Int. J. Appl. Math. Comput. Sci., 2023, Vol. 33, No. 1, 21–31
DOI: 10.34768/amcs-2023-0002

A GENETIC ALGORITHM BASED OPTIMIZED CONVOLUTIONAL NEURAL NETWORK FOR FACE RECOGNITION

NAMRATA KARLUPIA [a,*], PALAK MAHAJAN [a], PAWANESH ABROL [a], PARVEEN K. LEHANA [b]

[a] Department of Computer Science and Information Technology
University of Jammu
Baba Saheb Ambedkar Road, 180006, Jammu, India
e-mail: namrataphonsa@gmail.com

[b] Department of Electronics
University of Jammu
Baba Saheb Ambedkar Road, 180006, Jammu, India

Face recognition (FR) is one of the most active research areas in the field of computer vision. Convolutional neural networks (CNNs) have been extensively used in this field due to their good efficiency. Thus, it is important to find the best CNN parameters for its best performance. Hyperparameter optimization is one of the various techniques for increasing the performance of CNN models. Since manual tuning of hyperparameters is a tedious and time-consuming task, population based metaheuristic techniques can be used for the automatic hyperparameter optimization of CNNs. Automatic tuning of parameters reduces manual efforts and improves the efficiency of the CNN model. In the proposed work, genetic algorithm (GA) based hyperparameter optimization of CNNs is applied for face recognition. GAs are used for the optimization of various hyperparameters like filter size as well as the number of filters and of hidden layers. For analysis, a benchmark dataset for FR with ninety subjects is used. The experimental results indicate that the proposed GA-CNN model generates an improved model accuracy in comparison with existing CNN models. In each iteration, the GA minimizes the objective function by selecting the best combination set of CNN hyperparameters. An improved accuracy of 94.5 % is obtained for FR.

Keywords: convolutional neural network, hyperparameters, genetic algorithm, deep learning, evolutionary techniques.

1. Introduction

Face recognition (FR) coupled with Artificial Intelligence tools is the next big step in the field of security. Bio-metric frameworks based on face recognition are widely used in several applications like forensics, criminal investigations, etc. (Pujol *et al*., 2016). Face detection and alignment are the primary steps in FR. Further, features are extracted from the image and matched with the database containing all the facial images. In FR, traditional algorithms like Fisherfaces, meta-face, Bayesian face were not able to resolve the challenges like resolution, pose variations, aging, etc., for detecting global and local facial features (Guo and Zhang, 2019; Raju *et al*., 2022).

Deep learning research has flourished tremendously with artificial intelligence as its backbone. Deep learning has shown great potential in the domains of computer vision, speech recognition (Dargan *et al*., 2020), multimedia, natural language processing, image quality assessment (Mahajan *et al.*, 2021), medical imaging (Patro *et al*., 2022; Mahajan *et al.*, 2021), object based classification (Mahajan *et al.*, 2020a). In classification, the use of convolutional neural networks (CNNs) has produced substantial progress. CNNs are the core persuasive models being applied in deep learning for such tasks (Liu and An, 2020). However, in spite of the considerable progress made in the field of facial recognition using deep learning, there is still room for improving the classification performance of the system by incorporating optimization in a CNN architecture.

In CNNs, hyperparameters include variables such as hidden layers, padding, the learning rate, etc. that are set before network training (Wang *et al*.,

*Corresponding author

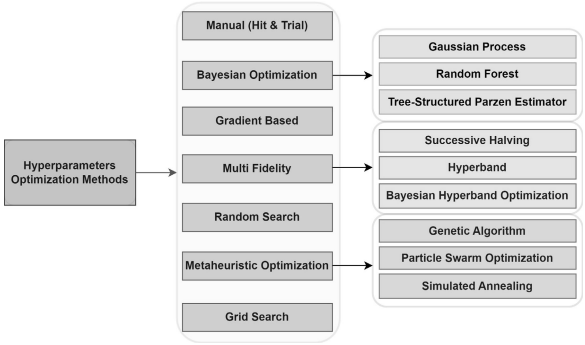


Fig. 1. Hyperparameter optimization techniques.

2019). There is no deterministic approach to find an optimal CNN configuration due to the large number of hyperparameters. They act significantly as they have direct control and impact on the performance of the training algorithm (Wu *et al*., 2019). It has been seen that a good choice of hyperparameters results in better outcomes (Vose *et al*., 2019). Hyperparameter tuning or optimization is a nonconvex optimization problem, and hyperparameters also exhibit nonlinearity interactions (Li and Abdallah, 2020). The number of hyperparameters grows exponentially and may sometimes get trapped in a local optimum (Wang *et al*., 2019).

One of the most challenging tasks in the CNN domain is to find the correct values of hyperparameters. The CNN hyperparameter optimization techniques used by researchers are mainly categorized as manual and automatic search techniques, as shown in Fig. 1. The manual search methods often deal with initializing the hyperparameters by hit or trial methods or on the basis of experience. It is the most common technique used for tuning CNNs and is very ineffective and time-consuming. The parameter combinations need to be in a structure so that there is a minimum loss or maximum accuracy in the model.

In automatic methods, grid search is a systematic process where each combination of hyperparameters is generated and evaluated on the model. It is effective for a small number of hyperparameters. As the number of parameters increases, the utilization of computational resources also rises exponentially (Bergstra and Bengio, 2012). Therefore, it is an expensive search technique that suffers from the curse of dimensionality. In a random search, the hyperparameter combinations are selected randomly in the search space. At a given execution time and resources, it tries to find better results than the grid method. Therefore, hyperparameter tuning can be framed as an optimization problem and can be represented mathematically as follows:

$$y^* = \arg\min f(y), \tag{1}$$

where $f(y)$ represents an objective function such as root mean square error and y is a set of hyperparameters in a search space Y that produces an optimum value of $f(y)$.

