Knowledge and Information Systems (2022) 64:2411–2434
https://doi.org/10.1007/s10115-022-01707-3

REGULAR PAPER

A novel approach for optimization of convolution neural network with hybrid particle swarm and grey wolf algorithm for classification of Indian classical dances

Jhansi Rani Challapalli[1] · Nagaraju Devarakonda[1]

Received: 12 April 2022 / Accepted: 26 June 2022 / Published online: 28 July 2022

Abstract
Deep learning is the most dominant area to perform the complex challenging tasks such as image classification and recognition. Earlier researchers have been proposed various convolution neural network (CNN) with different architectures to improve the performance accuracy for the classification and recognition of images. However, the fine-tuning of hyper parameters, resulting the optimal network, regularization of parameters is the difficult task. The metaheuristic optimization algorithms are used for solving such kind of problems. In this paper we proffer a fine tune automate CNN with Hybrid Particle Swarm Grey Wolf (HPSGW). This novel algorithm used to discover the optimal parameters of the CNN like batch size, number of hidden layers, number of epochs and size of filters. The proffered optimized architecture is implemented on MNIST, CIFAR are two bench mark datasets and Indian Classical Dance (ICD) for the classification of 8 Indian Classical Dances. The Proffered method improves the model performance accuracy of 97.3% on ICD Dataset, and other benchmark datasets MNIST, CIFAR with an improved accuracy of 99.4% and 91.1%. This auto-tuned network improved the performance by 5.6% for Indian Classical Dance Forms Classification compared to earlier methods and also reduces the computational cost.

Keywords Particle swarm optimization · Indian classical dance · Grey wolf · Convolution neural network

1 Introduction

Deep learning algorithms is one of the most dominant field widely implemented for a large variety of applications [1, 2] because of their higher levels of accuracy compared to previous approaches. Deep neural networks have shown that they can tackle classification issues with hierarchical model, lots of parameters, in addition learning from large databases. CNNs are

✉ Jhansi Rani Challapalli
challapalli.jhansirani@gmail.com

Nagaraju Devarakonda
dnagaraj_dnr@yahoo.co.in

1 School of Computer Science and Engineering, VIT-AP University, Amaravathi, A.P, India

the type of deep neural network that has been shown to be a reliable intended for image processing or video processing and pattern recognition, classification, approach. They are made up of a lot of convolutions, pooling, and layers and fully connected layers (FC). In current era, CNN has attracted consideration for producing improved results in a range of computer vision applications, including medical [3], NLP (Natural-language-Processing), AI-robotics, Radiology [4] and aerospace. While designing CNN a large no. of parameters such as bias, learning rate, optimizer, activation function, no. of layers, no. of filters, and size of each filter need to optimize. These parameters can be varying for various problems these parameters are referred as hyper parameters. The auto tuning of hyper parameters has been done experimentally if not properly tuned then the accuracy of the model becomes poor.

To address such problem, a number of researchers have been proposed using evolutionary computation methodologies to inevitably create the best CNN architectures to improve their performance such as harmonic search [5], microcanonical optimization algorithm [6], differential equation [7] to optimize CNN parameters. [8] suggested a method for developing the deep convolutional neural networks architectures for classification problems using genetic algorithms. [9] has been proposed a computationally effective algorithm for designing unsupervised deep neural network for big data. The authors of [10] give an examination of alternative evolutionary computing-based strategies for optimizing CNN designs, which were verified on benchmark datasets and produced viable outcomes. [11] proposed an automatic design of CNN topologies by using genetic algorithms and grammatical evolution. [12] proposed optimized CNN with PSO algorithm for sign language recognition. [13] proposed grey wolf-based hyper parameters optimized CNN classifier for detection of skin cancer. Even though there are so many challenges in classification of images. The scenario is that the classification of dance forms is a complex task due to some dances have same type of mudras, and dancers wear heavy costumes. Furthermore it is a time-consuming process to build efficient network manually.

In this research we proffer a novel HPSGW to progress the accuracy of CNN model by hyper parameter tuning for handling multi classification problems. The hyper parameter tuning can be done by considering the no. of layers, each kernel size, batch size and no. of epochs. The training process of CNN, the fitness function to be assessed in the HPSGW optimization process, which is most time-consuming element of hyper parameter tuning. To increase the hyper-parameter tuning efficiency, appropriate fitness function is constructed that can used to compute the fitness value to result CNN accuracy in short amount of time in order to classify the images. In terms of both precision and loss, the improved CNN, the experimental findings clearly illustrate the superiority of the proffered method over manual hyper parameter tuning optimization approaches. The key contributions of this paper are.

- Pre-process the images effectively in order to decrease the training time.
- Construct CNN models for MNIST, CIFAR, ICD Datasets without optimization.
- HPSGW optimizes CNN hyperparameters to model the most cost-effective CNN Classifier for Indian Dance Classification.
- The proffered method is compared with the CNN without auto tuning with hyper parameters tuning using HPSGW by applying on the bench mark datasets MNIST and CIFAR.
- Compare the proposed optimized CNN ICD model with the earlier ICD CNN models.
- Compare the proffered method performance with state of art results on bench mark datasets.
- Compare manual tuning models with auto tuning models for each data set by conducting statistical test.

The remaining paper is arranged as: Sect. 2 discusses some prevailing research studies, Sect. 3 exhibits related work, Sect. 4 focuses on the mathematical formulation of proffered work, Sect. 5 shows the experiment outcomes, Sect. 6 comparative analysis and discussion, and lastly, Sect. 7 finishes with the conclusion and prospective studies.

