JCSE International Journal of Computer Sciences and Engineering Open Access
Survey Paper Vol.-7, Issue-2, Feb 2019 E-ISSN: 2347-2693

A Brief Overview of Developing Convolutional Neural Network Using Genetic Algorithm

Mudasir Ali Lone[1*], Mohammad Islam[2]

[1,2]Department of CS&IT, Maulana Azad National Urdu University, Hyderabad, India

[*]Corresponding Author: aleemudasir@gmail.com, Tel.: +91-8297466309

DOI: https://doi.org/10.26438/ijcse/v7i2.812818 | Available online at: www.ijcseonline.org

Accepted: 25/Feb/2019, Published: 28/Feb/2019

Abstract—This paper presents an overview of developing Convolutional Neural Network using Genetic Algorithm. CNNs have been quite popular for image recognition and classification problems, but developing and training a CNN is a time-consuming and computationally costly and complex process. In this paper we discuss and review various GA based methods used for automatically generating and developing CNN networks and optimizing their networks for various pattern recognition problems and various tasks on image datasets. This paper looks at how using genetic approach for developing a network reduces its computational complexity compared to the traditional methods and increases the efficiency and accuracy of the network while also making the training process easier. We look at the genetic encoding used to generate a network and perform its evolution. A general survey of developing CNNs using GA is presented in order to understand the improvement in performance achieved through the given method. We look at the relative performance of CNNs developed through genetic approach and make a general comparison with the ones produced manually.

Keywords— *Convolutional Neural Network (CNN), Genetic Algorithm (GA), Neural Networks, Pattern Recognition, Image Classification, Structure Learning, Deep Learning.*

I. INTRODUCTION

One of the main issues concerning Neural Networks is to choose the most efficient network model for the particular data set. All the training and testing repeatedly goes into the same thing i.e. to decide the best fit model for our data. It's a very time consuming and a very hectic process. As the number of layers and number of neurons in each layer increase, the complexity of this process also increases. And in such scenarios brute force methods don't appeal much [1]. Going through each permutation on a deep network means going through millions of possibilities and then tweaking through those permutations to search for such neural network parameters that are most suitable for our data after every iteration is a highly complex job [2]. There has been development of many methods to overcome the problem of training a Neural Network, one of the very well-known methods is the backpropagation method where the error is back-propagated to previous layers in order to minimize the error. The error can be back-propagated in the form of Mean Square Error, Root Mean Square Error, etc. The aim of backpropagation is to change the parameters of the Neural Network in order to minimize the output error. While backpropagation is good for tuning the parameters to reduce error and helps in the learning process of Neural Network but the issue lies again in the high number of permutations that have to be skimmed though and the fact that backpropagation comes into play only after most of the Neural Network has been already designed, thus it doesn't help decrease the complexity process. That's where Genetic Algorithm can come into play. Genetic Algorithm is a directed randomized search technique. Genetic Algorithms have been used to train Neural Networks for a long time now, they are quite efficient when it comes to training Neural Networks [2], [3]. Genetic Algorithms are based on the evolutionary process where an individual tries to get better after every generation so that it's best coped to deal with the environment, and similarly in the case of Neural Networks it helps them to achieve the maximum efficiency level during the training process for various data sets [4] [5]. The aim of this paper is to provide a general overview of the Genetic Algorithm-Convolutional Neural Network hybrid techniques used in [1], [2], [3] to get better network structure in case of various Neural Network and Deep Learning problems [6].

The remaining sections of paper are organized as follows. First, the methods and techniques used in this paper are explained in Section II. Then, various GA-CNN architectures are discussed in section III. And finally, the conclusion and future work in this field is presented in Section IV.