Since hyperparameter optimization is an NP-hard problem, there is a need for an intelligent search approach for hyperparameter tuning. Table 1 contains details regarding hyperparameter optimization techniques. Metaheuristic optimization is predominantly advantageous for optimization problems when there is no precise method or when an optimal solution is not necessary, and a good solution is enough. It does not guarantee a global optimum but is still intelligent towards discovering a decent solution for rectifying the problem in less computational time and fewer resources. Hence, in machine learning, exact methods are computationally expensive, so at present, a metaheuristic approach has been chosen.

As parameter optimization is a combinatorial problem, metaheuristic methods have proven useful in resolving complex combinatorial problems. Metaheuristic algorithms like genetic algorithms GAs and bacterial foraging optimization (BFO) algorithms (Karlupia *et al*., 2019) apply properties such as exploration and exploitation that can be used to solve optimization problems with complex multiobjective functions. In the proposed work, a hybrid framework has been created by integrating an evolutionary technique, namely a GA, with a CNN for hyperparameter optimization. With face detection being a significant discipline in computer vision, GAs have been cast-off for face recognition based classification problems aimed at hyperparameter optimization in CNN architectures. The CNN structural hyperparameters reflected for learning include kernel size as well as the number of convolutional layers and filters for each layer. The task is to tune the values of filter size and its corresponding numbers. The performance of the configured CNN has been analyzed after applying GA optimization to three configuration settings, i.e., to two, three, and four convolutional layer combinations of CNNs. The objective of this work is to explore the applicability of CNNs on nature-inspired algorithms, i.e., GAs, and for this purpose, the domain has been chosen for face recognition. The main contributions of our work are as follows:

- We proposed a hybrid approach by integrating the evolutionary technique GA with CNNs for hyperparameter optimization.

- The optimized CNN architecture has been applied for face recognition with facial images that are taken from different angles.

Table 1. Hyperparameter optimization methods.

Methods	Advantages	Disadvantages
Grid search	It is a simple technique Can be parallelized Reliable in low-dimensional spaces Explores the entire search space (Bergstra and Bengio, 2012)	Sometimes trapped in local minima Suffers from the curse of dimensionality Inefficient for high dimensional hyperparameters Wastes time in poor performing areas Computationally expensive
Random	In less time and resources tries to find better solutions than grid search techniques (Bergstra and Bengio, 2012) Can be parallelized easily for independent evaluations	Does not guarantee an optimal result Sometimes shows poor performance Not reliable for training complex models
Bayesian	Currently most promising probabilistic informed search (Betró, 1991; Victoria and Maragatham, 2021) Mostly used for functions which are nonconvex or computationally expensive to compute	Gaussian processes possess the cubic complexity capability Shows poor performance for high dimensional data Limited parallelization capacity Poor scalability
Nature inspired	Used to solve complex combinatorial optimization problems Generally, randomly hop around the search space, so better than brute force methods Perfect for problems where near-optimal solutions instead of exact results are acceptable (Gadekallu *et al*., 2021)	May not find satisfactory solutions Initial tuning of parameters can be a challenging task For complex problems computations are polynomial time bounded (Victoria and Maragatham, 2021)

- To get the most out of the features of compounded convolution layers, we have analyzed the performance of the configured CNN after applying GA optimization for 3 configuration settings, i.e., to 2, 3, and 4 convolutional layer combinations of CNNs.

The structure of the paper is as follows. A brief introduction to the literature is given in Section 2. Section 3 illustrates the GA and various steps performed during hyperparameter optimization of the CNN. A detailed description of the CNN is given in Section 4. Various steps performed for the proposed work are included in Section 5. The experimental results and the analysis made are discussed in Section 6. Conclusion and future work have been presented in the last section.

2. Related work

Researchers worldwide have worked on various soft computing as well as traditional approaches to incorporate optimization techniques in various domains. Various hyperparameter tuning approaches such as random, grid, and Bayesian have been used by researchers to find the best hyperparameter combination set that produces the best performance of deep neural networks. However, these methods are not successful in deep learning architectures that have a larger number of hyper-parameters, so nowadays population based metaheuristic techniques are being used.

Syulistyo *et al.* (2016) used particle swarm optimization (PSO) for training CNNs to optimize recognition accuracy. A handwritten dataset from the MNIST data-base was taken that consists of 70,000 data points, out of which 60,000 were used for training. The fitness function is the root mean square error between the estimated output vectors and the actual output. The results obtained by PSO are compared with CNNs and deep belief networks (DBNs), and it was concluded that the proposed method has better accuracy (95.08 %) with just 4 epochs. Chhabra *et al*. (2017) employed hybrid PSO with a CNN, which reduce the dependency on the GPU system and the number of epochs required for training the CNN. The algorithm has the capability of attaining a 3 to 4% increase in accuracy with the reduced number of epochs and a reduced hardware need for training CNNs.

Another technique to optimize the hyperparameters for an image classification problem has been presented by Suganuma *et al*. (2018) using Cartesian genetic programming (CGP). Convolutional blocks and tensor concatenation are taken as the node functions in CGP. To experiment, the CIFAR-10 data set has been considered for CNNs architecture design. A Grey Wolf Optimization (GWO) based method for the optimization of hyperparameters of CNNs is recommended by Mohakud and Dash (2021). The efficiency of the

proposed model is tested by comparing it with the performance of PSO and GAbased models using the multi-class data set, skin lesion published by the International Skin Imaging Collaboration.

