Chapter 12
Deep Learning and Its Environmental Applications

Ahmed R. Nasser and Ali M. Mahmood

Abstract The rapid development of technologies brings a new set of challenges and difficulties in scientific studies and research in different fields. These challenges arose due to the considerable increment in the amount of data produced and the complexity of mathematical problems. Consequently, machine learning (ML) approaches, particularly deep learning (DL), receive enormous attention from academia recently. DL is a subfield of ML, which is intended to solve high complexity problems and address complex mathematical models accurately. This chapter explores DL algorithms from viewpoints including mathematical theories description of DL algorithms, the advantages of DL with its challenges, the present state of the art of DL applications, and future fields of knowledge, particularly solving complex problems of environmental systems. The focus is on three key issues including earthquake prediction, weather forecasting, and environment protection and sustainability. The reader of this chapter can gain comprehensive knowledge regarding DL with potential research issues and challenges to be solved in environmental systems.

Key words Machine learning · Deep learning · Environmental systems applications · Sustainability

12.1 Introduction

We introduce artificial intelligence and embellish its main subfields of machine learning and deep learning.

A. R. Nasser (✉) · A. M. Mahmood
Control and Systems Engineering Department, University of Technology-Iraq, Baghdad, Iraq
e-mail: ahmed.r.nasser@uotechnology.edu.iq; Ali.M.Mahmood@uotechnology.edu.iq

J. N. Furze et al. (eds.), *Earth Systems Protection and Sustainability*,
https://doi.org/10.1007/978-3-030-98584-4_12

12.1.1 Artificial Intelligence

Artificial intelligence (AI) can be defined as the science of creating intelligent machines (McCarthy et al. 1959). AI is a general name for technologies that make any guess or decision process. Contrary to the general view, AI can be an algorithm that works with or without a deep learning process. Up until the emergence of machine learning algorithms, AI studies were based on a structure that was described as "hard-coded," where all logical and mathematical operations were coded by the developer. The first AI chess game was simply an artificial intelligence algorithm, solely learning without the help of human knowledge via solving complex, multistep problems. This type of AI is called symbolic artificial intelligence (Hernández-Orallo 2017). Studies in the science of AI date back to the 1950s. Later, AI involved different subfields such as machine learning and deep learning as shown in Fig. 12.1, which illustrates the correlation of AI and its subdivisions (Goodfellow et al. 2016).

12.1.2 Machine Learning

Machine learning (ML) has been defined as a "field of study that gives computers the ability to learn without being explicitly programmed" (Samuel 1988). ML is a form of AI that allows a system to learn from data. ML employs data-learning algorithms to identify data and repetitively predict data-related results. Algorithms receive training data, which makes it possible to produce models with more accurate results.

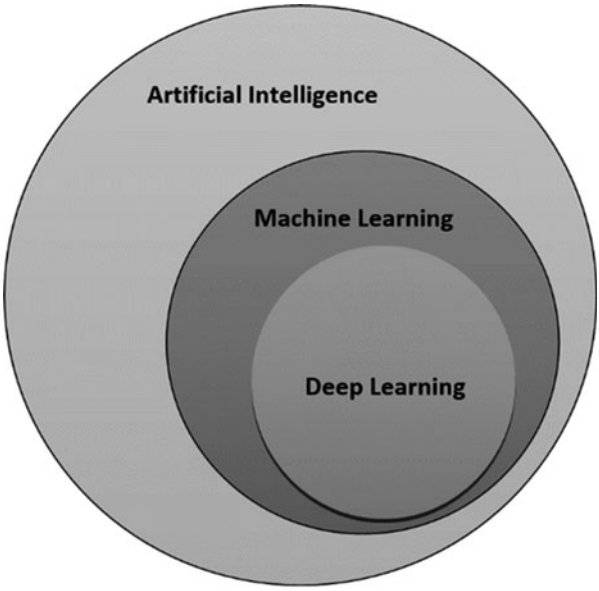

Fig. 12.1 Overview of artificial intelligence and its main subdivisions

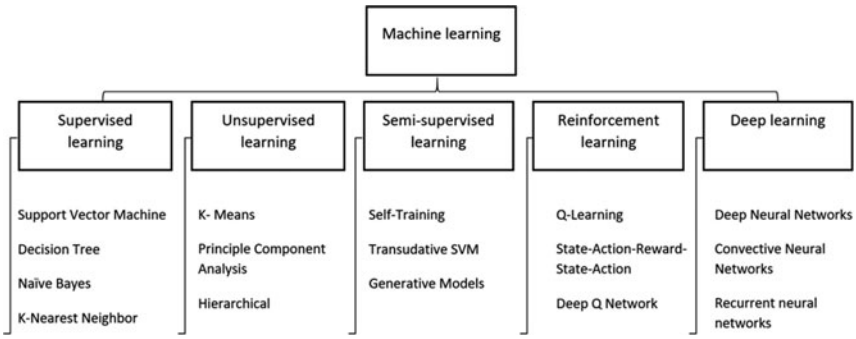

Fig. 12.2 The taxonomy of machine learning approaches

Once a system has been trained, it is expected to produce meaningful output when an input is given to the trained model (Franklin 2005). A prediction algorithm is created and once trained, providing the model with data returns a prediction based on educational data. The two main tasks for machine learning algorithms are prediction and classification (Ahmed and Hayri 2018; James et al. 2013). Prediction is used in the case of quantitative system output in models learned from data. In cases where the input data is qualitative, the methods used to determine or classify which class each data instance belongs to.

The intensive study of machine learning has led to many strategies and algorithms being proposed. Learning strategies used in ML are supervised, unsupervised, semi-supervised, reinforced, and deep learning (James et al. 2013). The taxonomy of machine learning approaches is shown in Fig. 12.2 (Dey 2016). Models created by supervised learning aim to train the relationship between target values through a group of input values and to produce the outputs closest to the target values. The best model obtained will give the closest output for the new input values. In unsupervised learning, the relationship between input values without target values is revealed. With the help of these relationships, close values are grouped, as in clustering algorithms. A new entry will belong to a set, depending on whichever one of the sets it links. Semi-supervised learning algorithm premise between the former two concepts, using a larger part of unlabeled data compared to the labeled data. In the reinforced learning algorithm, instead of a consultant, a criterion evaluates the obtained output as good or bad versus the given input and is used to give the target output. Deep learning is a more powerful method to solve data analytics and learning problems found in large datasets and will be discussed further in subsequent sections.

12.1.3 Deep Learning

Deep learning (DL) is a component of the machine learning family that is capable of performing feature extraction, classification, and transformation operations using large amounts of data; its structure and capabilities mimic the human brain to solve complex problems. Each DL algorithm can be called a machine learning algorithm as it performs learning from data. In contrast, not every machine learning algorithm is a DL algorithm, since DL is a specific kind of machine learning. The structure of DL methods is based on artificial neural networks (ANN) calculation with an additional deep network structure. The ability of DL to solve complex problems derives from the ability of ANN to solve nonlinear problems. In deep learning, increasing numbers of layers represent the depth of the model that provides a basis for solving complex problems (Patterson and Gibson 2017).

