

A Survey Study of the Deep Learning for Convolutional Neural Network Architecture

Tabark Mohammed Alkhalidi¹, *Noor* Essam², *Ahmed* Ali Talib Al-khazaali¹, *Marwa* Ali Alhamdany³, *Baqar* Assam Hataf¹, *Ali* J. Ramadhan^{1,4*} and *Ali* TaciZadeh⁴

¹University of Alkafeel, Najaf, Iraq

²Imam Jaafar Al-Sadiq University of Najaf, Najaf, Iraq

³Islamic University of Najaf, Najaf, Iraq

⁴University of Qom, Qom, Iran

Abstract. The deep learning (DL) computer paradigm has been the industry standard for machine learning (ML) during the past few years. It has gradually become the most widely used computational technique in machine learning. One of the benefits of DL is its ability to learn massive amounts of data. Deep learning has seen tremendous growth in the last several years and has been successfully used for many traditional applications. More importantly, DL has outperformed popular machine learning algorithms in several domains, such as cybersecurity, bioinformatics, robotics, etc. The field remains mostly uneducated although it has been contributed to several works reviewing the State-of-the-Art on DL, each of which only covered a specific aspect of the field. We thus propose a more holistic approach to this contribution, providing a more suitable basis upon which to construct a thorough understanding of DL. Concerning the most significant DL features, including the most recent advancements in the field, this evaluation specifically aims to provide a more thorough survey. This study specifically describes the kinds of DL networks and techniques, as well as their significance. The most common type of DL network, convolutional neural networks (CNNs), is then presented, and the evolution of it.

Introduction

Machine learning (ML) has been used extensively in research over the last few years, and it has been incorporated into many different applications, such as text mining, image classification, spam detection, video recommendation, and multimedia idea retrieval [1]. Deep learning (DL) is one of the machine learning methods that is most commonly utilized in these applications [2]. Representative learning (RL) is another term for deep learning (DL). For example, the startling rise in data-collecting capacity and the astounding developments in hardware technologies are the causes of the continuous release of new research in the domains of distributed learning and deep learning. High-performance computing is known as HPC. Despite having its foundation in the standard neural network, DL outperforms its predecessors in a pronounced way.

* Corresponding author: ali.j.r@alkafeel.edu.iq

Furthermore, to create multi-layered learning models, DL concurrently employs graphing techniques and transformations. Emerging deep learning (DL) methods have shown impressive outcomes for several uses, including audio and speech processing, visual data processing, natural language processing (NLP), and more.

DL algorithms are used to automatically extract features. As a result, researchers are encouraged to extract distinctive features with the least amount of physical work and domain expertise [3]. These algorithms' multi-layer data representation structure pulls high-level features from the later layers and low-level information from the earlier levels. It should be noted that this kind of architecture, which imitates the operations that take place in the fundamental sensory regions of the human brain, was originally inspired by artificial intelligence (AI). The human brain is capable of automatically deriving data representations from a wide range of scenes. To be more precise,

the input for this procedure is the received scene information, and the output is the labeled objects. This process is similar to how the brain works. It thereby emphasizes the main benefit of DL. Due to its remarkable performance, deep learning (DL) is currently one of the most well-known machine learning research trends. This article presents an overview of deep learning (DL), encompassing important ideas, architectures, applications,

difficulties, computational tools, and evolution matrices. Among the most well-liked and often utilized deep learning (DL) networks are convolutional neural networks (CNNs) [4, 5]. CNN has played a significant role in DL's recent surge in popularity. CNN is the most extensively used because it automatically finds significant properties without human supervision, which is a major advance above its predecessors.

Artificial Intelligence (AI)

Artificial intelligence has received a lot of interest recently and has been discussed in numerous articles. Artificial intelligence is an interdisciplinary discipline with many different techniques. First, what precisely are AI, machine learning, and deep learning? How are they connected?

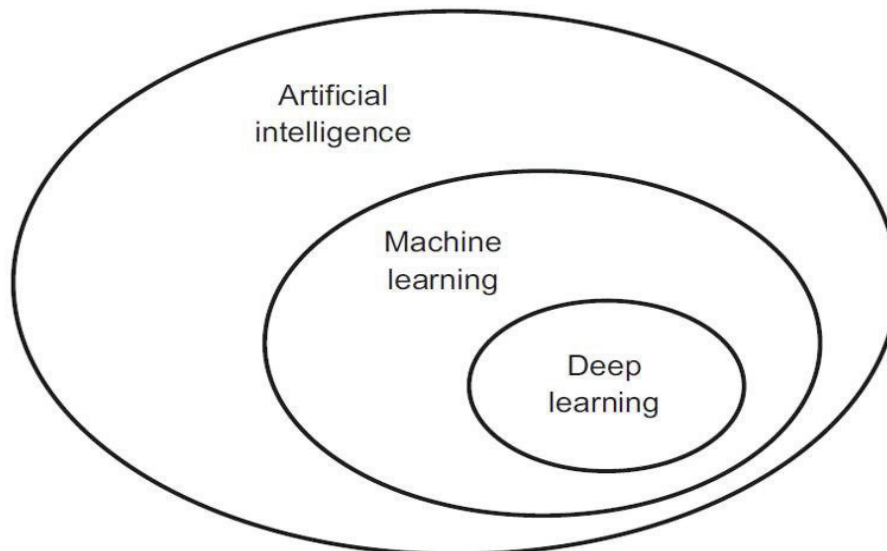


Fig. 1. Deep learning, machine learning, and artificial intelligence [6]

Artificial intelligence (AI)

It is a vast area of computer science that produces intelligent machines that can perform tasks that ordinarily need human intelligence. It is an attempt to replicate human intelligence in robots [7]. Artificial intelligence is typically classed based on how well it can imitate human traits [8].

Machine Learning (ML)

Strives to learn something from data. Statistical learning or predictive analytics are other names for this area of artificial intelligence, statistics, and computer science. Machine learning has increasingly permeated daily life in recent years [9]. Machine learning is the process of creating software that can change in response to fresh data. The ML approach entails looking for patterns in the data. However, analyzing data for human understanding is preferred. Machine learning uses that data to identify patterns and change software operations accordingly [10]. The four machine learning algorithms are semi-supervised learning, reinforcement learning, supervised learning, and unsupervised learning. The most prevalent sort of machine learning algorithm is supervised learning. In Figure 2 [11], various machine-learning algorithm types are displayed.

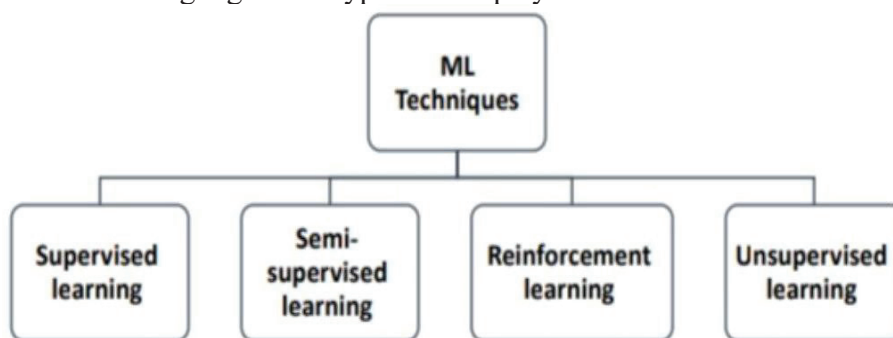


Fig. 2. Machine Learning algorithms [11]

Supervised machine learning algorithms: they apply prior knowledge to fresh data to forecast future events. Based on a given training dataset, the learning algorithm develops an inferential function to forecast output values [11].

