Deep Learning Techniques: An Overview

Amitha Mathew[1], P.Amudha[2] and S.Sivakumari[3]

Department of Computer Science and Engineering, School of Engineering,
Avinashilingam Institute for Home Science and Higher Education for Women,
Coimbatore, India
[1]amitha.ann@gmail.com,[2]amudharul@gmail.com,[3]sivakumari_cse@avinuty.ac.in

Abstract. Deep Learning is a class of machine learning which performs much better on unstructured data. Deep learning techniques are outperforming current machine learning techniques. It enables computational models to learn features progressively from data at multiple levels. The popularity of deep learning amplified as the amount of data available increased as well as the advancement of hardware that provides powerful computers. This article comprises of the evolution of deep learning, various approaches to deep learning, architectures of deep learning, methods, and applications.

Keywords: Deep Learning (DL), Recurrent Neural Network (RNN), Deep Belief Networks (DBN), Convolutional Neural Networks(CNN), Generative Adversarial Networks(GAN)

1 Introduction

Deep learning techniques which implement deep neural networks became popular due to the increase of high-performance computing facility. Deep learning achieves higher power and flexibility due to its ability to process a large number of features when it deals with unstructured data. Deep learning algorithm passes the data through several layers; each layer is capable of extracting features progressively and passes it to the next layer. Initial layers extract low-level features, and succeeding layers combines features to form a complete representation. Section 2 gives an overview of the evolution of deep learning models. Section 3 provides a brief idea about the different learning approaches, such as supervised learning, unsupervised learning, and hybrid learning. Supervised learning uses labeled data to train the neural network. In supervised learning, the network uses unlabeled data and learns the recurring patterns. Hybrid learning combines supervised and unsupervised methods to get a better result. Deep learning can be implemented using different architectures such as architectures like Unsupervised Pre-trained Networks, Convolutional Neural Networks, Recurrent Neural Networks, and Recursive Neural Networks, which are described in section 4. Section 5 introduces various training methods and optimization techniques that help in achieving better results. Section 6 describes the frameworks which allow us to develop tools that offer a better programming environment. Despite the various challenges in deep learning applications, many exciting applications that may rule the world are briefed in Section 7.

2 Evolution of Deep Learning

First Generation of Artificial Neural networks(ANN) was composed of perceptrons in neural layers, which were limited in computations. The second-generation calculated the error rate and backpropagated the error. Restricted Boltzmann machine overcame the limitation of backpropagation, which made the learning easier. Then other networks are evolved eventually [15,24]. Figure.1 illustrates a timeline showing the evolution of deep models along with the traditional model. The performance of classifiers using deep learning improves on a large scale with an increased quantity of data when compared to traditional learning methods. Figure.2 depicts the performance of traditional machine learning algorithms and deep learning algorithms [6]. The performance of traditional machine learning algorithms becomes stable when it reaches the threshold of training data whereas the deep learning upturns it's performance with increased amount of data. Now a days deep learning is used in a lot many applications such as Google's voice and image recognition, Netflix and Amazon's recommendation engines, Apple's Siri, automatic email and text replies, chatbots etc.

