Development Process and Application Models of Deep Learning

Anni Kang[1], Jiangang Chen[2]
[1]Hamden Hall Country Day School, Hamden, Connecticut, 06517, USA
[2]Shenzhen Institute of Information Technology, Shenzhen, Guangdong, 518172, China

Abstract: Deep learning is a technique and method of machine learning. Since its introduction, deep learning has achieved remarkable results in application. This paper first reviews the development of deep learning chronologically, then introduces basic application models in deep learning such as FNN, CNN, RNN, GAN, DRL, compares some commonly used deep learning development frameworks, and finally, it analyzes the shortcomings existed and the future study trend regarding deep learning.

Keywords: DL; FNN; CNN; RNN; GAN; DRL

1. Introduction

Deep learning (DL) is a technology and method of machine learning, and it is a new research hotspot in the field of machine learning. DL was proposed by Geoffrey E. Hinton from the University of Toronto in 2006[1]. The research has experienced a labyrinthian process. So far, deep learning has met the needs of dealing with effectiveness, performance, and intelligence in the big data era. It has demonstrated excellent information processing capabilities that go beyond traditional methods in large-vocabulary speech recognition, image and video recognition, computer vision, natural language processing, and target detection.

DL aims to build neural network to simulate human brain for analytical analysis and learning. Deep neural network (DNN) is superimposed by multiple single-layer nonlinear networks. Through nonlinear transformations, dimensions reduction, combining low-level features, the deep neural networks can form an abstract, easy-to-differentiate high-level representation and achieve end-to-end learning [2-3]. DL can obtain state information in the current environment and has strong perceptual ability [4].

Therefore, understanding DL is of great significance for promoting the development of related fields such as artificial intelligence and robotics. This paper first introduces the development process of DL, then explains the deep learning model, compares the commonly used development framework such as Theano, Caffe, Torch, TensorFlow, finally analyzes the shortcomings and future researches trend regarding DL.

2. Development Process of DL

2.1. Introduction of BP algorithm and formal formation of CNN

In 1986, David E. Rumelhart implemented a gradient descent method for finding weights that minimized the sum squared error of the system's performance. The major theoretical contribution of this work was a procedure called error propagation [5].

In 1989, Yann LeCun first proposed using the back propagation (BP) to train multi-layer convolutional neural networks to recognize the handwritten zip code digits provided by the U.S. Postal Service [6]. The LeNet-5 model he proposed in 1998 marked the formal formation of Convolutional Neural Networks (CNN), and multilayer neural networks trained with the BP algorithm constituted the best example of a successful gradient-based learning technique [7].

However, with the increase of the quantity of hidden layers, the traditional BP algorithm faced problems of gradient diffusion and huge computing quantity. Since the hardware could not support a large number of computing at a time, CNN was replaced by a non-neural network algorithm.

2.2. Proposal of deep learning

In 2006, Geoffrey E. Hinton derived a fast and unsupervised algorithm to learn deep directed belief networks. The algorithm was used to initialize a slower learning procedure that fine-tuned the weights using a contrastive version of the wake-sleep algorithm. The algorithm could find a fairly good set of parameters quickly, even in deep networks with millions of parameters and hidden layers [1].

Since then, unsupervised initial learning methods and deep learning frameworks have been extensively studied by researchers in the field of machine learning and artificial intelligence.

2.3. Breakthrough in LVSR and image classification

In 2012, The CD-DNN-HMMs for large-vocabulary speech recognition (LVSR) outperformed the conventional model, which was the first industry success of DL [8]; Alex Krizhevsky applied CNN to image processing for the first time and won the 1st place for the 2012 ImageNet LSVRC [9]. After that, DeepFace and DeepID enabled CNN to achieve landmark research results in the field of face recognition[10-13]; Google Brain team built high-level class-specific feature detectors from only unlabeled data by training a deep sparse auto encoder[14].

2.4. Proposal of ResNet

In 2015, He Kaiming used the Residual Neural Network (ResNet) to win the 1st place of the ILSVRC with a 3.57% top-5 error rate, significantly surpassing the performance level of the previous year's competition, while the parameters were less than VGG Net [15].

2.5. AlphaGo

In 2016, AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. AlphaGo, the DNN, is trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play, and a new search algorithm that combines Monte Carlo simulation with value and policy networks [16].

3. Basic NN Model

3.1. FNNs

direction, from the input layer, through the hidden layer, and to output the layer unidirectionally (fig. 1). FNNs can be regarded as one function. Complex mapping from input space to output space is realized through multiple recombination of simple nonlinear function[17].

