Activation Functions

Functions for introducing non-linearity in models

Input $\rightarrow$ Activation $\rightarrow$ Output

Why many activation functions?

- Non-linearity: Different ways of introducing non-linearity for learning complex patterns
- Gradient flow: Different backpropagation dynamics

@AIinMinutes

Linear Activation Function

The identity function

$$f(x) = x$$

Terms:

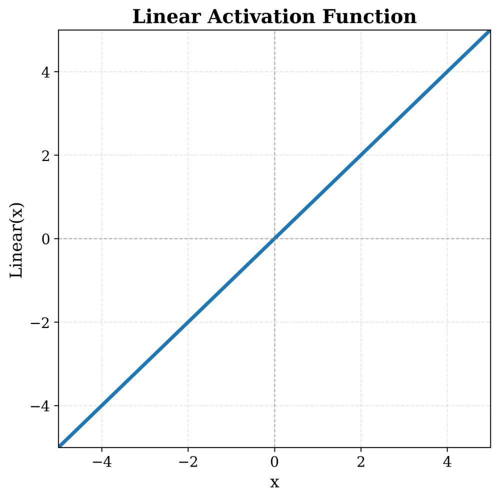
- Linear: Output proportional to input
- Identity: Returns same value as input

Key Properties:

- No non-linearity $\rightarrow$ Cannot learn complex patterns
- Network behaves as single linear transformation
- Used in output layer for regression problems

@AIinMinutes

Linear Activation Function



@AIinMinutes

Heaviside Step Function

Binary activation for early perceptrons

$$H(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$$

Terms:

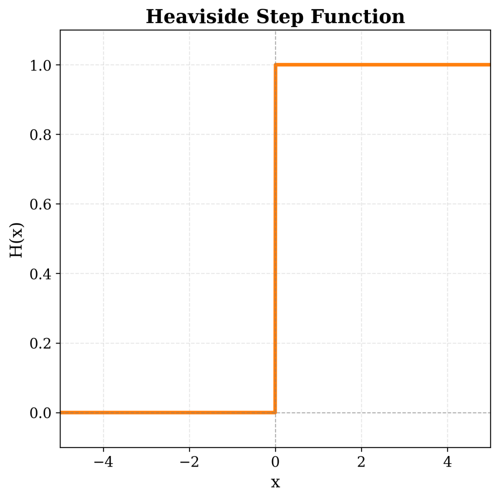
- Step Function: Jumps from 0 to 1 at $x = 0$
- Binary Activation: Makes yes/no decisions

Key Properties:

- Used in McCulloch-Pitts neurons and perceptrons
- Not differentiable at $x = 0$
- Incompatible with gradient-based learning

@AIinMinutes

Heaviside Step Function



@AIinMinutes

Sigmoid Activation Function

Smooth S-shaped squashing function

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

Terms:

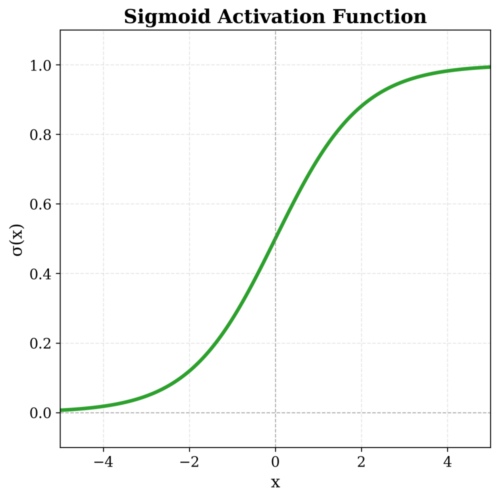
- Sigmoid Curve: S-shaped, maps to (0,1)
- Derivative: $\sigma'(x) = \sigma(x)(1 - \sigma(x))$

Key Properties:

- Used in logistic regression and early networks
- Probability interpretation for classification
- Suffers from vanishing gradient problem

@AIinMinutes

Sigmoid Activation Function



@AIinMinutes

Tanh Activation Function

Zero-centered sigmoid variant

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

Terms:

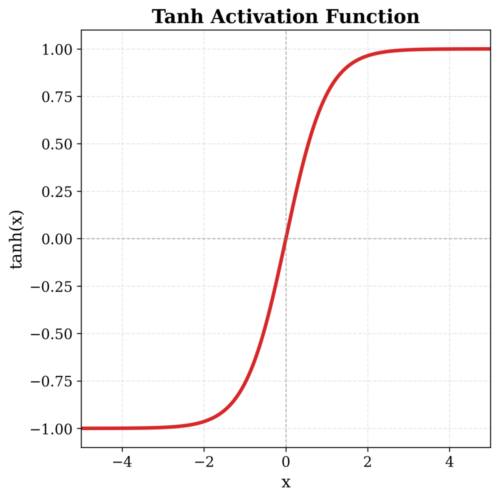
- Hyperbolic Tangent: Maps to (-1,1)
- Derivative: $\tanh'(x) = 1 - \tanh^2(x)$

