Introduction to Deep Learning

R. Indrakumari, T. Poongodi, and Kiran Singh

1 Introduction

The human brain is the incredible organ that dictates the signals received from sound, sight, smell, touch, and taste. The brain stores emotions, experiences, memories, and even dreams. The brain takes decisions and solves many problems that even the powerful supercomputers lack [1]. Based on this, researchers are dreamed of constructing intelligent machines like the brain. Later researchers invented robots to assist human activities, automatic disease detection microscopes, and self-driving cars. These inventions still required human interventions to do some computational problems. To tackle this problem, researchers want to build a machine that can learn by themselves and solve more complex problems in the speed of the human brain. These necessities pave the way to the most active field of artificial machine intelligence called deep learning.

2 Neurons

The basic unit of the human brain is the neurons. Very small portions of the brain, about the size of wheat, have over 10,000 neurons with more than 6000 connections with other neurons [2]. The information perceived by the brain is captured by the neurons, and the same is passed from a neuron to others for processing, and the final result is sent to other cells. It is depicted in Fig. 1. Dendrites are an antenna-like structure in the neurons that receives the inputs. Based on the frequency of usage,

R. Indrakumari (✉) · T. Poongodi · K. Singh
School of Computing Science and Engineering, Galgotias University,
Greater Noida, Uttar Pradesh, India

K. B. Prakash et al. (eds.), *Advanced Deep Learning for Engineers and Scientists*, EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-030-66519-7_1

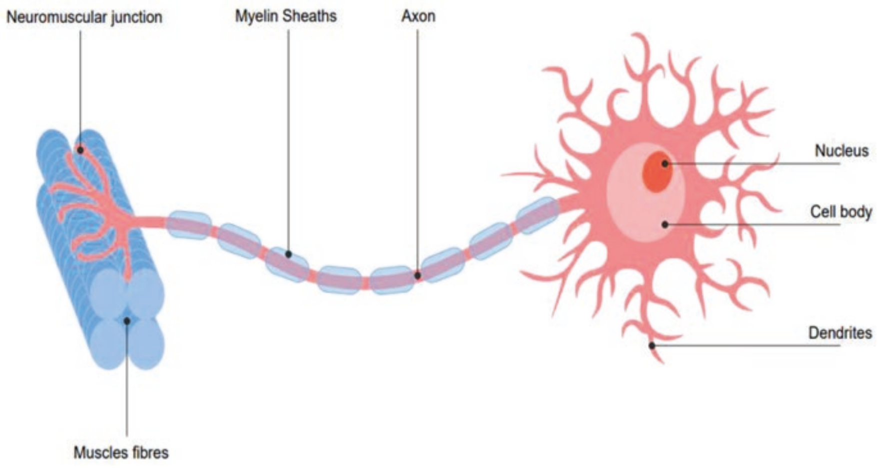

Fig. 1 Biological neuron's structure

the inputs are classified into strengthened and weakened. The connection strength estimates the involvement of the input pertaining to the neuron's output. The input signals are weighted by the connection strength and summed collectively in the cell body. The calculated sum takes the form of a new signal, and it is thriven along the cell's axon to reach the destination neurons.

In 1943, Warren S. McCulloch and Walter H. Pitts [3] concentrated on the functional understanding of the neurons that exist in the human brain and created a computer-based artificial model as shown in Fig. 2.

As in the biological neurons, the artificial neuron receives inputs x_1, x_2, $x_3 \ldots x_n$, and respectively input is multiplied by particular weights w_1, w_2, $w_3, \ldots w_n$, and the calculated sum is considered to make the *logit* of the neuron:

$$Z = \sum_{i=0}^{n} w_i x_i \tag{1}$$

Some logit may include a constant value called the bias. Finally, the logit is passed through a function f to make the desired output y = f (z).

3 History of Deep Learning

The history of deep learning started in the early 1940s when Warren McCulloch and Walter Pitts developed a computer model focusing on the human neural system. They applied mathematics and algorithms and called it "threshold logic" to imitate the thinking process. Deep learning is a subsequent derivative of machine learning that applies algorithms, processes the data, and develops abstractions. Various

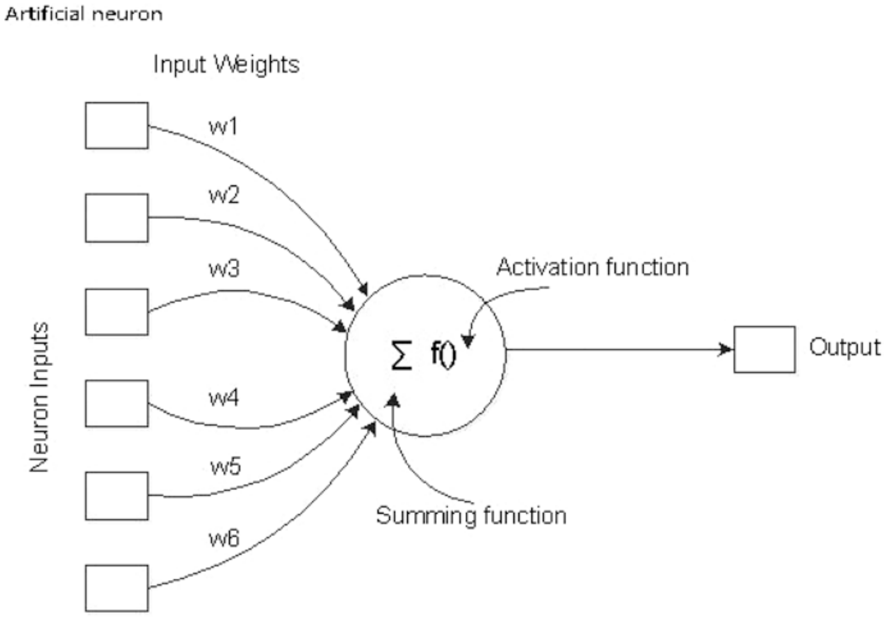

Fig. 2 Neuron in an artificial neural net

algorithms are applied to process data, to recognize objects and human speech. The output of the former layer is provided as the input to the next layer.

In 1960 Henry J. Kelley has started to develop the Backpropagation Model and was extended by Stuart Dreyfus in 1962. The early version of Backpropagation was not so efficient and clumsy. Following this, in 1965, Valentin Grigor'evich Lapa has proposed cybernetics and forecasting techniques, and Alexey Grigoryevich Ivakhnenko has proposed the data handling methodology using polynomial activation functions. The best feature chosen statistically is forwarded to the next layer manually.

