Unit - 1

Part – I - A Review of Machine Learning

Definition of Machine Learning

Tom M Mitchell, one of the patrons of machine learning, defined it as follows:

A computer program is said to learn from experience (E) with respect to some class of tasks (T) and performance measure (P), if its performance at tasks in T, as measured by P, improves with experience E.

Examples

i) Handwriting recognition learning problem

- Task T: Recognising and classifying handwritten words within images
- Performance P: Percent of words correctly classified
- Training experience E: A dataset of handwritten words with given classifications

Types of Machine Learning

Based on the methods and way of learning, machine learning is divided into mainly four types, which are:

- Supervised Machine Learning
- Unsupervised Machine Learning
- Semi-Supervised Machine Learning
- Reinforcement Learning

Supervised Learning

In supervised learning, the computer is provided with example inputs that are labeled with their desired outputs. The purpose of this method is for the algorithm to be able to "learn" by comparing its actual output with the "taught" outputs to find errors, and modify the model accordingly. Supervised learning therefore uses patterns to predict label values on additional unlabeled data. It is similar to how a teacher teaches his/her students. The teacher, or tutor, examines the performance of the students and corrects with the solution known to him/her or desired by him/her.

All supervised learning techniques are a form of classification or regression.

1. Classification techniques predict discrete responses–for example, whether an email is genuine or spam, or whether a tumour is small, medium, or large. Classification models are trained to classify data into categories. Applications include medical imaging, speech recognition, and credit scoring.

2. Regression techniques predict continuous responses–for example, changes in temperature or fluctuations in electricity demand. Applications include forecasting stock prices, handwriting recognition, and acoustic signal processing.

Unsupervised Machine Learning

Unsupervised learning is different from the Supervised learning technique; as its name suggests, there is no need for supervision. It means, in unsupervised machine learning, the machine is trained using the unlabeled dataset, and the machine predicts the output without any supervision.

In unsupervised learning, the models are trained with the data that is neither classified nor labelled, and the model acts on that data without any supervision.

The main aim of the unsupervised learning algorithm is to group or categories the unsorted dataset according to the similarities, patterns, and differences. Machines are instructed to find the hidden patterns from the input dataset.

Categories of Unsupervised Machine Learning

Unsupervised Learning can be further classified into two types, which are given below:

1) Clustering

The clustering technique is used when we want to find the inherent groups from the data. It is a way to group the objects into a cluster such that the objects with the most similarities remain in one group and have fewer or no similarities with the objects of other groups. An example of the clustering algorithm is grouping the customers by their purchasing behaviour.

Some of the popular clustering algorithms are given below:

- K-Means Clustering algorithm
- DBSCAN Algorithm
- Principal Component Analysis

2) Association

Association rule learning is an unsupervised learning technique, which finds interesting relations among variables within a large dataset. The main aim of this learning algorithm is to find the dependency of one data item on another data item and map those variables accordingly so that it can generate maximum profit. This algorithm is mainly applied in Market Basket analysis, Web usage mining, continuous production, etc.

Some popular algorithms of Association rule learning are Apriori Algorithm, FP-growth algorithm.

Semi-Supervised Learning

Semi-Supervised learning is a type of Machine Learning algorithm that lies between Supervised and Unsupervised machine learning. It represents the intermediate ground between Supervised (With Labelled training data) and Unsupervised learning (with no labelled training

data) algorithms and uses the combination of labelled and unlabeled datasets during the training period.

To overcome the drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced. The main aim of semi-supervised learning is to effectively use all the available data, rather than only labelled data like in supervised learning. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabeled data into labelled data. It is because labelled data is a comparatively more expensive acquisition than unlabeled data.

We can imagine these algorithms with an example. Supervised learning is where a student is under the supervision of an instructor at home and college. Further, if that student is self-analysing the same concept without any help from the instructor, it comes under unsupervised learning. Under semi-supervised learning, the student has to revise himself after analyzing the same concept under the guidance of an instructor at college.

Reinforcement Learning

Reinforcement learning works on a feedback-based process, in which an AI agent (A software component) automatically explore its surrounding by hitting & trail, taking action, learning from experiences, and improving its performance. Agent gets rewarded for each good action and get punished for each bad action; hence the goal of reinforcement learning agent is to maximize the rewards.

In reinforcement learning, there is no labelled data like supervised learning, and agents learn from their experiences only.

Real-world Use cases of Reinforcement Learning

- Video Games
- Resource Management
- Robotics
- Text Mining

Differences between Supervised and Unsupervised Learning

Supervised Learning	Unsupervised Learning
Supervised learning algorithms are trained using labeled data.	Unsupervised learning algorithms are trained using unlabeled data.
Supervised learning model takes direct feedback to check if it is predicting correct output or not.	Unsupervised learning model does not take any feedback.
Supervised learning model predicts the output.	Unsupervised learning model finds the hidden patterns in data.

In supervised learning, input data is provided to the model along with the output.	In unsupervised learning, only input data is provided to the model.
The goal of supervised learning is to train the model so that it can predict the output when it is given new data.	The goal of unsupervised learning is to find the hidden patterns and useful insights from the unknown dataset.
Supervised learning needs supervision to train the model.	Unsupervised learning does not need any supervision to train the model.

The Learning Machines

Fundamentally, machine learning is using algorithms to extract information from raw data and represent it in some type of model. We use this model to infer things about other data we have not yet modeled.

Neural networks are one type of model for machine learning; they have been around for at least 50 years. The fundamental unit of a neural network is a node, which is loosely based on the biological neuron in the mammalian brain. The connections between neurons are also modeled on biological brains, as is the way these connections develop over time.

In the early 2000s computational power expanded exponentially and the industry saw a "Cambrian explosion" of computational techniques that were not possible prior to this. Deep learning emerged from that decade's explosive computational growth as a serious contender in the field, winning many important machine learning competitions.

How can Machines Learn?