A hyperparameter optimization algorithm using a Bayesian optimization technique for better performance of the CNN model on the CIFAR-10 dataset has been used by Wu *et al*. (2019). The objective function was designed using the mean squared error and results are validated using error values in the CPU and GPU. It is seen that there is a 6.2 % reduction in the error in the case of GPU. Also, an amalgamation of a Bayesian and a GA is employed for hyperparameter optimization of a CNN model by Cui and Bai (2019). The proposed technique tries to overcome the limitations that the Bayesian optimization algorithm has for a large number of hyperparameters. An enhanced sine-cosine algorithm was applied to the benchmark dataset CIFAR-10 and experimental results show that the presented approach shows better results compared with other metaheuristic algorithms (Bacanin *et al.*, 2021). Raju *et al*. (2022) presented and explored different global and local techniques for face recognition. GAs were used for hyperparameter optimization of various CNN models for classification problems on various datasets like MNIST (Yoo *et al*., 2019), Chest Xray (Zhou, 2021), Acute Lymphoblastic Leukemia (Rodrigues *et al*., 2022), etc.

A major issue in the tuning of hyperparameters include training and evaluation of large datasets is that it can be very expensive; moreover, the type and range of hyperparameters are mostly unknown and need some expertise. Therefore, nowadays, automatic hyperparameter tuning is preferred instead of manual methods. Automatic parameter tuning not only reduces manual efforts but also tries to improve the performance efficiency of the learning model. Recently, population based metaheuristic optimization has been used to overcome the issues with manual methods. The characteristics of GAs offer advantages over other optimization algorithms. These characteristics include no need for gradient information, producing good results with multiple local maxima or minima, an ability of being parallelizable, and escaping local optima (Rojas, 1996; Aszemi and Dominic, 2019). Hence, a GA has been chosen for the proposed work.

3. Genetic algorithm

The GA is an evolutionary technique that is commonly used to resolve combinatorial optimization based problems. Each chromosome in the GA population represents a possible solution. A chromosome can be assumed as an amalgamation of genes. For a specific problem, a distinct GA encoding representation is used. The selection step of the GA is based on different methods like elite, roulette, rank, and tournament (Albadr *et al.*, 2020). In the crossover process of the GA, parts of the genetic order are crossed from both the parents to form a novel genetic sequence in the offspring. The goal is to find a chromosome in the population that meets the best fitness requirement.

The first step is to initialize a set of chromosomes that specifies the CNN configuration. In the next steps, GA operators like the selection of best parents, mutation, and crossover have been applied. A population is generated containing a set of chromosomes that comprises CNN configurations of a maximum number of filters and a maximum size of the filters. Based on each individual's fitness function, the whole population is sorted based on the rank from best to worst. The best-ranked parents are selected and crossed. In the mutation step, two bits at two randomly selected genes of an individual's indices are being flipped. The mutation rate is kept low to preserve the good qualities of newly generated chromosomes (Hassanat *et al*., 2019).

4. CNN

CNNs are the most widely used deep networks designed to employ computer vision. A CNN can be defined as a trainable multilayered ANN comprising various layers, where the input and output at each layer are represented as a feature map.

The basic architecture of the CNN consists of several layers such as convolution, pooling, ReLU, normalization, dropout, fully-connected, softmax, etc. each having unique characteristics (Wu *et al*., 2019). Figure 2 represents a typical classification based CNN model.

The convolution layer generates a cluster of feature images. For a 2-D image, $I(i,j)$ as input and a 2-D convolution kernel, K, of size $m \times n$, the convolution operation is characterized by (Aydogdu *et al.*, 2017)

$$C(i,j) = \sum_m \sum_n I(i+m-1, j+n-1)K(m,n). \quad (2)$$

Implementation of this equation for image convolution is illustrated in Fig. 3. A 2-D kernel filter is convolving over the input 2-D image. The kernel is swept over the image at every single location and the output is calculated. Rectified Linear Units (ReLUs) are used for nonlinearities. They are added as they empower the layers to captivate nonlinear data.

Afterwards, pooling is added to minimize the spatial size of the image. Max pooling and average pooling are the most common forms of pooling layers that output the maximum and the average in a sampling area after convolution mapping. Max pooling as an example is illustrated in Fig. 4.

Batch normalization has been the most suitable layer to optimize the network. It is between convolutional and

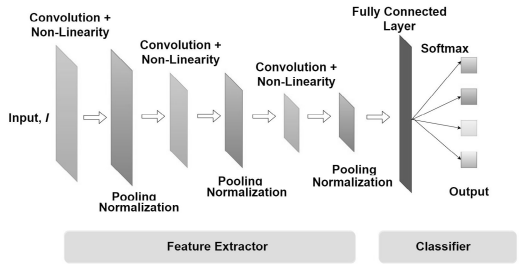


Fig. 2. Typical CNN architecture.

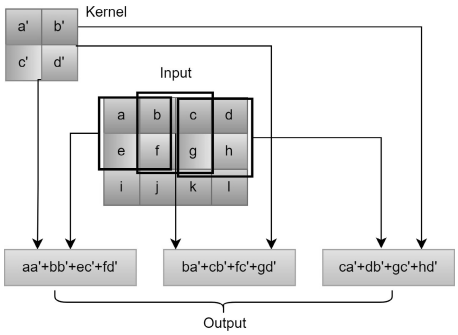


Fig. 3. Image convolution operation.

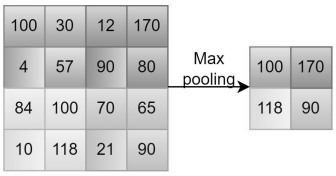


Fig. 4. Diagrammatic illustration of 2 × 2 max-pooling.

