Activation Functions in Neural Networks

Introducing Non-Linearity and Enabling Complex Learning

Sina Moghimi, Autumn 2024

МФТИ

Outline

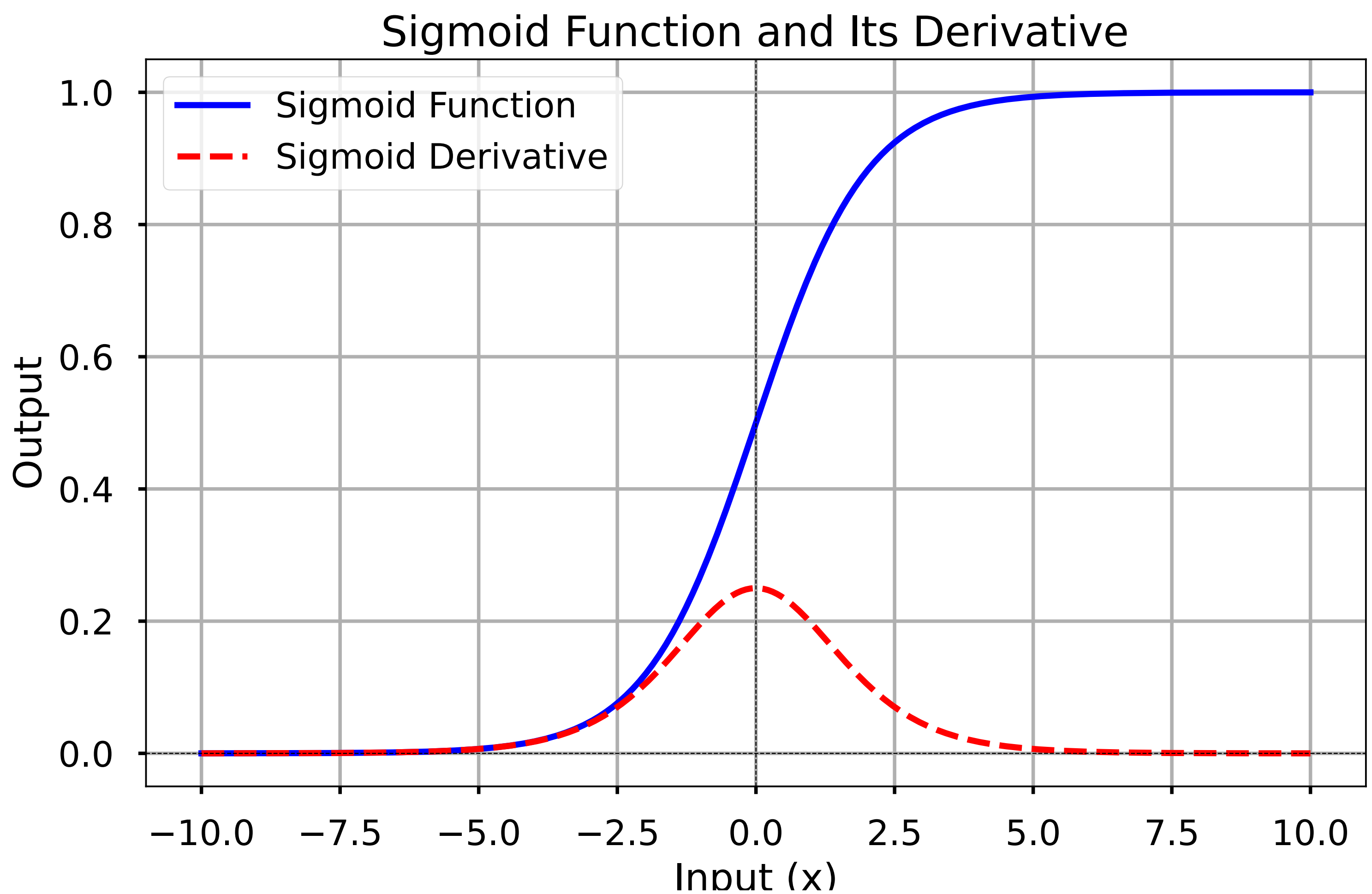
- Purpose of Activation Functions
- Sigmoid Activation Function
- Hyperbolic Tangent (Tanh)
- Rectified Linear Unit (ReLU)
- Mathematical Properties Comparison
- When to Use Sigmoid
- When to Use Tanh
- When to Use ReLU

Purpose of Activation Functions

- Introduce non-linearity
- Enable complex pattern learning
- Determine neuron output
- Importance in neural network architecture

Sigmoid Activation Function

Definition	Gradient
$\sigma(x) = \dfrac{1}{1+e^{-x}}$	$\sigma^{'}(x) = \sigma(x) \cdot (1 - \sigma(x))$



When to Use Sigmoid

- Sigmoid is commonly used in the **output layer** of a neural network for binary classification tasks.Enable complex pattern learning
  - It pairs well with a binary cross-entropy loss function for probabilistic outputs.

$$P(y = 1 \mid x) = \sigma(z)$$

Where z is the input to the output layer.