A computer learns something about the structures that represent the information in the raw data. Structural descriptions are another term for the models we build to contain the information extracted from the raw data, and we can use those structures or models to predict unknown data. Structural descriptions (or models) can take many forms, including the following:

• Decision trees

• Linear regression

• Neural network weights

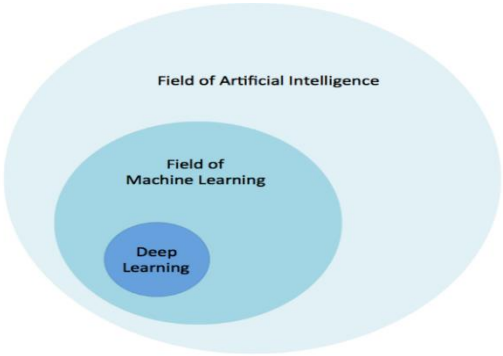
Each model type has a different way of applying rules to known data to predict unknown data. Decision trees create a set of rules in the form of a tree structure and linear models create a set of parameters to represent the input data.

Neural networks have what is called a parameter vector representing the weights on the connections between the nodes in the network.

Arthur Samuel, a pioneer in artificial intelligence (AI) at IBM and Stanford, defined machine learning as follows:

The field of study that gives computers the ability to learn without being explicitly programmed.

The field of AI is broad and has been around for a long time. Deep learning is a sub- set of the field of machine learning, which is a subfield of AI.



Biological Inspiration

Neural Network: A Neural network is an interconnected assembly of simple processing elements, units or, nodes whose functionalities loosely based on the animal neurons. The processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns.

Biological Neuron

In the human brain, learning process is performed by the nervous system. Neurons are the basic functional units of the nervous system, and they generate electrical signals called action potentials, which allows them to quickly transmit information over long distances. Almost all the neurons have three basic functions essential for the normal functioning of all the cells in the body.

These are to:

1. Receive signals (or information) from outside.

2. Process the incoming signals and determine whether or not the information should be passed along.

3. Communicate signals to target cells which might be other neurons or muscles or glands.

Basic Parts of a neuron

A biological neuron is mainly composed of 3 main parts and an external part called synapse.

1. Dendrite

Dendrites are responsible for getting incoming signals from outside.

2. Soma

Soma is the cell body responsible for the processing of input signals and deciding whether a neuron should fire an output signal.

3. Axon

Axon is responsible for getting processed signals from neuron to relevant cells.

4. Synapse

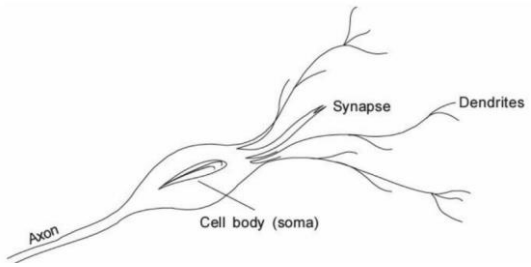
Synapse is the connection between an axon and other neuron dendrites.

Working of the Biological Neuron

The task of receiving the incoming information is done by dendrites, and processing generally takes place in the cell body. Incoming signals can be either excitatory —which means they tend to make the neuron fire (generate an electrical impulse) — or inhibitory — which means that they tend to keep the neuron from firing.

Most neurons receive many input signals throughout their dendritic trees. A single neuron may have more than one set of dendrites and may receive many thousands of input signals. Whether or not a neuron is excited into firing an impulse depends on the sum of all of the excitatory and inhibitory signals it receives. The processing of this information happens in soma which is neuron cell body. If the neuron does end up firing, the nerve impulse, or action potential, is conducted down the axon.

Towards its end, the axon splits up into many branches and develops bulbous swellings known as axon terminals (or nerve terminals). These axon terminals make connections on target cells.



Relation between Artificial Neuron and Biological Neuron

The human brain no doubt is a highly complex structure viewed as a massive, highly interconnected network of simple processing elements called neurons. However, the behaviour of a neuron can be captured by a simple model of artificial neuron.

We pass input values to a neuron using input layer. It might be something as simple as a collection of array values. It is similar to a dendrite in biological neurons.

Synapses act like a weight of the incoming stimulus and inspired the weights of ANN; dendrites accumulate the incoming weighted stimulus, inspired the summing function of ANN; cell body, that causes conversion of summed stimulus in to a new stimulus, inspires activation function; axon, which distributes the new stimulus to the corresponding neurons, inspires the output and output links; and lastly, threshold value with a role of activating or inactivating increase and decrease of the stimulus, inspires the bias.

Biological Neural Networks	Artificial Neural Networks
Stimulus	Input
Receptors	Input Layer
Neural Net	Processing Layer(s)
Neuron	Processing Element
Effectors	Output Layer
Response	Output and an entry

Differences between Artificial Neural Network and Biological Neural Network

ANN	BNN
Faster in speed.	Slower in speed.
Adaptability is not possible because new information destroys the old information.	Adaptability is possible, because new information is added without destroying old information.
Not fault-tolerant: It means, if information corrupted in the memory cannot be restored back.	Fault-tolerant: It means, irrespective of faults in the network connections, information is still preserved.
Memory and processing are separate.	Memory and processing elements are collocated.
Sequential and synchronous.	Parallel and asynchronous.
The activities are continuously monitored by a control unit.	There is no control unit to monitor the information being processed into the network.

Main Properties of ANN

There are two main properties of artificial neural networks that follow the general idea of how the brain works.

1. The most basic unit of the neural network is the artificial neuron. These artificial neurons pass on information they receive to other artificial neurons, often with transformations.

2. As the neurons in the brain can be trained to pass forward only signals that are useful in achieving the larger goals of the brain, we can train the neurons of a neural network to pass along only useful signals.

