New carpet pattern design with deep learning

Sabiha Unal Eyi and *Feyza Gurbuz

Department of Industrial Engineering, Nuh Naci Yazgan University, Kayseri, , Turkey.

Department of Industrial Engineering, Erciyes University, Kayseri, Turkey.

**Corresponding Author : feyza@erciyes.edu.tr*

Submitted : 16-12-2021

Revised : 28-06-2022

Accepted : 16-07-2022

ABSTRACT

The fact that digitalization touches every aspect of today's world has often been realized with the ideas of the industry and projects carried out for the development of the industry. While Artificial Intelligence shows its effective activities in many areas of the industry and it can also support designers in pre-production design. Within the scope of this study, it is aimed to produce new product designs by using the image data of the most demanded products obtained from a carpet manufacturing company with Deep Convolutional Generative Adversarial Networks. Before performing the model training with the images obtained from the carpet manufacturer company, the image generation performances of a ready-made data set and two different Python libraries, Keras and PyTorch, were compared. As a result, it was determined that the performance of the PyTorch Python library in generating images was higher quality, and the model built with real images was repeated. The synthetic designs produced were presented to the carpet manufacturer, and it was aimed to bring the role of the designer to a position that guides the design models with the designer's experience and knowledge in the design process, which is a complex and stochastic process before production.

Keywords: Deep Learning; Product Design; Generative Adversarial Network.

INTRODUCTION

The concept of Artificial Intelligence is based on the process of creating machines that can be called smart, which can be fed with all existing data types and can show similar actions to those created by human intelligence at the end of their training. The history of the concept which is defined as Artificial Intelligence today dates back to the end of the 18th century and the beginning of the 19th century (Import.io., 2018). In order for a machine that can be called intelligent to be trained and perform certain commands, the necessary codes must be processed into the machine. Coding is the cornerstone of computer science and coding has been based on weaving looms developed by Marie Jacquard. Inspired by Jacquard's loom, Charles Babbage came up with the idea of designing machines that could solve all mathematical problems. The physical parts and punch cards of this machine, which are called the "Analytical Machine", were the hardware and software parts of the first computer (Alpaydın, 2010). The Analytical Machine could not be completed due to the production techniques of its time, however, 50 years later punch cards were used for the "Tabulating Machine" designed by Hermann Hollerith to facilitate the increasing census in the USA. In 1950, Alan Mathison Turing, a British mathematician, and Computer Scientist mentioned the "Turing Test" in his article "Computing Machinery and Intelligence", which starts with the "Can machines think?" question. The aim of the Turing Test is whether it is logically possible for a machine to say what it is thinking and then distinguish between machine and human responses. This thinking is the goal that Artificial Intelligence wants to achieve.

The concept of Artificial Intelligence was introduced in 1956 at the Dartmouth Conference led by John McCarthy. IBM engineers, especially Marvin Minsky, Claude Shannon, and several researchers from the Massachusetts Institute of Technology were at the conference, moreover, these people are today considered the founders and leaders of Artificial Intelligence research. Although there were ups and downs in the field of AI from time to time until 1997, Deep Blue, which was accepted as the first computer that could play chess developed by IBM in 1997, defeated Garry Kasparov who was the Chess World Champion, having made the work in this area more popular. Although the concept of Artificial Intelligence is not strictly separated, it is divided into sub-modules. The first of these is Machine Learning, which first emerged in the 1980s. Until the birth of the Machine Learning concept, traditional coding methods were applied for computers to execute desired instructions. On the other hand, with Machine Learning, computers are now given inputs and outputs, and the computer is expected to code the instructions it needs to follow (Fogg, 2020).

