International Journal of Data Informatics and Intelligent computing (IJDIIC)
Vol. 2, No. 4, December 2023, pp. 1~10
DOI: 10.59461/ijdiic.v2i4.78

Deep learning algorithms and their relevance: A review

Nisha.C.M[1], N. Thangarasu[1]
[1]Department of Computer Science, Karpagam Academy of Higher Education, Coimbatore, India

Article Info

Article history:

Received September 23, 2023
Revised October 27, 2023
Accepted November 04, 2023

Keywords:

Neural networks
Convolutional Neural network
Recurrent Neural Network
Long Short-Term Memory
Generative Adversarial Network

ABSTRACT

Nowadays, the most revolutionary area in computer science is deep learning algorithms and models. This paper discusses deep learning and various supervised, unsupervised, and reinforcement learning models. An overview of Artificial neural network(ANN), Convolutional neural network(CNN), Recurrent neural network (RNN), Long short-term memory(LSTM), Self-organizing maps(SOM), Restricted Boltzmann machine(RBM), Deep Belief Network (DBN), Generative adversarial network(GAN), autoencoders, long short-term memory(LSTM), Gated Recurrent Unit(GRU) and Bidirectional-LSTM is provided. Various deep-learning application areas are also discussed. The most trending Chat GPT, which can understand natural language and respond to needs in various ways, uses supervised and reinforcement learning techniques. Additionally, the limitations of deep learning are discussed. This paper provides a snapshot of deep learning.

Corresponding Author:

Nisha. C.M
Research Scholar
Department of Computer Science
Karpagam Academy of Higher Education, Coimbatore
India

Email: nishacm2008@gmail.com

1. INTRODUCTION

Human life has always been searching for comfortable living and has changed stoneage, wheel invention, the industrial revolution, the invention of Electricity, and the Communication and IT revolution. Today, the world is facing a computer revolution and considers computers to be not mere slaves but more intelligent machines. People are now interested in creating machines that can see, hear, feel, think, and analyze like human beings. The beginning of the 1940s saw the start of artificial intelligence, which involved creating machines that mimic human behavior. Later, a subfield of AI called machine learning grew, using statistical methods to improve tasks through experience. Deep learning, a subfield of machine learning, is equipped for performance using neural networks similar to those in the brain for task completion. The enormous amount of data and improved hardware configurations like GPUs (Graphic processing units), TPUs(Tensor Processing units), and various deep learning algorithms have paved the way for the growth of deep learning within two decades. Several deep learning algorithms suitable for the identification of text, speech, images, and videos are identified, and their theoretical backgrounds are discussed in this article. Applications of deep learning algorithms in various fields are discussed in this article.

Deep learning evolved from machine learning a few decades ago, designed to train and operate on deep neural networks. Deep learning imitates the working of human neurons and the interconnection of neuron structures, advancing through intellectual and information theories [2]. The massive amount of data available due to the increased use of the internet and database systems has helped the progress of deep learning. The low cost of hardware, especially GPUs (Graphic processing units), facilitated the researchers' focus on deep learning rather than machine learning. Deep learning algorithms can perform both simultaneously, unlike machine learning algorithms, which require feature extraction and modeling separately. The advanced architecture of deep neural networks can extract features from unstructured data without much human effort [3].

The paper [4] discusses various deep learning applications, such as NLP (Natural Language Processing), Audio and speech processing, image processing, and social media analysis. NLP includes chatbot answering, Translating textual data to a different language, Identifying sentiments included in comments, Rephrasing, Identification, and precise creation. Visual data processing consists of classifying images, identifying objects in images, dividing images into various parts, extracting videos, visual datasets for detecting events, classification, and understanding. Audio and speech recognition include speech emotion recognition, speech enhancement, and speech separation. We can see deep learning applications in recommendation systems on Amazon, movie recommendations on Netflix, Self-driving cars by Tesla, automatic tagging of friends by Facebook, and spam detection in e-mails. Other areas include congestion avoidance in transportation, Disaster management systems for better decision-making capabilities, agriculture for predicting crop yields, soil moisture, identification of weeds and diseases, and weather prediction. Biomedicine for evaluating stroke, breast cancer, rare diseases, drug discovery, and chatbots for identifying patient symptoms have vast applications using deep learning. The latest application, ChatGPT, works based on neural networks similar to the working of the human brain, providing perfect answers in the text as per the questions asked, making humans' jobs easier.