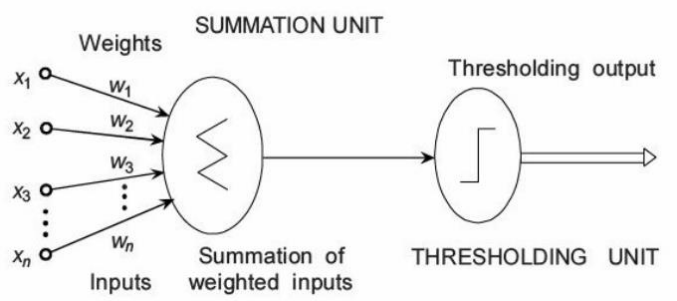
Model of Artificial Neuron

The basic model of the Neuron is based on the method of information that is to be processed. It can be summarized as follows in a network:

Signals (action potentials) appear at the unit's inputs.

Multiplying the signal by some number to indicate the strength of the synapse.

The weighted signals have to be summed to produce the overall units. If this activation exceeds a certain threshold value the unit produces an output response. This functionality is captured in the artificial neuron known as Threshold Logic Unit (TLU) originally proposed by McCulloch and Pitts (1943).



Here, x_1, x_2, x_3, ..., x_n are the n inputs to the artificial neuron. w_1, w_2, ... , w_n are the weights attached to the input links.

The total Input I received by the soma of the artificial neuron is

$$I = w_1x_1 + w_2x_2 + w_3x_3 + \ldots\ldots\ldots\ldots + w_nx_n$$

$$= \sum_{i=1}^{n} w_i x_i$$

Activation Function: The activation function is used to calculate the output of a neuron. To generate the final output y, the sum is passed on to a non-linear filter Φ called Activation function or Transfer function or Squash function which releases the output.

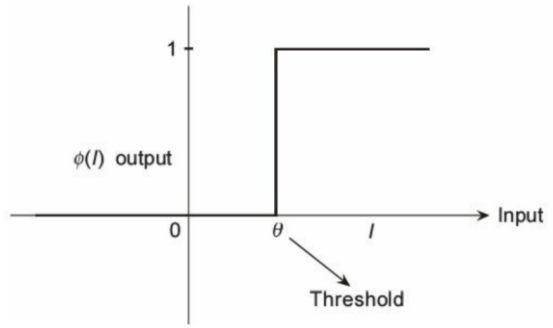
$$y = \Phi(I)$$

Threshold value: The threshold function is sometimes used to qualify the output of a neuron in the output layer. It is clear that the net value or activation is compared with the threshold and the neuron fires if the threshold value is exceeded.

$$y = \Phi(\sum_{i=1}^{n} w_i x_i - \theta)$$

where Φ is the step activation function known as Heaviside function and is such that

$$\emptyset(I) = \begin{cases} 1, & I > 0 \\ 0, & I \leq 0 \end{cases}$$

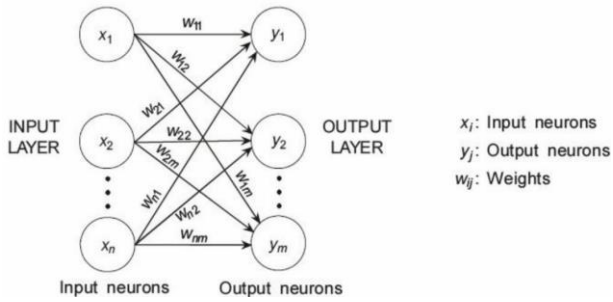


Neural Network Architectures

There are several classes of NN, classified according to their learning mechanisms. However, we identify three fundamentally different classes of Networks. All the three classes employ the digraph structure for their representation.

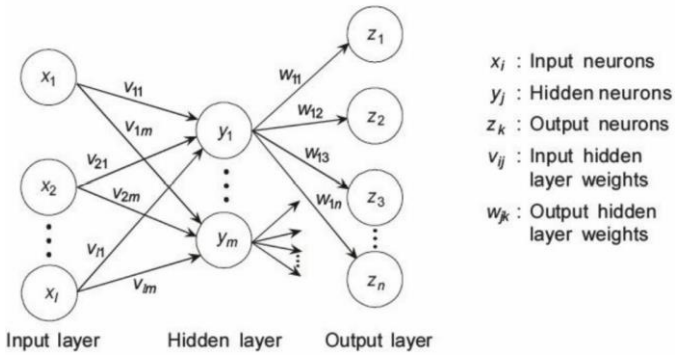
Single Layer Feedforward Network

This type of network comprises of two layers, namely the input layer and the output layer. The input layer neurons receive the input signals and the output layer neurons receive the output signals. The synaptic links carrying the weights connect every input neuron to the output neuron but not vice-versa. Such a network is said to be feedforward in type or acyclic in nature. Despite the two layers, the network is termed single layer since it is the output layer, alone which performs computation. The input layer merely transmits the signals to the output layer. Hence, the name single layer feedforward network.



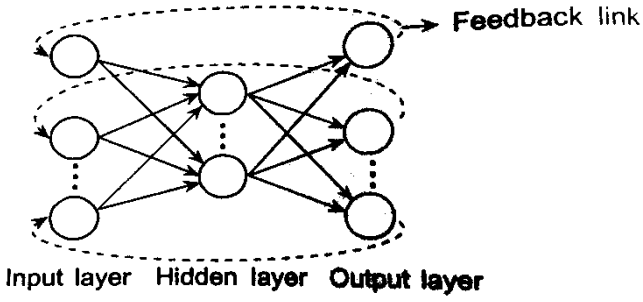
Multilayer Feedforward Network

This network, as its name indicates is made up of multiple layers. Thus, architectures of this class besides possessing an input and an output layer also have one or more intermediary layers called hidden layers. The computational units of the hidden layer are known as the hidden neurons or hidden units. The hidden layer aids in performing useful intermediary computations before directing the input to the output layer. The input layer neurons are linked to the hidden layer neurons and the weights on these links are referred to as input-hidden layer weights. Again, the hidden layer neurons are linked to the output layer neurons and the corresponding weights are referred to as hidden-output layer weights. A multilayer feedforward network with l input neurons, m1neurons in the first hidden layer, m2 neurons in the second hidden layer and n output neurons in the output layer is written as l − m1 − m2 − n.



