An Empirical Development of Hyper-Tuned CNN Using Spotted Hyena Optimizer For Bio-Medical Image Classification

Patitapaban Rath[1], Pradeep Kumr Mallick[2], Rajesh Siddavatam[3], Gyoo Soo Chae[4]

[1,2]School of Computer Engineering, Kalinga Institute of Industrial technology (KIIT) Deemed to be University, Odisha, Email: pabanrath@gmail.com, pradeep.mallickfcs@kiit.ac.in
[3]Presidency University, Bangalore, Email: rajesh.siddavatam@gmail.com
[4]Div. of Smart IT Engineering, Baekseok University, South Korea, Email: gschae00@gmail.com

Abstract

This paper attempted to obtain the optimized value by tuning the hyper-parameters of Convolutional Neural Network (CNN) by mimicking the biological and collaborative behaviour of Spotted Hyenas. To resolve the complications associated with tuning the hyper-parameters of CNN such as; learning rate, momentum, number of epochs, batch size, kernel size, kernel type, stride, padding, number of nodes in hidden layers, and activation functions a bio-image classification model has been developed empirically by utilizing the explorative strength along with exploitation in the search region using a recently developed meta-heuristic optimization approach inspired by the life of spotted hyenas such as; encircling, hunting, attacking and searching for prey to solve continuous optimization problems by mathematically modeled technique. The suggested method has been assessed by traditional Artificial Neural Network (ANN) based strategies as well as Deep learning network-based strategies. The efficacy of this hyper-tuned CNN classification model has been proven by various accuracy measures, convergence analysis, and statistical implications on two benchmark datasets from Kaggle data repository. From this study, it can be observed and concluded that the suggested nature-inspired method of tuning the hyper-parameters of CNN is an effective and trustworthy algorithm that has ability to classify the brain MRI images in an optimized manner.

Keywords: Brain MRI images, Convolutional Neural Network (CNN), Deep learning network, hyper-parameter tuning, Spotted Hyena.

Overview

A huge amount of image data is readily available for the last many years for medical image analysis and medical decisions. In practice, many computer-aided diagnosis (CAD) systems based on machine learning methodologies are being proposed, developed and widely used to facilitate medical professionals.[1,2] The challenges such as; obtaining the performance of the models in terms of accuracy, reliability, speed, need for more generic image analysis technologies for specific clinical tasks, efficient approach for ground truth generation to obtain the proper validation of those models developed using machine learning methodologies. The bio-medical image classification is one of the important and leading areas of research in image analysis.[3,4] In particular, machine learning and deep learning algorithms[5,6] have contributed to bio-medical image classification research. In short, bio-medical image classification is a process of organizing biomedical image databases into image categories before diagnostics. The selection of a machine learning algorithm to develop such an efficient and accurate model is the challenging issue faced by machine learning

Access this article online

Quick Response Code:

Website:
www.jnsbm.org

DOI:
10.4103/jnsbm.JNSBM_12_3_5

Address for Correspondence: Patitapaban Rath, School of Computer Engineering, Kalinga Institute of Industrial technology (KIIT) Deemed to be University, Odisha
E-mail: pabanrath@gmail.com

Submitted: 29th July, 2021 **Received:** 4th September, 2021
Accepted: 17th September, 2021 **Published:** 27th Oct, 2021

How to cite this article: Rath P, Mallick P K, Rajesh Siddavatam R, Gyoo Soo Chae G S. An empirical development of hyper-tuned cnn using spotted hyena optimizer for bio-medical image classification. J Nat Sc Biol Med 2021;12:300-6.

researchers.[7] Nowadays, CNN-based classification has been widely used in the medical image classification task due to its excellent feature extraction approach. While using CNN, it has been observed that it requires a set of parameters that is external to the data and the tuning of those parameters are done manually.[1,8,9] Those variables of the network structure of CNN are known as hyper-parameters. The key challenge is choosing a set of hyper-parameters in a reasonable time without manual intervention which helps to build an accurate and efficient model to classify the images into its categories. It has been studied that, numerous research has been found in optimizing the hyper-parameters based on many optimizing techniques such as; evolutionary algorithms and swarm-based algorithms. But very less work has been performed in validating the assumptions done for optimizing those hyper-parameters and obtaining a tuned CNN model which is theoretically and practically significant to the medical sector.[10] The objective of this research is to employ the CNN which will not only depend on the chosen architecture but also will be able to automate the selection of hyperparameters such as; learning rate, momentum, number of epochs, batch size, kernel size, kernel type, stride, padding, number of nodes in hidden layers and activation functions easily and quickly to ensure the best performance of the model in some datasets.[11-13] Therefore, CNN has been customized with a nature-inspired Spotted Hyena Optimization (SHO)[14-16] to obtain the optimal values for hyper-parameters and get a tuned CNN for image classification which can other hands automatically and efficiently learn the essential image features that are suitable for the classification task.

In past, the optimization of machine learning algorithms was usually achieved using gradient-descent and finite element tools and techniques, the mathematical calculations involved in those are limited to the calculation of accurate function gradients and are trapped in local optima and the slow convergence rate. Considering these limitations of those traditional optimization approaches, the meta-heuristics have become popular which are computationally inexpensive, flexible, and simple to experiment with. The meta-heuristic and nature-inspired swarm-based optimizations algorithms and searching strategies are best-used optimizers in this field of research.[1-3,6,7] In this work, an attempt has been made to tune the hyper-parameters of CNN using the social and hunting behaviour of the spotted hyenas which in turn is also a new type of meta-heuristic optimizer.

The current work explains the proposed SHO-based training of the CNN algorithm and during the training process the values of hyper-parameters of CNN are properly selected and optimal values of those parameters are obtained. The SHO uses the hyper-parameters of CNN in terms of vectors and assigns the best value for those above-mentioned hyper-parameters from the assigned lower and upper bound values of CNN parameters using mathematical modeling of the phases of SHO such as; encircling, hunting, attacking, and searching of prey. The main motivation and objective of choosing SHO to hybridize with CNN classifier can be noted as; the high exploration and exploitation approach which have shown the wide use of this meta-heuristic algorithm and it has been observed that it shows its good performance to attain the global optimum which has not been the case of other meta-heuristic approaches.[15-20]
A large number of studies on biomedical image classification for the design of CAD-based systems have been studied. This section deals with a detailed study on the classification of biomedical images such as MRI images.

The accurate performance and ability for data analysis have made machine learning algorithms a widely used approach for any type of data analytic problem. But it must be understood that designing the data analytic models using machine learning techniques is not that easy because it needs an effective selection of models as well as the selection of optimal values for the parameters of those models. Manual selection of parameters is now obsolete and there is a need to automate this learning process of the classifier which can automatically choose the best or optimal values.[21] This section discusses the recent work done on this hyper-parameter optimization. There are basically, two categories of parameters are set for any machine learning model for example; weights of neurons in ANN which are initialized and updated during the learning process those are called model parameters, and the other parameters such as; learning rate to train the ANN is known as hyper-parameters.[11-13] Recently, a hyper-parameter tuned model has been suggested by Kumar et al.[22] and authors have used the random forest and Bayesian classifier for experimentation on breast cancer dataset and they have achieved 3.7146 %, 5.27398 %, and 4.4413 % improved performance over the few existing methodologies such as; BPNN, FW + SSA-SVM, FW + GA-SVM, etc. Pan et al.[11] proposed a multispectral method based on CNN to find the optimal values of hyper-parameters of CNN. Authors have tested this model with Titan data and they have compared the proposed method with few classical CNN models such as; AlexNet, VGG16, and ResNet50 too. A gradient descent method has been proposed by Jiang et al.[23] to optimize and regularize the hyper-parameters of SVM using a bi-level of an optimization model. The authors have validated their model with few existing gradient descent methods and showed that the proposed strategy for SVM is achieving better accuracy in terms of prediction.

Similarly, Gul et al.[24] tried to propose a method to tune the hyper-parameters of SVM for providing accurate spillway type selection by utilizing the random forest method and they have achieved 93.81% accuracy comparing with few existing methods. A model for selecting hyper-parameters of CNN classifier based on a variant of PSO named as cPSO-CNN has been authored by Wang et al.[13] by adopting the ranking of PSO particles to reduce the cost function. In this work, the authors tried to optimize the basic hyper-parameters and also tried to optimize the pooling method, stride and padding along with kernel number of pooling layer and fully-connection layer parameters respectively. Another hyper-parameter tuning based on deep neural network has been attempted by Yoo[25]. The proposed method has been tested on auto-encoder and CNN for The Modified National Institute of Standards and Technology (MNIST) dataset. The proposed strategy achieved fast convergence in fewer computational resources. Few more hyper-parameters optimization has been for machine learning models. To cite them, Cui et al.[26] used Gaussian process-based Bayesian optimization, Aszemi et al.[27] used genetic algorithm, Singh et al.[28] proposed a Multi-level PSO algorithm to optimize the hyper-parameters of CNN.

Motivations

a. Bio-medical images, like MRIs or CT scans, contain a huge volume of rich data about the underlying tissue structure. Interpretation and analysis of these voluminous data are required to get a proper diagnosis and accurate evaluation of the extent of diseases.
b. This often requires machine learning techniques, like classification, to build automated expert systems that ensure a reliable and accurate diagnosis of diseases and thereby are of great help to the radiologist and medical experts.
c. Though there are many strategies developed for image analysis, still, the recent demand for biomedical image analysis has been one of the contributing factors.
d. Biomedical images such as; MRI, PET, CT scan, and Ultrasound images are being produced and are used for the detection of tumors or any kind of abnormalities in the human body which are clinically used and analyzed by radiologists. CAD-based systems are being in demand and contributing to the development of new methodologies to provide accurate diagnostic analysis.
e. There are many supervised and unsupervised methodologies applied to the design and development of CAD-based systems for medical image analysis with respect to segmentation and classification, where machine learning techniques are being widely utilized. In this study, an attempt has been made to explore the capabilities of few traditional ANN-based strategies along with deep learning-based methodologies. The CNN for image classification has been extensively used and experimented with.
f. In the case of CNN, choosing the best values for the hyper-parameters as discussed in the previous section is the focussed challenge among the researchers to overcome the problem of choosing or setting the values of those parameters manually.
g. Very little research has been done to validate the hyper-parameter values of CNN, and now many meta-heuristic optimization strategies have been gaining attention to obtain the optimal value of such hyper-parameters. However, this motivated to hybridize the SHO to automatically tune those parameters of CNN.