2 Literature review

For decades, object identification and visual categorization have been hot topics, with neural networks, particularly CNN, playing a crucial role. FC and hidden layers of networks produce a complex cost and time complexity when dealing with object recognition problems. CNN is now widely utilized for image classification as a solution to this challenge. CNN includes layers named as convolution, max-pooling, ReLu, dropout Layer, dense Layer, for feature extraction and SoftMax layer for classification. The size of Kernal, no. of kernels, stride of kernel, Relu activation function associated with convolution layer, size of kernel associated with pooling layer, dropout rate associated with dropout layer. The size of the dense layer, learning rate, number of convolutions, number of pooling layers, number of dense layers, and so on are all hyper-parameters associated to individual layer of the CNN model. The output of the CNN model is heavily influenced by these hyper-parameters. In order to increase the model's accuracy, several academics have suggested growing the no. of layers in CNN. In the instance of automatic detection of features in CNN, simple characteristics are identified by the first layers, tracked by complicated features extracted by the subsequent layers. The depth of the model is significant when extracting complicated characteristics. Each layer's feature selection is influenced by the number of kernels and kernel size. Researchers employ the principle by means of fewer kernels in the first layer and a large number of features in the second layer to produce better features. When training the model with any training technique such as optimizer Adam, gradient descent with momentum, and so on, the learning rate is crucial. If the learning rate is too small, the gradient will take a long time to reach the optimal solution. Gradient accelerates approaching the solution if it is large. As a result, selecting the appropriate learning rate is a crucial consideration. The dropout rate is used to regularize CNN. The model is more generalized when the dropout rate is correct.

Researchers have proposed various promising CNN models in the literature. For picture classification and object detection, [14] suggested a GoogleNet model with 22 deep layers. [15] proposed the Segnet pixel wise image labelling neural network. For image classification applications, [10] presented an optimized DCNN architecture with ACL (autonomous and continuous Learning), such as a genetic algorithm. [16] suggested a CNN model for image classification using quantum behaved binary PSO. The HPSGW was proposed by [17] to improve exploitation in PSO and exploration in GWO. It has been applied to unimodal, multimodal, and static dimension multimodal test functions to verify solution quality and improved performance. [18] has proposed hybrid optimization algorithm by PSO and GWO algorithm for IOT intrusion detection system to classify the data and detect intrusions. The optimized CNN network has been proposed by [19] by auto tuning of hyper parameters with GWO technique. This model has been implemented for covid-19 dataset for the diagnosis of covid-19. For recognition of human behaviour form unconstrained videos [20] proposed hybrid CNN-GWO. The obtained solutions of the GWO approach heavily influence the initial weights for the proposed deep CNN classifiers, which in turn minimizes the 'classification' mistakes. To improve the accuracy of CNN models for classification tasks, [21] presented a mix of GA and PSO. Benchmark datasets like MNIST, CIFAR-10, and SVHN, this hybrid CNN model has been tested. As a result, it has a higher level of accuracy than other methods.

PSO use in convolutional neural networks was proposed by [22]. The use of PSO in the training phase aims to improve recognition accuracy by optimizing the solution vectors' outcomes on CNN. [23] has proposed a self-adaptive extreme learning machine for the auto adjustment of parameters of network in the training process using self-adaptive learning algorithm. It selects the best neuron in network hidden layers. For the design of automatic architecture, [24] suggested an architecture evolution of convolutional neural networks utilizing monarch butterfly optimization. By using a variant of the PNN with self-adaptive approach, [25] proposed an enhanced probabilistic neural network with self-adaptive for transformer defect diagnostics. Spread can be self-adaptively changed and selected in this manner, and the optimal spread can then be used for both testing and training. [26] has proposed a novel approach that can be used to improve the detection of malware variants using deep learning. Apart from these, Table 1. depicts some of the recent Metaheuristic algorithms which has been proposed by earlier researchers for solving various types of engineering problems.

3 Related work

This section describes the basic concepts needed to implement the proffered methodology.

3.1 CNN Model for ICD classification

The CNN is widely employed in image classification because of its exciting characteristics such as automatic selection of features and end-to-end training. Pooling, dropout, and dense layers, in addition to the convolution layer, play important roles in CNN. CNN can accomplish efficient image processing through its convolution layer's automatic feature selection, pooling layer's feature reduction, and dense layer's classification. Figure 1 depicts the CNN network's structure.

Every layer of CNN is significant in its own right. The number of features taken from every layer is determined by the kernel size and no. of kernels. Random weights are used to initialize kernel weights, which are learned during model training. The convolution layer's output is used as the ReLu layer's input. Activation function that is nonlinear ReLu keeps the convolution layer's value within a certain range. Because of its easiness and non-negativity, the ReLU activation function is the furthermost common in CNN. The dropout layer is employed after ReLu to keep the model from overfitting. The feature map is then down sampled using the pooling layer. It aids the representation's invariance to modest input changes. To produce a new set of weight maps, the pooling layer works on each and every feature maps distinctly. We choose max pooling, which takes the maximum value from each patch of the feature maps. It aids in the extraction of low-level features such as the image's point and edge. The CNN's higher-level layers are usually FC layers. These FC layers take the pooling layer's output and use it to make a classification conclusion. The CNN model's last layer employs a softmax function, which outcomes a multiclass classification probability distribution. In the CNN model, regularization is a strategy to deal with overfitting. On adding a penalty to the loss function, regularization reduces overfitting. Dropout was a technique for dealing with overfitting in CNN that involves minimizing interdependent learning. After defining a CNN's structure, back propagation is used to tweak its core weights to meet the goal problem. Kernel length, wide variety of kernels, stride withinside the kernel, activation characteristic concerned in convolution layer, kernel length related to max-pooling, dropout rate utilized in dropout, length of the dense layer, learning rate wide variety of convolutions, wide variety of