Unsupervised machine learning algorithms: utilize data without classifying or labeling the training information. Unsupervised learning explores how computers may infer a function from unclassified data. It is possible to deduce hidden structures that the system is unable to find using unclassified data sets [11].

Semi-supervised machine learning algorithms: train with data that is both classified and unclassified. This technique can significantly increase learning accuracy. When knowledgeable and pertinent resources are needed to train acquired label data, semi-supervised learning is applied. Otherwise, obtaining declassified data is typically not a resource-intensive process [11].

Reinforcement of machine learning algorithms: By engaging with their surroundings and making mistakes or finding rewards, people learn. The pursuit of trial and error and a delayed reward is one of the key components of reinforcement learning. To decide which action is best, the agent needs the reinforcement signal [11][12].

Deep Learning (DL)

It is a subfield of machine learning that has proven important in many applications including text production, voice recognition, computer vision, language translation, and natural language processing.[13] While the term "learning" refers to obtaining new information from the artificial neural network itself, the word "deep" refers to the complexity of the architecture of the artificial neural network. Deep neural networks and feed-forward neural networks with multiple hidden layers rather than "shallow" neural networks, which often have only one hidden layer, are the most common form of neural network layering. As an illustration, in Figure 3, the different representations between shallow and deep neural networks [13].

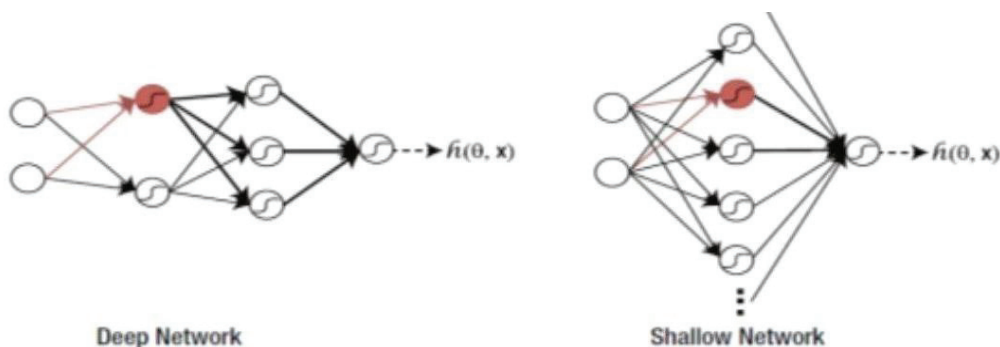


Fig. 3. The difference representations between shallow and deep neural networks [13]

With more data, deep learning models' performance keeps getting better. Figure 4 provides a more accurate depiction of what is discussed [12][13].

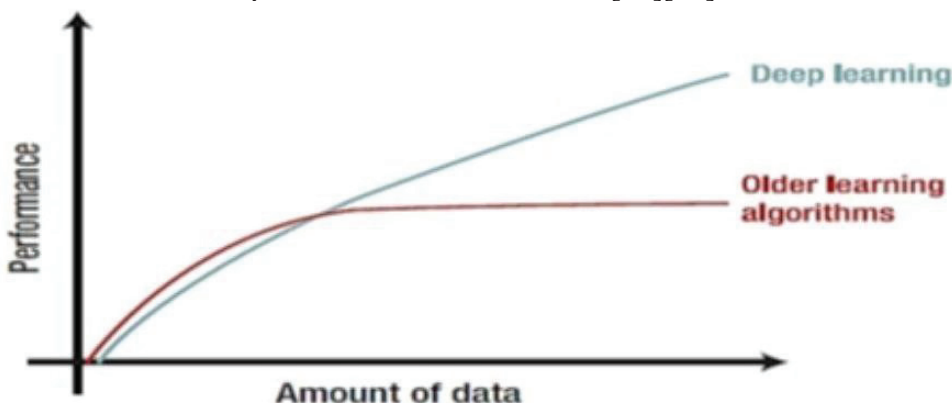


Fig. 4. Scaling data science techniques to the amount of data [13]

To understand how a simple neural network operates, first understand how a neuron functions. Similar to a genuine neuron, Neurons are the basic processing unit within a neural network. Each neuron has input values that enable it to take in external information. Based on this input, the neuron will do internal calculations and generate a value, as seen in Figure 5 by the arrows entering the red neurons. A neuron is fundamentally comparable to a mathematical function that produces an output value from a set of input values. What are the neurons' fundamental calculations, though, that lead to the output? The following example of a neuron and its parameters will be used to demonstrate it: (each parameter's representation within the neuron is visually as shown in the figure 5 [14].

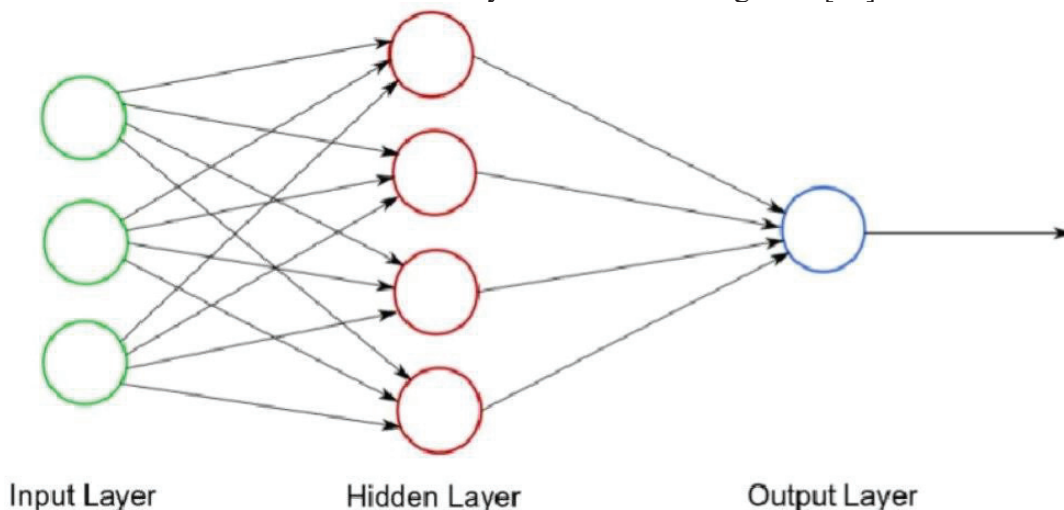


Fig. 5. A Schematic illustration of a neural network [14]

We can build layers by combining many neurons, and we can build any type of neural network by intricately combining layers. There are three main layers: the hidden layer, which receives the knowledge, and the input layer, which is in charge of learning the basic qualities. In this instance, the input is received in a simplified form and is transformed into more complex features by the hidden layer, which may have one or a million layers. The final layer, known as the output layer, is in charge of producing an output from the data that the hidden layer has provided [14]. In a neural network, backward and forward propagation are the two basic activities. A network receives an example to start the forward propagation process, which spreads the example throughout the network to create an output. The error is also transmitted from the end of the network to the beginning of the network using back propagation, which takes place after forward propagation. Both of these tasks are necessary throughout the training phase; backpropagation updates the network's layer weights, and as the network gets better at predicting, it should be able to do so more accurately than it did in the previous step [14].