Original Contributions

In this work, a pedagogical approach combining the machine learning techniques along with meta-heuristic optimizer for the development of a novel bio-medical image classification model has been conceptualized to get a big picture of challenges and issues associated with the design of such CAD-based systems for image classification and validation of proposed approach. The key contributions of this study are;

a. Collection of brain image datasets from Kaggle repository[29,30] to be used as input to the classifiers.
b. The hyper-tuned learning mechanism based on SHO and CNN has been explored to classify the brain MRI datasets, as it uses a pre-trained model to overcome the issues associated with huge datasets. The hyper-parameters of CNN have been tuned by SHO for both datasets.
c. A straightforward comparison has been made on traditional ANN models, deep neural networks (DNN), and hyper-tuned CNN learning-based image classification.
d. The performances of those models are recorded and improvements over each have been detailed.
e. The validation of the proposed hyper-tuned CNN model has been done based on accuracy, misclassification rate, precision, recall, specificity, F1-score, G-mean, etc.[31], and also the performance recognitions are recorded based on error-graphs during both training and testing phases. One of the most important evaluation metric receiver operating curves (ROC)[31] has also been obtained to check and confirm the performance ability of the hyper-tuned CNN image classification model.
f. The key observations with respect to experimentation have been discussed in detail.

This paper has been organized as; Section 2 discusses the methodologies adopted for experimentation. Section 3 gives a detail discussion of the broad scope of this research with the layout of experimentation and validation. Section 4 explains the research metric and empirics used for experimentation with datasets used, parameters setup, and result analysis. The key observations of experimentation are given in Section 5. The paper has been concluded in Section 6 with few future scopes of this work.

Preliminaries

In this section, various classifiers such as MLP, ELM, DNN and CNN from traditional ANN-based classifiers and deep learning-based classifiers used for experimentation and comparison are discussed in detail. The SHO has also been detailed in this section along with its working principle.

Traditional ANNs: MLP and ELM Classifiers

ANN is a part of machine learning that attracts the interest of researchers over decades.[1] The neural networks try to mimic the behaviour of the human brain to perform different machine learning tasks. The ANN consists of hundreds of artificial neurons (processing units) which are considered its main building block. Each neuron is made up of input and output units, where the neuron receives the inputs through the input units, and then it performs a kind of linear or nonlinear operation over it. Finally, forward the results to the next unit through its output units. There are many variants of ANN, here we have used MLP and ELM for experimentation and validation purpose which are discussed below.[2-7]

MLP Classifier

MLP is one of the simplest variants of ANN. This network consists of a single layer with the output layer and the target output. MLP can approximate any nonlinear function and it's widely used in many applications. Anyway, sometimes MLP fails to provide a good solution because of the inappropriate selection of the network architecture, the inappropriate initialization of the interconnections' weights, or the inappropriate selection of the activation functions.[32,33]

ELM Classifier

Another family of ANN proposed by Huang, et al in 2004 is called ELM[34,35] and it can be visualized as a single layer feed-forward neural network where the connections' weights between the input layer and the single hidden layer are randomly initiated. On the other hand, the connections' weights between the hidden layer and the output layer are calculated with the help of Moore–Penrose generalized inverse. In contrast to FLANN, the ELM network doesn't require pre-calculation of a higher-order combination of the input data, it doesn't use gradient descent to be trained, and it uses activation function at the hidden nodes. In general, ELM networks require less computational power, and it is less time-consuming which makes them a better alternative to traditional neural networks that use the gradient descent method to train. ELM networks can approximate any continuous non-linear function f(x) by satisfying the condition $\left|\lim_{L\to\infty}\left\|\sum_{i=1}^{L}\beta h_i(x)-f(x)\right\|=0.\right.$ Anyway, the universal approximation capability of ELM has been proven by various researchers. The choice of the activation function can highly affect the general performance of the ELM network, so that, it needs to be selected carefully and according to the nature of the problem under process. The final output of the ELM network can be calculated according to the following Eq. (1).

$$\tilde{Y} = H\,.\,\beta \tag{1}$$

Where, $H=\{H_1,H_2,\ldots H_k\}$ is the output of the hidden layer and β is the weights matrix of the hidden-output connections. Like any other ANN, the goal of ELM is to reduce the mean square error between the calculated and actual output. Another goal of ELM is to minimize the l_2 norm of β matrix. The reason behind that is to enhance the generalization performance of the ELM network. The updated weights' matrix β can be calculated by the following Eq. (2).

$$\beta = H^{\dagger}Y \tag{2}$$

Where, $H^{\dagger}$ is known as the Moore–Penrose generalized inverse of the hidden layer output. Despite the extraordinary performance of ELM and its variations, it is unable to deal efficiently with problems of very high complexity such as image classification and object recognition, and another limitation of ELM is the calculation of the inverse of the hidden output matrix which is considered computationally expensive, when the number of hidden nodes exceeds a specific limit.

Deep Networks: DNN and CNN Classifiers

Deep networks are another improved version of ANNs that can model complex non-linear relationships by adding multiple numbers of hidden layers in the ANN network and employs math modeling to increase the accuracy of the classifiers. Deep networks capitalize the ANN component to improve the capability of a model by adding stacks of hidden layers to grade the importance of the inputs to determine the output and get the benefit of adding more layers. This addition of multiple layers makes it deep; therefore it is also known as the deep net. There are many variants of the deep networks are there for classification and prediction purposes, out of them here we have used DNN and CNN which are discussed below.

DNN Classifier

Different variations of neural networks proposed over years, and maybe the simplest and more powerful idea is DNN. Simply, DNN is a neural network of at least two layers. The number of nodes at each layer and the maximum number of layers is bounded only with the computational power of the device running these networks. The deep structure of DNN allows it to learn deeper for hard problems. It is also able to adapt to any kind of problem and solve highly complicated problems. DNN networks start with a single input layer followed by a series of hidden layers. The last layer is the output layer where the expected output is obtained. At each hidden node, the input data are passed through an activation function which performs a non-linear transformation over this data. Let a^l to be the output of layer. Then, the output of the next layer could be calculated as follows using Eq. (3) and Eq. (4).

$$a^{l+1} = \sigma\left(\textstyle\sum_i z_{ij} + b_j^{l+1}\right) \quad (3)$$

$$z_{ij} = a_i^l w_{ij}^{(l,l+1)} \quad (4)$$

Where z_ij is the input of the next layer, b is the bias, $w^{(l,l+1)}$ is the weight connection between the layers l and l=1. In the case of multi-class classification, the output of the last layer can be computed by applying softmax on the last layer output. This family of networks has a lot of variations, each one of them used for a specific range of problems. The Restricted Boltzmann Machine, Deep Belief Network, Auto-encoder, Deep CNN are some variations of DNN networks. The most common learning algorithm used to train this kind of network is the back-propagation algorithm with the help of the gradient descent algorithm. The training process of DNN is not a trivial task and it requires a lot of focus and attention in order to avoid over-fitting. On the other hand, the architecture of the DNN along with the used activation functions needs to be selected carefully by the network designer in order to be able to solve the problem perfectly. Furthermore, the initial weight value of the interconnections in the network can highly affect the convergence speed of the network. It seems that DNN is a double-edged sword, according to the way it was developed. Anyway, a lot of optimization algorithms are proposed to tune the hyperparameters of the DNN including natural-inspired algorithms, Bayesian optimizations, and many others.[36-38]

CNN Classifier

Another famous variation of the ANN is CNN. These kinds of networks have the same principles of DNN, their differences concentrate on the structure of the network layer, the series of actions applied on the input data to generate the output and their application area. The CNN was inspired by the organization of the visual cortex and its connectivity pattern is more similar to the neurons in the human brain.CNN networks are mainly used for image classification, object recognition, and rarely for audio recognition. The spatial and temporal dependencies in the input image can easily be captured by CNN with the help of its special layers. A general structure of CNN is depicted in Figure 1.

The architecture of this classifier consists of an input layer, a series of convolution layers, and each followed by a pooling layer. Each convolution layer consists of a set of square filters or kernels, where each filter slides over all the pixels of the input image, which will result in a feature map. When the filters are applied, the amount of filter slide over the input image can be controlled to better reflect the overlapping of the pixels with their neighbours.[9,11,13] Rectified Linear Unit (ReLu) is one of the popular and widely used activation functions used in the network. In general, the CNN network takes an image as input, the input image pass through a set of convolution layers and pooling layer.

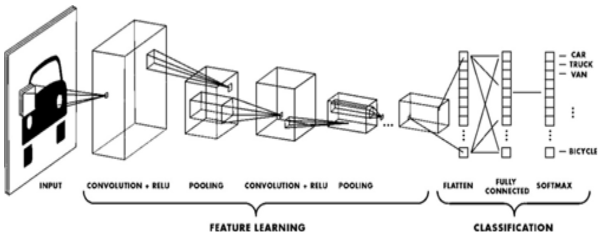

Figure 1: CNN structure[9]

The output of each convolution layer is a feature map. The output of the pooling layer is a reduced size output of the input data. The final hidden layer is flattened and fed to the fully connected layer which will provide the final results. The CNN needs a large number of images to be trained. The most common learning algorithm is the back-propagation algorithm with the help of gradient descent. Many popular architectures of CNN networks received huge popularity in the last decades and used are extensively in the industrial and research fields, we could mention LeNet, AlexNet, VGG-16, VGG-19 and many others.

