Journal of King Saud University – Computer and Information Sciences xxx (xxxx) xxx

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection

Rasmiranjan Mohakud, Rajashree Dash *

Computer Science & Engineering Department, Siksha O Anusandhan (Deemed to be University), Bhubaneswar, India

A R T I C L E I N F O

Article history:
Received 13 January 2021
Revised 1 May 2021
Accepted 27 May 2021
Available online xxxx

Keywords:
CNN
GWO
Hyper parameter optimization
Cancer detection

A B S T R A C T

In recent history, Convolutional Neural Networks have attained breakthroughs in addressing many intractable problems in the domain of image processing. But its performance builds upon its chosen hyper parameters and it is a tedious job to manually fine tune these hyper parameters. Hence, in this research, an Automated Hyper-parameter Optimized Convolution Neural Net is proposed which is further applied to uncover the class of skin cancer. The method has utilized a Grey Wolf Optimization algorithm for optimizing the hyper parameters of CNN, by adopting a proper encoding scheme. The effectiveness of the model is verified by comparing it with the performance of Particle Swarm Optimization and Genetic algorithm based hyper-parameter optimized CNN applied on the International Skin Imaging Collaboration skin lesion multi class data set. Simulation results infer that the proposed model is able to produce a testing accuracy up to 98.33% which is around 4% and 1% more compared to PSO and GA based models respectively. Similarly with the proposed model, the testing loss realized is around 0.17% which is 39.2% and 15% less compared to PSO and GA based models respectively. The experimental results clearly demonstrate that the proposed model performs competitively compared to other reported models.

1. Introduction

The major area of the human body is covered by the skin, which protects the internal organs from outer hazardous elements such as dust, heat, contaminated water, ultra violet rays and so on. But the direct contact of skin to these speculative factors has also a devastating impact on its well-being. It may be the cause of several skin diseases such as Moles, Rosacea, Eczema, and Cancer, that can affect people of any age group. Among these diseases, skin cancer is appearing as a common threat day to day. Around 300,000 new cases have been found in the year 2018 out of which 2490 females and 4740 males have lost their lives in 2019 (Siegel et al., 2019). Skin cancer is categorized into 8 types such as, Melanoma, Melanocytic Nevus, Basal Cell Carcinoma, Actinic Keratosis, Benign Keratosis, Dermatofibroma, Vascular Lesion, and Squamous Cell Carcinoma. Among all skin cancers, Melanoma is the most dangerous category which causes serious harm to the skin and spreads to other parts of the human body. Studies have shown that early detection of skin cancer reduces the death rate drastically (Razmjooy et al., 2012). Detecting the early stage of skin cancer is a tedious job for the dermatologist, which motivates us to model a simplified and automatic skin cancer detector to detect skin cancer from early-stage, which in turn can help the dermatologists a lot.

In recent decades, deep learning plays an important role in computer vision and image processing like object detection, image segmentation, lesion detection from dermatology image (Kharazmi et al., 2017), skin lesions characterization (Maglogiannis and Doukas, 2009). Convolutional neural network (CNN) has a profound effect in the field of image processing due to its automatic feature detection, higher prediction, and classification accuracy. High precision of CNN motivates researchers to apply it for solving several image processing problems such as tumor detection, skin lesion classification, Breast cancer detection and so on. The multi-layer architecture of CNN consists of Convolution, Relu, Dropout, Pooling and Dense layers. Each layer subsequently finds

* Corresponding author at: Department of Computer Science & Engineering, ITER, Siksha O Anusandhan (Deemed to be University), Bhubaneswar, 751030, India.

E-mail addresses: rashmiranjanmahakud@soa.ac.in (R. Mohakud), rajashreedash@soa.ac.in (R. Dash).

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.jksuci.2021.05.012
1319-1578/

Please cite this article as: R. Mohakud and R. Dash, 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, https://doi.org/10.1016/j.jksuci.2021.05.012

the complex feature from the simple feature. The depth of CNN plays a significant role in prediction and classification accuracy (Simonyan and Zisserman, 2014). An increase in the depth of CNN increases the number of hyper-parameters. CNN model has numbers of hyper-parameters such as numbers of layers, kernel size in each layer, numbers of kernels in each layer, kernel size associated with max-pooling, dropout rate used in dropout, size of the dense layer, learning rate, number of convolution, number of max-pooling, number of dense layers, number of epoch, batch size, optimizer, activation function and so on. Usually selection of these hyper-parameters is handled manually in a costly trial and error basis, which in turn results in long training time of CNN due to the evaluation of different hyper-parameters configuration in different rounds. In the interim, new CNNs tend to have more and more layers with an expeditious rise in the number of hyper-parameters. Hence it is almost futile to pinpoint the close to optimal hyper-parameter configuration for a CNN manually under a reasonable cost. On the other hand, proper hyper-parameter tuning can increase the overall performance of the CNN model. Hence this ambition has compelled many researchers to tackle the tuning of hyper-parameters of CNN as an optimization problem. Particle swarm optimization (Wang et al., 2019), univariate dynamic encoding (Yoo, 2019), and multilevel evolutionary optimization (Cui and Bai, 2019) are some of the optimization techniques, which are successfully applied for optimization of hyper-parameters of CNN.

Although these articles give a promising result on hyper-parameter optimization, still there is a room for further improvement. In the recent past, Grey Wolf Optimization (GWO) technique has appeared as a promising *meta*-heuristic technique solving different standard optimization problems, by mimicking the social hierarchy and hunting capability of grey wolves (Mirjalili et al., 2014). Though it has been utilized for addressing the feature extraction problem (Garg et al., 2019; Maddikunta et al., 2020), weight initialization of CNN (Kumaran et al., 2018), training of CNN (Chen et al., 2019; Xie et al., 2020; Agarwal and Sharma, 2020), but it has not applied for hyper-parameter optimization problem. Hence, in this research, for the first time it is applied for tuning the hyper-parameters of CNN in application to skin cancer classification. Then the optimized hyper-parameters are used in creating an efficient CNN model which is further trained using the back propagation algorithm for handling the skin cancer multi-class classification problem.

Before addressing the classification task, initially the collected skin cancer images are preprocessed through the steps of image resizing, color conversion and filtering. The suitable image preprocessing technique is selected by testing different image preprocessing techniques with a lightweight CNN over a small sample dataset. The sample dataset is generated by applying random sampling technique to the original dataset whereas the light weight CNN model is constructed with three convolutions, three relu, three droput, three max-pooling, one flatten layer and two dense layers. Image resizing performs reshaping of the random size data to a similar shape for the model. Reducing the images to genuine size can save the training and testing time of the CNN model. By performing simulation with five different popular image resizing techniques such as Nearest neighbor interpolation (NNI), Bilinear interpolation (BI), Resampling using pixel area relation (RPAR), Bicubic interpolation over 4×4 pixels neighborhood (BIPN), Lanczos interpolation over 8×8 neighborhood (LIN) over a small sample image dataset the BI technique is found to be the superior one and hence is further applied in the proposed model. Again observing the skin cancer data, it is figured out that color plays a less important role in the classification. So after resizing the images, they are converted from color to grayscale so as to further save the training time. After color conversion, the image filtering approach is applied to remove the unwanted features from the images such as hair, noise, small unaffected area and so on, which may enhance the further classification performance. By performing experiments with four different types of image filtering techniques such as Averaging, Gaussian, Median, and Bilateral over the sample dataset, Gaussian filtering is found to provide better accuracy and hence it is further used in the proposed model.

After preprocessing, an efficient encoding technique has been used for hyper-parameters to fit it into GWO. In literature the optimization goal is different depending on the definitions for hyper parameters. Some has adopted hyper-parameters in a narrow sense including only the parameters of each layer without changing the overall architecture and others have taken into it in a broader sense by considering the number of layers, learning rate along with the hyper-parameters of each layer. Considering the former one, after determining the architecture of the CNN, the main goal of this paper is to reach at a near to optimal values of hyper-parameters involved mainly in convolution, dropout and pooling layer. The most time consuming part of hyper-parameters tuning is the training of CNN which is defined as the fitness function to be evaluated in the GWO optimization process. Therefore, to improve the hyper-parameters tuning efficiency, a suitable fitness function is designed that will be able to calculate the fitness value in form of CNN accuracy in limited time. Classifying skin cancer, from early stage using dermoscopic images is even difficult for experts, so efficient automation is required for this. Hence, in this research, we have used CNN for its end to end training capacity, automatic feature selection, and classification accuracy. A Cost effective CNN model is created by using the optimized hyper-parameters resulting through GWO. Finally the CNN model is trained using the International Skin Imaging Collaboration (ISIC) dermoscopic skin cancer image dataset. The experimental results clearly demonstrate the superiority of the proposed method in comparison to Particle Swarm Optimization (PSO) and Genetic algorithm (GA) based hyper parameter optimization approach of CNN in terms of both accuracy and loss of the optimized CNN.

