Deep Learning – Artificial Neural Networks

Introduction

Deep Learning (DL), which is a branch of Machine Learning (ML) and Artificial Intelligence (AI). Using Artificial Neural Networks (ANN) for data processing, DL needs a massive data to train a model. ANN is a network that simulates neurons in human brain. As shown in Figure 1, an ANN which has 2 or more layers in the hidden layer can be called as a Deep Neural Network (DNN).

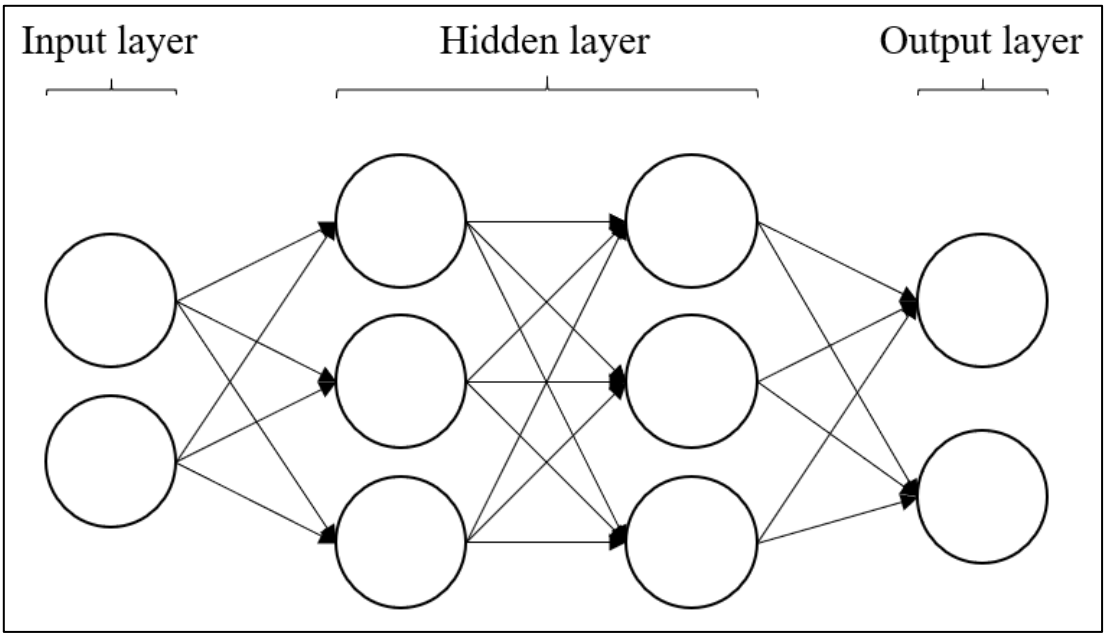

Figure. 1 Deep Neural Network, DNN

Perceptron

Perceptron is the basic component in a Neural Network (NN). As Shown in Figure 2, a perceptron also has an input layer and an output layer. There has an input set of tensor: $[x_1, x_2, \dots, x_n]$ for the input layer and every input has a corresponded weight w_i in the weights set: $[w_1, w_2, \dots, w_n]$, additionally adds a bias b so that we can easier find a solution.

For the output layer we first multiply every input x_i and weight w_i then sum up with the bias b, which comes out an equation: $z = (\sum_{i=1}^{n} x_i w_i) + b$. After we get z from the above equation we send it to the activation function $f()$ and returns the output: $output = f(z) = f((\sum_{i=1}^{n} x_i w_i) + b)$.