Spotted Hyena Optimizer (SHO)

Recently several optimized techniques have been used for more efficient optimization results. SHO is based on the hunting nature of spotted hyenas. This optimizer presents a lot of advantages over other optimizers. It is very simple to implement because of its simple algorithmic nature. The working principle is based on four basic steps, such as; encircling prey, hunting prey, prey attacking and pray search. This algorithm can be applied to solve various real-life engineering problems to find optimal results. In real life, the spotted hyenas live in a group having more than 100 members and they generally hunt in groups. There are different types of hyenas and spotted hyenas are proficient and massive hunter species than other species. The working principle of SHO is based on the following four steps.[14-20]

a. Encircling prey: The mathematical model of this behaviour is represented by the Eq. (5) and Eq. (6).

$$D_h = B \cdot Pp(x) - P(x) \quad (5)$$

$$P\,(x\,+\,1)\;=\;Pp\,(x\,)\;-\;E\cdot D_h \qquad (6)$$

Where, D_h = distance between preys and spotted hyenas, x=current iteration, B and E = Coeeficeint vectors and Pp = position of the prey.

The vectors B and E are calculated using Eq. (7), Eq. (8) and Eq. (9).

$$B\,=\,2.\,rd_1 \qquad (7)$$

$$E\,=\,2.\,h.\,rd_2-h \qquad (8)$$

$$h=5-\left(Iteration\,\times\left(\frac{5}{Max\;Iteation}\right)\right) \qquad (9)$$

$Where\;Iteration=1,2,3,\ldots,Max\;Iteration$

h is linearly decreased from 0 to 5 over *Max Iteration* which promotes more exploitation as the iteration increases. However, rd_1 and rd_2 are random vectors in [0, 1]. By using Eq. (5) and Eq. (6), a spotted hyena can update its position randomly.

b. Hunting

The following Eq. (10), Eq. (11) and Eq. (12) proposed in this mechanism are used for hunting the prey and creating a cluster of optimal solutions.

$$D_h=B.P_h-P_k \qquad (10)$$

$$P_k=Ph-E.D_h \qquad (11)$$

$$C_h=P_k+P_{(k+1)}+\cdots+P_{(k+N)} \qquad (12)$$

Where P_h *= position of spotted best spotted hyena,* P_k *= position of rest other spotted hyenas and N = no.of spotted hyenas.* This N can be computed as follows using Eq. (13).

$$N=\text{Count Nos.}\,(P_{h},\,P_{(h+1)},\,P_{(h+2)},\,\ldots,\,(P_{(h+M)})) \qquad (13)$$

Where *M* is a random vector in [0.5,1], nos defines the number of solutions and count all candidate solutions, after addition with *M*, which are far similar to the best optimal solution in a given search space, and C_h is a group or cluster of *N* number of optimal solutions.

c. Attacking Prey

In order to mathematically model for attacking the prey, we decrease the value of vector h. The variation in vector E is also decreased to change the value in vector h which can decrease from 5 to 0 over the course of iterations. The mathematical formulation for attacking the prey is done using Eq. (14).

$$P(x+1)=\frac{C_h}{N} \qquad (14)$$

Where, $P_{(x+1)}$ save the best solution and updates the positions of other search agents according to the position of the best search agent. The SHO algorithm allows its search agents to update their position and attack towards the prey.

d. Searching of Prey

This step is also named as exploration step. However, the searching process gives idea about the exploration ability of an algorithm. The SHO algorithm confirms the ability of applying random values which are < 1 or > 1. This algorithm has a vast application area and can be used to solve several real-world optimization complexities by utilizing the goodness of this algorithm. This algorithm takes low computational cost for better results than other meta-heuristics algorithms.

Broad Overview of the Proposed Bio-Medical Image Classification Model

In correspondence with this research, main objective is to empirically develop and validate an automated expert system and ensure the reliable and accurate bio-medical image classification of brain MRI datasets. However, even though there are many strategies developed for image analysis, one of the contributing factors to the development of new methods is the growing demand for accurate diagnostic analysis. In this study, the CNN has been extensively implemented and studied to develop an optimized bio-medical image classification model by tuning the hyper-parameters of CNN such as; learning rate, momentum, number of epochs, batch size, kernel size, kernel type, stride, padding, number of hidden layers and activation functions by exploring the social and behavioural activities of a meta-heuristic SHO algorithm.

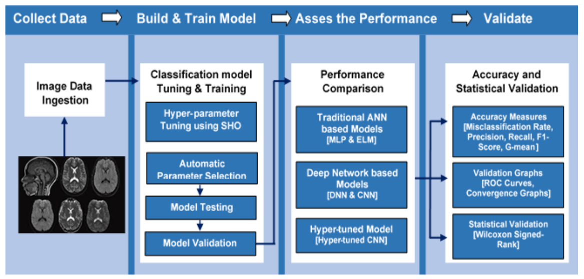
Collect Data
Build & Train Model
Asses the Performance
Validate
Image Data Ingestion
Classification model Tuning & Training
Hyper-parameter Tuning using SHO
Automatic Parameter Selection
Model Testing
Model Validation
Performance Comparison
Traditional ANN based Models [MLP & ELM]
Deep Network based Models [DNN & CNN]
Hyper-tuned Model [Hyper-tuned CNN]
Accuracy and Statistical Validation
Accuracy Measures [Misclassification Rate, Precision, Recall, F1-Score, G-mean]
Validation Graphs [ROC Curves, Convergence Graphs]
Statistical Validation [Wilcoxon Signed-Rank]

Figure 2: Overall design of the proposed bio-medical image classification model.

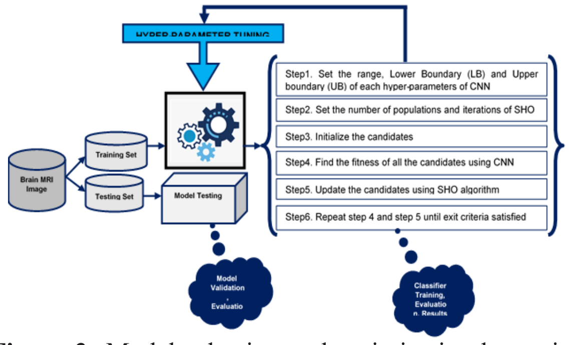
HYPER-PARAMETER TUNING
Step1. Set the range, Lower Boundary (LB) and Upper boundary (UB) of each hyper-parameters of CNN
Step2. Set the number of populations and iterations of SHO
Step3. Initialize the candidates
Step4. Find the fitness of all the candidates using CNN
Step5. Update the candidates using SHO algorithm
Step6. Repeat step 4 and step 5 until exit criteria satisfied
Brain MRI Image
Training Set
Testing Set
Model Testing
Model Validation , Evaluatio
Classifier Training, Evaluatio n. Results

Figure 3: Model selection and optimization by tuning hyper-parameters of CNN by hybridizing with SHO.

Moreover, the main importance has been given to hybridize the CNN with SHO to obtain the tuned hyper-parameters of CNN. The proposed study revolves around four stages of implementation such as; collection of data; build and train the model, asses the performance of the tuned CNN with various learning strategies such as MLP, ELM, DNN and basic CNN and finally validating the proposed method through various accuracy measures such as; misclassification, rate, precision, recall, F1-score, G-mean and also the improvement of hyper-tuned CNN oven MLP, ELM, DNN and basic CNN has been recorded. Various error-graphs, ROC and convergence curve graphs have been drawn to establish the efficacy of the proposed model. Additionally, the statistical validation measures have been used to validate the proposed hyper-tuned CNN for bio-medical image classification. The overall design of this proposed strategy has been provided in Figure 2.

The key issue lying with the design of any machine learning model is to better understand the architecture of your learning algorithm, its inputs, the parameters, and the output. While designing those models, the inputs and outputs may be known to the researcher but choosing the optimal values for those free parameters is not known and has to be explored empirically with a range of possible values. In a true sense, while designing any such model, ideally the algorithm must be able to perform this exploration of selecting and obtaining the optimal model parameters automatically. Those parameters, which define the architecture of any machine learning model are known as hyper-parameters and the model which is able to search for the optimal values for those parameters is known as hyper-parameter tuning. Those hyper-parameters are not the model parameters and cannot be directly trained from the data. Therefore, optimization methodologies are adopted to obtain the eligible parameter values.

Mathematically, this tuning of hyper-parameters can be defined as follows;

Let the classification model be defined as Ω, *Datasets→CM*, Where *CM* is the classification model.

$\Omega \in \vartheta$ is the obtained hyper-parameter configuration of CNN where; *learning rate, momentum, number of epochs, batch size, kernel size, kernel type, stride, padding, number of hidden layers* are the parameters to be considered.

The proposed hyper-tuned CNN algorithm estimates a classification model CM_{Ω} that minimizes a regularized loss function or misclassification rate ℵ.

$$\Omega(\mathrm{Datasets}_{\mathrm{Training}}) := \underset{CM_{\Omega}\in CM}{\arg\min} \aleph(CM_{\Omega}, \mathrm{Datasets}_{\mathrm{Training}}) + \Re(CM_{\Omega}$$

Now, the next step is to validate the obtained hyper-parameters (Ω) using $\mathrm{Datasets}_{\mathrm{Testing}}$, which can be done using the following Eq. (15).

$$\widehat{\Omega} := \underset{\Omega\in\vartheta}{\arg\min} \aleph \frac{(CM_{\Omega}, \mathrm{Datasets}_{\mathrm{Training}}), \mathrm{Datasets}_{\mathrm{Testing}}}{=:\mathrm{Misclassification\ Rate}(\Omega)} \quad (15)$$

Note: The values of Ω has been obtained utilizing the steps of SHO such as; encircling the prey, hunting, attacking and searching for the prey as given in Algorithm 1. The layout and overall processing of hyper-parameter tuning based on SHO have been shown in Figure 3 and Figure 4 respectively.