The Key contribution of this research is typified as follows:

- Proper image preprocessing is incorporated to save the overall training time of CNN.
- Hyper-parameters of the CNN are optimized using GWO to model a cost effective CNN classifier.
- The proposed model is also compared with PSO and GA based hyper-parameter optimization approach.
- Extensive experiments are conducted on ISIC dermoscopic multi class skin cancer image dataset to verify the effectiveness of the proposed model.

The rest of the paper is organized as follows. Section 2 describes a survey of related works. The detailed proposed model is discussed in section 3. Section 4 is about the experimental outcomes and its analysis. Finally Conclusion and future scope are briefed in section 5.

2. Literature survey

Object detection and image classification are the two emerging fields for decades; again neural networks especially CNN appeared to be more favored in this domain. The fully connected hidden layers of neural networks result in high cost and larger time complexity in addressing the object detection problems. As a solution to this problem, CNN is popularly used for image classification nowadays. CNN consists of different layers such as, convolution, max pool, relue, dropout, dense and softmax. There exist a number of hyper-parameters related to the each layer of CNN model such as kernel size, the number of kernels, stried in the kernel, activation function involved in convolution layer, kernel size associated with max-pooling, dropout rate used in dropout, size of the dense layer, learning rate, number of convolution, number of max-pooling, number of dense layers and so on. The outcome of the CNN model highly relies on these hyper-parameters. To increase the accuracy of the CNN model some researchers have suggested an increase in the number of layers in CNN. In the case of CNN applied for automatic feature detection, simple features are detected by initial layers, followed by complex features abstracted in the later layers. For abstracting complex features the depth of the model plays an important role. Feature selection in each layer depends on the number of kernels and kernel size. To get better features, researchers use the concept of using fewer kernels in the initial layer and large numbers of features in the latter layer. Learning rate plays an important role at the time of training the CNN model using any training algorithm like gradient descent, Adam optimizer, gradient descent with momentum and so on. If the learning rate is too small the gradient converges to the optimal solution slowly. If it is large, gradient accelerates near solution. Therefore selection of proper learning rate is another important factor. Regularization of CNN is done by the dropout rate. Proper dropout rate generalizes the model.

In literature, some promising CNN models such as GoogleNet (Szegedy et al., 2015), VGG (Badrinarayanan et al., 2015), SegNet (Ma et al., 2020) and ResNet (Li et al., 2019) are available in the field of object detection and image classification. The GoogLeNet proposed in (Szegedy et al., 2015) have a total of 22 layers with a 5 X 5 kernel size of pooling layer, rectified linear unit used in convolution layer, a fully connected layer of 1024 and a dropout rate of 70%. The VGG net proposed in (Badrinarayanan et al., 2015) has used different depth networks ranging from 11 layers (8 convolutions and 3 fully connected) to 19 (16 convolutions and 3 fully connected). In that network the width of the convolution layer i.e. number of kernels is varied from 64 to 512, kernel size is 3X3, dropout rate is 50%, learning rate is initialized between 10^{-1} to 10^{-2} and momentum of gradient descent is set to 0.9 . Similarly the SegNet applied in (Ma et al., 2020) is a 13 layer encoder and 13 layer decoder architecture, that has used kernel number from 64 to 512, kernel size 3X3, learning rate 0.1 and momentum 0.9. Looking at these networks, it is clear that they have very complex structure with lots of hyper-parameters. Tuning such a large number of hyper-parameters manually is a tedious job. Again the layers of CNN are gradually expanding to consummate the huge complex data.

Different approaches including nature based algorithms are proposed by different researchers to optimize them. Hyper-parameters annotation varies from researchers to researchers that lead to different optimization goals. Some consider hyper-parameters in a confined sense i.e. they consider hyper-parameters of the existing CNN. Whereas some consider hyper-parameter in an extensive sense i.e. they also include the number of layers, learning rate, dropout rate and so on. Hyper-parameter optimization using nature-inspired algorithms needs proper encoding of hyper-parameters, initializing population, defining an objective function to calculate the fitness of the solution, searching for the best solution and updating the worst solution according to the best solution. A survey on nature inspired algorithms used for hyper parameter optimization of CNN is presented in (Mohakud and Dash, 2019). WANG et al. has utilized a canonical PSO (cPSO) to optimize the hyper-parameters of four types of CNN models such as VGG, segNet, resNet and GoogLeNet (Wang et al., 2019). Instead of using the full CNN model, the authors have used a light weight model in fitness calculation. Again they have also adjusted the acceleration coefficient of cPSO in accordance with the range of hyper-parameters. A Genetic algorithm (GA) is recommended in (Ma et al., 2020) for optimizing the hyper-parameters of a DeepCNN. After encoding the DCNN parameters to GA, the initial population is created, and then crossover, mutation, and selection process are repeated to strengthen the fitness of the individuals. In (Cui and Bai, 2019) a multi scale and multilevel evolutionary GA is proposed for optimization of CNN hyper-parameters. The proposed approach is derived by combining the features of Multilevel Evolutionary Optimization (Akbari and Ziarati, 2011) with the Gaussian Process-based Bayesian Optimization (GPEI) (Snoek et al., 2012) approach. Here the authors have used a CNN with three convolutions and two fully connected layers. The roulette wheel selection process is applied to select two best solutions after random initialization of the population. Then the optimized parameters are derived through repeated application of crossover, mutation and selection process. A Quantum Behaved PSO (BQPSO) is proposed in (Li et al., 2019) to evolve the architecture of the CNN. A fixed length binary string is used here to represent the encoded CNN architecture parameters. In (Lopez-Rincon et al., 2018) a GA is used for optimizing the hyper-parameters of CNN, which is utilized for classification of microRNA in cancer. In (Albelwi and Mahmood, 2017) a Nelder-Mead Method is applied to optimize the hyper-parameters of a CNN by introducing a new objective function. The error rate and the learnt information from the mapped feature set are used unitedly in the objective function that results in developing a better architecture by increasing the convergence speed.

Although these articles give a promising result on hyper-parameter optimization, still there is room for further improvement. Being motivated by the successful application of Grey Wolf Optimization (GWO) technique in other application domains, we devote our research to optimize the hyper-parameters of the CNN model using this. A hybrid GWO-CNN model is proposed in (Garg et al., 2019) for efficient network anomaly detection in cloud setups, where CNN is utilized for classification of anomaly and GWO is applied for addressing the multi objective feature extraction problem. A two stage feature extraction technique is proposed in (Maddikunta et al., 2020) by combining PCA with GWO for enhancing the performance of a DNN based intrusion detection system to foretell the class label of unforeseen cyber-attacks. In (Kumaran et al., 2018) GWO is proposed to generate the initial weights of CNN which is further used in training of the network through gradient descent algorithm. The proposed hybrid CNN-GWO approach is applied in recognizing the human actions from the unconstrained videos. The authors have also observed enhanced performance of classification by fusing the evidences of the classifiers produced by the GWO. In (Chen et al., 2019) GWO is applied to optimize the weights of CNN in application to flight state identification. The high identification accuracy is clearly observed by considering two case studies through the application of the proposed model. The network topology and learning parameters of a CNN-LSTM network is also optimized in (Xie et al., 2020) by an enhanced GWO algorithm in application to time series analysis. In (Agarwal and Sharma, 2020) a recurrent ELM is proposed for breast cancer classification, where GWO is utilized for optimizing the weights of the network and CNN is applied for mass detection in the initial stage. Application of GWO in hyper- parameter optimization problems are rare in literature. Hence, in this research, it is applied for tuning the hyper-parameters of CNN and the network created using those optimized hyper-parameters is utilized to address the skin cancer multi-class classification problem.

3. Related methodologies

This section presents an explanation about the CNN along with its hyper-parameters and GWO based optimization approach.

3.1. Convolutional neural network

Automatic feature selection and end to end training are some of the exciting features of the CNN due to which it is popularly used in image classification. Other than convolution layer, pooling, dropout and dense layer are also bearing a vital role in CNN. CNN can perform efficient image processing by using automatic feature selection by its convolution layer, feature reduction by pooling layer, and classification using dense layers. The structure of CNN network is given in Fig. 1.

Each layer of CNN has its own significance. Number of features extracted from each layer depends on the number of kernels and kernel size. Kernel weights are initialized using the random weight, which are trained in the course of model training. Output of the convolution layer is used as the input of the relu layer. Nonlinear activation function relu keeps the value of the convolution layer in a particular range. Relu activation function is the most popular in CNN due to its simplicity, irreconcilability, and non-negativity in nature. After Relu the dropout layer is used to prevent the model from overfitting. Then the pooling layer is used to down sample the feature map. It helps the representation to become invariant to small changes in input. Pooling layer operates on each feature map separately to create a new set of pooled features. Among different pooling such as max pooling, average pooling and so on we choose max pooling which uses the maximum from each patch of the feature maps. It helps to extract the low level features like the point and edge of the image. Higher level layers of the CNN are generally fully connected layers. These dense layers take the end result of the pooling layer and generate a classification decision. Last layer of the CNN model uses a softmax activation function which outputs a probability distribution of multi class classification. Regularization is a way to handle overfitting in the CNN model. Regularization reduces overfitting by adding penalty to loss function. Dropout is a way to handle overfitting in CNN by reducing interdependent learning. Once the structure of a CNN is defined, its internal weights are tuned to fit the target problem through back propagation.

