Name: Shakib Sadat Shanto

ID: 20-43074-1

Sec: A

Sigmoid Function:

Sigmoid takes a real value as input and outputs another value between 0 and 1. It can be represented as follows,

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

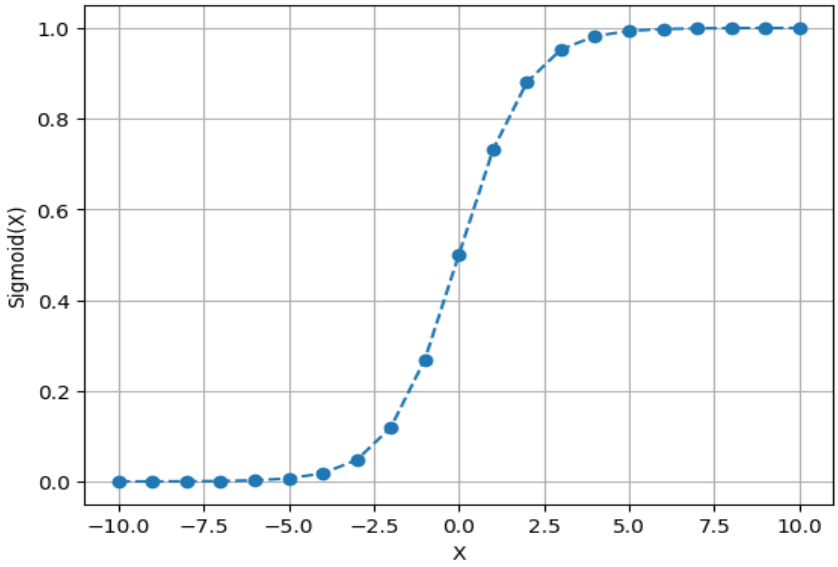

Fig: Sigmoid graph

It is nonlinear in nature. Combinations of this function are also nonlinear. It is especially used for models where we have to predict the probability as an output. It has a smooth gradient. The output of the activation function is always going to be in range (0,1) compared to (-inf, inf) of linear function. So, it's bound in a range.

Towards either end of the sigmoid function, the Y values tend to respond very less to changes in X. It gives rise to a problem of "vanishing gradients". Its output isn't zero centered. It makes the gradient updates go too far in different directions. 0 < output < 1, and it makes optimization harder. Sigmoid saturate and kill gradients.

Tanh Function:

Tanh squashes a real-valued number to the range [-1, 1]. It's non-linear. But unlike Sigmoid, its output is zero-centered. Therefore, in practice the tanh non-linearity is always preferred to the sigmoid nonlinearity. This activation function is simply the hyperbolic tangent (tanh) function. It can be represented as follows,

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

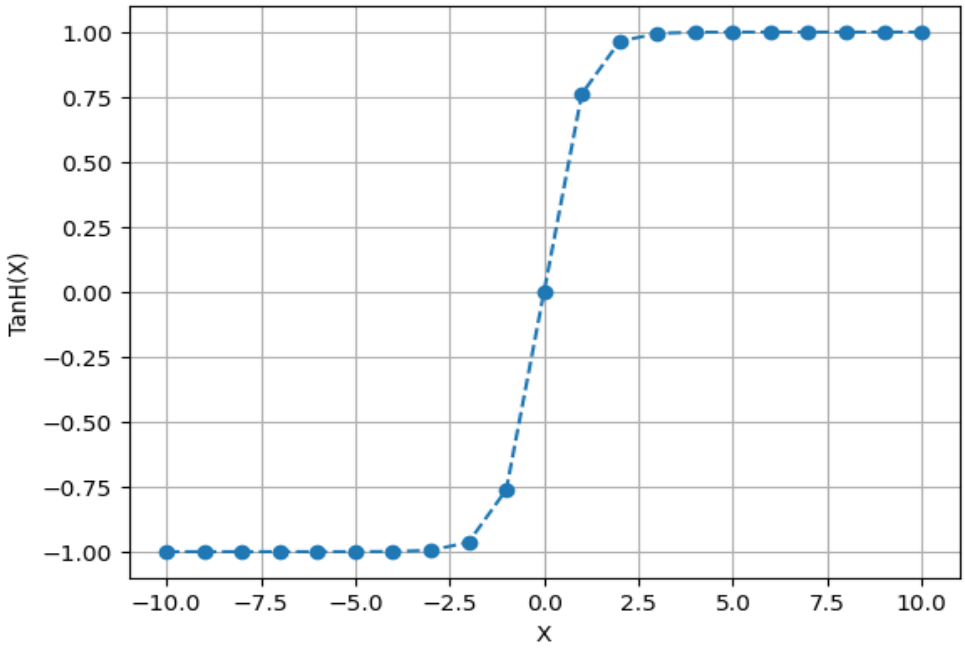

Fig: Tanh graph

The gradient is stronger for tanh than sigmoid. It is zero centered.

