Deep Convolutional Neural Network (DCNN) Models for Image Recognition: A Review

[1]Fatin A. Hamadain, [2]Abdalla A. Osman, [3]Ahmed Abdelrahman Mohamed Hamed

[1] Graduate College University of Bahri, Khartoum, Sudan
[2] Department of Computer Sciences, University of Elahlia, Wad Madani, Sudan
[3] Department of Computer Sciences and Mathematics, University of Bahri, Khartoum, Sudan

***Abstract*- The application of the artificial intelligence technique known as deep learning, which is a form of machine learning inspired by the structure and function of the brain, has had some success in the processing and analysis of visual media. The convolutional neural networks (CNNs) are one type of deep neural network that are typically utilized for image processing, particularly for images recognition and object classification. The requirement to construct a network in which neurons in the first layer extracted local visual features and neurons in the later layers combined these features to form higher-order features served as the primary impetus for the development of CNNs. Image recognition is the process of recognizing an image and assigning it to one of a set of categories. Image recognition can also be referred to as image classification. As a result, apps and software that utilize image recognition are able to ascertain what the subject matter of a photograph is and recognize its various components. Within this scope, this study investigates, analyzes, and reviews several deep convolutions neural network (DCNN) models that are designed specifically for these kinds of tasks.**

***Keywords*- Image recognition, object detection, deep learning, convolutional neural network, DCNN models.**

I. Introduction

The convolutional neural networks (CNNs or ConvNets) are based on artificial neural deep learning networks, which designed specifically for visual image analysis. The main reason CNNs popularity is because it can automatically discover important elements with little to no human participation [1]. CNNs have found a number of applications outside of traditional computer vision, such as in speech processing, face recognition, etc. Convolutional neural networks (CNNs) are just normalized multilayer perceptron. CNNs are so-called because every neuron in one layer is connected to every neuron in the next layer. Sometimes, overfitting occurs frequently in these networks due to their full connectedness [2].

The CNN architecture is multi-layered as presented in Figure 1. Each layer delivers special purpose, as follows. The first layer of CNNs is called a convolutional layer, which is the basis of any effective CNN, and multiple convolutional filters, or kernels, make up the system. For creating the final feature map, these filters are convolved with the input image (represented in N-dimensional metrics). The second layer is the pooling layer, which primarily is used to reduce the size of the feature maps. When making these maps, convolutional operations are used, and the size of massive feature maps are reduced into more manageable ones. At the same time, it preserves the vast majority of the most important data (or features) at each and every stage of the pooling procedure. The third layer is the activation function (non-linearity). All activation functions in all neural networks have the fundamental job of mapping the input to the output. Computing the weighted sum of the neuron input and its bias yields the input value. This means that the activation function is responsible for creating the output that indicates whether a neuron will fire in response to a certain input. The fourth layer is the fully connected layer. Each CNN design will have this layer at its conclusion. The fully connected (FC) method involves linking every neuron in this layer to every neuron in the one below it. As such, it serves as the CNN classifier [3].

The feed-forward artificial neural network (ANN) follows the same fundamental procedure as the standard multilayer perceptron (MLP) neural network. Input from the preceding pooling or convolutional layer is fed into the FC layer. This data is a vector that is produced by flattening the feature maps. The final layer is loss functions, which is the last layer in the CNN architecture, is where the final classification is obtained. The output layer of the CNN model employs various loss functions to compute the training-wide predicted error. The dissimilarity between actual and expected results is represented by this mistake. Then, it will be fine-tuned with the help of CNN's learning process.

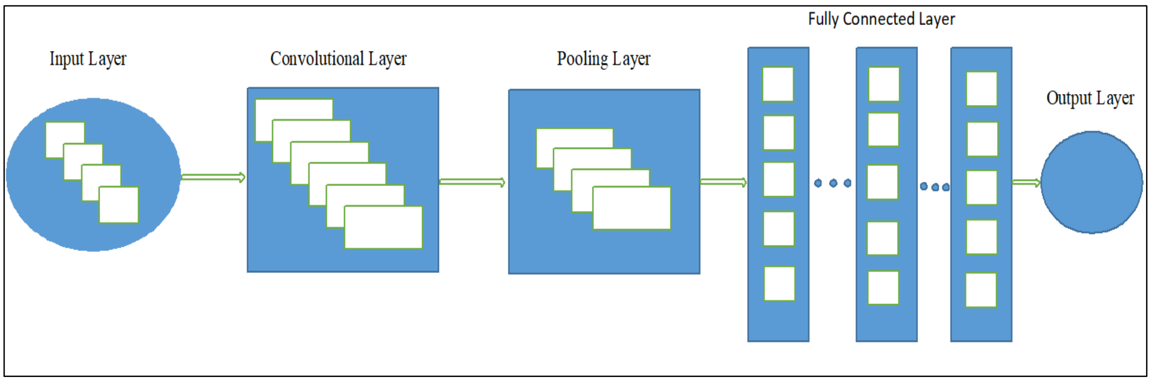

Figure 1: A typical convolutional neural networks (CNN) architecture

Image recognition is the process of identifying and detecting an object or a feature in a digital image or video. This concept is used in many applications like systems for factory automation, tollbooth monitoring, and security surveillance. Typical image recognition algorithms include: optical character recognition, pattern matching and gradient matching, face recognition, license plate matching, scene identification or scene change detection. Face recognition is the process of identifying one or more people in images or videos by analyzing and comparing patterns. Algorithms for face recognition typically extract facial features and compare them to a database to find the best match. Face recognition is an important part of many biometric, security, and surveillance systems, as well as image and video indexing systems. Machine learning and deep learning methods can be a useful approach to image recognition. A machine learning approach to image recognition involves identifying and extracting key features from images and using them as input to a machine-learning model. A deep learning approach to image recognition may involve the use of a convolutional neural network to automatically learn relevant features from sample images and automatically identify those features in new images [4].
This research aims to review deep convolutional neural network (DCNN) models for image recognition. The paper is organized as the next. First, it introduces topic domain and defines its terms. Second, it demonstrates the methodology of the convolutional neural networks (CNNs). Third, it reviews 22 convolutional neural networks (CNNs) models for image recognition, and summarizes them accordingly. Forth, it analyzes and discusses the reviewed 22 convolutional neural networks (CNNs) models for image recognition. Finally, it concludes the research, and recommends some future works and directions.