Again there exist a number of hyper-parameters related to the each layer of CNN model such as kernel size, the number of kernels, stried in the kernel, activation function involved in convolution layer, kernel size associated with max-pooling, dropout rate used in dropout, size of the dense layer, learning rate, number of convolution, number of max-pooling, number of dense layers and so on. The outcome of the CNN model highly relies on these hyper-parameters. It is practically impossible to derive the near to optimal hyper-parameter configuration for a CNN manually by exploring all possible combinations under a reasonable cost. Hence proper hyper-parameter tuning of CNN is structured as an optimization problem with an objective to increase the overall performance of the CNN model.

3.2. Grey wolf optimization

In 2014 GWO was proposed by Seyedali Mirjalili by mimicking the grey wolves' social behavior, leadership hierarchy and hunting in group property (Mirjalili et al., 2014). In wildlife terrain, grey wolves generally live in a group. The group size is in range from 5 to 12. They maintain a strict social dominant hierarchy. Most dominant male or female wolves are represented as the alphas in the top level of hierarchy, who are mainly responsible for taking decisions about the wolf pack regarding their eating, sleeping, hunting, habitat and so on. All other wolves follow the alpha wolves. The next level wolves in the hierarchy are the beta wolves; they follow the decision of the alpha and control the low level wolves. The next category represents the delta wolves that help the alpha and beta wolves in hunting and searching prey. They guard the boundaries of territory; notify the other wolves about any danger and take care of the wounded and weak wolves. Omega wolves are the lowest level wolves, who obey the order of all other wolves. The success in hunting of the wolves mainly builds upon their social hierarchy. Social behavior of grey wolf can be modeled mathematically, by considering prey location as the optimum solution and representing the position of wolf as the solution in the search space. The alpha wolves are the best solution due to their closest distance to the prey. The beta and delta wolves are the next best solution respectively according to their social hierarchy. In search space omega wolves update their position according to the position of alpha, beta, and delta wolves. Let the position of alpha, beta, delta and omega wolves are represented in search space as X_α,X_β, X_δ, $X_\omega$respectively. Prey encircling, hunting, attacking and searching are the major steps of GWO. Prey encircling is the process by which wolves encircle the prey at the time of hunting, which is mathematically represented by equation (1) to equation (4).

$$\vec{D} = \left|\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)\right| \tag{1}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} . \vec{D} \tag{2}$$

$$\vec{C} = 2 \cdot \vec{a} \cdot \overrightarrow{rand}_1 - \vec{a} \tag{3}$$

$$\vec{A} = 2 \cdot \overrightarrow{rand}_2 \tag{4}$$

where $\vec{X}(t)$, $\vec{X}_p(t)$ indicates the position vector of grey wolf and prey at current iteration. $\vec{C}$, $\vec{A}$ are coefficient vectors. $\overrightarrow{\mathrm{rand}}_1$, $\overrightarrow{\mathrm{rand}}_2$ are two random vectors in $[0, 1]$. $\vec{a}$ is a vector whose value decreases over the iteration form 2 to 0. The hunting process of the prey is guided by the alpha wolves. The beta and delta also participate in this process. It is assumed that these three wolves are knowledgeable about the probable region of prey which helps to

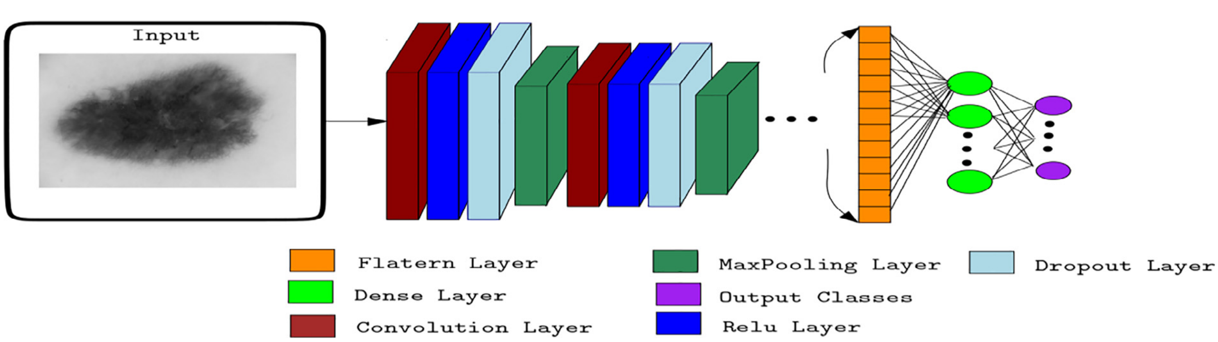


Fig. 1. CNN network.

derive three best search agents and these agents further help in updating the positions of other wolves as represented in equations (5) to (11).

$$\vec{D}_{\alpha} = \left|\vec{C}_1\vec{X}_{\alpha} - \vec{X}\right| \tag{5}$$

$$\vec{D}_{\beta} = \left|\vec{C}_2\vec{X}_{\beta} - \vec{X}\right| \tag{6}$$

$$\vec{D}_{\delta} = \left|\vec{C}_3\vec{X}_{\delta} - \vec{X}\right| \tag{7}$$

$$\vec{X}_1 = \vec{X}_{\alpha} - \vec{A}_1 \cdot \tag{8}$$

$$\vec{X}_2 = \vec{X}_{\beta} - \vec{A}_2 \cdot \vec{D}_{\beta} \tag{9}$$

$$\vec{X}_3 = \vec{X}_{\delta} - \vec{A}_3 \cdot \vec{D}_{\delta} \tag{10}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{11}$$

The attacking step is also same as exploitation, which is implemented by the factor $\vec{a}$. When pray stops, the moving wolves attack the pray. The value of $\vec{A}$ is a random value in range [−2r, 2r] and the value of r2, is in range [−1, 1]. The next position of search agent is any position between its recent position and prey position. So the attacking condition is suitable when $\left|\vec{A}\right| < 1$.

Searching or exploration for an optimum solution is modeled according to the searching behavior of wolves. Wolves diverge to search for prey and converge when they find prey. Wolves diverge for finding better prey if $\left|\vec{A}\right| > 1$, and converge towards the prey if $\left|\vec{A}\right| < 1$. A Random $\vec{C}$is used for avoiding local optimum and favoring exploration. It gives random value not only at the initial stage of the algorithm but also at the final stage of the algorithm, which enhances the exploration idea without bias. The basic steps of GWO algorithm is detailed as follows:

- Initialize the population of grey wolves and the controlling parameters a, A, C, max_iteration.
- Calculate the fitness of initial population using objective function
- Represent the best wolf as $\vec{X}_{\alpha}$, the second best wolf as $\vec{X}_{\beta}$, the third best wolf as $\vec{X}_{\delta}$
- while{it < max_iteration} do
- for{each search agent}
- update the position of current agent using equation (11).
- update a, A, C
- calculate the fitness of all agents
- update $\vec{X}_{\alpha}$, $\vec{X}_{\beta}$,$\vec{X}_{\delta}$
- end while
- return $\vec{X}_{\alpha}$

4. A GWO based hyper-parameter optimized CNN model for skin cancer detection

4.1. Problem definition

In this research, identifying the hyper- parameters of CNN is defined as an optimization problem; in which the parameters are represented by a list of real numbers. The objective is to optimize the hyper-parameters using GWO then use it for skin cancer classification to achieve more classification accuracy. The objective function is defined as follows:

$$acc = CNN\left(\overrightarrow{Hp}, \vec{W}, Td_i\right) \tag{12}$$

$$\underset{\overrightarrow{Hp}\epsilon\mathbb{R}^k}{maximizeacc}\ CNN\left(\overrightarrow{Hp}, \vec{W}, Td_i\right) wherei < i_{max} \tag{13}$$

Equation (12) represents a function denoting the architecture of CNN, taking inputs, $\overrightarrow{Hp}$, $\vec{W}$, Td_i.$\overrightarrow{Hp}$, represents the hyper-parameter vector of k dimension, $\vec{W}$represents the weight vector of CNN, $Td_i$is some selected data from training data respectively. This function returns the accuracy of the model. Equation (13) represents the objective function to maximize accuracy of CNN for the hyper-parameters. One of the important parameteri_{max}, is set by the user to control the number of iterations required by CNN for optimizing the hyper-parameters. A larger i_{max}value incurs more time towards optimization compared to smaller value. So the user has to set i_{max}value tactfully to make the model cost effective.