Recurrent Networks

These networks differ from feedforward network architectures in the sense that there is atleast one feedback loop. Thus, in these networks, for example, here could exist one layer with feedback connections. There could also be neurons with self-feedback links, i.e. the output of a neuron is fed back into itself as input.



RNN have a “memory” which remembers all information about what has been calculated. It uses the same parameters for each input as it performs the same task on all the inputs or hidden layers to produce the output. This reduces the complexity of parameters, unlike other neural networks.

What is Deep Learning?

One useful definition specifies that deep learning deals with a “neural network with more than two layers.”

Following are some of the facets in this evolution of neural networks:

• More neurons than previous networks

• More complex ways of connecting layers/neurons in NNs

• Explosion in the amount of computing power available to train

• Automatic feature extraction

We’ll define deep learning as neural networks with a large number of parameters and layers in one of four fundamental network architectures:

• Unsupervised pretrained networks

• Convolutional neural networks

• Recurrent neural networks

• Recursive neural networks

Part – II. Fundamentals of Deep Networks

Defining Deep Learning

The term "deep" usually refers to the number of hidden layers in the neural network.

Deep learning is a subset of machine learning, which is predicated on idea of learning from example. In machine learning, instead of teaching a computer a massive list of rules to solve the problem, we give it a model with which it can evaluate examples, and a small set of instructions to modify the model when it makes a mistake.

The basic idea of deep learning is that repeated composition of functions can often reduce the requirements on the number of base functions (computational units) by a factor that is exponentially related to the number of layers in the network.

Deep learning eliminates some of data pre-processing that is typically involved with machine learning.

For example, let's say that we had a set of photos of different pets, and we wanted to categorize by "cat" and "dog". Deep learning algorithms can determine which features (e.g. ears) are most important to distinguish each animal from another. In machine learning, this hierarchy of features is established manually by a human expert.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

Deep learning classifies information through layers of neural networks, which have a set of inputs that receive raw data. For example, if a neural network is trained with images of birds, it can be used to recognize images of birds. More layers enable more precise results, such as distinguishing a crow from a raven as compared to distinguishing a crow from a chicken.

Deep Learning consists of the following methods and their variations:

a) Unsupervised leaming systems such as Boltzman machines for preliminary training, Auto-encoders, Generative adversarial network.

b) Supervised learning such as Convolution neural networks which brought technoogy of pattern recognition to a new level.

c) Recurrent neural networks, allowing to train on processes in time.

d) Recursive neural networks, allowing to include feedback between circuit elements and chains.

Reasons for Using Deep Learning

1. Analyzing unstructured data: Deep learning algorithms can be trained to look at text data by analyzing social media posts, news, and surveys to provide valuable business and customer insights.

2. Data labelling: Deep learning requires labeled data for training. Once trained, it can label new data and identify different types of data on its own.

3. Feature engineering: A deep learning algorithm can save time because it does not require humans to extract features manually from raw data.

4. Efficiency: When a deep learning algorithm is properly trained, it can perform thousands of tasks over and over again, faster than humans.

5. Training: The neural networks used in deep learning have the ability to be applied to many different data types and applications. Additionally, a deep learning model can adapt by retraining it with new data.

Applications of Deep Learning

1. Aerospace and defense: Deep learning is utilized extensively to help satellites identify specific objects or areas of interest and classify them as safe or unsafe for soldiers.

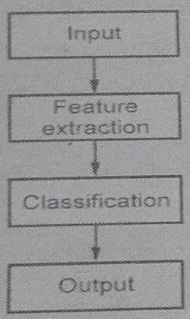
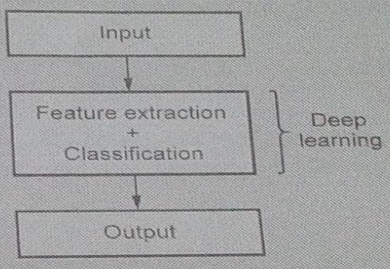
2. Financial services: Financial institutions regularly use predictive analytics to drive algorithmic trading of stocks, assess business risks for loan approvals, detect fraud, and help manage credit and investment portfolios for clients.

3. Medical research: The medical research field uses deep learning extensively. For example, in ongoing cancer research, deep learning is used to detect the presence of cancer cells automatically.

4. Industrial automation: The heavy machinery sector is one that requires a large number of safety measures. Deep leaming helps with the improvement of worker safety in such environments by detecting any person or objects that comes within the unsafe radius of a heavy machine.

5. Facial recognition: This feature utilizing deep learning is being used not just for a range of security purposes but will soon enable purchases at stores. Facial recognition is already being extensively used in airports to enable seamless, paperless check-ins.

Differences between ML and DL

Sr. No.	Machine Learning	Deep Learning
1.	Machine learning uses algorithms to parse data, learn from that data, and make informed decisions based on what it has learned.	Deep learning structures algorithms in layers to create an "artificial neural network" that can learn and make intelligent decisions on its own.
2.	Machine learning gives lesser accuracy.	Deep learning gives more accuracy.
3.	Machine learning requires less time for training.	Deep learning requires more time for training.
4.	Needs accurately identified features by human intervention.	It can create new features.
5.	Machine learning models mostly require data in a structured form.	Deep Learning models can work with structured and unstructured data both as they rely on the layers of the Artificial neural network.
6.	Algorithms are detected by data analysts to examine specific variables in data sets.	Algorithms are largely self-directed on data analysis once they are put into production.
7.	Machine learning can work on low-end machines.	Deep learning model needs a huge amount of data to work efficiently, so they need GPU's and hence the high-end machine
8.		

Advantages and Disadvantages of Deep Learning

Advantages of Deep Learning

No need for feature engineering.

DL solves the problem on the end-to-end basis.

Deep learning gives more accuracy.

Disadvantages of Deep Learning

DL needs high-performance hardware.

DL needs much more time to train.

It is very difficult to assess its performance in real world applications.

It is very hard to understand.

Evolutionary progress and resurgence (History of DL)

Neural networks had entered a "winter period" in the mid-1980s when the promise of AI fell short of what it could deliver.