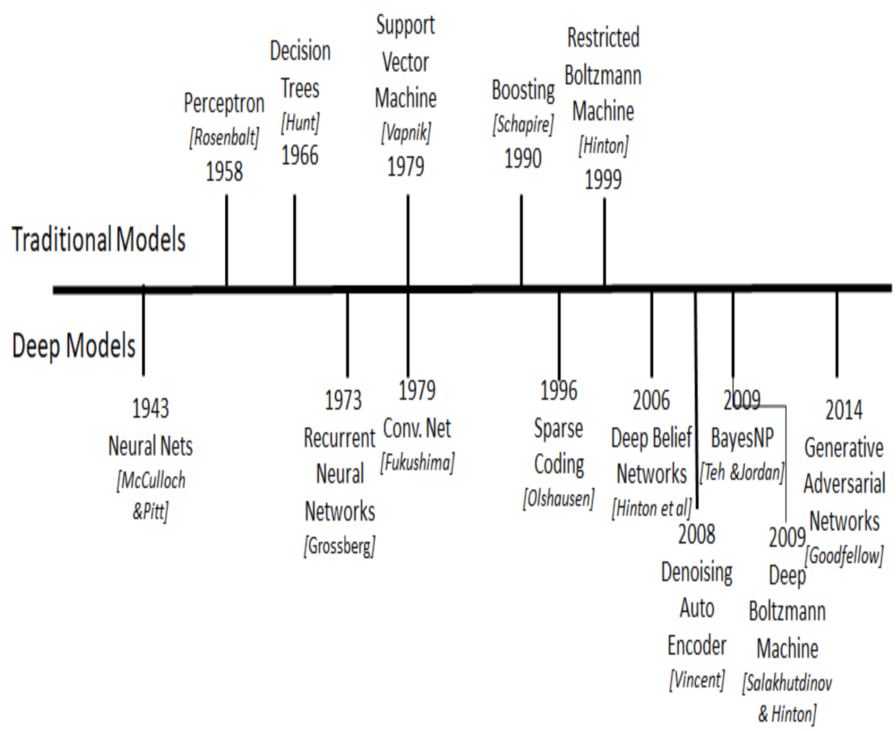

Fig. 1: Evolution of Deep Models

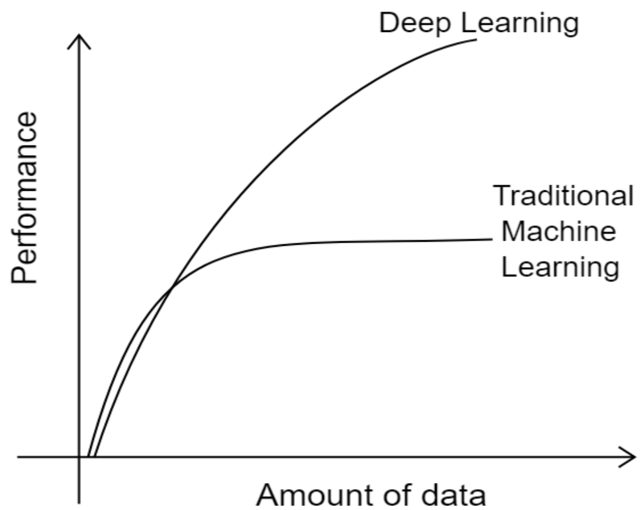

Fig. 2: Why Deep Learning ?

3 Deep Learning Approaches

Deep neural networks are successful in Supervised learning, Unsupervised learning, Reinforcement learning, as well as hybrid learning.

3.1 Supervised Learning

In supervised learning, the input variables represented as X are mapped to output variables represented as Y by using an algorithm to learn the mapping function f.

$$Y = f(X) \tag{1}$$

The aim of the learning algorithm is to approximate the mapping function to predict the output (Y) for a new input (X). The error from the predictions made during training can be used to correct the output. Learning can be stopped when all the inputs are trained to get the targeted output [11]. Regression for solving regression problems [18], Support Vector machines used for classification [21]], Random forest for classification as well as regression problems [20].

3.2 Unsupervised Learning

In unsupervised learning, we have the input data only and no corresponding output to map. This learning aims to learn about data by modeling the distribution in data. Algorithms can be able to discover the exciting structure present in the data. Clustering problems and association problems use Unsupervised learning. The unsupervised learning algorithms such as K-means algorithm is used in clustering problems [9], Apriori algorithm is used in association problems [10]

3.3 Reinforcement Learning

Reinforcement learning uses a system of reward and punishment to train the algorithm. In this, the algorithm or an agent learns from its environment. The agent gets rewards for correct performance and penalty for incorrect performance. For example, consider the case of a self-driving car, the agent gets a reward for driving safely to destination and penalty for going off-road. Similarly, in the case of a program for playing chess, the reward state may be winning the game and the penalty for being checkmated. The agent tries to maximize the reward and minimize the penalty. In reinforcement learning, the algorithm is not told how to perform the learning; however, it works through the problem on its own [16].