4.2. Overall design

Fig. 2 illustrates the schematic diagram of the proposed GWO based Hyper-parameter Optimized CNN model. The proposed model has three basic steps such as image preprocessing, optimizing hyper-parameters using GWO and creation, training of CNN using the resultant optimized hyper-parameters. The details are explained as follows:

4.2.1. Image preprocessing

After collecting the skin cancer images, they are preprocessed through the steps of image resizing, color conversion and filtering. Image resizing leads to reshape the random size data to a similar shape for the model. Reducing the images to genuine size can save the training and testing time of the CNN model. The collected ISIC data set data set has 25,000 Dermoscopy images, of size varying from 700 × 700 to 1000 × 1000. Multiple layers of CNN take a quite amount of time to process such a large image. So it puts the necessity of reducing the size of images keeping the aspect ratio preserved (Amanatiadis and Andreadis, 2008). Nearest neighbor (NNI) interpolation (Olivier and Hanqiang, 2012), Bilinear interpolation (BI) (Parsania and Virparia, 2016), Resampling using pixel area relation (RPAR) (Wang and Yuan, 2014), Bicubic interpolation over 4X4 pixel neighborhood(BIPN) (Parsania and Virparia, 2016), and Lanczos interpolation over 8X8 neighborhood (LIN) (Parsania and Virparia, 2016) are some of the popular image scaling algorithms used in literature for image preprocessing.

In our proposed model, BI is used for image resizing due to its simple features. The resulting accuracy obtained through BI is also compared with NNI, RPAR, BIPN and LIN approaches. Bipolar interpolation uses linear interpolation in X and Y directions. For finding the unknown value at point (p,q) i.e. $f(p,q)$, assuming that value of function f at four point $N_{11} = (p_1, q_1)$,$N_{12} = (p_1, q_2)$,$N_{21} = (p_2, q_1)$, $N_{22} = (p_2, q_2)$ are known. Linear interpolation on X axis is calculated using the following formula:

$$f(p, q_1) \approx \frac{p_2 - p}{p_2 - p_1} f(N_{11}) + \frac{p - p1}{p_2 - p_1} f(N_{21}) \tag{14}$$

$$f(p, q_2) \approx \frac{p_2 - p}{p_2 - p_1} f(N_{12}) + \frac{p - p1}{p_2 - p_1} f(N_{22}) \tag{15}$$

Then interpolation along Y axis calculated using equation (16) as follows:

$$f(p, q) \approx \frac{q_2 - q}{q_2 - q_1} f(p, q_1) + \frac{q - q1}{q_2 - q_1} f(p, q_2) \tag{16}$$

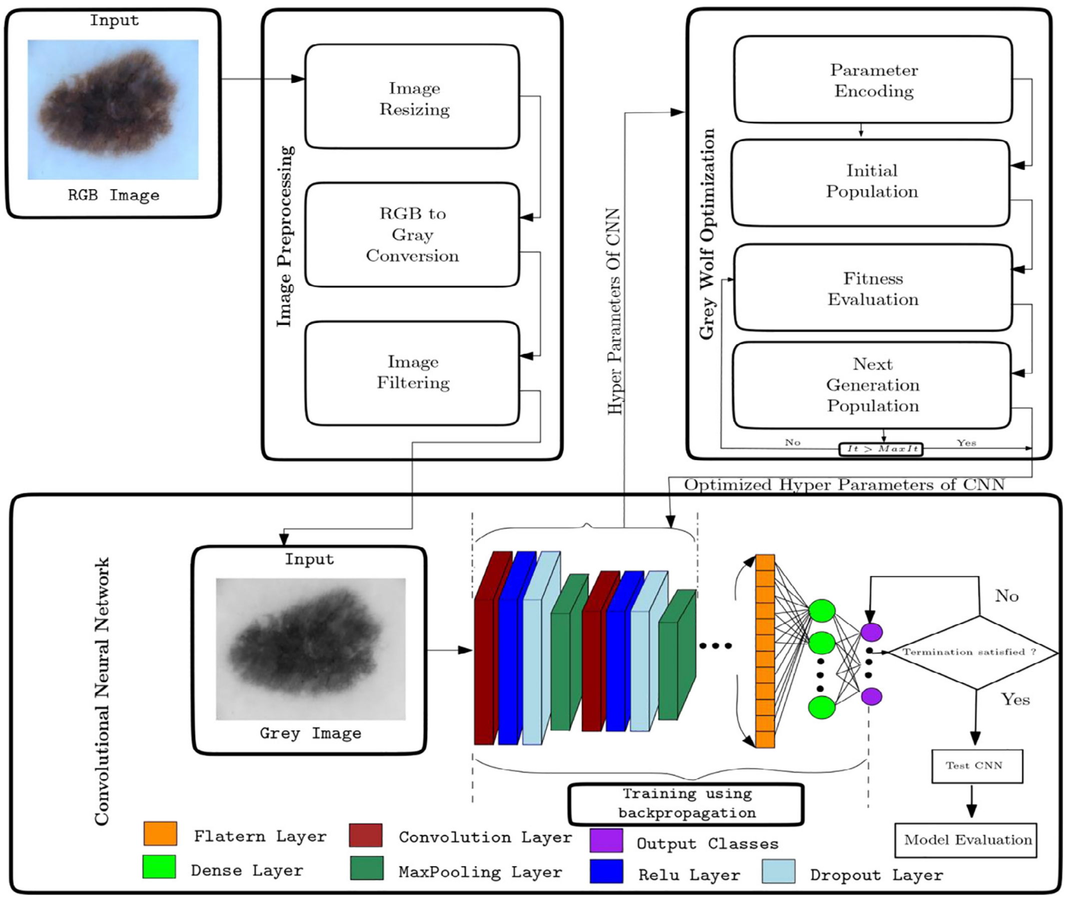


Fig. 2. Proposed GWO based Hyper-parameter Optimized CNN model.

Downsizing an image by simply skipping alternate pixels can lose some important features of the image whereas image interpolation provides a best approximation of a pixel's intensity value based on the values at surrounding pixels. Interpolation can be applied both for upsizing or downsizing an image. The process of downsizing of a gray image using equation (14) to (16) is explained as follows:

Let f(x,y) represent the intensity values at the integral lattice locations of a gray image. In Fig. 3, the part A represents the intensity matrix of a grey image. By applying equation (14) and (15) on the values at column 1 and 3 on each row 1 and 3, the interpolated value of pixel (2, 1) and pixel (2,3) is calculated as follows:

$$I_{2,1} = \frac{3-2}{3-1}.10 + \frac{2-1}{3-1}.20 = 15$$

$$I_{2,3} = \frac{3-2}{3-1}.30 + \frac{2-1}{3-1}.40 = 35$$

Then the intensity at pixel (2, 2) is calculated using equation (16) on the previously calculated interpolated values as follows:

$$I_{2,2} = \frac{3-2}{3-1}.15 + \frac{2-1}{3-1}.35 = 25$$

Similarly values of (3, 2), (2, 3), and (3, 3) are calculated. The corresponding interpolated downsize image is represented in part B of Fig. 3. The same procedure is applied on the red, green and blue channel of the RGB image to downsize it.

After observing the ICSC skin cancer data set, it is figured out that that major area of dermoscopy skin image is gray color. As intensity of the pixel plays an important role in classification, so after resizing the images, they are converted from color to grayscale so as to further save the training time. Out of some popular methods of color conversion, such as lightness, average, luminosity, in this model we have used the luminosity approach to convert the color image to a grayscale image. Luminosity uses the weighted

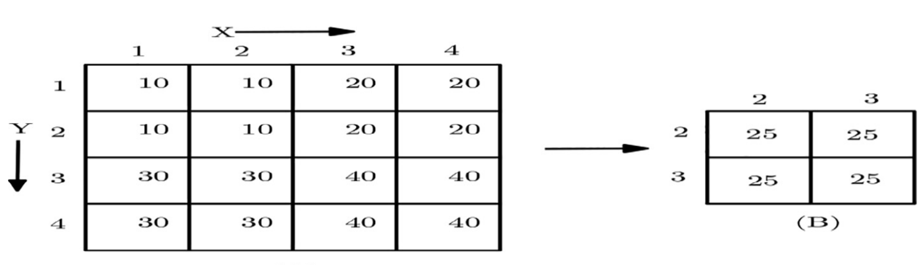


Fig. 3. Illustration of image downsizing using BI.

average concept but keeps the human perception in account, i.e. human beings are more sensitive to green color then others so weight given to green color is more than other colors. Luminosity method uses the equation (17) for converting color to gray scale image.

$$gray(R, G, B) = 0.21R + 0.72G + 0.07B \tag{17}$$

After color conversion, the image filtering approach is applied to remove the unwanted features from the images such as hair, noise, small unaffected area and so on, which may enhance the further classification performance. Gaussian (He et al., 2010), median, and bilateral (Van De Ville et al., 2003) are some popular image filtering approaches used in image processing. In the proposed model, we have used the bilateral filter as the image smoothing tool and its performance is also compared with the averaging, Gaussian and median filtering approach. The bilateral filter is a nonlinear image smoothing filter that replaces the pixel value by the weighted average of the nearby pixel. Weight of this filter is based on Gaussian distribution. Using a bilateral filter the intensity of a pixel is derived using the following equation:

$$Intensity(p) = \frac{1}{w}\sum_{p_i \in S} Intensity(p_i) f_r(\| Intensity(p_i) - Intensity(p) \|) g_s(\| p_i - p \|) \tag{18}$$

$$W = \sum_{p_i \in S} f_r(\| Intensity(p_i) - Intensity(p) \|) g_s(\| p_i - p \|) \tag{19}$$