Kunihiko Fukushima has developed the first convolutional neural networks with multiple pooling and convolutional layers. Later in 1979, he developed neocognitron, a multilayered and hierarchical artificial neural network design that can recognize visual patterns. Neocognitron is said to be the best model at that time as it uses new learning methods with top-down connections. It contains the selective attention model which recognizes the individual patterns. The developed neocognitron can be able to identify the unknown and missing information with a concept called inference.

In the late 1970s, Seppo Linnainmaa wrote a Fortran code for backpropagation. In 1985, Williams and Hinton studied that backpropagation can provide "interesting" distribution representations. Yann LeCun combined backpropagation with convolutional neural networks and showed the first practical demonstration to read "handwritten" digits at Bell Labs in 1989. Later many optimistic researchers

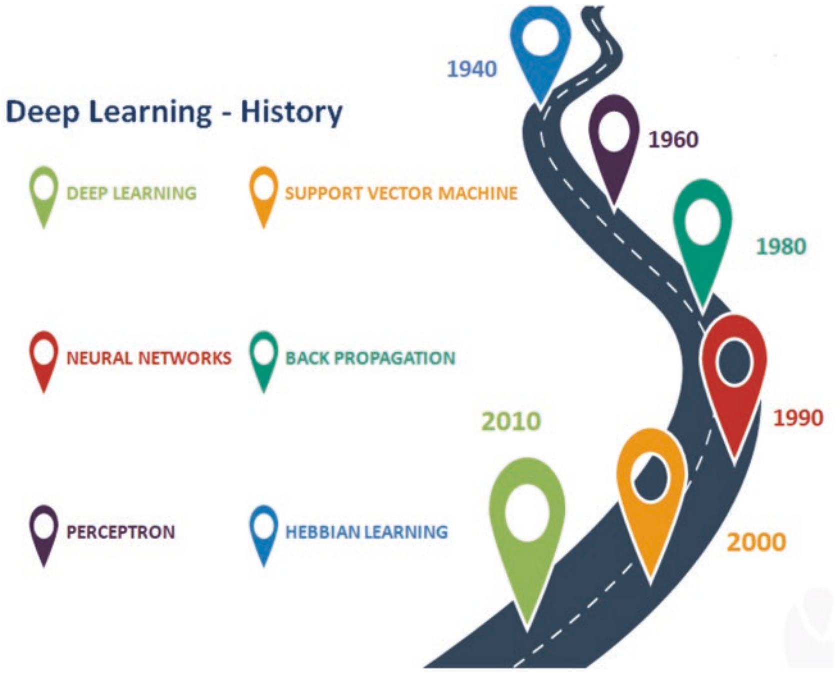

Fig. 3 Roadmap of deep learning history

exaggerated artificial intelligence; notably in 1995, Dana Cortes and Vladimir Vapnik have proposed a model to map and identify similar data, called the support vector machine. In 1997, Sepp Hochreiter and Juergen Schmidhuber have proposed long short-term memory (LSTM) for recurrent neural networks (Fig. 3).

The new era for deep learning began in 1999 as it is the evolution of graphics processing units (GPUs). In 2000, the vanishing gradient problem is identified which paved the way for the development of long short-term memory. Fei-Fei Li an AI expert assembled ImageNet which can process more than 14 million labeled images. During 2011 and 2012, AlexNet a convolutional neural network won many international competitions. In 2012, Google Brain announced a project called The Cat Experiment, which overcomes the limitations of unsupervised learning. At present, the evolution of artificial intelligence and the processing of big data are dependent on deep learning.

4 Feed-Forward Neural Networks

The neurons in the human brain are arranged as layered structure, and even most of the human intelligence part in the brain, the cerebral cortex, is of six layers [4]. The perceived information travels from layer to another layer until obtaining the conceptual understanding from the sensory input.

In Fig. 4, a three-layer perceptron is shown with the hidden layer that contains neurons with nonlinear activation functions. Arbitrarily complex decision and computation of any likelihood function can be easily done by a three-layer perceptron.

From Fig. 4, it is noted that the connection traverses from the low-level layer to the high-level layer and there are no communications among neurons which exist in the same layer as well from the higher to the lower level. Hence these setup is called the feed-forward networks. The middle layer in Fig. 4 is the hidden layer where the magic happens when the neural network tries to solve complex problems. Every layer in Fig. 4 has an equal number of neurons, which is not mandatory. The input and output are represented as vectors. Linear neurons are represented by a linear function in the form of $f_z = a_z + b$. Linear neurons are easy to compute but restricted with limitations. A feed-forward network with only linear neurons contains no hidden layer which enables the users to get vital features from the input layer. In practice, there are three possible types of neurons, namely, sigmoid neuron, tanh neurons, and ReLU neurons, that dumped the nonlinearity concept. The sigmoid neurons use the function

$$f = \frac{1}{1+e^{-z}} \tag{2}$$

The above equation represents that when the value of logit is actually small, then the output is very close to 0, and it is 1 when the value of logistic is very large. Between the values 0 and 1, the neuron takes the shape of S as shown in Fig. 4.

Based on the types of connections the neural network architecture is categorized into "recurrent neural networks" in which there exists a synaptic connection from output to the input whereas in "feed-forward neural networks" there exists a feedback operation from output to inputs. Neural networks are constructed as either single layers or multilayer.

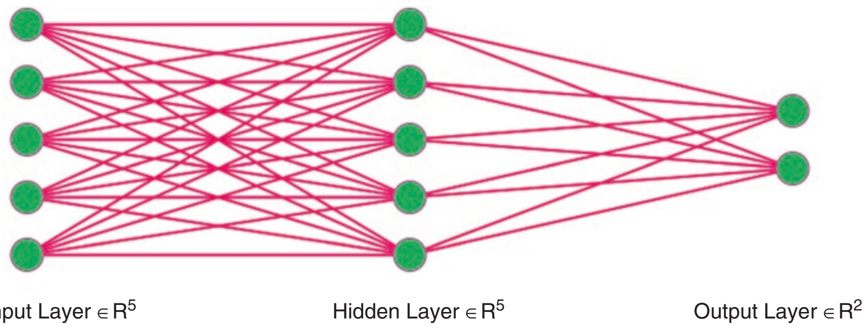

Fig. 4 Three-layer perceptron network with continuous inputs, two output, and two hidden layers

4.1 Backpropagation

Backpropagation is the heart of neural network training which fine-tunes the weights of neural net obtained in the previous epoch. It was developed in 1970, and researchers fully appreciated it after 1986 when David Rumelhart, Geoffrey Hinton, and Ronald Williams published a paper describing that backpropagation works faster and provides solutions for previously unsolved problems. Backpropagation is a kind of supervised learning method for multilayer artificial neural networks (ANNs) with applications ranging from classification, pattern recognition, medical diagnostics, etc. The backpropagation algorithm made the multilayer perceptron networks occupy a place in the neural network's research toolbox. The multilayer perceptron is perceived as a feed-forward network with more than one layer of nodes between the input and output nodes. It updates the synaptic weights by propagating a gradient vector back to the input in which the elements are defined as the derivative of an error measure for a parameter. The error signals are the significant difference between the actual and the desired outputs.