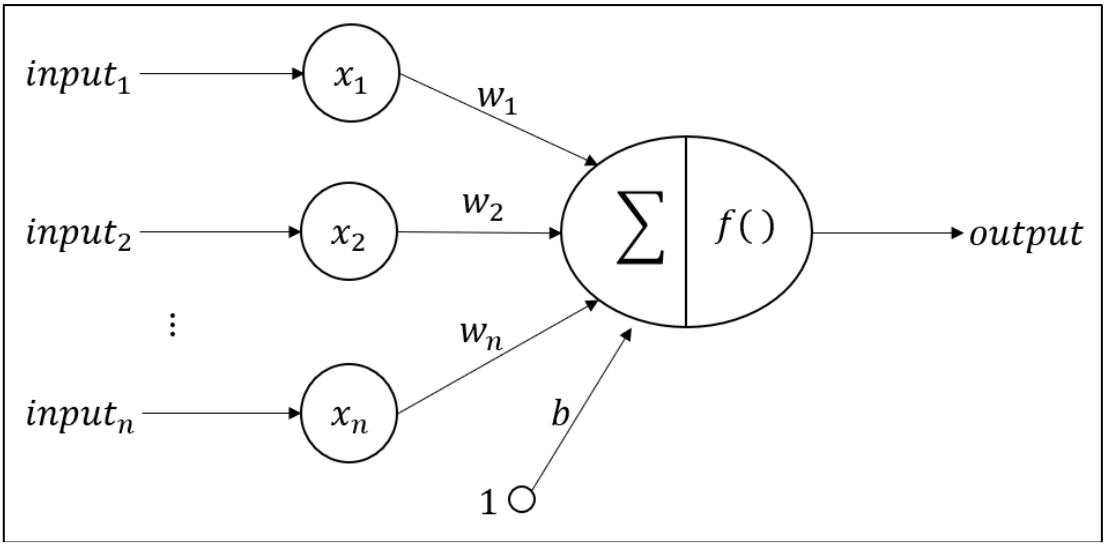

Figure 2. Perceptron

Activation Function

If there is no activation function in a NN no matter how many layers we pass through, it always constructs a linear function which can only deals with linear problems when fitting data. To solve a nonlinear problem, we use activation function to break the linearity which transfer the data into any range like 0 to 1 or -1 to 1 and that makes the NN fits more nonlinear problems. Here list some activation functions:

1. Sigmoid

As shown in Figure 3, a sigmoid function converts any data into a range of 0 to 1, the equation:

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

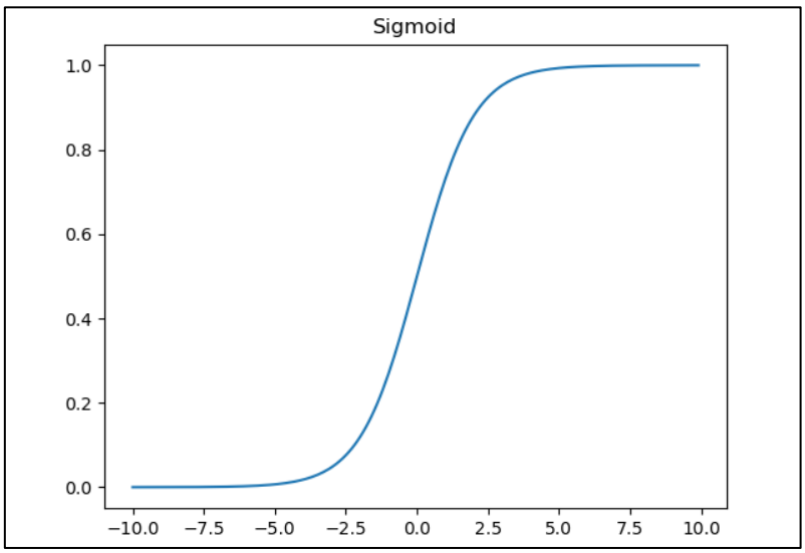

Figure 3. Sigmoid function

2. Rectified Linear Unit (ReLU)

As shown in Figure 4, the maximum value is only 0.25 when derivates the sigmoid function, this will encounter a problem called Vanishing Gradient Problem which happens when we do backpropagation using chain rule to calculate the gradient.