3.4 Hybrid Learning

Hybrid learning refers to architectures that make use of generative (unsupervised) as well as discriminative (supervised) components. The combination of different architectures can be used to design a hybrid deep neural network. They are used for action recognition of humans using action bank features and are expected to produce much better results [3].

4 Fundamental deep learning architectures

Deep learning architectures perform better than simple ANN, even though training time of deep structures are higher than ANN. However, training time can be reduced using methods such as transfer learning, GPU computing. One of the factors which decide the success of neural networks lies in the careful design of network architecture. Some of the relevant deep learning architectures are discussed below.

4.1 Unsupervised Pre-trained Networks

In unsupervised pre-training, a model is trained unsupervised, and then the model used for prediction. Some unsupervised pre-training architectures are discussed below [4].

Autoencoders : are used for the reduction of the dimension of data, novelty detection problems, as well as in anomaly detection problems. In an autoencoder, the first layer is built as an encoding layer and transpose of that as a decoder. Then train it to recreate the input using the unsupervised method. After training, fix the weights of that layer. Then move to the subsequent layer until we pre-train all the layers of deep net. Then go back to the original problem that we want to solve with deep net (Classification/Regression) and optimize it with Stochastic gradient descent by starting from weights learned using pre-training.

Autoencoder network consists of two parts [7]. The input is translated to a latent space representation by the encoder, which can be denoted as:

$$h = f(x) \tag{2}$$

The input is reconstructed from the latent space representation by the decoder, which can be denoted as:

$$r = g(h) \tag{3}$$

In essence, autoencoders can be described as in equation (4). r is the decoded output which will be similar to input x :

$$g(f(x)) = r \tag{4}$$

Deep Belief Networks: The first step for training the deep belief network is to learn features using the first layer. Then use the activation of trained features in the next layer. Continue this until the final layer. Restricted Boltzmann Machines (RBM) is used to train layers of the Deep Belief Networks (DBNs), and the feed-forward network is used for fine-tuning. DBN learns hidden pattern globally, unlike other deep nets where each layer learns complex patterns progressively [19].

Generative Adversarial Networks: Generative Adversarial Networks (GAN) were presented by Ian Goodfellow.It comprises of Generator network and Discriminator network. Generator generates the content while the discriminator validates the generated content. Generator creates natural-looking images, while the discriminator decides whether the image looks natural. GAN is considered as a minimax two-player algorithm. GANs uses convolutional and feed-forward Neural Nets [5].

4.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) are used mainly for images. It assigns weights and biases to various objects in the image and differentiates one from the other. It requires less preprocessing related to other classification algorithms. CNN uses relevant filters to capture the spatial and temporal dependencies in an image [12,25]. The different CNN architectures include LeNet, AlexNet, VGGNet, GoogleNet, ResNet, ZFNet. CNN's are mainly used in applications such as Object Detection, Semantic Segmentation, Captioning.

4.3 Recurrent Neural Networks

In recurrent neural networks (RNN), outputs from the preceding states are fed as input to the current state. The hidden layers in RNN can remember information. The hidden state is updated based on the output generated in the previous state. RNN can be used for time series prediction because it can remember previous inputs also, which is called Long-Short Term Memory [2].

5 Deep learning methods

Some of the powerful techniques that can be applied to deep learning algorithms to reduce the training time and to optimize the model are discussed in the following section. The merits and demerits of each method are comprised in the Table 1

Back propagation : While solving an optimization problem using a gradient-based method, backpropagation can be used to calculate the gradient of the function for each iteration [18].

Stochastic Gradient Descent : Using the convex function in gradient descent algorithms ensures finding an optimal minimum without getting trapped in a local minimum. Depending upon the values of the function and learning rate or step size, it may arrive at the optimum value in different paths and manners [14].

Learning Rate Decay : Adjusting the learning rate increases the performance and reduces the training time of stochastic gradient descent algorithms. The widely used technique is to reduce the learning rate gradually, in which we can make large changes at the beginning and then reduce the learning rate gradually in the training process. This allows fine-tuning the weights in the later stages [7].