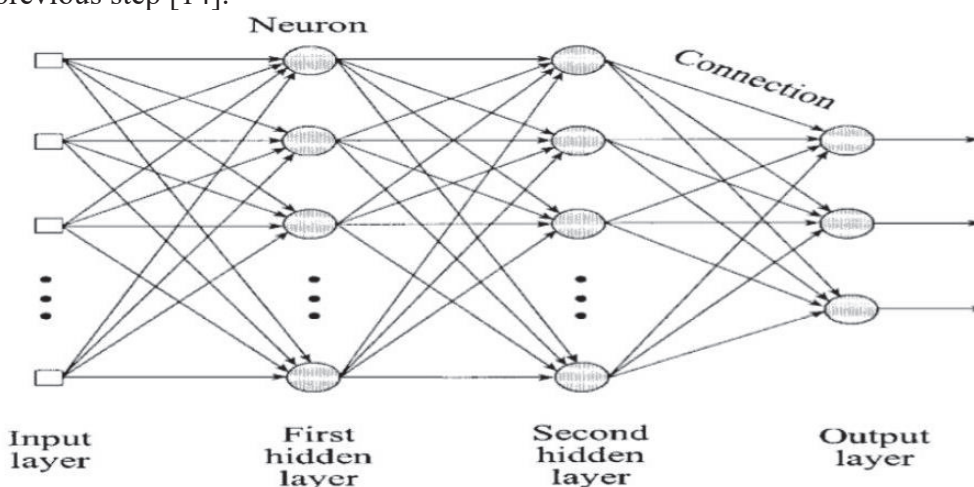


Fig. 6. Topological illustration of a simple neural network [14].

Despite the fact that this is merely a rudimentary illustration of a neural network, as Figure 6 illustrates, we must construct multiple-layer networks with several neurons in each layer.

Activation function (Non-linear Layer)

It is the process of converting inputs into outputs. The input and bias of the neuron are weightedly aggregated to determine the value of the input, if any. This suggests that the activation function produces the corresponding output in order to decide whether or not to fire a neuron in response to a certain input. Additionally, the activation function needs to be discriminating, which is an important feature as it allows the network to train using error backpropagation [15]. Neural networks can use different activation functions in different paradigms [16]. Simply put, a neuron is defined as [17]:

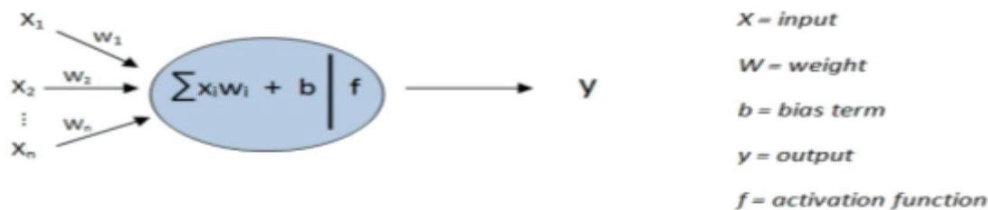
$$Y = f(\sum(\text{weight} * \text{input}) + \text{bias} \dots\dots\dots) \quad (1)$$

In Equation (1), the output, *y*, may take on any value between + infinity and - infinity. However, the issue is determining which value of *y* is the firing value [17]. Where an activation function *f* comes into play,

For a single neuron or group of neurons, (*f*) determines whether a value is sufficiently high or low to represent a neural network decision point. Activation functions may aid in normalizing each neuron's output to either a range of 0 to 1 or a range of -1 to 1[17].

The basic purpose of activation functions in neural networks is to compute and convert the weighted sum of *input* and *bias* at each node into an output value that may be fed into

the next hidden layer or used as output, as shown in Figure 7 below, where x = input, w = weights, b = bias [18].



$$y = f(\sum(\text{weight} * \text{input}) + \text{bias})$$

Fig. 7. Neural network activation functions [18]

CNNs and other deep neural network architectures frequently employ the following categories of activation functions [15].

Sigmoid:

Real numbers are required for this activation function's input, and its output can only be in the range of [0,1]. Equation 2 shows the S-shaped sigmoid function curve.

$$f(x)_{\text{sigmoid}} = \frac{1}{1+e^{-x}} \dots\dots (2)$$

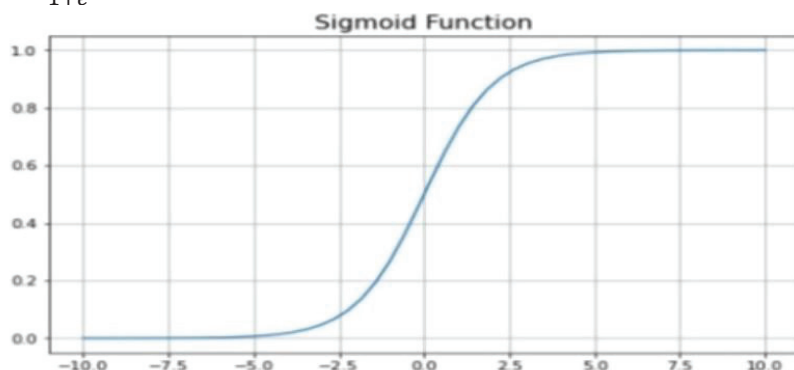


Fig. 8. Sigmoid activation function [16]

Tanh:

It is comparable to the sigmoid function in that it takes real values as input and restricts the output to the interval [-1,1]. Eq 3 is its mathematical representation [16].

$$f(x)_{\text{tanh}} = \frac{e^x - e^{-x}}{e^x + e^{-x}} \dots\dots (3)$$

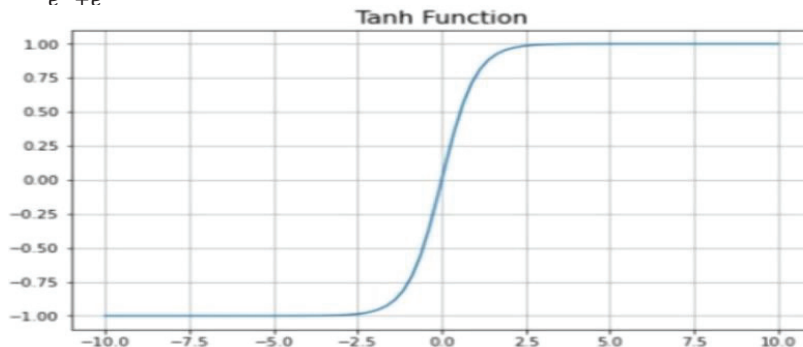


Fig. 9. Tanh activation function [16]

ReLU:

ReLU is the function that is utilized the most frequently on CNN. transforms the integer inputs into positive values. Eq 4 performs better the lesser the computational demand.

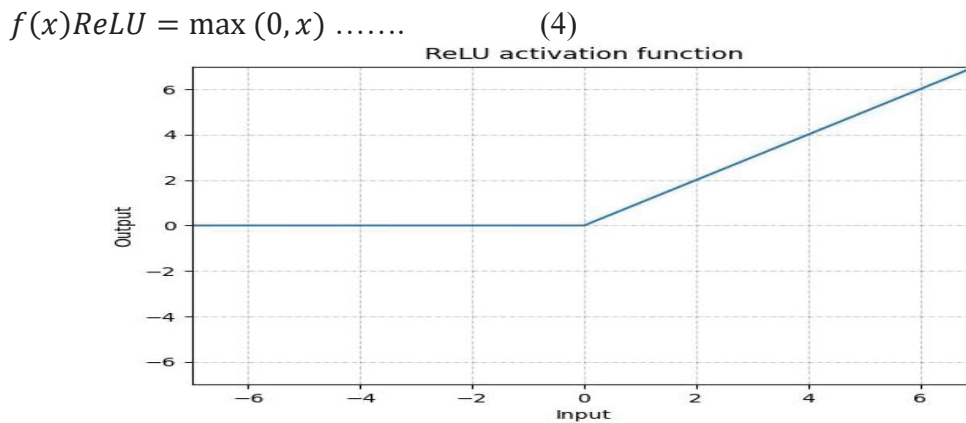


Fig. 10. ReLU Activation Function [16]

The main advantage of ReLU is its equivalent mathematical representation. While using ReLU some important concerns may arise from time to time. For example, have a look at the backpropagation technique for errors that have a larger gradient flowing through. The weights are changed so that the neurons become dormant when this gradient is run through the ReLU function. The term "dying ReLU" describes this issue. Certain ReLU choices can assist in allaying these worries.