Tanh is highly compute-intensive and suffers from saturation problem and thus vanishing gradient. In fact, when the neuron reaches the minimum or maximum value of its range, that respectively correspond to -1 and 1, its derivative is equal to 0.

ReLU Function:

ReLU stands for Rectified Linear Units. Despite its name and appearance, it's not linear and provides the same benefits as Sigmoid but with better performance. The range of ReLu is [0,∞). It can be represented as follows,

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

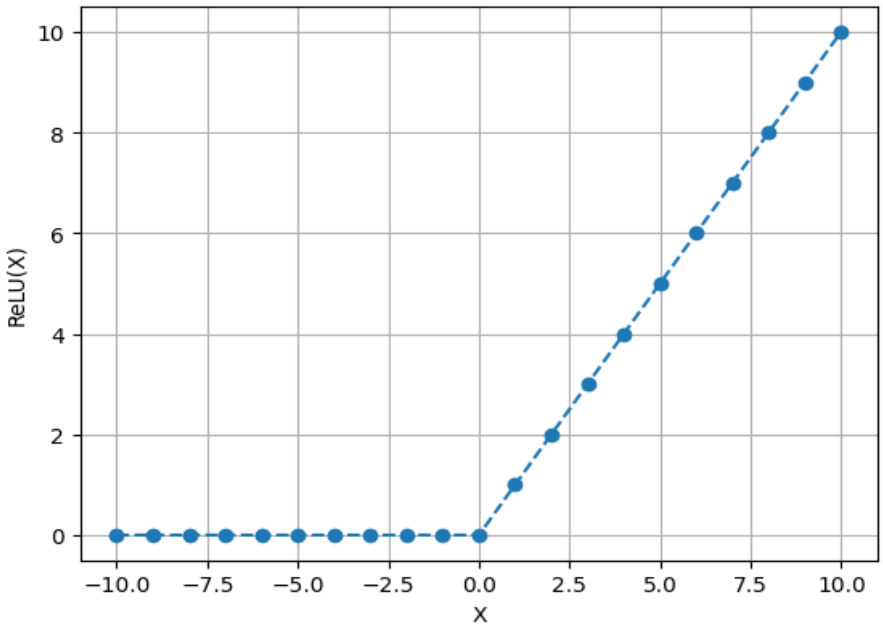

Fig: ReLU graph

It avoids and rectifies vanishing gradient problem. ReLu is less computationally expensive than tanh and sigmoid because it involves simpler mathematical operations. It converges much faster than sigmoid and tanh functions.

One of its limitations is that it should only be used within hidden layers of a neural network model. Some gradients can be fragile during training and can die. It can cause a weight update which will makes it never activate on any data point again. For activations in the region (x<0) of ReLu, gradient will be 0 because of which the weights will not get adjusted during descent. That means, those neurons which go into that state will stop responding to variations in error because gradient is 0 thus nothing changes. This is called the dying ReLu problem.

Step Function:

Step function is a threshold-based activation function which means after a certain threshold neuron is activated and below the said threshold neuron is deactivated. It can be represented as follows,

$$f(x) = \begin{cases} 1 & \text{if } x > \theta \\ 0 & \text{if } x \leq \theta \end{cases}$$

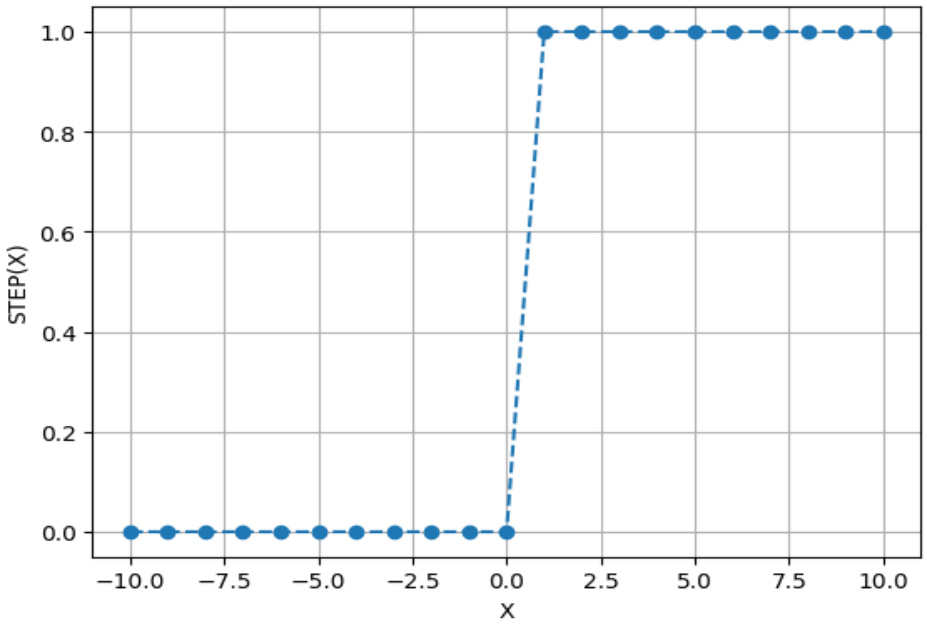

Fig: Step graph

For input values of (threshold value) or above, this equation produces a value of 1.0, whereas for all other values, it produces a value of 0. Threshold functions, sometimes referred to as step functions, only return 1 (true) for values greater than the defined threshold.