Dropout : The overfitting problem in deep neural networks can be addressed using the drop out technique. This method is applied by randomly dropping units and their connections during training [9]. Dropout offers an effective regularization method to reduce overfitting and improve generalization error. Dropout gives an improved performance on supervised learning tasks in computer vision, computational biology, document classification, speech recognition [1].

Max-Pooling: In max-pooling a filter is predefined, and this filter is then applied across the nonoverlapping sub-regions of the input taking the max of the values contained in the window as the output. Dimensionality, as well as the computational cost to learn several parameters, can be reduced using max-pooling [23].

Batch Normalization: Batch normalization reduces covariate shift, thereby accelerating deep neural network. It normalizes the inputs to a layer, for each mini-batch, when the weights are updated during the training. Normalization stabilizes learning and reduces the training epochs. The stability of a neural network can be increased by normalizing the output from the previous activation layer [8].

Skip-gram : Word embedding algorithms can be modeled using Skip-gram. In the skip-gram model, two vocabulary terms share a similar context; then those terms are identical. For example, the sentences "cats are mammals" and "dogs are mammals" are meaningful sentences which shares the same meaning "are mammals." Skip-gram can be implemented by considering a context window containing n terms and train the neural network by skipping one of this term and then use the model to predict skipped term [13].

Transfer learning: In transfer learning, a model trained on a particular task is exploited on another related task. The knowledge obtained while solving a particular problem can be transferred to another network, which is to be trained on a related problem. This allows for rapid progress and enhanced performance while solving the second problem [17].

Table 1: Comparison of Deep learning methods

Method	**Description**	**Merits**	**Demerits**
Back propagation	Used in Optimization problem	For calculation of gradient	Sensitive to noisy data
Stochastic Gradient Descent	To find optimal minimum in optimization problems	Avoids trapping in local minimum	Longer convergence time, computationally expensive
Learning Rate Decay	Reduce learning rate gradually	Increases performance, Reduces training time	Computationally expensive
Dropout	Dropsout units/ connection during training	Avoids overfitting	Increases number of iterations required to converge
Max-Pooling	Applies a max filter	Reduces dimension and computational cost	Considers only the maximum element which may lead to unacceptable result in some cases
Batch Normalization	Batch-wise normalization of input to a layer	Reduces covariant shift, Increases stability of the network, Network trains faster, Allows higher learning rates	Computational overhead during training
Skip-gram	Used in word embedding algorithms	Can work on any raw text, Requires less memory	Softmax function is computationally expensive, Training Time is high
Transfer learning	Knowledge of first model is transferred to second problem	Enhances performance, Rapid progress in training of second problem	Works with similar problems only

6 Deep learning frameworks

A deep learning framework helps in modeling a network more rapidly without going into details of underlying algorithms. Each framework is built for different purposes differently. Some deep learning frameworks are discussed below and are summarized in Table 2.

TensorFlow TensorFlow, developed by Google brain, supports languages such as Python, C++and R. It enables us to deploy our deep learning models in CPUs as well as GPUs [22].

Keras Keras is an API, written in Python and run on top of TensorFlow. It enables fast experimentation. It supports both CNNs and RNNs and runs on CPUs and GPUs [22].

PyTorch PyTorch can be used for building deep neural networks as well as executing tensor computations. PyTorch is a Python-based package that provides Tensor computations. PyTorch delivers a framework to create computational graphs [22].

Caffe Yangqing Jia developed Caffe, and it is open source as well. Caffe stands out from other frameworks in its speed of processing as well as learning from images. Caffe Model Zoo framework facilitates us to access pre-trained models, which enable us to solve various problems effortlessly [22].

Deeplearning4j Deeplearnig4j is implemented in Java, and hence, it is more efficient when compared to Python. The ND4J tensor library used by Deeplearning4j provides the capability to work with multi-dimensional arrays or tensors. This framework supports CPUs and GPUs. Deeplearnig4j works with images, csv as well as plaintext [22].