Softmax Activation Function

The Softmax function is an additional activation function utilized in neural computation. It is employed to extract a vector of real numbers' probability distribution from another vector of real numbers. For instance, the Softmax function returns a number that spans the range 0 to 1 with a probability of 1. In almost all the output layers of deep learning, architectures that used the Softmax function often appear in Figure 11. The primary difference between the Softmax and Sigmoid activation functions is that the Softmax is used for multivariate classification jobs, whilst the Sigmoid is used for binary classification problems. The softmax function in Equation (5) [17] is used to get the probability for the output layer z_i .

$$f(z_i) = \frac{e^{z_i}}{\sum_j e^{z_j}} \dots\dots (5)$$

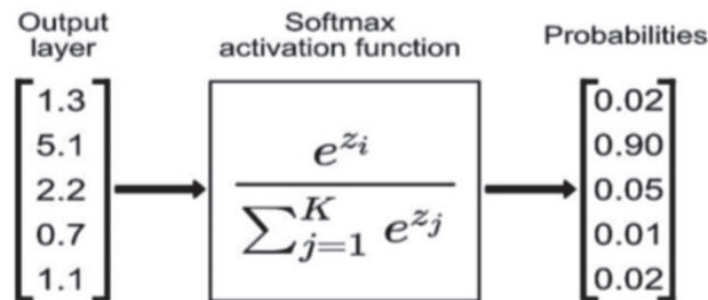


Fig. 11. Softmax Activation Function in output layers of the DL [17].

Convolution Neural Network

CNN is a deep-learning algorithm used to categorize images. CNN uses an image as input assigns learnable weights, removes objects or characteristics from the image, and then uses this information to distinguish one image from another. It used the name ConvNet ConvNet's operating principle is to compress the images into a format that is simpler to process [19][20]. ConvNets are beginning to get a lot of interest from the scientific

community since AlexNet was shown in 2012 to dominate all other techniques for visual categorization [21].

By performing matrix operations in parallel on consumer desktop GPUs, it was possible to train larger networks to classify over a large number of classes, as seen in ImageNet [22]. ConvNets are also used in language translation and other applications like object detection, speech recognition, scene analysis, human location estimate, and video caption production. Without a question, ConvNets are the most well-known and often applied deep learning algorithms [23].

CNNs are extensively employed in several domains, including face recognition and computer vision. [24]. At the neuronal level, CNNs in human and animal brains resemble conventional neural networks, with a greater degree of specificity in the visual cortex of the cat brain, which is made up of an intricate network of cells that CNN has replicated. Three main benefits of CNN [25]. Defined by Goodfellow et al [26]: Common parameters, sporadic interactions, and similar representations. Unlike traditional fully connected networks (FC), CNNs consist of layers arranged as an input layer in a specific order, a convolution layer, a corrected linear unit (ReLU) layer, a fully linked layer, a classification layer, and an output layer being the most common [25].

Convolution and bottom sampling are the two primary processes on which CNN was founded. The convolution is performed with the help of a filter that is trainable and adaptable to a predetermined size and weights that can be adjusted during the training and lower sampling phase [27].

This method speeds up the network while simplifying the training process by using a small number of parameters. Many convolution layers come before sub-sampling (pooling) layers in a popular CNN variation that resembles multi-layer realization (MLP), with FC layers coming in last. An illustration of the CNN structure for image classification is shown in Fig. 12. Each layer's input x in a CNN model is arranged in three dimensions: depth (r), width (w), and height (m), which are equal. Channel number is another name for depth. For example, an RGB image has three depth (r).

Similar to the input image, each convolutional layer, represented by k , has numerous kernels (filters) with three dimensions ($n \ m \ q$); the only distinctions are that n must be fewer than m and q must be equal to or less than r . Furthermore, as was already indicated [28], the kernels serve as the foundation for the local connections, which use the same parameters (weight W^k and bias b^k) to create k feature maps k , each of which is convolved with input and has a size of $(m \ n \ 1)$. Equation 6 illustrates how the convolution layer, like NLP, calculates the dot product of the input and the weights.

$$h^k = f(W^k * x + b^k) \quad \dots \quad (6)$$

However, the inputs are tiny portions of the original image. After that, we apply nonlinearity or an activation function to the output of the convolution layer to get the following results: Downsampling every feature map in the sub-sampling layers is the next stage. Consequently, the training process is accelerated and the overfitting problem is resolved by reducing the network parameters. The pooling function (like max or average) is applied to a P -sized surrounding area for every feature map, where P is the kernel size. The mid- and low-level information are then absorbed by the FC layers, which represent the last layers in a normal neural network, to create the high-level abstraction. For instance, the categorization scores in the top layer are constructed using softmax or support vector machines (SVMs). Each score denotes the probability of a specific class in a given scenario [28].

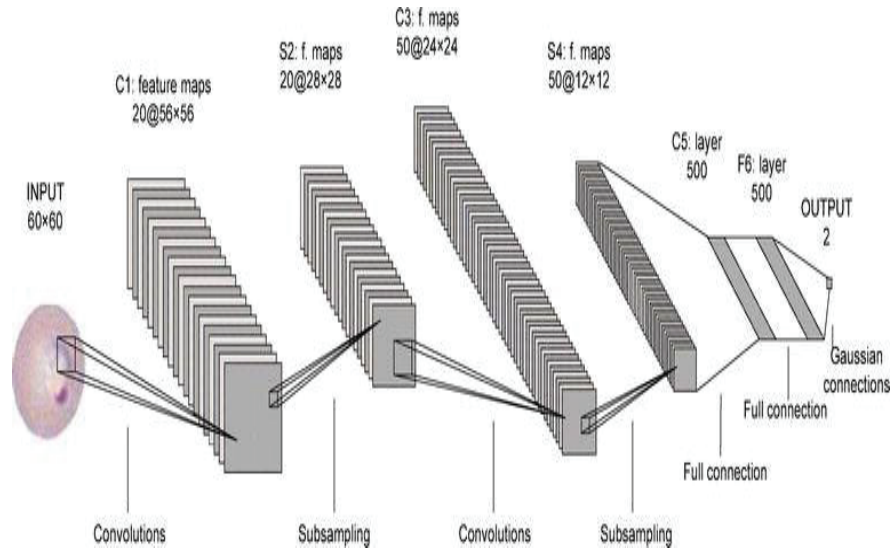


Fig. 12. An example of the CNN architecture for classifying images [28]

Filter Convolution

Several parameters affect how a filter is convolved across an image, things are explained in full here. Selecting an appropriate filter size, stride, and zero padding parameter values requires doing some simple arithmetic to see whether the convolution will work with the feature map dimensions (explained below). How to compute a convolution with a filter across an image's top three filter points is shown in Figure 13 [23].

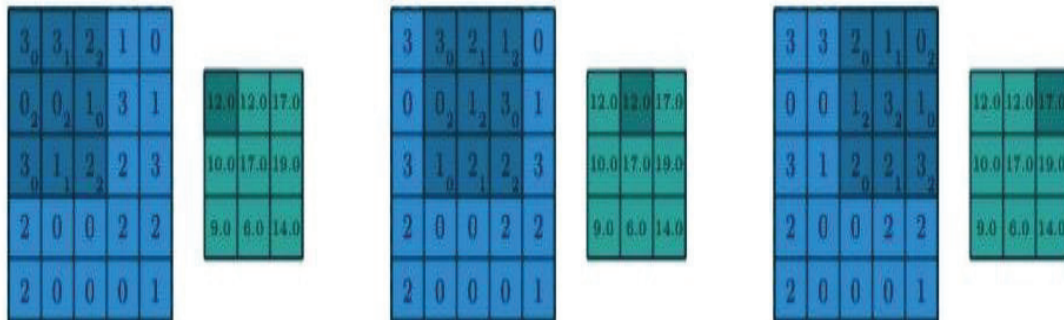


Fig. 13. Compute the discrete convolution's output values [23].