One important development in neural networks was Yann LeCun's work at AT&T Bell Labs on optical character recognition. His lab was focused on check image recognition for the financial services sector. Through this work, LeCun and his team developed the concept of the biologically inspired model of image recognition we know today as CNN.

As time went on, the research community created better artificial neuron variants (e.g., Long Short-Term Memory [LSTM] Memory Cell and Memory Cell with Forget Gate) over the course of the late 1990s.

During the 2000s, researchers and industry applications began to progressively apply these advances in products such as the following:

• Self-driving cars

• Google Translate

• Amazon Echo

In the near future, we'll continue to see deep learning being applied in unique and innovative ways.

Advances in network architecture

As research pressed the state of the art forward from multilayer feed-forward net- works toward newer architectures like CNNs and Recurrent Neural Networks, the discipline saw changes in how layers were set up, how neurons were constructed, and how we connected layers. Network architectures evolved to take advantage of specific types of input data.

Advances in layer types. Layers became more varied with the different types of architectures. Deep Belief Networks (DBNs) demonstrated success with using Restricted Boltzmann Machines (RBMs) as layers in pretraining to build features.

Advances in neuron types. Recurrent Neural Networks specifically created advancements in the types of neurons (or units) applied in the work around LSTM networks.

Hybrid architectures. Continuing the theme of matching input data to architecture type, we have seen hybrid architectures emerge for types of data that has both a time domain and image data involved.

Generative modeling

Generative modeling is not a new concept, but the level to which deep networks have taken it has begun to rival human creativity. From generating art to generating music to even writing beer reviews, we see deep learning applied in creative ways every day. Recent variants of generative modeling to note include the following:

• Inceptionism

• Modeling artistic style

• Generative Adversarial Networks

• Recurrent Neural Networks

Inceptionism. Inceptionism is a technique in which a trained convolutional network is taken with its layers in reverse order and given an input image coupled with a prior constraint. The images are modified iteratively to enhance the output in a manner that could be described as "hallucinative."

Modeling artistic style. Variants of convolutional networks have shown to learn the style of specific painters and then generate a new image in this style of arbitrary photographs.

GANs. The generative visual output of a GAN can best be described as synthesizing novel images by modeling the distributions of input data seen by the network.

Recurrent Neural Networks. Recurrent Neural Networks have been shown to model sequences of characters and generate new sequences that are lucidly coherent.

Part – III. Common Architectural Principles of Deep Networks

1. Parameters

These are the coefficients of the model, and they are chosen by the model itself. It means that the algorithm, while learning, optimizes these coefficients (according to a given optimization strategy) and returns an array of parameters which minimize the error. To give an example, in a linear regression task, you have your model that will look like y=b + ax, where b and a will be your parameter. The only thing you have to do with those parameters is initializing them.

2. Layers

This is the highest-level building block in deep learning. Layers are made up of nodes, which take one of more weighted input connections and produce an output connection. They're organised into layers to comprise a network. A variety of functions are performed by a variety

of layers, each layer possesses its own utility and significance. Many such layers, together form a Neural Network, i.e., the foundation of Deep Learning.

Layers are a fundamental architectural unit in deep networks. In DL4J we customize a layer by changing the type of activation function it uses.

We'll also look at how you can use combinations of layers to achieve a goal (e.g., classification or regression). Finally, we'll also explore how each type of layer requires different hyperparameters (specific to the architecture) to get our net- work to learn initially. Further hyperparameter tuning can then be beneficial through reducing overfitting.

3. Activation Functions

The activation function is used to calculate the output of a neuron. To generate the final output y, the sum is passed on to a non-linear filter Φ called Activation function or Transfer function or Squash function which releases the output. Activation functions are of two types: Linear and Non-Linear.

The activation function compares the input value to a threshold value. If the input value is greater than the threshold value, the neuron is activated. It's disabled if the input value is less than the threshold value, which means its output isn't sent on to the next or hidden layer.

The input to the activation function is the sum which is defined by the equation:

$$I = w_1x_1 + w_2x_2 + w_3x_3 + \ldots\ldots\ldots\ldots + w_nx_n$$

$$= \sum_{i=1}^{n} w_i x_i$$

Hidden layer activation functions.

Commonly used functions include:

• Sigmoid

• Tanh

• Hard tanh

• Rectified linear unit (ReLU) (and its variants)

Output layer for regression.

This design decision is motivated by what type of answer we expect our model to output. If we want to output a single real-valued number from our model, we'll want to use a linear activation function.

Output layer for binary classication.

In this case, we'd use a sigmoid output layer with a single neuron to give us a real value in the range of 0.0 to 1.0 (excluding those values) for the single class. This real-valued output is typically interpreted as a probability distribution.

Output layer for multiclass classication.

If we have a multiclass modeling problem yet we only care about the best score across these classes, we'd use a softmax output layer with an arg-max() function to get the highest score of all the classes. The softmax output layer gives us a probability distribution over all the classes.

Necessity of incorporating the activation function:

The primary purpose of the activation functions that are used in output layers of ML models is to squash the value between a bounded range like 0 to 1.

The primary purpose of the activation functions that are used in hidden layers of neural networks is to provide non-linearity without which neural networks cannot model non-linear relationships.

In broad terms, activation functions are necessary to prevent linearity. Without them, the data would pass through the nodes and layers of the network only going through linear functions (a*x + b). The composition of these linear functions is again a linear function and so no matter how many layers the data goes through, the output is always the result of a linear function.

However, depending upon the properties of the problem we might be able to make a better choice for easy and quicker convergence of the network.

- Sigmoid functions and their combinations generally work better in the case of classifiers.
- Sigmoids and tanh functions are sometimes avoided due to the vanishing gradient problem.
- ReLU function is a general activation function and is used in most cases these days.
- If we encounter a case of dead neurons in our networks the leaky ReLU function is the best choice.