Key Properties:

- Zero-centered outputs improve convergence
- Used in LSTMs for candidate memory updates
- Still suffers from gradient saturation

@AIinMinutes

Tanh Activation Function



@AIinMinutes

ReLU (Rectified Linear Unit)

Simple, efficient non-linearity

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

Terms:

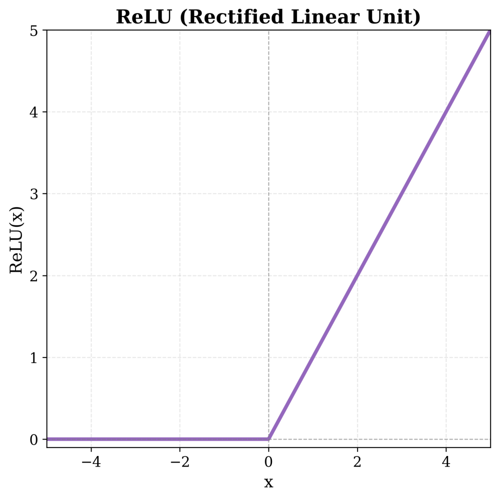
- Rectified: Zeroes out negative values
- Derivative: $f'(x) = \begin{cases} 1, & x > 0 \\ 0, & x < 0 \\ \text{undefined}, & x = 0 \end{cases}$

Key Properties:

- Computationally efficient
- Helps mitigate vanishing gradients
- Suffers from "dying ReLU" problem

@AIinMinutes

ReLU Activation Function



@AIinMinutes

Leaky ReLU

ReLU with small negative slope

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

Terms:

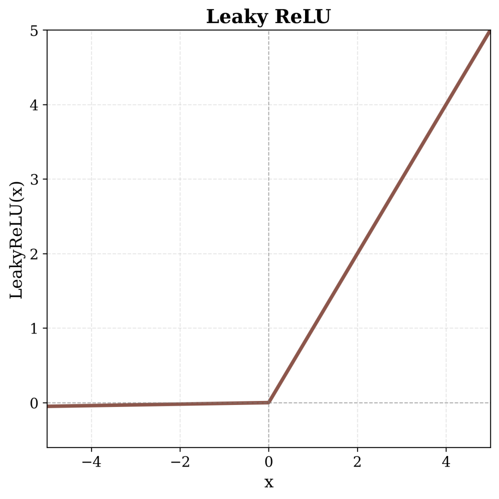
- Leak Factor α: Small constant (e.g., 0.01)
- Modified ReLU: Allows small negative outputs

Key Properties:

- Prevents "dying ReLU" problem
- Maintains gradient flow for negative inputs

@AIinMinutes

Leaky ReLU Activation Function



@AIinMinutes

Parametric ReLU (PReLU)

ReLU with learnable negative slope(s)

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

Terms:

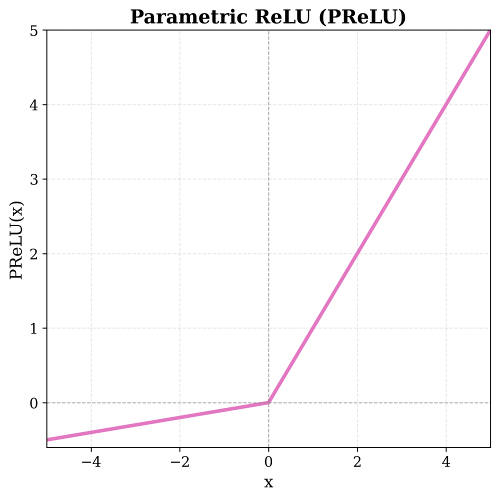
- α (Learnable Slope(s)): Trained parameter(s)
- Neuron/Layer/Global α: Different granularities

Key Properties:

- Adapts negative slopes to data patterns
- More flexible than standard ReLU
- Prevents "dying ReLU" problem

@AIinMinutes

Parametric ReLU Activation Function



@AIinMinutes

Exponential Linear Unit (ELU)

Smooth activation with saturation for negatives

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

Terms:

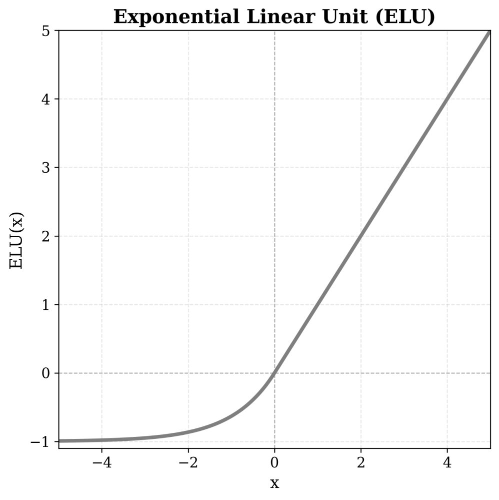
- α: Controls curve for negative inputs
- Exponential Component: Gradual negative values