The backpropagation algorithms are considered as a generalized view of the least-mean-square (LMS) algorithm that consists of a forward pass and a backward pass. The backpropagation computes specifically all the partial derivatives $\frac{\partial f}{\partial w_i}$ where w_i is the ith parameter and f is the output.

Consider a multilayer feed-forward neural network as shown in Fig. 2. Let us assume a neuron i is present in the output layer and the error signal for n^{th} iteration is given by the equation

$$ei(m) = di - yi(m) \tag{3}$$

where d_i is the desired output for neuron i and $y\ j\ (m)$ is the actual output for neuron i, computed using the current weights of the network at iteration m.

Equation 2 represents the instant error energy value y for the neuron i as

$$\varepsilon_i(m) = \frac{1}{2} e_i^2(m) \tag{4}$$

The instantaneous value $\varepsilon_i(m)$ is the sum of all $\varepsilon_i(m)$ for all neurons in the output layer as represented in Eq. 3

$$\varepsilon_i(m) = \frac{1}{2} \sum_{i \varepsilon S} e_i^2(m) \tag{5}$$

where S is the set of all neurons present in the output layer. For consideration, suppose a training set contains N patterns, and the average square energy for the network is given by Eq. 4:

$$\varepsilon_{avg} = \frac{1}{N}\sum_{n=1}^{N}\varepsilon(m) \tag{6}$$

The modes of backpropagation algorithms are (a) batch mode and (b) sequential mode. In the batch mode, the weight updates are done after an epoch is completed. In contrast to this, the sequential mode or stochastic mode updates are performed after the presentation of each training example. The following equation gives the output expression for the neuron i

$$y_i(m) = f\left[\sum_{i=0}^{n} w_{ij}(m)\, y_i(m)\right] \tag{7}$$

where n represents the total number of inputs to the neuron i from the previous layer and f is the activation function used in the neuron i.

The updated weight to be applied to the weights of the neuron i is directly proportional to the partial derivative of the instantaneous error energy $\varepsilon(n)$ for the corresponding weight, and it is represented as

$$\frac{\partial \varepsilon(m)}{\partial w_{ij}(m)} \tag{8}$$

Using the chain rule of calculus, it is expressed as

$$\frac{\partial \varepsilon(m)}{\partial w_{ij}(m)} = \frac{\partial \varepsilon(m)}{\partial e_i(m)}\frac{\partial e_i(m)}{\partial y_i(m)}\frac{\partial y_i(m)}{\partial w_{ij}(m)} \tag{9}$$

Equation 10 is obtained from Eqs. (2), (1), and (5)

$$\frac{\partial \varepsilon(m)}{\partial e_i(m)} = e_i(m) \tag{10}$$

$$\frac{\partial e_i(m)}{\partial y_i(m)} = -1 \tag{11}$$

$$\frac{\partial y_i(m)}{\partial w_{ij}(m)} = f'\left[\sum_{i=0}^{m} w_{ij}(m)\, y_i(m)\right]\frac{\partial\left[\sum_{i=0}^{m} w_{ij}(m)\, y_i(m)\right]}{\partial w_{ij}(m)}$$

$$= f'\left[\sum_{i=0}^{m} w_{ij}(m)\, y_i(m)\right] y_i(m) \tag{12}$$

where

$$f'\left[\sum_{i=0}^{m} w_{ij}(m)\,y_i(m)\right] = \frac{\partial f\left[\left[\sum_{i=0}^{m} w_{ij}(m)\,y_i(m)\right]\right]}{\partial\left[\left[\sum_{i=0}^{m} w_{ij}(m)\,y_i(m)\right]\right]}$$

Substituting Eqs. (8), (9), and (10) in Eq. 9, the following expression arrives

$$\frac{\partial \varepsilon(n)}{\partial w_{ij}(m)} = -e_j(m)\,f'\left[\sum_{i=0}^{m} w_{ij}(m)\,y_i(m)\right] y_i(m) \tag{13}$$

Delta rule is used to provide the correction $\Delta w_{ij}(m)$, and it is expressed as

$$\Delta w_{ij}(m) = -\eta\,\frac{\partial \varepsilon(n)}{\partial w_{ij}(m)} \tag{14}$$

where η is a constant pre-determined parameter for the learning rate in the back-propagation algorithm.

5 Types of Deep Learning Networks

The deep learning network is classified into three classes depending upon the techniques and architectures used for a particular application like synthesis, classification, and recognition. They are classified into:

(i) Unsupervised deep learning network
(ii) Supervised deep learning network
(iii) Hybrid deep learning networks

Unsupervised deep learning network captures higher-order correlation data for synthesis purposes when there is no clear target class defined. In supervised learning of deep networks, discriminative power is provided for pattern classification by portraying the distributions of classes accustomed on the data which is visible. It is otherwise known as discriminative deep networks. A hybrid deep neural network exploits both discriminative and generative components. Moreover, a hybrid deep neural network model is structured by converging homogeneous convolution neural network (CNN) classifiers. The CNN classifiers are trained to yield an output as one for the predicted class and zero for all the other classes.

6 Deep Learning Architecture

In this deep learning architecture section, the commonly used deep learning approaches are discussed. Representation is a significant factor in deep learning. In the traditional method, the input features are extracted from raw data to be fed in machine learning algorithms. It relies on domain knowledge and the practitioner's expertise to determine the pattern. Traditional Software Engineering methodology like create, analyze, select, and evaluate are time-consuming and laborious. In contrast, the appropriate features are learned from the data directly without any human intervention and facilitate the discovery of dormant relationship among data that might be otherwise hidden or unknown.

In deep learning, the complex data representation is commonly expressed as compositions of simpler representations. Most of the deep learning algorithms are constructed based on the conceptual framework of artificial neural network (ANN), and it comprises interconnected nodes called as "neurons" which are organized in layers shown in Fig. 5. The neuron which does not exist in these two layers is called hidden units, and it stores the set of weights W.