Algorithm 1: Proposed hyper-parameter tuning of CNN using SHO[14-20]	
Input:	Number of Spotted Hyenas (SHs), SH_i (i = 1, 2, ..., n)
Output:	Best SH
Description:	Each SH is a combination of CNN models hyper-parameters such as; (Learning rate, Momentum, Batch size, Number of nodes in each Convolution Layers, Kernel Size, Activation functions, Filters Size, Dropout and number of epochs)
1	Initialize population of n SHs randomly
2	Calculate the fitness value of each search agent
3	While (Epoch < max number of iterations)
4	For each SH
5	Update the position of each search agent according to Eq. (14)
6	End for
7	Update the control parameters E, B, D_h and h
8	Compute the fitness value of each SH
9	Update SH_h if it is better than the previous solution
10	Update the cluster C_h w.r.t. SH_h
11	Epoch = Epoch + 1
12	End while

Research Metrics and Empirics

This section discusses the system configuration, the various datasets used, the parameters chosen for FLANN, MLP, ELM, DNN, and CNN for experimentation. The selected range of parameters after tuning the CNN for both datasets has also been given.

System Configuration

This section describes the experiment carried out using the combination of MLP, ELM, DNN, CNN, hyper-tuned CNN based on SHO to handle the image classification model. Searching for the best combination of hyper-parameters requires computational resources. The experimental evaluation is carried out in the Google Colab environment, under Windows7 with 64 bit and 4GB RAM of Intel i3. Google Colaboratory or Colab requires zero configurations with free access to GPUs to write and execute Python in the researcher's browser which also provides an easy sharing facility. Additionally, this Colab harnesses the full power of popular libraries to analyse and visualize the data.

Datasets used for Experimentation

In this study, the datasets belonging to brain MRI are considered for experimentation and comparison and are collected from Kaggle Repository.[29,30] The first dataset

Glioma[29], that has been published and owned by Bhuvaji et al.[29] is Brain MRI images[30] and the second dataset is named as Brain MRI Images for Brain Tumor Detection had been published and maintained by Chakrabarty[30], a PG Diploma student at IIIT Bangalore. The detailed description of those two datasets is given in Table 1.

Table 1. Description of datasets

Dataset	Total Data size	Classes and class labels	No. of samples used for training	No. of samples used for testing
Glioma[29]	3264	Gliomatumour-01	826	100
		Pituitary-02	827	74
		Meningioma-03	822	115
		No tumour-04	395	105
Brain MRI Images[30]	253	No tumour-0	68	30
		Tumour- 1	108	47

Parameters Selection for MLP, ELM, DNN and basic CNN

As detailed in Table 2, the number of input nodes chosen for MLP and ELM is 2250. The number of nodes for the hidden layers for MLP utilized the Keras API therefore; there is no need of choosing the number of hidden layers, number nodes, and activation function. The ELM architectures use one hidden layer with 16384 nodes. The number of nodes in the output layer chosen for MLP and ELM is four. The ReLu activation function has been used for both input and hidden layers of ELM whereas; Softmax has been used for the output layer of ELM.

[Note: NN- No. of nodes, IL-Input Layer, HL- Hidden Layer, OL-Output Layer, AF-Activation Function, FC-Fully connected, KS-Kernel Size, IS-Input Shape, PS-Pool Size, LR-Learning Rate, Mom- Momentum, BS-Batch Size, FS-Filter Size, are the notations used in Table 2, Table 3, Table 4, Table 5, Table 6 and Table 7].

Table 2. Parameters for MLP and ELM

Layers	MLP	ELM
IL	NN=22500	NN=22500, AF=ReLu
HL	All the parameters are defined in Keras API	HL=1, No. of nodes=16384, AF=ReLu
OL	NN=4	NN=4, AF=Softmax

The various parameters chosen for DNN are listed in Table 3. The number of nodes in input layer and output layers have been chosen as (150,150, 1) and 4 respectively. Five hidden layers have been chosen and the number of hidden layers is 128 and the activation function used for input and hidden layers is ReLu whereas; Softmax has been used for output layer.

Table 3. Parameters for DNN

Layers	Parameter Values
IL	NN=(150,150,1), FC, AF=ReLu
HL 1	NN=128, FC, AF=ReLu
HL 2	NN=128, FC, AF=ReLu
HL 3	NN=128, FC, AF=ReLu
HL 4	NN=128, FC, AF=ReLu
HL 5	NN=128, FC, AF=ReLu
OL	NN=4, FC, AF=Softmax

Similarly, the parameters chosen for basic CNN are given in Table 4. The number of filters considered in convolution layer 1 is 64 whereas; 128 numbers of filters have been taken for convolution layer 2 to 5. The ReLu activation function has been used for convolution layer 1 to 5 along with max pool where dense is 1024, the pool size has been taken as (2, 2) for all the layers. The kernel size has been taken as (5, 5) for convolution layer 1 and (3, 3) for convolution layer 2 to 4, whereas; (2,2) for convolution layer 5. The input shape of the convolution layer 1 is (150, 150, 1) which is same as the DNN. The dropout values have been chosen as 0.25 for maxpool of convolution layer 1 and 2, it is chosen as 03 for the maxpool of convolution layer 3 to 5. The various parameters of the optimizer are considered to be Adam, lr=0.001, beta_1-0.9, beta_2=0.999 with 50 epochs and a batch size of 40.

Table 4. Parameters for CNN

Model	Sequential
CL 1	FS=64, KS=(5,5) Padding= same, AF=ReLu, IS=(150,150,1)
Max pool	PS=(2,2), dropout= 0.25
CL 2	FS=128, KS =(3,3) Padding= same, AF=ReLu
Max pool	PS=(2,2), dropout= 0.25
CL 3	FS=128, KS =(3,3) Padding= same, AF=ReLu
Max pool	PS =(2,2), dropout= 0.3
CL 4	FS=128, KS =(3,3) Padding= same, AF=ReLu
Max pool	PS =(2,2), dropout= 0.3
CL 5	FS=256, KS =(2,2) Padding= same, AF=ReLu
Max pool	PS =(2,2), dropout= 0.3 Dense=1024, AF=ReLu, dropout=0.5 Dense=4, AF=Softmax
Optimizer	Adam, lr=0.001, beta_1=0.9, beta_2=0.999
Epoch, Batch size	Epoch=50, batch size=40

CNN and Tuning Hyper-parameters of CNN based on SHO

In CNN, there are nine hyper-parameters that needs to be optimized to obtain permissible values for those nine parameters from a set of lower and upper bound values assigned to the CNN before starting of the training process as given in Table 5. It has been found that optimizing hyper-parameters in CNN is really a tiresome task for many of the researchers who are working in this domain. The manual selection of hyper-parameter values is cumbersome, therefore, an attempt has been made to empirically configure and obtain the values for the set of hyper-parameters with different combinations of values using SHO. In this work, the search for the optimal values for those hyper-parameters is done on both Glioma and Brain MRI images datasets. The hyper-parameters of CNN are selected as learning rate within the range of [0.001 to 0.1], momentum as [0.9, 0.95, 0.99], batch size has been considered to be within [32, 64, 128, 256]. The nodes in convolution layers are within [5-516], kernel size with [(2,2), (3,3), (5,5), (6,6)], ReLu, Sigmoid, LReLu are considered for activation function, the filter size is [64,128,192,256]. The dropout has been set to [0.1,0.3,0.5,0.9] and the number of epochs are experimented for [50,100,200,500,700,1000] numbers.

Table 5. Hyper parameters of CNN to be tuned

Hyper Parameters	Range
LR	Min. value-0.001, Max. value- 0.1
Mom.	[0.9, 0.95, 0.99]
BS	[32, 64, 128, 256]
NN in CLs	Min. value-5, Max. value- 516
KS	[(2,2), (3,3), (5,5), (6,6)]
AFs	[ReLu, Sigmoid, LReLu]
FS	[64,128,192,256]
Dropout	[0.1,0.3,0.5,0.9]
No. of epochs	[50,100,200,500,700,1000]

This searching strategy has been mathematically explained below for better understanding by the readers with an example.

Example: Parameter set-up for SHO;

a. Let Max Iteration: 2 and No. of candidates *(N)*: 5

Epoch-1:

Candidate-1 [0.001,0.9,32,5,(2,2),1,64,0.1,50]
Candidate-2 [0.005,0.95,64,500,(3,3),2,128,0.3,100]
Candidate-3 [0.05,0.95,64,516,(5,5),3,256,0.9,500]
Candidate-4 [0.1,0.99,128,1000,(3,3),2,128,0.5,200]
Candidate-5 [0.07,0.9,256,300,(6,6),1,128,0.3,700]

b. Input the dataset to each candidate and the error rate/fitness values are computed as;

Candidate-1: 0.5624
Candidate-2: 0.8622
Candidate-3: 0.7654
Candidate-4: 0.9532
Candidate-5: 0.4433 [Note: (Best Candidate i.e. P_h)]

c. Updated parameters of SHO;

Encircling:

$h = 5 - (Iteration \times (5 / Max\ Iteration))$
$h = 5 - (1 \times (5/2)) = 3$

$B = 2 \cdot rd_1$
$B = 2 \times 0.3 = 0.6$ *(suppose random numbers* $rd_1 = 0.3$ *and* $rd_2 = 0.2$)
$E = 2\,h \cdot rd_2 - h$
$E = 2 \times 3 \times 0.2 - 3 = -1.8$

Hunting:

$D_h = B{\cdot}P_h - P_k$ *and* $P_k = P_h - E \cdot D_h$. Where P_h defines the position of first best spotted hyena, P_k indicates the position of other spotted hyenas.