II. CONVOLUTIONAL NEURAL NETWORKS (CNNs) METHODOLOGY

In CNNs, a picture is fed into and processed by a convolutional network; its original meaning will be challenged. Upon receiving a standard color image, a convolutional network treats it as a rectangle box, its width and height determined by the number of pixels along those dimensions, and its depth three layers deep, one for each letter in RGB. Channels are the collective noun for these nested tiers of depth. Each pixel R, G, and B intensities will now be represented numerically as part of one of the three stacked two-dimensional matrices that make up the image volume. To better classify images, the convolutional network must first learn to identify which of these raw sensory data represent meaningful signals. To process images, a convolutional net takes in square patches of pixels rather than individual ones. That filter, like the patch, is a square matrix; however, it is significantly smaller than the image. The filter task is to identify patterns in the image pixels, and it is known as a kernel, which may be familiar to individuals who are acquainted with support-vector machines.
The key challenge in achieving well-behaved generalization in CNN models is over-fitting. The model is considered over-fit when it shows exceptional results on the training data but not the test data (unseen data). In contrast, an under-fitted model is one that does not pick up enough information from the training data. The model is considered just-fit if it works well on both the training data and the test data. Dropout is a method of generalization has gained considerable traction recently. Throughout each stage of training, a certain number of neurons are systematically removed. This not only forces the model to acquire new, unrelated characteristics, but also ensures that the feature selection power is spread evenly over the entire group of neurons. When a neuron is removed from a network during training, it is no longer included in either backpropagation or forward propagation. However, predictions are carried out using the full-scale network during the testing phase. Drop-Weights is identical to dropout except that, instead of removing neurons, weights representing connections between neurons are removed after each training epoch[4]. Data augmentation is the easiest technique to prevent over-fitting is to train the model using a large sample size of data. To accomplish this, data augmentation is employed. The size of the training dataset can be artificially increased using a variety of methods. Following this section is a detailed explanation of the data augmentation methods that were used. Batch normalization technique guarantees that the activations generated are effective. This efficiency has a Gaussian distribution with a unit variance. The output of each layer should be normalized by subtracting the mean and dividing by the standard deviation. Each layer of the network can view this as a pre-processing duty, and it can be further refined and combined with other networks as needed. Similarly, it is employed to lessen the "internal covariance shift" of the activation layers. The shift in internal covariance between layers is defined by the dispersion of activation. Continuous weight update during training, which may occur if training data samples are acquired from a number of diverse sources, amplifies this change to a significant degree (for example, day and night images). So, the training time will increase since the model will take longer to converge. A layer representing the batch normalization operation is added to the CNN structure to solve this problem [5][3]. The advantages of utilizing batch normalization are as follows:

1. It avoids the occurrence of the vanishing gradient issue.
2. It can prevent the negative effects of a bad starting weight.

3. It drastically shortens the amount of time needed for a network to converge (for large-scale datasets, this will be extremely useful).
4. It has trouble minimizing hyper-parameter training dependence.
5. Regularization has a small effect on over-fitting, making it less likely to occur.

III. CONVOLUTIONAL NEURAL NETWORKS (CNNs) MODELS FOR IMAGE RECOGNITION

1. LeNet:

Since its introduction in 1998, LeNet has become the most popular CNN architecture. LeNet was initially developed to classify the MNIST Dataset handwritten digits from 0 to 9. It consists of seven distinct layers, each of which can be trained independently. It takes photos as large as 32x32 pixels, which is significantly larger than the images used for training. RELU was chosen as the activation function [6]. LeNet is one of the earliest successful digit-recognition systems, it helped sort through handwritten digits. Unfortunately, this network proved not effective in terms of computing cost or accuracy while processing huge photos or classifying among a large number of object classes [6].

2. AlexNet:

AlexNet has five convolutional layers, beginning with an 11x11 kernel, to perform its transformations. It was the first architecture to use three large linear layers with dropout, RELU activation functions, and max-pooling layers. Using the network, we were able to divide photos into a thousand distinct groups. The network structure is quite similar to that of the original LeNet, but it includes far more filters, allowing it to handle much more extensive item classification [7]. For overfitting, it employs regularization alternative, dropout. AlexNet architecture basic layout, displaying its five convolutional and three fully connected layers. It was with two GPUs that the original network was trained. It's 8 levels deep, and each one has its own tunable preferences. Input to the Model consists of RGB images. The RELU activation function is employed across all stages. Specifically, two Dropout layers were employed. Softmax is the activation function utilized in the last layer of the network [7].

3. GoogleNet/InceptionNet:

Google Inception Network, or GoogleNet, performed at virtually human levels to win the ILSVRC 2014 competition. Google model is an enhanced version of the original LeNet blueprint. It has modeled after the inception module. GoogleNet is a 22-layer deep convolutional neural network that is based on Inception. GoogleNet is now used in many different computer vision applications, including face detection and identification, adversarial training, and many others. Nine inception modules are used in the InceptionNet/GoogleNet architecture, with max-pooling layers in between each module (to halve the spatial dimensions). There are 22 of them in all (27 with the pooling layers). Once the final inception module is complete, it uses global average pooling [8].

4. ResNet:

ResNet, created by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang, is a popular deep learning model. In 2015, a paper titled "Deep Residual Learning for Image Recognition" appeared. Among the many deep learning models currently in use, ResNet is widely regarded as among the most powerful and widely used. ResNets, which are built from residual blocks, are fast even when training hundreds of layers because they use skip-connections and a lot of batch normalization [9].

