**Each artificial neuron is modeled as follows**:

* Each neuron receives inputs.
* Adds weights and biases to the input.
* Sum inputs with weights and bias.
* This triggers the Activation function.
* Activation function will take the weighted sum of the previous step and fire the output.

##### **Supervised Learning**

In **Supervised Learning**, you give the list of Inputs and Outputs for a Neural Network to Learn.

Based on the actual output, the Neuron adjusts its weights and biases. **This is achieved by a process called** **Training**.

Neural Networks learn using **Back Propagation**.

* When the input is combined with the weights and bias to trigger the activation, it is called **forward propagation**.
* When the error estimates are propagated backward to adjust the weights and biases, it is called **backpropagation**.

##### **Steps in Back Propagation**

* During training, the input combines with random weights and bias values.
* They trigger the activation function.
* The output of the activation function is the predicted output.
* The predicted output is compared to the actual output.
* The difference is propagated backward.
* The bias values and weights are updated based on the error values.

##### **Shallow Learning**

You would have come across this term **Shallow Learning**. When does shallow learning happen in Neural Networks?

* When you just have one hidden layer between your inputs and outputs.

Another way of interpreting that idea is, you are just performing one task at a time.

##### **Shallow Learning - Tasks and Algorithms**

Some of the tasks that can be grouped under shallow learning are:

* Feature Extraction - **Preprocessing**
* Mapping specific features into vector space - **Kernel methods**
* Rule-based decision making - **Decision Trees**
* Combining various estimates - **Ensemble methods**

##### **Deep Learning**

**In deep learning, you have two or more hidden layers.** **This allows the algorithm to perform multiple tasks.**

* There are Multiple Stack Layers of Neurons
* There are Specialized layers based on data type - Image, Text, and Voice

##### **Structured Data**

* Any data that is available in Data Base format is **structured data**.
* When it comes to structured data, shallow learning models can be applied for **supervised learning**.

Unstructured Data comprise:

* Voice
* Text
* Image
* Videos

**Deep Learning Models work well for unstructured data.**

7 of 12

##### **Voice Data**

* Sequence Models work well for **Voice Data**.
* Recurrent Neural Networks can be trained for **Translation** and many **Conversational** Systems.
* Recurrent Neural Networks work well for **Text Data**.
* However, even a general multi layer perceptron can be trained for text data.
* **Word2Vec** is a popular **Deep Learning Representation of Text Data**.

When it comes to **Images**, **Convolutional Neural Networks** (**CNN**s) are the best. CNNs can be stacked to extract features and detect objects from Images.

##### **Restricted Boltzmann Machine**

**Restricted Boltzmann Machine (RBM)** was conceptualized by Geoffrey Hinton. It is used in

* **Dimension Reduction**
* **Regression**
* **Classification**.

RBM has a very simple architecture. You will learn that in detail in the following cards.

##### **RBM - Architecture**

RBMs have two layers:

* **Input**
* **Hidden**

Each input layer is linked to all the hidden layers. Each node in the input layer is not connected to one another.

In the above figure, v represents input layer and h represents hidden layer.

##### **Learning in RBM**

* RBMs learn by re-constructing the inputs.
* They learn in an unsupervised fashion.
* They re-construct the input in a backward manner and update the parameters accordingly.

##### **Deep Belief Network**

* **Deep Belief Network (DBN)** is an extension of RBM.
* DBNs are stacked RBMs layer by layer.
* The diagram above explains how they are stacked.

##### **Learning in DBN**

* DBN uses pretraining before applying backpropagation.
* This pretraining is similar to how RBM learn by reconstruction.
* Generative weights determine how parameters in one layer depend on the parameters in the previous layer.
* In the forward pass, it is the observed weight.
* In the reverse pass, it is the generative weights.

This avoids the learning from getting into local minima.

##### **Applications of DBN**

* Sequence Recognition in Videos
* Facial Recognition
* Motion capture

##### **Machines Interpreting Images**

* Machines understand images based on the pixel intensity.
* So any image in matrix representation will have numbers from 1 to 255.
  + **1** - White
  + **256** - Dark shade of Green

##### **Feature Extractions**

* Every image has the same objects represented in slightly different manner.
* Each object has to be recognized.
* Every feature should be extracted.
* How can you identify objects and extract each feature and send it to a neural network for learning?