4.2.2. Creation of CNN architecture

CNN model is the array of different layers such as convolution, relu, dropout, pooling and dense layers. Each layer has its own significance. Even though CNNs have shown promising results in several classification problems, finding the best CNN layout for a specific application is far from trivial (Lopez-Rincon et al., 2018). Layouts proposed in literature are mostly derived through trial and error basis or impelled by the earlier related works. In our specific case, by referring to the related works done in (Wang et al., 2019; Yoo, 2019; Szegedy et al., 2015; Badrinarayanan et al., 2015; Li et al., 2019) the initial CNN structure is constructed with 3 convolution layers, 3 relu layers, 3 dropout layers, 3 pooling layers, one flatten layer and two dense layers. Each layer of CNN is associated with some hyper-parameters such as numbers of kernels and their size are required to be specified in the convolution layer of the CNN. Kernel size is used to select the features. The range of kernel size should not be too small such as 2×2 or 1×1 because it helps to fine grain the local and no information of neighbor to include in the feature extraction. Also it should not be too large as greater than 6 X 6 which leads to excluding the fine details. In our study, range of kernel size is taken in between $3x3$ and$5x5$. The number of kernels in each layer represents the features to be selected for the next layer. The initial layer selects simple features whereas later layers select complex features. So in the initial layer the number is set less compared to later layers, where it may lie in the range of 40 to 250. Pooling layers are controlled by its pooling size, that down sample the feature for the next layer. Large pooling size can exclude the fine details and small size can fine grain the local. So in this study the range is selected in between $3x3$ and$5x5$. Dropout layers are controlled by dropout rate to regularize the model. Dropout rate has a great impact on preventing the network from overfitting. Range of dropout rate is considered in between 0.2 to0.4. Dense layers play the role for classification. Number of convolutions and the number of poolings control the entire architecture of CNN. Increases in the number of convolutions can over-fit and less in numbers can under-fit the model. More the number of pooling may exclude the features and less may include the same features repeatedly. So in this study simulation is done by keeping the range of number of convolution layers and pooling layers in between 2 and 10. These numbers are fixed after several tests as a good balance between performance and time.

4.2.3. Hyper-parameter optimization of CNN using GWO

Hyper-parameters annotation of CNN varies from researchers to researchers that lead to different optimization goals. Some tackle the problem in a narrow sense by considering the hyper-parameters of different layers keeping the architecture of the network fixed. Whereas some consider hyper-parameters in a broad sense including the number of layers, learning rate, dropout rate and so on. In this study we only focus on the former one. After deciding the architecture of CNNN, its hyper-parameters optimization using GWO is conducted through four steps i.e. parameter encoding, population initialization, fitness evaluation, and next generation population creation. In parameter encoding step, the hyper-parameters of the CNN such as kernel size *(Ks)* and number of kernel *(Nk)* associated with convolution layer (Cv), kernel size in pooling layer (Ps), dropout rate (Dr) are encoded as a vector having k number of parameters. With l number of convolution layers, m number of pooling layers and d number of dropout rate, the encoded parameters is represented as a k dimensional vector with k=(2 L + m + d). Values of these parameters are set to some random values in the specified range. Each parameter vector is known as an agent, where the i^{th}parameter vector is defined as in equation (20).

$$\overrightarrow{HP}_i = \{P_{i1,} P_{i2,} P_{i3,} \ldots . P_{ik}\} \tag{20}$$

With 3 convolution layers, 3 dropout layers and 3 pooling layers, the parameter vector will include total 12 hyper-parameters (Cv1_Nk, Cv1_Ks, Mp1_Ps, DL1_Dr, Cv2_Nk, Cv2_Ks, Mp2_Ps, DL2_Dr, Cv3_Nk, Cv3_Ks, Mp3_Ps, DL3_Dr) where Cv_i represents the i^{th} convolution layer, Mp_i represents i^{th} pooling layer, DL_i represents i^{th} dropout layer, Nk represents the number of kernels, Ks represents kernel size, Ps represents pooling size, Dr represents dropout rate.

After encoding the CNN hyper-parameters, initial populationS_n, which is the collection of n agents are initialized randomly as follows:

$$S_n = \left\{ \overrightarrow{HP}_{1,} \overrightarrow{HP}_{2,} \overrightarrow{HP}_3 \cdots \cdots .. \overrightarrow{HP}_n \right\} \tag{21}$$

where $\overrightarrow{HP}_i$, is ak dimension vector of CNN hyper-parameters, P_{ik} is a vector of random value of range specified in encoding scheme section.

The most time consuming part of hyper-parameter optimization of the CNN, is the training which is also used as the fitness function of the heuristic approach. To find the fitness of candidate solutions, some researchers have trained the model until it converges (Szegedy et al., 2015), some others have used selective data sets as in (Xie et al., 2020). In this study, we have trained the model using a light weight model with a small random sample and discard the unnecessary iteration by checking if the change in the accuracy is minute or not. Selected data from ICSC data set, for a predefined iteration has been used to find the fitness of an agent using the equation (13). The coefficient vectors $\overrightarrow{A}$ and $\overrightarrow{C}$ of GWO are generated using equation (3) and (4) respectively. The random parameters$\overrightarrow{A}$, and $\overrightarrow{C}$ assist the candidate solution to have hyper-spear in different radii. After calculating the fitness value of each agent using equation (12), the best three agents of the population are represented as the alfa, beta, and delta wolves respectively. Then the position of the other agents are updated according to the position of $\overrightarrow{X}_\alpha$, $\overrightarrow{X}_\beta$, $\overrightarrow{X}_\vartheta$ using equation (11). The process of fitness calculation of new agents, selection of three best agents and position updating of other agents is repeated for per-specified iterations. Finally the best agent resulting in maximized accuracy of CNN is considered as the optimized hyper-parameters of CNN and these optimized hyper-parameters are used for creating optimized CNN models.

The general procedure of the proposed method is illustrated in Fig. 4.

4.2.4. Model evaluation criteria

The proposed model is finally evaluated based on some standard classification matrices such as Accuracy, Precision, Recall, F-Score, Macro F, Average Precision (AP) (Siegel et al., 2019; Szegedy et al., 2015; Badrinarayanan et al., 2015; Li et al., 2019) along with the Categorical cross entropy loss (loss) value defined as follows:

$$\mathrm{Loss} = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i \mathrm{Loss} = -\sum_{i=1}^{N} y_i \cdot \log \hat{y}_i \tag{22}$$

where N is the number of classes, $\hat{y}_i$ is the model predicted value for ith class, y_i is the corresponding target value.

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \tag{23}$$

$$\mathrm{Precisson} = \frac{\mathrm{Tp}}{\mathrm{Tp} + \mathrm{Fp}} \tag{24}$$

$$\mathrm{Recall} = \frac{\mathrm{Tp}}{\mathrm{Tp} + \mathrm{Fn}} \tag{25}$$

$$F_{\mathrm{Score}} = \frac{2\mathrm{Tp}}{2\mathrm{Tp} + \mathrm{Fp} + \mathrm{Fn}} \tag{26}$$

$$MacroF = \left(\sum\nolimits_{i=1}^{numofclasses} F_{Score\,i}\right)/numofclasses \tag{27}$$

$$AP = \left(\sum\nolimits_{i=1}^{numofclasses} Precisson_i\right)/numofclasses \tag{28}$$

where Tp, Tn, Fp, Fn, refer to true positive, true negative, false positive, and false negatives. However, metrics were measured at the whole image level, rather than the pixel level.

5. Experimental result analysis

In this section the effectiveness of the proposed model is verified by comparing it with two other nature inspired techniques such as PSO and GA applied on the ISIC skin lesion multiclass data set. All the experiments are conducted in python by using the Keras, Scikit-learn and Opencv library on an Intel Xeon processor with 128 GB of RAM with one NVIDIA GeForce GTX 1080 Ti GPU card. The ISIC dataset is a collection of 25330dermoscopic skin lesion images of 8 categories such as Melanoma (Me), Melanocytic Nevus (MN), Basal Cell Carcinoma (BCC), Actinic Keratosis (AK), Benign Keratosis (BK), Dermatofibroma (Bb), Vascular Lesion (VL) and Squamous Cell Carcinoma (SCC). The 80% of images is selected for training purposes; whereas 20% is used for testing. Table 1 describes the details about the dataset used in simulation. The number of images corresponding to each class used in training and test set is represented in Table 2.