Table 1 List of Some Recent Metaheuristic Algorithms

Reference	Algorithm	Inspiration	Year	Purpose	Applied On	Tested with
[27]	White Shark Optimizer (WSO)	Behaviour of White Sharks	2022	Improves global optimality and avoid local minima	Solve the bench mark problems of the CEC-2011 EA Competition	29 Benchmark Functions
[28]	African vultures optimization (AVO)	Foraging and behaviour of Africal Vulture	2021	Focuses on Global Optimization	11 Engineering design Problems	36 benchmark functions
[29]	Artificial Gorilla Troops optimization (AGTO)	Social Intelligence	2021	Designed to perform Exploration and Exploitation	7 Engineering problems particularly on high dimension al problems	52 bench mark functions
[30]	Political Optimizer (PO)	Socio inspired	2020	Excellent convergence speed with good exploration capability in early iterations	4 engineering problems	50 unimodal and multimodal benchmark functions
[31]	Forensic-based investigation algorithm (FBI)	Suspect investigation–location–pursuit	2020	Faster convergence and a shorter computational time	Solved high dimension problems	1000 bench mark functions
[32]	Jellyfish Search (JS)	Behaviour of jellyfish in the ocean	2020	Global optimization problem	Structural optimization problems	50 small/average and 25 large bench mark functions
[33]	Golden Eagle Optimizer (GEO)	Nature inspired swarm based	2020	Highlight exploration and exploitation for a global optimization method	For both single and multi-object problems	33 bench mark functions
[34]	Firebug Swarm Optimization (FSO)	Swarm Intelligence	2021	Global optimization problems	highly multi-modal problems	2013 benchmark functions suite
[35]	Harris Hawks optimization (HHO)	Cooperative behaviour and stalking style of Harris's hawks	2019	Focuses on dynamic patterns and behaviours to develop optimization problem	Real-world engineering problems	29 bench mark functions

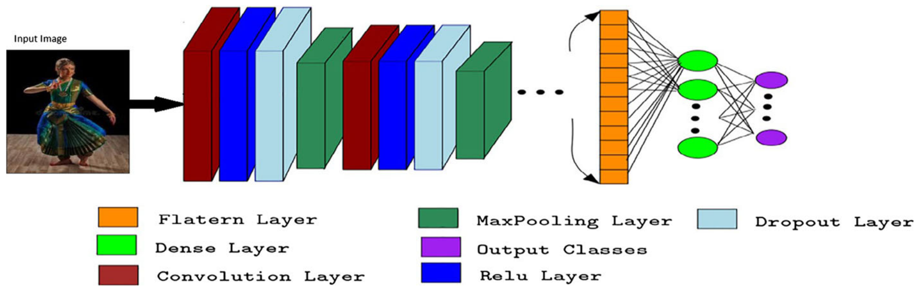


Fig. 1 ICD CNN Architecture

max-pooling, wide variety of dense layers, and so forth are all hyper-parameters associated with each and every layer of the CNN model. These hyperparameters have a substantial effect at the CNN model's output. Manually determining the near-optimal hyper-parameter setup for a CNN by investigating all potential combinations at a reasonable cost is almost impossible. As a result, proper CNN hyper-parameter refinement is framed as an optimization problem having an intent of improving CNN model's overall performance.

3.2 Particle swarm optimization (PSO) algorithm

Kennedy and Eberhart first published the PSO algorithm in [36], and its outcome has been inspired by imitations of animal social behaviour like bird grouping and fish schooling. The birds scatter or congregate while hunting for food before settling on a location where they can get the food. While the birds are moving from one location to another in quest of food, there is constantly one bird it can smell the food extremely clearly; this means that the bird is attentive of the location where prey can be located and has the right prey resource communication. The birds will simultaneously fly to the location where food can be obtained since they are communicating the message. This method is based on animal behaviour for calculating global optimization functions/problems, and each swarm/crowd member is referred to as a particle. Two mathematical equations update the positions of each crowd partner in global search space in the PSO technique. These are the mathematical equations.

$$v_i^{t+1} = w * v_i^t + c1r1\left(xBest_i^t - x_i^t\right) + c2r2\left(gBest_i^t - x_i^t\right) \tag{1}$$

The next velocity of each particle v_i^{t+1} can be calculated by using Eq. (1). Where w is the initial inertia, v_i^t is the previous velocity of particle i at time$t$, $r1$, $r2$ are random numbers whose values among [0, 1], $c1$ and $c2$ are the initial acceleration constants. $xBest_i^t$ is local best fitness value of swarm, $gBest_i^t$ is global best fitness value of swarm. x_i^t is the previous position of the swarm i at the time$t$.

$$c2r2(gBest_i^t - x_i^t) \tag{2}$$

where x_i^{t+1} is the next position of the particle it can be calculated by using the previous particle position x_i^t and its velocity v_i^{t+1} by using Eq. (2). By Eqs. (3) & (4) at a given time t we can update the local and global best values.

$$xBest_i(t+1) = \begin{cases} xBest_i(t) iff(xBest_i(t) \le f(x_i(t+1)) \\ x_i(t+1) iff(xBest_i(t) \ge f(x_i(t+1)) \end{cases} \tag{3}$$