5. MobileNetV1:

MobileNet is built on top of depthwise separable convolutions, a type of factorized convolution that breaks down a standard convolution into a depthwise convolution and a pointwise convolution. MobileNets uses a method called depthwise convolution, which applies a single filter to each input channel individually. Once the results of the depthwise convolution have been obtained, they are combined using 11 convolutions in the pointwise convolution. The inputs are combined and filtered in a standard convolution, which then produces a new set of outputs. The depth-separable convolution technique is used to split this into a filtering layer and a combining layer [10].

6. ZfNet:

Originally introduced in 2013 by Zeiler and Fergus, the ZfNet is a multilayer variant of the Deconvolutional NN (DeconvNet). ZfNet was created as a means of quantitatively depicting the operation of a network. The network activity visualization was made to observe how well CNN was doing by measuring the activation of its neurons. Five convolutional layers are shared across the design, along with max-pooling layers, dropout layers, and three completely linked layers. A 77-size filter and a smaller stride value were used in the initial processing stage. Finally, the ZfNet ends with a Softmax layer [11].

7. Depth based CNNs:

With the assumption that deeper networks have a better chance of approximating the objective function with more nonlinear mappings and enlarged feature hierarchies, deep CNNs are built and designed with this goal in mind. The achievement of guided training is generally attributed to the high density of the network used.
Theoretical investigations have demonstrated that deep networks are superior to shallow systems in expressing a wide range of functions. All functions can be estimated by a single hidden layer, according to a universal approximation theorem proven by Csáji in 2001. Due to the ever-increasing number of neurons required, this is sometimes computationally unrealistic. Bengio and Delalleau claim that deeper networks can preserve the network expressive capacity while incurring less overhead in this regard [12].

8. Highway Networks:

Highway Networks was introduced by Srivastava; it is a deep convolutional neural network (CNN) designed on the idea that a more complex network will lead to better learning results. For starters, training and convergence times for deep networks are painfully sluggish. Highway Networks provide a state-of-the-art cross-layer connection mechanism for the rapid development of deep networks through the application of depth learning to discover more accurate feature representations [13].
Systems that rely on several pathways inside a CNN are sometimes referred to as highway networks. High-speed highway networks with 50 layers have a better convergence rate than thin yet deep systems. The authors in [13] demonstrated that a basic network underperforms when there are more than 10 layers of hidden units. Highway Networks, on the other hand, were found to converge substantially faster than simple networks, even at a depth of 900 layers [13].

9. Wide ResNet:

Some have pointed to the issue of feature reuse as a fundamental shortcoming of deep residual networks. In this scenario, it is possible that certain feature alterations or blocks contribute relatively little to learning. Because of this issue, the Wide ResNet was developed. The residual units, say Zagoruyko and Komodakis, are the primary factor in deep residual networks' learning capacity, whereas the depth itself just plays a supporting role. ResNet was built to be more shallow than deep, harnessing the potential of the residual blocks [14].
Through the introduction of a new parameter, k, Wide ResNet was able to expand the scope of the network. Wide ResNet has proven that layer spreading can improve performance almost as much as, if not more than, increasing the density of residual networks [14].

10. VGG:

VGG is an old convolutional neural network architecture. Studying how to pack more nodes into existing networks served as the basis for this concept. In this network, tiny filters of size 3 × 3 are used. The network consists of just a single fully-linked layer and two additional, simpler pooling layers. To model the relationship between network depth and representational capability, VGG was built with 19 layers, while AlexNet and ZfNet both use 12 [15].
ZfNet, a leading network from the 2013-ILSVRC competition, claims that using narrow filters improves CNN performance. Using a stack of 3x3 filters, VGG demonstrated that the effect of a big size filter could be achieved using smaller filters of a different size. Previously, VGG had used 11x11 and 5x5 filters (5x5 and 7x7). By limiting the number of parameters, the computational cost is decreased when using small-file-size filters. These findings prompted a shift in CNN's research toward using filters of decreasing size [15].

11. PolyNet:

With PolyNet, for the first time, neural networks can be automatically trained and optimized, yielding better results for AI and machine learning. Through exhaustive network and space exploration, PolyNet is able to automatically optimize efficiency and usefulness by making smart choices about weights and structure. The revolutionary PolyNet is the first of its kind: a fully operational, locally hosted neural network training system that guarantees no sensitive information leaves your facility. In fact, this is what sets PolyNet apart from other similar networks [16].

12. Inception v2:

Inception v2 is a convolutional neural network design that makes use of batch normalization, and it represents the second version of the Inception framework. The framework of Inception 2.0. Instead of one 55% convolution, we use two 33% convolutions. It improves computational performance by reducing the amount of time needed for computing; this is because a 55 convolution is 2.78 times more expensive than a 33 convolution. Therefore, having two 33-layer structures rather than 55 improves the performance of the building as a whole [8].

13. Inception v3 V4 and Inception-ResNet:

As successors to Inception-V1 and V2, Inception-V3 and Inception-V4 also include a ResNet model. The goal of developing Inception-V3 was to lessen the computational burden of deep neural networks without compromising their generalization ability. To bypass the huge filters, the authors used 1x1 convolution as a bottleneck, and then used smaller, asymmetric filters of 5x5 and 7x7 sizes instead (1x7 and 1x5). When 1x1 convolution is used in tandem with a large size filter, the resulting process is reminiscent of a cross-channel correlation. The feasibility of 1x1 filters in NIN design was examined in [17] research. Ingeniously, Szegedy and his team used the same strategy [17].
After splitting the input data into three or four smaller 3D spaces using a 1x1 convolutional operation, Inception-V3 maps all correlations in these spaces using standard (3x3 or 5x5) convolutions. The authors merged the efficacy of residual learning with inception block to create Inception-ResNet. Here, the residual link acted as a filter summing.
With increased depth and breadth, Inception-V4 with residual connections (Inception-ResNet) was shown by the authors to have the same generalization capability as plain InceptionV4. Training Inception networks with residual connections appears to significantly quicken the process, as evidenced by the fact that Inception-ResNet converges faster than Inception-V4 [17].

