Machine Learning

What is a Neural Network? | Deep Learning

The MNIST data is referenced in this case to explain the Machine Learning/Neural Networks. The Human brain recognizes these hand written numbers (or data) very easily but this is a difficult task for a computer. The MNIST data set of the hand written numbers is a classic example to know the implementation/working of a Neural Network in recognizing the 28x28 grayscale pixels of handwritten digits between 0-9. More advanced Neural Networks such as Convolutional Neural Networks and Long Short-Term Memory Networks are used for image recognition and speech recognition respectively. The basic Neural Network that's used in this case is known as a Multilayer Perceptron.

A neuron in the network is basically a unit that holds a value in between 0 and 1. So the network in this case starts with 784 (28x28) neurons that corresponds to the grayscale value (represented in between 0 and 1) of the hand written digit. This value gives us the information about a neuron's Activation. All these neurons represent the first layer or the input layer of the network. The last layer or the output layer has 10 neurons that represent the digits (0-9 respectively). This network has two hidden layers with each layer having 16 neurons (an arbitrary number). The functionality of this hidden layer can be dealt later. This network operates in a way that the activations of one layer help in determining the activations of the next layer. This mechanism of activations of one layer influencing the activations of the next is the main idea behind the processing of the data. If we feed the data (28x28 pixels) to the first layer, the pattern of activations in this layer causes a specific pattern of activations in the next layer which in turn causes a pattern in the next which finally causes a pattern in the output layer and the brightest neuron (highest activation value) of the output layer representing the corresponding digit, is the network's answer to what the given input image represents.

Machine Learning

If we consider how one layer influences another as a black box for now, then if we want to recognize digits we can first recognize patterns for example a 9 has a loop on top with a long line, 8 had two loops and 7 has one vertical line and horizontal line etc. But this is also very hard to recognize by the network directly. So, if we look for small edges that make up this pattern then it can be doable. Considering the 2nd hidden layer as the neurons that recognize (activates when they find) loops and 1st hidden layer as the neurons that detect edges from the input neurons that in turn help in recognizing patterns in the next hidden layer, we can finally get the output layer respond to patterns from the previous layer which then activates the corresponding neuron that represents a digit (between 0-9). This algorithm works in theory but can be tested to see if it works as intended. This layered approach works well in many image and speech recognition algorithms.

Now to determine the activations of a layer: we take the input layer activations and multiplying each activation unit with a certain parameter or weight and then summing all these weighted activations produces a value that might be out of the range (0 and 1). So, we pass this value to a function that translates or maps it to a smaller value in the range and 0 and 1. This can be done using many functions such as Sigmoid (-ve inputs correspond to 0 and +ve inputs corresponds to 1 and the values between them are increasing in range of 0 and 1), ReLU (Rectified Linear Unit used in recent Machine Learning algorithms as its easier to train the network with this activation/squishing function) etc. The value generated after mapping or squishing the weighted activation's sum is the activation value of the next layer. The parameters or weights determine the activation of the next layer, tweaking these brightens the neuron up or down and this continues to the next layer. If we want to activate the neuron only if the sum of weighted activations is greater than a certain value known as bias, we add/subtract this to/from the sum before passing it to the logistic function (Sigmoid function…). So the activation of current layer is basically to see how positive the weighted sum of previous layer's activations including a bias is? This is done for all the neurons in the second layer with different parameters or weights along with a certain bias i.e., this particular layer has 784x16 weights and 16 biases. This can be observed as an All-to-One mapping of current layer neurons to the next layer neurons with the connections as weights and biases. This process of connected weights and biases is done to the remaining layers to generate the neural network and this particular network with 784 input layer neurons and 16 (two) hidden layer neurons and 10 output layer neurons has 13002 connected weights and biases ((784x16 + 16x16 + 16x10) weights and (16 + 16 + 10) biases) that affect the behaviour of the network.

We have a network but for this to work the computer has to find a valid configuration of these weights and biases to find the activations that recognizes a handwritten digit. Tweaking these weights and biases properly will produce a result that's accurate. Getting a proper understanding of what the weights and biases correspond to helps us evaluate the activations and helps if the network doesn't work as intended. The weighted sum can be represented as Matrix operations and the activations can be represented as follows:

$$a^{(1)} = \sigma(Wa^{(0)} + b)$$

Where W is the matrix of weights between layer(0) and layer(1), $a^{(0)}$ is the activation vector of input layer(0) and $a^{(1)}$ is the activation vector of the next layer(1), b is the bias vector between layer(0) and layer(1), and σ is the Sigmoid function.

The Network itself can be considered as a function that takes in all these weights and biases as inputs to give 10 activations as output. Next we understand how to train the network to recognize the digits which is what is the learning aspect of a Network. learning aspect of a Network.