- Probabilistic Interpretation
  - When you need the model's output to be interpreted as a **probability** (e.g., logistic regression).
- Sigmoid can be used to normalize outputs into a fixed range [0, 1]
- In shallow networks(with few layers) where the vanishing gradient problem is not as significant.

МФТИ

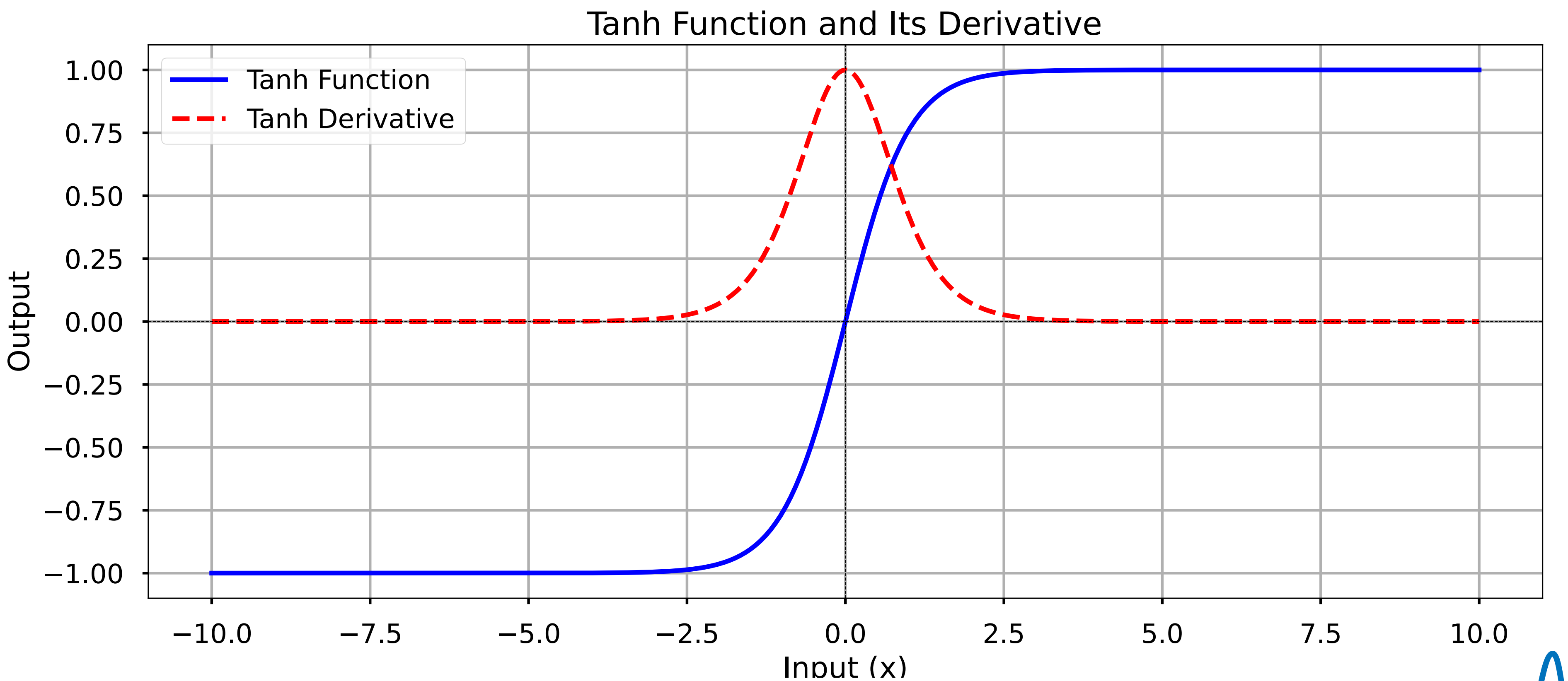
When to Avoid Sigmoid

- Deep Neural Networks
  - Vanishing gradient problem
  - Gradients become very small for large or small input values, slowing down or preventing learning in deeper layers.
- Zero-Centered Data
  - Sigmoid out put is in Range [0,1] so for outputs near zero, you would need to use ***Tanh***
- Wherever ReLU is working fine, using sigmoid may add unnecessary computational overhead.