Historically, the elementary learning algorithm of the supervised multilayer deep-feed perceptron was introduced by Ivakhnenko and Lapa (1966). In this study, the best features of each layer were selected by statistical methods and transmitted to the next layer, where the applied learning algorithm was the least-squares method. The first DL architecture was introduced by Fukushima in 1979 as depicted in Fig. 12.3. A self-organizing network was developed with "teacherless learning" in the structure that is inspired from visual nervous systems. Fukushima's nets included multiple interconnected layers similar to modern neural nets (Fukushima 1980). The lack of learning in deep architectures is manifested in the backpropagation of errors in multiple layers. Although backpropagation algorithms have been proposed in previous years, the first successful deep neural network application was developed by LeCun et al. (1989). However, it was found that this method is unsuitable for critical time applications since training lasted approximately 3 days. Later authors

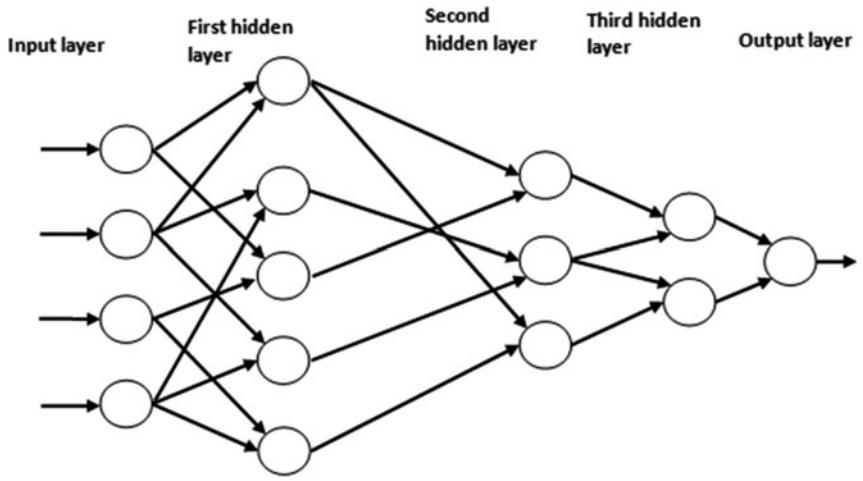

Fig. 12.3 The first deep network architecture

(LeCun et al. 1990) applied coiled backpropagation to classify handwriting numbers. In 1995, the wake-sleep algorithm was developed (Hinton et al. 1995). In this method, network training with six interconnected layers and hundreds of hidden layers was formed, with training over 2 days. In 1997, recurrent neural networks such as long short-term memory was proposed, which made important improvements (Hochreiter and Schmidhuber 1997).

Due to the cost of computation in the traditional ANN, from the 1990s to the 2000s, the application of vector machines has become a popular choice. Consequently, major advances were made owing to the faster processing of computers and the emergence of graphics processing units (GPUs). Computational speed has increased 1000-fold over 10 years. The neural network has begun to replace support vector machines (Schmidhuber 2015). GPUs have been used to increase training speed and have proven efficacy with a normalized method called "dropout" (Hinton et al. 2012) that was prepared later to reduce overfitting in fully connected layers (Krizhevsky et al. 2012). Large companies such as Google, Facebook, and Microsoft have realized this trend and utilize deep learning (Şeker et al. 2017).

12.1.4 Differences Between Machine Learning and Deep Learning

The presence of automatic feature extraction signifies the main difference between traditional machine learning and DL. To extract features, DL algorithms need a big dataset. The traditional machine learning model specifies the characteristics or (features) of each class before training and classifying (Fig. 12.4), while in deep learning these features are automatically extracted and learned (Vieira and Ribeiro 2018). Deep learning does not require an extra feature extraction stage because the network learns to extract features while training. In the multilayered perceptron structure, less than two hidden layers can be used, whereas a deep learning neural network makes use of many hidden layers. Although both networks have error-based

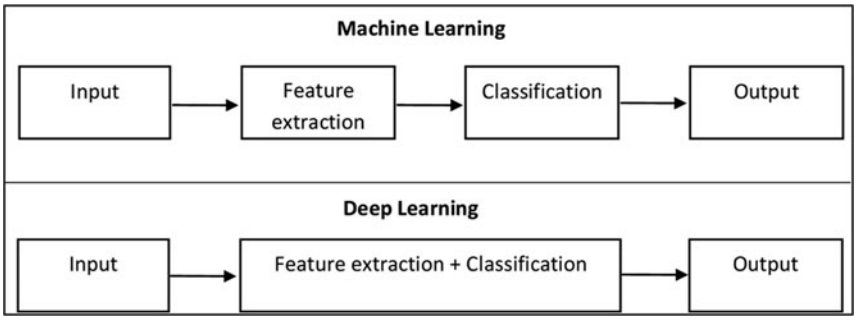

Fig. 12.4 Machine learning versus deep learning

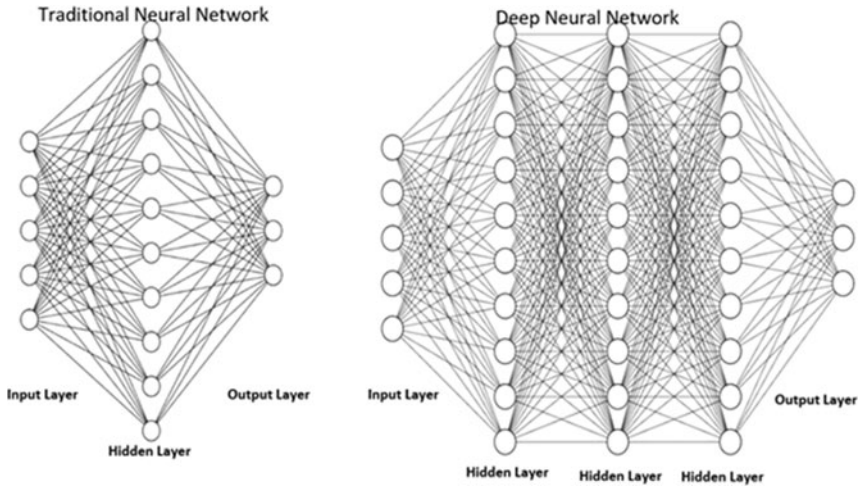

Fig. 12.5 Comparison between a simple neural network and deep learning neural network

Table 12.1 Comparing machine learning and deep learning

Comparison parameter	Machine learning	Deep learning
Data dependency	Excellent performance in small/medium dataset	Excellent performance in a large dataset
Hardware requirements	It can work on a relatively low-end machine	It requires a powerful machine, preferably GPU: it needs hardware capable of performing multiple matrix multiplications
Feature extraction	They need to know the characteristics that data represent	It does not need to know the best feature representing data
Training time	It may take from a few minutes to hours	It may take up to weeks. This is because the artificial neural network needs to calculate a significant amount of weight

learning, supervised learning is done in the multilayered perceptron structure, while DL performs potentially unrestrained learning (Goodfellow et al. 2016).

The main feature that distinguishes deep learning network structure from a simple artificial neural network is that it has multiple hidden layers and a more complex network structure as can be seen in Fig. 12.5.

In Table 12.1, machine learning and deep learning are compared in terms of data dependence, hardware dependencies, feature extraction, and training time. For deep learning algorithms to perform well compared to standard machine learning, data and powerful hardware may be required (Kin 2019).

12.2 Deep Learning Processes

The main process of DL involves the stages of definition, selection, and testing. Definition of the problem and determination of the suitability of the solution with deep learning leads to the definition of related datasets and being ready for analysis. Consequently, the appropriate deep learning architecture is chosen, followed by training of the system using the dataset and selected deep learning architecture. These steps enable the final process of testing the performance of the trained system with test data that was not used for the training.