Filter Size

What features can be recognized by the filters depends on their size concerning image size (or activation layer). For instance, images having 224*224 pixels are subjected to 11*11 filters in the input layer of AlexNet. Only 4.3% of the (square) image side length is accounted for by each filter length on average. It is not possible to extract more than 0.24 percent of the input image area with these first-layer filters [23].

Padding

A zero-valued margin known as padding is added around the image. The padding depth can be changed after convolution to prevent the output from the current convolutional layer from getting smaller. The below-mentioned Figure 14 shows this impact. Because some area is lost at each convolution, the drop in output dimension might become significant with multiple successive convolutional layers. By zero-padding the input in a convent with numerous layers, It is possible to mitigate the effects of the output decline. In order to avoid dimensional reduction and preserve the input dimension at the output, use zero padding, as seen in Figure 14. To prevent outputs from getting too small in networks with several layers, this method may be crucial [23].

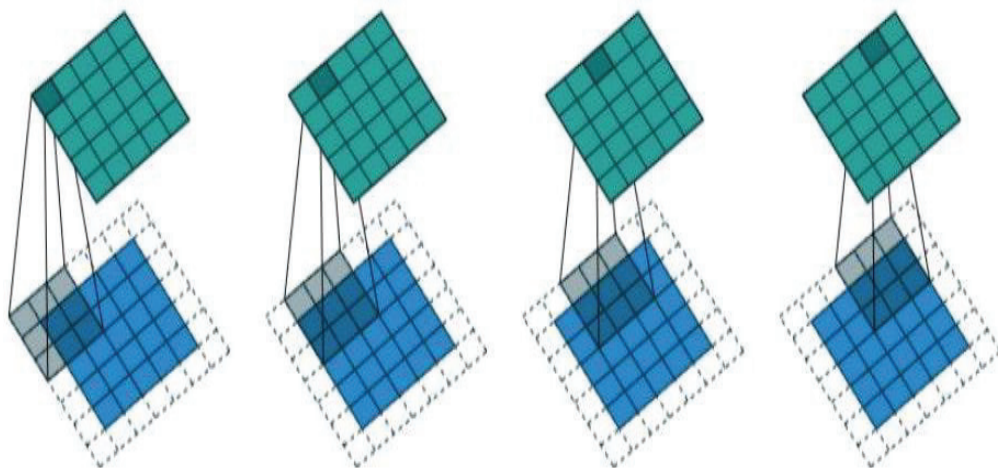


Fig. 14. Zero padding counteracts the effect of output size decrease to keep the input size at the output [23].

Stride

Stride, abbreviated as S, is a parameter that sets the number of pixels that a filter will shift in a horizontal direction (and subsequently vertical direction when it is convolved across the following row). Stride is used in some network topologies to reduce layer size instead of max pooling. This method was employed by Google Net [29].

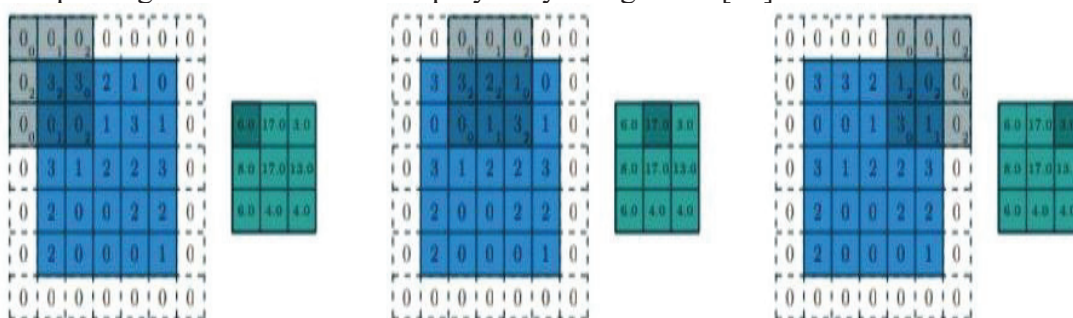


Fig. 15. Convolution with stride > 1 and padding [23].

Convolution layer

The most crucial element in the construction of a CNN is the convolutional layer. It consists of several convolutional filters, sometimes referred to as kernels. To create the output feature map, these filters—which are indicated by N-dimensional metrics—are convolved with the input image [23]. To recognize line pictures, the convolutional layer is processed using either the Sobel method or alternative methods. The Sobel mask method uses several iterations of picture convolution with horizontal and vertical filters. Along with various pixels, each image also has a pixel density, also referred to as the resolution or contrast. Pixels control image qualification and presentation. More pixels will be present in images with higher resolution. Often referred to as computer graphics, Digital images, often known as bitmap images or computer graphic images, are images that are created digitally. The grid will show a bitmap, computer graphic, or digital picture as a square table of continuous pixels. The pixel is a tiny unit of measurement used in computer graphics, including display sheets and other multimedia. Considering its location, each pixel will be given its own space. In a color imaging system, the intensity of each pixel varies. Red, green, blue, yellow, and black all stand for different levels of color intensity. A multidimensional vector is reduced to a one-dimensional vector by the flattened convolutional layer [30][31]. applying a filter to the image and generating the feature map is shown in Figure 16.

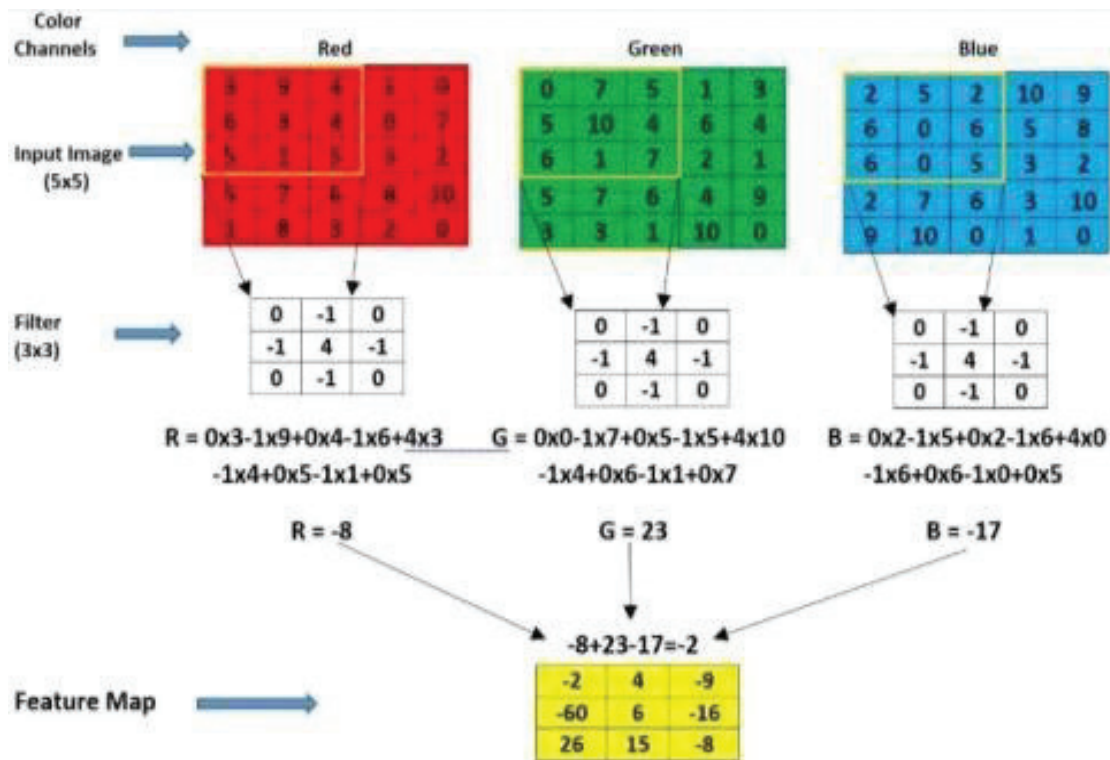


Fig. 16. A 3*3 filter is applied on an input image of size 5*5*3 in a convolutional layer of CNN [23]