The proposed model is implemented on two CNN models; one is the lightweight and another is the main model. The Lightweight model is used for finding the fitness of agents in GWO. The main CNN model is designed after the hyper- parameters optimized by GWO. The light weight model is constructed with 3 convolution layers, 3 relu layers, 3 dropout layers, 3 max-pooling layers, one flatten layer and two dense layers. The number of kernels corresponding to three convolution layers is set to 50, 70, 100 respectively and the kernel size of the convolution layers are fixed at 3. The kernel size of the pooling layer is fixed at 2 whereas the dropout rate is set to 0.2. The first dense layer includes 500 nodes whereas the second dense layer which is the output layer includes 8 nodes corresponding to 8 classes of the original dataset. The output layer uses the softmax activation function to produce the final

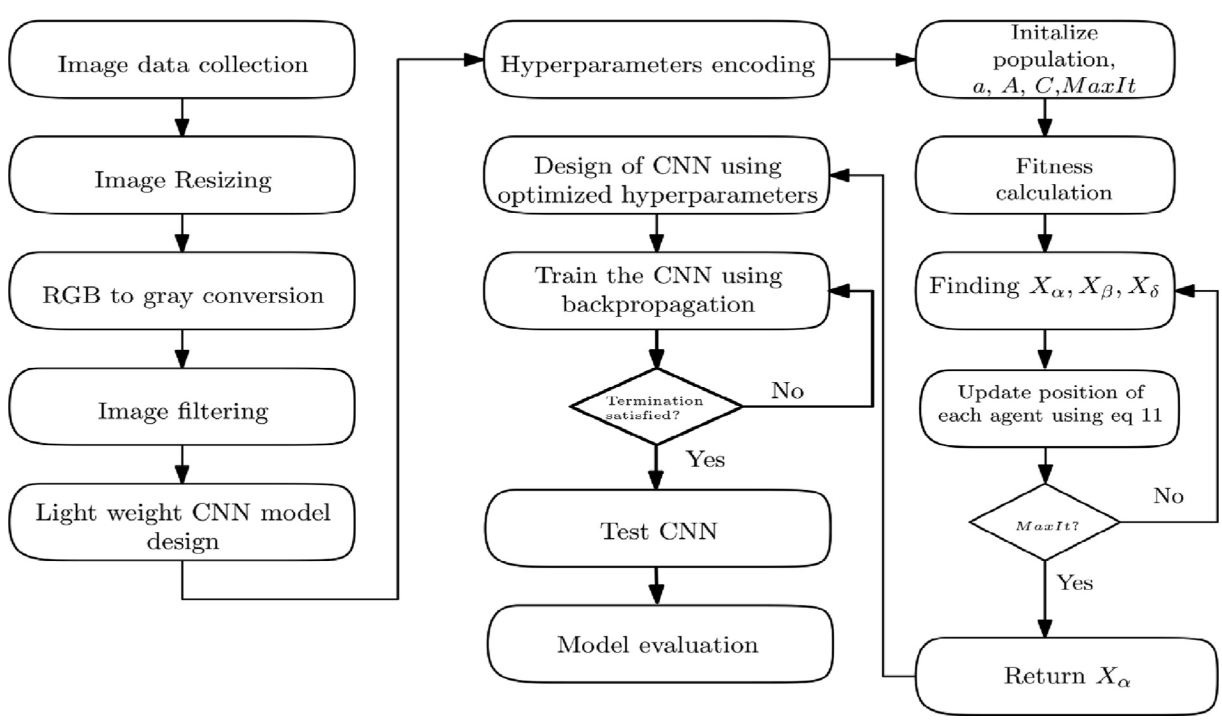


Fig. 4. Flow diagram of the proposed model.

Table 1
ISIC dataset.

Dataset name	Total number of images	Total number of classes	Number of images in training set (80%)	Number of images in test set (20%)
ISIC	25,330	8	20,264	5066

output class. The lightweight CNN is trained by a stochastic gradient descent optimizer by keeping the learning rate fixed at 0.0010, and batch size at 64.

Before using image preprocessing on the actual data set, the lightweight CNN model is tested with a sample data set for 200 epochs, which gives 75.23% training accuracy. The sample dataset consists of 1000 images randomly selected from the ISIC dataset of size 700x700 to 1000x1000. To save the training time instead of three channel images (RGB Image), the gray scale images are used in experimentation by converting RGB to gray image. Due to the huge number of input images and each one having large size, without image preprocessing, the accuracy may not be achieved up to the expectation level and the time for training will be enormous. To reduce the time of training and for improving the performance of the model, five image resizing techniques such as NNI, BI, RPAR, BIPN and LIN are tested over the sample dataset. The accuracy of the light weight CNN model over the sample dataset using the different image resizing techniques are presented in Table 3. From Table 3, it is clearly observed that BI image resizing is producing better accuracy compared to other four approaches. But the model accuracy is not up to the mark. To increase the accuracy, further experimentation is done with the image filtering approaches to remove the unwanted features from the image. The accuracy of light weight CNN with four different types of image filtering techniques such as Averaging, Gaussian, Median, and Bilateral applied over the sample dataset is summarized in Table 4. From Table 4, Gaussian filtering is found to provide better accuracy and hence it is further used in the proposed model. Fig. 5 clearly illustrates the outcome of different steps of image preprocessing on a sample skin cancer image data.

Though the architecture of the CNN is selected by referring to the architectures proposed in (Wang et al., 2019; Yoo, 2019; Szegedy et al., 2015; Badrinarayanan et al., 2015; Li et al., 2019), further simulation is also performed on different architectures especially changing the number of convolution layer, relu layer, pooling layer and number of kernels in convolution. For all these architectures, the other parameters of the network such as number of dense layers, kernel size of each layer, dropout rate, learning rate, epoch, and optimizer are kept fixed. Analyzing the accuracy and loss values represented in Table 5, it is clear that the CNN architecture with 3 convolution layers, 3 relu layers, 3 dropout layers, 3 max-pooling layers and 2 dense layers is producing better values compared to the other two architectures. So keeping this architecture fixed, further simulation is done to optimize the hyper-parameters of the CNN such as the kernel size (Ks) and number of kernels (Nk) associated with convolution layer (Cv), kernel size in pooling layer (Ps) and dropout rate (Dr).

Table 2
Class wise count for ISIC data set.

Image class	Number of image	Number of images in training set	Number of images in test set
Me	4522	3592	930
MN	12,875	10,298	2577
BCC	3323	2701	622
AK	867	689	178
BK	2623	2058	565
Db	239	201	38
VL	253	199	54
SCC	628	526	102
Total :	25,330	20,264	5066

Table 3
CNN accuracy for different image resizing technique.

Image Resizing Technique	Accuracy in (%)
NNI	74.37
BI	**76.56**
RPAR	76.25
BIPN	76.28
LIN	75.15

Table 4
CNN accuracy using different image filtering.

Image filtering method	Accuracy in (%)
Averaging	76.50
Gaussian	**77.37**
Median	76.75
Bilateral	75.87

After fixing the architecture of CNN, the hyper-parameters of CNN are encoded so as to fit into the population of GWO. With 3 convolution layers, 3 pooling layers and 3 dropout rates, the size of the encoded vector is set to 12 representing the hyper-parameters of the CNN. The range of NK in convolution layer1 is randomly set in between 50 and 100, in convolution layer 2 is set in between 100 and 150, and in convolution layer 3 is set in between 150 and 200. The value KS in each of the three convolution layers is set randomly within the range 3X3 to 5X5. The value of PS in each pooling layer is taken within the range 2X2 to 4X4. The range of DR in all dropout layers is set between 0.2 and 0.4. With a population size 10, 10 encoded vectors are generated randomly taking values from the specified range to represent the initial populationS_n. The same population is used in all the three optimization techniques. The Fitness value i.e. the maximum accuracy of the fittest agent for different optimization techniques is given in Table 6. Fitness of fittest agent over the iterations is plotted in Fig. 6. Table 5 and the Fig. 7 clearly show the superiority of GWO approach of hyper-parameter optimization of CNN compared to the other two techniques over the sample data set. The optimized hyper-parameters resulting from the three techniques are also depicted in Table 7. Form Table 7, it is observed that the number of kernels in deeper layers is more than the number of kernels in initial layers, kernel size is generally 3X3 in convolution layer, maxpool size is 2X2 in each layer, and dropout rate is obtained as 0.2.

Then the optimized CNN model is created using the hyper-parameters of Table 7 and is trained using the sample ISIC data set divided into 80% training and 20% testing set. Table 8 presents the training, testing accuracy, training and testing loss of the optimized CNN model for different nature inspired algorithms. From table 8, it is observed that the training accuracy values of PSO, GA are better than GWO, but testing accuracy of GWO is better than the other two nature inspired algorithms. Although training loss is equal in each case, but testing loss is less in case of GWO optimized CNN.

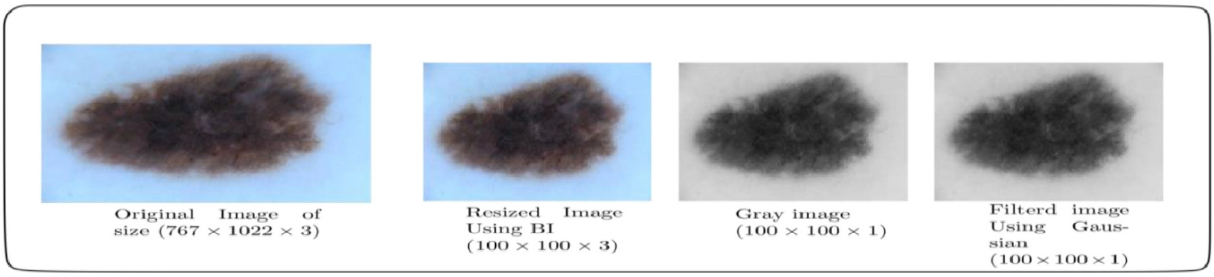


Fig. 5. Outcome of different steps of image preprocessing.