12.3 Theories of Deep Learning Algorithms

Deep learning consists of three main algorithms: deep neural network (DNN), convolutional neural networks, and recurrent neural networks. The theories of each deep learning algorithm are illustrated in subsections.

12.3.1 Deep Neural Network

Artificial neural networks are an artificial intelligence technique developed by modelling the processing structure of the human brain. Biological nerve cell neurons form the nervous system. ANNs are also called neurons. The neural network architecture includes the number of neurons, the number of layers, and the types of interlayer connections. The human brain consists of a complex network connection of neurons. These neurons serve to transmit information from the brain to the body. According to scientific studies, the human brain may have an average of 100,000,000,000 neurons (Gurney 2014). Each of these neurons is connected to others, with an average of 6000 connections each. These networks of connections are responsible for our perception and learning of everything around us. Artificial neural network neurons consist of input, weights of inputs, bias, activation function, and output. Figure 12.6 illustrates the artificial neural network neuron and its connections.

By adding bias b value to the sum of the product of a neuron's output-input values $(x_1, x_2, \ldots, x_n)$ and their weights, $(w_1, w_2, \ldots, w_n)$ are obtained by passing $(z = \sum_{1}^{n} x_i * w_i + b)$ through the activation function α.

DNNs are a type of multilayer perceptron (MLP) artificial neural network with a large number of hidden layers (Patterson and Gibson 2017). Multilayer artificial neural networks are an example of DNN.

12.3.2 Multilayer Artificial Neural Networks

The human brain has a layered structure. Data from the senses is transferred from one layer to another and transformed into information. This data is processed by transferring from the bottom layer of the visual cortex to the top layer. The last layer determines which object is visible in the image. This layered structure in the human nervous system is modelled to create an artificial neural network, called a multilayer artificial neural network. The first layer on the left is called the input layer, and the neurons in the input layer are called input neurons. The middle layer is called the hidden layer, and the right layer is called the output layer (Schmidhuber 2015). Multilayer structure is shown in Fig. 12.7

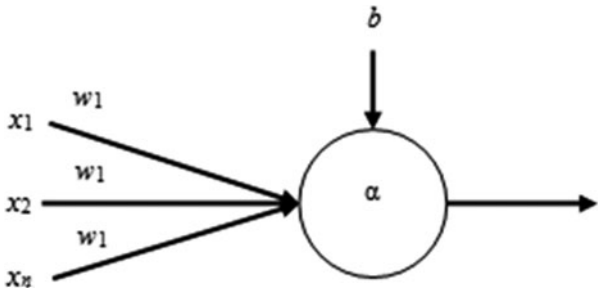

Fig. 12.6 Artificial neural network from the 1990s to the 2000s

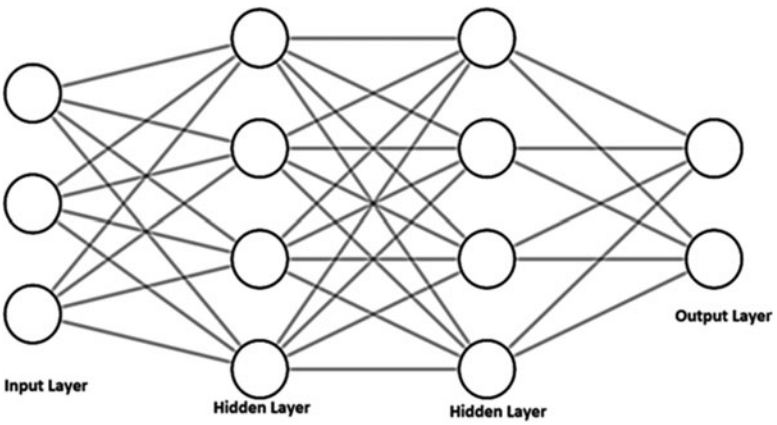

Fig. 12.7 Multilayer artificial neural network

12.4 Convolutional Neural Networks

In the last decade, convolutional neural networks (CNN) have shown significant development in research and applications. Convoluted neural networks can work with large amounts of data produced today. In general, CNNs show very high accuracy images and video classification applications. The success of CNN is seen in self-driving vehicles and online services offered by large companies such as Tesla or Google. The high level of data, new-generation CNN algorithms, and high-performance GPUs led to create a new trend in the industrial revolution (Hu et al. 2015).

Traditional CNN architecture includes five main layers, the input layer followed by the convolution layer, then the pooling layer, the fully connected layer, and finally the output layer. Researchers are working with CNN architectures such as "AlexNet," "ResNet," "Inception," and "VGG," which are created with different layouts of these five layers. In this section, the three main layers of CNN architecture will be examined in more detail. An example of CNN structure is shown in Fig. 12.8 (Hu et al. 2015).

12.4.1 Convolution Layer

The concept of convolution was first introduced by LeCun et al. (1990). Convolution is a customized linear process. These networks are simply networks that perform convolution instead of matrix multiplication in one layer (Goodfellow et al. 2016). The discrete-time convolution of CNNs is expressed by Eq. (12.1) (Goodfellow et al. 2016):

$$s_t = (x * w)_t = \sum_{i=-\infty}^{\infty} x_i * w_{t-i} \tag{12.1}$$

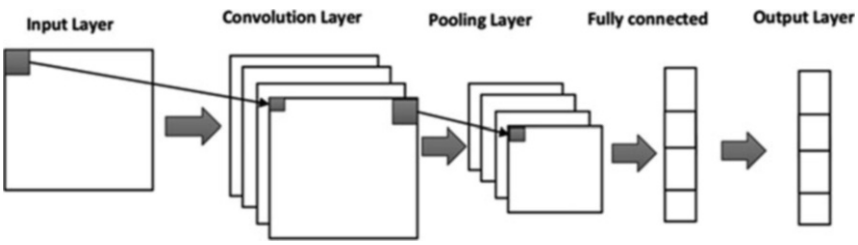

Fig. 12.8 Example of convoluted neural network structure

In Eq. (12.1), filter (kernel) w, input x, time t, and s are expressed as a result. When a two-dimensional input such as a picture is used as the input convolution process, Eq. (12.2) is expressed by Goodfellow et al. (2016):

$$s_{i,j} = (I * K)_{i,j} = \sum_m \sum_n I_{i,j} K_{i-m,j-n} \tag{12.2}$$

In Eq. (12.2), the terms i and j refer to the positions of the new matrix to be obtained as a result of the convolution process. In many cases, the center of the filter is positioned at the origin (Goodfellow et al. 2016).

12.4.2 Pooling Layer

In pooling operations, the output size is reduced by using functions, such as average or maximum value, to summarize the subregions. The pooling process subtracts a value by averaging or calculating the maximum of values within a specified area. The pooling process is implemented by the sliding window method. The sliding window creates value from the corresponding input field according to the pooling method determined each time and adds it to the output layer (Dumoulin and Visin 2016). Through pooling layers in a convolutional neural network, small portions of the aforementioned input are reduced to a single constant value according to the preferred method. Accordingly, pooling layer calculations are less costly than convolution layer calculations.