14. DenseNet:

The goal of DenseNet was to address the issue of vanishing gradients in a similar fashion to how Highway Networks and ResNet addressed the issue of decreasing gradients. Multiple layers may produce negligible results because ResNet expressly retains

information via additive identity transformations. DenseNet uses cross-layer connectivity, albeit with some tweaks, to fix this problem. In a feed-forward structure, DenseNet links each previous layer to the next layer [18].
Despite its thin layer structure, the parametric cost of DenseNet grows with the amount of feature-maps used. The network efficiency is increased when gradients are made available to each layer via the loss function. For problems with little training data, the regularizing impact of direct admittance to the gradient helps to prevent excessive hyper parameter optimization [18].

15. Pyramidal Net:
Because of the dense stacking of several convolutional layers, the depth of feature maps grows in consecutive levels in earlier deep CNN architectures like AlexNet, VGG, and ResNet. Subsampling layers that follow each convolutional layer or block reduce the spatial dimension, nevertheless. Due to the significant rise in feature-map depth and the accompanying loss of spatial information, they concluded that the learning ability of deep CNNs is constrained.
To improve ResNet learning capabilities, the authors in [19] presented a new network architecture called the Pyramidal Net. When compared to ResNet, whose spatial breadth decreases suddenly with depth, the width per residual unit in Pyramidal Net progressively grows. Instead of maintaining the same spatial dimension within each residual block until down sampling, this method allows pyramidal Net to cover all possible locations. The pyramidal Net gets its name from the continual upward growth of its depth in the characteristics map.
There are two methods in which Pyramidal Net can grow a network: by adding nodes or by multiplying them. In contrast to the geometric growth experienced by the multiplicative pyramidal structure, the linear growth experienced by the additive pyramidal structure is the distinguishing feature between the two forms of broadening. However, as the size of the Pyramidal Net grows, so do the dimensions of space and time [19].

16. Xception:
The state-of-the-art Inception architecture Xception uses depthwise separable convolution. Xception swelled the size of the original inception block and swapped out the many spatial dimensions (1x1, 5x5, 3x3) for a single dimension (3x3) and a 1x1 convolution to reduce computational complexity. By decoupling spatial and feature-map (channel) correlation, Xception improves the network computational efficiency [20].
The computation is simplified by Xception use of cross channel correlation through pointwise convolution after each feature-map is convolved along the spatial axes individually (1x1 convolutions). Even though Xception transformation technique doesn't reduce the total number of parameters, it does boost learning efficiency and overall performance. Even though Xception transformation technique doesn't reduce the total number of parameters, it does boost learning efficiency and overall performance [20].

17. Channel Boosted CNN using TL:
To improve the network representational capacity, the study in [20] developed Channel boosted CNN (CBCNN), a novel CNN design based on adding more input channels. Deep discriminative models can be made more effective by applying supplementary deep generative models to produce extra channels.
The explanatory causes driving the variation in the data, CB-CNN employs generative learners called AEs. To construct a more accurate representation of the input data, a unique application of the inductive TL concept is proposed, which involves supplementing the learnt distribution of the input data with the original channel space. The channel-boosting phase of CB-CNN is encoded in a generic block that is initially appended to a deep network. This is because, as stated by CB-CNN, TL can be used during both the generating and discriminating stages. Adding to the study significance are the supplementary learners, which include multi-deep learners and generative learning models. As a result, the CNN-based deep discriminator gains improved representational power [20].

18. Residual Attention Neural Network:
The authors in [21] presented a Residual Attention Network to enhance the network representation of characteristics (RAN). The goal of adding attention to CNN was to make a network that could learn about objects and their properties. The feed-forward CNN RAN was built using stacked residual blocks and the attention module. The attention module trunk and mask subsystems each employ a distinct bottom-up, top-down learning mechanism.
As a result of combining two separate learning algorithms within the attention module into a single feed-forward process, rapid feedforward processing is now paired with top-down attention feedback in a single step. Low-resolution feature-maps rich in semantic information are built using a bottom-up feed-forward architecture. Top-down architecture, on the other hand, creates numerous detailed features for each pixel.
Stacking several attention modules increased RAN recognition of cluttered, complicated, and noisy pictures. Since RAN is built in layers, each feature-map can have its weight dynamically adjusted to reflect its relative importance [21].

19. Attention Based CNNS:
The ability of a neural network to discriminate between similar examples is highly dependent on the level of abstraction it operates at (NN). In addition to learning various abstraction layers, it is vital for picture identification and localization to focus on context-relevant features. The term "attention" is used to describe this effect in the human visual system. A human sees a situation in fragments, focusing on the context-relevant elements. This method not only aids in focusing attention, but it also draws several conclusions about the meaning of objects in a given space, making it easier to learn how things are organized visually [22].
There are a number of ways in which the interpretability of RNNs and LSTMs are similar. RNNs and LSTMs use attention modules to generate sequences of data, with new samples weighted based on how often they appeared in earlier iterations. The idea of

attention was implemented in CNN by a variety of researchers in order to improve representation and circumvent computational limitations. CNN's ability to recognize things in challenging environments is aided by this attentional concept [22].

20. Feature-Map (ChannelFMap) Exploitation based CNNs:

CNN's hierarchical learning and automatic feature extraction capabilities have proven useful for MV tasks. The accuracy of modules used for classification, segmentation, and detection depends heavily on the features chosen. CNN dynamically selects features by modifying the weights associated with a kernel, also called a mask. Furthermore, a series of feature extraction stages are carried out, allowing for a rich set of features to be extracted (called as feature-maps or channels in CNN). However, some of the feature-maps aren't particularly helpful when it comes to categorizing objects [23].
An increase in background noise from using a large number of features could cause the network to become over-trained. The generalization performance of a network can be significantly influenced by feature-map selection in addition to the engineering of the network itself. The term "channel" will be used interchangeably with "feature-map" throughout this section because that is how many academics refer to feature-maps [23].

21. Squeeze and Excitation Network:

In order to improve channel interdependencies with little computational cost, Squeeze-and-Excitation Networks are introduced as a new component for Convolutional Neural Networks (CNN). Squeeze-and-excitation blocks are a type of architecture that may be added to a convolutional neural network to boost performance while only slightly increasing the total number of parameters [24].
Adding parameters to the channel of each convolutional block allows Squeeze-and-Excitation Networks to dynamically change the weight assigned to each feature map [24].