$$\mathrm{gBest(t+1) = max\{f(y), f(gBest(t))\}\ where,} \tag{4}$$

$$\mathrm{y \in pBest_0(t), pBest_1(t), ..., pBest_2(t)(t)}$$

3.3 Grey wolf optimization algorithm

In addition, GWO is a metaheuristic optimization algorithm introduced by [37]. It was inspired by Grey Wolves' social order and hunting strategy. The alpha, beta, and omega wolves in each group are ranked, with the remaining subordinate wolves categorized as delta. The population in GWO is separated into 4 groups: Alpha(α), Beta(β), Delta(δ), and Omega(γ), which are used to pretend the leadership structure. Alpha wolves are the group's decision-makers, overseeing all of the group's living activities, including the hunt. Beta wolves are the alpha wolf' subordinates, and they support the alpha wolves' decisions. Omega are present in the group's next rank and preserve the group's hierarchical structure of dominance. The rest of the wolves who are subordinate to the Omega are known as Delta. GWO solution is classified into three levels based on the solution's fitness and optimality. For the optimizing problem, the alpha decision is most likely the best option. In addition, swarm intelligence approaches are employed to address the optimization problem when there is no leader to supervise the entire period. The GWO approach overcomes this issue by giving grey wolves the ability to lead themselves. GWO, which were inspired by grey wolves and had the potential of handling image recognition and classification challenges, were used in the swarm intelligence improvisations. The GWO variation retained the quality of grey wolf leadership for the sake of nature by performing the hunting mechanism. If the wolf isn't an alpha (α), beta (β), or omega (γ), the subordinate must surrender to superiors. Hunting, prey finding, prey attacking, and prey surrounding are the primary processes in the GWO that are employed for optimization. Each swarm agent's encircling behaviour is depicted in the mathematical Eqs. 5, 6.

$$\overrightarrow{D} = \left|\overrightarrow{C}.\overrightarrow{X}_P(t) - \overrightarrow{X}(t)\right| \tag{5}$$

$$\overrightarrow{X}(t+1) = \left|\overrightarrow{X}_P(t) - \overrightarrow{A}.\overrightarrow{D}\right| \tag{6}$$

where D is the enriching behaviour of each agent, current iteration is t, the prey position is Xp, the position of Grey Wolf is X and A, C are coefficient vectors. A and C can be computed by using Eqs. 7 8.

$$\overrightarrow{A} = 2 * \overrightarrow{a} * \overrightarrow{r1} - \overrightarrow{a} \tag{7}$$

$$\overrightarrow{C} = 2 * \overrightarrow{r2} \tag{8}$$

Here 'a' is linearly dropped from 2 to 0 consequently in a no. of iterations, random vectors such as r1 and r2 whose values are selected within a interval of [0, 1]. Equations 9, 10 and 11 are utilized to update the Grey Wolf's place in relation to the pray's position. By altering the values of A and C, several locations surrounding the optimal search agent will be grasped in relation to the present place. The following mathematical formulae can be used to calculate the Grey Wolf's hunting procedure.

$$\overrightarrow{D}_\alpha = \left|C1.\overrightarrow{X}_\alpha - \overrightarrow{X}(t)\right| \tag{9}$$

$$\overrightarrow{D}_{\beta} = \left|C2.\overrightarrow{X}_{\beta} - \overrightarrow{X}(t)\right| \tag{10}$$

$$\overrightarrow{D}_{\delta} = \left|C3.\overrightarrow{X}_{\delta} - \overrightarrow{X}(t)\right| \tag{11}$$

$$\overrightarrow{X}_{1} = \left|\overrightarrow{X}_{\alpha} - A_1\overrightarrow{D}_{\alpha}\right| \tag{12}$$

$$\overrightarrow{X}_{2} = \left|\overrightarrow{X}_{\beta} - A_2\overrightarrow{D}_{\beta}\right| \tag{13}$$

$$\overrightarrow{X}_{3} = \left|\overrightarrow{X}_{\delta} - A_3\overrightarrow{D}_{\delta}\right| \tag{14}$$

where, $\overrightarrow{\mathrm{D}}$, $\overrightarrow{\mathrm{D}}_{\beta}$, $\overrightarrow{\mathrm{D}}_{\delta}$ are distance vectors, $\overrightarrow{\mathrm{X}}$, $\overrightarrow{\mathrm{X}}_{\beta}$, $\overrightarrow{\mathrm{X}}_{\delta}$ are the positions vectors wolves ,β and δ respectively andA_1,A_2,A_3,C1,C2C3 are coefficient vectors. The position of the Grey Wolves can be updates as

$$\overrightarrow{X}(t+1) = \frac{\overrightarrow{X}_{1} + \overrightarrow{X}_{2} + \overrightarrow{X}_{3}}{3} \tag{15}$$

The random values are in the range [-2a, 2a], and the chosen value is compared to the gap. If the random value |A|< 1 is 1, the attacks must be directed at prey. If the prey's searching ability is investigated, the prey's attacking ability is investigated, and the results are used to move against the prey. The population members are obliged to stay away from prey divergence.

4 Proposed methodology

This section describes the inspiration for, and the mathematical simulation that is involved in HPSGW. And how to build an automated CNN with metaheuristic algorithm named as HPSGW for classification problems. The network built by auto tuning of hyperparameters such as No. of Layers, No. of Kernals, Batch size and No. of Epochs. The final CNN was built with best hyper parameter which gives best optimal value.

4.1 CNN architecture optimized by hybrid PSGW algorithm

The HPSGW algorithm has been created without altering the overall calculations of both PSO and GWO approaches. Practically all real-world glitches can be resolved effectively using PSO methodology. Though, there must be a way to learn the likelihood of the PSO tricking by local minimum. The GWO is used to support PSO in our proposed strategy to lessen the probability of slipping towards local minimum. To avoid local minima, the PSO algorithm moves some particles in to random locations with a limited chance of success. To avoid these dangers, the GWO exploration capability is employed to send some particles to sites that are somewhat boosted by GWO method rather than random positions. However, because the GWO method is used in adding to the PSO algorithm, the running duration is improved. The proposed PSGW Algorithm merges the two functionalities

$$\overrightarrow{D}_{\alpha} = \left|C1.\overrightarrow{X}_{\alpha} - w * \overrightarrow{X}(t)\right| \tag{16}$$

$$\overrightarrow{D}_{\beta} = \left|C2.\overrightarrow{X}_{\beta} - w * \overrightarrow{X}(t)\right| \tag{17}$$

$$\overrightarrow{D}_{\delta} = \left|C3.\overrightarrow{X}_{\delta} - w * \overrightarrow{X}(t)\right| \tag{18}$$

Equations 19 and 20 are used to combine PSO and GWO variants the to update the velocities as

$$v_i(t+1) = w * (v_i t + c1r1(x_1 - x_i t) + c2r2(x_2 - x_i t) + c3r3((x_3 - x_i t)) \tag{19}$$

$$x_i(t+1) = x_i t + v_i(t+1) \tag{20}$$

To choose an optimal ideal value, the fitness function is applied. The fitness function is calculated using the Rosen Brock function and the objective function in the proposed hybrid optimization technique. Requiring the following Eq. 21, the Rosen Brock function is efficiently optimized by adapting an appropriate coordinate system without using any gradient information or generating local approximation models f(x).