Table 5
Outcome of CNN with different architecture.

Parameters	CNN1	CNN2	CNN3
Number of convolution	3	2	4
Number of pooling	3	2	4
Number of relu	3	2	4
Number of dence layer	2	2	2
Number of kernels in convolution	50, 70, 100	50, 70	50, 70, 100, 130
Accuracy	77.37	74.567	76.232
Loss	6.083	8.474	7.422

Table 6
Fitness of fittest agents.

Optimization Technique	Fitness (Training Accuracy in (%))
GWO	**81.46**
PSO	81.22
GA	81.01

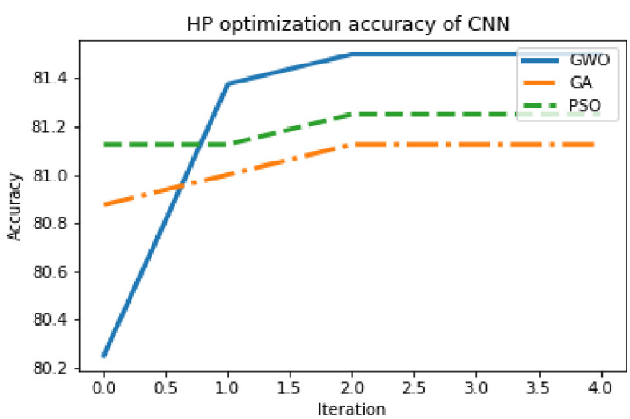


Fig.6. Fitness of fittest agent versus iteration.

The accuracy and loss of various hyper-parameter optimized CNN models obtained for the sample data set is depicted in Fig. 7. It clearly shows that the accuracy of the optimized models is initially fluctuating but as the model is trained more, the curve becomes smooth. Loss plot shows that GWO loss transition is smoother than others.

Finally the hyper-parameter optimized CNN model is trained using the original ISIC skin cancer data set with 25,330 images for 200 epochs. 80% of data is used for training and 20% is used for testing. Table 9 gives the comparison of accuracy and loss of the proposed model, using different nature inspired techniques on actual dataset. Simulation results of Table 9 infer that the proposed model is able to efficiently detect the classes of skin cancer with a testing accuracy up to 98.33% which is around 4% more compared to PSO based model and 1% more compared to GA based model. Similarly with the proposed model the testing loss realized is around 0.17% which is 39.2% less compared to PSO based model and 15% less compared to GA based model. Fig. 8 plots the accuracy rate and the loss corresponding to each epoch generated over the original dataset for the three hyper-parameter optimization techniques. The accuracy plot clearly reveals that the accuracy with GWO is increasing more smoothly than others. After 50 epoch, all techniques give approximate accuracy, but GWO gives more accuracy than others. From the loss plot, it is also observed that GWO gives better loss rate than others. Table 10 presents the confusion matrix resulting from the GWO based hyper-parameter optimized CNN for 5066 testing data set. Finally quite acceptable precision, recall, f-score and the true positive values obtained on the test dataset by using the proposed model is presented in Table 11. A comparison of the proposed GWO based hyper-parameter optimized CNN with PSO and GA based hyper-parameter optimized CNN model in terms of two other standardized classification matrices such as AP and Macro F is presented in Fig. 9. The AP score observed for the proposed

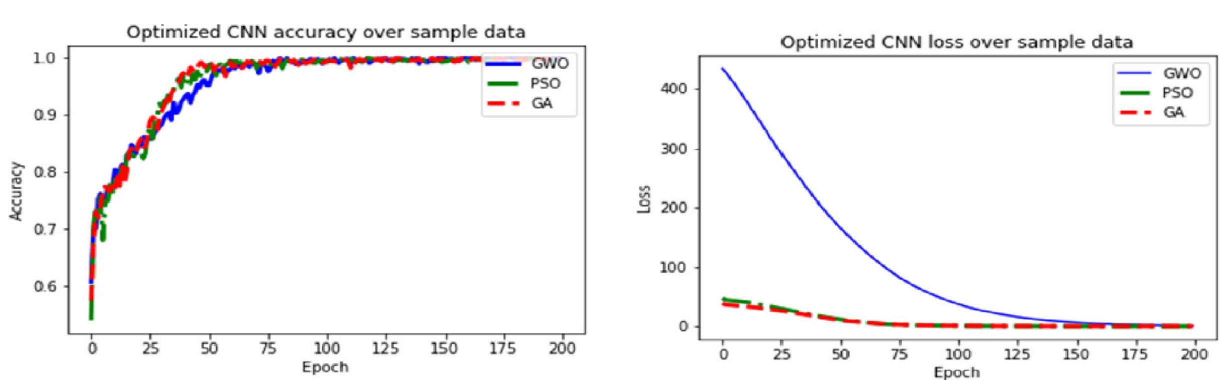


Fig.7. (a) Accuracy (b) Loss of various hyper-parameter optimized CNN model with respect to epoch for sample data set.

Table 7
Optimized hyper-parameters of CNN along with resulting accuracy.

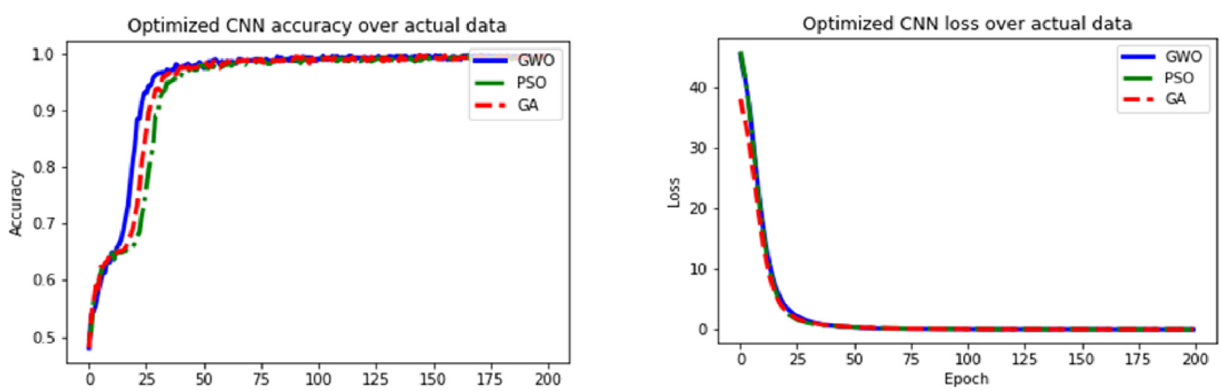
Optimization Technique	Cv1 Nk	Cv1 Ks	Mp1 Ps	Dl1 Dr	Cv2 Nk	Cv2 Ks	Mp2 Ps	Dl2 Dr	Cv3 Nk	Cv3 Ks	Mp3 Ps	Dl3 Dr	Fitness (%)
GWO	55	3	2	0.1	130	3	2	0.1	155	3	2	0.1	81.46
PSO	55	3	2	0.2	130	3	2	0.2	195	3	2	0.2	81.22
GA	100	3	2	0.21	114	3	2	0.2	200	3	2	0.2	81.01

Table 8
Accuracy and Loss % of the hyper-parameter optimized CNN on sample dataset.

Optimization technique	Training accuracy of CNN in (%)	Testing accuracy of CNN in (%)	Training loss of CNN in (%)	Testing loss of CNN in (%)
GWO	99.12	**93.00**	0.18	**0.58**
PSO	99.25	87.12	0.18	0.60
GA	99.25	87.00	0.18	0.75

Table 9
Accuracy and Loss % of the hyper-parameter optimized CNN on actual dataset.

Optimization technique	Training accuracy of CNN in (%)	Testing accuracy of CNN in (%)	Training loss of CNN in (%)	Testing loss of CNN in (%)
GWO	**98.70**	**98.33**	**0.13**	**0.17**
PSO	97.58	94.66	0.20	0.28
GA	97.97	97.33	0.18	0.20



(a) (b)

Fig. 8. (a) Accuracy, (b) Loss of various hyper-parameter optimized CNN model with respect to epoch for actual dataset.

Table 10
Confusion matrix of test data using the proposed model.

	Predicted Label								
Actual Label		**Me**	**MN**	**BCC**	**AK**	**BK**	**DB**	**VL**	**SCC**
	Me	922	8	0	0	0	0	0	0
	MN	10	2562	2	0	2	0	1	0
	BCC	1	5	615	0	1	0	0	0
	AK	0	1	0	177	0	0	0	0
	BK	0	2	0	0	563	0	0	0
	DB	0	2	0	0	0	36	0	0
	VL	0	5	0	0	0	0	49	0
	SCC	0	0	0	0	0	1	0	101

Table 11
Classification output of test data obtained by the proposed model.