Deep Learning is another sub-module of Artificial Intelligence which is based on Artificial Neural Networks. The basis of Deep Learning is to constitute a similar neural network by observing the multi-layered neural network structure that humans use to solve complex problems. Although approximately 65 years have passed since the birth of the Artificial Intelligence concept, the technological breakthroughs in the last 20 years are equivalent to the technologies realized in previous years. Studies on Image Processing have become one of the most consequential research interests with the growth of the capacities of computers, and the development of Graphics Processing Units and Tensor Processing Units. Seeing, understanding, and making sense of an image is a sense that is not difficult for humans and animals in comparison with machines. Performing this complex task on computers is used quite effectively in various fields such as medicine, engineering, aviation, banking, agriculture, defense industry, and manufacturing. Studies on this subject started to gather speed in 2014 with the article 'Generative Adversarial Networks' by Ian Goodfellow. In this context, the image data that are ensured from a carpet manufacturing company has been used for producing new product designs with Deep Convolutional Generative Adversarial Networks.

METHOD

While discussing with designers and manufacturers how many projection designs to produce for only one season, approximately five hundred projections are designed by about twenty designers is learned. Nonetheless, all designs could not be made actual, which means a huge workload on designers. Starting from this idea, we planned to benefit from a Generative Adversarial Network, which is nowadays used in various areas such as producing images or detecting fraud in an image, for the design of a commercial product, a carpet.

Though there are different Python libraries, in this study before deciding to use which libraries for the model, we chose to compare two of them, which are Keras and PyTorch. After determining PyTorch is the most effective library for our study, we conducted a series of training with the sample dataset. After that, synthetic images produced in different libraries were compared over production speed, error rate, and performance metrics in a graph. In conclusion, the Deep Convolutional Generative Adversarial Network was coded with PyTorch on a real dataset, given the carpet manufacturer company, for the study, and 64 synthetic images were produced. Then generated images were served to the company.

Artificial Neural Network and Convolutional Neural Network

Artificial Neural Network aims to improve the computer's decision-making and problem-solving skills by modeling people's thinking and learning structure. In order to have computers solve the problems that people solve in their daily lives at the same speed, it is necessary to know how the interaction between the neurons in their brains, which show organized behavior while solving these problems. We use algorithms to help computers solve complex problems, and neurons in the brain work algorithmically. We use algorithmic thinking in all decisions we make in our daily lives, consciously or unconsciously. An algorithm is a set of commands applied when converting input to output (Malik, 2005). Neurons are the basic functional units of neuroscience and their main task is to transfer information (Malik et al., 2016). Neurons consist of four main parts which are cell body, dendrite, axon, and axon terminals. After the sensor data from the previous nerve cell is weighted in the nucleus inside the cell body, it proceeds along the axon and connects to another nerve cell via axon terminals. Thus, interneuronal communication takes place. The neurons used in Artificial Neural Networks are also designed with inspiration from the nervous structure of living things (Sietsma, Dow, 1991).

A convolutional Neural Network is a special neural network structure that applies the mathematical operation called convolution, across layers, of data that can be thought of as a two- or three-dimensional pixel grid, such as a one-dimensional grid or image data that regularly receives data. Convolutional Neural Network consists of three basic layers which are the Convolution Layer, Pooling Layer, and Fully Connected Layer. In CNN, a series of linear activations is performed to detect certain features of the convolutional layer data. A Non-Linearity Layer with a non-linear activation function should be added after each Convolution Layer. Then there is the Joining Layer to changes the output of the layer, reducing the number of weights. After the Joining Layer, the Flatten Layer is prepared for the Full Link Layer, which is one of the most important layers in the multidimensional matrix convolution process in the layers. Finally, with Full Link Layer classification or object recognition, the network ends (Radford, Metz, Chintala, 2016). The representation of a classical CNN is as in Figure 2.

