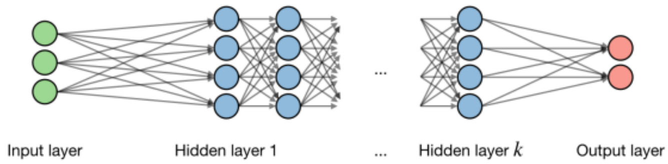
Deep Neural Networks

R Basheer Ahammad

Neural networks are a class of models that are built with layers. Commonly used types of neural networks include convolutional and recurrent neural networks.

1 Architecture

The vocabulary around neural network architectures is described in the figure below:



By noting i as the i-th layer of the network and j as the j-th hidden unit of the layer, we have:

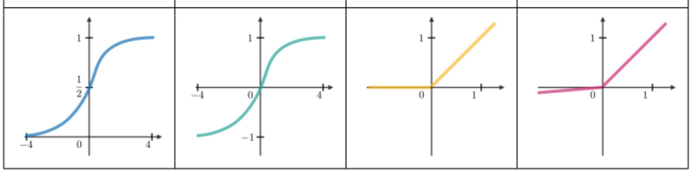
$$z[i]_j = w[i]_j^T x + b[i]_j$$

where we note w, b, and z as the weight, bias, and output respectively.

1.1 Activation Function:

Activation functions are used at the end of a hidden unit to introduce nonlinear complexities to the model. Here are the most common ones:

Sigmoid	Tanh	ReLU	Leaky ReLU
$g(z) = \frac{1}{1+e^{-z}}$	$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$	$g(z) = \max(0,z)$	$g(z) = \max(\epsilon z, z)$ with $\epsilon \ll 1$



1.2 Cross Entropy Loss:

In the domain of neural networks, the widely employed cross-entropy loss function $L(z, y)$ is defined as:

$$L(z, y) = -\left(y \log(z) + (1 - y) \log(1 - z)\right)$$

1.3 Learning Rate:

The learning rate, often denoted by η, signifies the rate at which the weights are updated. This rate can either be fixed or dynamically modified. Presently, one of the most popular approaches is Adam, a method that dynamically adjusts the learning rate.

1.4 Backpropagation:

Backpropagation is a technique used to adjust the weights within a neural network based on the comparison between the actual output and the desired output. The derivative of the loss function with respect to a weight w is computed through the chain rule, which involves the calculation of partial derivatives as shown below:

$$\frac{\partial L(z, y)}{\partial w} = \frac{\partial L(z, y)}{\partial a} \times \frac{\partial a}{\partial z} \times \frac{\partial z}{\partial w}$$

Consequently, the weight is updated using the following formula:

$$w \leftarrow w - \eta \frac{\partial L(z, y)}{\partial w}$$

Here, η represents the learning rate.

1.5 Updating Weights:

In a neural network, weights are updated as follows:

1. Take a batch of training data.
2. Perform forward propagation to obtain the corresponding loss.
3. Backpropagate the loss to get the gradients.
4. Use the gradients to update the weights of the network.

1.6 Dropout:

Dropout is a technique meant at preventing overfitting the training data by dropping out units in a neural network. In practice, neurons are either dropped with probability p or kept with probability 1-p.

1.7 Batch Normalization:

This step involves the use of hyperparameters γ and β to normalize the batch $\{x_i\}$. By considering μ_B and σ_B^2 as the desired mean and variance for the batch correction, the procedure is executed as follows:

$$x_i \leftarrow \frac{\gamma}{\sqrt{\sigma_B^2 + \epsilon}}(x_i - \mu_B) + \beta$$

This normalization step is typically applied after a fully connected or convolutional layer and before a non-linearity layer. Its objective is to facilitate the utilization of higher learning rates and decrease the heavy reliance on initializations.