2. DEEP LEARNING MODELS

Figure 1 shows deep learning models are categorized mainly as supervised, unsupervised, and deep reinforcement models. Labeled data is used for training the model in supervised models, but in unsupervised models, data is classified by studying the common characteristics among data. Classification and Regression are two subfields of supervised learning algorithms.

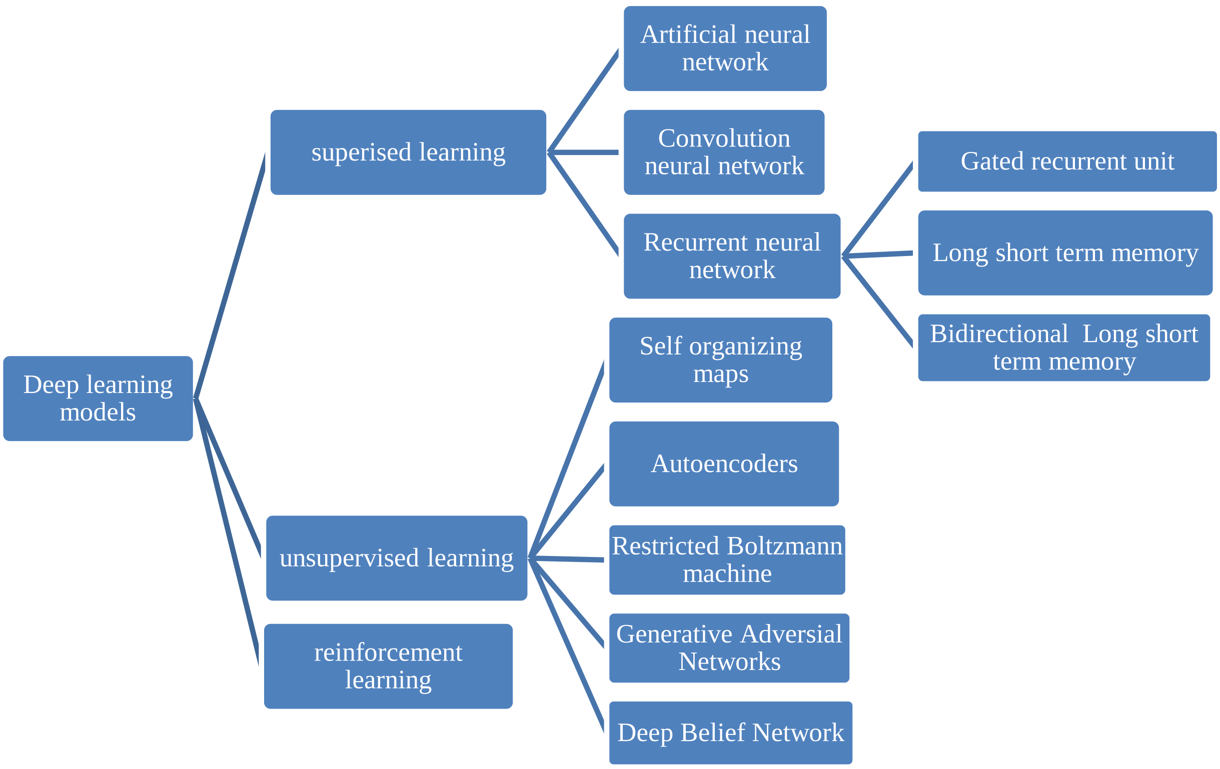

Figure 1. Deep learning models

2.1. Supervised Method

The supervised learning model works under supervision, and there is knowledge about the required output. Labeled data is used in supervised learning, and most collected data is required for training purposes. Another small percentage of data is used for testing, and the remaining portion of data is used for validation[1]. Supervised learning is best suited for future prediction, extraction of knowledge, fraud detection, and compression.

2.1.1. Artificial Neural Networks (ANN)

Artificial Neural Networks (ANNs) are very powerful deep learning models used for the functioning of complex systems. The paper [2] reviews a wide variety of ANN applications in fields such as science, engineering, medical science, agriculture, ecology, management, finance, insurance, nuclear field, mining, quality prediction of crude oil, crime detection in financial management, and money laundering. ANNs are inspired by the working of the human brain, utilizing a massive number of neural networks that consist of various processors interconnected with each other. An ANN mainly involves an input layer for receiving data. It utilizes hidden layers to apply mathematical calculations to the accepted data. Lastly, an ANN contains output layer for predicting the outcomes. Each layer consists of a number of nodes for processing data. Data is passed from one layer to another layer only if the outcome of one node in one layer is above a particular limit.