Artificial neural network weights can be augmented by minimizing the loss function, for instance, negative log-likelihood, and it is denoted in Eq. 1:

$$E(\theta, D) = -\sum_{i=0}^{D}\left[\log P\left(Y = y_i |, x_i |, \theta\right)\right] + \lambda \quad \theta \quad p \tag{15}$$

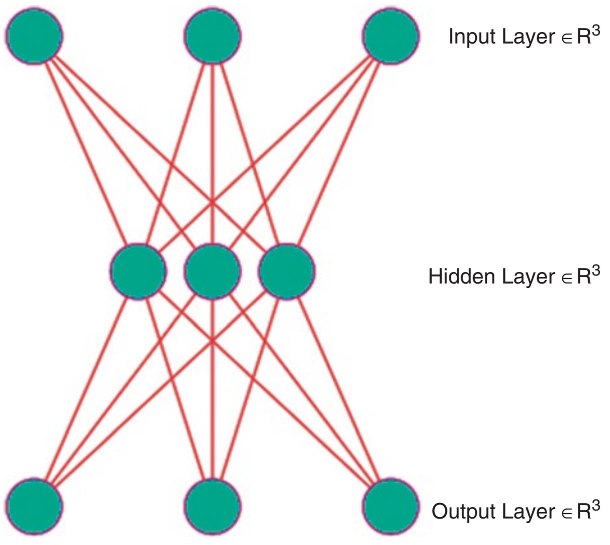

Fig. 5 Neural network with 1, 2, 1 input, hidden, and output layers

The first term minimizes the total log loss in the whole training dataset D.

The second term minimizes the p-norm of learned parameter θ_i, and it is controlled by λ a tunable parameter.

It is referred as regularization, and it prevents a model to be overfitting. Normally, the loss function can be optimized using a backpropagation mechanism, and it is meant for weight optimization that reduces the loss by traversing backward from the final layer in the network. Some of the deep learning open-source tools are Keras3, Theano2, TensorFlow1, Caffe6, DeepLearning4j8, CNTK7, PyTorch5, and Torch4. Some commonly used deep learning models discussed are based on optimization strategy and ANN's architecture. The deep learning algorithms are categorized into supervised and unsupervised techniques. The supervised deep learning architecture includes convolutional neural networks, multilayer perceptrons, and recurrent neural networks. The unsupervised deep learning architecture includes autoencoders and restricted Boltzmann machines (Fig. 6).

6.1 *Supervised Learning*

6.1.1 Multilayer Perceptron (MLP)

Multilayer perceptron holds many hidden layers; the neurons in the base layer *i* is completely connected to neurons in *i* + *1* layer. Such type of network is restricted to have minimal hidden layers, and the data is allowed to transmit in one direction only. A weighted sum is computed for the outputs received from the hidden layer in each hidden unit. Equation 16 represents a nonlinear activation function σ of the computed sum. At this point, d refers to the number of units available in the previous layer, and x_j is referred as the output received from the previous layer j_{th} node. b_{ij} and w_{ij} are considered as bias and weight terms that are associated with each x_{ij}. Tanh or

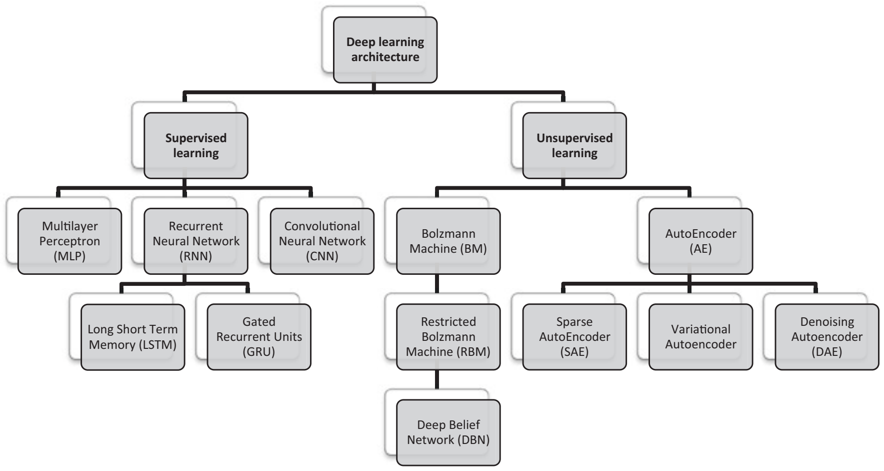
Deep learning architecture
Supervised learning
Unsupervised learning
Multilayer Perceptron (MLP)
Recurrent Neural Network (RNN)
Convolutional Neural Network (CNN)
Bolzmann Machine (BM)
AutoEncoder (AE)
Long Short Term Memory (LSTM)
Gated Recurrent Units (GRU)
Restricted Bolzmann Machine (RBM)
Sparse AutoEncoder (SAE)
Variational Autoencoder
Denoising Autoencoder (DAE)
Deep Belief Network (DBN)

Fig. 6 Deep learning architecture and output layers

sigmoid are taken as the nonlinear activation functions in the conventional network, and rectified linear units (ReLU) [8] are used in modern networks.

A multilayer perceptron comprises of multiple hidden layers where

$$hi = \sigma\left(\sum_{j=1}^{d} x_j w_{ij} + b_{ij}\right) \tag{16}$$

After optimizing hidden layer weights during training, a correlation among the input x and output y is learned. The availability of many hidden layers makes the input data representation in a high-level abstract view because of the hidden layer's nonlinear activations. It is one of the simplest models among other learning architectures which incorporate completely connected neurons in the final layer.

6.1.2 Recurrent Neural Network (RNN)

CNN is an appropriate choice if the input data has a neat spatial structure (e.g., collection of pixels in an image), and RNN is a logical choice if the input data is ordered sequentially (e.g., natural language or time series data). One-dimensional sequence is fed into a CNN; the output of the extracted features will be shallow [8], meaning only closed localized relationships among few neighbors are considered for feature representations. RNNs are capable of handling long-range temporal dependencies. In RNN, hidden state *ht* is updated based on the triggering of current input *xt* at a time *t* and the previously hidden state *ht-1*. Consequently, the final hidden state contains complete information from all of its elements after processing an entire sequence. RNN includes:

1. Long short-term memory (LSTM)
2. Gated recurrent units (GRU)

The symbolic representation of RNN is shown in Fig. 7, with its equivalent extended representation, for instance, three input units, three hidden units, and an output. The input time step is united with the present hidden state that depends on the previous hidden state.