II. METHODS AND TECHNIQUES

A. Convolutional Neural Networks (CNNs)

CNNs have been very efficient when it comes to image processing and pattern recognition, they have been employed with deep layers to extract various features within an image [7], [8], [9], [10], [11], [12]. CNNs consist of various layers like convolutional layer, pooling layer, fully connected layer. We can view CNN as a composite function trained by back-propagation of error signals, where the value of error signal is defined by the difference in the values of expected and predicted layers at the final layer output. CNNs can also be trained through transfer learning, where features from pre-trained networks can be extracted and then generalized to be used for other recognition problems [3]. Convolutional layer slides across the input to extract features from the input vector. Each convolutional layer consists of a set of filters. In pooling layer, max pooling is generally used. Pooling layer is used to reduce the dimensions of input for the next layer. Pooling layer performs sub-sampling of the inputs by selecting the maximum/minimum/average value in each region and just passing that value to the next layer. Unlike the pooling and convolutional layers, a fully connected layer consists of exhaustive connections of the current layer with the next one [2]. A fully connected layer works in the same manner as a multilayer perceptron, in this layer each input node is directly connected to each output node with each of these connections having their respective weights. Weights of these connections are optimized through back-propagation. Fully connected layers bring non-linearity in the network and are added at the end of the CNN. CNNs consist of filters that are used to magnify the properties of pixels and they're found in both pooling as well as convolutional layers [13], [14], [15]. When it comes to choosing an efficient structure of CNN for a particular problem it gets tricky. Training can be usually accomplished by backpropagation based on various error functions based on difference between the expected and predicted results. We can start from scratch and give random structure to the CNN and then similarly check for other structures to find the most efficient one. CNN models have been largely used for classification and recognition of images. CNN was originally proposed by Y. LeCun et al [16] for image recognition. The network proposed by them was called LeNet-5. It consisted of two convolutional layers where each layer was followed by a sub-sampling layer and finally ended with a fully connected layer. CNNs have been used in image classification where generating feature descriptors in order to distinguish between images is necessary. CNNs have become quite popular in research and for their various applications in the industry [7]—[12]. The size of various layers in a CNN impacts the level of details that can be recognized by the CNN. The number of representation of an image during the convolutional process is controlled by the hyper-parameters. The issue with CNNs is the training time required for CNNs to come up with a network structure that's most optimal for different problems. CNNs have been used in many applications like scene recognition, action recognition, object detection, face recognition and so on [17], [18]. While CNNs provide best solutions for various image classification problems, but constructing optimal network is still a challenge, as it's time consuming and computationally costly due to the vastness of search space.

B. Genetic Algorithm

Genetic Algorithms are search functions that have been actually inspired by the evolutionary natural selection process. GAs are used to generate solutions to different optimization and search problems. They have been efficiently employed to find the optimum results in searching problems. GAs are stochastic random search techniques. GAs are very efficient when it comes to finding the solutions to search problems in form of local maxima or local minima [4]. There are two pre-requisites for a genetic algorithm, we need a genetic representation or genetic encoding for the solution domain and we need a fitness function to evaluate the individuals generated by the genetic process. The main idea of genetic algorithm is to let individuals evolve through genetic operations. GA initially consists of population of individuals (chromosomes). Each chromosome represents an individual or a solution. These chromosomes are evaluated by fitness function and replaced by stronger off-springs produced through crossover and mutation in each generation [19], [20]. Crossover and mutation are considered two of the basic genetic operators. There are various mechanisms to implement crossover operator, like heuristic crossover, multipoint crossover, two-point crossover, arithmetic crossover. And there are various mechanisms to implement mutation operator as well, like uniform mutation, boundary mutation, non-uniform mutation. The two basic requirements of a genetic algorithm are the genetic representation or genetic encoding of the domain of solution and the fitness function required to evaluate each individual [1], [2]. The basic idea of genetic algorithm is to evolve individuals through various genetic operations. Genetic operations like selection, mutation, crossover are commonly used and then parameters like fitness function are used to evaluate the outputs produced in various generations. Selection operation is used to select the best-performing individuals on the basis of their fitness score, crossover function creates new off-springs based on the selected parents, and mutation makes some minor changes in the genetic representation of parents to introduce newer and diverse individuals in the population [4], [21].