2.1.2. Convolutional Neural Networks (CNN)

The Convolutional neural network (CNN) is mainly used for visual data processing and classification. The convolution layer is the main block of a CNN where the majority of computations take place. The convolution layer requires an input image and a kernel or filter to move over the image, for extracting features, a process known as convolution. The two important hyper parameters used in CNN are the number and size of kernels[3]. The kernel is moved across the image and the dot product of image pixels and kernel values is calculated. There is a chance for losing information at the border of the image with certain kernel sizes and zero values are introduced to enlarge the image through padding[4].

6	4	1	2	5
3	0	8	9	3
4	1	5	7	6
3	6	7	8	2
7	2	5	6	9

2	1	0
3	2	1
0	2	4

Figure 2. Input Image with Kernal

The dot product is (6*2) +(4*1) +(1*0) +(3*3) +(0*2) +(8*1) +(4*0) +(1*2) +(5*4) =55 and it is shown in the table given below. The kernel is moved right, and the next dot product with image pixels is generated. (4*2) + (1*1) + (2*0) + (0*3) + (8*2) + (9*1) + (1*0) +(5*2) + (7*4) =72.

55		

55	72	87
65	74	84
61	81	104

Figure 3. Feature Maps

A feature map or activation map is created as shown above, after the kernel is strided through the entire image. The distance between two kernel positions is called the stride, and often it is taken as one. The second layer is called the pooling layer, which performs downsampling for dimensionality reduction. Max pooling and average pooling are the most sought-after pooling methods. The maximum value is chosen from a filter, usually a 2x2 matrix with a stride of 2, by means of the max pooling method. Stochastic pooling, spatial pyramid pooling, and Def-pooling are the three approaches connected with pooling layers discussed in[5]. Overfitting problem of max, pooling is avoided through the stochastic pooling method, in which activation is randomly chosen. Fixed-length exemplification can be done through spatial pyramid pooling, which is a solution for images of various sizes and scales. Visual pattern deformations are more effectively studied using def-pooling rather than max-pooling and average pooling.

9	7	6	3
5	4	8	2
3	6	3	1
2	1	2	8

9	8
6	8

Figure 4. Max Pooling Example

The fully connected layer is the last layer, which transforms the down-sampled feature maps into a dimensional array of numbers[3] consisting of final outputs. Since values ranging from 0 to 1 are suitable for the prediction of outputs like classification, the softmax function is used in a fully connected layer. In the early layers of convolution and pooling, the Relu function is used. During the training of datasets, loss functions are calculated by comparing the predicted output and actual output. Then, backpropagation is performed by updating the parameters for better results. After several hyperparameter tuning sessions during the training phase, validation is performed using a completely new dataset.

2.1.3. Recurrent Neural Network

RNNs (Recurrent Neural Networks) are mainly used for the prediction of time-series data, especially in speech recognition. An RNN contains a recurrent loop, as given in the figure, to retain information for a particular time period. Hidden state functions are used as memory in RNNs. Even though RNNs are most suited for time series data, they can also be used for pattern recognition, NLP (Natural language Processing), machine translation, voice recognition, and speech processing. Usually, RNNs are used for processing one-dimensional data in a sequence, but they can also be used for processing two-dimensional data like images. In the image given below, ‘x’ symbolizes the input, ‘y’ denotes the output, and ‘h’ indicates the hidden layer.

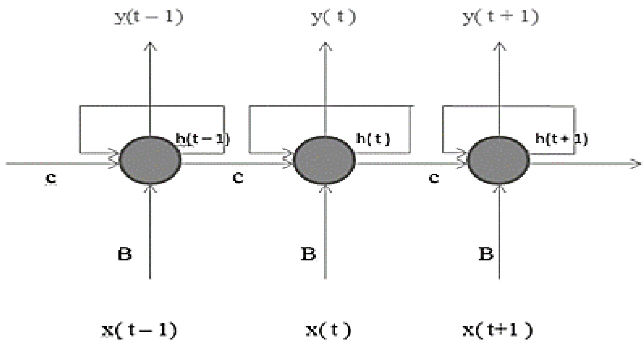

Figure 5. Recurrent neural network(RNN)

Here, h(t) is calculated by means of the equation f(h(t-1), x(t)). x(t) is the symbol used to denote the present input, and h(t-1) is used to represent the earlier state, while h(t) is used to denote the new state. An RNN Encoder-Decoder is proposed in the paper[6], which consists of an encoder for obtaining a fixed-length vector from a variable-length source and a decoder for obtaining a target sequence. RNNs suffer from vanishing gradient problems and exploding gradient problems, and in order to overcome this, variations of RNNs, namely LSTM and GRU, are used to solve the problems[7]. This paper also introduces an Independently recurrent neural network (IndRNN) in which the Hadamard product is used to process recurrent inputs, and neurons in one layer are not dependent on each other. Several layers can be stacked together so that the depth of the network can be increased and can be efficiently utilized for long sequences using Relu activation functions. This solves the problem of exploding gradient and vanishing gradient.