For -1 :

$D_h = B{\cdot}P_h - P_k$ *can be computed as*
$D_h = 0.6 \times$ [0.07, 0.9, 256, 300, (6, 6), 1, 128, 0.3, 700] - [0.001, 0.9, 32, 5, (2, 2), 1, 64, 0.1, 50]

$D_h =$ [0.042 0.54 153.6 180 (3.6, 3.6) 0.6 76.8 0.18 420] - [0.001, 0.9, 32, 5, (2, 2), 1, 64, 0.1, 50] = [0.041 -0.36 121.6 175 (1.6, 1.6) -0.4 12.8 0.08 370]

And $P_k = P_h - E{\cdot}D_h$ *can be computed as;*

$P_k =$ [0.07, 0.9, 256, 300, (6, 6), 1, 128, 0.3, 700] - (-1.8)× [0.041 -0.36 121.6 175 (1.6, 1.6) -0.4 12.8 0.08 370]

$P_k =$ [0.07, 0.9, 256, 300, (6, 6), 1, 128, 0.3, 700] - [-0.0738 0.648 -218.88 -315 (-2.88, -2.88) 0.72 -23.04 -0.144 -666]

$P_k =$ [0.143 0.252 474.8 615 (8.88, 8.88) 0.28 151.04 0.444 1366]

Similarly, compute P_k for rest of the candidates
$C_h = P_k + P_{(k+1)} + \cdots + P_{(k+N)}$

$C_h =$ [0.143 0.252 474.8 615 (8.88, 8.88) 0.28 151.04 0.444 1366] + P_k *of Candiate 2* + . . . + P_k *of Candiate* 5

Attacking prey (Exploitation):

$P(x+1) = C_h / N$, Suppose, $P(x+1) =$ [0.243 0.352 574.8 715 (9,9) 0.4 155 0.6 1400]

Where, $P(x+1)$ is the position of the prey.

New Position of Candidates can be computed as;

$P_{p(x)} = P(x+1) =$ [0.243 0.352 574.8 715 (9,9)0.4 155 0.6 1400]

$D_h = B \cdot P_{p(x)} - P\,(x)$

$D_{h\ for\ Candiadte\ 1} = B.P_{Prey\ position} - P_{(Candidate\ 1)}$

$D_{\text{h for Candiadte 1}} = 0.6 \times$ [0.243 0.352 574.8 715 (9,9)0.4 155 0.6 1400]- [0.001, 0.9, 32, 5, (2, 2), 1, 64, 0.1, 50]

$P_{(\text{Candidate 1})} =$ [0.243 0.352 574.8 715 (9, 9) 0.4 155 0.6 1400] - (-1.8) × D_h

Now, bringing the values within boundaries, for example for P_((Candidate 1)) , the first parameter value 0.243

which is the value obtained for learning rate, now as this obtained is more than the max value (0.1), now it needs to lower down to 0.1 instead of 0.243. Similarly, other obtained parameter values are to be mapped to the range of permissible values as mentioned in t Table 5. During the investigation through experimentation, the set of values obtained for the Glioma and Brain MRI images datasets are detailed in Table 6 and Table 7 respectively. Those values are obtained with respect to the set of permissible values assigned for CNN for experimentation with respect to Table 5.

Table 6. Hyper parameters of CNN after tuning for Glioma dataset

Hyper Parameters	Range
LR	0.009
Mom	0.9
BS	32
No. of nodes in CL 2	256
No. of nodes in CL 3	256
No. of nodes in CL 4	256
No. of nodes in CL 5	128
KS of IL	(5,5)
KS of CL 2	(5,5)
KS of CL 3	(5,5)
KS of CL 4	(3,3)
KS of CL 5	(3,3)
AFs	ReLu
FS of IL	128
FS of CL 2	128
FS of CL 3	64
FS of CL 4	64
FS of CL 5	128
A number of epochs	500

Table 7. Hyper parameters of CNN after tuning for Brain MRI dataset

Hyper Parameters	Range
LR	0.003
Mom	0.9
BS	32
No. of nodes in CL 2	256
No. of nodes in CL 3	256
No. of nodes in CL 4	128
No. of nodes in CL 5	128
KS of IL	(5,5)
KS of CL 2	(5,5)
KS of CL 3	(3,3)
KS of CL 4	(3,3)
KS of CL 5	(2,2)
AFs	ReLu
FS of IL	128
FS of CL 2	128
FS of CL 3	128
FS of CL 4	64
FS of CL 5	64
A number of epochs	200

Here, to establish the performance of hyper-tuned CNN based on SHO, the ROC curve has been drawn for both of the datasets considering the false positive and true positive rates at different thresholds of the proposed hyper-tuned CNN classifier. Figure 4 establishes the performance of hyper-tuned CNN for both the datasets used for experimentation in this work with respect to the error-graph. From this graph, it can be inferred that the error-rates of the Brain MRI images dataset is lowering down significantly and becoming zero before 100 the iteration and for the Glioma dataset, the error-rate is lowering down and reaching to zero before the 300 the iteration while classification is done using proposed SHO based hyper-tuned CNN classifier. It can be seen from Figure 5 that, both false positive and true positive rates are increasing giving rise to a two-dimensional area underneath the entire ROC curve with maximum Area Under Curve (AUC) value providing an aggregate measure of performance across all possible classification thresholds.

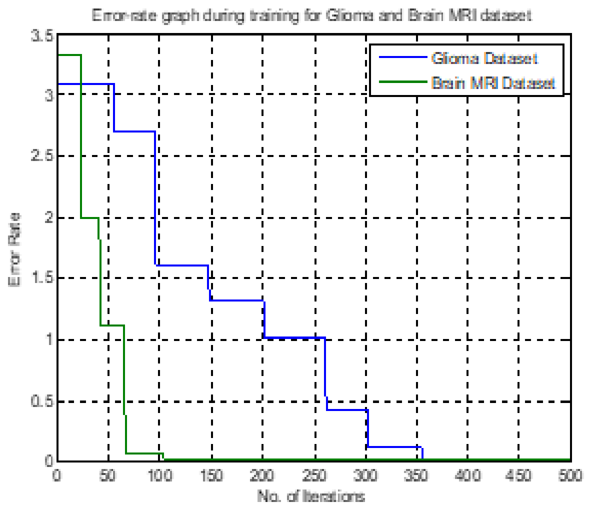

Figure 4: Error-graphs showing the effectiveness of hyper-tuned CNN using SHO for Glioma and Brain MRI images datasets

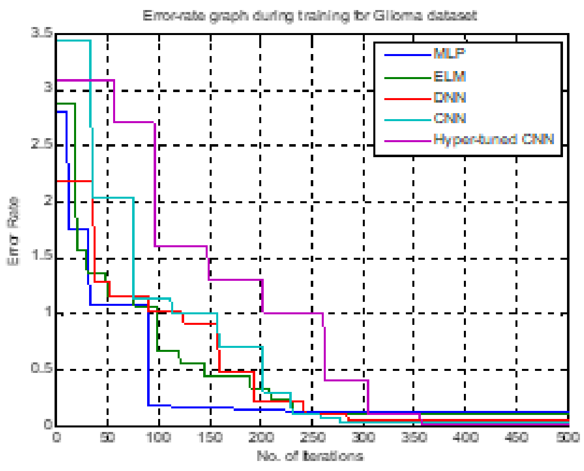

Figure 5: ROC curves showing the effectiveness of hyper-tuned CNN using SHO for Glioma and Brain MRI images datasets

Model Comparison and Validation

This section discusses the performance evaluation of the proposed optimized hyper-parameters CNN-based on the SHO algorithm. The proposed hyper-parameter tuned CNN has been compared with variants of ANN-based and deep learning-based classifiers such as MLP, ELM, DNN, and CNN. The various performance measures based on classification accuracy have been recorded for both Glioma and Brain MRI image datasets. It can be seen that the result of hyper-parameters values that minimize the cost as obtained by using the proposed bio-medical image classifier is showing good results in comparison to other strategies considered for experimentation and comparison. Table 8 depicts the comparison results of MLP, ELM, DNN, and CNN over the proposed tuned model. In this validation, for Glioma dataset it can be seen that for Glioma Tumour class the accuracy, misclassification rate, precision, recall, specificity, F-score and G-mean is showing better result with 98.98%, 1.02%, 98%, 98%, 99.32%, 98%, and 98.66% respectively

Table 8. Performance of evaluation of traditional ANN based strategies, DNN based strategies and the proposed hyper-parameter tuned CNN based on SHO for Glioma dataset

Models	Class Labels	Performance Measures						
		Accuracy	Misclassification Rate	Precision	Recall	Specificity	F1-score	G-mean
MLP	Glioma Tumour	0.8376	0.1624	0.6698	0.7100	0.8810	0.6893	0.7909
	Meningioma Tumour	0.8071	0.1929	0.6560	0.7130	0.8459	0.6833	0.7766
	No Tumour	0.8655	0.1345	0.7766	0.6952	0.9273	0.7337	0.8029
	Pituitary Tumour	0.8959	0.1041	0.7391	0.6892	0.9437	0.7133	0.8065
ELM	Glioma Tumour	0.8858	0.1142	0.7570	0.8100	0.9116	0.7826	0.8593
	Meningioma Tumour	0.8832	0.1168	0.7851	0.8261	0.9068	0.8051	0.8655
	No Tumour	0.9239	0.0761	0.9032	0.8000	0.9689	0.8485	0.8804
	Pituitary Tumour	0.9315	0.0685	0.8219	0.8108	0.9594	0.8163	0.8820
DNN	Glioma Tumour	0.9442	0.0558	0.8750	0.9100	0.9558	0.8922	0.9326
	Meningioma Tumour	0.9467	0.0533	0.9196	0.8957	0.9677	0.9075	0.9310
	No Tumour	0.9492	0.0508	0.9126	0.8952	0.9689	0.9038	0.9313
	Pituitary tumour	0.9467	0.0533	0.8533	0.8649	0.9656	0.8591	0.9139
CNN	Glioma Tumour	0.9822	0.0178	0.9697	0.9600	0.9898	0.9648	0.9748
	Meningioma tumour	0.9695	0.0305	0.9328	0.9652	0.9713	0.9487	0.9682
	No Tumour	0.9848	0.0152	1	0.9429	1	0.9706	0.9710
	Pituitary Tumour	0.9772	0.0228	0.9221	0.9595	0.9812	0.9404	0.9703
Hyper-parameter tuned CNN	Glioma Tumour	0.9898	0.0102	0.9800	0.9800	0.9932	0.9800	0.9866
	Meningioma Tumour	0.9848	0.0152	0.9658	0.9826	0.9857	0.9741	0.9841
	No Tumour	0.9923	0.0077	0.9904	0.9810	0.9965	0.9856	0.9841
	Pituitary Tumour	0.9873	0.0127	0.9726	0.9595	0.9938	0.9660	0.9765