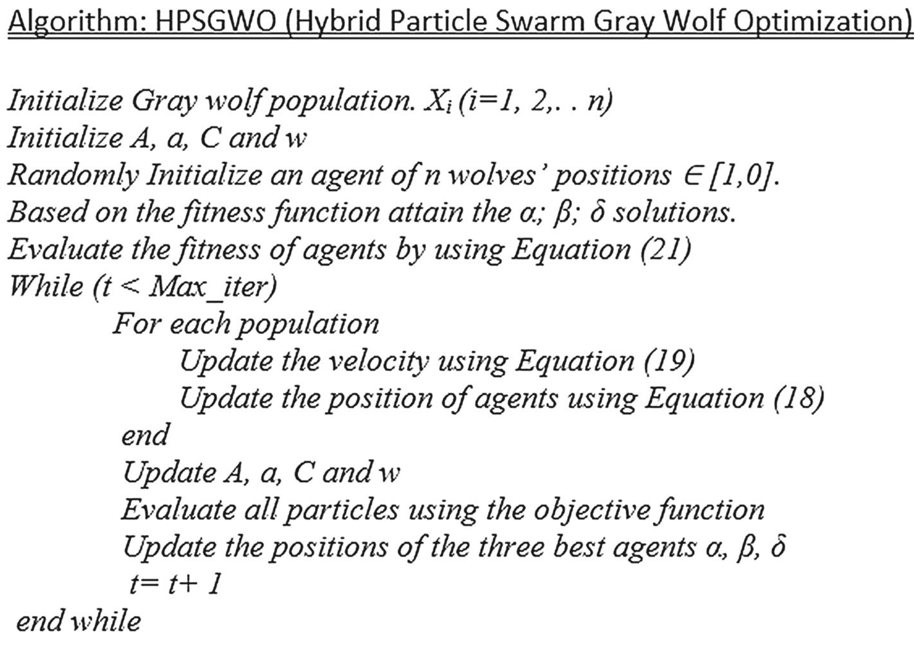
$$f(x) = \sum_{i=1}^{N-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (1 - x_i)^2 \right] \tag{21}$$

The pseudo code for the HPSGW algorithm is shown in Fig. 2.

The objective of this work is to implement CNN architecture with good performance by using hybrid PSGW algorithm. The parameters needed to optimize the network architecture are.

- No. of Convolution-layers.
- Size of Filter or kernel used in every convolution.
- No. filter used to extract feature maps.
- Batch size means the no. of input images passed through CNN in every training block.

The proposal's main theme is shown in Fig. 3 with the "training and optimization" block



```
Algorithm: HPSGWO (Hybrid Particle Swarm Gray Wolf Optimization)

Initialize Gray wolf population. $X_i$ (i=1, 2,. . n)
Initialize A, a, C and w
Randomly Initialize an agent of n wolves' positions $\in [1,0]$.
Based on the fitness function attain the $\alpha$; $\beta$; $\delta$ solutions.
Evaluate the fitness of agents by using Equation (21)
While (t < Max_iter)
        For each population
            Update the velocity using Equation (19)
            Update the position of agents using Equation (18)
        end
        Update A, a, C and w
        Evaluate all particles using the objective function
        Update the positions of the three best agents $\alpha$, $\beta$, $\delta$
        t= t+ 1
end while
```

Fig. 2 Pseudo code for HPSGW Algorithm

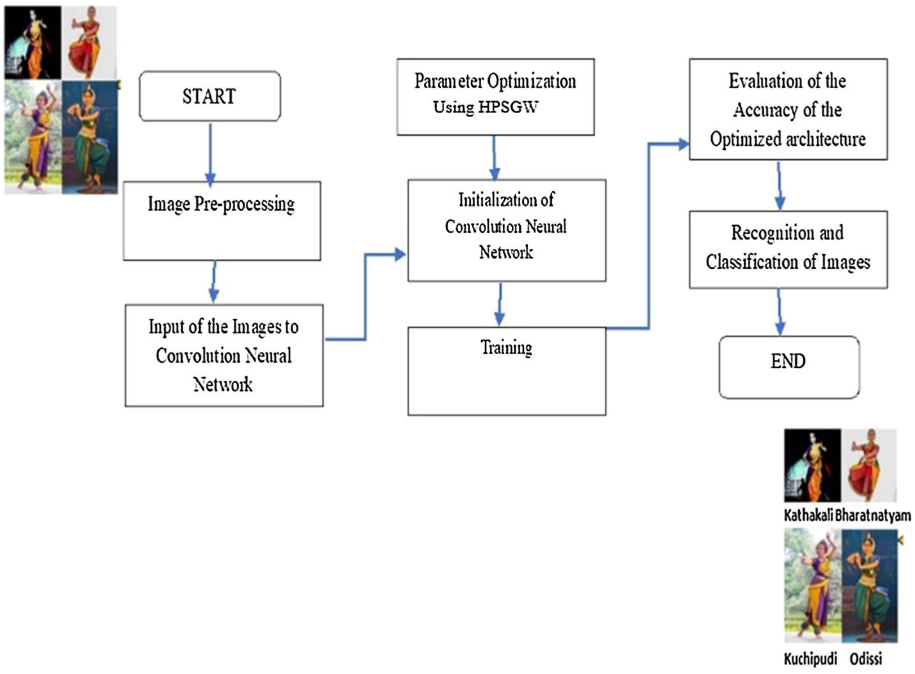

START
Image Pre-processing
Input of the Images to Convolution Neural Network
Parameter Optimization Using HPSGW
Initialization of Convolution Neural Network
Training
Evaluation of the Accuracy of the Optimized architecture
Recognition and Classification of Images
END
Kathakali Bharatnatyam
Kuchipudi Odissi