2.1.3.1. Long Short-Term Memory

Cells are used in LSTM to retain values for a sufficient amount of time so that they can be utilized for future calculations. The LSTM consists of 3 cells in which the input gate accepts and stores data. The forget gate decides whether to remember or forget data. The only required information in the cell is provided by the output gate. A state vector transmission line is also present for long-term memory, whereas the three input gates are used for short-term memory. The paper [8] discusses the working of the LSTM model. Firstly, in an LSTM model, the output of a forget gate is determined using the formula,

$$f_t = \sigma (W_f . [h_{t-1}, xt] + b_f) \tag{1}$$

Here, f_t is the symbol for denoting the output. σ is the symbol representing the activation function. h_{t-1} is the sign depicting the output of the prior cell. x_t is the the sign showing the input of the present cell. Weight is signified by $W_{f,}$ and bias is indicated by b_f. Then, LSTM determines what to be retained in the cell. For that, two formulas are used, one for representing the input gate and another for finding the new candidate value vector $C_t^{'}$.

$$i_t = \sigma (W_i . [h_{t-1}, xt] + b_i) \tag{2}$$

Where W_i is the weight of the input cell and b_i is the bias of the input gate.

$$C_t^{'} = \tanh (W_c . [h_{t-1}, xt] + b_c) \tag{3}$$

Where Wc is the weight and b_c is the bias of the new candidate value vector. The new state C_t is obtained by the formula,

$$C_t = f_t * C_{t-1} + i_t * C_t'. \tag{4}$$

Then comes the output gate, which also consists of two states.

$$O_t = \sigma (W_o . [h_{t-1}, xt] + b_o) \tag{5}$$

where O_t is the output gate, and h_t is the final outcome obtained by,

$$h_t = O_t * \tanh(C_t). \tag{6}$$

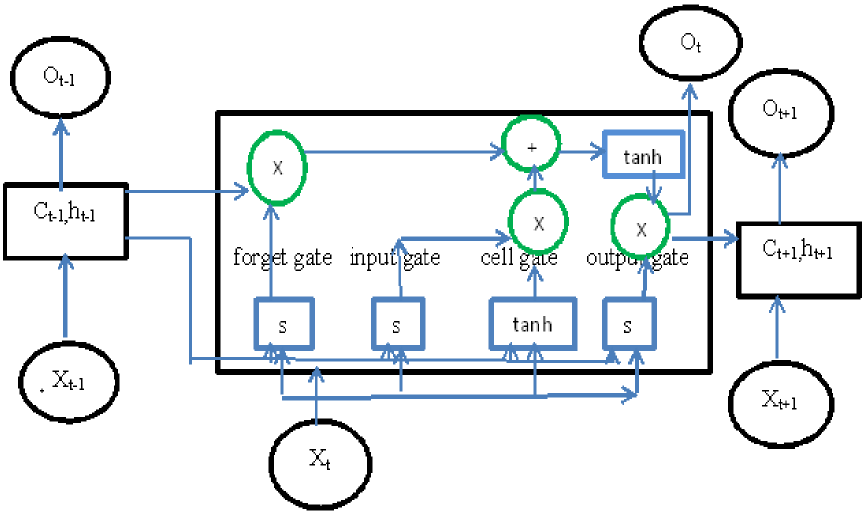

Figure 6. Long short-term memory(LSTM)

2.1.3.2. Bidirectional -LSTM

LSTMs can only move in the forward direction, and future input information cannot be obtained. Bi-LSTM, as the name suggests, can move in both directions so that future and past inputs can be used for decision-making. Hence, Bi-LSTMs are very suitable for natural language processing since they produce meaningful output. Because of long gradient chains, Bi-LSTMs are very costly to train.