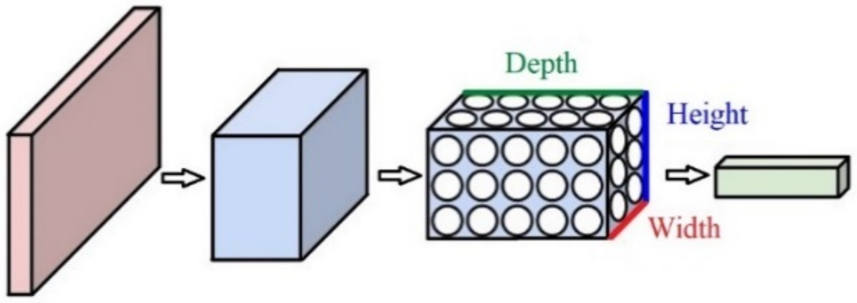

Figure 2 Convolutional neural network representation

Generative Adversarial Network

Generative Adversarial Network is a deep learning algorithm developed to generate new images with the same statistics as the training data. Generative Adversarial Network consists of two neural networks called Generator and Discriminator that work simultaneously but in contention with each other (Bengio, 2014). Random noise is given as input to the generating network, and it is expected to generate fake data from this data so that it can fool the discriminator network. On the other hand, the discriminator network tries to distinguish between the fake data produced from the generator network and the real data. As a result of these processes, the generating network gradually learns to produce data with the same statistics as the real data, and it becomes difficult for the discriminator network to distinguish between fake and real data. Generative Adversarial Network is based on Game Theory. The two-player game theory formula is included in Formula 1 and Figure 3 shows the relationship diagram between the structures in the Generative Adversarial Network (Goodfellow et al., 2014).

$$V(D,G) = E_{x \sim P_{data}(x)}[\log \log D(x)] + E_{z - p_z(z)}[\log(1 - D(G(z)))]$$

Formula 1 The two-player game theory (Wang et al., 2018)

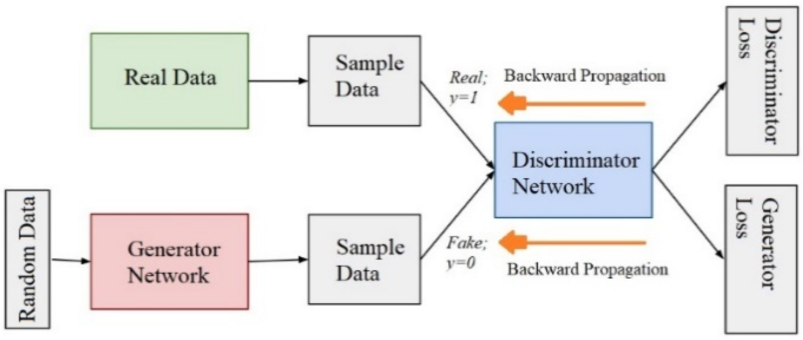

Figure 3 Architecture of generative adversarial network

FINDINGS AND RESULTS

In the study, the sample data set "African Fabric Images", which is a free data set provided on the Kaggle platform, was used before synthetic designs were produced with the real data set. There are 1059 3D colored carpet/fabric images in the sample data set. In the created Generative Adversarial Network model, all 1059 data in the data set were used during the training. 64 images from the images in the data set are included in Figure 4 as an example.

As a result of the comparison of the sample data set and different Python libraries, the model was repeated with the most demanded 64 carpet images provided by a carpet manufacturer. All of the images were used during the training. The real data set provided by the company consists of 3D-colored carpet images.

Deep Convolutional Generative Adversarial Network architecture constructed in the study is a specialized kind of Generative Adversarial Network architecture. The most important features of Deep Convolutional Generative Adversarial Network architecture are; that the pooling layer is not included in the generator and discriminator network, the ReLU activation function is used in layers other than the output layer in the generator network, and the LeakyReLU activation function is used in all layers of the discriminator network (Li, 2021).