The output size (o) of the pooling layer, the relationship between the input size (i), the window (part) size (k), and the number of steps (s) on which operations will be performed are shown in Eq. (12.3) (Dumoulin and Visin 2016).

$$o = \left(\frac{i-k}{s}\right) + 1 \tag{12.3}$$

The relationship in Eq. (12.3) applies to all types of docking (Dumoulin and Visin 2016).

12.4.3 Fully Connected Layer

This layer works as an artificial neural network. The values generated as a result of convolution and pooling operations are processed by this layer as input, and the result is generated by the number of classes in the output layer.

12.4.4 Network Hyperparameters

To achieve the best results for convolutional networks as well as basic neural networks, hyperparameters should be ideally adjusted. Hyperparameter details include depth, number, and input size.

The depth is the number of filters that are applied; each filter looks for different attributes in the input. For example, when the convolution layer receives an image as input, different neurons in each filter will be active due to the corner or different colors present in that input. All neurons that look at a certain part of the input and look for different characteristics are called depth columns (Goodfellow et al. 2016).

The number of steps is the hyperparameter that specifies how many pixels the filter will shift over the input.

Padding is the process of filling the input with zeros. The reason for padding is to control the size of the output after convolution. The effect of the fill on the output size is determined using Eq. (12.4) for the input size (i), filter size (k), number of strides (unit strides), and any fill (p) value (Dumoulin and Visin 2016; Goodfellow et al. 2016).

$$o = (i - k) + 2p + 1 \tag{12.4}$$

Given the input size and output size are the same as the result of convolution, the fill value should be 1. This is called half padding and can be expressed mathematically. For any value i and ($k = 2n + 1$, $n \in N$), the result is calculated with ($o = i$), where the number of strides (s) $= 1$ and the padding p $= [k/2]$ (Dumoulin and Visin 2016; Goodfellow et al. 2016).

Although convolution generally decreases the size of the output relative to the input, it is sometimes necessary to reverse it. The appropriate fill value should be preferred to increase the output size. For any value i and k, the number of strides is 1, and the padding is calculated with Eq. (12.5), where p $= k - 1$ (Dumoulin and Visin 2016; Goodfellow et al. 2016).

$$o = i + (k + 1) \tag{12.5}$$

Generally, the result for any input size (i), filter size (k), padding value (p), and some strides (s) is calculated by Eq. (12.6) (Dumoulin and Visin 2016; Goodfellow et al. 2016):

$$o = \left[\frac{i + 2p - k}{s}\right] + 1 \tag{12.6}$$

In artificial neural networks, the pooling layer ensures the output does not change against minor deflection of the input. The most common type is maximum pooling, which breaks up the input and takes only the maximum value of each segment. Since the docking process is the continuous application of the input to different parts like

the convolution process, the expression defined in Eq. (12.6) can be used. Since the fill value is zero in the pooling process ($p = 0$), the result is calculated by Eq. (12.7) (Dumoulin and Visin 2016; Goodfellow et al. 2016):

$$o = \left[\frac{i-k}{s}\right] + 1 \tag{12.7}$$

Equation (12.7) can be used in any kind of pooling operation that applies to any i, k, and s value (Dumoulin and Visin 2016; Goodfellow et al. 2016).

12.5 Recurrent Neural Networks

Recurrent neural networks (RNN) represent a class of artificial neural networks with connections among units with a directed loop (Goodfellow et al. 2016). RNN is a feedforward type of neural network reinforced through the addition of edges extending along with adjacent time steps, which gives the concept of time to the normal neural network model. Similar to feedforward networks, RNN may not have loops between conventional edges. However, the repetitive edge connecting adjacent time steps can form loops, including length loops that are self-connecting from an edge. At time t, repetitive edges are retrieved from the current data point x_t and the hidden node values $h_{(t\ -\ 1)}$ from the previous state of the network (LeCun and Bengio 1995).

The output y_t is calculated by the values of the hidden node h_t at time t. The input $x_{(t\ -\ 1)}$ at time $t - 1$ can affect the output y_t and the other repetitive connections at time t; x input at time b can affect output at time t y and later through repetitive connections. In a simple RNN, all calculations required are shown for calculation at any time step forward as in Eqs. (12.8) and (12.9) (LeCun and Bengio 1995):

$$h_t = \sigma(w_{hx}x_t + w_{hh}h_{t-1} + b_h) \tag{12.8}$$

$$y_t = \text{softmax}\left(w_{yh}h_t + b_y\right) \tag{12.9}$$

Here, w_{hx} refers to the conventional weight matrix between the hidden layer and the input, while w_{hh} is the repetitive weights matrix between the hidden layer in the adjacent time steps and itself. The bias parameters b_h and b_y allow each node to learn the offset. In Fig. 12.9, a block diagram of a simple RNN structure is given.

In Sect. 12.7, long short-term memory is detailed with RNN.

12.6 Long Short-Term Memory

One of the common structures that use components of RNN architecture is long short-term memory (LSTM) (Goodfellow et al. 2016). The structure of LSTM networks consists of memory blocks with memory cells and gate units. There are three special doors in the LSTM memory block: entry, forget, and exit. The core structure of the LSTM is demonstrated in Fig. 12.10. The data input flow to the memory cell is controlled by the input port. The output port manages the output flow from the memory cell to the rest blocks. Forget gate determines the extent to which the former block outputs are effective in the current block. This gate is managed by a simple single-layer neural network. In this gate, the activation is determined in the Eq. (12.10) following Chen and Wang (2017).

$$f_t = \sigma\left(W[x_t, h_{t-1}, c_{t-1}] + b_f\right) \tag{12.10}$$

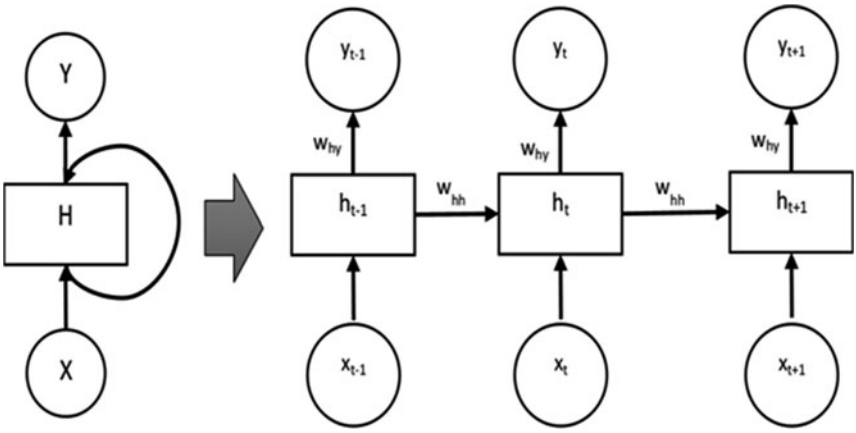

Fig. 12.9 Recurrent neural network structure

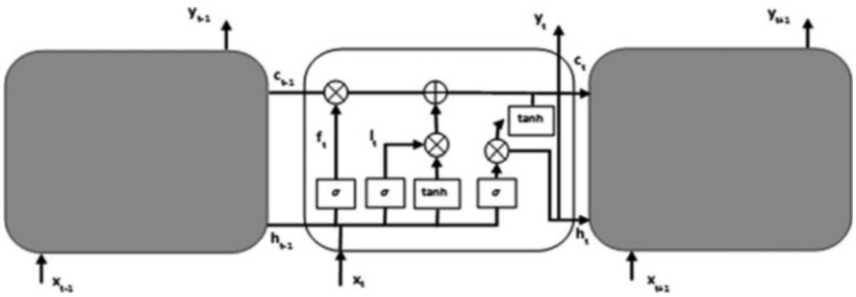

Fig. 12.10 The internal structure of long short-term memory (LSTM)

Here, x_t forms the former block output $h_{t\ -\ 1}$, the former LSTM block memory c_{t-1}, and the bias vector b_f. W is the weight vector of the inputs. The sigmoid activation function represents the output of the forget gate and is implemented by multiplying the previous block memory. Thus, the extent that which the earlier block memory will be effective for the current network is calculated. When the activation of the output vector includes values near zero, the prior memory is forgotten.