Key Properties:

- Smooth and differentiable everywhere
- Reduces bias shift during training
- Negative values saturate to -α

@AIinMinutes

ELU Activation Function



@AIinMinutes

Swish-β

Non-monotonic with smooth gradients

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

Terms:

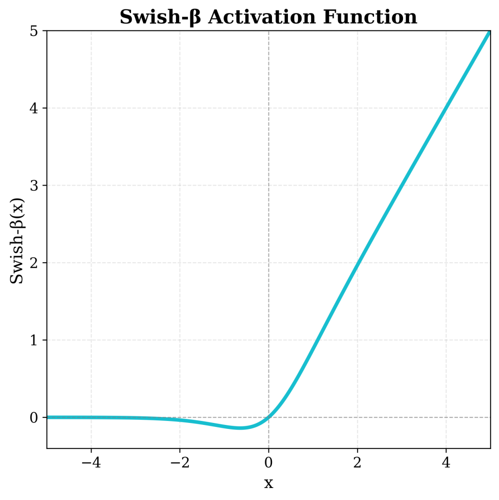
- $\sigma(x)$: Sigmoid function
- β: Learnable parameter controlling non-linearity

Key Properties:

- When $\beta = 1$, becomes standard Swish/SiLU
- Smooth and differentiable everywhere
- Handles negative values (non-monotonically) for better gradient flow (controlled by β)

@AIinMinutes

Swish- Activation Function



@AIinMinutes

SiLU (Sigmoid Linear Unit)

Also known as Swish-1 activation

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

Terms:

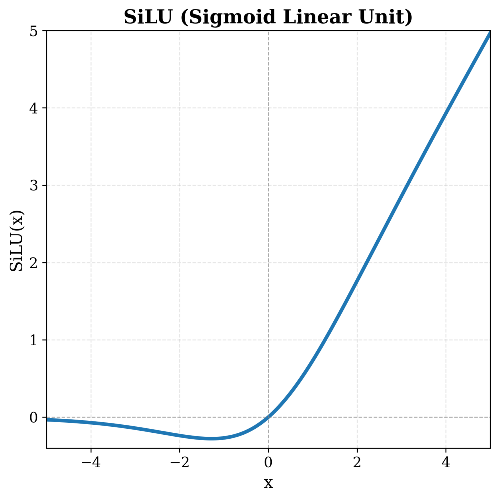
- $\sigma(x)$: Standard sigmoid function
- Swish-1: Special case of Swish where $\; = 1$

Key Properties:

- Smooth, non-monotonic activation function
- Unbounded above, bounded below
- Used in modern neural networks

@AIinMinutes

SiLU Activation Function



@AIinMinutes

GeLU (Gaussian Error Linear Unit)

Uses normal CDF for modulating inputs

$$f(x) = x \cdot \Phi(x) \approx x \cdot \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right)$$

Terms:

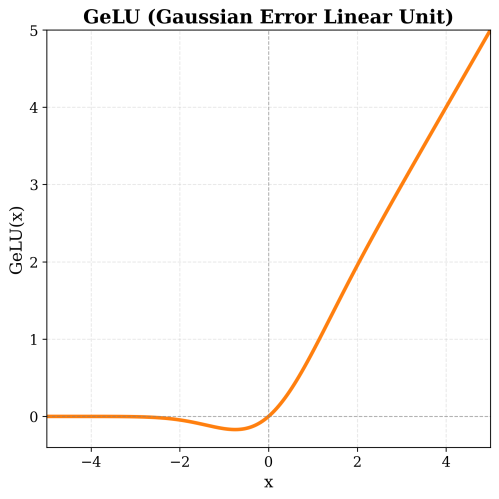
- $\Phi(x)$: CDF of standard normal distribution
- erf: Error function related to normal distribution

Key Properties:

- Smooth and differentiable
- Probabilistic interpretation of activation
- Used in transformer architectures (e.g., GPT)

@AIinMinutes

GeLU Activation Function



@AIinMinutes

GLU (Gated Linear Unit)

Input controls its own flow

$$\mathrm{GLU}(x) = \sigma(xW + b) \odot (xV + c)$$

Terms:

- W, V: Learnable weight matrices
- b, c: Bias vectors
- $\sigma(x)$: Sigmoid activation function

Key Properties:

- Selective information flow via gating
- Smooth gradient flow with no dead zones
- Widely used in language models

@AIinMinutes

SwiGLU

Advanced gating using Swish

$$\mathrm{SwiGLU}(X) = \mathrm{Swish}_\beta(XW_1) \odot (XW_2)$$

Terms:

- $\mathrm{Swish}_\beta(x) = x \cdot \sigma(\beta x)$
- W_1, W_2: Learnable weight matrices
- $\odot$: Element-wise multiplication

Key Properties:

- Smoother gating than standard GLU
- Better information flow and gradient propagation
- Used in modern transformers (LLaMA, PaLM)

@AIinMinutes