Important Activation Functions

1. Sigmoid or Logistic Activation Function

A sigmoid function produces a curve with an "S" shape. The example sigmoid function shown on the left is a special case of the logistic function, which models the growth of some set.

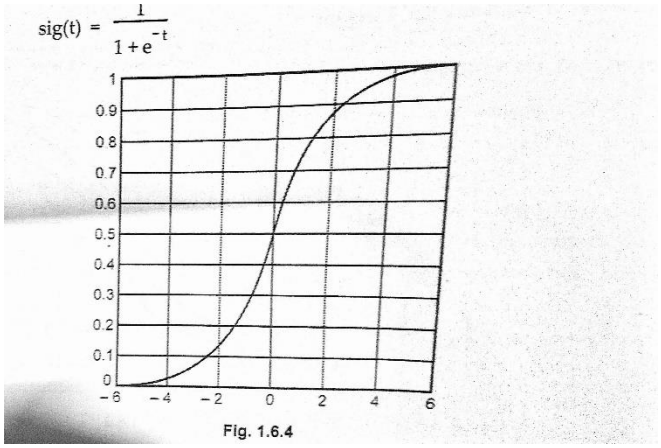


Fig. 1.6.4

In general, a sigmoid function is real-valued and differentiable, having a non- negative or non-positive first derivative, one local minimum, and one local maximum.

The logistic sigmoid function is related to the hyperbolic tangent as follows:

$$1 - 2\,sig(x) = 1 - 2\frac{1}{1+e^{-x}} = -\tanh\frac{x}{2}$$

Sigmoid functions are often used in artificial neural networks to introduce nonlinearity in the model.

A neural network element computes a linear combination of its input signals, and applies a sigmoid function to the result.

A reason for its popularity in neural networks is because the sigmoid function satisfies a property between the derivative and itself such that it is easy to perform.

$$\frac{d}{dt}sig(t) = sig(t)\,(1 - sig(t))$$

learning algorithms.

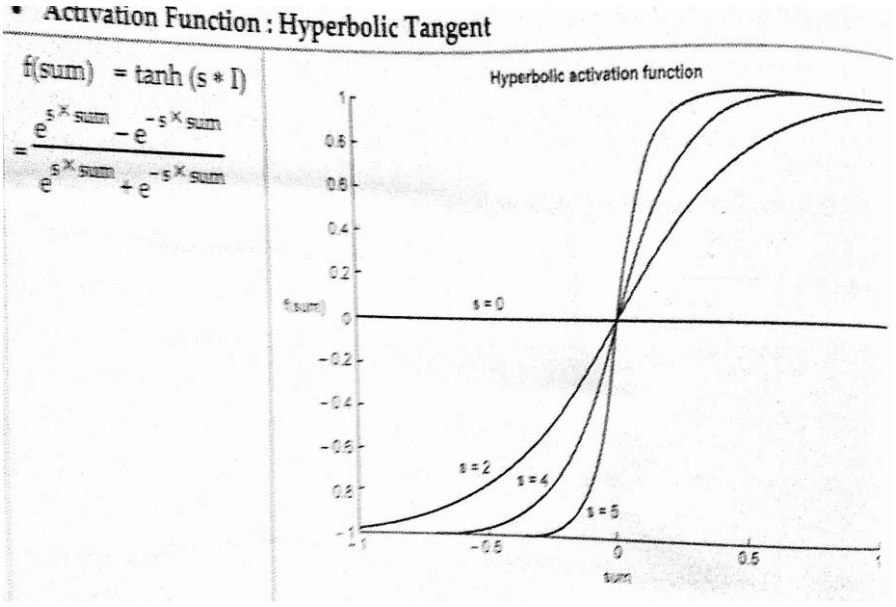
2. Hyperbolic Tangent Function

tanh is also like logistic sigmoid but better. The range of the tanh function is from (-1 to 1). tanh is also sigmoidal (s - shaped).

The advantage is that the negative inputs will be mapped strongly negative and the zero inputs will be mapped near zero in the tanh graph.

The function is **differentiable**.

The function is **monotonic** while its **derivative is not monotonic**.



3. ReLU

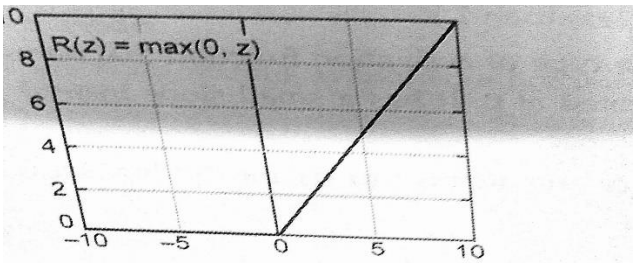
Rectified Linear Unit (ReLU) solve the vanishing gradient problem. ReLU is a non- linear function or piecewise linear function that will output the input directly if it is positive, otherwise, it will output zero.

It is the most commonly used activation function in neural networks, especially in Convolutional Neural Networks (CNNs) and Multilayer perceptron's.

Mathematically, it is expressed as

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

where x: input to neuron



The derivative of an activation function is required when updating the weights during the back-propagation of the error. The slope of ReLU is 1 for positive values and 0 for negative values. It becomes non-differentiable when the input x is zero it can be safely assumed to be zero and causes no problem in practice.

ReLU is used in the hidden layers instead of Sigmoid or tanh. The ReLU function solves the problem of computational complexity of the Logistic Sigmoid and Tanh functions.

Advantages of ReLU function

a) ReLU is simple to compute and has a predictable gradient for the backpropagation of the error.

b) Easy to implement and very fast.

c) The calculation speed is very fast. The ReLU function has only a direct relationship.

d) It can be used for deep network training.

Disadvantages of ReLU function

a) When the input is negative, ReLU is not fully functional which means when it comes to the wrong number installed, ReLU will die. This problem is also known as the Dead Neurons problem.

b) ReLU function can only be used within hidden layers of a Neural Network model.