C. GA-CNN Implementation

GA with CNN has been employed by various researchers to find the efficient structure of the CNN for a particular pattern recognition problem [1], [2], [3]. The design of a CNN network can be expressed as a search problem and formulated into a genetic encoding that's used to search for the optimum result in the form of a maxima or a minima. We have seen newer genetic operators and newer genetic encoding methods being introduced that have helped us in efficient representation of the search space and an efficient execution of the evolutionary search strategy [1], [2]. GA can also help to use the already existing well-performing CNN structures and employ them on newer data sets through transfer learning while evolving their structures to fit the needs of the new datasets like this [3]. It has been observed that various CNN networks perform efficiently for different datasets and different tasks. Thus GA can be used to choose the most optimal network for a particular problem or a particular dataset automatically [21], [22]. The list of parameters that are considered while constructing a CNN are: activation functions, total number of network layers (depth of the network), number of convolutional layers, number of pooling layers, size of convolutional filters, size of pooling layer, number of convolutional filters in each convolutional layer and so on. Using GA with Neural Networks for learning the network structure has been under research for some time now. Researchers have paid quite an attention to using GA-based methods for optimizing networks for pattern recognition [23]—[26]. Leung et al [21] had proposed a method for tuning of both network structures and parameters of a neural network. In recent years, the possibility of learning network structures, hyper-parameters and parameters in deep learning using GA has also been studied by various researchers [20], [21], [22]. GA can be used to optimize various parameters of a CNN that control its network structure. GA has been used for developing and evolving different features/parameters that affect the development of network structure of a CNN [19]. GA usually compares various CNN by training them on different image datasets that contain the respective pattern recognition problem. After some specified generations of the GA, the optimal CNN for the given problem is chosen as the output of the evolutionary process. GA has been used to overcome the difficulty of constructing an accurate CNN network by optimizing the network parameters. The existing research where GA has been used to improvise the CNN results are the following three: Network-structure optimization, hyper-parameter optimization and loss function optimization [1], [2], [3]. In hyper-parameter optimization, GAs are used to optimize network parameters like learning rate, number of layers in convolution, the optimization function type used [20]. Structure optimization uses GA to optimize the size of convolutional layers that are used in the given network [21], [22]. Loss function optimization is done to optimize an existing loss function using GA.

When it comes to evolving the structure of a CNN with the help of GA, we need a coding technique to represent the structure of a code, like binary code. Connections between various components of a CNN like Convolutional or Pooling layers can be easily shown in binary representation. This technique has been efficiently employed in the forth-discussed works [1], [2], [3].

III. GA-CNN ARCHITECTURES

A. Genetic-CNN

Lingxi et al have proposed a Genetic CNN where they propose to learn the network structure of a CNN automatically instead of having to start from the scratch [2]. The possible number of network structures in a CNN highly increases with the increase in the depth of layers in the network, therefore they have proposed to use the genetic algorithm to go through the search space of probable networks and find the one that fits the requirement of the dataset or the given task. In Genetic CNN, they have used a binary string of fixed length to represent a network structure. The genetic encoding of structures is required in order to make the structures go through the evolutionary process. The genetic algorithm is initialized by creating a set of various randomized individuals and then use standard genetic operations like selection, crossover, and mutation that have been defined as per the specific requirement and set in order to kill the weaker individuals and let the fittest survive in each generation and also attempt to generate newer competitive individuals [2]. For each individual from the total population, its competitiveness is defined by the value of fitness function and it is equivalent to the recognition accuracy of each individual on the given dataset. The recognition accuracy of the individuals during evolutionary process is obtained by training the individual network produced by each generation of GA on the given dataset and then after that evaluating the final produced network on a validation set of the given data before going in for the final tests. In the proposed Genetic CNN, the process has been executed on two relatively smaller datasets like MNIST and CIFAR-10 and final results have been shown in terms of recognition accuracy and it has been observed that the proposed method is able to search high-quality network structures and further evolve them [2]. The genetic process in the proposed work starts initially with a population of N chromosomes/individuals where each individual represents a CNN network structure. Then the genetic process evolves the given population through a certain number of generations and in every generation the whole population goes through selection, mutation and crossover operations. At the beginning of every generation, selection process is performed to select the most fit individuals and in the proposed work, fitness of each individual has been defined as the recognition rate. The probability of an individual being selected has been made directly proportional to its fitness value. The proposed evolutionary process starts with an initial population of 20 individuals. In the proposed work, Russian Roulette process has been utilized to determine which individuals will survive, which ensures that best individuals survive and the worst ones are eliminated [2]. Mutation involves flipping of each bit of an individual independently with a probability of $Q_m = 0.1$. Q_m is often kept small so that the individual is not changed more than required and the diversity is also maintained. Whereas crossover process changes two individuals at the same time with the probability $P_c = 0.2$ by exchanging bits of chromosomes of the selected parents with each other to create an offspring that has properties inherited from its parents. Then each individual that survives at the end of each generation is evaluated in order to obtain the value of fitness function. The evaluation of surviving individuals is done on a given dataset by training them from scratch. Then the process continues from the selection again for the next generation till 50 rounds [2].