Class Label	Precision	Recall	F-Score	True positive
Me	0.99	0.99	0.99	930
MN	0.99	0.99	0.99	2577
BCC	1.00	0.99	0.99	622
AK	1.00	0.99	1.00	178
BK	0.99	1.00	0.99	565
DB	1.00	0.85	0.97	38
VL	0.98	0.91	0.94	54
SCC	1.00	0.99	1.00	102

model is 99.37 which is around 4.1% more than the PSO based model and 2.2% more than the GA based model. Similarly the Macro F value observed for the proposed model is 98.37 which is around 2.3% more than the PSO based model and 0.9% more than the GA based model.The experimental results demonstrated in Tables 8 to 11 clearly demonstrate the superiority of the proposed method towards the other methods included in the study. The performance of the proposed GWO based automated hyper-parameter optimized CNN is good enough for diagnosing the different classes of the skin cancer from its image in different environmental conditions.

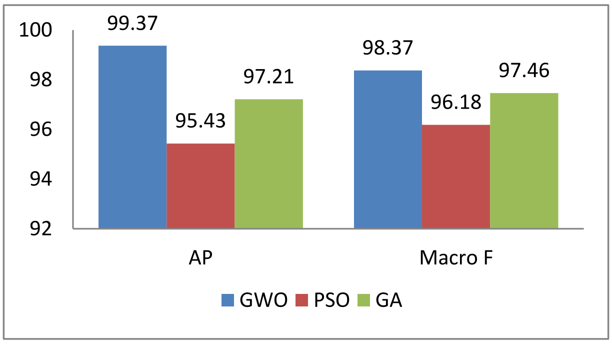


Fig. 9. Performance comparison of the three hyper parameter optimized CNN.

6. Conclusion

Classifying skin cancer manually in the early stage from the dermoscopic images is a quiet challenging task even for the experts, which has paved the path for its efficient automation. In this paper, an automated cost effective CNN is developed using GWO technique to diagnose the category of skin cancer from the input images. The GWO algorithm is adopted here to select the relevant hyper-parameters of the CNN so as to optimize the CNN architecture by improving its performance in addressing the skin cancer multi class classification problem. Proper image preprocessing by using BI for image resizing followed by color to gray scale image conversion and Gaussian filtering approach is also incorporated in the model to save the overall training time of CNN. After preprocessing, the hyper-parameters of the CNN are encoded efficiently so as to fit into the optimization algorithm. The efficacy of the model is verified by comparing it with PSO and GA based hyper-parameter optimization approaches of the same CNN model over the ISIC skin lesion multiclass data set. The final outcome clearly represents the better achievement of the proposed model in terms of accuracy and loss rate of CNN both during training and testing phase compared to the other two optimization techniques included in the experiment.

In future, the work can further be extended by including more hyper-parameters such as training size, training rate, regularization rate and activation functions in the process of optimization. The model performance can also be enhanced by using suitable optimization algorithms like Arithmetic Optimization Algorithm (Abualigah et al., 2021), hybrid Salp Swarm Algorithm (Abualigah et al., 2020), Grasshopper optimization algorithm (Abualigah and Diabat, 2020), Group search optimizer (Abualigah, 2021) and so on in the image preprocessing and weight updating phase of the model. The optimized model can be utilized in other domains with binary and multiclass datasets.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Abualigah, L., 2021. Group search optimizer: a nature-inspired meta-heuristic optimization algorithm with its results, variants, and applications. Neural Comput. Appl. 33 (7), 2949–2972.
Abualigah, L., Diabat, A., 2020. A comprehensive survey of the Grasshopper optimization algorithm: results, variants, and applications. Neural Comput. Appl., 1–24
Abualigah, L., Shehab, M., Diabat, A., Abraham, A., 2020. Selection scheme sensitivity for a hybrid Salp Swarm Algorithm: analysis and applications. Eng. Computers, 1–27.
Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., Gandomi, A.H., 2021. The arithmetic optimization algorithm. Comput. Methods Appl. Mech. Eng. 376, 113609. https://doi.org/10.1016/j.cma.2020.113609.
Agarwal, R., Sharma, H., 2020. A New Enhanced recurrent extreme learning machine based on feature fusion with CNN Deep features for breast cancer detection. In: Advances in Computer, Communication and Computational Sciences, pp. 461–471.
Akbari, R., Ziarati, K., 2011. A multilevel evolutionary algorithm for optimizing numerical functions. Int. Jo. Industr. Eng. Comput. 2 (2), 419–430.
Albelwi, S., Mahmood, A., 2017. A framework for designing the architectures of deep convolutional neural networks. Entropy 19 (6), 242.
Amanatiadis, A., Andreadis, I., 2008. Performance evaluation techniques for image scaling algorithms. In: 2008 IEEE International Workshop on Imaging Systems and Techniques. IEEE, pp. 114–118.
Badrinarayanan, V., Handa, A., &Cipolla, R. (2015). Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling. arXiv preprint arXiv:1505.07293.
Chen, X., Kopsaftopoulos, F., Wu, Q., Ren, H., Chang, F.K., 2019. A self-adaptive 1D convolutional neural network for flight-state identification. Sensors 19 (2), 275.
Cui, H., Bai, J., 2019. A new hyperparameters optimization method for convolutional neural networks. Pattern Recogn. Lett. 125, 828–834.
Garg, S., Kaur, K., Kumar, N., Kaddoum, G., Zomaya, A.Y., Ranjan, R., 2019. A hybrid deep learning-based model for anomaly detection in cloud datacenter networks. IEEE Trans. Netw. Serv. Manage. 16 (3), 924–935.
He, K., Sun, J., Tang, X., 2010. Guided image filtering. In: European Conference On Computer Vision. Springer, Berlin, Heidelberg, pp. 1–14.
Kharazmi, P., AlJasser, M.I., Lui, H., Wang, Z.J., Lee, T.K., 2017. Automated detection and segmentation of vascular structures of skin lesions seen in Dermoscopy, with an application to basal cell carcinoma classification. IEEE J. Biomed. Health. Inf. 21 (6), 1675–1684.
Kumaran, N., Vadivel, A., Kumar, S.S., 2018. Recognition of human actions using CNN-GWO: a novel modeling of CNN for enhancement of classification performance. Multimedia Tools Appl. 77 (18), 23115–23147.
Li, Y., Xiao, J., Chen, Y., Jiao, L., 2019. Evolving deep convolutional neural networks by quantum behaved particle swarm optimization with binary encoding for image classification. Neurocomputing 362, 156–165.
Lopez-Rincon, A., Tonda, A., Elati, M., Schwander, O., Piwowarski, B., Gallinari, P., 2018. Evolutionary optimization of convolutional neural networks for cancer miRNA biomarkers classification. Appl. Soft Comput. 65, 91–100.
Ma, B., Li, X., Xia, Y., Zhang, Y., 2020. Autonomous deep learning: A genetic DCNN designer for image classification. Neurocomputing 379, 152–161.
Maglogiannis, I., Doukas, C.N., 2009. Overview of advanced computer vision systems for skin lesions characterization. IEEE Trans. Inf Technol. Biomed. 13 (5), 721–733.
Mirjalili, S., Mirjalili, S.M., Lewis, A., 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61.
Mohakud, R., Dash, R., 2019. Survey on hyperparameter optimization using nature-inspired algorithm of deep convolution neural network. Intell. Cloud Comput., 737–744
Olivier, R., Hanqiang, C., 2012. Nearest neighbor value interpolation. Int. J. Adv. Comput. Sci. Appl 3 (4), 25–30.
Parsania, M.P.S., Virparia, D.P.V., 2016. A comparative analysis of image interpolation algorithms. Int. J. Adv. Res. Comput. Commun. Engineering 5 (1), 29–34.
Razmjooy, N., Mousavi, B.S., Soleymani, F., 2012. A real-time mathematical computer method for potato inspection using machine vision. Comput. Math. Appl. 63 (1), 268–279.
Maddikunta, P.K.R., Parimala, M., Koppu, S., Reddy, T., Chowdhary, C.L., Alazab, M., 2020. An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. Comput. Commun. 160, 139–149.
Siegel, R.L., Miller, K.D., Jemal, A., 2019. Cancer statistics, 2019. CA 69 (1), 7–34.
Simonyan, K., &Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.
Snoek, J., Larochelle, H., Adams, R.P., 2012. Practical bayesian optimization of machine learning algorithms. Adv. Neural Inform. Process. Syst., 2951–2959
Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: Proceedings of the IEEE Conference On Computer Vision And Pattern Recognition, pp. 1–9.
Van De Ville, D., Nachtegael, M., Van der Weken, D., Kerre, E.E., Philips, W., Lemahieu, I., 2003. Noise reduction by fuzzy image filtering. IEEE Trans. Fuzzy Syst. 11 (4), 429–436.
Wang, Q., Yuan, Y., 2014. Learning to resize image. Neurocomputing 131, 357–367.
Wang, Y., Zhang, H., Zhang, G., 2019. cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. Swarm Evol. Comput. 49, 114–123.
Xie, H., Zhang, L., Lim, C.P., 2020. Evolving CNN-LSTM models for time series prediction using enhanced grey wolf optimizer. IEEE Access 8, 161519–161541.
Yoo, Y., 2019. Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. Knowl.-Based Syst. 178, 74–83.