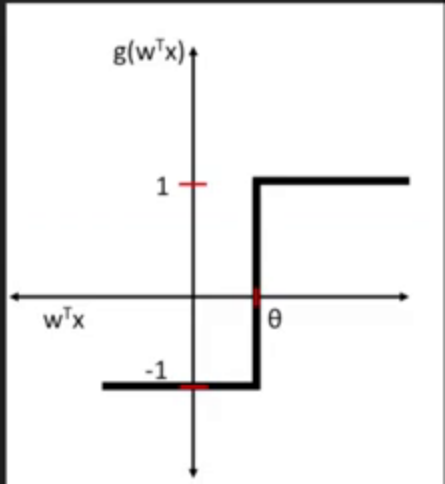
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



Unit step function.

Step Activation Function

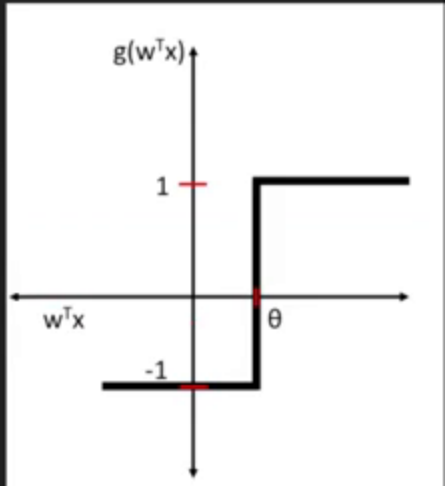
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



Unit step function.

Step Activation Function

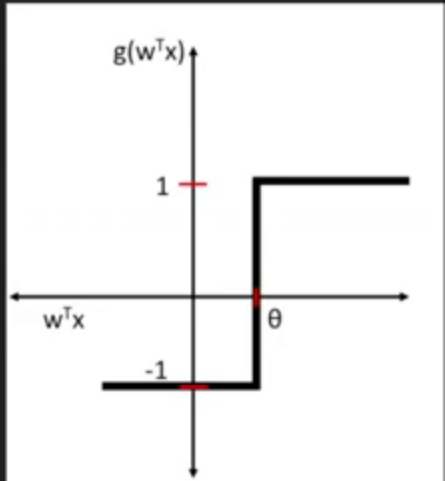
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



Unit step function.

Step Activation Function

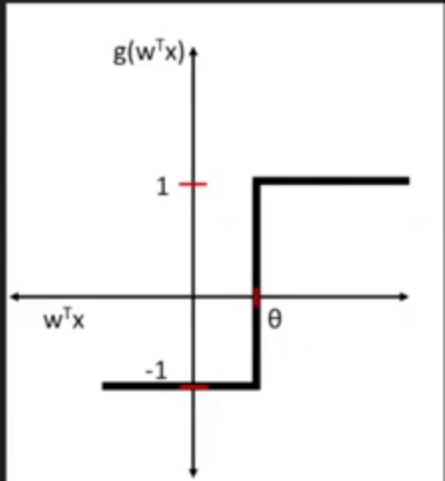
- Motivation: Neurons firing in Human brain
- Definition:

$$f(x) = \begin{cases} 0 & for\ x < 0 \\ 1 & for\ x \geqslant 0 \end{cases}$$

- Derivative:

```
f'(x) = 0, for all x
```

- Properties:
  - Not-continuous
  - Bounded
  - Not 0-centered
- Performance: Poor
- Use case: Hidden & Output layers



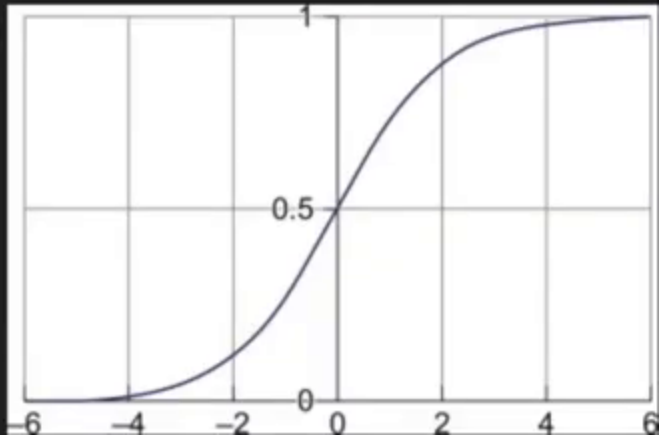
Unit step function.

Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

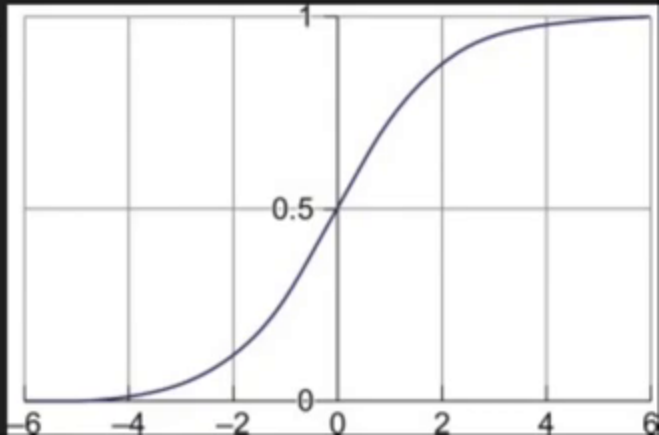


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

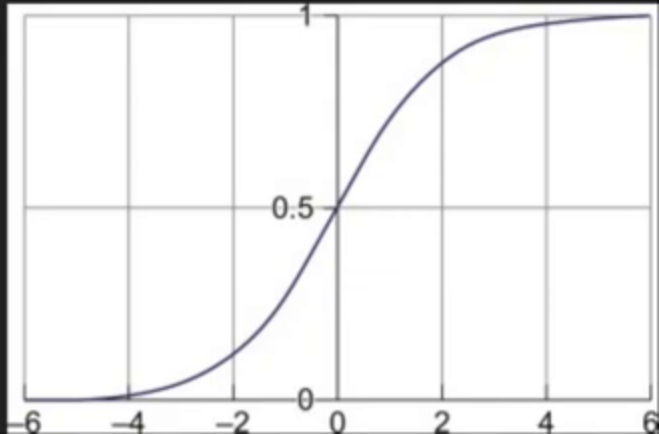


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

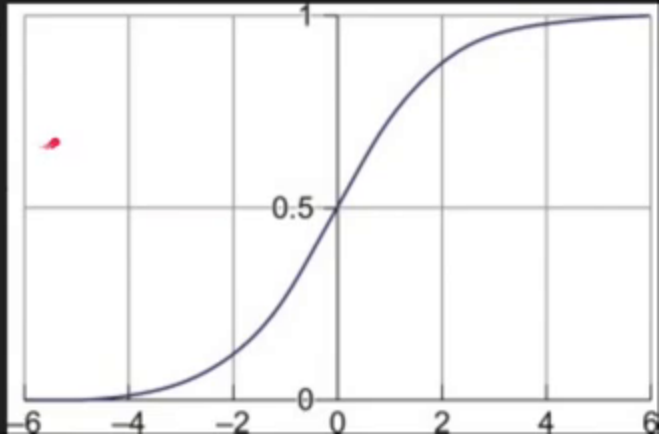


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

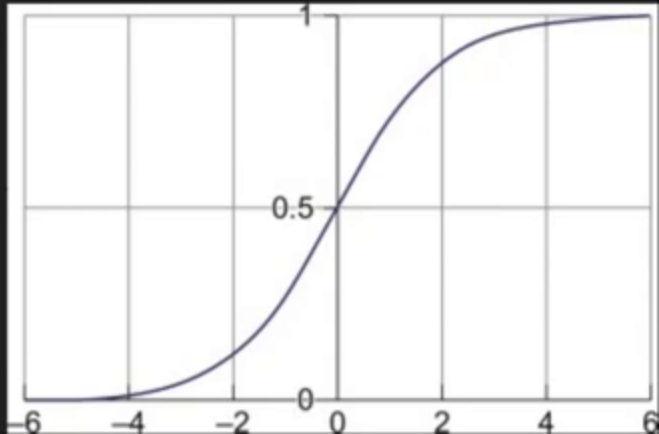


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

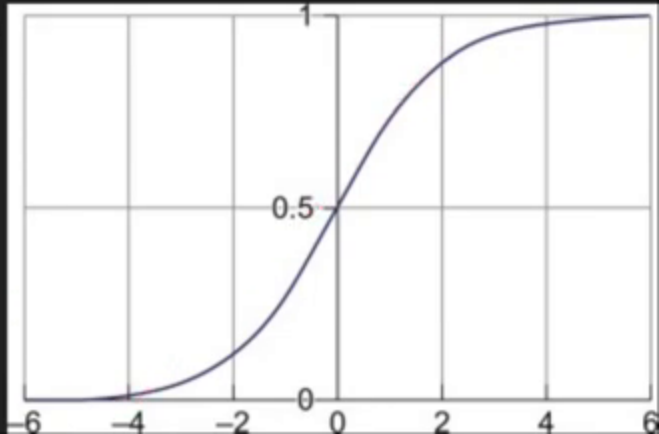


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

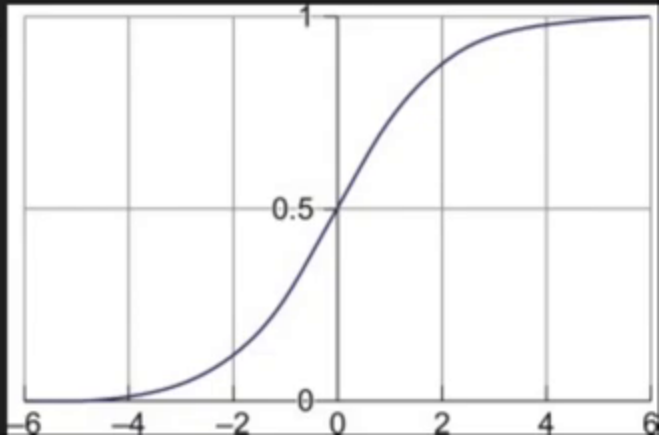


Sigmoid Activation Function

- Motivation: Smooth curve and probabilistic output
- Definition:

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

- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers

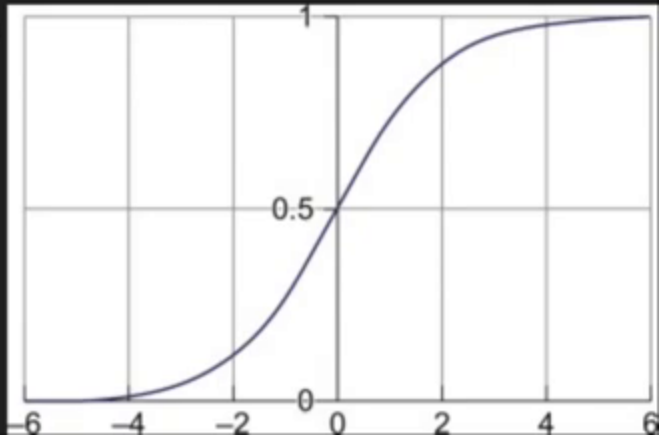


Sigmoid Activation Function

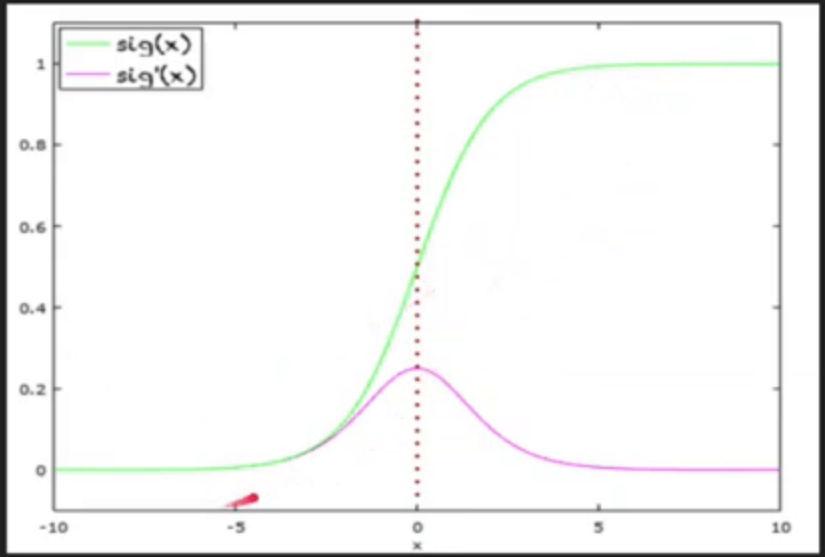
- Motivation: Smooth curve and probabilistic output
- Definition:

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

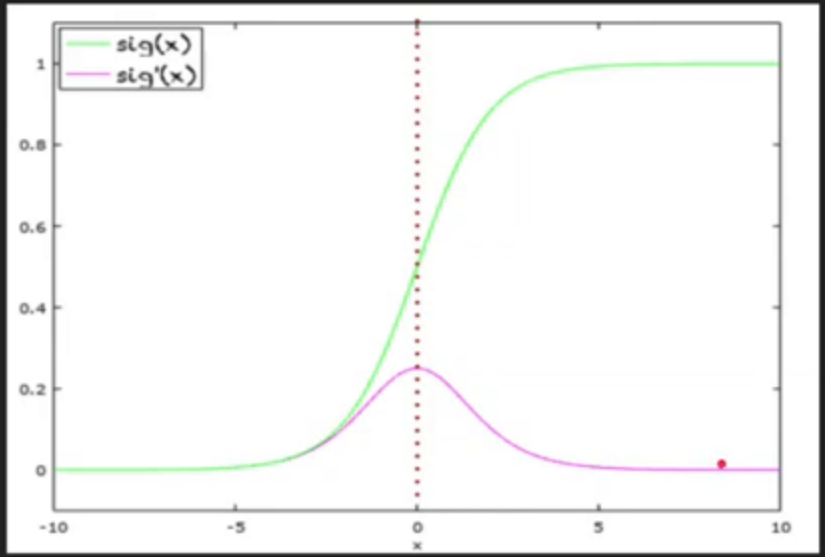
- Derivative: $f'(x) = \sigma(x) \cdot (1 - \sigma(x))$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - Not 0-centered
- Performance: Avg
- Use Case: hidden & output layers



Vanishing Gradient



Vanishing Gradient



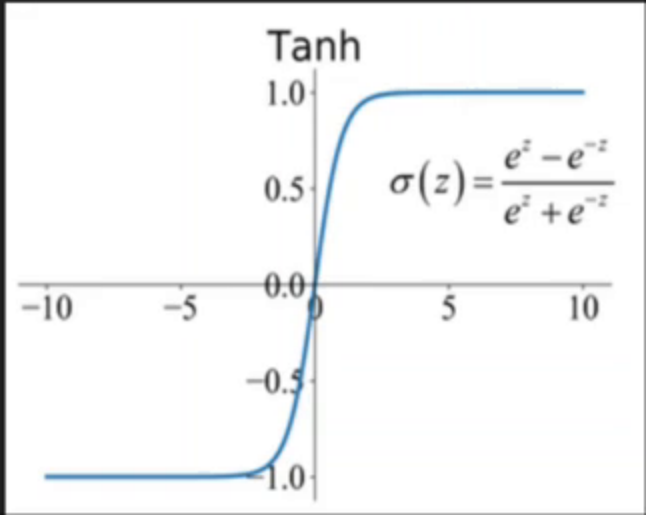
Tanh Activation Function

- Motivation: Smooth curve and probabilistic output, 0-centered function
- Definition:

Tanh Function

$$a = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Derivative: $1 - (\tanh(x))^2$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - 0-centered
- Performance: Better than sigmoid
- Use Case: hidden layers



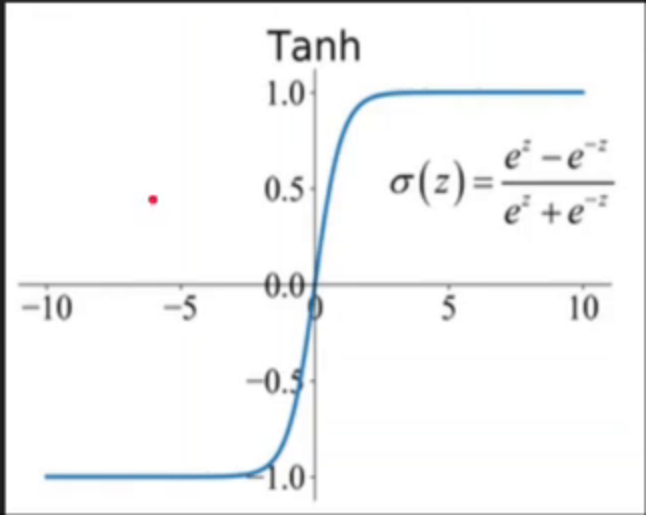
Tanh Activation Function

- Motivation: Smooth curve and probabilistic output, 0-centered function
- Definition:

Tanh Function

$$a = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Derivative: $1 - (\tanh(x))^2$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - 0-centered
- Performance: Better than sigmoid
- Use Case: hidden layers



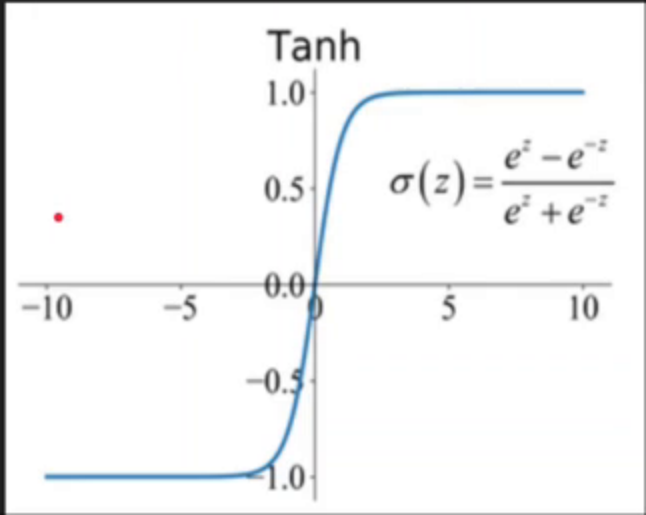
Tanh Activation Function

- Motivation: Smooth curve and probabilistic output, 0-centered function
- Definition:

Tanh Function

$$a = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Derivative: $1 - (\tanh(x))^2$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - 0-centered
- Performance: Better than sigmoid
- Use Case: hidden layers



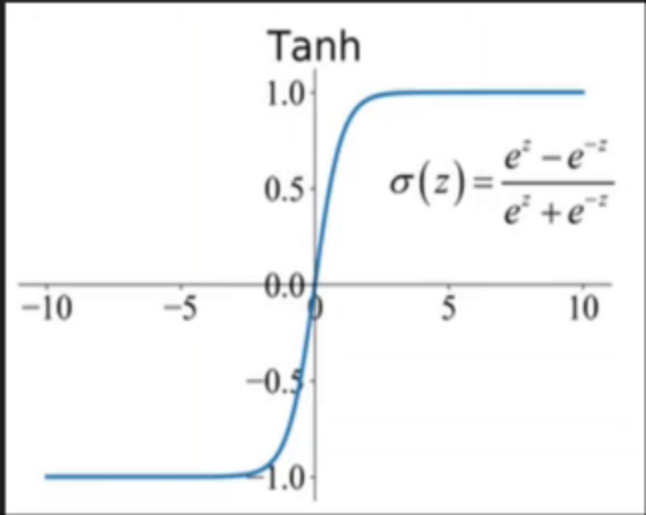
Tanh Activation Function

- Motivation: Smooth curve and probabilistic output, 0-centered function
- Definition:

Tanh Function

$$a = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Derivative: $1 - (\tanh(x))^2$
- Properties:
  - Continuous
  - Non-linear
  - Monotonic
  - Bounded
  - 0-centered
- Performance: Better than sigmoid
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

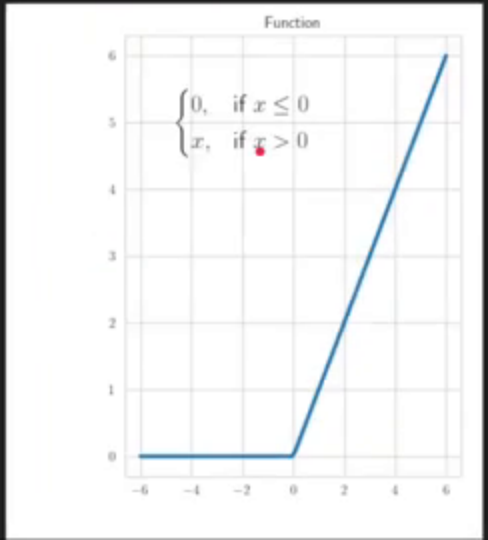
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

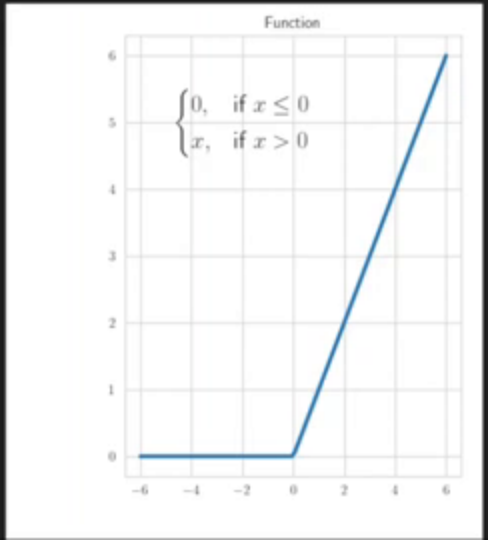
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

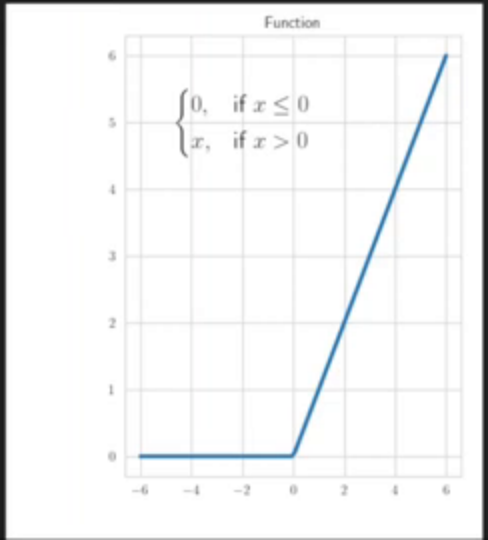
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

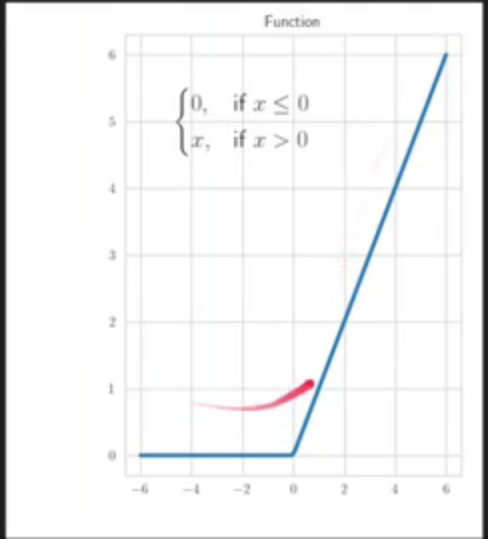
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

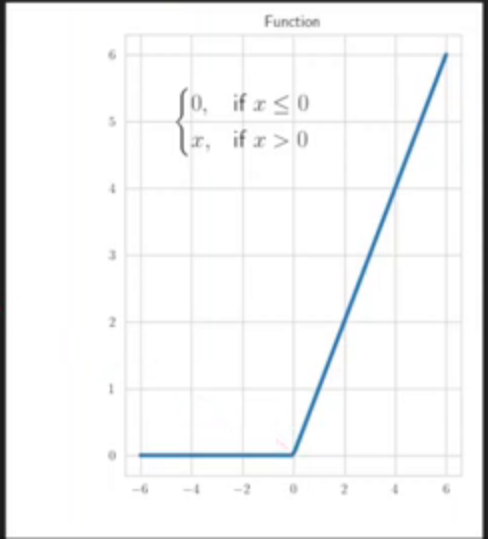
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

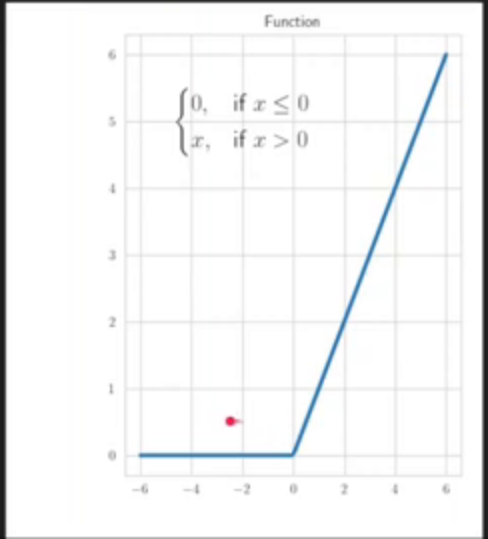
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

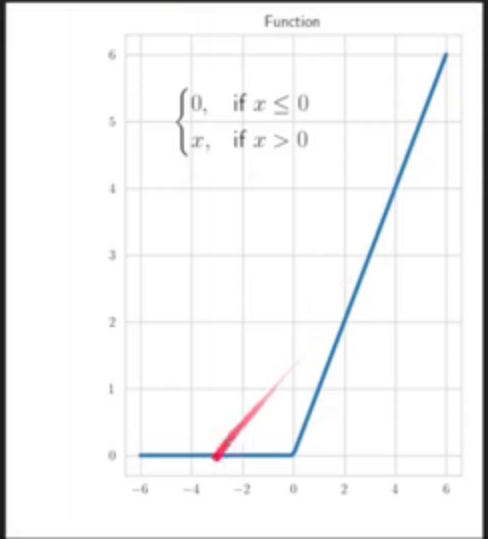
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

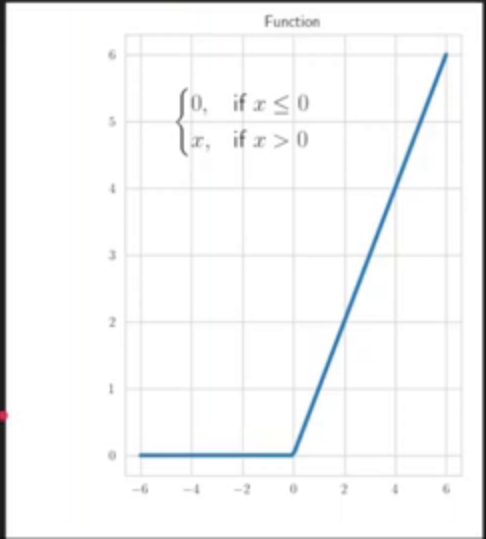
$$ReLU$$

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

- Derivative:

$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



ReLU Activation Function

- Motivation: Solve vanishing gradient problem, linear-like function
- Definition:

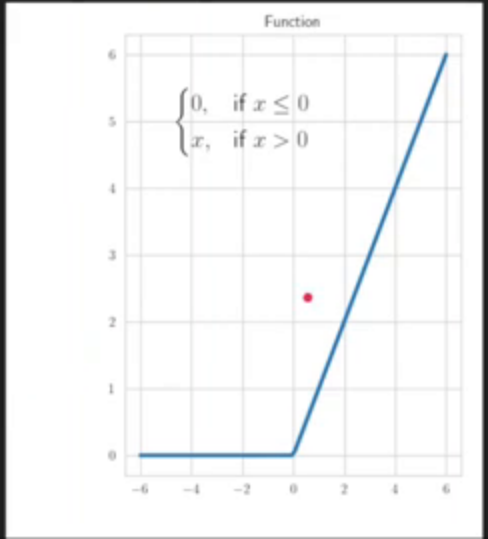
$$ReLU$$

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

- Derivative:

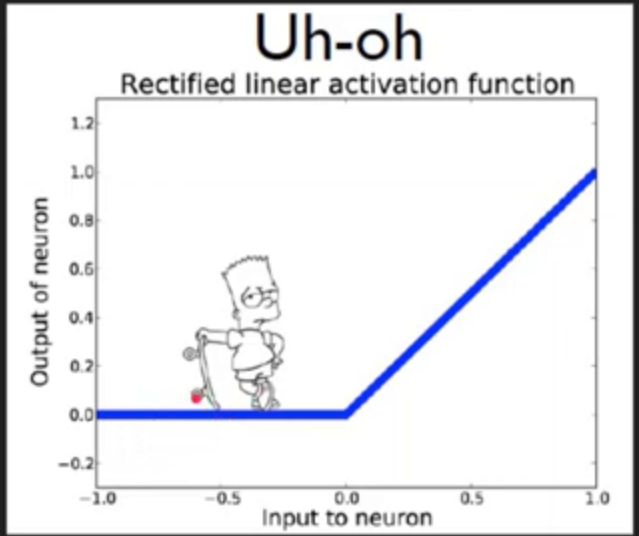
$$f'(x) = 1\ for\ x > 0$$
$$= 0\ for\ x < 0$$
$$= NA\ for\ x = 0$$

- Properties:
  - Not-Continuous
  - Piecewise linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Better than sigmoid, tanh
- Use Case: hidden layers



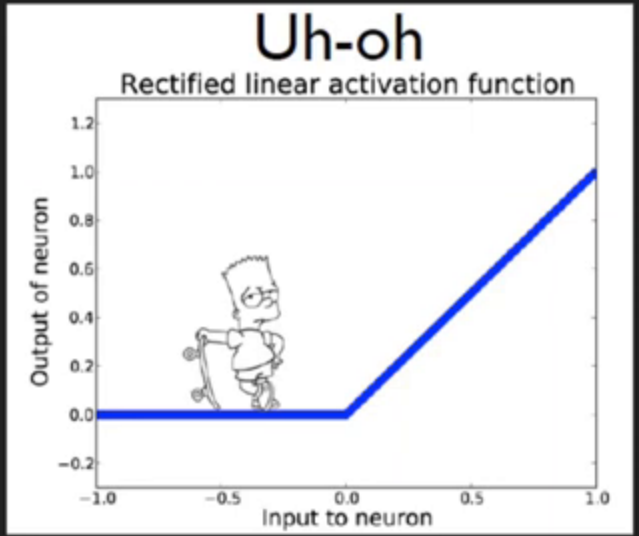
Dead Neurons

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \Rightarrow 0 \end{cases}$$



Dead Neurons

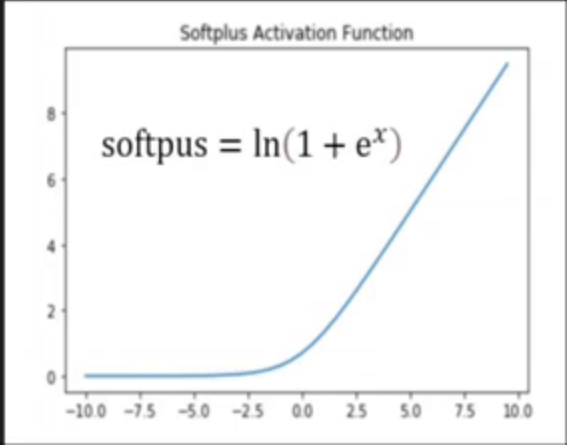
$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \Rightarrow 0 \end{cases}$$



Softplus Activation Function

- Motivation: ReLU-like function but smooth curve
- Definition: $f(x) = \ln(1 + e^x)$

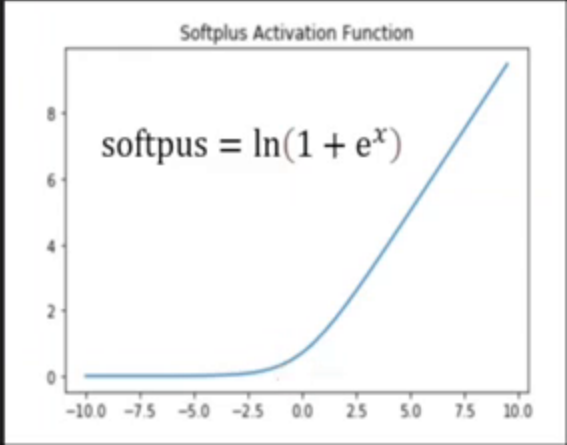
- Derivative: $dy/dx = 1 / (1 + e^{-x})$
- Properties:
  - Continuous
  - Non-Linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Same as relu, but unstable
- Use Case: hidden layers



Softplus Activation Function

- Motivation: ReLU-like function but smooth curve
- Definition: $f(x) = \ln(1 + e^x)$

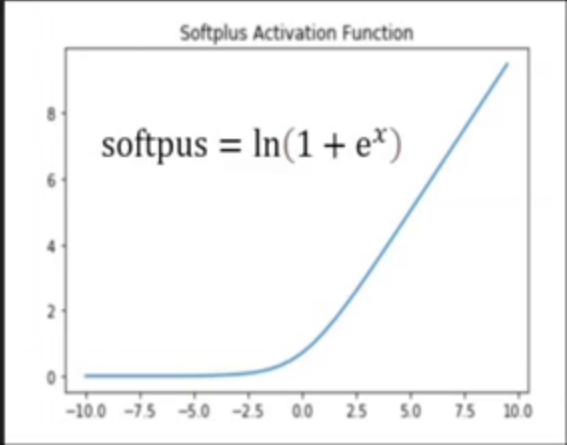
- Derivative: $dy/dx = 1 / (1 + e^{-x})$
- Properties:
  - Continuous
  - Non-Linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Same as relu, but unstable
- Use Case: hidden layers



Softplus Activation Function

- Motivation: ReLU-like function but smooth curve
- Definition: $f(x) = \ln(1 + e^x)$

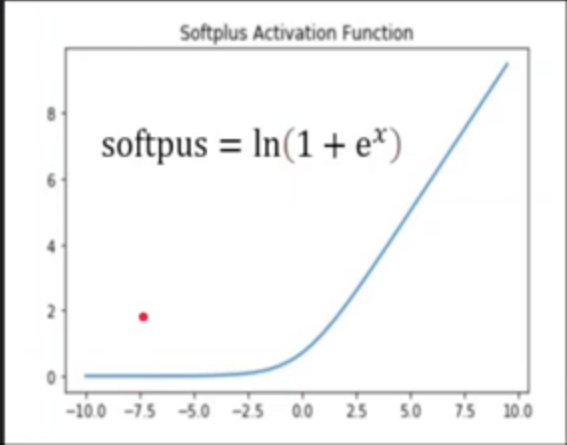
- Derivative: $dy/dx = 1 / (1 + e^{-x})$
- Properties:
  - Continuous
  - Non-Linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Same as relu, but unstable
- Use Case: hidden layers



Softplus Activation Function

- Motivation: ReLU-like function but smooth curve
- Definition: $f(x) = \ln(1 + e^x)$

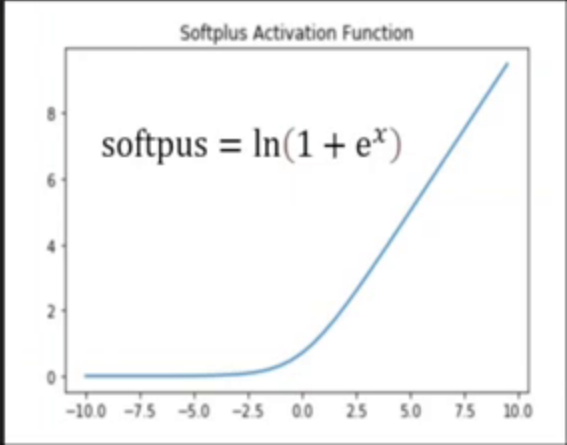
- Derivative: $dy/dx = 1 / (1 + e^{-x})$
- Properties:
  - Continuous
  - Non-Linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Same as relu, but unstable
- Use Case: hidden layers