4. LReLU

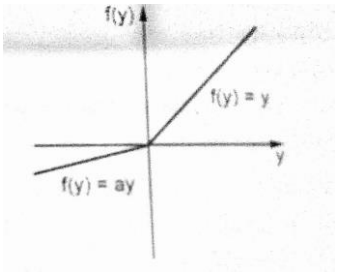
The Leaky ReLU is one of the most well-known activation functions. It is the same as ReLU for positive numbers. But instead of being 0 for all negative values, it has a constant slope (less than 1.).

Leaky ReLU is a type of activation function that helps to prevent the function from becoming saturated at 0. It has a small slope instead of the standard ReLU which has an infinite slope.

Leaky ReLUs are one attempt to fix the "dying ReLU" problem.

The leak helps to increase the range of the ReLU function. Usually, the value of a is 0.01 or so.

The motivation for using LReLU instead of ReLU is that constant zero gradients can also result in slow learning, as when a saturated neuron uses a sigmoid activation function.

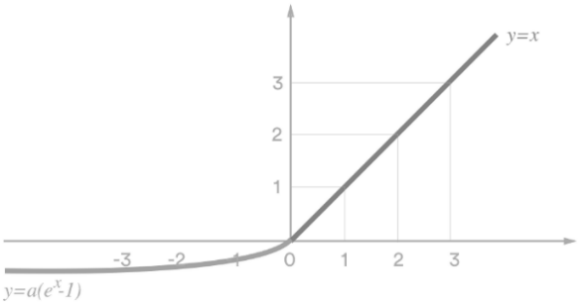


5. EReLU

An Elastic ReLU (EReLU) considers a slope randomly drawn from a uniform distribution during the training for the positive inputs to control the amount of non- linearity.

The ERELU is defined as: EReLU(x) = max(Rx; 0) in the output range of [0,1) where R is a random number

At the test time, the ERELU becomes the identity function for positive inputs.

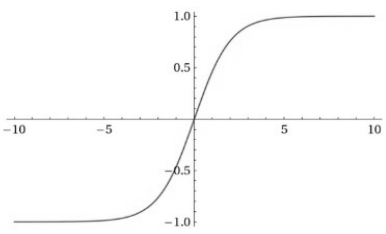


6. Softmax

This function is a suitable function for classification problems involving a multi-class target variable.

Softmax returns the probability result of each class as output. So if you are building a neural network with Softmax, you need to have as many neurons as your target variable in your output layer.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{for } j = 1, \ldots, K.$$



4. Loss Functions

Loss functions quantify the agreement between the predicted output (or label) and the ground truth output. We use loss functions to determine the penalty for an incorrect classification of an input vector.

Some loss functions: • Squared loss • Logistic loss • Hinge loss • Negative log likelihood

In simple terms, the Loss function is a method of evaluating how well your algorithm is modeling your dataset. It is a mathematical function of the parameters of the machine learning algorithm.

If the value of the loss function is lower then it's a good model otherwise, we have to change the parameter of the model and minimize the loss.

Loss functions in Deep Learning

1. Regression
    1. MSE(Mean Squared Error)
    2. MAE(Mean Absolute Error)
    3. Hubber loss
2. Classification
    1. Binary cross-entropy
    2. Categorical cross-entropy
3. AutoEncoder
    1. KL Divergence
4. GAN
    1. Discriminator loss
    2. Minmax GAN loss

5. Hyperparameters

Hyperparameters are parameters whose values control the learning process and determine the values of model parameters that a learning algorithm ends up learning.

While designing a machine learning model, one always has multiple choices for the architectural design for the model. This creates a confusion on which design to choose for the model based on its optimality. And due to this, there are always trials for defining a perfect machine learning model.

The parameters that are used to define these machine learning models are known as the hyperparameters and the rigorous search for these parameters to build an optimized model is known as hyperparameter tuning.

Hyperparameters are not model parameters, which can be directly trained from data. Model parameters usually specify the way to transform the input into the required output, whereas hyperparameters define the actual structure of the model that gives the required data.

(a) Layer Size

Layer size is defined by the number of neurons in a given layer. Input and output layers are relatively easy to figure out because they correspond directly to how our modeling problem handles input and output.

For the input layer, this will match up to the number of features in the input vector. For the output layer, this will either be a single output neuron or a number of neurons matching the number of classes we are trying to predict.

It is obvious that a neural network with 3 layers will give better performance than that of 2 layers. Increasing more than 3 doesn't help that much in neural networks. In the case of CNN, an increasing number of layers makes the model better.

(b) Magnitude: Learning Rate

The amount that the weights are updated during training is referred to as the step size or the learning rate. Specifically, the learning rate is a configurable hype parameter used in the training of neural networks that has a small positive vale often in the range between 0.0 and 1.0.

For example, if learning rate is 0.1, then the weights in the network are updated 0.1 (estimated weight error) or 10% of the estimated weight error each time the weights are updated. The learning rate hyper-parameter controls the rate or speed at which the model learns.

Large learning rates () make the model learn faster but at the same time it may cause us to miss the minimum loss function and only reach the surrounding of it. In cases where the learning rate is too large, the optimizer overshoots the minimum and the loss updates will lead to divergent behaviours.

On the other hand, choosing lower learning rate values gives a better chance of finding the local minima with the trade-off of needing larger number of epochs and more time.

Momentum can accelerate learning on those problems where the high-dimensional weight space that is being navigated by the optimization process has structures that mislead the gradient descent algorithm, such as flat regions or steep curvature.