RNN includes LSTM and GRU models, the most popular variants referred to as gated RNN. The conventional RNN consists of interconnected hidden units, whereas

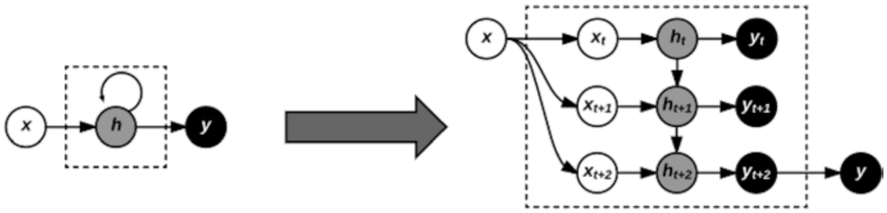

Fig. 7 RNN with extended representation

a gated RNN is substituted by a cell that holds an internal recurrence loop, and significantly the gates in this model control the information flow. The main advantage of gated RNN lies in modelling longer-term sequential dependencies.

6.1.3 Convolutional Neural Network (CNN)

CNN is a famous tool in recent years, particularly in image processing, and are stirred by the organization of the cat's visual cortex [5]. The local connectivity is imposed on the raw data on CNN. For example, more significant features are extracted by perceiving the image as a group of local pixel patches rather considering 50 x 50 image as individual 2500 unrelated pixels. A one-dimensional time series may also be viewed as a set of local signal segments. In particular, the equation for one-dimensional convolution is given as

$$C_{1\mathrm{d}} = \sum_{a=-\infty}^{\infty} \mathrm{x}(\mathrm{a}).\mathrm{w}(\mathrm{t}-\mathrm{a}) \tag{17}$$

where x refers to the input signal and w refers to the weight function or convolution filter.

The equation for two-dimensional convolution is given, where k is a kernel and X is a 2D grid:

$$C_{2d} = \sum_{m}\sum_{n} X(m,n)\,K(i-m,j-n) \tag{18}$$

The feature maps are extracted by calculating the weights of the input in a filter or a kernel. CNN encompasses sparse interactions considered as filters normally smaller than the input that results in less number of parameters. Parameter sharing is correspondingly encouraged in CNN because every filter is functional to the entire input. However, in CNN the same input is received from the previous layer which perfectly learns several lower level features. Subsampling is applied to aggregate the features which are extracted. The CNN architecture consists of two convolutional layers trailed by a pooling layer as depicted in Fig. 8. The application of CNNs is best in computer vision [6, 7].

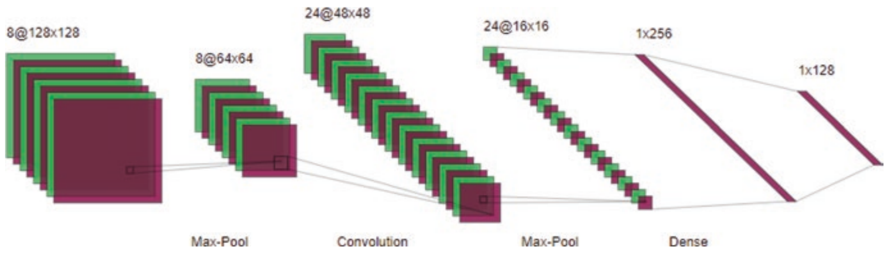

Fig. 8 ConvNet and output layers

6.2 *Unsupervised Learning*

6.2.1 Autoencoder (AE)

Autoencoder (AE) is the deep learning model that exemplifies the concept of unsupervised representation learning. Initially, it has pertained to supervised learning models once the labeled data was limited, but it is still remained to be useful for complete unsupervised learning such as phenotype discovery. In AE, the input is encoded into a lower-dimensional space z, and it is decoded further by reconstructing $\overline{x}$ of the corresponding input x. Hence, the encoding and decoding processes of an encoder are respectively given in equation with a single hidden layer. The encoding and decoding weights are represented as W and W0, and the reconstruction error is minimized. Z is a reliable encoded representation.

$$\mathrm{z} = \sigma\left(Wx + b\right) \tag{19}$$

$$\overline{x} = \sigma\left(W'z + b'\right) \tag{20}$$

As soon as an AE is well trained, then a single input is fed in the network and the innermost hidden layer activated to serve as input for the encoded representation (Fig. 9).

The input data is transformed into a structure where AE stores the utmost significant derived dimensions. It is similar to traditional dimensionality reduction techniques, namely, singular value decomposition (SVD) and principal component analysis (PCA). Deep autoencoder networks can be trained in a greedy manner, which is referred as the stacking process. Some of the autoencoder variants are:

1. Sparse autoencoder (SAE)
2. Variational autoencoder (VAE)
3. Denoising autoencoder (DAE)

6.2.2 Restricted Boltzmann Machine (RBM)

RBM is an unsupervised learning architecture that learns input data representation. It is almost similar to AE; instead, RBMs estimate the probability distribution of the available input data. Hence, it is perceived as a generative model where the data was generated in the underlying process. The canonical RBM is a model that consists of binary visible units $\vec{v}$, and hidden units $\vec{h}$ along with the energy function as shown in the equation:

$$E\left(v,h\right) = -b^T v - c^T v - Wv^T h \tag{21}$$

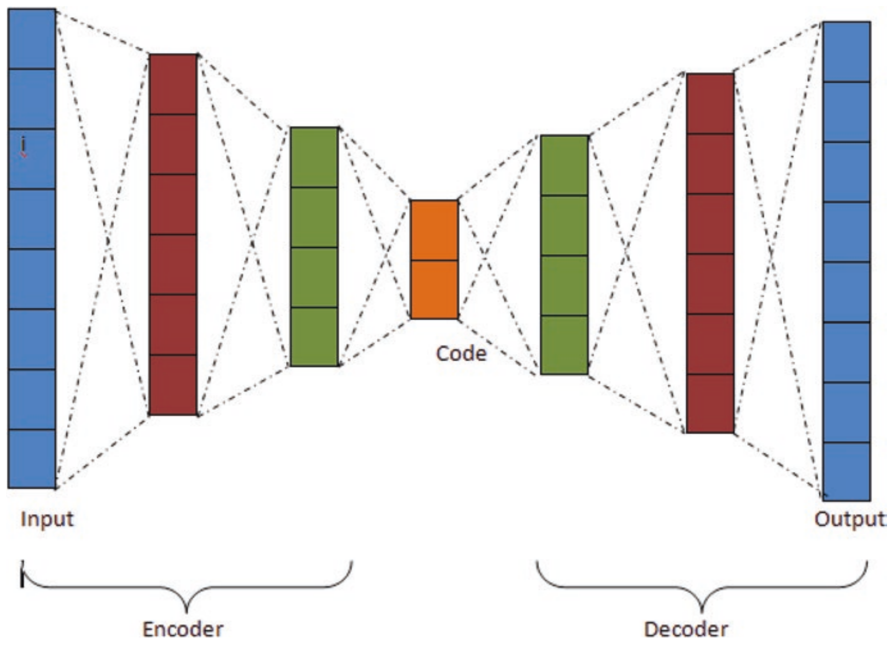

Fig. 9 Autoencoder example and output layers

In Boltzmann machine (BM), every unit is completely connected, while in restricted Boltzmann machine, there is no connection among the hidden units. Restricted Boltzmann machine is typically trained using a stochastic optimization like Gibbs sampling, and it yields the learned representation of the given input data that is viewed as the final form of h. Moreover, RBMs can be stacked hierarchically to construct a deep belief network (DBN) particularly for supervised learning.