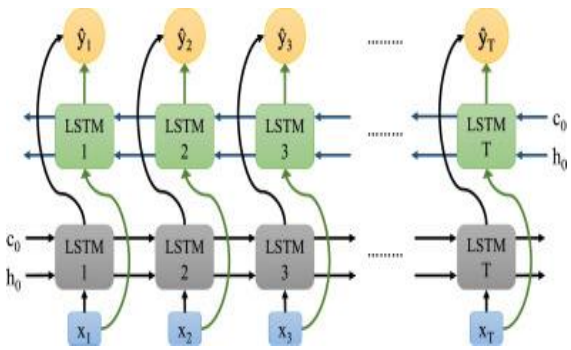

Figure 7. Bidirectional LSTM [9]

2.1.3.3. Gated Recurrent Unit

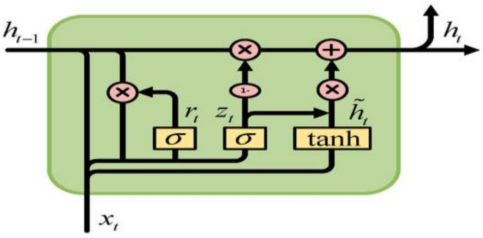

Figure 8. Gated Recurrent Unit(GRU)[10]

GRU is comparable to LSTM, but it contains only two gates. The LSTM's input gate and forget gate are joined to construct the update gate in GRU. An update gate decides the need to modify the content of the cell. Previous cell contents are changed into new ones based on the decision of the update gate[11]. The reset gate is used to determine how much previous data should be forgotten.GRUs are mainly used for modeling sequential data such as speech identification, NLP, and video processing.

The first stage in GRU is generating a candidate hidden state to control the preceding hidden state. The formula used for calculating the candidate's current state, as given in the figure, is

$$H_t^{\sim} = \tanh\,(xt * U_g) + (rt\,.\,H_{t-1}) * W_g) \tag{7}$$

Where it is the outcome of the reset gate, Ug and Wg are weight matrices of the reset gate. The input is denoted by $x_{t,}$ and the preceding hidden layer is represented by h_{t-1}. Depending on the value of $rt_{,}$ the previous state h_{t-1} is considered. Then, the current hidden state is generated by the current hidden state. The equation used here is,

$$H_t = U_t\,.H_{t-1} + (1-U_t).H_t^{\sim}. \tag{8}$$

2.2. Unsupervised learning method

Unsupervised learning models learn on their own, and no outcomes are predicted earlier for a sample. Lots of unlabelled data is needed for training and for correct prediction since no intervention from the outside exists. A general introduction to various unsupervised models is discussed here.

2.2.1. Self-Organising Maps

SOMs are used to represent low-dimensional features obtained from high-dimensional data. A self-organizing map consists of only two layers, namely, the input layer and the feature map. SOM does not use backpropagation and gradient descent but uses competitive learning to retain input features[12].

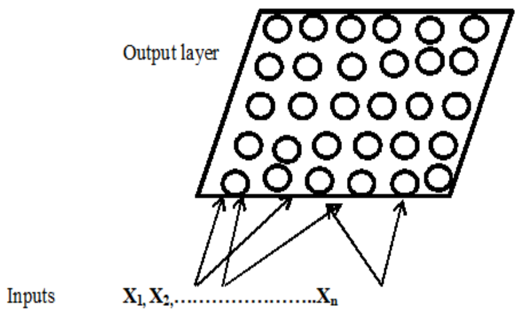

Figure 9. Self organising maps(SOM)

2.2.2. Stacked Auto Encoders

Autoencoders are a combination of encoders and decoders in which both encoding and decoding work is performed. The main objective of a stacked autoencoder is dimensionality reduction and the regeneration of original data. All the inputs are taken and transformed into hidden layers, and these hidden layers are then transformed back into input layers. Usually, several autoencoders are stacked on top of one another. The paper[13] explains the training of stacked autoencoders in two stages, namely pre-training and fine-tuning. There can be 'n' autoencoders in which training starts from bottom to top[14]. Training of the first encoder is performed to minimize the error, and the output of the first encoder is passed to the input of the second encoder, continuing this to the last layer. The classifier gets the output of the last layer, which is often fine-tuned. Fine-tuning starts by marking the nodes with output so that learning can occur, and training of the network is done through back propagation[11]. Various types of autoencoders, like sparse autoencoders, are discussed in [12].

2.2.3. Restricted Boltzmann Machine

The Restricted Boltzmann Machine(RBM) has two layers: the input layer for accepting the observed part and the hidden layer for extracting the required features from it. The output of one RBM can be sent to the input of another RBM and so on to form a stacked RBM.