Genetic CNN aims to use the proven ability of CNNs in visual recognition with the well-known ability of genetic algorithms to find optimum global search results for various search problems in order to find an optimum structure for the given CNN that's best suited for a particular database, thus easily circumventing the issue that's faced while designing the structures of CNN while we tend to go deeper and deeper with the layers.

For CIFAR10 dataset, they have started with an initial generation of 20 individuals and then run the genetic process for a total of 50 generations [2]. The results of test experiments of the proposed method for CIFAR10 dataset are shown in Table I.

Table 1 CIFAR10 test set Recognition Accuracy [2]

Gen	Max %	Min %	Avg %	Med %	Std-D
00	75.96	71.81	74.39	74.53	0.91
05	76.24	72.60	75.32	75.65	0.89
10	76.72	73.92	75.68	75.80	0.88
30	76.95	74.38	76.42	76.53	0.46
50	**77.06**	**75.34**	**76.58**	**76.81**	**0.55**

Genetic CNN has also shown that structures developed for smaller datasets can be easily transferred to larger datasets like ILSVRC2012 with some required adjustments and achieves better performance than VGGNet [2].

In the proposed framework, genetic algorithm with CNN has been used only to explore the network search space and find the optimum network structure and there's an interesting scope to utilize the genetic algorithm not just for producing the optimal network structure but for both generating the network and training the weights of the generated network simultaneously. The structures obtained in the proposed genetic process have seen observed to have similarity to the DenseNet and the DenseNet has heavier computational overhead than the proposed network, therefore it can be been concluded that genetic algorithm helps to find more efficient network structures than DenseNets [2]. In this work, Genetic Algorithm has only been used for proposing newer network structures, the parameters of each GA proposed network structure and the accuracy in classification has been obtained through different stand-alone training-from-scratch methods. Every network structure generated through genetic process is trained from scratch and the fitness/quality of the individuals is ascertained by their recognition accuracy on given datasets. It has been observed that the structures generated through GA perform better than those that have been manually designed [2].

B. Automatic CNN Selection Using GA

Haiman Tian et al [3] proposed a new approach on the basis of genetic algorithm in order to select or regenerate the optimum CNN model for given datasets from a given list of pre-trained CNN model. They have proposed a genetic encoding model to denote different pre-trained models in the given network population. When the evolutionary process of genetic algorithms is executed for the given network model representations, the genetic code that represents the best model gets selected or even new model versions are produced. This reduces the computational cost, as instead of the need to start the whole process of training the network from the scratch we can employ transfer learning to use pre-trained models for different newer tasks and datasets [3]. Transfer learning helps to avoid the problem of having to set from scratch the millions of parameters in deep learning network structures by instead choosing an already developed network and further optimizing it for a new dataset or a new task. In the proposed method, the same has been done and the proposed method employs GA to improve the performance of CNN for image classification problems through transfer learning.

Table 2 Evaluation Result on CIFAR10 Dataset [3]

Models	Precision	Recall	AvgW F1	Avg F1
MobileNet	0.446	0.083	0.446	0.14
VGG16	0.010	0.100	0.018	0.018
ResNet50	0.471	0.09	0.469	0.15
Incep-v3	0.502	0.102	0.503	0.169
Proposed	**0.651**	**0.169**	**0.648**	**0.268**

The evaluation results of the proposed framework on CIFAR 10 dataset are shown in Table II where Precision, Recall, weighted average F1 score, and average F1 score are taken into account. The number of epochs in the experiment have been set to 1200 for training the network [3].

