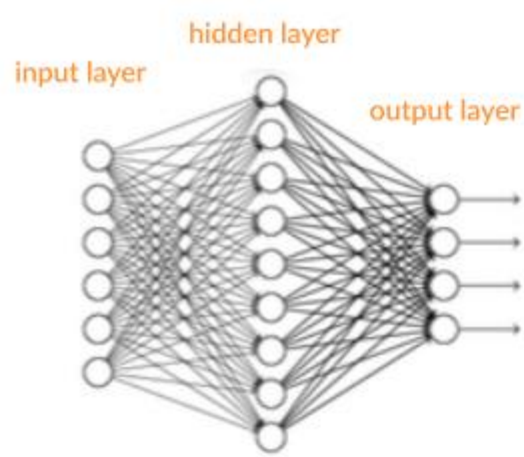
# TensorFlow Activation Functions

Neural network activation functions are a crucial component of deep learning. Activation functions determine the output of a deep learning model, its accuracy, and also the computational efficiency of training a model—which can make or break a large scale neural network. Activation functions also have a major effect on the neural network's ability to converge and the convergence speed, or in some cases, activation functions might prevent neural networks from converging in the first place.

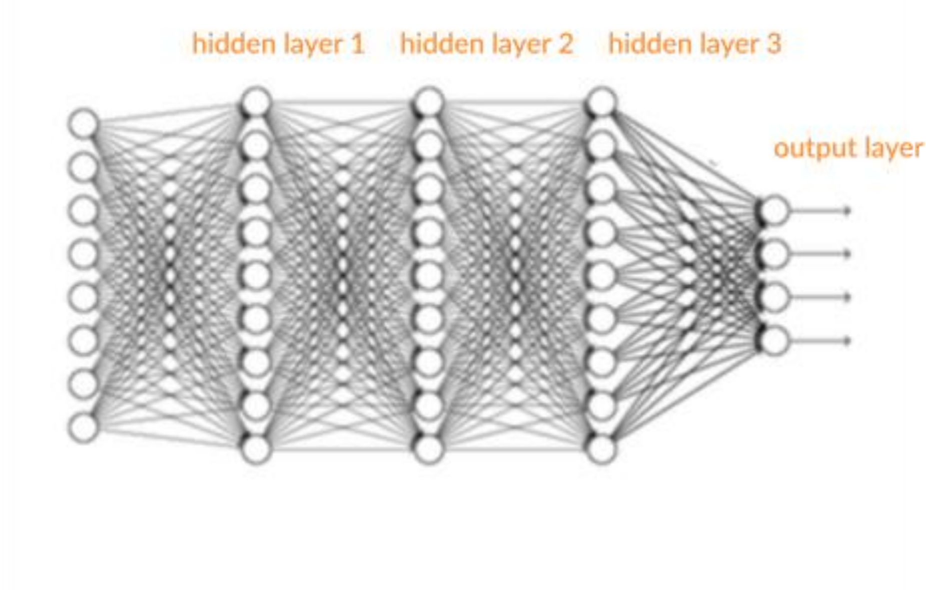
Activation functions are mathematical equations that determine the output of a neural network. The function is attached to each neuron in the network, and determines whether it should be activated ("fired") or not, based on whether each neuron's input is relevant for the model's prediction. Activation functions also help normalize the output of each neuron to a range between 1 and 0 or between -1 and 1.

Artificial Neural Networks (ANN) are comprised of a large number of simple elements, called neurons, each of which makes simple decisions. Together, the neurons can provide accurate answers to some complex problems, such as natural language processing, computer vision, and AI.

A neural network can be "shallow", meaning it has an input layer of neurons, only one "hidden layer" that processes the inputs, and an output layer that provides the final output of the model. A Deep Neural Network (DNN) commonly has between 2-8 additional layers of neurons.




Non-Deep feedforward neural network.