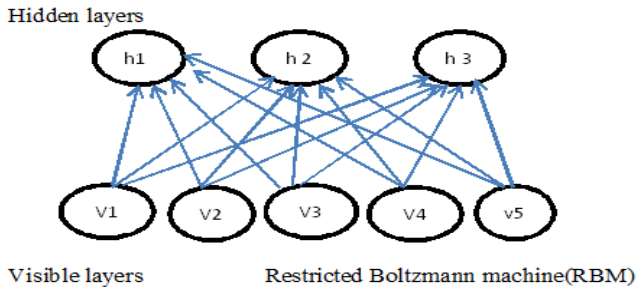

Figure 10. Restricted Boltzmann machine(RBM)

The RBM can be considered a stochastic neural network in which nodes are used to represent neurons, and edges represent connections among nodes[15]. An RBM is represented by a deterministic function $\{0,1\}^{m,}$ which maps to R^n. The learning process in RBM can be done using the Gradient method algorithm in which logpv(Θ) is considered as the likelihood function or by the contrastive divergence method [16].

2.2.4. Generative Adversarial Networks

Generative Adversarial Network (GAN) works on the basis of dual competing parts, namely, the generator for creating data and the discriminator for identifying the original one. GANs are used to classify inputs into real and fake data using a very small number of labeled examples. Missing data can also be imputed using effective generative models. In addition to this, GANs can be used for image editing, image synthesis, and image retrieval. The generator always attempts to create fake data along with real data to mislead the discriminator. On the other hand, the discriminator is competent in discriminating fake data from genuine data and classifying it correctly. The generator is trained using a backpropagation algorithm with respect to the derivative of the output produced by the discriminator from the inputs[17].

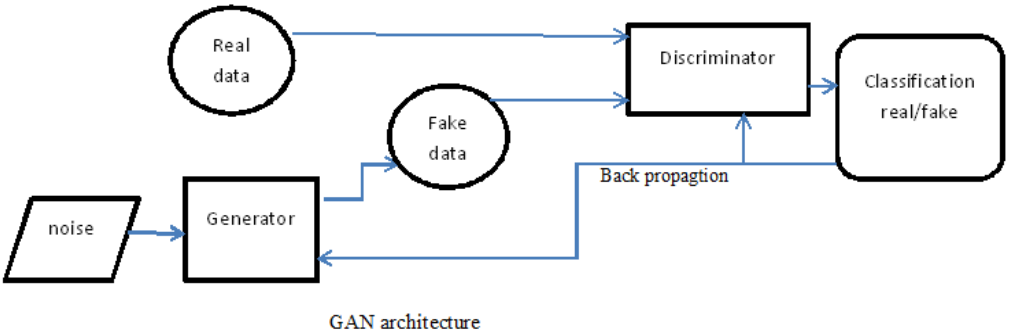

Figure 11.Generative adversarial network(GAN)

Various types of GANs are discussed in [18], including Fully Connected GANs, Convolutional GANs, Conditional GANs, GANs with inference models, and adversarial autoencoders. In fully connected GANs, both the generator and the discriminator use multiple fully connected layers after multiple convolution layers. The discriminator uses the average pooling method. Fractionally stridden convolutions are used in Convolutional GANs without pooling layers and fully connected layers. For faster convergence, conditional GANs are used, in which the output by the generator can be controlled by giving appropriate class labels. An encoder with a decoder is present in the generator of GAN inference models to fool the discriminator. The discriminator has to identify which pair is the real from the fake pair of data. Gans's adversarial loss idea is combined with autoencoder architecture to form adversarial autoencoders. Instead of a random vector z, an adversarial autoencoder uses an image as its input.

2.2.5. Deep Belief Network

Several RBMs are piled together to build a deep belief network (DBN). Since an RBM consists of a visible layer and a hidden layer, the DBN is formed with multiple hidden layers interconnected between them. Each layer of the RBM is trained separately, and the result of one RBM is then passed to the subsequent RBM, facilitating smooth learning of the DBN. Using the backpropagation method, fine-tuning is done to minimize the error.

2.3. Reinforcement learning

In reinforcement learning, the outcomes are not known beforehand and are not conveyed to the model during training. Instead, based on the obtained results, the model is given a reward if the predictions are accurate, enabling the model to identify whether the training has been successful. Reinforcement learning is best suited for game-playing and training robots since it learns by providing appropriate rewards for optimal moves. The model must interact with the environment to achieve the best results from the rewards and penalties it receives. The concepts of 'state' and 'action' are mentioned in the paper [19]. 'Action' refers to the decision made by an agent, while 'state' represents the factor the agent considers when making that decision. The value function helps an agent make decisions in an environment to maximize long-term rewards. In reinforcement learning, a value function is used to estimate the expected cumulative reward or value of being in a certain state or taking a specific action in a particular state.