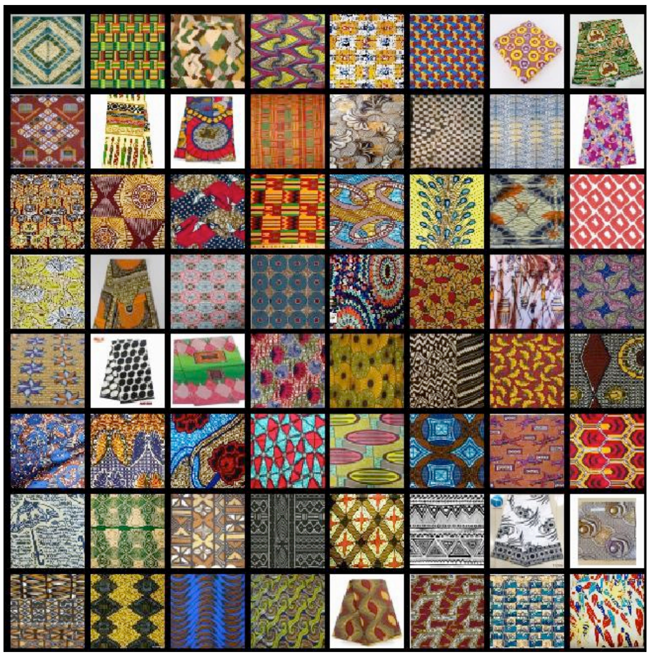

Figure 4 Sample images from the “African Fabric Images” sample data set used in the training

Deep Convolutional Generative Adversarial Network architecture constructed in the study is a specialized kind of Generative Adversarial Network architecture. The most important features of Deep Convolutional Generative Adversarial Network architecture are; that the pooling layer is not included in the generator and discriminator network, the ReLU activation function is used in layers other than the output layer in the generator network, and the LeakyReLU activation function is used in all layers of the discriminator network (Li, 2021).

In the study, the Deep Convolutional Generative Adversarial Network model was coded with two kinds of Python libraries which are Keras and PyTorch and the produced images were compared.

In the constructed generator network model, the images in the datasets were reduced to $64\times64\times3$ dimensions. Then, the normalization process ($X = \frac{(X-127.5)}{127.5}$)was applied to reduce the Pixel values of 3D color images between $0 - 255$ to between $[-1.0, 1.0]$ values (Chollet, 2016).

In the generator network model, the Batch Normalization method is used between the convolution operations and the activation functions with the convolution operations. The reason for using the Batch Normalization method is to reduce the training time and increase the performance of the model by providing resistance against the possibility of vanishing gradients (Lemarechal, 2012). As the activation function, the ReLU activation function is used in the 4 layers outside the output layer, and the Tanh activation function is used in the output layer.

In the discriminator network model, the Leaky ReLU activation function is used in 5 layers, including the output layer, after the convolution processes.

The Deep Convolutional Generative Adversarial Network model, which was coded with the Keras library and used a sample data set, was modeled to be run with 500 epochs. The model was only able to perform 450 epochs, being operated for 22 hours on an Asus FX503V computer with Intel(R) Core (TM) i5-7300HQ CPU, 2.50GHz, 2.50GHz processor, 8.00 GB RAM and 64-bit.

The synthetic images produced at the end of every 50 epochs are shown in Figure 5. In Figure 5, the images produced at the end of the first 50 epochs in the upper left corner and at the end of 450 epochs in the lower right corner are placed, respectively.

The model was coded with the PyTorch library, however, at this time it was able to perform 650 epochs, being operated for 20 minutes on the same computer.

In conclusion, considering the performance of the two libraries, it was decided to perform the carpet dataset on the model which was coded with the PyTorch library. Furthermore, the model was able to perform 700 epochs in 27 minutes. The model coded with the PyTorch library performed faster than the model coded with the Keras library with the existing hardware. For this reason, it was possible to repeatedly run the model coded with the PyTorch library with different training round numbers. The images produced at the end of 100, 200, 300, 400, 500, 600, 650, 700, and 1000 epochs, respectively, in the Deep Convolutional Generative Adversarial Network model coded with the PyTorch library are shown in Figure 6.