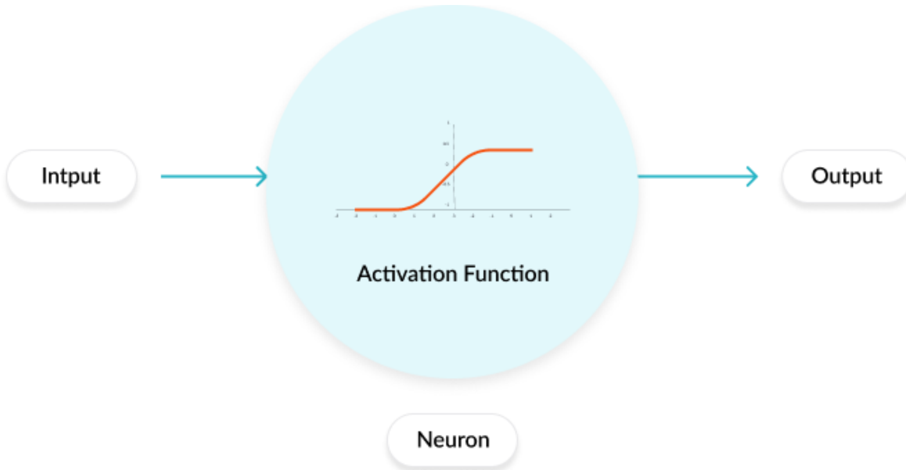



Deep Neural Network

## Role of the Activation Function in a Neural Network Model

In a neural network, numeric data points, called inputs, are fed into the neurons in the input layer. Each neuron has a weight, and multiplying the input number with the weight gives the output of the neuron, which is transferred to the next layer.

The activation function is a mathematical “gate” in between the input feeding the current neuron and its output going to the next layer. It can be as simple as a step function that turns the neuron output on and off, depending on a rule or threshold. Or it can be a transformation that maps the input signals into output signals that are needed for the neural network to function.




Increasingly, neural networks use non-linear activation functions, which can help the network learn complex data, compute and learn almost any function representing a question, and provide accurate predictions.

**The basic process carried out by a neuron in a neural network is:**

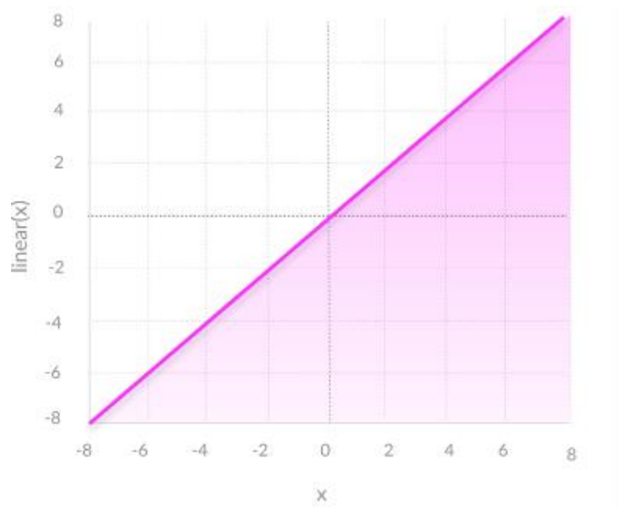

## Types of Activation Functions

1. Linear Function
2. Non Linear Function

## Linear Activation Function

A linear activation function takes the form:

$$\mathbf{A = cx}$$




It takes the inputs, multiplied by the weights for each neuron, and creates an output signal proportional to the input.

**1. Not possible to use backpropagation** (gradient descent) to train the model—the derivative of the function is a constant, and has no relation to the input, X. So it's not possible to go back and understand which weights in the input neurons can provide a better prediction.

**2. All layers of the neural network collapse into one**—with linear activation functions, no matter how many layers in the neural network, the last layer will be a linear function of the first layer (because a linear combination of linear functions is still a linear function). So a linear activation function turns the neural network into just one layer.

A neural network with a linear activation function is simply a linear regression model. It has limited power and ability to handle complexity varying parameters of input data.

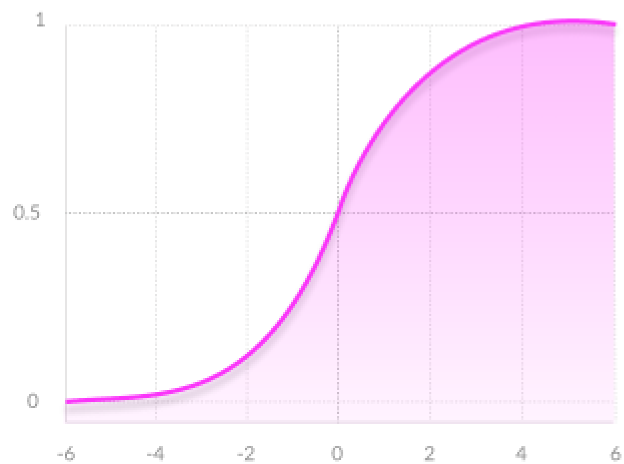
## Non-Linear Activation Functions

1. **The sigmoid function** has a smooth gradient and outputs values between zero and one. For very high or low values of the input parameters, the network can be very slow to reach a prediction, called the *vanishing gradient* problem.
2. **The TanH function** is zero-centered making it easier to model inputs that are strongly negative strongly positive or neutral.
3. **The ReLu function** is highly computationally efficient but is not able to process inputs that approach zero or negative.
4. **The Leaky ReLu** function has a small positive slope in its negative area, enabling it to process zero or negative values.
5. **The Parametric ReLu** function allows the negative slope to be learned, performing backpropagation to learn the most effective slope for zero and negative input values.