Another gate represents the section in which the new memory is formed by a simple neural network (NN) with the "tanh" activation function and the former block memory effect. The mathematical expressions of these operations are shown in Eqs. (12.11) and (12.12), following Chen and Wang (2017).

$$i_t = \sigma(W[x_t, h_{t-1}, c_{t-1}] + b_i) \tag{12.11}$$

$$c_t = f_t * c_{t-1} + i_t * \tanh\left(W[x_t, h_{t-1}, c_{t-1}] + b_c\right) \tag{12.12}$$

Finally, the exit door is the section where the output of the existing LSTM block is produced. These outputs are calculated as in Eqs. (12.13) and (12.14) (Chen and Wang 2017):

$$o_t = \sigma(W[x_t, h_{t-1}, c_{t-1}] + b_o) \tag{12.13}$$

$$h_t = \tanh\left(c_t\right) * o_t \tag{12.14}$$

As with other networks, deep learning structures have become widespread, and the design of LSTM networks to include many layers has revealed the deep structures of these networks.

12.7 A Comparison Between Deep Learning Algorithms

A comparison between DNNs, CNNs, and RNNs deep learning methods is presented in Table 12.2.

12.8 Advantages and Challenges of Deep Learning

The following subsections describe the main advantages of using DL over other ML techniques and the recent challenges that face researchers using DL.

Table 12.2 Comparison between DNNs, CNNs, and RNNs

Features	Deep neural networks	Convolutional neural networks	Recurrent neural networks
Definition	DNNs are a type of MLP with a large number of hidden layers	CNNs are a type of neural network with different special-purpose layers that have been designed to map image data to an output variable	RNNs are a type of neural network with feedback loops in the recurrent layer which act as memory and used to work with sequence prediction problems
Architecture	DNNs include different interconnected layers such as an input layer, one or more hidden layers, and an output layer	The architecture of CNNs is similar to feed-forward artificial neural networks with several different special-purpose layers such as convolution, pooling, and fully connected layers	The architecture of RNNs consists of different interconnected recurrent layers with feedback loops that act as internal memory to process arbitrary sequences of inputs
Input/output size	Both input and output size are fixed	CNN uses inputs with fixed size and generates outputs with a fixed size	RNN can handle arbitrary input/output sizes
Suitability of classification/ regression problems	DNNs are suitable for classification and regression prediction problems	CNNs are suitable for classification and regression prediction problems	RNNS are suitable for Classification problems and Regression prediction problems
Type of data	DNNs can be used on Tabular data or time-series data	CNNs can be used on spatial data or image data	RNNs can be used on temporal data or sequential data
Applications	Applications deal with image data, text data, and time-series data	Applications deal with image data and video data	Applications deal with Text data, Speech data, and Generative models. However, RNNs are not appropriate for tabular and image data
Complexity	Complexity depends on data size and number of hidden layers	Complexity depends on data size, filter size, number of layers, as well as setting the hyperparameters	Complexity depends on data size, number of layers, and amount of required memory

12.8.1 Advantages of DL

The main advantages of DL over regular machine learning are given according to Alom et al. (2019) and Pouyanfar et al. (2018).

- Similar to neural network-based approaches, DL can be utilized in many different applications and data types.
- DL can automatically extract features from the data; therefore, no pre-feature extraction process is required.

- DL is able to drive new feature sets to form a dataset with a limited series of features.
- DL can automatically adapt to overcome the natural variations in the training data.
- With the usage of GPUs, DL can perform a massive amount of computations that can be achieved in a parallel manner. Therefore, DL can be used to generate complex models from large volumes of training data.

Hence, the flexible architecture of DL can be adapted to new problems relatively easily. The extreme performance of DL over other solutions makes DL preferable for different applications such as speech recognition, computer vision, natural language processing, and games.

12.8.2 Challenges of DL

There are several challenges for DL including Alom et al. (2019) and Pouyanfar et al. (2018).

- Data availability and quality: DL performance increases with the amount of available training data. In most industrial applications, large amounts of training data are not often easily available. This becomes a big challenge for DL to work with this small amount of training data and provides high performance. Another challenge for DL is the quality of the training data. When the data is used for training, DL models contain high noise, which can lead DL to degrade and reduce its performance.
- Security: DL is often used to improve and strengthen cybersecurity applications. However, DL networks are vulnerable to malicious attacks by modifying the input to security DL models.
- Higher processing power: When working with large amounts of training data, DL algorithms perform intensive computations and require a huge amount of processing power. Providing powerful computation platforms while keeping the cost down becomes an important challenge for DL.
- Hyperparameter optimisation: DL performance depends on identifying the optimal hyperparameters which are the network parameters required to be initialized before the training process. Identifying the optimal values of DL networks, hyperparameter is the most important challenge for DL, and setting these parameters incorrectly can impact the performance of DL and cause overfitting problems.

12.9 Utilization of DL in Environmental Systems Applications

Natural phenomena affecting planet Earth represent one of the hottest global topics. These phenomena lead to an increase in natural disasters and threaten humans and other life on Earth. Hence, the prediction of changes in environmental systems could be interpreted in reality as protecting thousands of human lives as well as saving resources and money. The losses of facts and statistics regarding natural disasters can be found in the Insurance Information Institute (https://www.iii.org/ accessed 12 May 2021). This topic has attracted the attention of decision-making leadership and the scientific community. Several factors contribute to disasters. In this context, the application of deep learning in predicting different phenomena has been applied. The prediction of earthquakes, rainfall, and environmental protection is discussed in subsections.

12.9.1 Application of DL in Earthquake Prediction

Earthquakes cause great and sudden destruction; these natural disasters negatively affect the lives of thousands of people. Severe earthquakes cause many psychological and negative effects as well as loss of life and property. Every year, there are a large number of earthquakes in different locations with various magnitudes. Figure 12.11 shows the number of earthquakes across the world that occurred in the last decade for the period 2009–2019 according to the United States Geological Survey (https://earthquake.usgs.gov/earthquakes/map/?extent=-55.77657,-520.66406&extent=84.16085,-183.16406) Retrieved 2019.