22. Competitive Squeeze and Excitation Networks:

Competitive Inner-Imaging Squeeze and Excitation for Residual Network was proposed in 2018 by Hu and colleagues (CMPESE Network) [25]. The authors used the SEblock idea for better learning of deep residual networks. To better discriminate between classes, SE-Network adjusts the feature-maps. For ResNet, residual information is solely used to determine the weight of each feature-map, which is the root cause of the issue with SE-Net. By doing so, the impact of SEblock is mitigated and ResNet data is no longer necessary. To solve it, the authors in [25] created motifs (statistics) unique to each feature map, drawing from both residual and identity mapping [25].
The following Table I summarizes the reviewed 22 convolutional neural networks (CNNs) models for image recognition.

Table I: A summary of the reviewed 22 convolutional neural networks (CNNs) models for image recognition

No.	CNN Model Name	Year	Used for	Layer	Input Size	Parameters
1	LeNet [6]	1998	Recognizing the handwritten and machine-printed characters.	7	32×32 = 1024	60000
2	AlexNet [7]	2012	Any object-detection task	5	227×227 = 51529	62300000
3	GoogleNet [8]	2014	Face detection and recognition, adversarial training.	22	224×224 = 50176	60000000
4	ResNet [9]	2015	Computer vision applications.	4	224×224×3 = 150528	25000000
5	MobileNetV1 [10]	2014	To reduce the model size and complexity.	2	224×224 = 50176	4200000
6	ZfNet [11]	2013	To visually determine which parameters should be tuned to get better accuracy instead of trial and error.	5	224×224 = 50176	48700000
7	Depth based CNNs [12]	2015	Image classification with large image datasets.	5-10	448×448 = 200704	15000000
8	Highway Networks [13]	2015	Text sequence labeling and speech recognition tasks.	4	32×32×3 = 3072	1400000
9	Wide ResNet [14]	2017	To decrease depth and increase the width of residual networks.	5 convo	1000×1000 = 1000000	23000000
10	VGG [15]	2014	Image recognition architectures	41	224×224 = 50176	138000000
11	PolyNet [16]	2017	Generalizes residual unit using Polynomial compositions	3 convo	331×331 = 109561	133000000
12	Inception v2 [8]	2014	Batch normalization.	164	299×299 = 89401	56000000
13	Inception v3 V4 and Inception-ResNet [17]	2014	Split, Transform, Merge Uses asymmetric filter.	48	299×299 = 89401	25000000
14	DenseNet [18]	2016	To improve the declined accuracy caused by the vanishing gradient in high-level neural networks.	2 block	29×29 = 841	76284

15	Pyramidal Net [19]	2017	Increases width gradually per unit.	110	224×224 = 50176	1700000
16	Xception [20]	2018	To classify images.	71	299×299 = 89401	22800000
17	Channel Boosted CNN using TL [20]	2018	Learning discriminative patterns.	-	-	-
18	Residual Attention Neural Network [21]	2018	Reduce the number of parameters in the network while improving the classification performance.	100	56×56 = 3136	8600000
19	Attention Based CNNS [22]	2018	helps neural networks solve complicated tasks by dividing them into smaller areas of attention and processing them sequentially.	6 convo	320×320 = 102400	48960000
20	Feature-Map (ChannelFMap) Exploitation based CNNs [23]	2018	Residual and identity mappings both are responsible for rescaling the channel.	32 map	28×28×1 = 784	-
21	Squeeze and Excitation Network [24]	2018	Improves channel interdependencies at almost no computational cost.	2 block	224×224 = 50176	27500000
22	Competitive Squeeze and Excitation Networks [25]	2018	Residual and identity mappings both are responsible for rescaling the channel.	3 convo	299×299 = 89401	36920000

IV. ANALYSIS AND DISCUSSION

This research reviewed 22 deep convolutional neural networks (DCNN) models to explore and investigate their attributes, as shown in Table 1. The review process involves year of DCNN model, the usage of DCNN model, number of DCNN model layers, DCNN model input image size, and parameters number of DCNN model. In this research, the Pearson correlation coefficient (r) is utilized to measures statistically the degree to which two variables move in relation to each other [26]. It shows the strength of a relationship between two variables, and is expressed numerically by the correlation coefficient. The Pearson correlation coefficient (r) values range between -1 and 1. When is equal to 1, it means a perfect positive correlation, and while is equal to 0 means there is no correlation, and in case is equal to -1 means a perfect negative correlation.

The Pearson correlation coefficient (r) between year of DCNN model and number of DCNN model layers is presented in Figure 2. The Pearson correlation coefficient (r) is equal to 0.3721 that is a positive correlation, which indicates that number of DCNN model layers increase with the years advances.

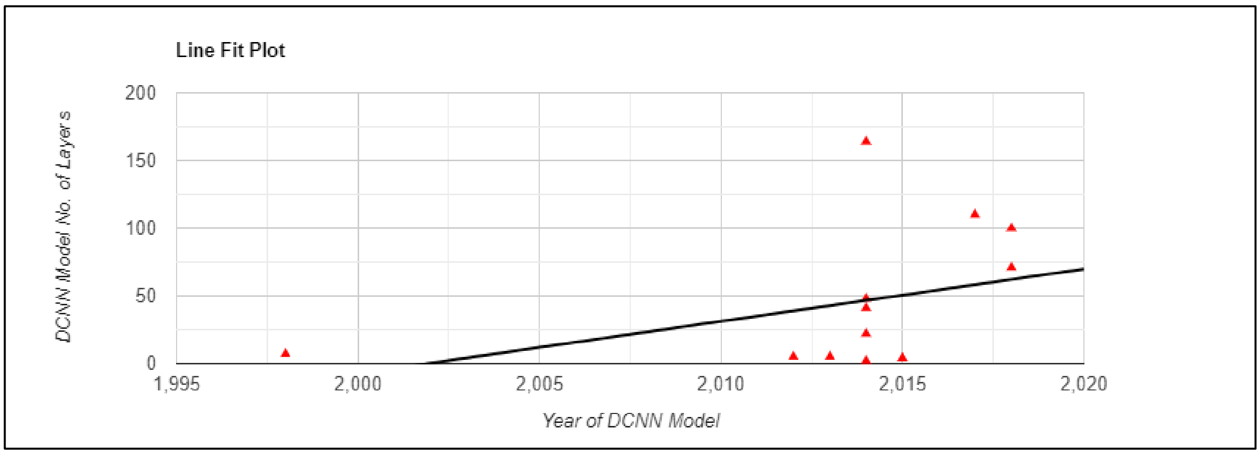

Figure 2: The Pearson correlation coefficient (r) between year of DCNN model and number of DCNN model layers