Softplus Activation Function

- Motivation: ReLU-like function but smooth curve
- Definition: $f(x) = \ln(1 + e^x)$

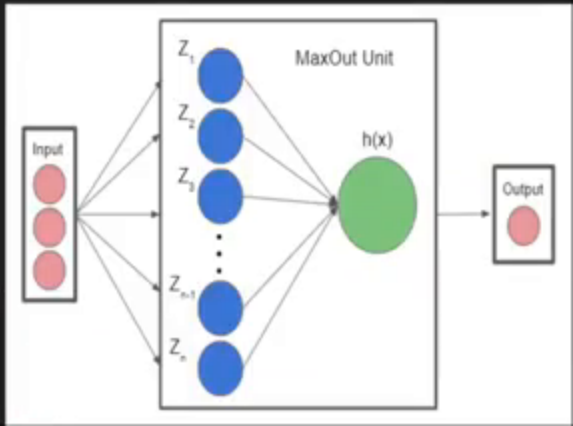
- Derivative: $dy/dx = 1 / (1 + e^{-x})$
- Properties:
  - Continuous
  - Non-Linear
  - Monotonic
  - UnBounded
  - Not 0-centered
- Performance: Same as relu, but unstable
- Use Case: hidden layers



Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

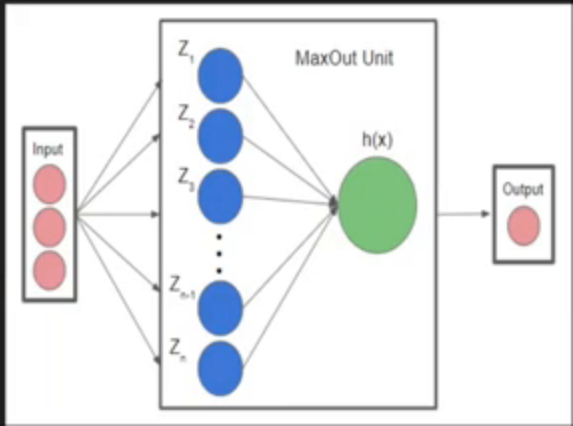
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

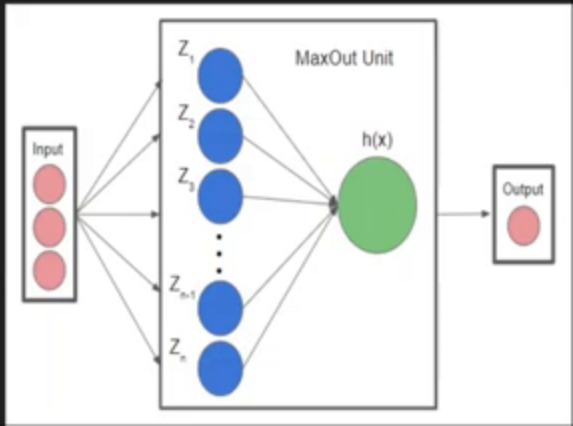
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

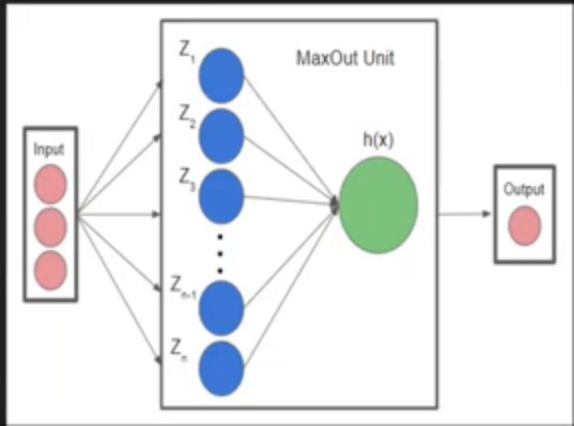
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

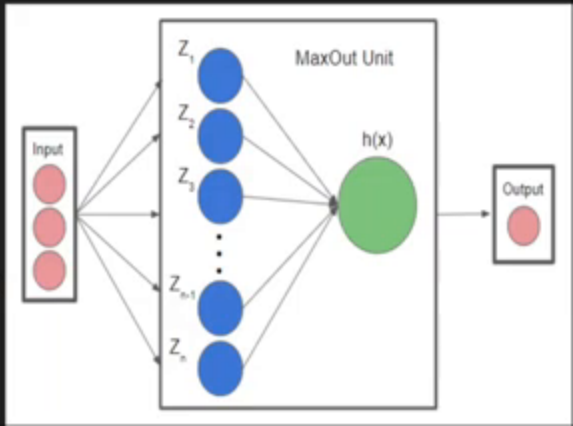
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

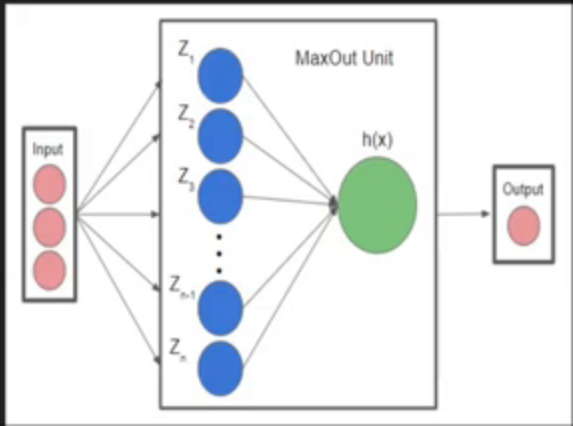
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



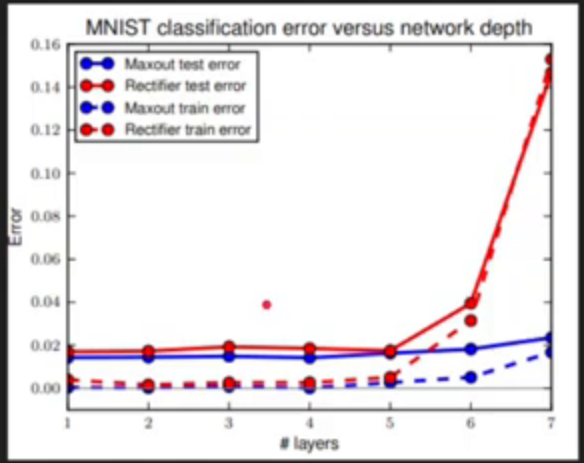
Maxout Activation Function

- Motivation: Solve dead neurons issue
- Definition: $$h_i(x) = \max_{j \in [1,k]} z_{ij}$$

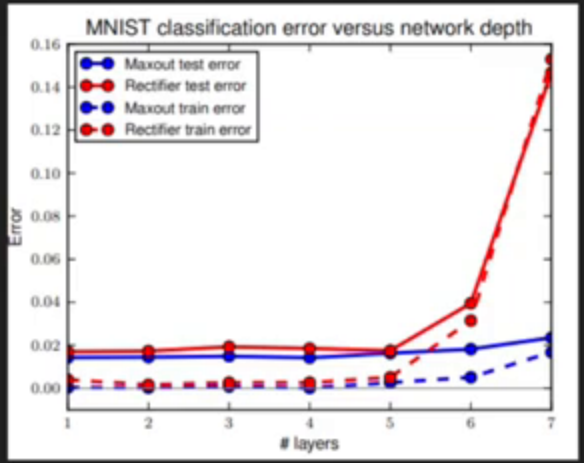
- Derivative: 1 or -1
- Properties:
  - Not-Continuous
  - Piecewise-Linear
  - UnBounded
  - 0-centered
- Performance: Better than relu, but complex
- Use Case: hidden layers



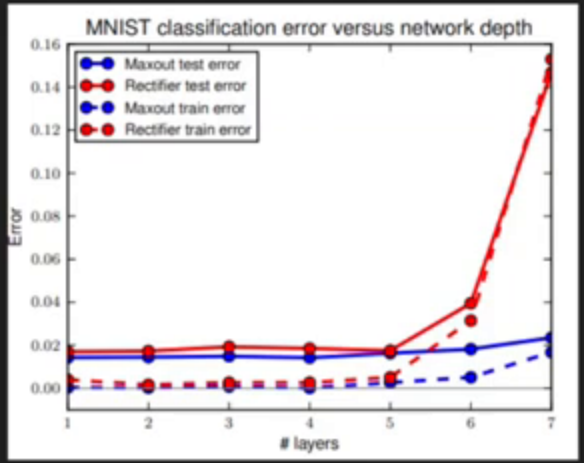
Maxout Activation



Maxout Activation



Maxout Activation

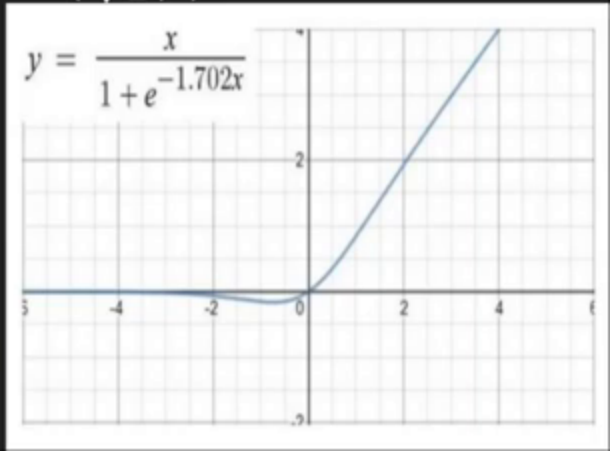


GeLU Activation Function

- Motivation: Combine ReLU & Dropout
- Definition:

$$GeLU(x) = \frac{1}{2}x\left(1 + erf\left(\frac{x}{\sqrt[2]{2}}\right)\right)$$

- Derivative: $\phi(x) + xP(X = x)$
- Properties:
  - Continuous
  - Non-Linear
  - UnBounded
  - 0-centered
  - Non-monotonic
- Performance: Better than relu, Complex
- Use Case: hidden layers