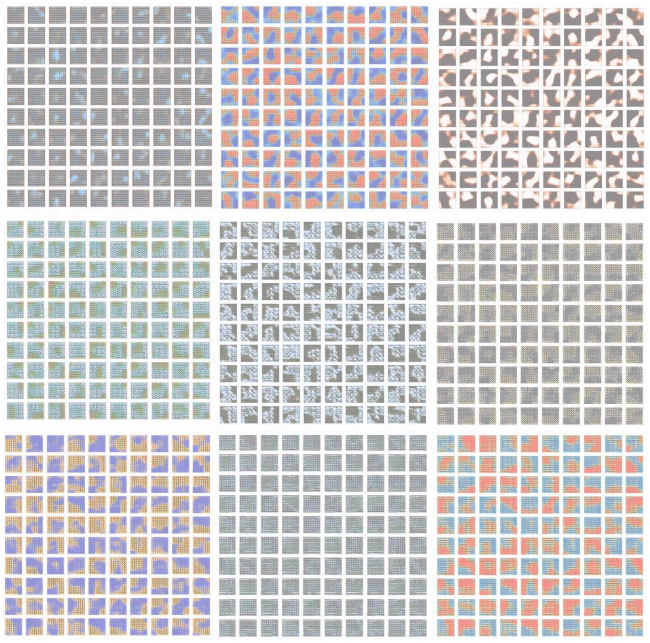

Figure 5 The images produced after every 50 epochs up to 450 epochs in the Deep Convolutional

In conclusion, considering the performance of the two libraries, it was decided to perform the carpet dataset on the model which was coded with the PyTorch library. Furthermore, the model was able to perform 700 epochs in 27 minutes. The model coded with the PyTorch library performed faster than the model coded with the Keras library with the existing hardware. For this reason, it was possible to repeatedly run the model coded with the PyTorch library with different training round numbers. The images produced at the end of 100, 200, 300, 400, 500, 600, 650, 700, and 1000 epochs, respectively, in the Deep Convolutional Generative Adversarial Network model coded with the PyTorch library are shown in Figure 6.

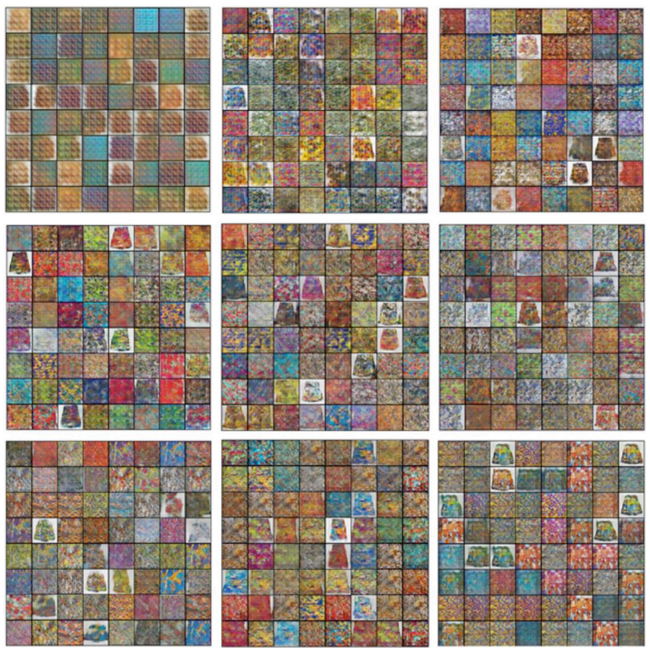

Figure 6 The images produced at the end of each 100, 200, 300, 400, 500, 600, 650, 700, and 1000 epochs, respectively, in the Deep Convolutional Generative Adversarial Network model coded with the PyTorch library using the sample dataset

In order to prove the accuracy of the observations, the error rates of the generator and discriminator network in the Deep Convolutional Generative Adversarial Network model coded with the PyTorch library during the training were plotted on the loss graphs. The images produced by the model could be detected, by ending the training at the points where the error of the generating network is minimum, where the error of the discriminator network is maximum. The error rates of the generator and discriminator network in the model coded with the PyTorch library during the training are given in Figure 7.