7 Platforms for Deep Learning/Deep Learning Frameworks

Many software packages are available for researchers to ease the construction of deep learning architectures, but few years back non-deep learning professionals faced many difficulties to hadle the software packages. This circumstance lasted until Google introduced the DistBelief system in 2012. Following DistBelief, similar software packages like TensorFlow, Microsoft Cognitive Toolkit (previously CNTK), DeepLearning4j, Caffe, Torch, Keras, Neural Designer, H2o.ai, and Deep Learning Kit have extensively spurred the industry (Fig. 10).

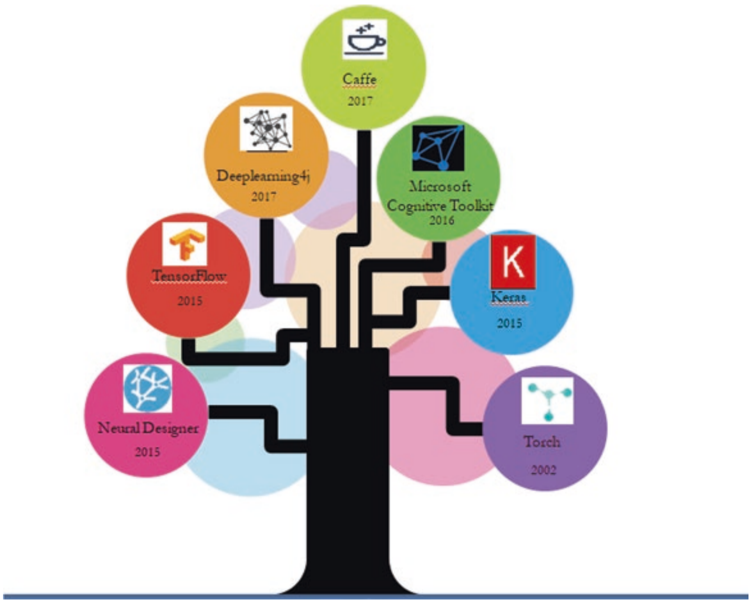

Fig. 10 Deep learning platform and output layers

7.1 TensorFlow

The concept of TensorFlow is highly associated with the mathematics involved in engineering and physics. Later TensorFlow has made its way to computer science which is associated with logic and discrete mathematics. Advanced machine learning concepts utilize the manipulation and calculus of tensors. TensorFlow is an open-source end-to-end machine learning library for production and research. It offers APIs for expert and beginner-level learners to develop applications for the cloud, mobile, web, and desktop. For beginner-level learners, TensorFlow recommends Keras API to develop and train the deep learning models. For advanced operations like forward passes, customizing layers, and training the loops with auto-differentiation, define-by-run interface API is recommended. Pre-made estimators are available to implement common machine learning algorithms. The architecture of TensorFlow is divided into four functioning parts, namely, data processor, model builder, training, and estimating the model. It accepts the inputs as tensors or multi-dimensional array, constructs operation flowchart which explains the multiple operations subjected to input, and finally comes out as output. Hence the name TensorFlow arises as the tensors flow through a list of operations and produce the desired output on the other side. TensorFlow is based on static graph computation, to visualize the constructed neural network with the help of TensorBoard. TensorFlow supports algorithms like classification, linear regression, deep learning wipe, deep learning classification, boosted tree classification, and boosted tree regression.

7.2 Microsoft Cognitive Toolkit

Microsoft Cognitive Toolkit is also a deep learning toolkit that considers neural networks as a sequence of computational procedure using a directed graph. The former version of this toolkit is the Computational Network Toolkit (CNTK). The latest version is CNTK v.2.0 Beta 1, available with new Python and C++ APIs using BrainScript as its own language. The Computational Network Toolkit libraries are developed using the C++ language. The Python APIs preserve abstractions for model definition, data reading, learning algorithms, and disbursed training. CNTK 2 is considered as the supplement of Python API with the feature of protocol buffer serialization developed by Google. CNTK 2 supports Fast R-CNN algorithm that supports the object-detection algorithm. Fast R-CNN algorithm is based on reusability concept that reuses computations from the convolution layers by adding a ROI pooling scheme. Microsoft Cognitive Toolkit is an open-source, multi-GPU machine, highly supportive for neural network training to classify and recognize images, text, and speech. Microsoft Cognitive Toolkit is the backbone for Skype live translation, Xbox, Bing, and Cortana. It supports a range of neural network types like convolutional neural network (CNN), feed-forward network (FFN), recurrent/long short-term memory (RNN/LSTM), sequence-to-sequence with attention, and batch normalization. Microsoft Cognitive Toolkit supports unsupervised learning, reinforcement learning, generative adversarial networks, and automatic hyperparameter tuning. Parallelism can be achieved with the highest accuracy, even for the largest models in the GPU memory.

7.3 Caffe

Convolution Architecture For Feature Extraction (Caffe) is a deep learning framework developed by Berkeley AI Research with speed, expression, and modularity as its features. The speed of Caffe makes it suitable for industry development and research works. It can process nearly 60 M images per day with a single NVIDIA K40 GPU. The extensible code feature promotes active development. Expressive architecture encourages innovation and application. The usage of hard coding is minimized in model development.

7.4 DeepLearning4j

DeepLearning4j is a free and open-source deep learning library based on Java that provides complete solution for deep learning in various applications like deep predictive mining and knowledge discovery on CPUs and GPUs (graphics processing units). DeepLearning4j integrates the algorithms of artificial intelligence (AI) and techniques that are applicable for cyber forensics, business intelligence, robotic process automation (RBA), predictive analysis, network intrusion detection and prevention, face recognition, recommender systems, anomaly detection, regression, and many others. DeepLearning4j can import models from the advanced deep learning frameworks like Keras, Theano, Caffe, and TensorFlow. DeepLearning4j can initiate the interface for both Python and Java programming without any compatibility problem. The features of DeepLearning4j are based on microservice architecture. It provides scalability on Hadoop for big data and supports GPUs for scalability on Amazon Web Services (AWS) cloud. It is based on a distributed architecture with multi-threading; provides parallel computing and training and APIs for Java, Python, and Scala; and supports CPUs and GPUs, and massive amounts of data can be processed using clusters. The libraries and components associated with DeepLearning4j are ND4j, JavaCPP, DataVec, and RL4J. ND4j is the combined application of NumPy and Java virtual machine (JVM). ND4j is a library that provides rapid processing of matrix data, numerical computations, and performance-aware execution of multi-dimensional objects including linear algebra, signal processing, optimization, gradient descent, transformations, etc. JavaCPP is an interface and bridging tool for C++ and Java without ant third-party and intermediate applications. DataVec is a tool for ETL (extract, transform, and load) which facilitates the transformation of raw data to vector format with preprocessing to make it companionable for training in machine learning implementations. It supports binary, videos, images, text, CSV, etc. RL4J is the reinforcement learning for Java platforms with the integration of Deep Q-Learning, Asynchronous Actor-Critic Agents (A3C), etc.