Fig. 5. Illustration of the softmax operation.

nonlinear layers to accelerate network training and reduce sensitivity towards network initialization. The normalized activation is given (Ioffe and Szegedy, 2015) as

$$\hat{x}_i = \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}}, \tag{3}$$

where x_i are inputs over mini-batches, μ_β and σ^2 represent the mean and the variance, respectively, ϵ defines numerical stability for the low mini-batch variance. Followed by shifting input to β, a learnable offset and scales through the learnable factor γ, the activation can be defined as

$$y_i = \gamma \hat{x}_i + \beta, \tag{4}$$

where x_i is the normalized value obtained. A dropout layer can also be appended to adjust the layer's input elements to zero for a defined probability. For the node activation probability, a binomial distribution is set to provide training to each node, and based on the value the node is made available or being dropped out. If the value is zero, then the node is restricted; otherwise it is made available. A fully connected layer is conjoined at the end of the network. Every node in this layer is connected with every other node from its preceding layer. It blends all features learned through preceding layers that help in recognizing larger patterns. To resolve image classification, a softmax layer must follow the classification output layer in the CNN. The softmax operation normalizes the outputs of each unit between 0 and 1 so that their output's aggregate to 1 which has been depicted by Fig. 5. The mathematical representation of the softmax activation function is as follows:

$$\sigma(z_r) = \frac{e^{z_r}}{\sum_{j=i}^{K} e^{z_j}}, \tag{5}$$

where $0 \leq z \leq 1$ and z represents the input vector. k is the number of classes. The softmax layer's output is then taken by the classification layer followed by an assignment of a given input by the classification layer to one of K mutually exclusive classes.

5. Proposed model

A highly efficient approach for the image classification based CNN-GA model has been proposed. This work exploits the utility of the GA for the model hyperparameter optimization of the CNN. Several choices for designing the model architecture are available; therefore, it is a tedious task to find the best model architecture. The model parameters that have been used for analysis include the size of the kernel also known as filter size as well as the number of filters and of hidden layers. The hidden layers define the depth of a given CNN structure. A low-depth network is incapable of absorbing intricate learning tasks for image based processing, whereas a very deep network structure can overfit effortlessly. In consequence, the number of convolution layers is one of the main parameters to be decided for proper results. The second parameter, i.e., kernel size, depends on the problem and an optimal kernel size helps to learn the designs. The third parameter considered for optimization is the number of filters. Its

optimal value is needed to avoid a loss of information or the network overfitting (Bibaeva, 2018). The network architecture for hyperparameter optimization using the CNN-GA framework is illustrated in Fig. 6.

The proposed model has two subblocks. In the first block, the GA was used for hyperparameter optimization followed up with a second block that prepares the training of the CNN using the obtained optimized hyperparameters.

In the GA block, a random initial population of chromosomes is generated. For three convolutional layers, each chromosome is comprised of six genes that correspond to one solution illustrated in Fig. 7. The first three genes of the chromosome denote the number of filters and the next three genes represent the corresponding size of each filter. After initialization, the fitness of each chromosome is calculated, which depends on the error generated by the CNN. Two parents with the best solution or fewer errors are selected and crossover is performed. In the crossover operation, random indices for crossover points are generated and then using these crossover points selected parents are crossed to generate offspring as shown in Fig. 8. In mutation, random indices of a gene in chromosomes are generated and the values in these indices are swapped. Figure 9 represents the mutation process of the GA. The mutation is carried out to avoid the algorithm to stuck in local minima. The above procedure is carried out iteratively to get the best combination of genes in a chromosome. Each individual is evaluated by its accuracy or error generated by training the CNN network. Similarly, the experiment is repeated for different numbers of convolutional layers and the numbers of genes in chromosomes are set accordingly .

The CNN block takes CNN hyperparameters, i.e., the number of convolution layers, the maximum number of filters, and the corresponding kernel size from the GA block. The other parameters like the learning rate or the number of epochs are also initialized in this block. The dataset was split into training and testing sets and subsequent layers of the CNN like the ReLU function, max-pooling, or the softmax layer are applied. The error generated by the CNN is used as an objective function for the GA. The objective is to solicit multinomial logistic regression through the proposed architecture that maximizes the log-probability of the true label beneath its prediction distribution (Mahajan *et al.*, 2020b). The steps involved in generating an optimized set of CNN hyperparameters for the proposed work are presented in Algorithm 1.

5.1. Objective function. An objective function was designed to select the best combination of hyperparameters. The classification error generated by CNN was used as an objective function for the GA. The objective is to find the best combination of the hyperparameters using the GA and then this combination is applied for face detection,

$$Fitness\,Value = 100 - accuracy\%. \tag{6}$$

5.2. Parameter setting for the GA. The initial population consists of chromosomes with each chromosome having a length of 6. First, three genes of chromosomes represent 'Number of filters' and the next three represents 'Size of filter' for the CNN. An initial population of 20 individuals was been created and the number of generations considered is 50. Parameters like the mutation rate or the crossover probability are set as 0.01 and 0.50, respectively, as shown in Table 2.

Algorithm 1. GA-CNN algorithm.

Require: *max_iter*: maximum number of GA iterations or generations
max_pop: maximum number of solutions
individual_fit: fitness value of each individual in terms of validation error
cross_prob: crossover probability rate
mut_prob: random mutation probability
children_list: population generated after crossover operation
conv_layers: number of convolutional layers
Input: (*max_iter*, *cross_prob*, *mut_prob*,*max_pop*)
Output: (Optimized hyperparameters, i.e., *filter size*, *number of filters*, *number of convolutional layers)*
for *conv_layer* = 2 to 4 **do**
  initialize *iter* = 0
  initialize population with random hyperparameter combination
  for each individual in population
    calculate *individual_fit* = validation error
  end for
  while *iter* ≤ *max_iter* **do**
    for individuals in population **do**
      select best parents from population based on fitness function value
      according to select $_{prob}$
      apply mutation to *children_list* with defined mutation probability
      apply crossover to generated *children_list*
      po = *children_list*
      calculate *individual_fit* for new population = validation error
    end for
    iter = *iter* + 1
  end while
end for