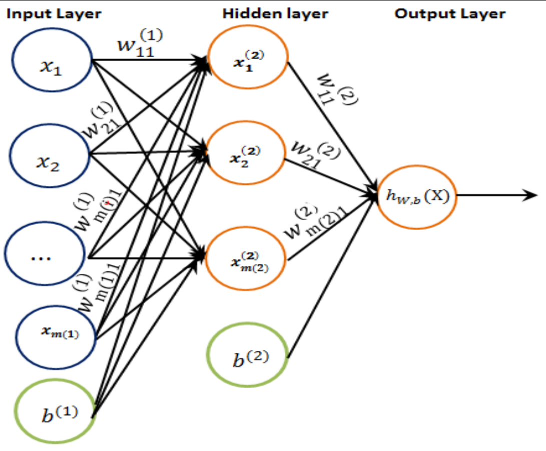


Figure 1. Feedforward Neural Network. $m(l)$ **expresses the quantity of nodes in the** l **layer.** $f^l(\bullet)$ **expresses the activation function in the** l **layer.** $W^{(l)} \in R_{m(l)\times m(l-1)}$ **expresses the weight matrix of all connections from the** $l-1$ **layer to the** l **layer.** $b^{(l)} \in R_{m(l)\times 1}$ **expresses the bias from the** $l-1$ **layer to the** l **layer.** $z^{(l)} \in R_{m(l)\times 1}$ **expresses the net input vector of nodes in the** l **layer.** $x^{(i)} \in R_{m(i)\times 1}$ **expresses the activation vector of the nodes in the** i **layer. When** $i=1, x^{(1)} = x = \left(x_1, x_2, \cdots, x_{m(1)}\right)$ **is the value vector of nodes in the input layer.**

$$x_i^{(2)} = f^2\left(w_{1i}^{(1)}x_1 + w_{2i}^{(1)}x_2 + \cdots + w_{m(1)i}^{(1)}x_{m(1)} + b^{(1)}\right).$$

$$h_{W,b}(x) = f^3\left(w_{11}^{(2)}x_1^{(2)} + \cdots + w_{m(2)1}^{(2)}x_{m(2)}^{(2)} + b^{(2)}\right).$$

BP algorithm is often used to calculate the efficient gradient in neural network training. Error-based BP algorithm is:

Algorithm: BP algorithm based on random gradient descent.

Input: training set $D=\left\{\left(X^{(n)}, y^{(n)}\right)\right\}, n=1,\cdots N$, verification set V ,learning rate α , regularization coefficient λ ,the number of network layers L ,the quantity of nodes in the l layer $m(l), 1\le l\le L$.

```
1   Random initialization $W^{(l)}, b^{(l)}$ .
2   repeat
3       Random reordering of sample in training set $D$ ;
4       for $n = 1\cdots N$ do
5           Select sample $\left(X^{(n)}, y^{(n)}\right)$ from training set $D$ ;
6           Feedforward calculates net input $z^{(l)}$ and activation $X^{(l)}$ of each layer;
7           Backpropagation calculates the error $\delta^{(l)}$ of each layer;
8           $W^{(l)} \leftarrow W^{(l)} - \alpha(\delta^{(l)}(X^{(l-1)})^T + \lambda W^{(l)})$ ;
9           $b^{(l)} \leftarrow b^{(l)} - \alpha\delta^{(l)}$ ;
10      end
```

11 Until the error rate of neural network model in the verification set V does not descend any more;
Output W, b

3.2. CNNs

CNNs, a class of deep FNNs, are constituted of convolutional, pooling, and fully connected layers. The convolutional layers are inspired by the local Receptive Field. They extract features through several convolution kernels and pass the result to the next layer. Pooling layers combine the outputs of neuron
ake Tahoe: MIT Press, 2013.