Natural disasters such as earthquakes do not provide any obvious warning before occurrence; associated losses cannot be prevented. The prediction of earthquakes has always been an interesting subject, and the success of the prediction may save many

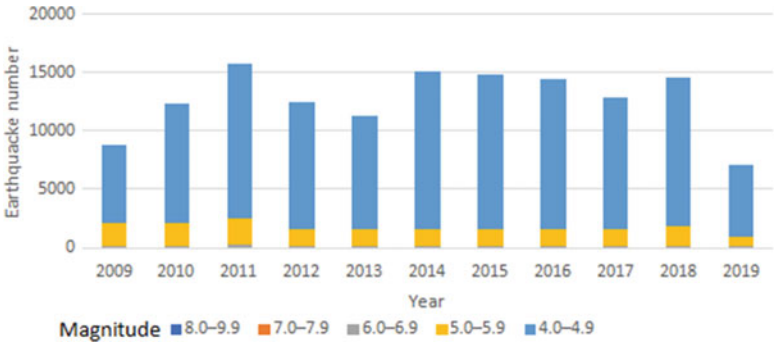

Fig. 12.11 The number of earthquakes worldwide for 2009–2019

thousands of human lives. There are earthquake studies based on mathematical analysis and machine learning algorithms like decision trees and support vector machines, ANNS, and precursors signal in the literature (Sobolev 2015). However, due to the unpredictable complex and, dynamic nature of earthquakes, these methods do not provide consistently accurate results (Sobolev 2015). Deep learning algorithms such as RNN and CNN can capture the complex nonlinear correlations among data with high performance and, therefore, receive huge attention from researchers on the subject of earthquake prediction.

12.9.2 Using RNN Deep Learning for Earthquake Prediction

The work presented by Wang et al. (2017) exploits the DL method of RNN for earthquake prediction. In earthquake prediction, regular machine learning methods are incapable of providing respectable results due to the unpredictable and the dynamic nature of earthquake data. Deep learning methods such as RNNs have the advantage of capturing the complex nonlinear relations in data and so are suitable for complex time-series prediction required in earthquake prediction. Due to crust movement in earthquakes, spatiotemporal correlations of earthquake history data are considered to increase the accuracy of predictions. LSTM variation of RNNs has a strong capability of learning and capturing the nonlinear correlation in earthquake data with long-term intervals.

To improve earthquake prediction, earthquakes have been modelled based on spatiotemporal correlations dividing the area of interest into different subregions.

The model was built based on the raw information of earthquakes such as events ($E = \{e_1, e_2, \ldots\}$), time (t), latitude (La), longitude (Lo), and magnitude (Ma) (Wang et al. 2017). The first step of building the earthquake model is selecting the area of interest in a rectangle shape with four vertices expressed as View, Vsw, Vne, and Vse. The whole area is designated M equal to m_h (horizontal edge) multiplied by m_v (vertical edge); this may be divided into multiple subregions (SR) using Eqs. (12.15) and (12.16).

$$\mathrm{SR_{ve}} = |\mathrm{La(Vnw)} - \mathrm{La(Vsw,)}|/m_h \tag{12.15}$$

$$\mathrm{SR_{he}} = |\mathrm{Lo(Vnw)} - \mathrm{Lo(Vne)}|/m_v \tag{12.16}$$

where $\mathrm{SR_{ve}}$ is the vertical edged subregion and $\mathrm{SR_{he}}$ is the horizontal edged subregion. Each earthquake event is allocated its corresponding subregion as in Eqs. (12.17) and (12.18):

$$i = \left\lfloor \frac{|\mathrm{Lo(e)} - \mathrm{Lo(Vne)}|}{\mathrm{SR_{he}}} \right\rfloor \tag{12.17}$$

$$j = \left\lfloor \frac{|\mathrm{La(e)} - \mathrm{La(Vsw)}|}{\mathrm{SR_{ve}}} \right\rfloor \tag{12.18}$$

Consequently, a temporal segmentation method may be applied to produce the events frequency of each subregion in each time interval Δt. This results in the multi-hot input vector with dimension Mx 1 (M is the total number of subregions) which is later combined with earthquake time series vector L to generate a two-dimensional vector called X and used as an input to the LSTM deep learning algorithm.

For earthquake prediction, LSTM deep learning algorithm is used for training the two-dimensional vector X. At each LSTM layer, h_t is calculated using the following equation:

$$h_t = o_t.\varnothing(c_t) \tag{12.19}$$

The dropout operation is applied to prevent overfitting issues. Then, the results pass through dense network layers to calculate $h_t^D = W_D W_P h_t^L + b$ which passed to softmax activation function and then calculate the loss function based on cross-entropy. The network is learned by minimizing the loss function using a gradient descent scheme.

Two case studies are used for evaluating the performance of the system by training the LSTM using a one-dimensional data vector and the second by training the LSTM using a two-dimensional data vector.

To train the system, the one-dimensional earthquake data is obtained from USGS (US Geological Survey) website for the period between 2006 and 2016 with a 1-month interval and magnitudes greater than 2.5. The results using one-dimensional data show 63.50% for the overall prediction accuracy.

In the second case study, the two-dimensional earthquake data (spatiotemporal) is obtained from USGS website for mainland China in the period 1966–2016 with a 1-month interval and magnitudes greater than 4.5.

The evaluation result for the second case study shows a prediction accuracy of 74.81%. Finally, to compare LSTM deep learning method with the regular machine learning method, the two-dimensional earthquake data are used to train MLP neural networks and obtained an accuracy of 66.99%, which is much lower than the results obtained by using the LSTM deep learning method.

12.9.3 Using CNN Deep Learning for Earthquake Prediction

The work presented by Perol et al. (2018) utilizes the CNN DL method for earthquake prediction. There are many human-induced non-natural causes of earthquakes. Injection of wastewater under the surface of the Earth has a major impact on inducing a large number of moderate and small size earthquakes which can later trigger large magnitude earthquakes. Many earthquake detection methods are designed to detect earthquakes with large magnitudes and not consider the small magnitude earthquakes covered by seismic noise. Detecting small size earthquakes is the key to unlocking their causes to reduce future risks.

In deep learning, CNN (ConvNetQuake) is used to detect and locate earthquakes based on seismogram data. CCN is trained on a big dataset of labeled raw seismic waveforms and learns a compact representation that can discriminate seismic noise from earthquake signals. A large number of raw seismic waveform data along with labels are used to train CNN to discriminate earthquakes from seismic noise. Unlike the regular detection methods, which detect earthquake waveforms based on similarity to other waveforms, the used method is considered a nonlinear local filter that is used to select features in the waveforms that are most relevant to classification. This helps to detect earthquake signals which were missed during training and increases the system performance.

In the modelling of earthquakes using CNN ConvNetQuake, the input of networks takes a three-channel window of waveform data with M number of classes, represented as a two-dimensional tensor $Z^0_{t,c}$ where c is the number of channels and t is 10 s sampling period. The input is fed into a feed-forward network consisting of eight convolutional layers connected to a fully connected Z layer which outputs class scores. Each convolution layer processes the data based on Eq. (12.20):

$$Z^i_{t,c} = \sigma\left(b^i_c + \sum_{\acute{c}=1}^{c_i}\sum_{\acute{t}=1}^{3} Z^{i-1}_{c,st+\acute{t}-1}.W^i_{c\acute{c}}\right) \; i = 1,2,3 \tag{12.20}$$

where σ is the activation function called nonlinear ReLU (rectified linear unit). The output and input channels are indexed with c and $\acute{c}$, respectively, the time dimension is indexed with t and $\acute{t}$, c_i is the number of channels in layer i, and W represents the weights of the network.

The output of the eighth layer is reshaped into a one-dimensional tensor called $\overline{Z}^8$ which contains 128 features. The $\overline{Z}^8$ features vector is followed by a fully connected layer to compute class scores following Eq. (12.21).

$$z_c = \sum_{\acute{c}=1}^{128} \overline{Z}^8_{\acute{c}}.W^9_{c\acute{c}} + b^9_c \tag{12.21}$$

This fully connected layer enables the network to learn and combine multiple parts of the signal to generate the final class score.