Definition of a kernel: A network of distinct values or numbers serves as the representation of the kernel. Each value has a name: the kernel weight. The equations' weights are chosen at random. To serve as kernel weights, random numbers are chosen at the beginning of the CNN training process. There are several ways to initialize the weights as well. The kernel then gains the ability to extract important features by varying these weights during each training session [23].

Convolutional operation: The CNN input format is displayed prior to the convolutional procedure. While CNN receives data in the form of an image with several channels, a classic neural network receives its input in vector format. For example, the grayscale picture format contains one channel, but the RGB image format has three. To further understand the convolutional procedure, let's have a look at an example of a 4*4 grayscale image using a 2*2 random weight-initialized kernel. First, the kernel traverses the whole image in both horizontal and vertical directions. In addition, the computation of the dot product—a single numerical value produced by multiplying and joining the values of the input picture and the kernel—is performed. Figure 17 shows the main computations that are finished at each stage. The input value in the output parameter map is indicated by multiplying both values after they have been summed to get the resultant product values. In the preceding example, the kernel receives one step instead of padding applied to the input image [28].

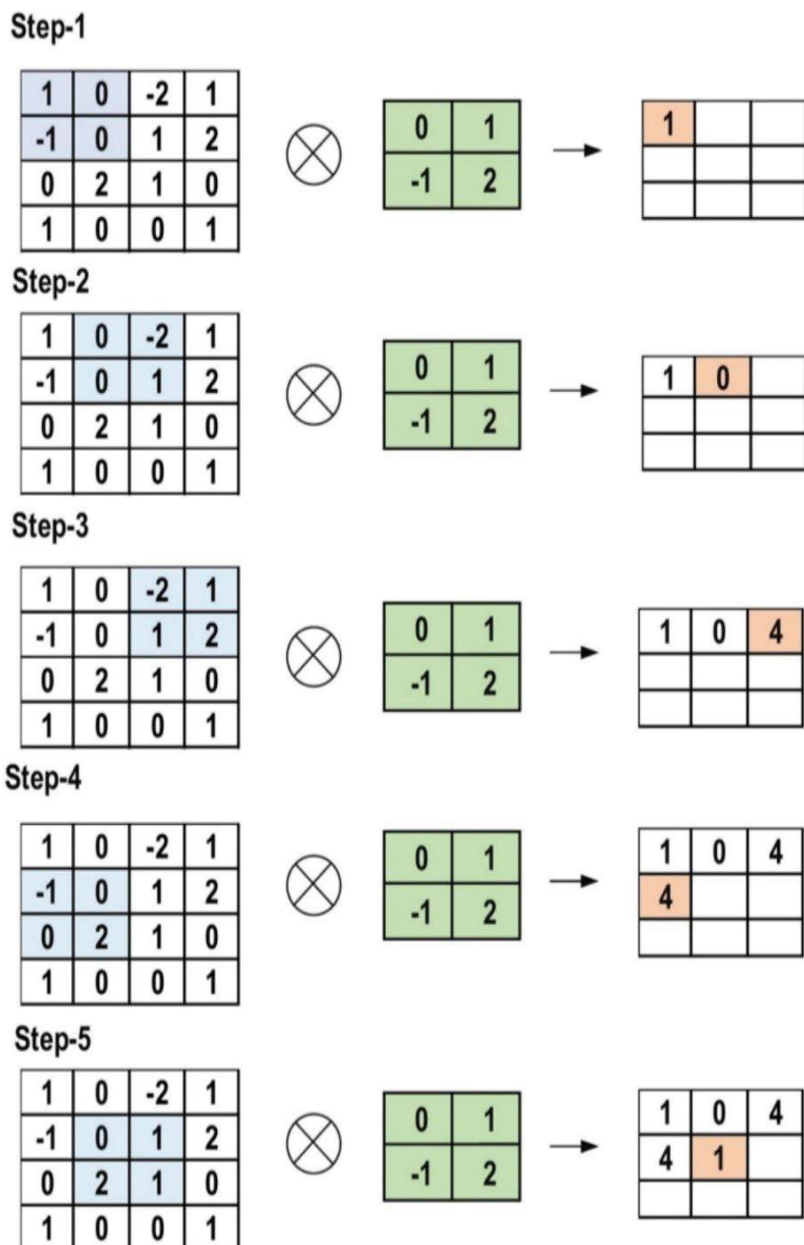


Fig. 17. The initial calculations performed at each step of the convolutional layer [28]

Connectivity is sparse: FC neural networks Every layer's neurons are in communication with every layer's neurons above it. Conversely, in CNNs, very few weights are available between two neighboring layers. since of this, this method is memory-efficient since it requires less weights or connections and less memory to store the weights. Moreover, compared to the dot (.) operation, the matrix operation in CNN is significantly more computationally expensive [23].

Weight Sharing: Since all of CNN's weights interact with each other. In the surrounding layers, there are no precise weights between any two neurons.

the input matrix's pixels for the total, learn a single set of weights The amount of time and money spent on training will be greatly reduced since additional weights for each neuron do not need to be learned [23].

Pooling Layer

By aggregating the output of groups of neurons in one layer into one neuron in the next layer, one of the concepts utilized to extract significant features from convolutional neural

networks and minimize data dimensionality is the aggregation layer. This technique keeps track of the important features while reducing the repeatability of neural network features. Additionally, it can enhance the network instruction cycle and prevent the overfitting issue in high-complexity problem analysis [32]. Many pooling methods are offered in various pooling levels. Global max pooling, global average pooling (GAP), global min pooling, max pooling, tree pooling, gated pooling, average pooling, and max pooling are a few of these options.

The most popular and often utilized pooling algorithms are GAP, max, and min. These three methods of pooling are depicted in Figure 18. This is the pooling layer's primary flaw in that it assists the CNN in determining whether a specific feature is present in an input image, but it only concentrates on accurately locating the feature. Consequently, CNN's overall performance can deteriorate. As such, the CNN mode overlooks important information [32].