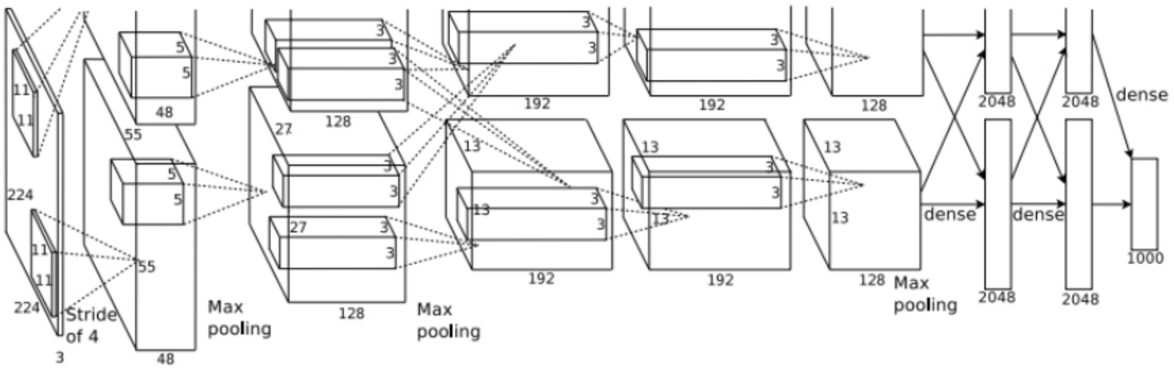


Figure 2. Typical CNN architecture (Image from reference 9)

The “levels” of features can be enriched by the number of stacked layers. But with the network depth increasing, accuracy gets saturated (which might be unsurprising) and then degrades rapidly. ResNets address the degradation problem: Letting stacked layers fit of $F(X) = h(X) - X$. The original mapping is recast into $F(X) + X$. The formulation of $F(X) + X$ can be realized by feedforward neural networks with “shortcut connections”. Shortcut connections are those skipping one or more layers. The shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers (Fig. 3). Identity shortcut connections add neither extra parameter nor computational complexity[15].

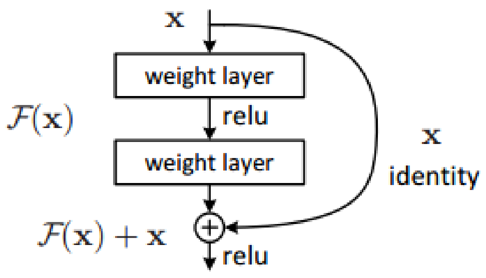


Figure 3. Residual learning: a building block.

Shortcut connections skip directly one or more layers to protect the completeness of information, and RESNETs ease the training of networks.

3.3. RNN & LSTM

In the above-mentioned NN, the information of each layer can only be propagated to the next layer. Handling sample is mutually independent at each moment, so the application of time sequence such as voice and text message can not be applied. Recurrent Neural Networks (RNNs) resolve the question through feedback connections added in hidden layer, which makes RNN store the states of all previous moments (Fig. 4). This endows RNNs with memory function. These features make RNN ideal for modeling time-series signals [18-19].

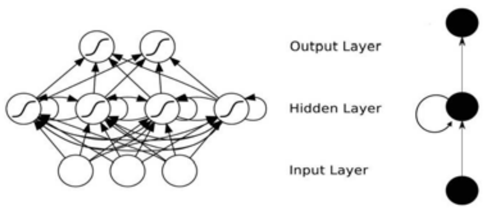
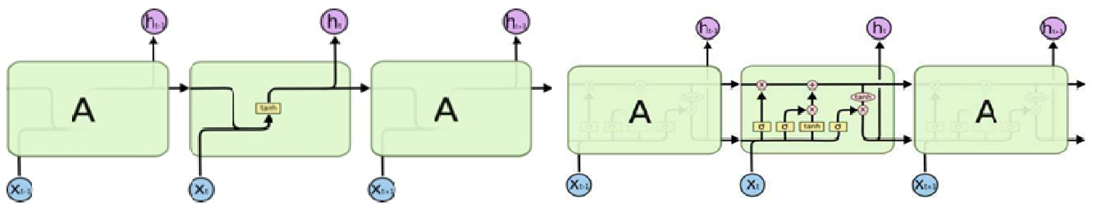


Figure 4. RNN structure diagram

The hidden layer of RNN only has one state, tanh state. Tanh state is sensitive to short-term input, but it is hard to store information for long term due to gradient vanishing.

Long short-term memory (LSTM) is a deep learning system where one cell state, storing values over arbitrary time intervals, is added into the hidden layer in RNN besides the tanh state. LSTM includes input gate, forget gate, and output gate regulating the flow of information into and out of the cell. The values meeting rules in the cell are kept. The values not meeting rules are discarded through forget gate (fig. 5). LSTM avoids problems of the gradient vanishing and explosion through saving long-term state in the cell [20].



RNN LSTM

Figure 5. Architecture of RNN and LSTM

4. Application Model of DL

Two application models GAN and DRL are developed on the above-mentioned basic NN model.