Finally, the softmax normalization function is applied to the class scores to generate the final prediction result. The L2-regularized cross-entropy loss function is used as the cost function optimiser for training the ConvNetQuake.

For building the earthquake detection model using ConvNetQuake, continuous ground velocity data records were used. This data was obtained from two local stations GS.OK027 and GS.OK029 in the state of Oklahoma. This data consists of 2918 windows of labeled earthquakes and 831,111 windows of seismic noise for the period 15 February 2014 and 16 November 2016. The training set contains 2709 events and 700,039 noise windows, and the test set contains 209 events and 131,072 windows of noise. The evaluation results of the earthquake detection model using ConvNetQuake deep learning show 94.8% precision and 100% recall which leads to a value of 97% of F-score.

ConvNetQuake earthquake detection method shows better and faster prediction results compared to autocorrelation and "Fingerprint and Similarity Thresholding" (FAST) earthquake detection methods.

12.9.4 Applications of DL in Climate and Weather Forecasting

Predicting changes in climate and weather is a very important issue. Such predictions aid the location of the best places to plant crops as well as prepare for emergency conditions including floods and drought. Machine learning technology can be used successfully to predict the behavior of the climate and the weather. The focus here will be on the application of deep learning approaches to predict rainfall.

12.9.4.1 Rainfall Prediction Using DL

With the gradual increase in temperature of the world associated with global warming, rainfall prediction is increasingly vital for the economy and daily life of many countries. An efficient method of predicting rainfall is a key topic for scientists. Input parameters/attributes which can be utilized to predict rainfall are shown in Table 12.3. Deep learning is one of the promising computation tools that can be used to predict rainfall by using more complex neural network architectures. Recurrent

Table 12.3 Input parameters/attributes for the classifier (Niu and Zhang 2015)

Parameters	Attributes
Temperature	Minimum and maximum values
Humidity	Average value
Pressure	Average, minimum, and maximum values
Evaporation	Minimum and maximum values
Wind	Average and maximum values
Sunshine	Total time

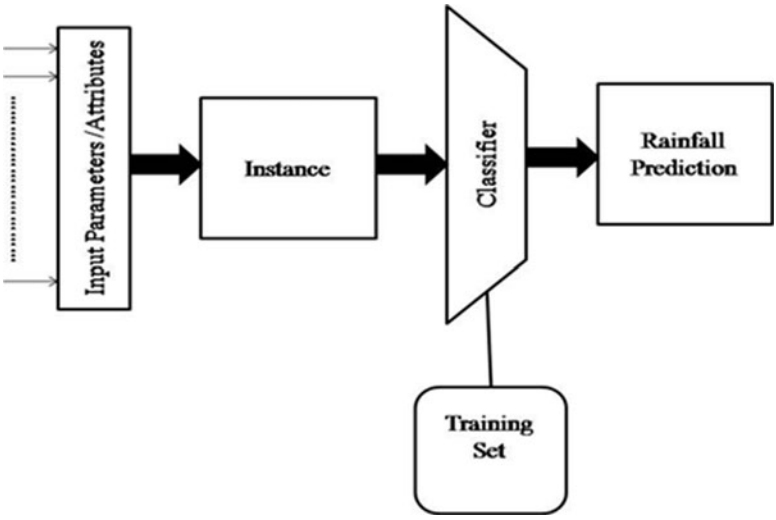

Fig. 12.12 The general structure of rainfall prediction model based on historical data (Niu and Zhang 2015)

neural networks model changes in rainfall through time (Patterson and Gibson 2017). Figure 12.12 shows the general structure of rainfall prediction.

Several methods have been used for rainfall forecasting. Two of those commonly used are empirical and dynamical methods. Regression, stochastic, artificial neural network, and fuzzy logic are examples of empirical methods, which rely on the analysis of available historical data of the weather with the correlation to different atmospheric variables. The second approach for prediction is represented by employing physical models, which are dependent on modeling equations of the climate system based on the atmospheric conditions (Zaw and Naing 2008).

The common forecasting processes of empirical methods to quantify the amount of rainfall are as follows:

(i) Gathering data
(ii) Reduction explanatory predictors
(iii) Building a model using one of the empirical methods
(iv) Checking through the validation procedure

However, RNN DL algorithm is applicable for rainfall prediction problems as it can model changes in data across time which is a property of rainfall data. Due to the fact that RNNs are generally used for classification problems and rainfall prediction problem is considered a regression problem, a regression output layer can be added to a RNN. Figure 12.13 shows the architecture of RNN used for regression problems. The general equation of deep RNNs is shown in Eq. (12.22) (LeCun and Bengio 1995).

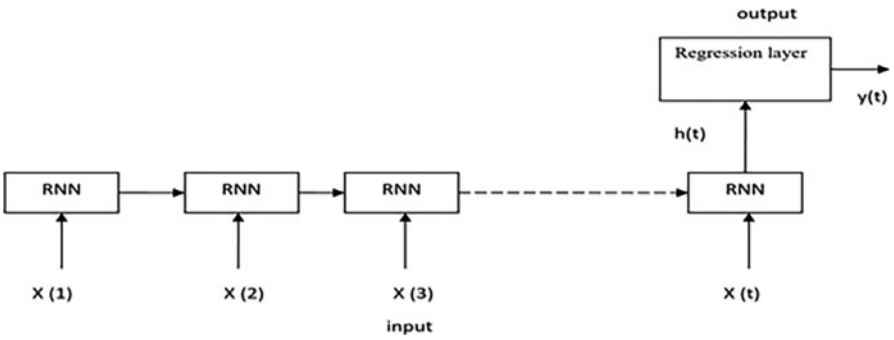

Fig. 12.13 RNN architecture for regression problems

$$y_t = W_{hy} \tanh\left(W_{hh}h_{t-1} + W_{xh}x_t\right) \tag{12.22}$$

where x_t is input and y_t is the output and W is the network weight.

12.9.5 Applications of DL in Environmental Protection and Sustainability

Environmental issues such as air pollution or water contamination are global problems, requiring special efforts from governments and individuals. Hence, using DL techniques, as part of information technology, in predicting conditions of the environment is a key research topic. One of the research trends is to rely on Google search data to expect the state of pro-environmental consumption, which can be achieved by using the consumption index. This index has been determined as an indication of pro-environmental consumption policy, calculated for each country by international organizations. It is worth mentioning that there are different indexes developed for specific purposes, including the Environmental Performance Index (EPI) and the Environmental Sustainability Index (ESI). The values of indexes range from 0 to 100. Calculation methods are of importance and can be achieved based on the principle of *proximity to target* to find how close the policies of countries are in environmental objectives recognized by international organizations (Lee et al. 2017). For forecasting the pro-environmental consumption index, a model is built using a RNN deep learning algorithm. RNN is one of the deep learning technologies that builds through the ANN model but can use prior relationships. In the operation of the RNN model, the data of a previous period (h_{t-1}) is incorporated recursively as input data, and the output (y_t) for time (t) will be calculated as shown in Eq. (12.23).

$$y_t = f\left(h_{t-1} \times W^h + x_t \times W^x\right) \tag{12.23}$$

The RNN model consists of a Gompertz of activation function, 100 Epoch repetitions, and the number of hidden layers and is 10 with 20 batch sizes and a 0.02 learning rate. The model was built using RNN for pro-environmental consumption forecasting was tested and verified using 84 different datasets and provides better results than traditional neural network methods (Lee et al. 2017).