They have started with four different pre-trained models as the original population and have then used GA to automatically select the most efficient model for the required dataset from the given population. They have proposed a new encoding model to represent the give pre-trained models. Using GA they select the optimal CNN model and then extract the respective network features from the selected model. The proposed work focuses on two important things. First, creating a genetic-encoding model that's aimed at improving the process of finding the optimal network structures. Second, an adaptive neural network in order to handle both balanced and imbalanced datasets [3]. During the genetic process a set of pre-trained CNN models are selected in which each individual represents the set of possible CNN combinations that will be further utilized to generate a new network. The genetic-code representation of each of the individuals present in the set has been done in binary like 1001, where 0 represents that a model is not selected and 1 represents that a model has been selected [3]. The fitness function of each individual in the given set is used to determine whether a particular individual will be selected as a parent that contributes in production of next generation off-springs through crossover or not. In the proposed method, they have also used SVM classifier to validate each individual in the population in order to reduce the computational complexity of the final validation process and to ensure that the final output is more reliable. The proposed method used MobileNet, VGG16, ResNet50 and Inception-V3 as its four pre-trained CNN models and used it on three different datasets namely Disaster, Network Camera 10K and CIFAR-10 wherein it has shown considerable improvement in performance of proposed model for the Network Camera 10K and CIFAR-10 datasets while it has shown similar performance as that of ResNet50 for Disaster Dataset [3].

C. GA-CNN for Image Classification

In a paper on convolutional neural network structure optimized through genetic algorithm for concrete crack detection by Spencer Gibb et al [1], they have proposed a method to utilize GA to evolve various parameters of that affect the structure of a CNN. The GA used in the proposed work is utilized in order to compare the CNNs by training them on a dataset that consists of images of concrete cracks. The GA goes through several generations and then chooses the best CNN suited for the problem and makes its comparison with the state-of-art CNN network for crack detection [1]. Through the proposed work, they have shown that it's possible with the help of GA to generalize the job of optimization of the structure of a CNN for classification of images. They have described concrete crack detection as a problem of image classification where the traditional image analysis fails due to certain difficulties encountered during the crack detection process, like background clutter, illumination, and the shape of cracks [1]. Therefore to choose a CNN structure that performs well for various similar scenarios is quite a challenging and a very time-consuming process, as there are different features and parameters that affect the structure of the network. To overcome this challenge, GA has been utilized to search and traverse through the set of possible structures of the network. It has been shown that deep learning and CNNs provide the optimum output and results for crack detection and other similar image classification problems, but to construct an optimal network is a time-consuming task because of the large size of the network search space.

Table 3 Result on tests performed on the two networks by [1]

Network	Accuracy	False Positives	False Negatives
Earlier Network	0.888	481	64
Proposed Network	0.94	280	14

In the proposed work, their objective was to develop an accurate CNN by optimizing the network parameters in order to eliminate the difficulty encountered in constructing an efficient network. In the proposed research, they have applied GA to a variable depth CNN i.e. GA is used to evolve, modify and update the depth of network, size of network layers, and hyper-parameters of the network [1]. The individuals in the given population for GA are represented by 14 bits. To evaluate the given individuals, a convolutional neural network is constructed on the basis of the genetic chromosome of the individual. The number of convolutional bits have been represented by 4 bits, max-pooling layers by 0 bits, size of convolutional filters and number of convolutional filters by 5 bits each, and it's these aforementioned parameters that have been evolved by utilizing GA in the proposed work. The fitness function has been defined as, $f = c \div t$, where the number of correct classifications are represented by c, and the total classifications are represented by t [1]. In the proposed work, the training data used for the development of the CNN consisted of 3000 images with 1500 each belonging to both 'crack' class and 'no crack' class. The GA uses mutation, crossover and random choice are used to generate offsprings. Random choice is used in case mutation and crossover doesn't happen, wherein a random parent is selected from the population to generate an offspring. In the proposed work, they have run GA for 10 generations with the 30 as the population size of each generation [1]. By generation 9, it has been shown that the maximum fitness of proposed method goes up-to 89.17% and that is 9% higher than the earlier CNN network for crack detection proposed by Y.-J. Cha et al [27]. To make a detailed comparison, they tested both the proposed GA based network and the earlier network on 10 test images that had been split into 256 pixel by 256 pixel sub-images. The result is shown in Table III. From the proposed research, it was noted that going deeper is not always better, as it was seen in this research that networks with just 7 to 11 convolutional layers performed better than the networks that had 16 convolutional layers. It was also shown that having more filters at each layer produces better results than having less filters, as higher the number of filters, deeper is the representation of image features.