Considering the Meningioma Tumour class, the values recorded are 98.48%, 0.52%, 96.58%, 98.26%, 98.57%, 97.41% and 98.41. The values for the No Tumour class are recorded as, 99.23%, 0.77%, 99.04%, 98.10%, 99.65%, 98.56% and 98.41%, similarly, for the Pituitary class, the observed accuracy measures as stated above are 98.73%, 0.127%, 97.26%, 95.95%, 99.38%, 96.6% and 97.65% respectively which are on the higher side of performance measured values over MLP, ELM, DNN, and CNN. The accuracy measures for the Brain MRI image dataset as shown in Table 9 records 97.40%, 0.26%, 97.87%, 97.87%, 96.67% and 97.87% for accuracy, misclassification rate, precision, recall, specificity, F-score, and G-mean respectively, which are also in-turn shows the better performance of the proposed model for Brain MRI image dataset over MLP, ELM, DNN and CNN classifiers.

Table 9. Performance of evaluation of traditional ANN based strategies, DNN based strategies and the proposed hyper-parameter tuned CNN based on SHO for Brain MRI images dataset

Models	Accuracy	Misclassification Rate	Precision	Recall	Specificity	F1-score	G-mean
MLP	0.7143	0.2857	0.7907	0.7234	0.7000	0.7556	0.7116
ELM	0.8182	0.1818	0.8667	0.8298	0.8000	0.8478	0.8148
DNN	0.8571	0.1429	0.9091	0.8511	0.8667	.8791	0.8589
CNN	0.9221	0.0779	0.9362	0.9362	0.9000	0.9362	0.9179
Hyper-parameter tuned CNN	0.9740	0.0260	0.9787	0.9787	0.9667	0.9787	0.9727

The average training accuracy of the proposed tuned bio-medical image classification is 97.99% and 98.99% for Glioma and Brain MRI image datasets. Similarly, the average testing accuracy can be seen that 98.76% and 99.23% for Glioma and Brain MRI image datasets respectively and are detailed in Table 10.

Table 10. Observed training and testing average-accuracy of hyper-parameter tuned CNN for Glioma and Brain MRI datasets

Datasets	Training	Testing
Glioma	97.99%	98.76%
Brain MRI Images	98.99%	99.23%

The effectiveness of the proposed image classification model for Glioma and Brain MRI image datasets for error-rate, accuracy and improvement over other compared classification methods have been given in Table 11 and Table 12 during training and testing phases respectively. It can be witnessed from Table11 that, there is significant improvement for MLP with 11.23%, ELM with 9.03%, whereas; for DNN and CNN it has 2.63% and 134% improvement for Glioma dataset. Similarly, for Brain MRI image dataset, the improvement of proposed tuned CNN is 24.23%, 16.02%, 11.21% and 7.11% for MLP, ELM, DNN and CNN respectively during training of the instances. During the testing phase as detailed in Table 12, it can be seen that, the percentage of improvement shown by proposed model over MLP, ELM, DNN and basic CNN are 12.93%, 10.78%, 3.77% and 1.75% respectively for Glioma dataset and 25.35%, 17.01%, 12.68% and 7.24% respectively for Brain MRI image dataset.

Table 11. Effectiveness of hyper-parameter tuned CNN using SHO of Glioma and Brain MRI datasets for both traditional ANN based and DNN based classification strategies during training phase of classification

Datasets	Measures	Traditional ANN based strategies		Deep Network based strategies	
		MLP	ELM	DNN	CNN
Glioma	Error-Rate (%)	13.24%	11.04%	4.63%	3.35%
	Accuracy (%)	86.76%	88.96%	95.37%	96.65%
	Improvement (%)	11.23%	9.03%	2.62%	1.34%
Brain MRI Images	Error-Rate (%)	25.24%	17.31%	12.32%	8.22%
	Accuracy (%)	74.76%	82.69%	87.68%	91.78%
	Improvement (%)	24.23%	16.2%	11.21%	7.11%

Table 12. Effectiveness of hyper-parameter tuned CNN using SHO of Glioma and Brain MRI datasets for both traditional ANN based and DNN based classification strategies during testing phase of classification

Datasets	Measures	Traditional ANN based strategies		Deep Network based strategies	
		MLP	ELM	DNN	CNN
Glioma	Error-Rate (%)	14.17%	12.02%	5.01%	1.24%
	Accuracy (%)	85.83%	87.98%	94.99%	98.76%
	Improvement (%)	12.93%	10.78%	3.77%	1.75%
Brain MRI Images	Error-Rate (%)	26.12%	17.78%	13.45%	8.01%
	Accuracy (%)	73.88%	82.22%	86.55%	91.99%
	Improvement (%)	25.35%	17.01%	12.68%	7.24%

For better visualization, the error-rate of the MLP, ELM, DNN, basic CNN and the proposed hyper-parameter tuned CNN for Glioma and Brain MRI images datasets are plotted in graphs as given in Figure 6(a) and Figure 6(b) respectively. The error-rate of the proposed hyper-parameter tuned CNN is dropping significantly approximately during 65 number of iterations whereas; the error-dropping rate of MLP, ELM, DNN and basic CNN can be observed at approximately 137, 178, 73, 128 number of iterations. The accuracy performance with respect to error-rate of hyper-tuned CNN is performing very well and can be witnessed from those two graphs.

Figure 7 (a) to Figure 7(d) show the ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO of Glioma class, Meningioma, No Tumour and Pituitary classes of Glioma dataset respectively. Though, Glioma is a multi-class dataset having four classes, the ROC curves for each class has been plotted. From Figure 7(a), Figure 7(b), Figure 7(c) and Figure 7(d) it can be seen that, the AUC for the proposed image classification model is much more than the AUC of MLP, ELM, DNN and basic CNN for all the four classes of Glioma dataset.

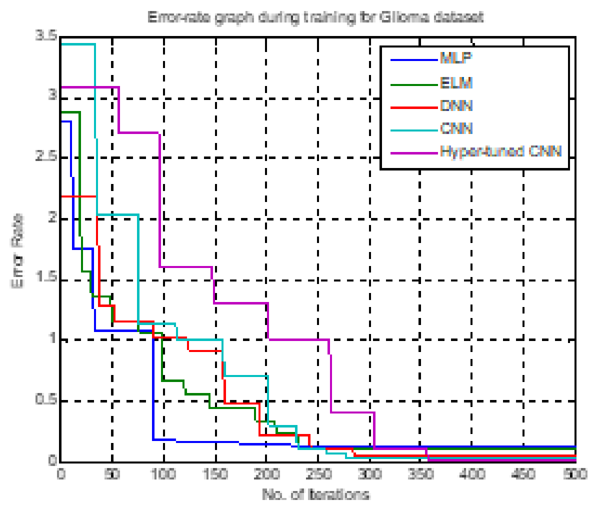

Figure 6(a): Error-rate graph of Glioma dataset during training phase

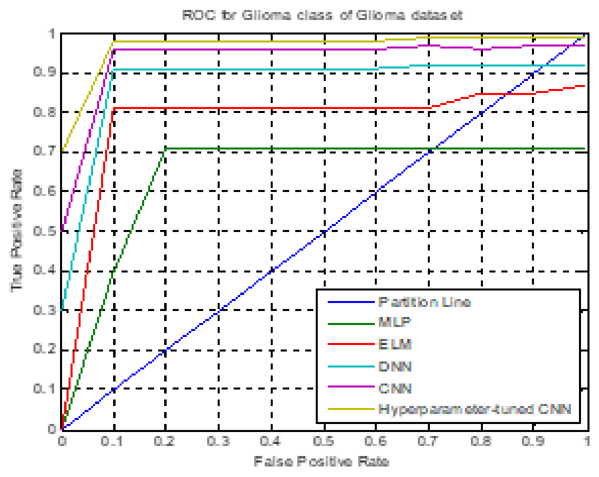

Figure 6(b): Error-rate graph of Brian MRI images dataset during training phase

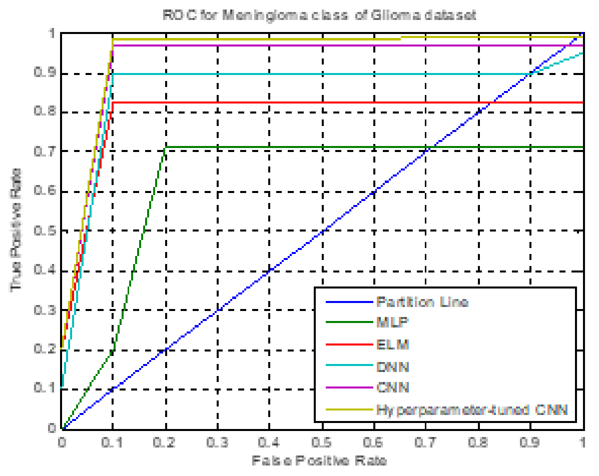

Figure 7(a): ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for Glioma class of Glioma dataset