Fig. 3 CNN optimization with HPSGW Algorithm

being the atmost essential element of the entire process, in which the CNN model is initialized to collaborate with parameter optimization using the HPSGW. The HPSGW is initialized according to the parameters specified for the execution (described below), and the particles are generated as a result. Each agent represents a complete CNN training since it represents the possible best solution and each position will have a parameter value to be optimized.

The training procedure is an iterative cycle that culminates in the evaluation of all agents created for each iteration by the HPSGW. The computing cost is larger, and it is determined by size of the database, population size, HPSGW iterations, and the no. of agents in every iteration. If the HPSGW is run 100 times with 5 agents, the CNN model training procedure is run 500 times as well. The steps for using the HPSGW Algorithm to optimize the CNN are as follows.

1. Initially load the Dataset MNIST, CIFAR and ICD for training process. Before training pre-process, the dataset such as scaling, colouring and resizing.
2. HPSGW population is generated. The population parameters include no. of agents, inertia weight, and no. of iterations, No. of Agents, accelerator factors (A1, A2, A3).
3. CNN architecture is initialized, with the parameters obtained by HPSGW no. of layers, the filter size, no. filters, and the batch size.
4. CNN model training and model validation. CNN reads and process the input images collected from the datasets for training, process, validation process, and testing process; this step results in accuracy for each model. These results will return to the HPSGW as immediate step of objective function.
5. Derive the objective function (fitness for each agent).

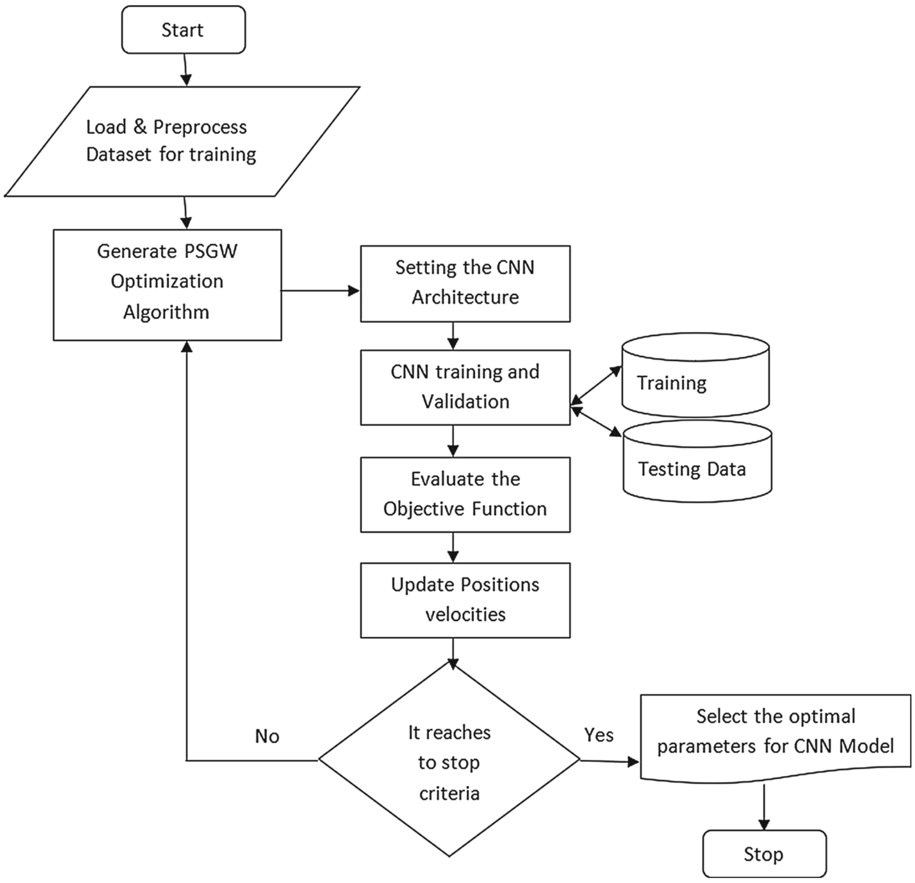

Start
Load & Preprocess Dataset for training
Generate PSGW Optimization Algorithm
Setting the CNN Architecture
CNN training and Validation
Training
Testing Data
Evaluate the Objective Function
Update Positions velocities
It reaches to stop criteria
No
Yes
Select the optimal parameters for CNN Model
Stop

Fig. 4 Flow Chart of HPSGW

6. Revise the parameters. At each new iteration, every agent revises their velocity and position.
7. This process is repeated until stop criteria matched.
8. Finally, the CNN is modelled with best optimal (fitness) value.

Figure 4 illustrates the hyper tuning process of CNN using HPSGW using flow chart.

5 Experimental results

In this work, HPSGW algorithm is implemented on 3 standard datasets MNIST, CIFAR-10, and ICD Dataset which were collected from Kaggle. Alternative convolution and pooling layers make up the CNN used in this study. A complete FC layer is built on the top of the network.

