ABSTRACT

DEEP LEARNING

(From Anish Singh Walia-15BIT0116)

An **Artificial-Neural Network** is a complicated learning Algorithm which is inspired form the Working of a human brain(**central Nervous System**) with ***neurons*** and ***synapses****(electrical signals and outputs of a neuron).*

In simple terms A ***Deep Neural Network*** is a Neural Network with a stack of multiple & lots of *hidden layers* in between the input units and the output units.

The *complexity* and *flexibility* of the neural network depends on the number *of* ***Hidden layers*** it has. In order to make sense and learn something really complicated such as images, videos, speech, text etc , hidden layers are very necessary.

The more the *Hidden layers* the more *complex* the network becomes and more powerful the Model becomes for representing and learning **highly non-linear complex functional mappings** between inputs and Outputs.

Basically A deep Neural Network has the power to Represent almost anything from continuous to discontinuous functional mappings between X and Y . We can enhance the power of a neural network Model by just adding more and more *Hidden layers* in between so as to make it more complex and strong.

Deep Neural Networks (Multilayer Networks) uses ***BACKPROPAGATION*** algorithm for learning the important complex structural features of the network , along with a Optimization technique called ***Gradient(derivative) Descend by*** by adjusting the internal parameters of the Network such as Weights and bias values to reduce the Loss(Error) Function and generate a accurate Model with correct Outputs.

Today Deep learning is used almost everywhere-

1) Self Driving cars
2) Transcribe speech into text
3) Image processing, Computer Vision.
4) Image captioning ,etc.

The internal parameters of the Network are ***Weights(Wi)*** . They are like *'****knobs****'* that define the output – input function of a Machine. By updating and adjusting the Weights values we can actually get the desired outputs we want with *most accuracy and least Errors*.

The internal core working of backpropagation and Gradient descend is based on the 3 key mathematical terms-

1) **Derivative – Slope of a tangent line, rate of change**
2) **Partial Derivative – derivative of a variable with others held constant-Gradient**
3) **Chain Rule**

How *BACKPROPAGATION works?*

Ans)

*Back propagation at its core is simply applying **Chain Rule** through all possible paths in the Network. We continuous propagate forward and backwards simultaneously updating weights values and reducing errors until we reach a state where we have minimum errors corresponding to some optimum weights values.*

1)**Forward Propagation**

In this process the data is fed to input layers , and we do the Weighted sum of the dot products of input values and corresponding weights. This sum of dot products are just the input signal.

Not to convert the input signals to the Output signals for that particular Input neuron we use a particular *Non-linear* function called ***ACTIVATION FUNCTION(ReLu). Activation function*** is very important for learning complex Non-linear functions . One of the most used activation function is *SIGMOID* function. It is also called Rectified linear Units –RELU , which are non-linear functions & only due to this our Model is able to learn Complex non-linear functions.

SIGMOID(a) = 1 / 1 + exp (-a) , a 'S' shaped curve.

Now the output signal of the previous layer is used as the Input signal for the Next Higher layer in the Network. This is called a **FEED-FORWARD** Network- Signals flows in one direction from inputs towards outputs.

This is how we reach the top of the Network , the output layer with some calculated Output (Y) values.

2) **Weight Updates & Backward Propagation of Errors**

In this process after reaching the top of the network we first calculate the Error values and propagate backwards in the Network with Errors . The averaged squared difference between ($Y_{actual} - Y_{calculated}$) is the **Error Term** . Now to properly adjust the Weights we calculate *the Gradient of all weights with respect to the outputs* which indicates that by what amount the error would change if weights were increased or decreased.

The Weights vector is then updated in the opposite direction of Gradients Vector. This Process is called **Gradient Descend**.

This ultimately Reduces the LOSS function and we get the desired and accurate output values with optimum weights.

Now we simply test the accuracy and performance of the Model on a unseen , random *Test Set* which was not used to train the Network. This tests the ability of the Model to **GENERALIZE**.

We can even train a Deep Neural Network in a **UNSUPERVISED** manner with an **unlabeled** data and having only some *raw inputs* with no Labels.