3. LIMITATIONS

The limitations of DBN, SOP, and MLP are discussed in [20], encompassing vanishing gradient and exploding gradient problems, overfitting of data and model compression in medical applications, imbalanced data problems in ad-hoc network applications, data interpretability, deficiency of training data in intrusion detection, data underspecification and disastrous forgetting in biological applications, and so on. The paper[21] also delves into the dimensionality reduction problem, time consumption, and the costly nature of training data.

During backpropagation, the gradients determine the learning rate, and sometimes, the gradients become very small due to poor training, leading to the vanishing gradient problem. When gradients become very small, the updates to the weights in early layers of the network become negligible, making it challenging for these layers to learn and adapt. As a result, the network might not effectively learn the features and patterns in the data, particularly in the lower layers. This issue can significantly slow down or even halt the learning process. The Exploding gradient is the opposite of the vanishing gradient. The exploding gradient problem occurs when very large gradients occur for larger weights, making learning impossible during backpropagation. Both vanishing and exploding gradients are problematic for effectively training deep neural networks. Techniques like weight initialization, appropriate activation functions, gradient clipping, and batch normalization are often used to mitigate these issues and stabilize the training process[22]. Moreover, using alternative activation functions like ReLU (Rectified Linear Unit) and its variants can help alleviate the vanishing gradient problem.

Overfitting occurs when the algorithm performs well during the training phase but yields disparate results for unknown data values. Premature convergence is a problem in deep learning resulting from the early settling of weights and biases to a local minimum instead of reaching the global minimum[23]. Compared to machine learning algorithms, deep learning models require more time for training, validation, and testing data. Large data sets are required for training the deep learning models, which are quite time-consuming and expensive. For training deep learning models, TPUs (Tensor processing units) and GPUs are required, which are expensive.

4. CONCLUSION

The paper discusses the importance of deep learning in the present scenario, providing an overview of various deep learning algorithms. The architecture of deep learning models and their significance in various fields is also discussed. Various supervised learning algorithms like ANN, CNN, RNN, and variants of RNN like GRU, LSTM, and Bi-directional LSTM are discussed in this paper. Unsupervised learning models like SOM, stacked autoencodes, RBM, GAN, and DBN are also explored. The paper gives an insight into reinforcement learning. Additionally, it mentions various limitations of deep learning models, such as vanishing gradient, exploding gradient, and overfitting. While computers have historically excelled at numerical tasks, they were not initially suitable for complex tasks such as image identification, speech recognition, video processing, and natural language processing, domains where humans demonstrated superior capability. However, in recent times, thanks to the advancement of deep learning models, computers have made remarkable progress and now excel in tasks that were once considered the exclusive domain of human competence. Different deep learning models are suitable for various application fields, and it is up to the researcher to identify the suitable deep learning models for solving the problem of their interest. We are in a situation now where we can't think of living without deep learning applications, as any kind of data, such as text, image, audio, and video, can be processed with much higher accuracy and speed.

REFERENCES

[1] W. G. Hatcher and W. Yu, "A Survey of Deep Learning: Platforms, Applications, and Emerging Research Trends," *IEEE Access*, vol. 6, pp. 24411–24432, 2018, doi: 10.1109/ACCESS.2018.2830661.

[2] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, p. e00938, Nov. 2018, doi: 10.1016/j.heliyon.2018.e00938.

[3] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, pp. 611–629, Aug. 2018, doi: 10.1007/s13244-018-0639-9.

[4] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects," *IEEE Trans. Neural Networks Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022, doi: 10.1109/TNNLS.2021.3084827.

[5] Y. Guo, Y. Liu, A. Oerlemans, S. Lao, S. Wu, and M. S. Lew, "Deep learning for visual understanding: A review," *Neurocomputing*, vol. 187, pp. 27–48, Apr. 2016, doi: 10.1016/j.neucom.2015.09.116.

[6] K. Cho et al, "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation", [Online]. Available: http://arxiv.org/abs/1406.1078

[7] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently Recurrent Neural Network (IndRNN): Building A Longer and Deeper RNN," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, IEEE, Jun. 2018, pp. 5457–5466. doi: 10.1109/CVPR.2018.00572.

[8] B. Luo, Y. Fang, H. Wang, and D. Zang, "Reservoir inflow prediction using a hybrid model based on deep learning," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 715, no. 1, p. 012044, Jan. 2020, doi: 10.1088/1757-899X/715/1/012044.