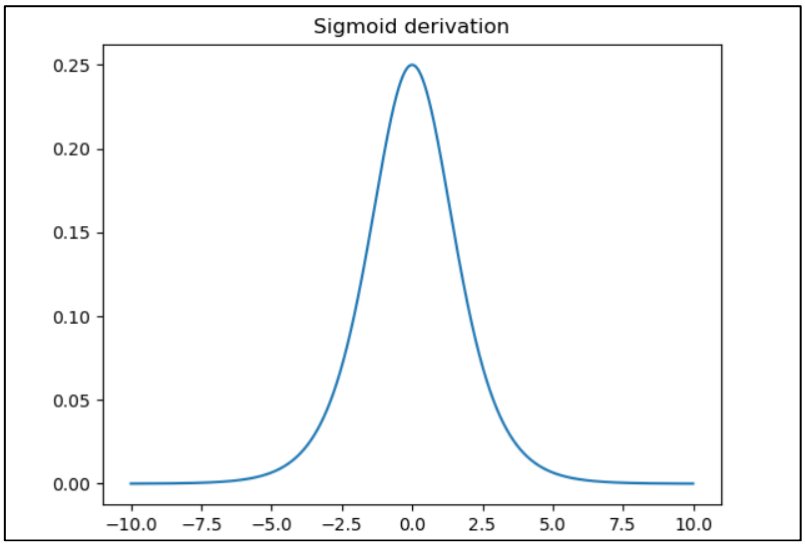

Figure 4. Derivative of Sigmoid function

ReLU function outputs 0 when the input is smaller than 0 else outputs a linear function which is the same as input. We can see that in Figure 5, the ReLU function has a derivation of 1 when it is greater than 0, which avoids the Vanishing Gradient Problem.

The equation:

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

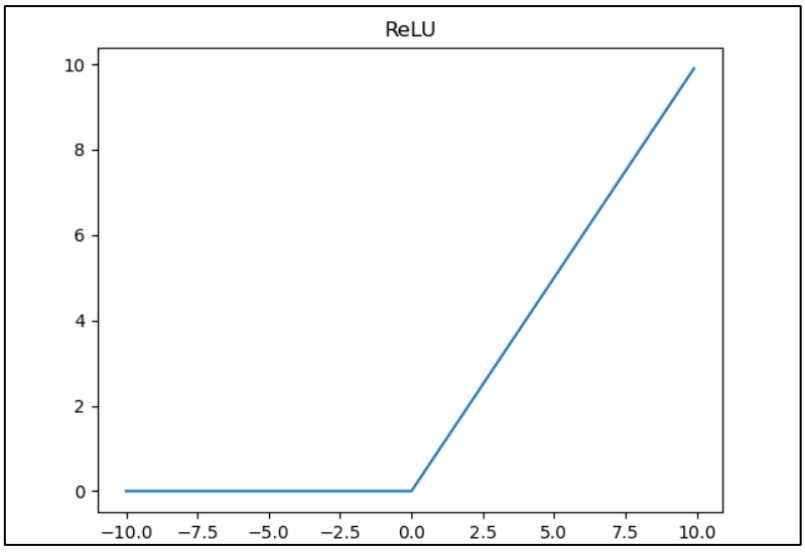

Figure 4. ReLU function

3. Hyperbolic Tangent (Tanh)

Tanh is a trigonometric function, Figure 5 shows that the function has an output in a range of -1 to 1 while Sigmoid and ReLU do not have negative values, the equation:

$$\tanh x = \frac{\sinh x}{\cosh x} = \frac{e^x - e^{-x}}{e^x + e^{-x}} = \frac{e^{2x} - 1}{e^{2x} + 1}$$