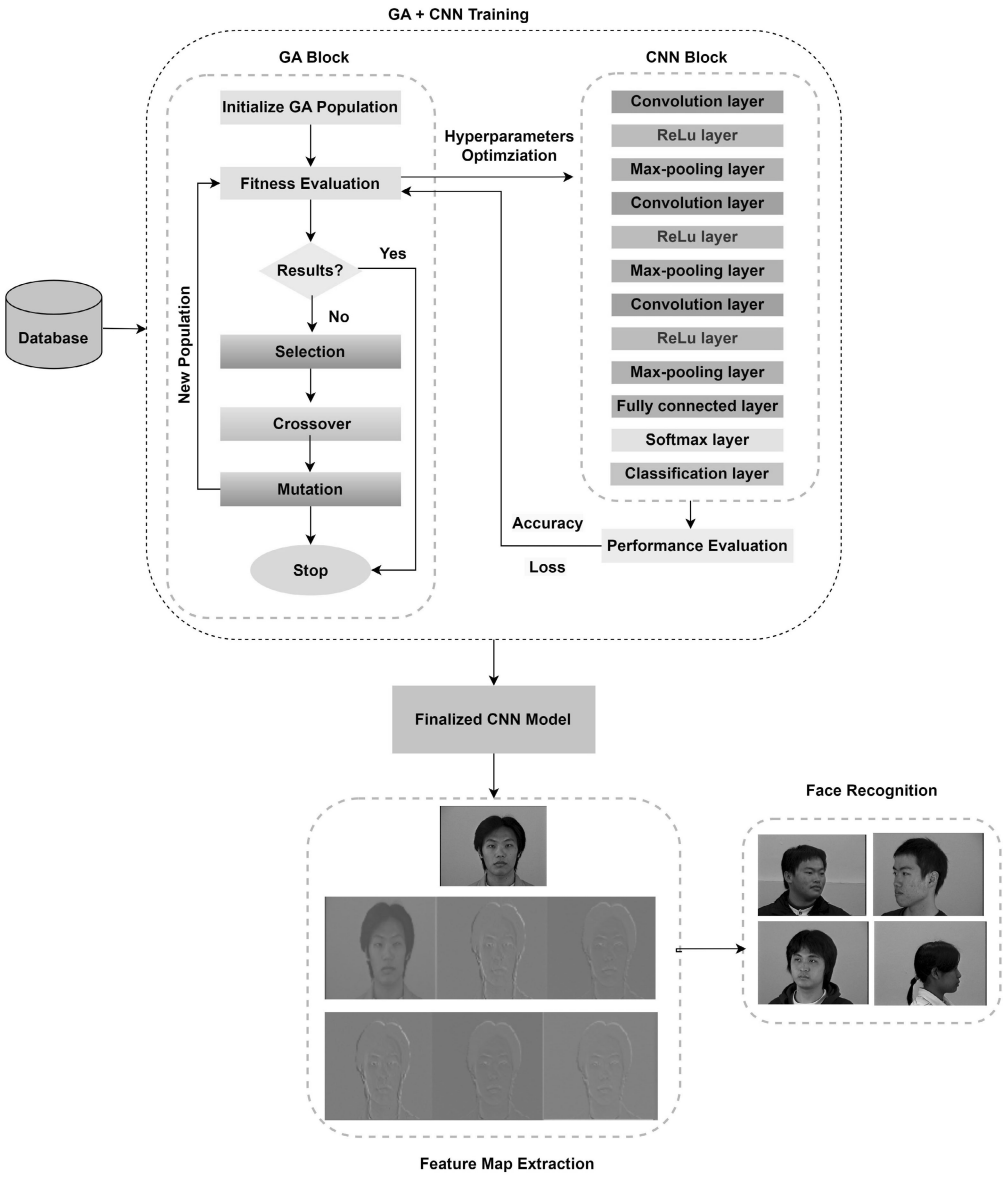


Fig. 6. Face recognition architecture using a CNN with GA based optimization.

6. Experimental results

This section exemplifies the capability of the proposed model CNN-GA model to find the optimal structures for face recognition. The task is to tune the values of filter sizes and their number. The experimental work is performed on Intel(R) Core (TM) i7-8750H CPU with 20 GB RAM accompanied by NVIDIA GeForce GTX 1050Ti. The software platform used is MATLAB R2019b operating on Windows 10.

6.1. Data description. For determining the viability of the proposed model, face recognition based classification is performed on the Robotics Lab's Face detection dataset[1]. The Face dataset consists of 6660 images with 90 subjects. Each subject has 74 images that are taken from different angles, i.e., every 5 degrees in the pan rotation from the right to the left profile. Every subject represents a class category with 74 images each of size 640×480. Figure 10 illustrates several ground-truth images for the dataset representing various face angles for different subjects.

6.2. Observation. First, we tried to manually design the CNN structure by using the hit and trial method. This approach gives a very poor classification accuracy

[1] http://robotics.csie.ncku.edu.tw/Databases/FaceDetect_PoseEstimate.htm#Our_Database_.

of about 60%. To experiment with the proposed methodology, the number of convolution layers, the maximum number of filters, and the maximum kernel size considered are shown in Table 3.

The CNN model is trained for different configurations of convolutional layers. Figure 11 depicts varied transitions for the accuracy-loss profile while training through various iterations. In certain graphs, a low accuracy can be observed for the initial iterations of the GA, and the accuracy rises as iterations increase. The GA with 10 chromosomes and 10 generations was executed 10 times. Figure 12(a) shows the convergence curve for the objective function using 2 convolutional layers of the CNN. Since the objective function of the GA depends on the error generated by the CNN, it can be observed that with the increase in iterations the fitness function starts decreasing but does not converge till 10 iterations, and the accuracy achieved is 85%.