*This ability to train a Multilayer Neural Network in a **unsupervised** manner actually makes possible to train a network using **small** data sets and avoid **Overfitting** .*

Usually on small data sets a deep complex Neural network model will definitely ***OVERFIT due** to the high Complexity, complicated* Network, because it will **memorize** the training examples.

To avoid **Overfitting** , the general solution is using **highly complex large training data sets ,**having lots of parameters. So that the model will try to understand the underlying patterns and relations between the inputs and outputs and be able to **learn complex arbitrary functions**.

1) Convolutional Neural Networks(CNN)

It is a particular type of deep *Feed Forward* Neural Network which is much easier to train & generalizes much better.

Convolutional Neural Networks are used to process highly complicated ***unstructured*** and ***semi structured*** data which comes in the form of **multiple arrays** and which has **high dimensions.**

1) 1-D signals and sequential data-text, speech etc.

2) 2-D images with pixel intensities & audio spectrums.

3) 3-D videos or volumetric images.

4 key concepts behind CNN-

a) **Local connections b) Pooling Layers c)Shared weights d) Use of many hidden layers**

2 layers- Convolution layer-First layer of a CNN is always a *Convolutional* layer used to detect conjunctions of features from previous layers, which is to learn features from the previous layer.

Pooling layers - It reduces the *dimensionality* of each Feature map & used to reduce the computational complexity of our Neural net. It merges semantically similar features into one. A pooling unit computes the maximum of a local patch of units in 1 feature map(or more).

2 or 3 stages of convolution , Non-linear, pooling layers are stacked together to generate a complex Network followed by more Convolutional layers .

Now *Backpropagation* in a CNN is typically same as that in a usual A-NN , allowing the Weights to be trained and updated to reduce errors using *Gradient Descend*.

Recent ConvNets architectures have 10 to 20 layers of ReLUs(Rectified linear Units), hundreds of millions of weights, and billions of connections between units(synapses). Due to the Technological advancements we are now able to process such complex data in hours.

A new Regularization technique called ***DROPOUT*** is used to reduce Overfitting in CNN models. Dropout layers are added in between.

CNNs now are widely being used in *Image Captioning , Image Processing & Computer Vision* by Tech giants such as **Google , Facebook ,Microsoft** etc.

2) Recurrent Neural Networks(RNN)

It is another form of Neural network model and initially it was used when Backpropagation was introduced . For **Sequential** data and inputs use we RNNs to process such data. Such as text , speech, language .

RNNs processes a input sequence *one at a time* in a sequence . The Hidden units(layers) in between contain a *state vector* which contains all the past information about the *previous elements in the sequence.*

Rest of the process is same , we use Backpropagation and Other optimizations techniques to decrease the loss function .

The best part in RNN is that we can reconstruct and modify the activities of the raw inputs in the previous layer below.

Recent advancements in its architecture have made RNN models to very nicely predict the next alphabet in a sequence, next word in a sequence.

This power of RNNs can be used in *Natural Language Processing* applications etc.

Combination of RNNs and CNNs can be used in *Image Captioning*.

First, a CNN Model processes the image, extracting high-level features. Then an RNN runs, generating a description of the image. As it generates each word in the description, the RNN focuses on the CNN model interpretation of the relevant parts of the image.

Deep learning has changed our lives. Now due to the data explosion in the past few years and growth of *unstructured and semi-structured data* , we need some complicated techniques to analyze such ***BIG COMPLEX DATASETS.*** **It is Deep learning which has brought evolutions in analyzing and learning complex, Non-linear structures from such unstructured , high dimensional data.**

We are using applications based on deep learning and Machine learning almost **everyday** . In our smart-phones , on our laptops , internet, everywhere. From 'recommendations systems' used by Amazon to 'Siri' used in our Iphones. It is everywhere and surrounding us.Tech giants such as Facebook , Google , Microsoft , Adobe , IBMs Watson Super Computer all are using Deep Learning to enhance their applications make sense of the *Big complex datasets* and develop a shear experience for their customers and users.

References-

Deep learning Yann LeCun1,2, Yoshua Bengio3 & Geoffrey Hinton4,5-436 | NATURE | VOL 521 | 28 MAY 2015