4.1. GAN

Generative Adversarial Network (GAN), put forwarded by google brain scientist Ian Goodfellow in 2014, is one kind of new deep learning framework implemented by a system of two NNs contesting with each other. One of the two is the Generator G used to generate data as a fake sample set; The other one, Discriminator D, is used to evaluate the probability of that sample is true instead of generated by G[21-22].

Fig. 6 shows GAN structure and operational principle. The Generator G receives random noise to generate figure. The Discriminator D judges whether the input figure x is true or not. Output represents that it is the probability of a true figure. Through alternate iterative training of these two networks, after the false sample generated by the Generator G enters the Discriminator D, the result of differentiating network is close to 0.5 to reach the effect of mixing the spurious with the genuine.

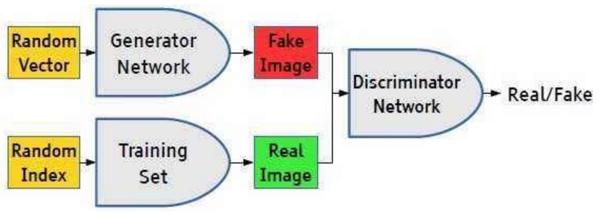


Figure 6. GAN structure and operational principle

GAN come in many variants. There are SGAN, WGAN, CGAN, DCGAN, InfoGAN, StackGAN etc, which can all be applied to solve actual problem in various scenes.

4.2. Deep Reinforcement Learning (DRL)

Comparing with supervised learning and unsupervised learning, Reinforcement Learning (RL) is another branch of machine learning. RL learns constantly according to the rewards or punishments obtained from interaction with the environment. The normal form of RL is very similar to the process that human being learns knowledge and is regarded as an important way to actualize universal AI.

Reinforcement Learning includes four elements: state, action, transition probability, and reward function. They are described by Markov decision process generally. The interaction is shown in fig. 7. Agent chooses one action to work on the environment. After environment accepts the action, state changes and one reinforced signal (reward or punishment) is generated and back to Agent at the same time. Agent chooses next action according to reinforced signal and current state of environment [23].

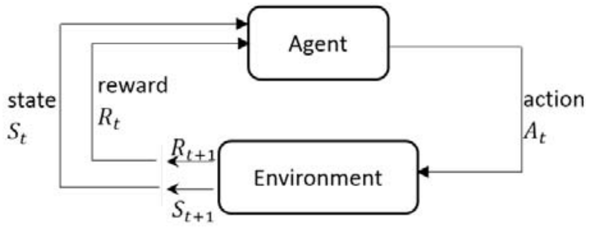


Figure 7. RL structure and operational principle

DeepMind adopted deep learning network to fit Q function in Reinforcement learning in 2013 and put forward Deep Q-learning (DQN) [24]. DQN is suitable for low-dimension and discrete action space. The improvement of DQN includes Double DQN, Dueling Network, Prioritized Replay and Rainbow. There is another one kind of DRL algorithm such as Actor-Critic, DDPG, A3C and PPO etc. These algorithms are suitable for the continuous action space.

5. Deep Learning Development Framework

Up till now, the development platforms of deep learning put forward by universities, scientific research institutes, and large companies have reached more than 40 kinds. Table 1 compares deep learning development framework used generally.

Table 1. Deep learning development framework used generally

	Theano	Caffe	Torch	TensorFlow	CNTK
Developers	University of Montreal	Berkeley Vision and LearningCenter	Facebook	Google Brain Team	Microsoft
Lower-Layer Development Language	Python	C++	LuaJIT/C++/CUDA	C++	C++
Interface language	Python	command line/Python/ Matlab/C++	Lua	Python/Java/Go	Python/C/C++ /command line
Running Mode	GPU/CPU	CPU or GPU	CPU/GPU	CPU or GPU	CPU or GPU
OS	Windows/Linux/Mac	Windows/Linux/Mac/ Mobile devices(Android, iOS)	Windows or Linux	Windows/Linux/Mac / Mobile devices(Android, iOS)	Windows/Linux/Mac
Models Supported	CNN/RNN	CNN/AlexNet/VGG/Inception/ResNet	CNN/RNN	CNN/RNN	MLP/CNN/RNN/LSTM/Sequence-to-Sequence

6. Ending Statement

DL has multilayer nonlinear mapping structure and can complete complex function approximation. Moreover, Once a model is trained, this model can be transplanted just by modifying the parameters. In addition, DL integrates feature learning with predictive learning into one model to realize end-to-end learning, which avoids the separated disadvantage of featured extraction and predictive model learning.