Similarly, the Pearson correlation coefficient (r) between year of DCNN model and DCNN model input image size is presented in Figure 3. The Pearson correlation coefficient (r) is equal to 0.1615 that is a positive correlation, which indicates that DCNN model input image size raises with the years advances.

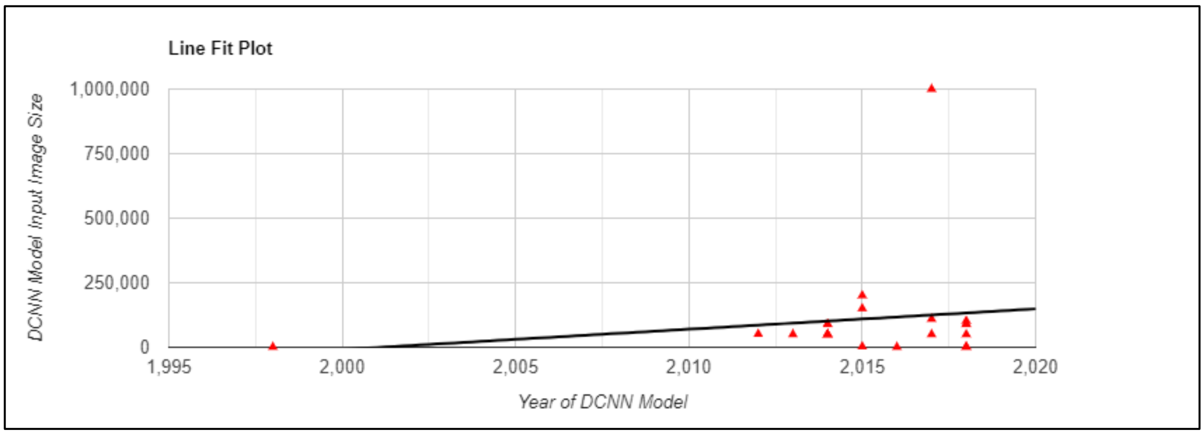

Figure 3: The Pearson correlation coefficient (r) between year of DCNN model and DCNN model input image size

Likewise, the Pearson correlation coefficient (r) between year of DCNN model and parameters number of DCNN model is presented in Figure 4. The Pearson correlation coefficient (r) is equal to 0.1115 that is a positive correlation, which indicates that parameters number of DCNN model boost with the years advances.

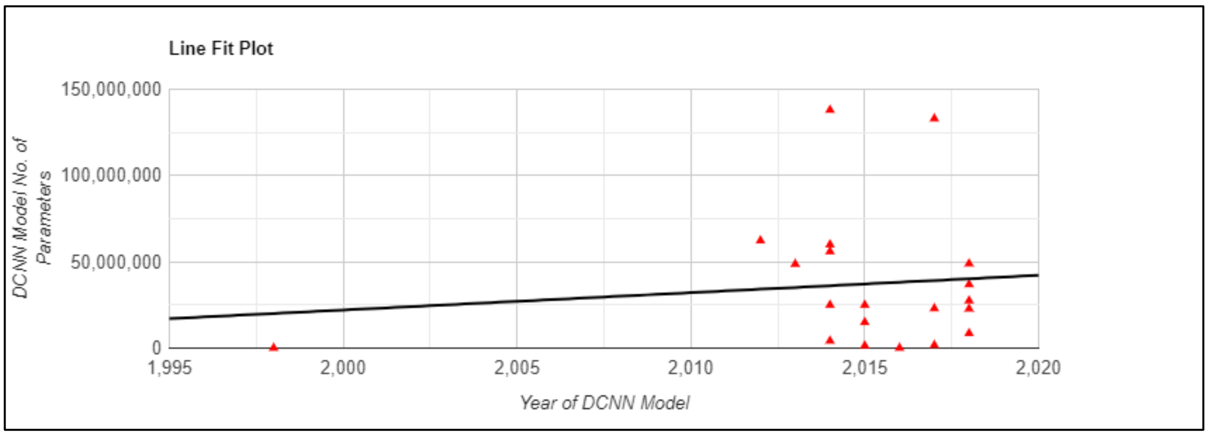

Figure 4: The Pearson correlation coefficient (r) between year of DCNN model and parameters number of DCNN model

In addition, the Pearson correlation coefficient (r) between DCNN model input image size and number of DCNN model layers is presented in Figure 5. The Pearson correlation coefficient (r) is equal to 0.1093 that is a positive correlation, which indicates that DCNN model input image size increases with a raise of number of DCNN model layers.