МФТИ

Tanh Activation Function

Definition	Gradient
$tanh(x) = \dfrac{e^x - e^{-x}}{e^x + e^{-x}}$	$\dfrac{d}{dx} tanh(x) = 1 - tanh^2(x)$



МФТИ

When to Use Tanh

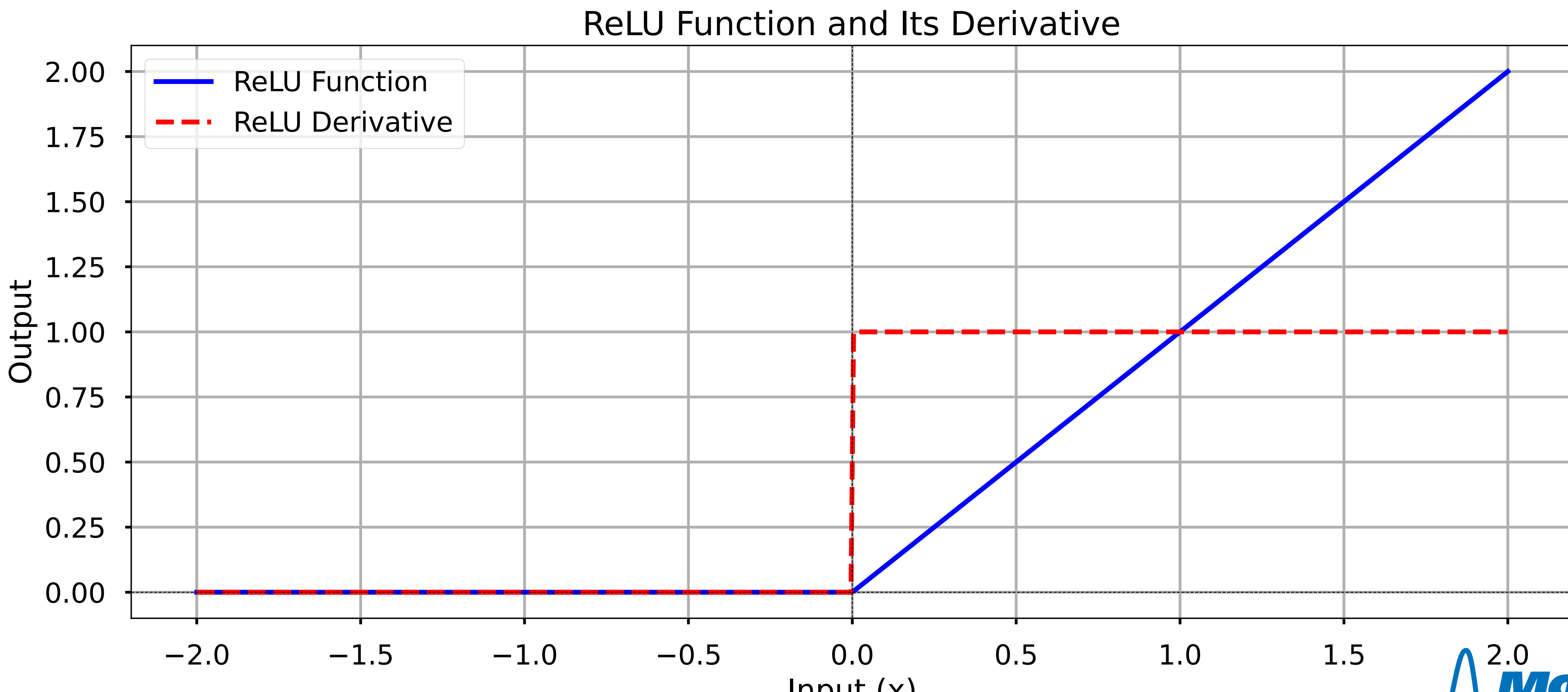
- Zero-Centered data as Tanh maps the input to range [-1, 1]
- Gradient Descent Optimization
    - Prevent biased weight updates and leading to faster convergence compared to sigmoid.
- Sequential Data and Recurrent Neural Networks (RNNs)
    - Tanh is commonly used in hidden layers of **recurrent neural networks (RNNs)** because of its ability to represent relationships between positive and negative input values.
- If you need symmetric output
- Shallow networks

When to Avoid Tanh

- Deep neural networks due to vanishing gradient problem.
- If Unbounded Output is Required
- Wherever ReLU is working fine, using Tanh may add unnecessary computational overhead.

ReLU Activation Function

Definition	Gradient
$f(x) = max(0,\ x)$	$f^{'}(x) = \begin{cases} 1, & if \quad z \geq 0 \\ 0, & if \quad z < 0 \end{cases}$



When to Use ReLU

- Hidden Layers in Deep Neural Networks
    - Avoids Vanishing Gradient Problem
    - Faster Convergence
- When You Need Computational Efficiency
- Sparse Activation
    - Activates only neurons with positive inputs
- Regression Tasks or Unbounded Output
- Common in Convolutional Neural Networks (CNNs)

When to Avoid ReLU

- Dead Neurons
  - For negative inputs, it outputs zero, and dont participate in the training.
- If Bounded Output is Required

Comparison

Property	Tanh	Sigmoid	ReLU
Range	[-1, 1]	[0, 1]	[0, ∞]
Zero-Centered	Yes	No	No
Vanishing Gradient	Yes	Yes	No (for positive inputs)
Use Case	Symmetric Data, Hidden Layers	Binary Outputs	Deep Networks, Unbounded tasks