Table 2: Comparison of Deep Learning Frameworks

Deep Learning Framework	**Release Year**	**Language written in**	**CUDA supported**	**Pre-trained models**
TensorFlow	2015	C++, Python	Yes	Yes
Keras	2015	Python	Yes	Yes
PyTorch	2016	Python, C	Yes	Yes
Caffe	2013	C++	Yes	Yes
Deeplearning4j	2014	C++, Java	Yes	Yes

7 Applications of Deep Learning

Deep learning networks can be used in a variety of applications such as self-driving cars, Natural Language Processing, Google's Virtual Assistant, Visual Recognition, Fraud detection, healthcare, detecting developmental delay in children, adding sound to silent movies, automatic machine translation, text to image translation, image to image synthesis, automatic image recognition, Image colorization, earthquake prediction, market-rate forecasting, news aggregation and fraud news detection.

8 Conclusion

Deep learning is continuously evolving faster; still, there are a number of problems to deal with and can be solved using deep learning. Even though a full understanding of the working of deep learning is still a mystery, we can make machines smarter using Deep learning, sometimes even smarter than human. Now the aim is to develop deep learning models that work with mobile to make the applications smarter and more intelligent. Let deep learning be more devoted to the betterment of humanity and thus making our domain a better place to live.

References

1. Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2897–2905, 2018. doi: 10.1109/TPAMI.2017.2784440.
2. Filippo Maria Bianchi, Enrico Maiorino, Michael C Kampffmeyer, Antonello Rizzi, and Robert Jenssen. An overview and comparative analysis of recurrent neural networks for short term load forecasting. *arXiv preprint arXiv:1705.04378*, 2017.
3. Li Deng, Dong Yu, et al. Deep learning: methods and applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014. doi: 10.1007/978-981-13-3459-7_3.
4. Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.
5. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
6. Palash Goyal, Sumit Pandey, and Karan Jain. Introduction to natural language processing and deep learning. In *Deep Learning for Natural Language Processing*, pages 1–74. Springer, 2018. doi: 10.1007/978-1-4842-3685-7_1.
7. Nathan Hubens. Deep inside: Autoencoders - towards data science, Apr 2018.
8. Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

9. Anil K Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010. doi: 10.1016/j.patrec.2009.09.011.
10. Sotiris Kotsiantis and Dimitris Kanellopoulos. Association rules mining: A recent overview. *GESTS International Transactions on Computer Science and Engineering*, 32(1):71–82, 2006.
11. Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24, 2007.
12. Quoc V Le et al. A tutorial on deep learning part 2: Autoencoders, convolutional neural networks and recurrent neural networks. *Google Brain*, pages 1–20, 2015.
13. Chaochun Liu, Yaliang Li, Hongliang Fei, and Ping Li. Deep skip-gram networks for text classification. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 145–153. SIAM, 2019.
14. Jonathan Lorraine and David Duvenaud. Stochastic hyperparameter optimization through hypernetworks. *arXiv preprint arXiv:1802.09419*, 2018.
15. Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. Evolving deep neural networks. In *Artificial Intelligence in the Age of Neural Networks and Brain Computing*, pages 293–312. Elsevier, 2019. doi: 10.1016/B978-0-12-815480-9.00015-3.
16. Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.
17. Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. doi: 10.1109/TKDE.2009.191.
18. Abhishek Panigrahi, Yueru Chen, and C-C Jay Kuo. Analysis on gradient propagation in batch normalized residual networks. *arXiv preprint arXiv:1812.00342*, 2018.
19. Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009. doi: 10.1016/j.ijar.2008.11.006.
20. Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2001.
21. George AF Seber and Alan J Lee. *Linear regression analysis*, volume 329. John Wiley & Sons, 2012.
22. Pulkit Sharma. Top 5 deep learning frameworks, their applications, and comparisons!, May 2019.
23. Toshihiro Takahashi. Statistical max pooling with deep learning, July 3 2018. US Patent 10,013,644.
24. Bhiksha Wang, HaohanandRaj. On the origin of deep learning. *arXiv preprint arXiv:1702.07800*, 2017.
25. Rikiya Yamashita, Mizuho Nishio, Richard Kinh Gian Do, and Kaori Togashi. Convolutional neural networks: an overview and application in radiology. *Insights into imaging*, 9(4):611–629, 2018. doi: 10.1007/s13244-018-0639-9.