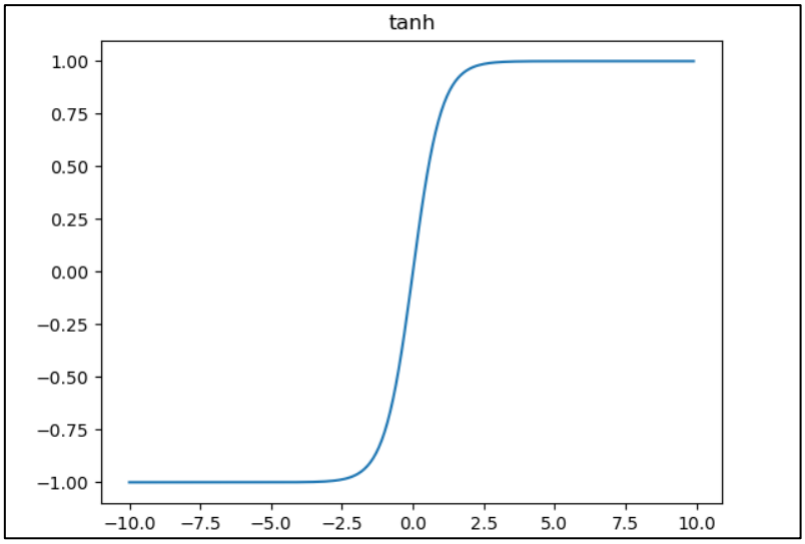

Figure 5. Tanh function

4. Softmax

Softmax function converts the inputs into real numbers between 0 and 1 which presents in a form of probability, the formula is:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ for } i = 1, 2, \ldots, K \text{ and } z = (z_1, z_2, \ldots, z_K) \in \mathbb{R}^K$$

Multilayer Perceptron (MLP)

With single perceptron we can solve some linearly separable problems, but when we are about to do some problems that is not linearly separable, we need 2 or more layers of perceptron. As described at the beginning of the article, Figure 1 is an MLP, its every node connects to every node in the next layer which is also called Full Connected and this kind of NN layer is called Dense Layer.

Convolutional Neural Network (CNN)

When it comes to computer vision people occurs to CNN, it is powerful makes the computer recognizing images. The core concept of CNN is the computation of convolution and pooling to the input, after that we do classification and outputs the result.

1. Architecture

Figure 6 shows us the basic CNN architecture. First input an image to the network and do the Feature Extraction using 2 or more sets of a convolutional layer and a pooling layer, then pass the feature to the Fully Connected Layer for the classification and finally outputs the result. The input layer, the output layer and the fully connected layer is the same as MLP, the main difference is on the convolutional layer and the pooling layer.

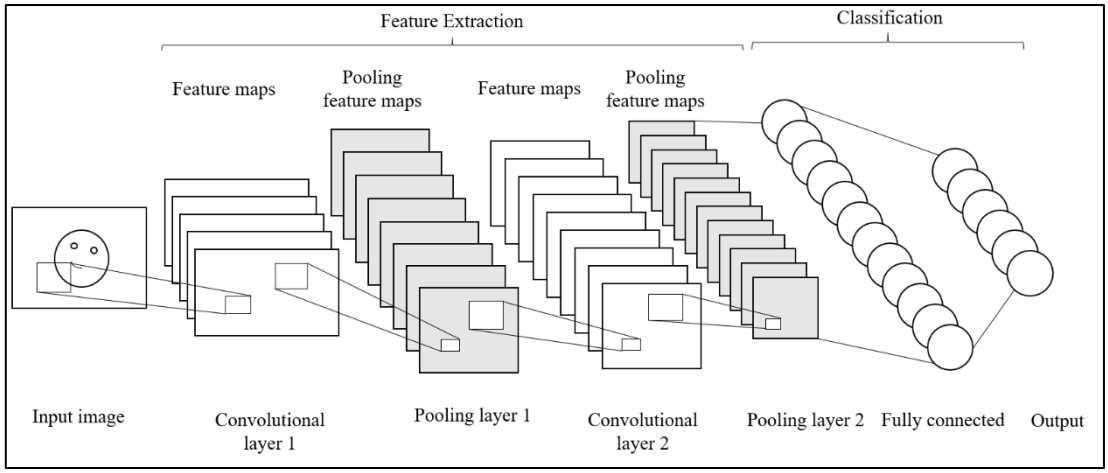

Figure 6. CNN architecture

2. Convolutional Layer

For a single image input there is a 3-dimensional tensor: (image width, image height, image depth), which image depth means the channels a image has, if a image uses RGB it has 3 channels in its image depth while grayscales has only 1 channel.