7.5 Keras

Keras is an open-source neural network library developed by François Chollet, with features like fast, modular, and user friendly in Python platform that works on top of TensorFlow or Theano. Backend is a library within Keras to handle low-level computation as Keras is dedicated to advanced API wrapper. Backend performs the low-level computations like convolutions, tensor products with the aid of Theano or TensorFlow. It is capable of running on top of Theano, CNTK, or TensorFlow. Keras high-level API compiles the model which is designed with optimizer and loss functions and handles training process with fit function. Keras can support many platforms and devices, and it can be deployed in Web browser with .js support, Raspberry Pi, iOS with CoreML, Android with TensorFlow Android, and Cloud engine. Keras supports parallel data processing, and hence it handles huge volume of data and speeds up the training process.

7.6 Neural Designer

Neural Designer is a deep learning tool which is used to implement analytics algorithms and make it easy to handle. It is designed with a graphical user interface that defines the flow of work and gives accurate result. It is easy to handle as there is no programming or block diagrams involved. The user interface helps the user and instructs the procedure in a well-defined manner. Neural viewer is a visualization tool which displays the correct results in the form of exportable charts, tables, and pictures. Neural Designer has advanced algorithms that allow the users to construct incomparable predictive models. With the help of complicated data preprocessing methodology, the calculation of principal components and cleaning of outliers are made easy. In Neural Designer, the user can construct the most powerful predictive models with the help of various error and regularization methods. In addition to this, powerful strategies like Levenberg-Marquardt and quasi-Newton method are included to produce more precise computations. It also contains few strategies for testing the generalization capabilities of a predictive model. It holds higher processing speed and better memory management with CPU parallelization characteristics by means of GPU acceleration with CUDA and OpenMP.

7.7 Torch

Torch (Lua) is a Matlab-like environment for deep and not-so-deep machine learning solutions with flexible tensor implementation facility. Some of the features automatically compute gradients and hold multi-backend tensor for faster CPU/GPU computation, and rapid prototyping is possible as it supports high-level language.

8 Deep Learning Application

8.1 Speech Recognition

Speech recognition utilizes deep learning concepts and becomes the foremost application of deep learning by cautiously using its power. In 2010, deep learning makes its footprint in the speech recognition application. Gaussian mixture model (GMM) is the traditional speech recognition system which is based on hidden Markov models (HMMs). Here the speech signal is considered as short-time stationary signal or piecewise stationary signal, and hence Markov model is apt for this application. The limitation of this method is that it is inefficient for modelling nonlinear functions [20]. In contrast to HMMs, neural networks prove its efficiency in discriminative training. Neural networks provide better results for short-time signals; when it is continuous speech signals, the efficiency is questionable because it is not able to model temporal dependencies for continuous signals. In 2012, Microsoft announced a deep learning-based novel version of their Microsoft Audio Video Indexing Service (MAVIS). The result published by Microsoft clearly showed that deep learning-based application reduces word error rate (WER) when compared to Gaussian mixtures [21].

8.2 Deep Learning in HealthCare

Healthcare system is facing a new era by utilizing advanced technologies and providing right treatment for right patient at right time. Deep learning is the most powerful tool that allows a machine to learn from huge volume of data to make decisions. Researchers found that processing pharmaceutical information with multilayer neural networks produces accurate predictive decisions in various clinical applications. Deep learning architecture is based on hierarchical learning structure, and it has the capacity to integrate heterogeneous data to produce greater generalization. Many studies proved that deep learning paved its way toward the next-generation healthcare system that can accommodate billions of patient records to predict diseases, personalized prescriptions, clinical trials, and treatment recommendations. Using the temporal deep learning approach, Wang et al. have won the Parkinson's Progression Markers Initiative data challenge in identifying the subtypes of Parkinson's disease [9]. The traditional matrix- or vector-based approach is not considered as the best method as Parkinson's disease is vastly progressive and it is difficult to identify the disease progression patterns. Moreover, Wang et al. identified another three Parkinson's disease subtypes using LSTM RNN model, which demonstrates the dominance and potential of deep learning models in healthcare issues.

The deep learning architecture applicable for healthcare system mostly falls on recurrent neural networks (RNNs) [10], convolutional neural networks (CNNs) [11], autoencoders (AEs) [12], and restricted Boltzmann machines (RBMs) [13]. The major application of deep learning in healthcare system falls on image processing, especially to predict Alzheimer's disease using magnetic resonance imaging (MRI) scans [14, 15]. The risk of osteoarthritis can be detected using CNNs that represent low-field knee MRI hierarchically to automatically segment cartilage. Deep learning is used to segment multiple sclerosis lesions in multi-channel 3D MRI in order to predict malignant and benign breast nodules. Deep learning is being used to process electronic health record (EHR) data in the field of laboratory test, diagnosis, and medication and for free-text clinical notes. Area under the receiver operating characteristic curve and F-score methodology prove that deep learning accuracy is superior than traditional machine learning process [16]. DeepCare, a deep dynamic network, is a RNN with long short-term memory (LSTM) hidden units, pooling, and word embedding which identifies the current state of illness and predicts the future consequences [17]. Choi et al. [18] have developed Doctor AI model that predicts diagnoses and medications using RNNs with gated recurrent unit (GRU). Miotto et al. [19] used a three-layer stacked denoising autoencoder (SDA) and proposed a deep patient representation from the EHRs to predict risks based on random forest classifiers.

8.3 Deep Learning in Natural Language Processing

Natural language processing (NLP) is a computational methodology for automatic analysis of human language. Research activities in document text and language are increasing nowadays in the signal processing community. Deep learning contribution is in language modelling to give a probability to sequence of linguistic symbol or word. Natural language processing (NLP) started in the era of batch processing and punch card in that the analysis takes up to 7 minutes [22]. Deep learning algorithms and architectures are doing impressive advancements in the fields of pattern recognition. Collobert et al. [23] explained that a deep learning [24] framework can perform well in many natural language processing tasks such as semantic role labeling (SRL), named-entity recognition (NER), and POS tagging [25]. Following Collobert et al., various versions of advanced deep learning-based algorithms have evolved to resolve various difficult natural language processing tasks [26–28].