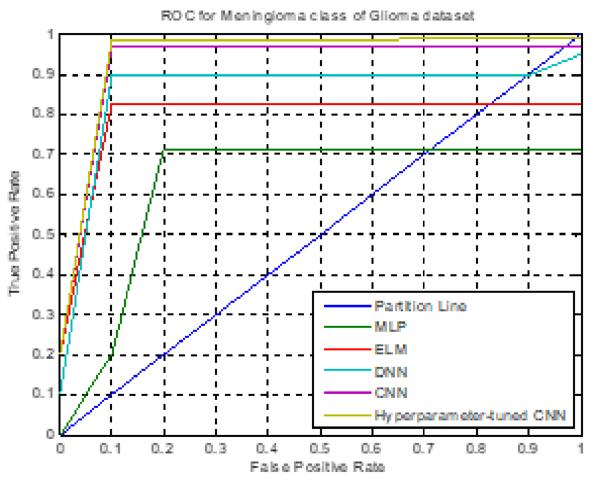

Figure 7(b): ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for Meningioma class of Glioma dataset

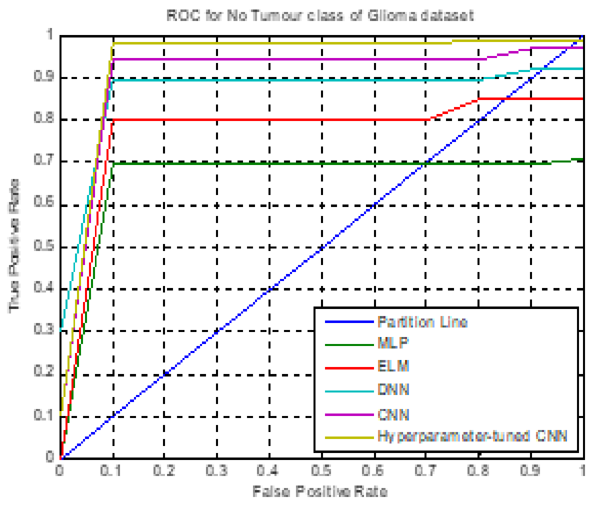

Figure 7(c): ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for No Tumour class of Glioma dataset

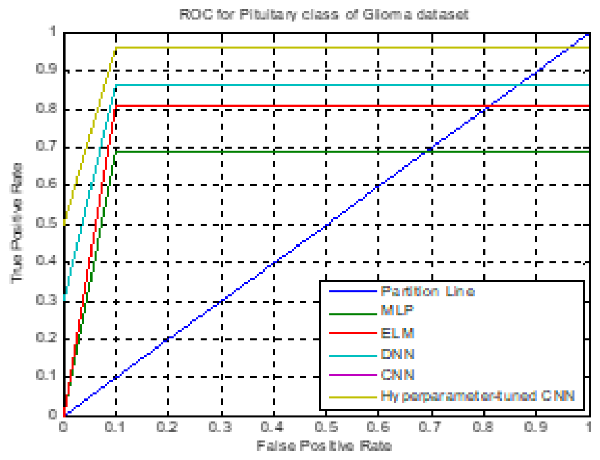

Figure 7(d): ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for Pituitary class of Glioma dataset

Similarly, the Figure 8 (a) shows the ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for Brain MRI images dataset and it can be witnessed that, the AUC for the proposed image classification model is much more than the AUC of MLP, ELM, DNN and basic CNN for this dataset also.

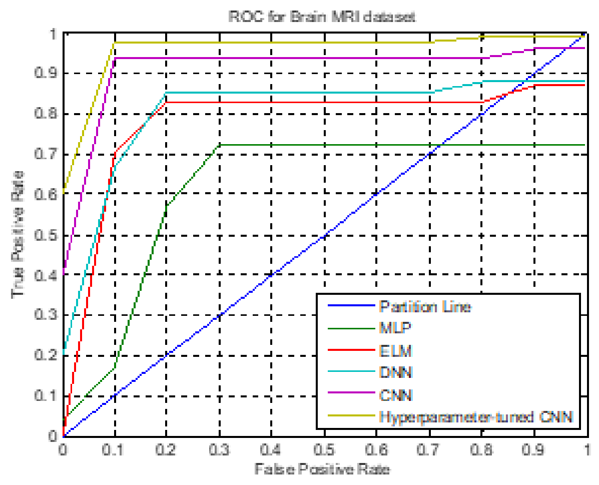

Figure 8: ROC curves of MLP, ELM, DNN, CNN and the proposed hyper-parameter tuned CNN using SHO for Brain MRI images dataset

Statistical Validation

The statistical validation of the propose tuned CNN along with MLP, ELM, DNN and CNN classifiers has been done using Friedman test[39,40] and the average rank values are recorded and shown in Table 13.

Table 13. Average rank of classifiers by Friedman Test

Datasets	MLP	ELM	DNN	CNN	Hyper-tuned CNN
Glioma	0.8676 (5)	0.8896 (4)	0.9537 (3)	0.9665 (2)	0.9799 (1)
Brain MRI Images	0.7476 (5)	0.8269 (4)	0.8768 (3)	0.9178 (2)	0.9889 (1)
Total Rank (T_r)	10	8	6	4	2
T_1^2	100	64	36	16	4

It can be computed as;

$$F_r = \frac{12}{NK(K+1)}(T_1^2 + T_2^2 + \cdots + T_K^2) - 3N(K+1)$$

Where, N = no.of dataset K = no.of hypothesis

$$F_r = \frac{12}{2 \times 5 \times 6}(100 + 64 + 36 + 16 + 4) - 3 \times 2 \times 6 = 8$$

Critical value for $\alpha = 0.05$, $K = 2$ and $N = 2$ is 7.6 which is observed from the Friedman distribution table.

F_r is larger than 7.6, it implies the null hypothesis is rejected.

Discussions and Key Observations

This research proposed a method to obtain the optimal values for the hyper-parameters of CNN using the collaborative behaviour of SHO technique. The findings of this study are discussed and mentioned below.

An empirical bio-medical image classification model has been proposed, experimented and validated.

Two Brain MRI datasets such as; Glioma with multi-classes and Brain MRI images with two-classes are used for experimentation.

The basic objective of this study was to choose the best values for the hyper-parameters such as; learning rate, momentum, number of epochs, batch size, kernel size, kernel type, stride, padding, number of layers in hidden layers, activation functions etc. The focussed challenge among the researchers is to overcome the problem of choosing or setting the values of those parameters manually.

This has been addressed based on SHO algorithm which in-turn has been hybridized with the processing steps of CNN and the obtained optimal values for those hyper-parameter are depicted in Table 6 and Table 7 for Glioma and Brain MRI datasets respectively. Additionally, the processing steps of this proposed hybrid technique have been elaborated through an example.

The error-graphs and ROC curves from Figure 4 and Figure 5 respectively witnesses the effectiveness of hyper-tuned CNN using SHO for Glioma and Brain MRI images datasets.

A detailed comparison has been made on traditional ANN models, deep neural networks and hyper-tuned CNN learning-based image classification and the results showed good performance of the proposed model, which is illustrated in Table 8 and Table 9 for Glioma and Brain MRI images datasets.

After validating and establishing the proposed model, the performance of this model has been compared with traditional ANN strategies such as; MLP and ELM as well as deep network based strategies such as; DNN and basic CNN. The improved results for training and testing phases are given in Table 11 and Table 12, based on the values of Table 10 such as; 97.99% and 98.76% for training and testing of Glioma and 98.99% and 99.23% for Brain MRI images datasets respectively.

The error-rate graphs for both the datasets are given in Figure 6(a) and Figure 6(b). Additionally, the ROC curves for both the datasets are shown in Figure 7(a) to Figure 7(d) and Figure 8 for Glioma and Brain MRI images datasets respectively.

The statistical validation has also been done based on

Friedman Test and can be seen in Table 13.

Finally, the observation can be outlined as; the working principle of SHO is somehow similar to the working principle of PSO. In case of PSO, the mathematics was used for updating the current position considers the local best and global best position whereas; SHO only considers global best position and another key advantage of this SHO is it does not remember the past results like PSO. This advantage of SHO makes this algorithm computationally effective and efficient in terms of its computational power as well as memory requirement.

Conclusion and Future Scope

The authors in this paper presented the empirical model for bio-medical image classification which in turn can automate the selection of hyper-parameters of the chosen CNN classifier. For this purpose, authors first tried to investigate the selection of optimal hyper-parameters of CNN using SHO. The biological and collaborative behaviour of spotted hyenas utilizing the explorative strength along with exploitation in the search region has resulted in an improved version of CNN for choosing the optimal values of hyper-parameters which outperforms the traditional ANN based strategies as well as deep learning based other methods such as; DNN and basic CNN. The key advantage of SHO such as; updating the current position of the hyenas utilizing only the global best position and less memory requirement also shows the effectiveness of this meta-heuristic optimization strategy. However, it can be observed that, it satisfies enough over par around 97.99% and 98.76% for Glioma and 98.99% and 99.25% for Brain MRI images datasets for training and testing phases respectively. Given more time, few more two-class and multi-class image datasets and few more recent optimization techniques can be explored for obtaining the optimal values of hyper-parameters of CNN. Further, dataset used for testing SHO algorithm can be compared with MNIST and CIFAR-10 datasets as future study.

Conflict of Interest

There is no conflict of interest.

Funding:

This research is only for perusing Ph.D. work and no funding agency is involved in this work.