Every convolutional layer has 1 or more kernels to extract the feature from the image, and the kernel is also called the filter. The filter is the weight of a convolutional layer, as shown in Figure 7, a 5x5 image with a 3x3 filter gets a feature map size of 3x3.

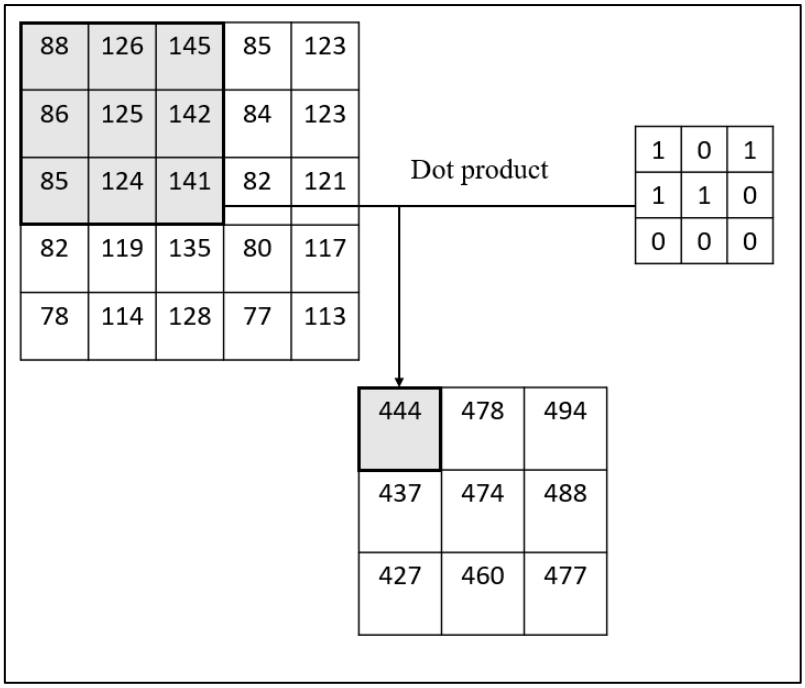

Figure 7. Convolution

The size of a feature map is determined by the stride and the Zero-Padding. Stride is the step moving filter in pixels, convolution can be faster when stride is bigger. Zero-Padding maintains the size by adding 0 around the image. Suppose that input image size is W, the filter size is F, the stride is S and the numbers of padding zero is P:

$$Feature\ size = \frac{W - F + 2 * P}{S}$$

3. Pooling

Pooling is used to compress the image and reserve the important information and convert the feature map from convolutional layer to a new smaller feature map, which is called Down Sampling. There is Max pooling, Min pooling and Average pooling. As shown in Figure 8, Max pooling extracts the maximum value in a feature map while Min pooling extracts the minimum value and Average pooling extracts the average.

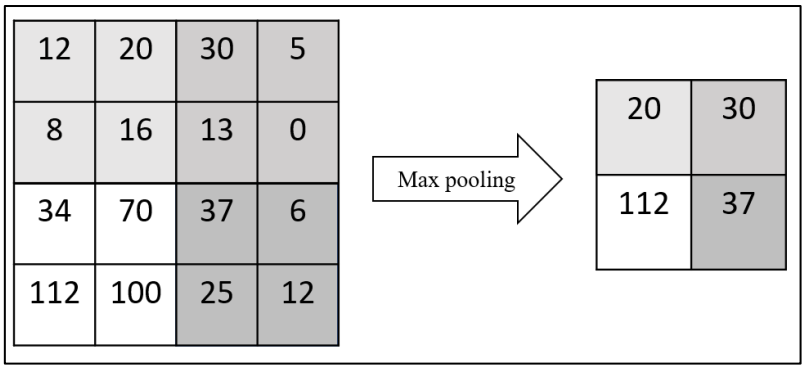

Figure 8. Max pooling

4. Dropout Layer

Dropout is an optimization in NN that can help against Overfitting without increasing the training data. Dropout layer adds some randomness into the Loss function and break the perceptron Co-adaptations between layers to fix the error from the previous layer.