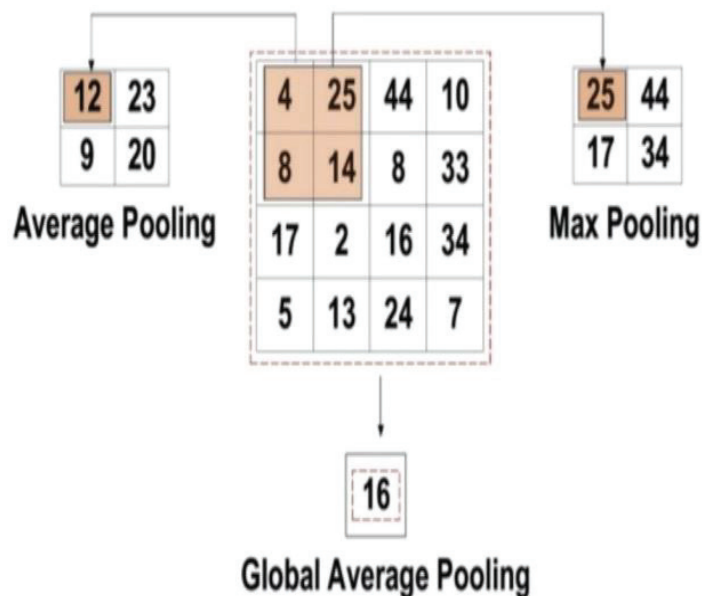


Fig. 18. Three types of pooling operations [23]

Fully Connected Layer

The completely linked layers of a CNN Usually located close to the conclusion of any CNN architecture is this layer [23]. When every input originates from the same source, it's the layers. connected to each activation unit of the layer below it, which is the recipient of the convolution's output. The last layer of the neural network performs attribute extraction following a grouping layer and prior to a classification layer. Put another way, the completely linked input layer is composed of perceptual weight values that change based on the structure. The best possible performance is ensured by the entirely connected output layer. Figure 19 displays the likelihood of each label [30].

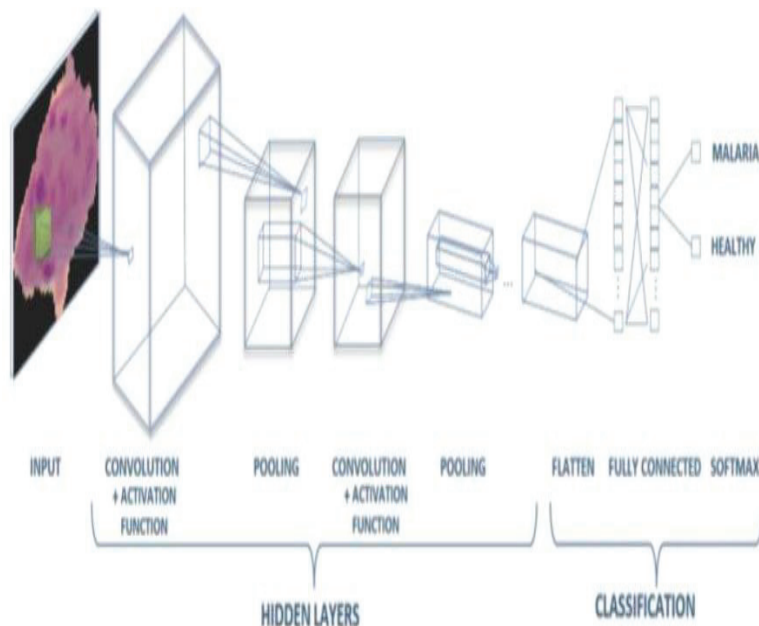


Fig. 19. Convolutional neural network (CNN) model [30]

Regularization to CNN

The primary issue with CNN models that prevents them from attaining well-behaved generalization is overfitting. When a model performs exceptionally well on training data but poorly on test data (unseen data), it is said to be overfitted. Conversely, if the model does not learn enough from the training set, it becomes underfitted. When a model shows strong performance on both training and test sets of data, it is considered to have "good balance". Figure 20 depicts these three groups. To lessen overfitting and aid in regularization, a number of intuitive ideas are employed [23].

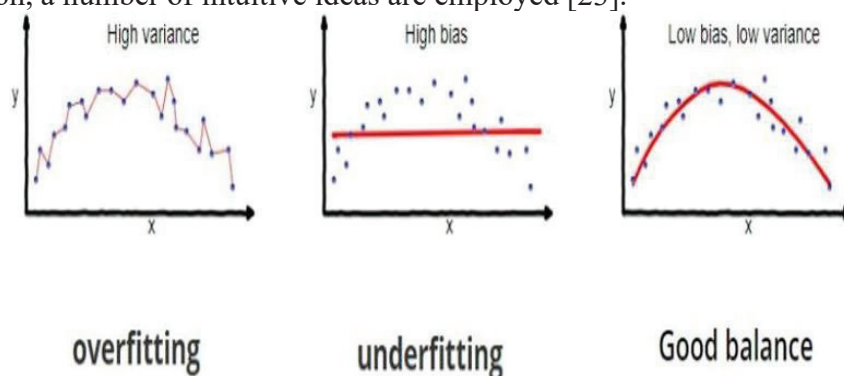


Fig. 20. Overfitting and Underfitting issues [30]

Dropout: A common generalization strategy is this one. Every training session involves the random dropping of neurons. Consequently, this forces the model to disperse the feature selection power over the entire set of neurons and learn a huge number of independent features. The dropped neuron will not take part in backward or forward propagation during training. The full-scale network, on the other hand, is utilized to make predictions during the testing process [23].

Optimizers

Gradient descent is an optimization method used to build models based on deep learning and neural networks by recursively advancing in the direction of the maximum gradient, which is determined by the negative of the gradient. Since each data instance of the dataset is required to compute each value, it is subject to a local minimum. The neural network

weights have been changed. The hyperparameters of the model are tuned by stepwise descent. Optimizers are tools for controlling how neural networks learn [33].

One popular optimization strategy in neural network model training is stochastic gradient descent (SGD). This algorithm makes use of the gradient descent technique. Because SGD, which is based on this assumption, only takes into account one sample at a time, each iteration's direction isn't always the same as the overall model optimization [30]

$$u = u - \mu - \nabla u g(u, x^i, y^i) \quad (7)$$

SGD is an optimization method that, sample by sample, lowers the sample step-gradient descent performance loss function. The input is χ_i , the label training is y_i , and the SGD learning rate is $0.001(\eta)$. μ serves as the cost function in the gradient calculation.

The gradient descent can be accelerated by using the square prop strategy, similar to the momentum method. To further optimize the issue of excessive swing amplitude in the loss function update. RMSprop was not officially published by Hinton. Nevertheless, it gained rapid recognition as one of the most popular gradient descent optimization methods for deep learning. The average score for this is 0.9 [34]. Here is the formula.

$$\varphi_{t+1} = \varphi_t - \frac{\mu}{\sqrt{E(g^2)_{t+\varepsilon}}} g_t \quad \dots\dots \quad (8)$$

The value (0.9) for the solution fraction at time step t was found by the study using Hinton's input (γ , or rho) [35]. The diagonal matrix is $[g \ 2 \ t]$, the decaying average is θ_{t+1} , and the RMSprop learning rate is $0.001 (\eta)$.

Significant learning from the prior time point is used by Nadam [36], which has a number of obvious implications for gradient descent weight updates. Based on Adam's optimizer, Nadam is a well-liked optimizer because it eliminates the drawbacks of Adagrad's reduced learning rate [37], establishing the model and facilitating additional research. In addition, it reduces parameter choppy problems and is faster than gradient descent.

$$\varphi_{t+1} = \varphi_t - \frac{\mu}{\sqrt{v_{t+\varepsilon}}} (\beta_1 m_t + \frac{(1-\beta_1)g_t}{1-\beta_1}) \quad \dots\dots \quad (9)$$

Nadam's learning rate is $0.002(\eta)$; the optimizer's (Nadam) improved efficiency in time step t is represented by the objective function (θ_t) using $\varepsilon = 1e-08$ and $\beta_1 = 0.9$, depending on work using v_t and m_t .

Conclusion

In conclusion, it is imperative to include a succinct explanation that compiles all pertinent information gathered throughout this comprehensive investigation

Even now, DL finds it challenging to model multiple complicated data modalities at once. Another well-liked approach in the most recent developments in DL is multimodal DL.