However, how to increase training speed under the guaranteeing of a certain training precision is one of the subjects in DL. Besides, the nature of DL belongs to “black box”. The theory about it is not complete, so exploring theoretical support of DL and combining complex reasoning with Deep learning system will be the study area for future. Moreover, in real life, most things are unlabeled. How to handle large-scale and unlabeled data is the problem that should be solved by DL.

References

[1] HINTON G E, OSINDERO S, TEH Y W. A fast learning algorithm for deep belief nets [J]. Neural Computation, 2006, 18(7): 1527-1554.

[2] BENGIO Y. Learning deep architectures for AI [J]. Foundations and Trends in Machine Learning, 2009, 2(1): 1-127.

[3] BENGIO Y, DELALLEAU O. On the expressive power of deep architectures [C] // Proc of the 14th International Conference on Discovery Science. Berlin : Springer-Verlag, 2011: 18-36.

[4] ZHAO Dong-bin, SHAO Kun, ZHU Yuan-heng, LI Dong, CHEN Ya-ran, WANG Hai-tao. Review of deep reinforcement learning and discussions on the development of computer Go [J]. Control Theory & Applications, 2016, 33(6): 701-717.

[5] D. E. Rumelhart, G. E. Hinton, R. J. Williams. Learning Internal Representations by Error Propagation [C] //Parallel distributed processing: explorations in the microstructure of cognition. Cambridge:MIT Press, 1986:318-362.

[6] Y. LeCun, B. Boser, J. S. Denker. Backpropagation Applied to Handwritten Zip Code Recognition [J]. Neural Computation, 1989, 1(4): 541-551.

[7] Y. Lecun, L. Bottou, Y. Bengio, P. Haffner. Gradient-based learning applied to document recognition [J]. Proceedings of the IEEE Conference, 1998, 86(11): 2278-2324.

[8] DAHL G E, YU D, DENG L, et al. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition [J]. IEEE Transactions on Audio, Speech, and Language Processing, 2012, 20(1): 30-42.

[9] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks [C] //Advances in Neural Information Processing Systems. Lake Tahoe: MIT Press, 2012: 1097-1105.

[10] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, Lior Wolf. DeepFace: Closing the Gap to Human-Level Performance in Face Verification [C] //CVPR ’14 Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition. IEEE Computer Society, 2014: 1701-1708.

[11] Yi Sun, Xiaogang Wang, Xiaoou Tang. Deep Learning Face Representation from Predicting 10,000 Classes [J]. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014: 1891-1898.

[12] Yi Sun, Yuheng Chen, Xiaogang Wang, Xiaoou Tang. Deep Learning Face Representation by Joint Identification-Verification [J]. Advances in Neural Information Processing Systems 27 (NIPS 2014), 2014.

[13] Yi Sun, Xiaogang Wang, Xiaoou Tang. Deeply Learned Face Representations Are Sparse, Selective, and Robust [J]. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015: 2892-2900.

[14] LE Q V. Building high-level features using large scale unsupervised learning [C] //Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing. Vancouver: IEEE, 2013: 8595-8598.

[15] HE K M, ZHANG X, REN S, et al. Deep residual learning for image recognition [C] //Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016.

[16] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, et al. Mastering the game of Go with deep neural networks and tree search [J]. Nature, 2016, 529: 484-489.

[17] Bishop C M. Pattern Recognition and Machine Learning [M]. Springer-Verlag New York Inc., 2006.

[18] Elman J L. Finding structure in time [J]. Cognitive science, 1990, 14(2): 179-211.

[19] Williams R J, Zipser D. Gradient-based learning algorithms for recurrent networks and their computational complexity.[M]// Backpropagation. L. Erlbaum Associates Inc. 1998.

[20] Hochreiter S, Schmidhuber J. Long short-term memory [J]. Neural computation, 1997, 9(8): 1735-1780.

[21] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets. In: Proceedings of the 27th International Conference on Neural Information Processing Systems. Montreal,Canada: ACM, 2014. 2672-2680.

[22] Ratliff L J, Burden S A, Sastry S S. Characterization and computation of local Nash equilibria in continuous games. In: Proceedings of the 51st Communication, Control, and Computing (Allerton). Monticello, IL, USA: IEEE, 2013. 917-924.

[23] Sutton R S, Barto A G. Reinforcement Learning: An Introduction [M]. MIT Press, 1998.

[24] MNIH V, KAVUKCUOGLU K, SILVER D, et al. Playing Atari with deep reinforcement learning [C] //Proceedings of the NIPS Workshop on Deep Learning. Lake Tahoe: MIT Press, 2013.