6. **Softmax** is a special activation function use for output neurons. It normalizes outputs for each class between 0 and 1, and returns the probability that the input belongs to a specific class.
7. **Swish** is a new activation function discovered by Google researchers. It performs better than ReLu with a similar level of computational efficiency.

Modern neural network models use non-linear activation functions. They allow the model to create complex mappings between the network's inputs and outputs, which are essential for learning and modeling complex data, such as images, video, audio, and data sets which are non-linear or have high dimensionality.

## 1. Sigmoid / Logistic




### Advantages

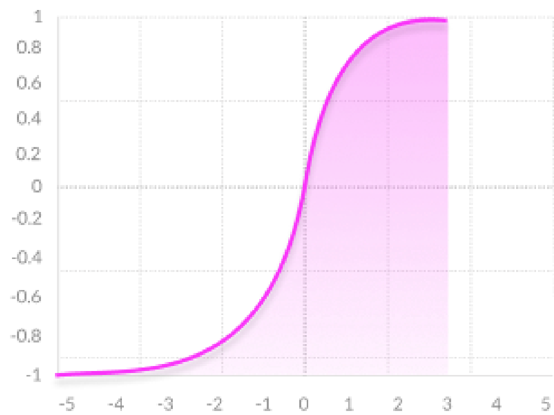
- **Smooth gradient**, preventing "jumps" in output values.
- **Output values bound** between 0 and 1, normalizing the output of each neuron.
- **Clear predictions**—For X above 2 or below -2, tends to bring the Y value (the prediction) to the edge of the curve, very close to 1 or 0. This enables clear predictions.

### Disadvantages

- **Vanishing gradient**—for very high or very low values of X, there is almost no change to the prediction, causing a vanishing gradient problem. This can result in the network refusing to learn further, or being too slow to reach an accurate prediction.

- **Outputs not zero centered**.
- **Computationally expensive**

## TanH / Hyperbolic Tangent



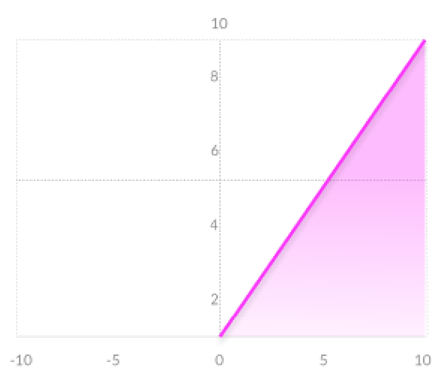

### Advantages

- **Zero centered**—making it easier to model inputs that have strongly negative, neutral, and strongly positive values.
- Otherwise like the Sigmoid function.

### Disadvantages

- Like the Sigmoid function

## ReLU (Rectified Linear Unit)



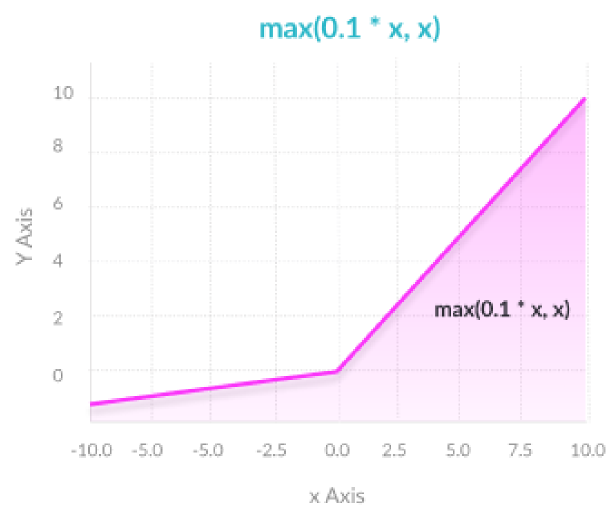
### Advantages

- **Computationally efficient**—allows the network to converge very quickly
- **Non-linear—**although it looks like a linear function, ReLU has a derivative function and allows for backpropagation

### Disadvantages

- **The Dying ReLU problem**—when inputs approach zero, or are negative, the gradient of the function becomes zero, the network cannot perform backpropagation and cannot learn.