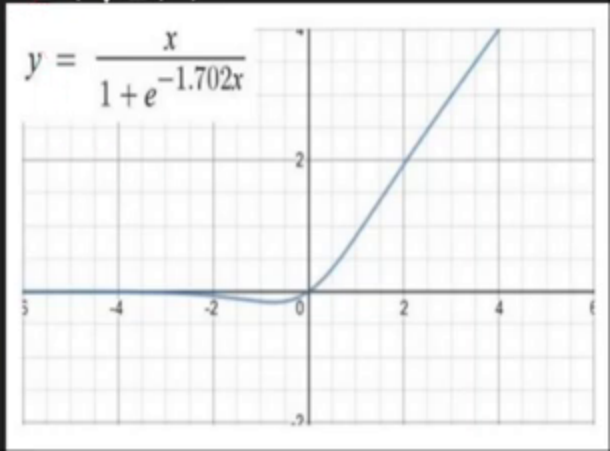


GeLU Activation Function

- Motivation: Combine ReLU & Dropout
- Definition:

$$GeLU(x) = \frac{1}{2}x\left(1 + erf\left(\frac{x}{\sqrt[2]{2}}\right)\right)$$

- Derivative: $\phi(x) + xP(X = x)$
- Properties:
  - Continuous
  - Non-Linear
  - UnBounded
  - 0-centered
  - Non-monotonic
- Performance: Better than relu, Complex
- Use Case: hidden layers

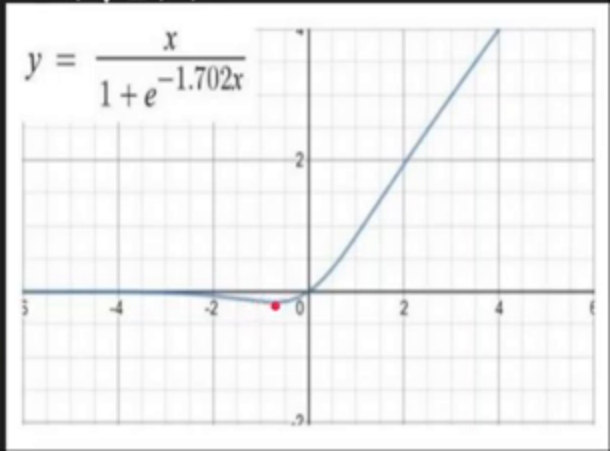


GeLU Activation Function

- Motivation: Combine ReLU & Dropout
- Definition:

$$GeLU(x) = \frac{1}{2}x\left(1 + erf\left(\frac{x}{\sqrt[2]{2}}\right)\right)$$

- Derivative: $\phi(x) + xP(X = x)$
- Properties:
  - Continuous
  - Non-Linear
  - UnBounded
  - 0-centered
  - Non-monotonic
- Performance: Better than relu, Complex
- Use Case: hidden layers

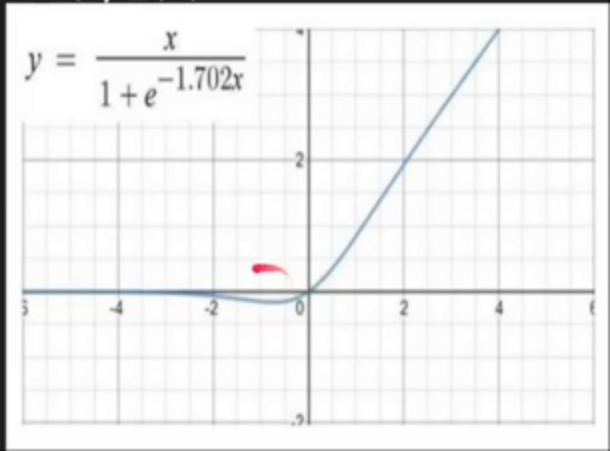


GeLU Activation Function

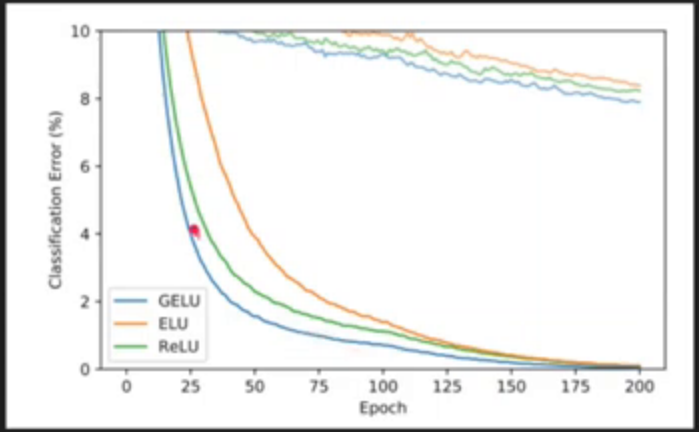
- Motivation: Combine ReLU & Dropout
- Definition:

$$GeLU(x) = \frac{1}{2}x\left(1 + erf\left(\frac{x}{\sqrt[2]{2}}\right)\right)$$

- Derivative: $\phi(x) + xP(X = x)$
- Properties:
  - Continuous
  - Non-Linear
  - UnBounded
  - 0-centered
  - Non-monotonic
- Performance: Better than relu, Complex
- Use Case: hidden layers



GeLU Activation

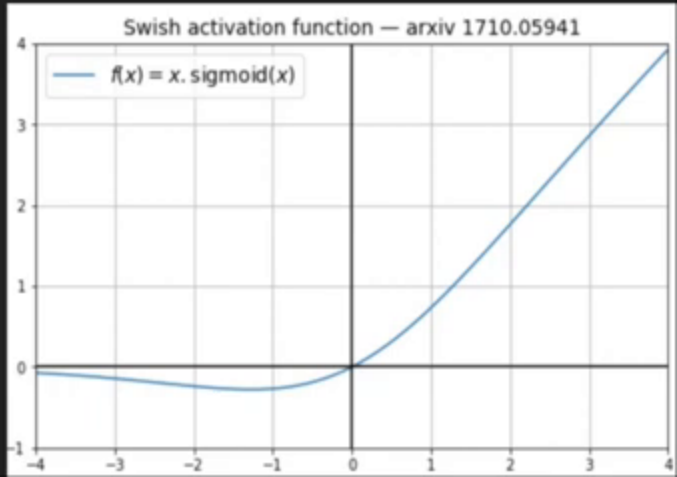


CIFAR-10

Swish Activation Function

- Motivation: Search for better activation using NAS
- Definition: $f(x) = x.\sigma(x)$

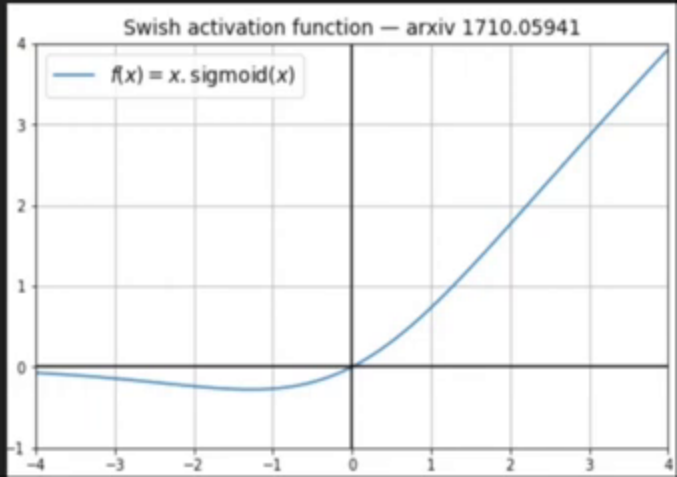
- Derivative: $f(x) + \sigma(x)(1 - f(x))$
- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu
- Use Case: hidden layers



Swish Activation Function

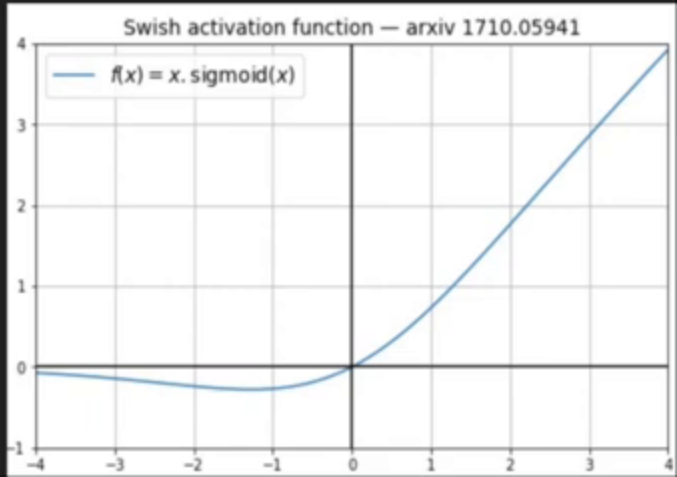
- Motivation: Search for better activation using NAS
- Definition: $f(x) = x.\sigma(x)$