Sustainability denotes the production of manufactured products through economic processes that minimize negative environmental impacts while conserving energy and natural resources (Lee et al. 2017). Hence, important points that should be applied to maintain sustainable manufacturing are (Dincer and Acar 2015) minimizing the reduction of natural resources and the possibility of fulfilling the present and future energy requirements, providing products with high efficiency, and avoiding or keeping toxic emissions including greenhouse gases emission to minimum levels. DL shows its extreme power in many areas and applications. However, sustainability and energy efficiency problems still require more investigation. DL is a powerful tool that will provide the framework to work towards environmental sustainability solutions in the future (Meng et al. 2018).

12.10 Conclusion

This chapter has focused on DL technology with environmental applications. DL can be considered a promising tool in the field of environmental challenges including both classifications and predictions. The chapter also discussed the main algorithms of DL with a brief comparison revealing the advantages and the challenges of each. There are several applications of DL in environmental systems; three of the most common applications have been exemplified, earthquake prediction, rainfall prediction, and environmental protection and sustainability. In environmental systems applications, the advantage of using DL is that it achieves better accuracy and reliability compared to regular ML techniques.

DL could be applied in earthquake prediction for both classification and regression problems, where it is used to predict the occurrence and magnitude of earthquakes. Rainfall prediction can be considered as a regression problem, and DL may be used to predict the amount of rain in particularly vulnerable regions. In environmental protection and sustainability applications, problems can be treated as classification and regression problems, and DL can be utilized to predict several issues such as the amount of air pollution and water contamination. Regarding sustainability applications, DL can be used to find proper solutions for production problems in manufacturing, which may ensure the efficient consumption of natural resources. Hence, DL is a powerful tool representing a key to unlocking complexity and validating future applications of environmental systems.

References

Ahmed RN, Hayri S (2018) A large-scale Arabic sentiment corpus construction using online news media. J Eng Appl Sci 13:7329–7340

Alom MZ, Taha TM, Yakopcic C et al (2019) A state-of-the-art survey on deep learning theory and architectures. Electronics 8(3):292

Chen J, Wang D (2017) Long short-term memory for speaker generalization in supervised speech separation. J Acoust Soc Am 141(6):4705–4714

Dey A (2016) Machine learning algorithms: a review. Int J Comput Sci Inf Technol 7(3): 1174–1179

Dincer I, Acar C (2015) A review of clean energy solutions for better sustainability. Int J Energy Res 39(5):585–606

Dumoulin V, Visin F (2016) A guide to convolution arithmetic for deep learning. arXiv. https://doi.org/10.48550/arXiv.1603.07285

Franklin J (2005) The elements of statistical learning: data mining, inference and prediction. Math Intell 27:83–85. https://doi.org/10.1007/BF02985802

Fukushima K (1980) Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol Cybern 36(4):193–202

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. MIT press. https://doi.org/10.1007/s10710-017-9314-z

Gurney K (2014) An introduction to neural networks. CRC Press. https://doi.org/10.1117/3.633187

Hernández-Orallo J (2017) Evaluation in artificial intelligence: from task-oriented to ability-oriented measurement. Artif Intell Rev 48(3):397–447

Hinton GE, Dayan P, Frey BJ et al (1995) The "wake-sleep" algorithm for unsupervised neural networks. Science 268(5214):1158–1161

Hinton GE, Srivastava N, Krizhevsky A et al (2012) Improving neural networks by preventing co-adaptation of feature detectors. arXiv. https://doi.org/10.48550/arXiv.1207.0580

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hu W, Huang Y, Wei L et al (2015) Deep convolutional neural networks for hyperspectral image classification. J Sens 2:1–12. https://doi.org/10.1155/2015/258619

Ivakhnenko AG, Lapa VG (1966) Cybernetic predicting devices (No. TR-EE66-5). Purdue Univ Lafayette Ind School of Electrical Engineering

James G, Witten D, Hastie T et al (2013) An introduction to statistical learning. Springer Texts in Statistics. Springer

Kin ZB (2019) Classification of Turkish sign language alphabet by deep learning method. Master's thesis, Başkent University Institute of Science and Technology

Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: NIPS'12 Proceedings of the 25th International Conference on Neural Information Processing Systems – Vol 1. https://doi.org/10.5555/2999134.2999257

LeCun Y, Bengio Y (1995) Convolutional networks for images, speech, and time series. In: Arbib MA (ed) The handbook of brain theory and neural networks. MIT Press

LeCun Y, Boser B, Denker JS et al (1989) Backpropagation applied to handwritten zip code recognition. Neural Comput 1(4):541–551

LeCun Y, Boser BE, Denker JS et al (1990) Handwritten digit recognition with a back-propagation network. In: Touretzky D (ed) Advances in neural information processing systems (NIPS 1989). Morgan Kaufmann, Denver

Lee D, Kang S, Shin J (2017) Using deep learning techniques to forecast environmental consumption level. Sustainability 9(10):1894. https://doi.org/10.3390/su9101894www.mdpi.com/journal/sustainability

McCarthy JJ, Minsky ML, Rochester N (1959) Artificial intelligence. Research Laboratory of Electronics (RLE) at the Massachusetts Institute of Technology (MIT)

Meng Y, Yang Y, Chung H et al (2018) Enhancing sustainability and energy efficiency in smart factories: a review. Sustainability 10(12):4779

Niu J, Zhang W (2015) Comparative analysis of statistical models in rainfall prediction. In: 2015 IEEE international conference on information and automation. IEEE, pp 2187–2190

Patterson J, Gibson A (2017) Deep learning: a practitioner's approach. O'Reilly Media, Inc

Perol T, Gharbi M, Denolle M (2018) Convolutional neural network for earthquake detection and location. Sci Adv 4(2):e1700578. https://doi.org/10.1126/sciadv.1700578

Pouyanfar S, Sadiq S, Yan Y et al (2018) A survey on deep learning: algorithms, techniques, and applications. ACM Comput Surv 51(5):92

Samuel AL (1988) Some studies in machine learning using the game of checkers II – recent progress. IBM J Res Dev 1967:601–617. Reprinted In: Levy DL (ed) Computer games. Springer-Verlag

Schmidhuber J (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117. https://doi.org/10.1016/j.neunet.2014.09.003

Şeker A, Diri B, Balık HH (2017) A review of deep learning methods and applications. Gazi J Eng Sci 3(3):47–64

Sobolev GA (2015) Methodology, results, and problems of forecasting earthquakes. Her Russ Acad Sci 85(2):107–111

Vieira A, Ribeiro B (2018) Deep learning: an overview. In: Viero A, Ribeiro B (eds) An introduction to deep learning business applications for developers. Apress, Berkeley. https://doi.org/10.1007/978-1-4842-3453-2_2

Wang Q, Guo Y, Yu L et al (2017) Earthquake prediction based on spatio-temporal data mining: an LSTM network approach. IEEE Trans Emerg Top Comput 8(1):148–158

Zaw WT, Naing TT (2008) Empirical statistical modeling of rainfall prediction over Myanmar. World Acad Sci Eng Technol 2(10):500–504