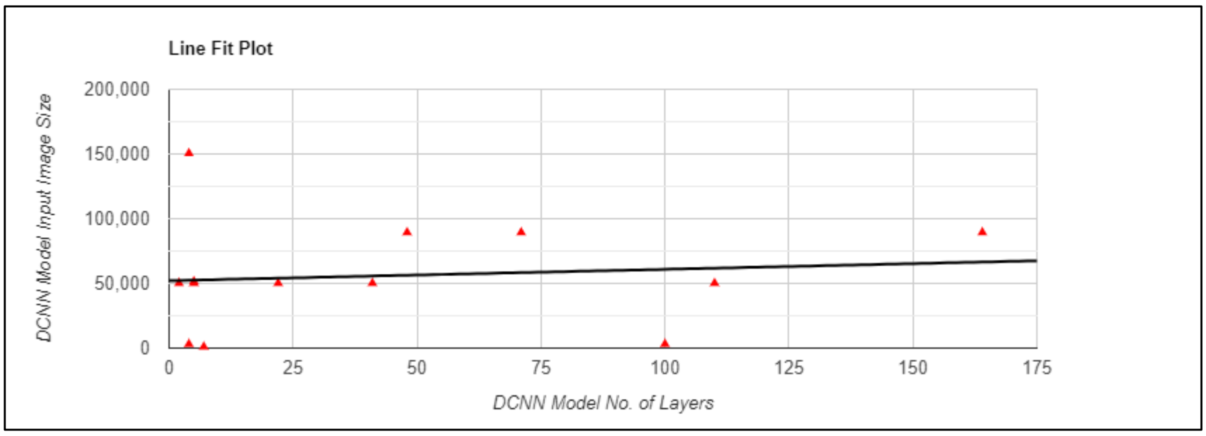

Figure 5: The Pearson correlation coefficient (r) between DCNN model input image size and number of DCNN model layers

Also, the Pearson correlation coefficient (r) between number of DCNN model layers and parameters number of DCNN model is presented in Figure 6. The Pearson correlation coefficient (r) is equal to 0.01436 that is a positive correlation, which indicates that number of DCNN model layers increases with an advance of parameters number of DCNN model.

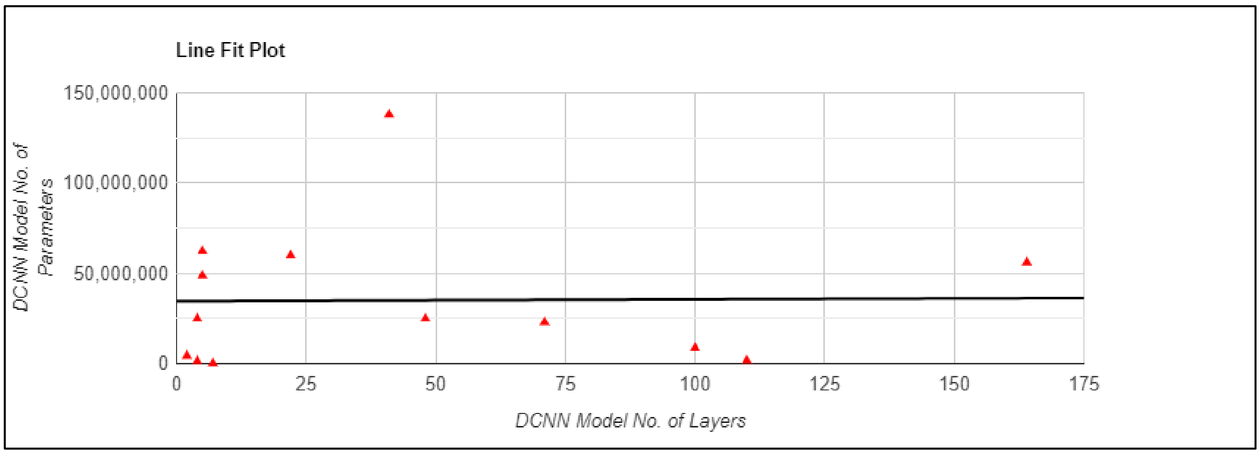

Figure 6: The Pearson correlation coefficient (r) between number of DCNN model layers and parameters number of DCNN model

Finally, the Pearson correlation coefficient (r) between DCNN model input image size and parameters number of DCNN model is presented in Figure 7. The Pearson correlation coefficient (r) is equal to -0.03091 that is a negative correlation, which indicates that DCNN model input image size decreases with a raise of parameters number of DCNN model.

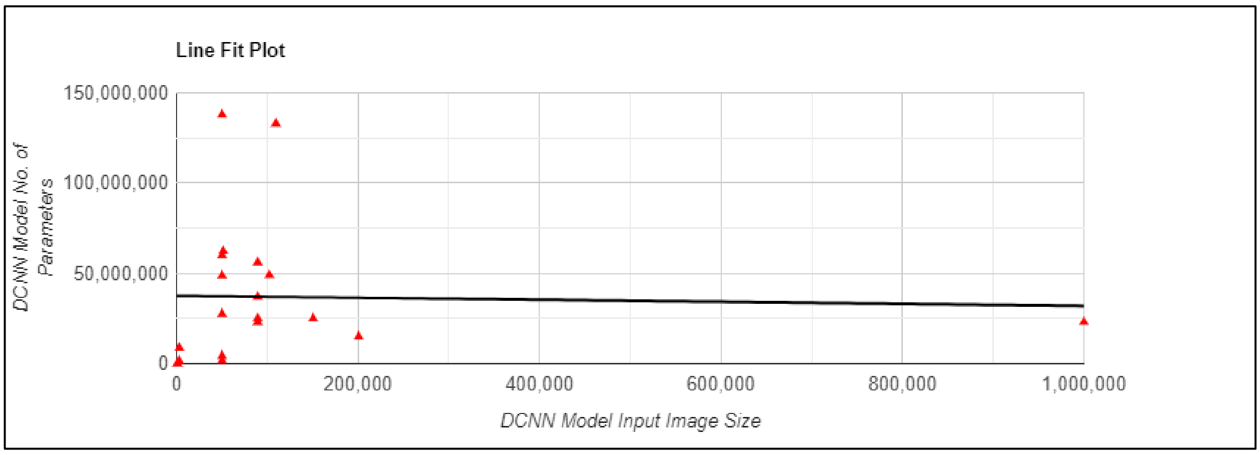

Figure 7: The Pearson correlation coefficient (r) between DCNN model input image size and parameters number of DCNN model

V. CONCLUSION AND FUTURE DIRECTION

In this research, several deep convolutional neural network (DCNN) models were reviewed in order to explore and investigate the current situations and state-of-the-art for recognizing images by using deep convolutional neural network (DCNN) models. Consequently, the review summarized the attributes of 22 deep convolutional neural network (DCNN) models for image recognition according measures, such as year of DCNN model, the usage of DCNN model, number of DCNN model layers, DCNN model input image size, and parameters number of DCNN model. Subsequently, the Pearson correlation coefficient (r) was utilized to measures statistically the degree to which two variables move in relation to each other.