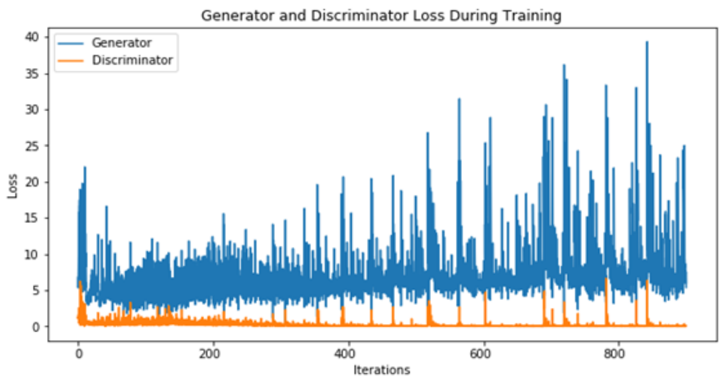

Figure 7 Error rates of the generator and discriminator network in the model coded with the PyTorch library using the sample data set

In the graph in Figure 7, the error of the generator network increased from time to time during the training and the error of the discriminator network decreased, in this case, the images produced by the generating network do not show similar distributions with the real images. However, when the training process was followed step by step, it was determined that after the 650th epoch, the difference between the error rates of the generator and the discriminator network widened and the similarity of the produced images with the real images decreased.

Because the model started to work on the dataset more than necessary and caused overfitting. For this reason, the training was stopped at the 650th epoch.

The created Deep Convolutional Generative Adversarial Network model was repeated, by running 700 epochs, on a real dataset containing 64 images obtained from a carpet manufacturer. The images produced by the model at the end of the training are shown in Figure 8, and the loss values of the generator and discriminator network during the training period are shown in Figure 9, respectively.

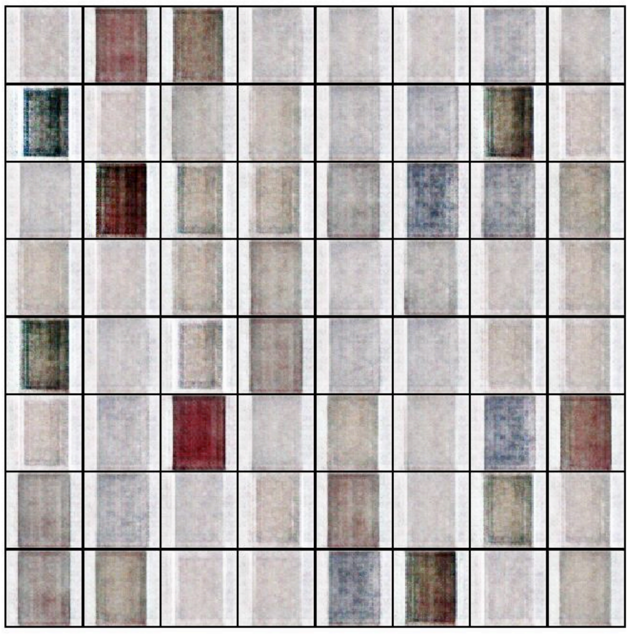

Figure 8 The images produced as a result of 700 epochs coded with PyTorch library using a real dataset

When looked at Figure 8, it is enough in order to give design advice to the designer although generated 64 images are not totally appropriate for producing. Even though the patterns of the images produced are not completely clear, color transitions support the designer to think up an idea.