Table 2 Parameters of CNN and HPSGW

Parameters of CNN	
Learning Function	Adam
Activation Function	SoftMax
Nonlinearity Function	ReLu
Epochs	100(max)
Parameters of HPSGW	
Agents	5
Inertia weight	0.5
No. of Iterations	100
Upper Bound	100
Lower Bound	10
Threshold	0.5
Accelerator Factor (C1, C2, C3)	2

5.1 Datasets description

MNIST Dataset: Handwritten digits 0–9 are included in the MNIST [23]. There are 60,000 samples in this collection. With batch size 128, 50,000 sample photos were used in training procedure and the remainder were used in testing procedure. All the photos are of equal size, at 28 × 28 pixels. Before the training, the pixels are normalized to [0, 1].

CIFAR Dataset: This dataset comprises of 60,000 natural RGB images of pixel size 32 × 32, divided into 10 classes, with 50,000 for training procedure and 10,000 for testing procedure.

ICD Dataset: This dataset has 560 images of size 224 × 224 of various 8 categories of Indian classical dance form images like Bharatnatyam, Odissi, Kathakali, manipuri, kuchipudi, mohiniyattam, sattriya and Kathak. Among 364 images were used for training and remaining are used for testing. Each image varies in size and resolution so apply pre-process techniques before training and testing.

5.2 Parameters used in the experimentation

In CNN model Parameter setting, ReLu layer uses nonlinearity function, soft max layer uses activation function for classification, learning function is an Adam optimizer and epochs are employed as static parameters. No. of particles, iterations, weight of inertia, and the accelerator factor are all fixed parameters in the HPSGW design. Table 2 shows the static parameters utilized in HPSGW and CNN. The no. of hidden layers, size of the filters utilized in each hidden layer, the no. of filters, and the batch size are the dynamic parameters optimized by HPSGW.

5.3 Experiment results obtained by CNN-HPSGW and without HPSGW

In this segment we represent optimized results conducted on three different data sets. This experimentation consists of 100 executions applied on each dataset to obtain optimal network with few parameters and maximum accuracy.

5.3.1 MNIST dataset with /without HPSGW

The first experiment was done on MNIST dataset without HPSGW. During training phase, the model was trained with 60,000 images and tested 10,000 images. This method results model accuracy 97.4%, sensitivity rate 98.4% and specificity 99%. Figure 5 represents the true positive rate and false positive rate for each classification labels.

After finetuning of hyper parameters of CNN with HPSGW returns the optimized CNN architecture at 10th iteration. This algorithm finds the best hyper parameters with best global score with those hyper meters the model was built. This method results model accuracy 99.4%, sensitivity rate 98.4% and specificity 99%. Roc curve in Fig. 6 represents the true positive rate and false positive rate for each classification labels for MNIST.

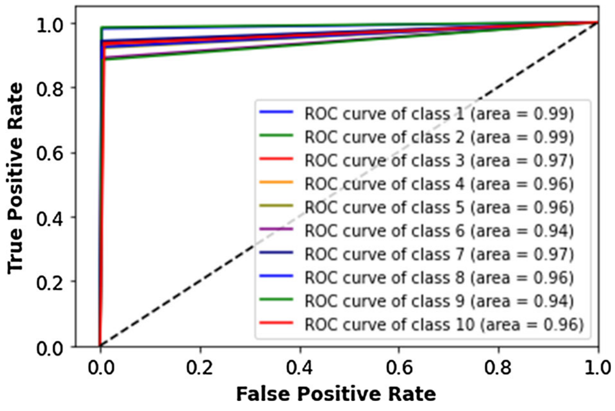


Fig. 5 ROC Curve (MNIST)

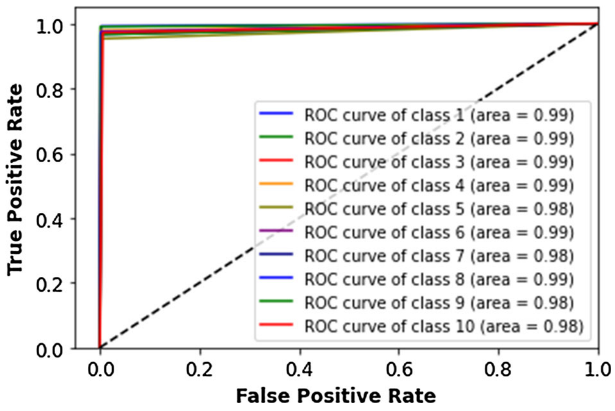


Fig. 6 ROC Curve (MNIST-CNN-HPSGW)

5.3.2 CIFAR results with/without HPSGW

Subsequently the experiment was conducted on CIFAR dataset. During training phase, the model was trained with 50,000 images and tested with 10,000 images. This method results model accuracy 87.6%, sensitivity rate 87% and specificity 88%. Figure 7 represents the true positive rate and false positive rate for each classification labels by ROC Curve.

Next the experiment was conducted on the same CIFAR data set using HPSGW with same set of training and test data. In the training phase the CNN architecture was build based on the parameters which gives best fitness value. The classification accuracy of 99.1%, sensitivity rate is 91% and specificity rate is 91%. Figure 8 shows the positive rate by ROC curve.