##### **Convolution Layer**

When the image is superimposed with a filter, and run through it once, the output obtained is another image with reduced dimensions. **This one layer of the filter is called convolution layer.**

The above GIF explains a sample filter.

##### **Convoluted Image**

The image shown above is an example of a convoluted image after applying a filter.

##### **Pooling Layer**

**Pooling** **reduces the size of an image.** Only those pixels that stand out are brought out of pooling.

##### **Pooling Image**

The above-seen image is an image pooled after convolution.

You can also observe the dimensions reduced in the image after pooling.

##### **Output Layer**

The **Output layer** is the last layer where the image is converted into a 1-Dimensional matrix.

##### **The Big Picture**

This is how a typical **Convolutional Neural Network Architecture** looks like.

##### **Imagenet Database**

**ImageNet** is described as a database of millions of Images scraped from the internet that has been done for **Object Recognition Software Research**.

* All the images have been annotated by hand.
* Every year they conduct a ImageNet Large Scale Visual Recognition Challenge (ILSVRC) to gauge the accuracy of many software to classify objects correctly.

##### **Pre-Trained Models**

**There are multiple pre-trained models available for Image Classification and object detection.**

Some of them include

* AlexNet
* ResNet
* VGG19
* InceptionResNet
* GoogLeNet
* DenseNet
* NASNet

**Some of these models have participated in the ImageNet Challenge in the past.**

##### **Alex Net**

A sample AlexNet architecture for Object Detection. For detailed explanation learn [Convolutional Neural Networks](https://play.fresco.me/course/420) course.

##### **How do Pre-Trained Models Work?**

You can load any of the available pre-trained models and pass a random image. The models will output the probability of each object in that image.

##### **Dealing with Sequence Data**

You are working in a Conversational System Architecture.

* You want to leverage the power of Deep Learning for this initiative.
* What kind of Architecture will you choose?
* Will a traditional Multi-layer perceptron help?

##### **RNN for Sequence Data**

Sequence Data has a temporal or time component to it. The current component is dependent on the previous component. When you go for a normal Deep Neural Network Architecture, you cannot do justice to the temporal part.

Hence the model of choice is **Recurrent Neural Network (RNN)**.

##### **Single RNN**

The image shows the architecture of a Single RNN. During training and implementation, each RNN is stacked against multiple RNNs to Input and Output the Sequence of Data. This helps these neurons in the temporal aspect and retaining memory.

##### **Forward Propagation**

* Input is provided to the NNet at each given time step.
* In the successive time step, the output of the (n-1)'th neuron is the input of the n'th neuron in the sequence.
* This is repeated until all the information is loaded.
* Once the time steps are exhausted, the output is calculated.
* Depending on the error, it is backpropagated to the weights and biases to adjust.

##### **Back Propagation**

The image shows how the RNNs are stacked in a series. It also shows how the weights and biases are calculated in each time step.

##### **Recursive Neural Tensor Networks**

**RNTNs** are networks that are exclusively used for **Natural Language Processing**.

NLP is an application of computer science in understanding and synthesizing Natural Human Spoken Language. Since RNTS have a recursive tree structure, they are used in NLP.

##### **NLP**

**NLP** has many components to understand Natural Language, some of them include:

* Lemmatization
* Named Entity Parsing
* Part of Speech Tagging
* Segmentation

A combination of the components mentioned above is applied to Natural Language to make them Machine Understandable.

RNTNs are leveraged for a lot of these transformations.

##### **Word2vec**

**Word2vec** is a popular deep learning model where words are represented as vectors in n dimension space.

When the words are interpreted as vectors, understand the intent and context becomes easy in NLP.

Word2vec has a pre-trained corpus and vector created. Each curated word corpus is transformed into a word2vec format to understand the span of words for analysis.

##### **Autoencoders**

* **Autoencoder architecture** comprises two deep belief networks stacked against each other symmetrically.
* The number of input and output layers is the same in Autoencoders.
* They are predominantly used in Dimension Reduction as a substitute for Principal Component Analysis.

##### **Learning in Autoencoders**

Since **Autoencoders** are an extension of Deep Belief Networks, they learn in an unsupervised manner. They learn by reconstructing the input.

In de-noising autoencoders, if the input is fed with noise, the output is a de-noised version of the input.

##### **Applications**

Autoencoders are applied in the following areas:

* Data Compression
* Image Search
* Information Retrieval
* Topic Modeling