9 Conclusion

Deep learning is a branch of machine learning algorithm that is used to model data with the help of architectures consisting of multiple nonlinear transformations. This chapter explained the basis for deep learning, and its evolution is discussed with a roadmap. The architectures, learning methods, and its applications are elaborated. The platforms and their brands are discussed deeply with their applications. The general applications and the life-saving applications of deep learning in healthcare system are clearly explained. The future of deep learning is very bright, and the great thing about deep learning is that it is doing extremely well at dealing with vast amount of disparate data that are relevant in current era of smart sensors that collect huge amount of information.

References

1. Kuhn, Deanna, et al. Handbook of Child Psychology. Vol. 2, Cognition, Perception, and Language. Wiley, 1998
2. Restak, Richard M. and David Grubin. The Secret Life of the Brain. Joseph Henry Press, 2001
3. McCulloch, W.S., Pitts, W.: A logical calculus of the ideas immanent in nervous activity. Bull. Math. Biophys. **5**(4), 115–133 (1943)
4. Mountcastle, V.B.: Modality and topographic properties of single neurons of cat's somatic sensory cortex. J. Neurophysiol. **20**(4), 408–434 (1957)
5. Hubel, D.H., Wiesel, T.N.: Receptive fields and functional architecture of monkey striate cortex. J. Physiol. **195**, 215–243 (1968)
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. Adv Neural Inf Process Syst, 1097–1105 (2012)

7. Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1–9
8. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
9. The Michael J. Fox Foundation for Parkinson's Research: subtyping Parkinson's disease with deep learning models. https://www.michaeljfox.org/foundation/grant-detail.php
10. Williams, R.J., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Comput. **1**, 270–280 (1989)
11. Lecun, Y., Bottou, L., Bengio, Y., et al.: Gradient-based learning applied to document recognition. Proc. IEEE. **86**, 2278–2324 (1998)
12. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science. **313**, 504–507 (2006)
13. Smolensky P. Information processing in dynamical systems: Foundations of harmony theory (No. CU-CS-321-86). Colorado University at Boulder Dept of Computer Science 1986
14. Liu S, Liu S, Cai W, et al. Early diagnosis of Alzheimer's disease with deep learning. In: International Symposium on Biomedical Imaging, Beijing, China 2014, 1015–1018
15. Brosch, T., Tam, R.: Manifold learning of brain MRIs by deep learning. Med Image Comput Comput Assist Interv. **16**, 633–640 (2013)
16. Manning, C.D., Raghavan, P., Schutze, H.: Introduction to Information Retrieval, vol. 3. Cambridge university press, Cambridge (2008)
17. Pham T, Tran T, Phung D, et al. Deep Care: a deep dynamic memory model for predictive medicine. arXiv 2016. https://arxiv.org/abs/1602.00357
18. Choi E, Bahadori MT, Schuetz A, et al. Doctor AI: predicting clinical events via recurrent neural networks. arXiv 2015. http://arxiv.org/abs/1511.05942v11
19. Miotto, R., Li, L., Kidd, B.A., et al.: Deep patient: an unsupervised representation to predict the future of patients from the electronic health records. Sci. Rep. **6**, 26094 (2016)
20. Singh, H., Bathla, A.K.: A survey on speech recognition. Int. J. Adv. Res. Comput. Eng. Technol. **2**(6), 2186–2189 (2013)
21. Xie, Y., Le, L., Zhou, Y., Raghavan, V.V.: Deep learning for natural language processing. In: Handbook of statistics. Elsevier, Amsterdam, The Netherlands (2018)
22. Cambria, E., White, B.: Jumping NLP curves: a review of natural language processing research. IEEE Comput. Intell. Mag. **9**(2), 48–57 (2014)
23. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J Machine Learn Res. **12**(Aug), 2493–2537 (2011)
24. Prakash, K. B., Ruwali, A., Kanagachidambaresan, G. R.: Introduction, in to tensor flow, programming with tensor flow, EIA/Springer innovations in communication and computing. https://doi.org/10.1007/978-3-030-57077-4_1
25. JHA, A.K., Ruwali, A., Prakash, K.B., Kanagachidambaresan, G.R.: Tensor Flow Basics, programming with tensor flow, EIA/Springer innovations in communication and computing. https://doi.org/10.1007/978-3-030-57077-4_2
26. Kanagachidambaresan, G.R., Manohar Vinoothna, G., Prakash, K.B.: Visualizations, programming with tensor flow, EIA/Springer innovations in communication and computing. https://doi.org/10.1007/978-3-030-57007-4_3
27. Prakash, K.B., Ruwali, A., Kanagachidambaresan, G.R.: Regression, programming with tensor flow, EIA/Springer innovations in communication and computing. https://doi.org/10.1007/978-3-030-57007-4_4
28. Vadla, P.K., Ruwali, A., Lakshmi, M.V.P., Kanagachidambaresan, G.R.: Neural network, programming with tensor flow, EIA/Springer innovations in communication and computing. https://doi.org/10.1007/978-3-030-57007-4_5

Ms. Indrakumari is working as an Assistant Professor in School of Computing Science and Engineering, Galgotias University, NCR Delhi, India. She has completed M.Tech in Computer and Information Technology from Manonmaniam Sundaranar University, Tirunelveli. Her main thrust areas are big data, Internet of Things, data mining, and data warehousing and its visualization tools like Tableau, QlikView.

Dr. T. Poongodi is working as an Associate Professor in School of Computing Science and Engineering, Galgotias University, NCR Delhi, India. She has completed Ph.D in Information Technology (Information and Communication Engineering) from Anna University, Tamil Nadu, India. Her main thrust research areas are big data, Internet of Things, ad hoc networks, network security, and cloud computing. She is a pioneer researcher in the areas of big data, wireless network, and Internet of Things and has published more than 25 papers in various international journals. She has presented a paper in national/ international conferences; published book chapters in CRC Press, IGI Global, and Springer; and edited books.

Ms. Kiran Singh is presently working as an Assistant Professor in the Department of Computer Science and Engineering at Galgotias University. She received her MCA degree from Maharshi Dayanand University in 2008 and M.Tech in Computer Science and Engineering from Rajiv Gandhi Proudyogiki Vishwavidyalaya in 2015, Bhopal. She has overall experience of 11 years. Her research interests include image processing, big data, and IOT. She has published papers in international journal and conference.