The Pearson correlation coefficient (r) between year of DCNN model and number of DCNN model layers is a positive correlation, which indicates that number of DCNN model layers increase with the years advances. In the same way, the Pearson correlation coefficient (r) between year of DCNN model and DCNN model input image size is a positive correlation, which indicates that DCNN model input image size raises with the years advances. Equally, the Pearson correlation coefficient (r) between year of DCNN model and parameters number of DCNN model is a positive correlation, which indicates that parameters number of DCNN model boost with the years advances. In addition, the Pearson correlation coefficient (r) between DCNN model input image size and number of DCNN model layers is a positive correlation, which indicates that DCNN model input image size increases with a raise of number of DCNN model layers. Also, the Pearson correlation coefficient (r) between number of DCNN model layers and parameters number of DCNN model is a positive correlation, which indicates that number of DCNN model layers increases with an advance of parameters number of DCNN model. In the contrary, the Pearson correlation coefficient (r) between DCNN model input image size and parameters number of DCNN model is a negative correlation, which indicates that DCNN model input image size decreases with a raise of parameters number of DCNN model.

In future work, this research recommends reviewing and investigating more DCNN models for image recognition, especially DCNN models have been proposed after 2018. In addition, it recommends studying effectiveness of large scale and high performance models on DCNN models for image recognition.

REFERENCES:

[1] M. Achparaki *et al.*, "Advancements in Deep Learning Theory and Applications," *Intech*, p. 13, 2012, [Online]. Available: http://dx.doi.org/10.1039/C7RA00172J%0Ahttps://www.intechopen.com/books/advanced-biometric-technologies/liveness-detection-in-biometrics%0Ahttp://dx.doi.org/10.1016/j.colsurfa.2011.12.014

[2] L. Alzubaidi *et al.*, *Review of deep learning : concepts , CNN architectures , challenges , applications , future directions*. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.

[3] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A Survey of the Recent Architectures of Deep Convolutional Neural Networks," 2019.

[4] L. Li, "Application of deep learning in image recognition," *J. Phys. Conf. Ser.*, vol. 1693, no. 1, 2020, doi: 10.1088/1742-6596/1693/1/012128.

[5] Purwono, A. Ma'arif, W. Rahmaniar, H. I. K. Fathurrahman, A. Z. K. Frisky, and Q. M. ul Haq, "Understanding of Convolutional Neural Network (CNN): A Review," *Int. J. Robot. Control Syst. Vol 2, No 4*, vol. 2, no. 4, pp. 739–748, 2023, doi: 10.31763/ijrcs.v2i4.888.

[6] Yann LeCun, Leon Bottou, Yoshua Bengio, and And Patrick Haffner, "LeNet," 1998.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, Jun. 2017, doi: 10.1145/3065386.

[8] C. Szegedy *et al.*, "Going Deeper with Convolutions," Sep. 2014, [Online]. Available: http://arxiv.org/abs/1409.4842

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, [Online]. Available: http://arxiv.org/abs/1512.03385

[10] A. G. Howard *et al.*, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," Apr. 2017, [Online]. Available: http://arxiv.org/abs/1704.04861

[11] M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," Nov. 2013, [Online]. Available: http://arxiv.org/abs/1311.2901

[12] B. Kang, Y. Lee, and T. Q. Nguyen, "Depth Adaptive Deep Neural Network for Semantic Segmentation," Aug. 2017, doi: 10.1109/TMM.2018.2798282.

[13] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway Networks," May 2015, [Online]. Available: http://arxiv.org/abs/1505.00387

[14] S. Zagoruyko and N. Komodakis, "Wide Residual Networks," May 2016, [Online]. Available: http://arxiv.org/abs/1605.07146

[15] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Sep. 2014, [Online]. Available: http://arxiv.org/abs/1409.1556

[16] X. Zhang, Z. Li, C. C. Loy, and D. Lin, "PolyNet: A Pursuit of Structural Diversity in Very Deep Networks," Nov. 2016, [Online]. Available: http://arxiv.org/abs/1611.05725

[17] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," Feb. 2016, [Online]. Available: http://arxiv.org/abs/1602.07261

[18] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely Connected Convolutional Networks," Aug. 2016, [Online]. Available: http://arxiv.org/abs/1608.06993

[19] D. Han, J. Kim, and J. Kim, "Deep Pyramidal Residual Networks," Oct. 2016, [Online]. Available: http://arxiv.org/abs/1610.02915

[20] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," 2018.

[21] F. Wang *et al.*, "Residual Attention Network for Image Classification," Apr. 2017, [Online]. Available: http://arxiv.org/abs/1704.06904

[22] J. Zang *et al.*, "Attention-based Temporal Weighted Convolutional Neural Network for Action Recognition," Mar. 2018, [Online]. Available: http://arxiv.org/abs/1803.07179

[23] Y. Shi, M. Wang, S. Chen, J. Wei, and Z. Wang, "Transform-Based Feature Map Compression for CNN Inference," 2018.

[24] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, "Squeeze-and-Excitation Networks," Sep. 2017, [Online]. Available: http://arxiv.org/abs/1709.01507

[25] Y. Hu, G. Wen, M. Luo, D. Dai, J. Ma, and Z. Yu, "Competitive Inner-Imaging Squeeze and Excitation for Residual Network," Jul. 2018, [Online]. Available: http://arxiv.org/abs/1807.08920

[26] C. Yeo, P. Ahammad, K. Ramchandran, and S. S. Sastry, "High-speed action recognition and localization in compressed domain videos," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 8, pp. 1006–1015, 2008, doi: 10.1109/TCSVT.2008.927112.