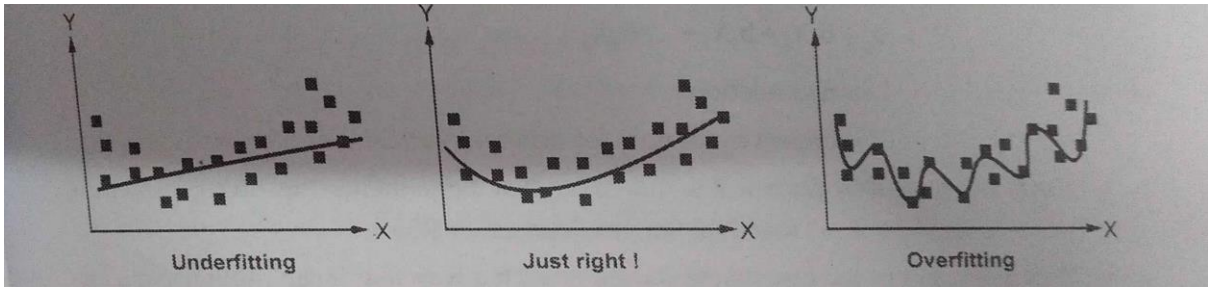
AdaGrad. AdaGrad is one technique that has been developed to help augment finding the “right” learning rate.

RMSProp. RMSprop is a very effective, but currently unpublished adaptive learning rate method.

AdaDelta. AdaDelta is a variant of AdaGrad that keeps only the most recent history rather than accumulating it like AdaGrad does.

ADAM. ADAM derives learning rates from estimates of first and second moments of the gradients.

(c) Regularization



Just have a look at the above figure, and we can immediately predict that once we try to cover every minutest feature of the input data, there can be irregularities in the extracted features, which can introduce noise in the output. This is referred to as "Overfitting".

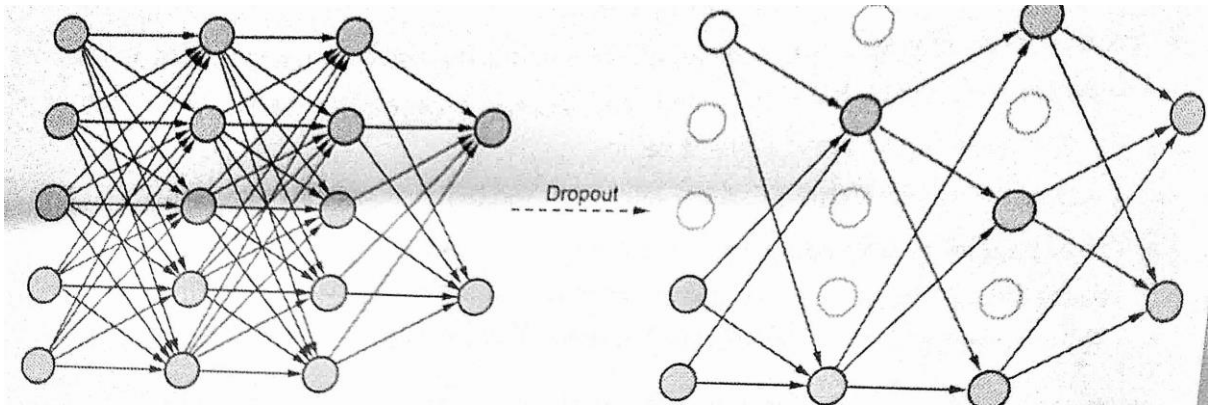
This may also happen with the lesser number of features extracted as some of the important details might be missed out. This will leave an effect on the accuracy of the outputs produced. This is referred to as "Underfitting".

This also shows that the complexity for processing the input elements increases with overfitting. Also, neural networks being a complex interconnection of nodes, the issue of overfitting may arise frequently.

To eliminate this, regularization is used, in which we have to make the slightest modification in the design of the neural network, and we can get better outcomes.

(d) Dropout

Dropout was introduced by "Hinton et al" and this method is now very popular. It consists of setting to zero the output of each hidden neuron in chosen layer with some probability and is proven to be very effective in reducing overfitting.



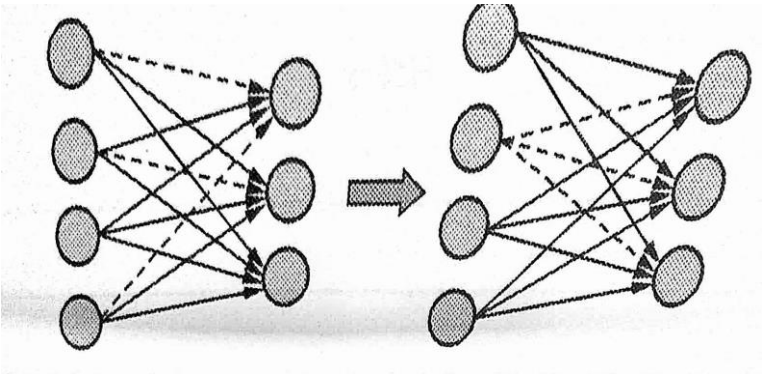
To achieve dropout regularization, some neurons in the artificial neural network are randomly disabled. That prevents them from being too dependent on one another as they learn the correlations. Thus, the neurons work more independently, and the artificial neural network learns multiple independent correlations in the data based on different configurations of the neurons.

It is used to improve the training of neural networks by omitting a hidden unit. It also speeds training.

Dropout is driven by randomly dropping a neuron so that it will not contribute to the forward pass and back-propagation.

(e) DropConnect

DropConnect, known as the generalized version of Dropout, is the method used for regularizing deep neural networks.



DropConnect has been proposed to add more noise to the network. The primary difference is that instead of randomly dropping the output of the neurons, we randomly drop the connection between neurons.

In other words, the fully connected layer with DropConnect becomes a sparsely connected layer in which the connections are chosen at random during the training stage.

Feed forward vs Feed backward Networks

Feed-forward ANNs allow signals to travel one way only: from input to output. There are no feedback (loops); *i.e.*, the output of any layer does not affect that same layer. Feed-forward ANNs tend to be straightforward networks that associate inputs with outputs. They are extensively used in pattern recognition. This type of organisation is also referred to as bottom-up or top-down.

Feedback (or recurrent or interactive) networks can have signals traveling in both directions by introducing loops in the network. Feedback networks are powerful and can get extremely complicated. Computations derived from earlier input are fed back into the network, which gives them a kind of memory. Feedback networks are dynamic; their 'state' is changing continuously until they reach an equilibrium point. They remain at the equilibrium point until the input changes and a new equilibrium needs to be found.