The experiment is repeated by using three convolutional layers. The convergence curve for 3 convolutional layers of CNN is shown in Fig. 12(b). The accuracy achieved using three convolutional layers is about 90%. Figure 12(c) represents the convergence of the CNN's classification error using four convolutional layers and the accuracy achieved is 94.5%. Thus, the best combination of hyperparameters is generated by a four-convolutional-layer CNN with the highest accuracy. In expansion to the standard 80–20 training–testing split of the dataset, furthermore, we included both 70–30 and 60–40 training–testing data splits. A comprehensive breakdown of the classification parameter's performance for configured GA-CNN models for different convolutional layers can be found in Table 4. As can be observed in Table 4, the GA-CNN model with four convolutional layers performed preeminently within the cluster with the ratio of the 70–30 split giving the overall best outcome in terms of precision, recall, and the F1-score. The results for the split can be found in Figs. 13 and 14, respectively.

Also, a comparison was made with the existing CNN techniques like AlexNet, VGG-16, ResNet50, and InceptionV3 models as shown in Table 5. In the case of existing CNNs, the observed performance in terms of accuracy was less than 90%. For the proposed GA-CNN model the accuracy observed is equal to 94.5% for the four-layer model. As hyperparameter tuning was incorporated using GA parameters, the proposed method yields high accuracy compared with non-optimization based techniques.

Table 2. GA parameter initialization.

	Parameter	Value
1	population size	10
2	no. of generations	10
3	mutation rate	0.02
4	crossover	0.50
5	mutation method	swap mutation
6	selection method	rank based
7	fitness function	CNN error

Table 3. CNN hyper-parameters.

	Parameter	Value
1	number of convolutional layers	2, 3, 4
2	maximum Kernel size	20
3	maximum number of filters	50

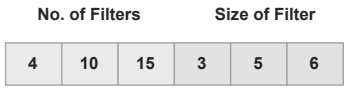


Fig. 7. Chromosome encoding.

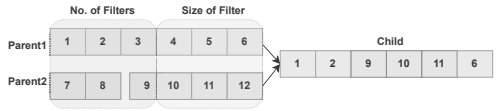


Fig. 8. Crossover.

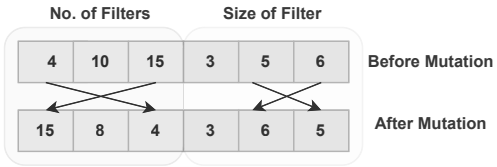


Fig. 9. Mutation.

Fig. 10. Snippet of the face detection database.

7. Conclusion

The paper implements a GA for the hyperparameter optimization of a CNN model for face recognition. Since the complexity of finding an optimal set of parameters increases with an increase in the number of hyperparameters, metaheuristic based optimization techniques are used to find the best combination of hyperparameters. The work presents a thorough experimental analysis of a GA based CNN network for different convolutional layers. The GA tries to

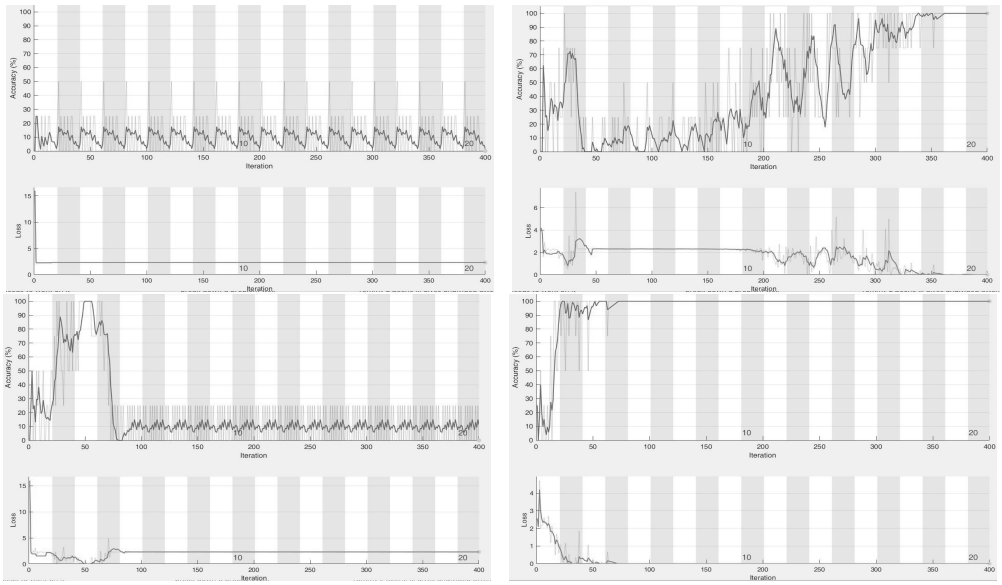
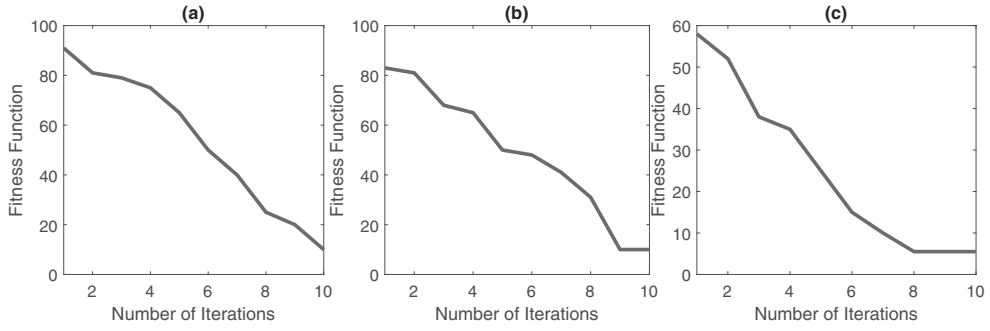


Fig. 11. Transitions in the accuracy-loss curve while training the CNN through various iterations with the GA.