- Derivative: $f(x) + \sigma(x)(1 - f(x))$
- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu
- Use Case: hidden layers



Swish Activation Function

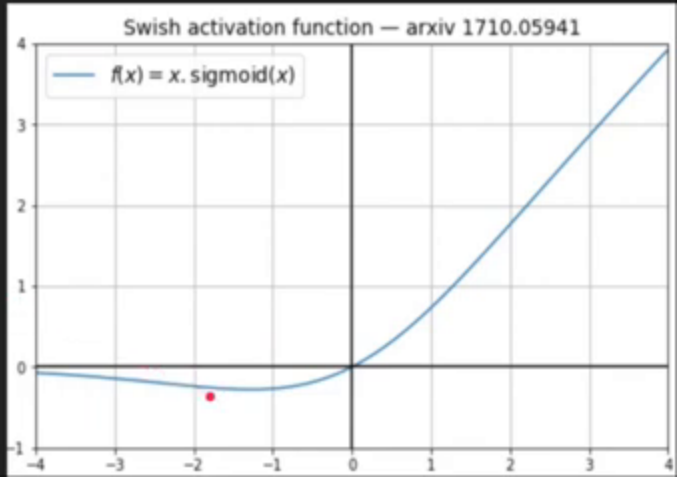
- Motivation: Search for better activation using NAS
- Definition: $f(x) = x.\sigma(x)$
- Derivative: $f(x) + \sigma(x)(1 - f(x))$
- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu
- Use Case: hidden layers



Swish Activation Function

- Motivation: Search for better activation using NAS
- Definition: $f(x) = x.\sigma(x)$

- Derivative: $f(x) + \sigma(x)(1 - f(x))$
- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu
- Use Case: hidden layers



Swish Activation Function

- Motivation: Search for better activation using NAS
- Definition: $f(x) = x.\sigma(x)$

- Derivative: $f(x) + \sigma(x)(1 - f(x))$
- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu
- Use Case: hidden layers

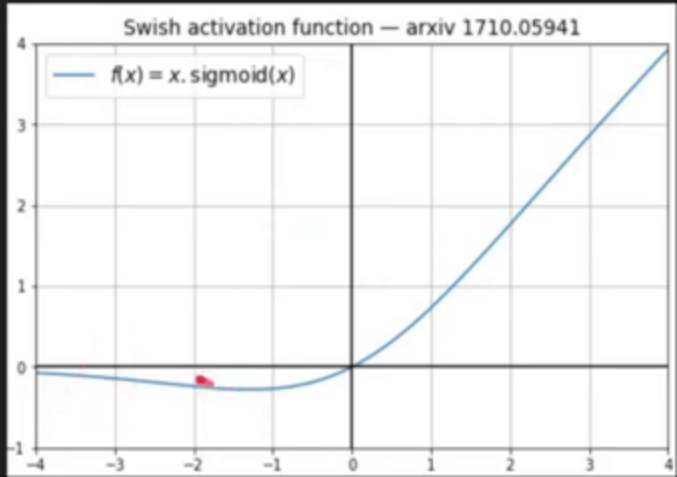


Image Classification

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
ReLU	93.8	95.3	**94.8**
Swish	**94.7**	**95.5**	**94.8**

Table 2: CIFAR-10 accuracy.

Model	ResNet	WRN	DenseNet
LReLU	74.2	78.0	83.3
PReLU	74.5	77.3	81.5
Softplus	**76.0**	**78.4**	**83.7**
ELU	75.0	76.0	80.6
SELU	73.2	74.3	80.8
ReLU	74.2	77.8	**83.7**
Swish	75.1	**78.5**	**83.8**

Table 3: CIFAR-100 accuracy.

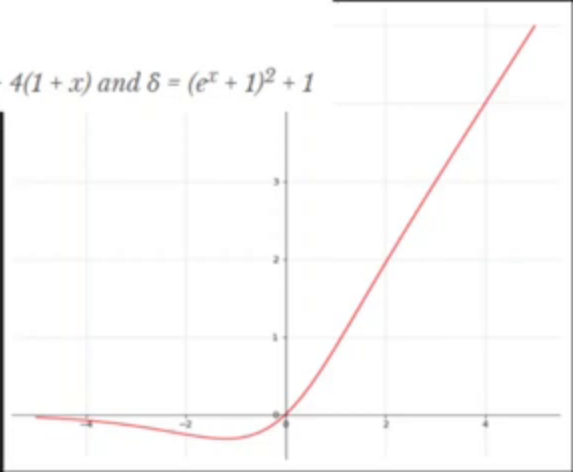
Mish Activation Function

- Motivation: Inspired by Swish
- Definition: $f(x) = x.\tanh\left(\ln\left(1+e^{x}\right)\right)$
- Derivative:

$$dy/dx = e^{x} \,.\, \omega \,/\, \delta^{2}$$

$$\text{whereas } \omega = e^{3x} + e^{2x}\,(4) + e^{x}\,(6+4x) + 4(1+x) \text{ and } \delta = (e^{x}+1)^{2} + 1$$

- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu & swish
- Use Case: hidden layers



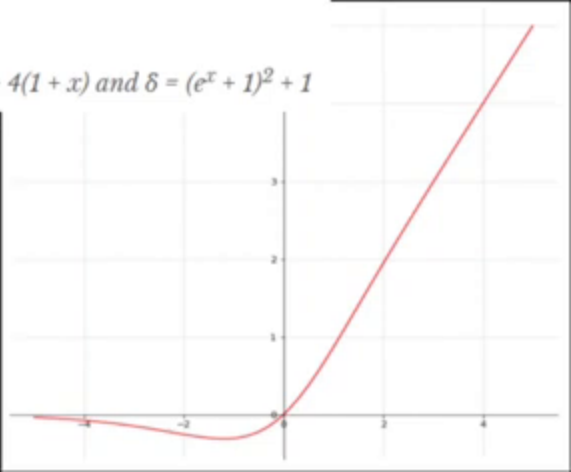
Mish Activation Function

- Motivation: Inspired by Swish
- Definition: $f(x) = x.\tanh\left(\ln\left(1+e^{x}\right)\right)$
- Derivative:

$$dy/dx = e^{x} \cdot \omega / \delta^{2}$$

whereas $\omega = e^{3x} + e^{2x}(4) + e^{x}(6+4x) + 4(1+x)$ and $\delta = (e^{x}+1)^{2} + 1$

- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu & swish
- Use Case: hidden layers



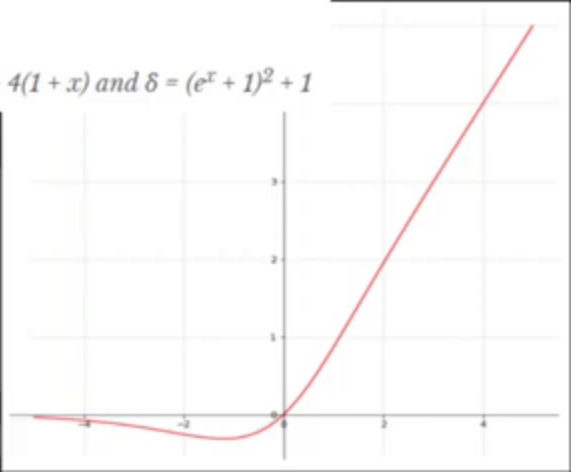
Mish Activation Function

- Motivation: Inspired by Swish
- Definition: $f(x) = x.\tanh\left(\ln\left(1+e^{x}\right)\right)$
- Derivative:

$$dy/dx = e^{x} \cdot \omega / \delta^{2}$$

$$\text{whereas } \omega = e^{3x} + e^{2x}(4) + e^{x}(6+4x) + 4(1+x) \text{ and } \delta = (e^{x}+1)^{2} + 1$$

- Properties:
  - Continuous
  - Non-Linear
  - Non-monotonic
  - UnBounded
  - 0-centered
- Performance: Better than relu & swish
- Use Case: hidden layers



Performance

Table 5. CIFAR-10 Results (Test Top-1 Accuracy)

Model	Mish	Swish	ReLU
ResNet v2-20	**92.02%**	91.61%	91.71%
WRN 10-2	**86.83%**	86.56%	84.56%
SimpleNet	**91.70%**	91.44%	91.16%
Xception Net	**88.73%**	88.56%	88.38%
Capsule Net	**83.15%**	82.48%	82.19%
Inception ResNet v2	**85.21%**	84.96%	82.22%

Performance

Table 5. CIFAR-10 Results (Test Top-1 Accuracy)