## Leaky ReLU




### Advantages

- **Prevents dying ReLU problem**—this variation of ReLU has a small positive slope in the negative area, so it does enable backpropagation, even for negative input values
- Otherwise like ReLU

### Disadvantages

- **Results not consistent**—leaky ReLU does not provide consistent predictions for negative input values.

## Softmax

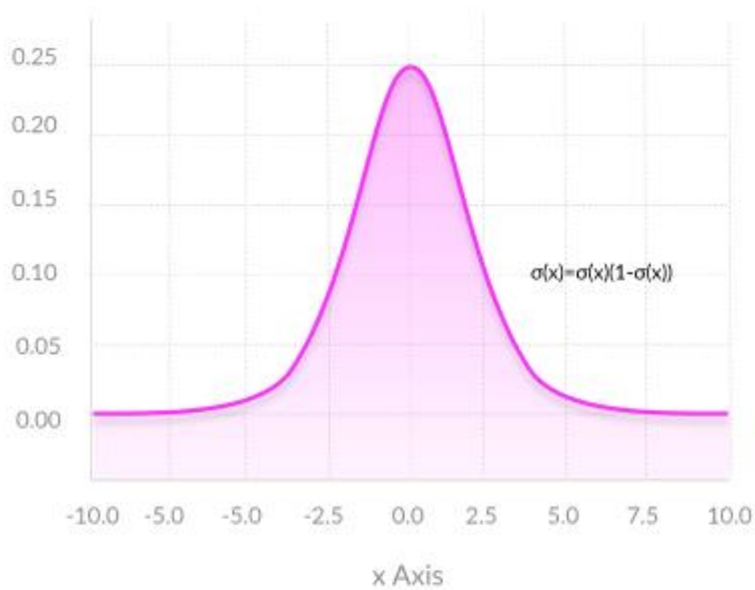
## Advantages

- **Able to handle multiple classes** only one class in other activation functions—normalizes the outputs for each class between 0 and 1, and divides by their sum, giving the probability of the input value being in a specific class.
- **Useful for output neurons**—typically Softmax is used only for the output layer, for neural networks that need to classify inputs into multiple categories.

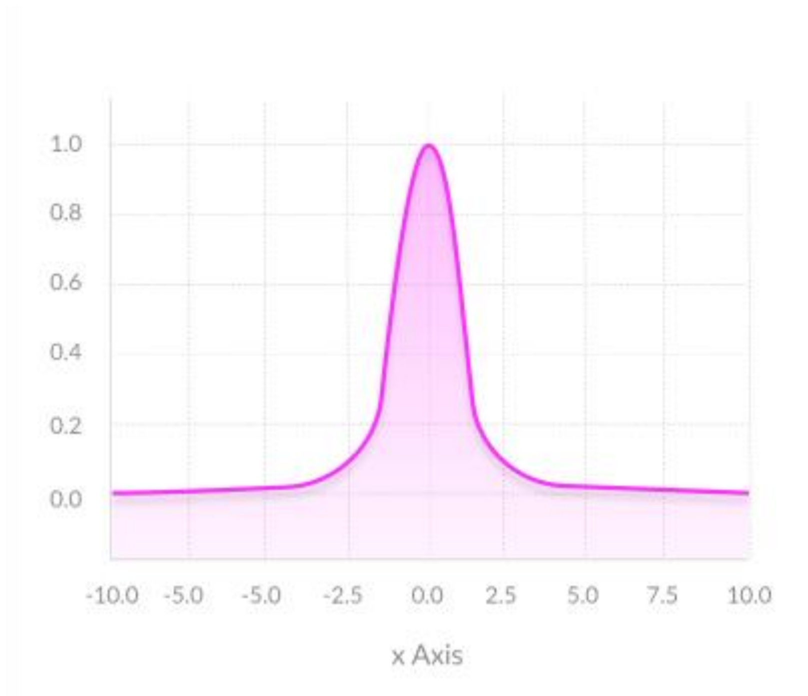
## Derivatives or Gradients of Activation Functions

The derivative—also known as a gradient—of an activation function is extremely important for training the neural network.

Neural networks are trained using a process called backpropagation—this is an algorithm which traces back from the output of the model, through the different neurons which were involved in generating that output, back to the original weight applied to each neuron. Backpropagation suggests an optimal weight for each neuron which results in the most accurate prediction.




Sigmoid




TanH