(a) (b) (c)

Fig. 12. Fitness value for each iteration using two convolutional layers (a), three convolutional layers (b) and four convolutional layers of the CNN (c).

Table 4. Performance measured using different training–testing splits.

	80–20			70–30			60–40		
GA-CNN model	Precision	Recall	F1-score	Precision	Recall	F1-score	Precision	Recall	F1-score
2 convolutional layers	78.38	79.28	78.51	75.32	74.27	79.5	70.51	77.75	78.71
3 convolutional layers	79.54	79.53	79.72	78.89	79.14	78.23	77.23	77.30	77.88
4 convolutional layers	82.71	80.35	81.27	89.49	89.75	85.19	81.17	79.90	80.56

find the best combination set of CNN hyperparameters for face recognition. The proposed model generates optimum results using a four-convolutional-layer based CNN model. Thus, the proposed work presents a reliable optimized CNN based FR system. Further, the inclusion of more hyperparameters like the learning rate, the number of epochs, etc., and applying them to other datasets may be considered. Also, the incorporation of the other metaheuristic algorithms for further optimization may be considered for future work.

References

Albadr, M.A., Tiun, S., Ayob, M. and Al-Dhief, F. (2020). Genetic algorithm based on natural selection theory for optimization problems, *Symmetry* **12**(11): 1758.

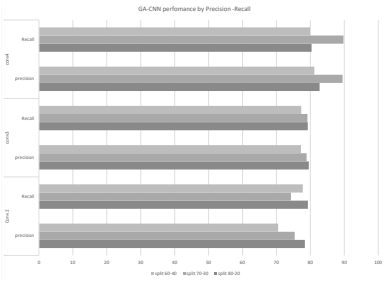


Fig. 13. GA-CNN performance in terms of precision and recall.

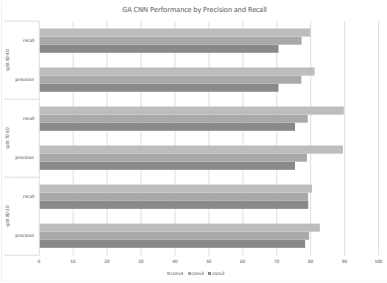


Fig. 14. GA-CNN performance in terms of F1-score.

Table 5. Performance comparison with different CNN models.

	Method	Accuracy
1	VGG-16 (Sikha and Bharath, 2022)	85.002
2	Inception V3	87.11
3	ResNet 50	86.066
4	AlexNet	78
5	CNN without optimization	60
6	Proposed GA-CNN (2 layers)	85
7	Proposed GA-CNN (3 layers)	90
8	Proposed (4 layers) GA-CNN	94.5

Aszemi, N.M. and Dominic, D.D. (2019). Hyperparameter optimization in convolutional neural network using genetic algorithms, *International Journal of Advanced Computer Science and Applications* **10**(6): 269–278.

Aydogdu, M.F., Celik, V. and Demirci, M.F. (2017). Comparison of three different CNN architectures for age classification, *11th IEEE International Conference on Semantic Computing (ICSC), San Diego, USA*, pp. 372–377.

Bacanin, N., Zivkovic, M., Salb, M., Strumberger, I. and Chhabra, A. (2021). Convolutional neural networks hyperparameters optimization using sine cosine algorithm, *in* S. Shakya *et al.* (Eds), *Sentimental Analysis and Deep Learning: Proceedings of ICSADL 2021*, Springer, Singapore, pp. 863–878.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization, *Journal of Machine Learning Research* **13**(2): 281–305.

Betró, B. (1991). Bayesian methods in global optimization, *Journal of Global Optimization* **1**: 1–14.

Bibaeva, V. (2018). Using metaheuristics for hyper-parameter optimization of convolutional neural networks, *IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP), Aalborg, Denmark*, pp. 1–6.

Chhabra, Y., Varshney, S. and Ankita (2017). Hybrid particle swarm training for convolution neural network, *International Conference on Contemporary Computing, Noida, India*, pp. 1–3.

Cui, H., Bai, J. (2019). A new hyperparameters optimization method for convolutional neural networks, *Pattern Recognition Letters* **125**: 828–834.

Dargan, S., Kumar, M., Ayyagari, M.R. and Kumar, G. (2020). A survey of deep learning and its applications: A new paradigm to machine learning, *Archives of Computational Methods in Engineering* **27**: 1071–1092.

Gadekallu, T.R., Alazab, M., Kaluri, R., Maddikunta, P.K.R., Bhattacharya, S., Lakshmanna, K. and Parimala, M. (2021). Hand gesture classification using a novel CNN-crow search algorithm, *Complex & Intelligent Systems* **7**: 1855–1868.

Guo, G. and Zhang, N. (2019). A survey on deep learning based face recognition, *Computer Vision and Image Understanding* **189**: 102805.

Hassanat, A., Almohammadi, K., Alkafaween, E., Abunawas, E., Hammouri, A. and Prasath, V.S. (2019). Choosing mutation and crossover ratios for genetic algorithms—A review with a new dynamic approach, *Information* **10**(12): 390.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International Conference on Machine Learning, Lille, France*, pp. 448–456.

Karlupia, N., Sambyal, P., Abrol, P. and Lehana, P. (2019). BFO and GA based optimization of illumination switching patterns in large establishments, *6th International Conference on Computing for Sustainable Global Development (INDIACom), New Delhi, India*, pp. 349–354.

Li, Y. and Abdallah., S. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice, *Neurocomputing* **415**: 295–316.

Liu, J. and An, F.-P. (2020). Image classification algorithm based on deep learning-kernel function, *Scientific Programming* **3**: 1–14.