Model	Mish	Swish	ReLU
ResNet v2-20	**92.02%**	91.61%	91.71%
WRN 10-2	**86.83%**	86.56%	84.56%
SimpleNet	**91.70%**	91.44%	91.16%
Xception Net	**88.73%**	88.56%	88.38%
Capsule Net	**83.15%**	82.48%	82.19%
Inception ResNet v2	**85.21%**	84.96%	82.22%

Performance

Table 5. CIFAR-10 Results (Test Top-1 Accuracy)

Model	Mish	Swish	ReLU
ResNet v2-20	**92.02%**	91.61%	91.71%
WRN 10-2	**86.83%**	86.56%	84.56%
SimpleNet	**91.70%**	91.44%	91.16%
Xception Net	**88.73%**	88.56%	88.38%
Capsule Net	**83.15%**	82.48%	82.19%
Inception ResNet v2	**85.21%**	84.96%	82.22%

Softmax Activation Function

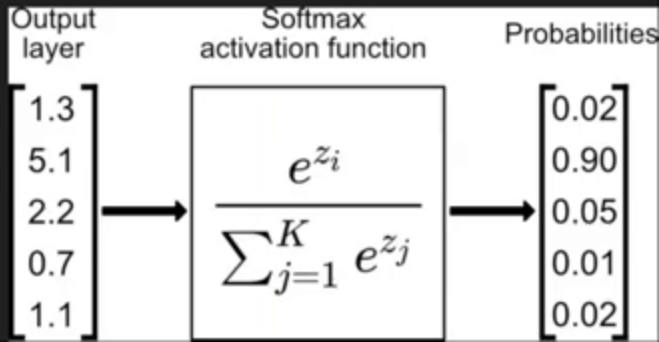
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

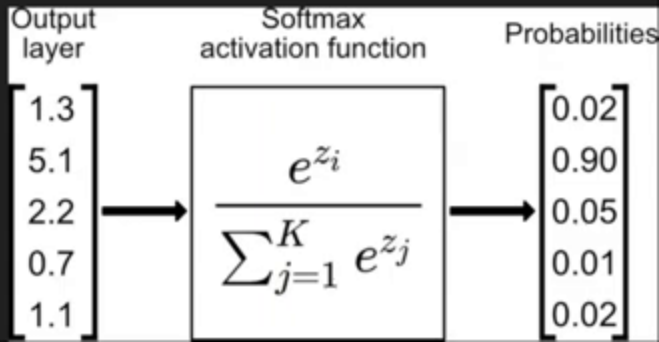
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

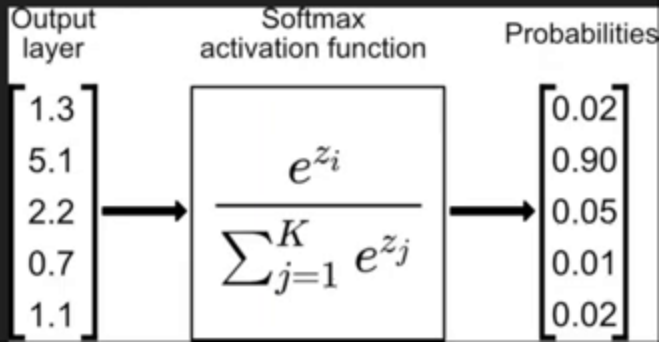
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

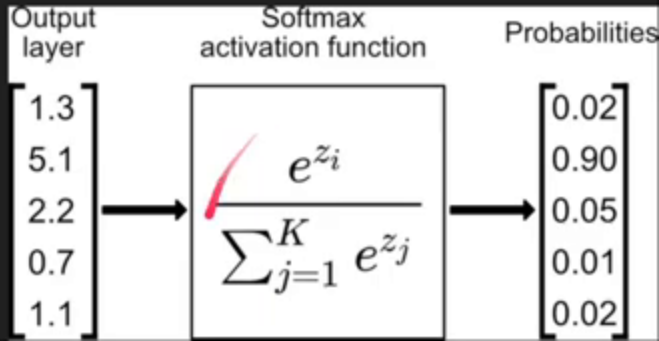
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

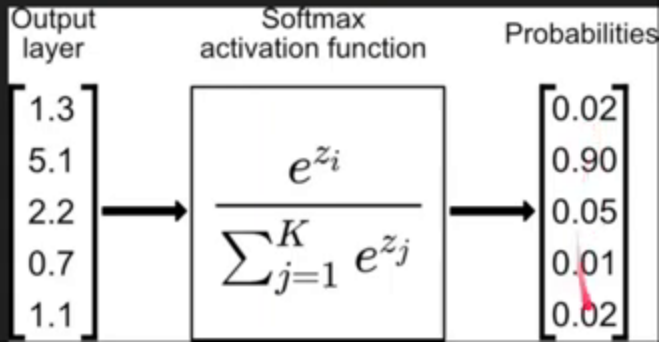
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

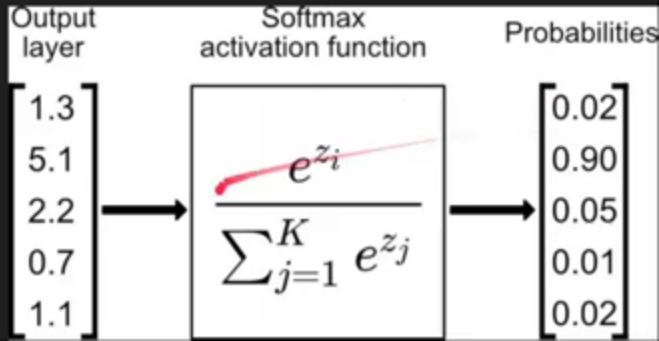
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

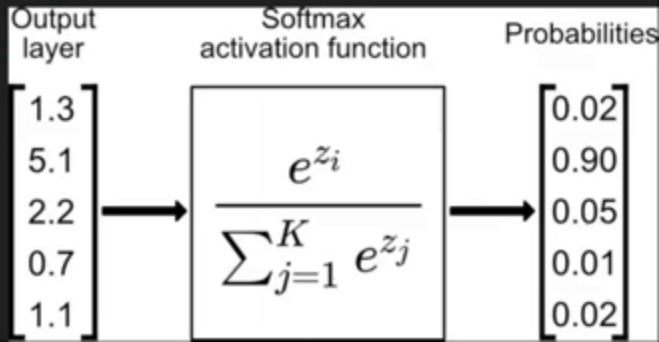
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

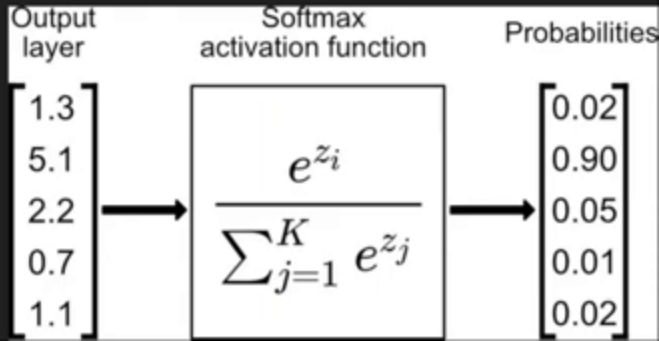
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

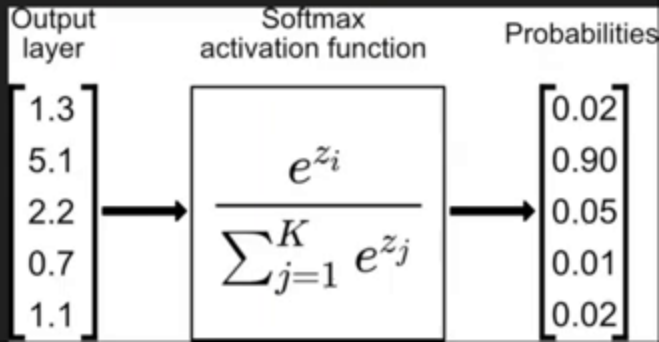
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

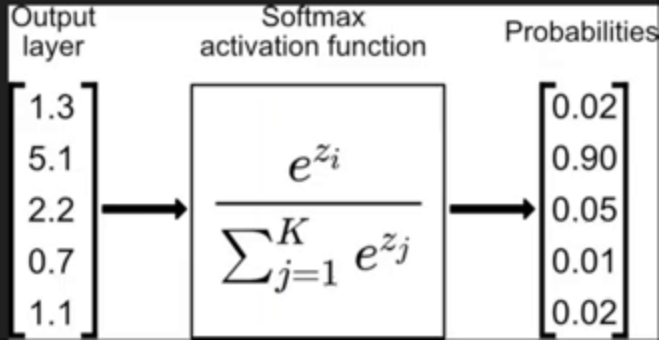
- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers



Softmax Activation Function

- Motivation: Multiclass classification, probabilistic output
- Definition:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}$$

- Derivative:

$$\frac{\partial S(z_i)}{\partial z_j} = \begin{cases} S(z_i) \times (1 - S(z_i)) & \text{if } i = j \\ -S(z_i) \times S(z_j) & \text{if } i \neq j \end{cases}$$

- Properties:
  - Non-Linear
  - Bounded
  - Not 0-centered
- Performance: NA
- Use Case: Output layers