Table 4 Parameters of the optimal network obtained from GA [1]

Parameter	Value
No. of Conv Layers	11
No. of Filters / Layer	20
Size of Filters	15 by 15

The most important outcome from this research was that the networks that had to be constructed through GA in order to find the optimal network was just 300 while as the total number of networks needed to be constructed through an exhaustive search would be approximately 16,000. It was also noted that having odd filter sizes produced better results than even sized filters, because of the convolutional operation that needs a filter centered in the pixels of input image. The GA-based method is used for automatically searching for the network structures that are most needed and most-optimal for the given image classification problem. It has been seen that the networks generated through GA perform pretty good when they are used for crack detection in images of concrete and their results have been shown to be comparable to state-of-the-art networks and even better than them [1]. We can generalize the process used for concrete crack detection to various kinds of image datasets in order to increase the efficiency/performance of already existing convolutional neural networks for the given image classification problems by optimizing and evolving their network structures.

IV. CONCLUSIONS AND FUTURE SCOPE

Designing a CNN is a highly complex job as each parameter and hyper-parameter affects the performance of the network and getting each of the details right is a very tedious job and needs highly level of domain-expertise, therefore GA makes it largely easy for researchers to develop the CNNs automatically for a given dataset or for a given pattern recognition task. As we have seen GA makes quite an improvement over the traditional brute-force method when it comes to choosing the optimal network structure for a CNN for a given dataset [1]. GA provides an efficient way to automatically select the best model for different tasks. GA comes with the benefits of global exploration of the given search space, fast-convergence and robustness [4]. As the search for network structures is automatically done, it also reduces the need for a deep knowledge of the features of the network and the design process of neural networks [3]. We have seen that the networks produced through GA perform better than the already existing structures for the given problems/datasets [23]—[26]. The networks produced through GA while performing better are also shown to be less computationally expensive and less complex that those that are manually designed.

In future, GA can be used to determine more parameters than the ones discussed in the above papers and also used to optimize the weights of network connections. GA can be used to vary the range of filter counts for all convolutional layers. We can utilize GA to create varying combinations of different CNN layers to find the optimal combination. GA can also be used to choose an optimal activation function and optimizer for the given network. In the above discussed works, GA has been mostly used for exploring the network structure whereas the training of the network-weights has been done separately. In future, we can use GA to both explore the network structure as well as train the weights of networks at the same time. There has to be improvement in genetic encoding so that all the features and parameters of a given network can be represented efficiently and further improved through GA. Strictly speaking, the above discussed works in this paper can be classified as semi-automatic, as there’s still some manual work needed in each one of them in different parameters. In future, researchers can focus on developing fully-automatic GA-CNN frameworks where there’s no need at all of manual work or manual tuning of parameters. There’s also an interesting future scope for developing improved GAs that are particularly well-suited for representation of CNNs through genetic encoding, exploration of CNN network structure and CNN training problems.

REFERENCES

[1] S. Gibb, H. M. La, S. Louis, “*A Genetic Algorithm for Convolutional Network Structure Optimization for Concrete Crack Detection*,” in IEEE Congress on Evolutionary Computation, pp. 1-8, 2018.
[2] L. Xie, A. Yuille, “*Genetic CNN*,” in IEEE International Conference on Computer Vision, pp. 1388-1397, 2017.
[3] H Tian, S Pouyanfar, J Chen, S.C. Chen, S. Iyengar, “*Automatic Convolutional Neural Network Selection for Image Classification Using Genetic Algorithms*,” in IEEE International Conference on IRI, pp. 444-451, 2018.
[4] Mitchell Melanie, “*An Introduction to Genetic Algorithms*,” The MIT Press, pp. 49-62, 1996.
[5] Le Zhang , P.N. Suganthan, “*A survey of randomized algorithms for training neural networks,”* Elsevier Information Sciences, Vol. 364, pp. 146-155, 2016.
[6] Jürgen Schmidhuber, “*Deep learning in neural networks: An overview*,” Elsevier Neural Networks, Vol. 61, pp. 85-117, 2015.
[7] Tong He, Weilin Huang, Yu Qiao, and Jian Yao, “*Text-Attentional Convolutional Neural Network for Scene Text Detection*”, IEEE Transaction on Image Processing, Vol. 25, Issue 6, pp. 2529-2541, 2016.
[8] Marco Grond, Willie Brink, and Ben Herbst, “*Text Detection in Natural Images with Convolutional Neural Networks and Synthetic Training Data*,” in PRASA-RobMech International Conference, 2016.
[9] H. Lit, Z. Lin, X. Shen, J. Brandt, G. Huat, “*A Convolutional Neural Network Cascade for Face Detection*”, in IEEE Conference on CVPR, pp. 5325-5334, 2015.