DL requires large datasets, ideally labeled data, in order to train the models and forecast unknown data. When analyzing data in real-time is required if there aren't enough datasets available, this task becomes especially challenging (such as in the case of healthcare data). To address this issue, research on data augmentation and TL has been conducted in recent years.

While supervised learning is the foundation of many deep learning models now in use, machine learning is steadily moving toward semi-supervised and unsupervised learning in order to handle real-world data without the need for manual human categorization.

CNN performance is highly impacted by hyper-parameter selection. The performance of CNN will be impacted by even the smallest alteration to the hyper-parameter values. Thus, choosing the right parameters for an optimization method is an important factor that needs to be taken into account.

To train CNNs effectively, powerful hardware resources such as GPUs are needed. Additionally, they are necessary to investigate the effectiveness of CNN in embedded and smart systems.

References

1. Al-Dulaimi K, Chandran V, Nguyen K, Banks J, Tomeo-Reyes I. Benchmarking hep-2 specimen cells classification using linear discriminant analysis on higher order spectra features of cell shape. *Pattern Recogn Lett.* 2019;**125**:534–41.
2. Potok TE, Schuman C, Young S, Patton R, Spedalieri F, Liu J, Yao KT, Rose G, Chakma G. A study of complex deep learning networks on high-performance, neuromorphic, and quantum computers. *ACM J Emerg Technol Comput Syst (JETC)*. 2018;**14**(2):1–21
3. LeCun Y, Bengio Y, Hinton G. Deep learning. *Nature*. 2015;**521**(7553):436–44.
4. Yao G, Lei T, Zhong J. A review of convolutional-neural-network-based action recognition. *Pattern Recogn Lett.*2019;**118**:14–22.
5. Dhillon A, Verma GK. Convolutional neural network: a review of models, methodologies and applications to object detection. *Prog Artif Intell.* 2020;**9**(2):85–112.
6. O. ElJundi, W. Antoun, N. El Droubi, H. Hajj, W. El-Hajj, and K. Shaban, "hULMonA: The Universal Language Model in Arabic," no. **1**, pp. 68–77, 2019, doi: 10.18653/v1/w19-4608.
7. J. Puri, "ARTIFICIAL INTELLIGENCE: A COMPARATIVE STUDY," *Alochana Chakra Journal*, vol. **IX**, no. V, May, p. 9522, 2020.
8. Eban Escott, "What are the 3 types of AI? A guide to narrow, general, and super artificial intelligence," *Codebots*, 2017. third-even-possible (accessed Jun. 04, 2022).
9. C. Müller and Sarah Guido, *Introduction to Machine Learning with Python*. 2017.
10. S. E. Ibrahim, "Predicate Project Outcomes Using Machine Learning," *International Journal of Science and Research (IJSR)*, vol. **6**, no. 6, pp. 1653–1656, 2017.
11. S. Ray, "A Comparative Analysis and Testing of Supervised Machine Learning Algorithms," no. August, 2018, doi: 10.13140/RG.2.2.16803.60967.
12. Al-Khayyat, Ebtihal Aziz Mustafa . "Building Online an Arabic Word Embedding Model Using Deep Learning." (2022).
13. P. Goyal, S. Pandey, and K. Jain, *Deep Learning for Natural Language Processing: Creating Neural Networks with Python*. Apress, 2018.
14. S. Wu, W. Ren, C. Yu, G. Chen, D. Zhang and J. Zhu, "Personal recommendation using deep recurrent neural networks in NetEase," 2016 IEEE 32nd International Conference on Data Engineering (ICDE), 2016, pp. 1218-1229, doi: 10.1109/ICDE.2016.7498326.
15. K. Kumar and G. S. M. Thakur, "Advanced Applications of Neural Networks and Artificial Intelligence: A Review," *International Journal of Information Technology and Computer Science*, vol. **4**, no. 6, pp. 57–68, 2012.
16. J. Heaton, *Artificial Intelligence for Humans, Volume 3: Deep Learning and Neural Networks*. 2015.
17. John Paul Mueller and L. Massaron, *Deep Learning For Dummies*, vol. 53, no. 9. 2019.
18. C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv*, pp. 1–20, 2018

19. Sandhya,yamala.Eswaran,k.kumar sahuo,prasanta (2020). Malaria disease detection using deep learning technique. *Journal international journal advance sciences and technology* 7736-7745
20. J. Kim, H. Cho, J. Pyo, B. Kim and S. -C. Yu, "The convolution neural network based agent vehicle detection using forward-looking sonar image," *OCEANS 2016 MTS/IEEE Monterey*, 2016, pp. 1-5, doi: 10.1109/OCEANS.2016.7761209.
21. Alex Krizhevsky, Ilya Sutskever, and Georey E Hinton. Imagenet classication with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 10971105, 2012.
22. Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*. IEEE, 2009.
23. Murphy, John. "An overview of convolutional neural network architectures for deep learning." *Microway Inc* (2016): 1-22.
24. Fang W, Love PE, Luo H, Ding L. Computer vision for behaviour-based safety in construction: a review and future directions. *Adv Eng Inform.* 2020;43:100980.
25. Hubel DH, Wiesel TN. Receptive felds, binocular interaction and functional architecture in the cat's visual cortex. *J Physiol.* 1962;**160**(1):106
26. Goodfellow I, Bengio Y, Courville A, Bengio Y. *Deep learning*, vol. 1. Cambridge: MIT press; 2016.
27. Alqudah, A.M., Alquraan, H., Qasmieh, I.A. (2019). Segmented and non-segmented skin lesions classification using transfer learning and adaptive moment learning rate technique using pretrained convolutional neural network. *Journal of Biomimetics, Biomaterials and Biomedical Engineering*, **42**: 67-78.
28. Alzubaidi, Laith, et al. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions." *Journal of big Data* 8.1 (2021): 1-74.
29. Christian Szegedy, Wei Liue, Yangqing Jia, Pierre Sermanet Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Go- ing deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.
30. Sriporn, Krit. Tsai,cheng-fa.Tsai, Chia-en. Wang,Paohsi (2020). Analyzing malaria disease using effective deep learning approach
31. Uchida, K.; Tanaka, M.; Okutomi, M. Coupled convolution layer for convolutional neural network. *Neural Networks 2018*, **105**, 197–205. [CrossRef] [PubMed]
32. Nasr-Esfahani, E.; Rafiei, S.; Jafari, M.H.; Karimi, N.; Wrobel, J.S.; Samavi, S.; Soroushmehr, S.M.R. Dense pooling layers in fully convolutional network for skin lesion segmentation. *Comput. Med. Imaging Graph.* 2019, **78**, 101658
33. Liu, Q.; Xiang, X.; Qin, J.; Tan, Y.; Tan, J.; Luo, Y. Coverless steganography based on image retrieval of DenseNet features and DWT sequence mapping. *Knowl. Based Syst.* 2020, **192**, 105375.
34. Ruder, S. An overview of gradient descent optimization. *arXiv* 2017, arXiv:1609.04747v2. 35.
35. Hinton, G.E.; Srivastava, N.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R.R. Improving neural networks by preventing co-adaptation of feature detectors 2012. *arXiv* 2012, arXiv:1207.0580.
36. Kc, K.; Yin, Z.; Wu, M.; Wu, Z. Depthwise separable convolution architectures for plant disease classification. *Comput. Electron. Agric.* 2019, **165**, 104948.

37. Duchi, J.C.; Bartlett, P.L.; Wainwright, M.J. Randomized smoothing for (parallel) stochastic optimization. In Proceedings of the 2012 IEEE 51st IEEE Conference on Decision and Control(CDC), Maui, HI, USA, 10–13 December 2012; Institute of Electrical and Electronics Engineers (IEEE); pp. 5442–5444.