[9] S. R. Bin Shah, G. S. Chadha, A. Schwung, and S. X. Ding, "A Sequence-to-Sequence Approach for Remaining Useful Lifetime Estimation Using Attention-augmented Bidirectional LSTM," *Intell. Syst. with Appl.*, vol. 10–11, p. 200049, Jul. 2021, doi: 10.1016/j.iswa.2021.200049.

[10] F. Pan *et al.*, "Stacked-GRU Based Power System Transient Stability Assessment Method," *Algorithms*, vol. 11, no. 8, p. 121, Aug. 2018, doi: 10.3390/a11080121.

[11] S. Dargan, M. Kumar, M. R. Ayyagari, and G. Kumar, "A Survey of Deep Learning and Its Applications: A New Paradigm to Machine Learning," *Arch. Comput. Methods Eng.*, vol. 27, no. 4, pp. 1071–1092, Sep. 2020, doi: 10.1007/s11831-019-09344-w.

[12] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Comput. Sci.*, vol. 2, no. 6, p. 420, Nov. 2021, doi: 10.1007/s42979-021-00815-1.

[13] Q. Zhang, L. T. Yang, Z. Chen, and P. Li, "A survey on deep learning for big data," *Inf. Fusion*, vol. 42, pp. 146–157, Jul. 2018, doi: 10.1016/j.inffus.2017.10.006.

[14] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, p. 100379, May 2021, doi: 10.1016/j.cosrev.2021.100379.

[15] A. Fischer and C. Igel, "An Introduction to Restricted Boltzmann Machines," 2012, pp. 14–36. doi: 10.1007/978-3-642-33275-3_2.

[16] Yuming Hua, Junhai Guo, and Hua Zhao, "Deep Belief Networks and deep learning," in *Proceedings of 2015 International Conference on Intelligent Computing and Internet of Things*, IEEE, Jan. 2015, pp. 1–4. doi: 10.1109/ICAIOT.2015.7111524.

[17] I. Goodfellow *et al.*, "Generative adversarial networks," *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020, doi: 10.1145/3422622.

[18] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, Jan. 2018, doi: 10.1109/MSP.2017.2765202.

[19] R. S. Sutton and A. Barto, "Reinforcement learning: an introduction, Nachdruck. in Adaptive computation and machine learning," 2014.

[20] J. Naskath, G. Sivakamasundari, and A. A. S. Begum, "A Study on Different Deep Learning Algorithms Used in Deep Neural Nets: MLP SOM and DBN," *Wirel. Pers. Commun.*, vol. 128, no. 4, pp. 2913–2936, Feb. 2023, doi: 10.1007/s11277-022-10079-4.

[21] S. Pouyanfar *et al.*, "A Survey on Deep Learning," *ACM Comput. Surv.*, vol. 51, no. 5, pp. 1–36, Sep. 2019, doi: 10.1145/3234150.

[22] E. Gawehn, J. A. Hiss, and G. Schneider, "Deep Learning in Drug Discovery," *Mol. Inform.*, vol. 35, no. 1, pp. 3–14, Jan. 2016, doi: 10.1002/minf.201501008.

[23] A. Shrestha and A. Mahmood, "Review of Deep Learning Algorithms and Architectures," *IEEE Access*, vol. 7,

pp. 53040–53065, 2019, doi: 10.1109/ACCESS.2019.2912200.

BIOGRAPHIES OF AUTHORS

Nisha. C.M, M. Sc, NET Assistant Professor in Computer Application at CHMKM Govt.Arts & Science CollegeTanur, Malappuram(dt), Kerala, India. She received a B. Sc in Mathematics and an M. Sc in Computer Science from Calicut University. She qualified for the UGC-NET examination in June 2012. Currently, she is a Part-time Ph. D research scholar at Karpagam Academy of Higher Education, Coimbatore, Tamilnadu. Her areas of interest are deep learning, Machine learning, and IoT. She can be contacted at email: nishacm2008@gmail.com

Thangarasu Nainan, MCA, M. Phil, Ph. D Assistant Professor, Department of Computer Science, Karpagam academy of Higher Education, Coimbatore-21, Tamilnadu. He received his bachelor's degree in computer applications from Periyar University and a master's degree in computer applications from Hindusthan University. He was awarded a Ph. D in Computer Science from Bharathiyar University, Coimbatore in 2019. He has published articles in the National and International Indexed journals, including SCI, WoS, and SCOPUS. His areas of interest are cyber security, Machine learning, and deep learning. He can be contacted at email: drthangarasu.n@kahedu.edu.in