References

1. Khairandish MO, Sharma M, Jain V, Chatterjee JM, Jhanjhi N. A Hybrid CNN-SVM Threshold Segmentation Approach for Tumor Detection and Classification of MRI Brain Images. IRBM. 2021: doi: https://doi.org/10.1016/j.irbm.2021.06.003.
2. El-Dahshan E-SA, Hosny T, Salem A-BM. Hybrid intelligent techniques for MRI brain images classification. Digital signal processing. 2010; 20(2): 433-41. doi: https://doi.org/10.1016/j.dsp.2009.07.002.
3. Zhang Y, Dong Z, Wu L, Wang S. A hybrid method for MRI brain image classification. Expert Systems with Applications. 2011; 38(8): 10049-53. doi: https://doi.org/10.1016/j.eswa.2011.02.012.
4. Mohan G, Subashini MM. MRI based medical image analysis: Survey on brain tumor grade classification. Biomedical Signal Processing and Control. 2018; 39: 139-61. doi: https://doi.org/10.1016/j.bspc.2017.07.007.
5. Abd-Ellah MK, Awad AI, Khalaf AA, Hamed HF. A review on brain tumor diagnosis from MRI images: Practical implications, key achievements, and lessons learned. Magnetic resonance imaging. 2019; 61: 300-18. doi: https://doi.org/10.1016/j.mri.2019.05.028.
6. Tiwari A, Srivastava S, Pant M. Brain tumor segmentation and classification from magnetic resonance images: Review of selected methods from 2014 to 2019. Pattern Recognition Letters. 2020; 131: 244-60. doi: https://doi.org/10.1016/j.patrec.2019.11.020.
7. Zhao X, Ang CKE, Acharya UR, Cheong KH. Application of Artificial Intelligence techniques for the detection of Alzheimer's disease using structural MRI images. Biocybernetics and Biomedical Engineering. 2021; 41(2): 456-73. doi: https://doi.org/10.1016/j.bbe.2021.02.006.
8. Basheera S, Ram MSS. A novel CNN based Alzheimer's disease classification using hybrid enhanced ICA segmented gray matter of MRI. Computerized Medical Imaging and Graphics. 2020; 81: 101713. doi: https://doi.org/10.1016/j.compmedimag.2020.101713.
9. Harish P, Baskar S. MRI based detection and classification of brain tumor using enhanced faster R-CNN and Alex Net model. Materials Today: Proceedings. 2020: doi: https://doi.org/10.1016/j.matpr.2020.11.495.
10. Shahamat H, Abadeh MS. Brain MRI analysis using a deep learning based evolutionary approach. Neural Networks. 2020; 126: 218-34. doi: https://doi.org/10.1016/j.neunet.2020.03.017.
11. Pan S, Guan H, Chen Y, et al. Land-cover classification of multispectral LiDAR data using CNN with optimized hyper-parameters. ISPRS Journal of Photogrammetry and Remote Sensing. 2020; 166: 241-54. doi: https://doi.org/10.1016/j.isprsjprs.2020.05.022.
12. Kousalya K, Saranya T. Improved the detection and classification of breast cancer using hyper parameter tuning. Materials Today: Proceedings. 2021: doi: https://doi.org/10.1016/j.matpr.2021.03.707.

13. Wang Y, Zhang H, Zhang G. cPSO-CNN: An efficient PSO-based algorithm for fine-tuning hyper-parameters of convolutional neural networks. Swarm and Evolutionary Computation. 2019; 49: 114-23. doi: https://doi.org/10.1016/j.swevo.2019.06.002.
14. Dhiman G, Kumar V. Spotted hyena optimizer: a novel bio-inspired based metaheuristic technique for engineering applications. Advances in Engineering Software. 2017; 114: 48-70. doi: https://doi.org/10.1016/j.advengsoft.2017.05.014.
15. Pangle WM, Holekamp KE. Functions of vigilance behaviour in a social carnivore, the spotted hyaena, Crocuta crocuta. Animal Behaviour. 2010; 80(2): 257-67. doi: https://doi.org/10.1016/j.anbehav.2010.04.026.
16. Yirga G, Ersino W, De Iongh HH, et al. Spotted hyena (Crocuta crocuta) coexisting at high density with people in Wukro district, northern Ethiopia. Mammalian Biology. 2013; 78(3): 193-97. doi: https://doi.org/10.1016/j.mambio.2012.09.001.
17. Dhiman G, Kaur A. Spotted hyena optimizer for solving engineering design problems. IEEE; 2017:114-19. doi: https://doi.org/10.1109/MLDS.2017.5.
18. Kumar V, Kaur A. Binary spotted hyena optimizer and its application to feature selection. Journal of Ambient Intelligence and Humanized Computing. 2020; 11(7): 2625-45. doi: https://doi.org/10.1007/s12652-019-01324-z.
19. Dhiman G, Kumar V. Multi-objective spotted hyena optimizer: a multi-objective optimization algorithm for engineering problems. Knowledge-Based Systems. 2018; 150: 175-97. doi: https://doi.org/10.1016/j.knosys.2018.03.011.
20. Dhiman G, Kaur A. A hybrid algorithm based on particle swarm and spotted hyena optimizer for global optimization. Soft computing for problem solving. Springer; 2019:599-615. doi: https://doi.org/10.1007/978-981-13-1592-3_47.
21. Yang L, Shami A. On hyperparameter optimization of machine learning algorithms: Theory and practice. Neurocomputing. 2020; 415: 295-316. doi: https://doi.org/10.1016/j.neucom.2020.07.061.
22. Kumar P, Nair GG. An efficient classification framework for breast cancer using hyper parameter tuned Random Decision Forest Classifier and Bayesian Optimization. Biomedical Signal Processing and Control. 2021; 68: 102682. doi: https://doi.org/10.1016/j.bspc.2021.102682.
23. Jiang W, Siddiqui S. Hyper-parameter optimization for support vector machines using stochastic gradient descent and dual coordinate descent. EURO Journal on Computational Optimization. 2020; 8(1): 85-101. doi: https://doi.org/10.1007/s13675-019-00115-7.
24. Gul E, Alpaslan N, Emiroglu ME. Robust optimization of SVM hyper-parameters for spillway type selection. Ain Shams Engineering Journal. 2021; 12(3): 2413-23. doi: https://doi.org/10.1016/j.asej.2020.10.022.
25. Yoo Y. Hyperparameter optimization of deep neural network using univariate dynamic encoding algorithm for searches. Knowledge-Based Systems. 2019; 178: 74-83. doi: https://doi.org/10.1016/j.knosys.2019.04.019.
26. Cui H, Bai J. A new hyperparameters optimization method for convolutional neural networks. Pattern Recognition Letters. 2019; 125: 828-34. doi: https://doi.org/10.1016/j.patrec.2019.02.009.
27. Aszemi NM, Dominic P. Hyperparameter optimization in convolutional neural network using genetic algorithms. International Journal of Advanced Computer Science and Applications. 2019; 10(6): 269-78. doi: http://dx.doi.org/10.14569/IJACSA.2019.0100638.
28. Singh P, Chaudhury S, Panigrahi BK. Hybrid MPSO-CNN: Multi-level Particle Swarm optimized hyperparameters of Convolutional Neural Network. Swarm and Evolutionary Computation. 2021; 63: 100863. doi: https://doi.org/10.1016/j.swevo.2021.100863.
29. Bhuvaji S, Kadam A, Bhumkar P, Dedge S, Kanchan S. Brain Tumor Classification (MRI): Classify MRI images into four classes. Kaggle. 2020: doi: https://www.doi.org/10.34740/KAGGLE/DSV/1183165.
30. Chakrabarty N. Brain MRI Images for Brain Tumor Detection. Available from: https://www.kaggle.com/navoneel/brain-mri-images-for-brain-tumor-detection/metadata.
31. Amakdouf H, El Mallahi M, Zouhri A, Tahiri A, Qjidaa H. Classification and recognition of 3D image of charlier moments using a multilayer perceptron architecture. Procedia Computer Science. 2018; 127: 226-35. doi: https://doi.org/10.1016/j.procs.2018.01.118.
32. Doğantekin A, Özyurt F, Avcı E, Koc M. A novel approach for liver image classification: PH-C-ELM. Measurement. 2019; 137: 332-38. doi: https://doi.org/10.1016/j.measurement.2019.01.060.
33. Cai Y, Zhang Z, Yan Q, Zhang D, Banu MJ. Densely connected convolutional extreme learning machine for hyperspectral image classification. Neurocomputing. 2021; 434: 21-32. doi: https://doi.org/10.1016/j.neucom.2020.12.064.
34. Rajee M, Mythili C. Gender classification on digital dental x-ray images using deep convolutional neural network. Biomedical Signal Processing and Control. 2021; 69: 102939. doi: https://doi.org/10.1016/j.bspc.2021.102939.
35. Ghassemi N, Shoeibi A, Rouhani M. Deep neural network with generative adversarial networks pre-training for brain tumor classification based on MR images. Biomedical Signal Processing and Control.

2020; 57: 101678. doi: https://doi.org/10.1016/j.bspc.2019.101678.

36. Yang G, Ding F. Associative memory optimized method on deep neural networks for image classification. Information Sciences. 2020; 533: 108-19. doi: https://doi.org/10.1016/j.ins.2020.05.038.
37. Espíndola RP, Ebecken NF. On extending f-measure and g-mean metrics to multi-class problems. WIT Transactions on Information and Communication Technologies. 2005; 35: 25-34. doi: https://www.doi.org/10.2495/DATA050031.
38. Yousef WA, Wagner RF, Loew MH. Comparison of non-parametric methods for assessing classifier performance in terms of ROC parameters. IEEE; 2004:190-95. doi: https://doi.org/10.1109/AIPR.2004.18.
39. Statistics How To. Friedman's Test / Two Way Analysis of Variance by Ranks. Accessed 29/09/2021, Available from: https://www.statisticshowto.com/friedmans-test/.
40. Upper Critical Values for The Friedman Test. Accessed 29/09/2021, Available from: https://www.york.ac.uk/depts/maths/tables/friedman.pdf.