Mahajan, P., Abrol, P. and Lehana, P.K. (2020a). Effect of blurring on identification of aerial images using convolution neural networks, *in* P.K. Singh *et al.* (Eds.), *proceedings of ICRIC 2019: Recent Innovations in Computing*, Springer, Cham, pp. 469–484.

Mahajan, P., Abrol, P., Lehana, P.K. (2020b). Scene based classification of aerial images using convolution neural networks, *Journal of Scientific and Industrial Research* **79**(12): 1087–1094.

Mahajan, P., Jakhetiya, V., Abrol, P., Lehana, P., Subudhi, B.N. and Guntuku, S.C. (2021). Perceptual quality evaluation of hazy natural images, *IEEE Transactions on Industrial Informatics* **17**(12): 8046–8056.

Mahajan, P., Karlupia, N., Abrol, P. and Lehana, P.K. (2021). Identifying COVID-19 pneumonia using chest radiography using deep convolutional neural networks, *62nd International Scientific Conference on Information Technology and Management Science of Riga Technical University (ITMS), Riga, Latvia*, pp. 1–6.

Mohakud, R. and Dash, R. (2021). Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection, *Journal of King Saud University: Computer and Information Sciences* **34**(8): 1319–1578.

Patro, K.K., Prakash, A.J., Samantray, S., Pławiak, J., Tadeusiewicz, R. and Pławiak, P. (2022). A hybrid approach of a deep learning technique for real-time ECG beat detection, *International Journal of Applied Mathematics and Computer Science* **32**(3): 455–465, DOI: 10.34768/amcs-2022-0033.

Pujol, F.A., Mora, H. and Girona-Selva, J.A. (2016). A connectionist computational method for face recognition, *International Journal of Applied Mathematics and Computer Science* **26**(2): 451–465, DOI: 10.1515/amcs-2016-0032.

Raju, K., Chinna Rao, B., Saikumar, K. and Lakshman Pratap, N. (2022). An optimal hybrid solution to local and global facial recognition through machine learning, *in* P. Kumar *et al.* (Eds), *A Fusion of Artificial Intelligence and Internet of Things for Emerging Cyber Systems*, Springer, Cham, pp. 203–226.

Rodrigues, L.F., Backes, A.R., Travençolo, B.A.N. and de Oliveira, G.M.B. (2022). Optimizing a deep residual neural network with genetic algorithm for acute lymphoblastic leukemia classification, *Journal of Digital Imaging* **35**(3): 623–637.

Rojas, R. (1996). *Neural Networks: A Systematic Introduction*, Springer, Berlin/Heidelberg.

Sikha, O. and Bharath, B. (2022). VGG16-random Fourier hybrid model for masked face recognition, *Soft Computing* **26**(22): 12795–12810.

Suganuma, M., Shirakawa, S. and Nagao, T. (2018). A genetic programming approach to designing convolutional neural network architectures, *International Joint Conference on Artificial Intelligence, Stockholm, Sweden*, pp. 5369–5373.

Syulistyo, A.R., Purnomo, D.M.J., Rachmadi, M.F. and Wibowo, A. (2016). Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN), *Complex & Intellelligent Systems* **9**(1): 52–58.

Victoria, H. and Maragatham, G. (2021). Automatic tuning of hyperparameters using Bayesian optimization, *Evolving Systems* **12**: 217–223.

Vose, A., Balma, J., Heye, A., Rigazzi, A., Siegel, C., Moise, D., Robbins, B. and Sukumar, S. R. (2019). Recombination of artificial neural networks, *ArXiv:* abs/1901.03900.

Wu, J., Chen, X.-Y., Zhang, H., Xiong, L.-D., Lei, H. and Deng, S.-H. (2019). Hyperparameter optimization for machine learning models based on bayesian optimization, *Journal of Electronic Science and Technology* **17**(1): 26–40.

Yoo, J.-H., Yoon, H.-i., Kim, H.-G., Yoon, H.-S. and Han, S.-S. (2019). Optimization of hyper-parameter for CNN model using genetic algorithm, *2019 1st International Conference on Electrical, Control and Instrumentation Engineering (ICECIE), Kuala Lumpur, Malaysia*, pp. 1–6.

Wang, Y., Zhang, H. and Zhang, G. (2019). CPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks, *Swarm and Evolutionary Computation* **49**: 114–123.

Zhou, M. (2021). Heuristic hyperparameter optimization for convolutional neural networks using genetic algorithm, *arXiv:* 2112.07087.

Namrata Karlupia is currently a senior researcher in the Department of Computer Science and IT, University of Jammu. She holds an MTech in computer science from the University of Jammu. She works in the area of nature-inspired algorithms, machine learning, and deep learning.

Palak Mahajan holds an MTech degree in engineering from Shri Mata Vaishno Devi University, Jammu. She is currently pursuing her PhD and works as a lecturer in the Department of Computer Science and Information Technology, University of Jammu. She is interested in image processing, deep learning, computer vision, and machine learning.

Pawanesh Abrol is a professor in the Department of Computer Science and Information Technology, University of Jammu. He holds a PhD in computer science from the University of Jammu, as well as an MBA (HRM). Dr. Abrol has contributed more than 50 research publications in various reputed national and international proceedings and journals. His research covers aura based texture analysis, image forgery as well as authentication and eye gaze technologies.

Parveen Kumar Lehana holds an MS degree from Kurukshetra University and a PhD degree in signal processing from the Indian Institute of Technology in Bombay. He is currently a professor in the Department of Electronics, University of Jammu. He has wide experience in teaching, research, and guiding MS, MEng, and PhD students. He has published more than 200 research papers in national/international journals.

Received: 15 May 2022
Revised: 19 December 2022
Accepted: 20 December 2022