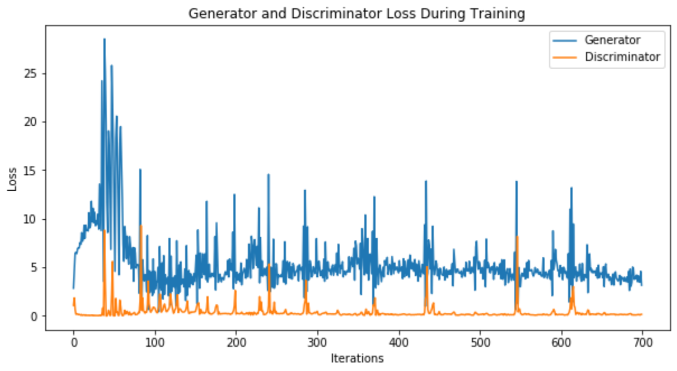

Figure 9 Error rates of the generator and discriminator network in the model coded with the PyTorch library using the real data set

When Figure 9 are looked, it is clearly seen that gap namely error values between the generator and the discriminator network during the training period decreased at the end of the 700 epochs where the discriminator could not detect the real images from the images produced by the generator.

CONCLUSION

In the study carried out, using two different Python libraries, the carpet/fabric image dataset was trained with the Deep Convolutional Generative Adversarial Network model. In the next steps of the study, synthetic image generation performances of different Python libraries were examined in terms of the speed and quality of the images produced, and it was concluded that synthetic image generation problems were solved faster by using the Pytorch library. The aim of the study is to support the decision-making process of the designer in the product design process by producing new designs with similar distributions to the ready-made designs in a shorter time. The model built in this context was repeated using the PyTorch library, which has proven to have higher image generation performance, and the best-selling product images of a carpet company. The training was be able to carry out with 64 images available in accordance with the company's security and data protection strategies.

After presenting synthetic product designs to the design team, it is recognized that the model needs to be trained with much more data and highly equipped computers in order to include the produced synthetic product designs in the production and design process. However, the product designs obtained as a result of the study are proof that the work is supportable and will lead to new projects. It is planned to repeat the study with higher performance hardware, and more and various data sets in the future.

REFERENCES

Import.io. 2018. A history of machine learning and deep learning. Retrieved from https://www.import.io/post/history-of-deep-learning/.

Alpaydın, E. 2010. Introduction to machine learning (2nd ed.). Manning Publications.

Wang, J., Ma, Y., Zhang, L., Gao, R., & Wu, D. 2018. Deep learning for smart manufacturing: methods and applications. Journal of Manufacturing Systems, 48: 144-156. https://doi.org/10.1016/j.jmsy.2018.01.003

Wang, H., Raj, B., & Xing, E.P. 2017. On the Origin of Deep Learning.

Goodfellow, I., Bengio, Y., & Courville, A. 2016. Deep Learning (1st ed.). MIT.

A.Fogg, 2020. A history of machine learning and deep learning, 30 May 2018. [Online]. Available: https://www.import.io/post/history-of-deep-learning/.

Goodfellow, I., Pouget-Abadie, J., Mirza, M. & Xu, B. 2014. Generative adversarial networks. Advances in Neural Information Processing Systems, 2: 2672-2680.

Malik, N. 2005. Artificial neural networks and their applications. National Conference on Unearthing Technological Developments & Their Transfer for Serving Masses.

Malik, N. & Tikoo, S. 2016. Detection, segmentation, and recognition of face and its features using neural network. Journal of Biosensors and Bioelectronics, 7, 2.

Sietsma, J., & Dow, R.J. 1991. Creating artificial neural networks that generalize. Neural Networks, 4: 67-79. https://doi.org/10.1016/0893-6080(91)90033-2.

Radford, A., Metz, L., & Chintala, S. 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. CoRR, abs/1511.06434.

Bengio, Y. 2014. Generative adversarial nets. Advances in neural information processing systems, 27.

Wang, J., Ma, Y., Zhang, L., Gao, R., & Wu, D. 2018. Deep learning for smart manufacturing: methods and applications. Journal of Manufacturing Systems, 48: 144-156. https://doi.org/10.1016/j.jmsy.2018.01.003.

Li, F.-F. 2021. Convolutional neural networks for visual recognition, Stanford University.

Chollet, F. 2016. Deep Learning with Python (2nd ed.). Manning Publications.

Lemarechal, C. 2012. Cauchy and the gradient method. Documenta Math., Extra Vol: Optimization Stories: 251-254.