[10] H. Liu, X. Shen, H. Ren, "*FDAR-Net: Joint Convolutional Neural Networks for Face Detection and Attribute Recognition*", in 9th International SCID, pp 184-187, 2016.

[11] C. Zhang, Z. Zhang, "*Improving Multiview Face Detection with Multi-Task Deep Convolutional Neural Networks*", in IEEE WCACV, pp. 1036-1041, 2014.

[12] Kaipeng Zhang, Zhanpeng Zhang, Zhifeng Li, and Yu Qiao, "*Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks*", IEEE Signal Processing Letters, pp. 1499-1503, 2016.

[13] C. Nebauer, "*Evaluation of convolutional neural networks for visual recognition*", IEEE transactions on Neural Networks,Vol. 9, Issue 4, pp. 685-96, 1998.

[14] J. Schmidhuber, "*Deep learning in neural networks: An overview*", Elsevier Neural Networks, Vol. 61, pp. 85-117, 2015.

[15] W. Liu, Z. Wanga, X. Liu, N. Zeng, Y. Liu,, F. E. Alsaadi, "*A survey of deep neural network architectures and their applications*", Neurocomputing, Vol. 234, pp. 11-26.

[16] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "*Gradient-based learning applied to document recognition*", Proceedings of the IEEE, Vol. 86, no. 11, pp. 2278-2324, 1998.

[17] Christophe Garcia, Manolis Delakis, "*Convolutional Face Finder: A Neural Architecture for Fast and Robust Face Detection*", IEEE Trans. on PAMI, Vol. 26, pp. 1408-1423.

[18] Zhan Wu, Min Pengl, Tong Chen, "*Thermal Face Recognition Using Convolutional Neural Network*", in International Conference on OIP, pp. 6-9, 2016.

[19] Eli (Omid) David, Iddo Greental, "*Genetic Algorithms for Evolving Deep Neural Networks*", GECCO, 2014.

[20] S.R. Young, D.C. Rose, T.P. Karnowski, S.H. Lim, R.M. Patton, "*Optimizing Deep Learning Hyper-Parameters Through an Evolutionary Algorithm*", in MLHPC, 2015.

[21] F. Leung, H.K. Lam, S.H. Ling, K.S. Tam, "*Tuning of the Structure and Parameters of a Neural Network Using an Improved Genetic Algorithm*", IEEE Trans. on Neural Networks, Vol. 14, No. 1, pp 79-88, 2003.

[22] J.T. Tsai, J.H. Chou,T.K. Liu, "*Tuning the Structure and Parameters of a Neural Network by Using Hybrid Taguchi-Genetic Algorithm*". IEEE Transactions on Neural Networks, 17, 69-80, 2006.

[23] D. Hossain, G. Capi, M. Jindai, "*Optimizing Deep Learning Parameters Using Genetic Algorithm for Object Recognition and Robot Grasping*", Journal of Electronic Science and Technology, Vol. 16, pp. 11-15, 2018.

[24] Nacéra Benamrane and Abdelkader Fekir, "*Medical Images Segmentation By Neuro-Genetic Approach*" in Ninth International Conference on Information Visualisation, pp. 981-986, 2005.

[25] Ms. K. Umamaheswari, Dr.S. Sumathi, Dr. S.N. Sivanandam, "*Neuro–Genetic approaches to classification of Face Images with effective feature selection using hybrid classifiers*", in International Conference on Advanced Computing and Communications, pp. 286-291, 2006.

[26] T.A. Tarique, M.A. Zamee, M.I. Khan, "*A New Approach for Pattern Recognition with Neuro-Genetic System Using Microbial Genetic Algorithm*", in International Conference on EE&ICT, pp. 1-4, 2015.

[27] Y.-J. Cha, W. Choi, O. Bykztrk, "*Deep learning-based crack damage detection using convolutional neural networks*", Comp.-Aided Civil and Infrastruct. Engineering, Vol. 32, pp. 361-378, 2017.