ELU Function:

Exponential Linear Unit or its widely known name ELU is a function that tend to converge cost to zero faster and produce more accurate results. Different to other activation functions, ELU has an extra alpha constant which should be positive number. ELU is very similar to RELU except negative inputs. They are both in identity function form for non-negative inputs. It can be represented as follows,

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

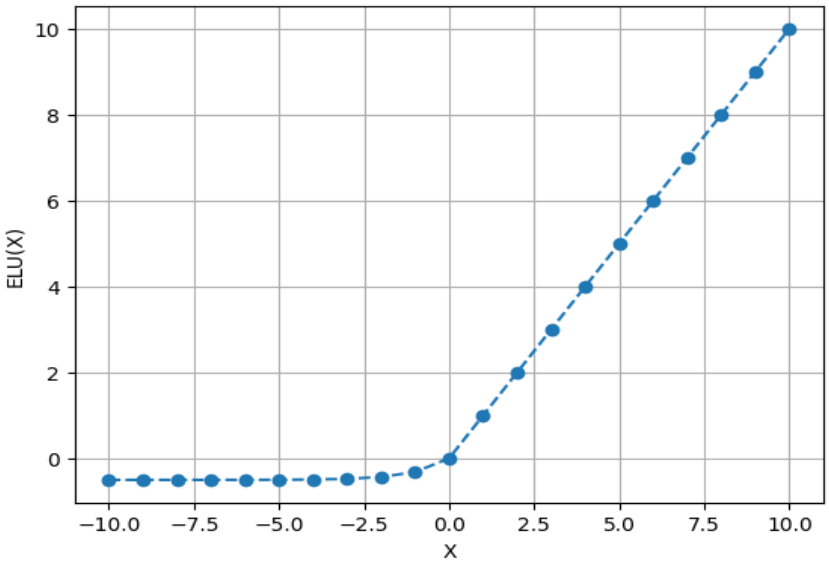

Fig: ELU graph

ELU becomes smooth slowly until its output equal to -α whereas RELU sharply smooths. ELU is a strong alternative to ReLU. Unlike to ReLU, ELU can produce negative outputs.

SELU Function:

SELU or Scaled Exponential Linear Units, is activation function that induce self-normalization. SELU network neuronal activations automatically converge to a zero mean and unit variance. It can be represented as follows,

$$f(x) = \lambda x \qquad \text{if } x > 0$$
$$f(x) = \lambda\alpha(e^x - 1) \qquad \text{if } x \leq 0$$

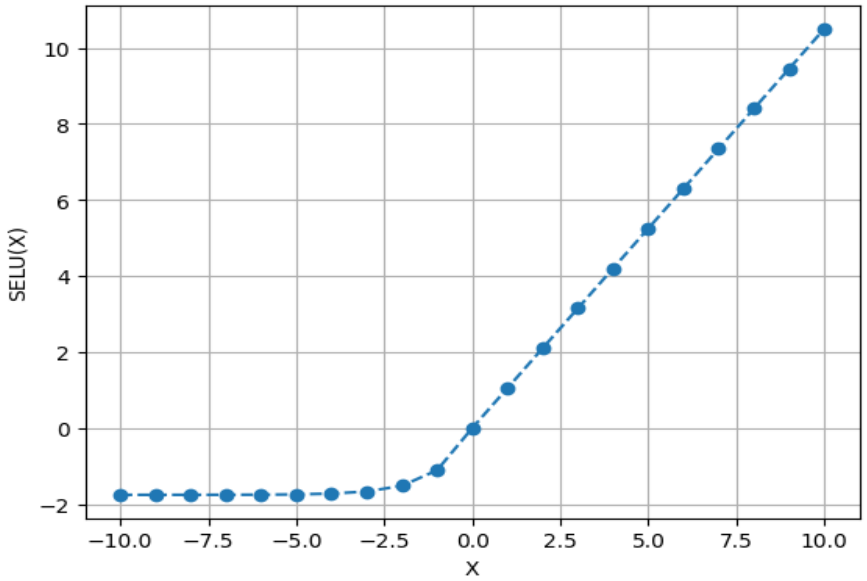

Fig: SELU graph

If x is larger than 0, the output result is x multiplied by lambda. If the input value x is less than or equal to zero, we have a function that goes up to 0, which is the output y, when x is zero. Essentially, when x is smaller than zero, it takes the exponential of the x-value minus 1, then we multiply it with alpha α and lambda λ. Like ReLU, SELU does not have vanishing gradient problem and hence, is used in deep neural networks. SELU learn faster and better than other activation functions without needing further procession. Moreover, other activation function combined with batch normalization cannot compete with SELU.

SELU is a relatively new activation function so it is not yet used widely in practice. ReLU stays as the preferred option.