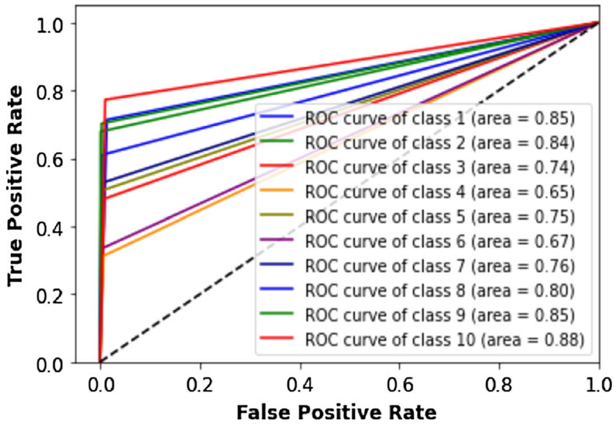


Fig. 7 ROC Curve (CIFAR)

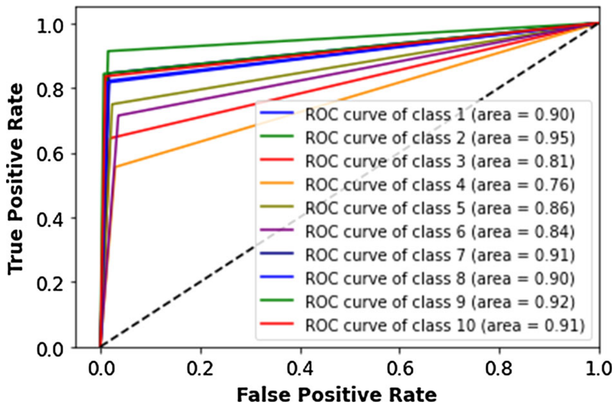


Fig. 8 ROC Curve (CIFAR-CNN-HPSGW)

5.3.3 ICD Result with/without HPSGW

Subsequently the experiment was conducted on ICD dataset. During training phase, the model was trained with 80% training images and testing with 20% test images. This model results accuracy of 95.4%. Figure 9 shows the positive rate of this model by ROC curve.

Next the experiment was conducted the same data set using HPSGW with same set of training and test data. In the training phase the CNN architecture was build based on the parameters which gives best fitness value in 100 iterations (Fig. 10).

Iteration 10 gives the global best score. The classification accuracy of 97.3%, sensitivity rate is 99% and specificity rate is 99%. After the fine tuning of hyperparameters it results the best optimal CNN architecture which consists of 5 batches of layers each layer consists 2 convolution layers and 1 pooling layer for feature extraction and 2 fully connected layers are added at the end for classification. Figure 11 depicts the Accuracy graph during the training and testing phases. Figure 12 illustrates the model loss and Fig. 10 shows the positive rate by ROC Curve. Figure 13 illustrates the layered architecture of optimized CNN.

Figure 14 shows some validation results on ICD Dataset for classifying various dance

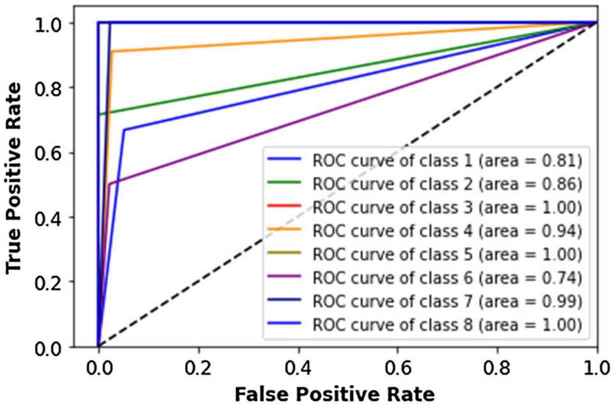


Fig. 9 ROC Curve (ICD)

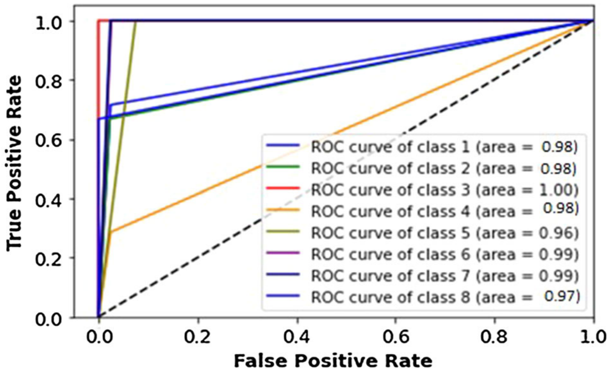


Fig. 10 ROC Curve (ICD-CNN-HPSGW)

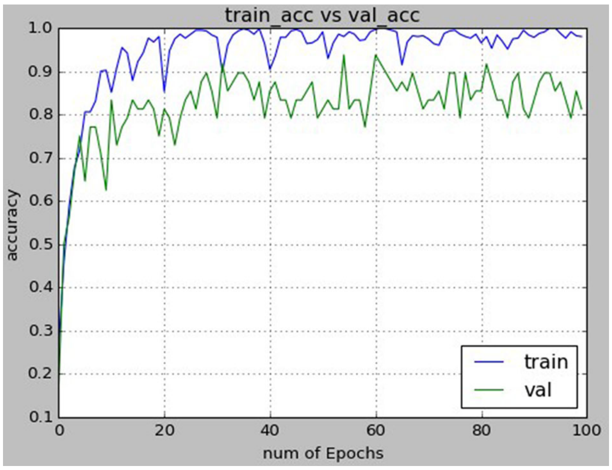


Fig. 11 Accuracy Graph

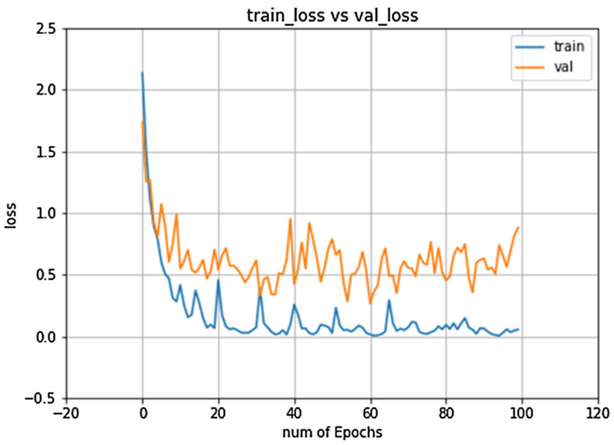
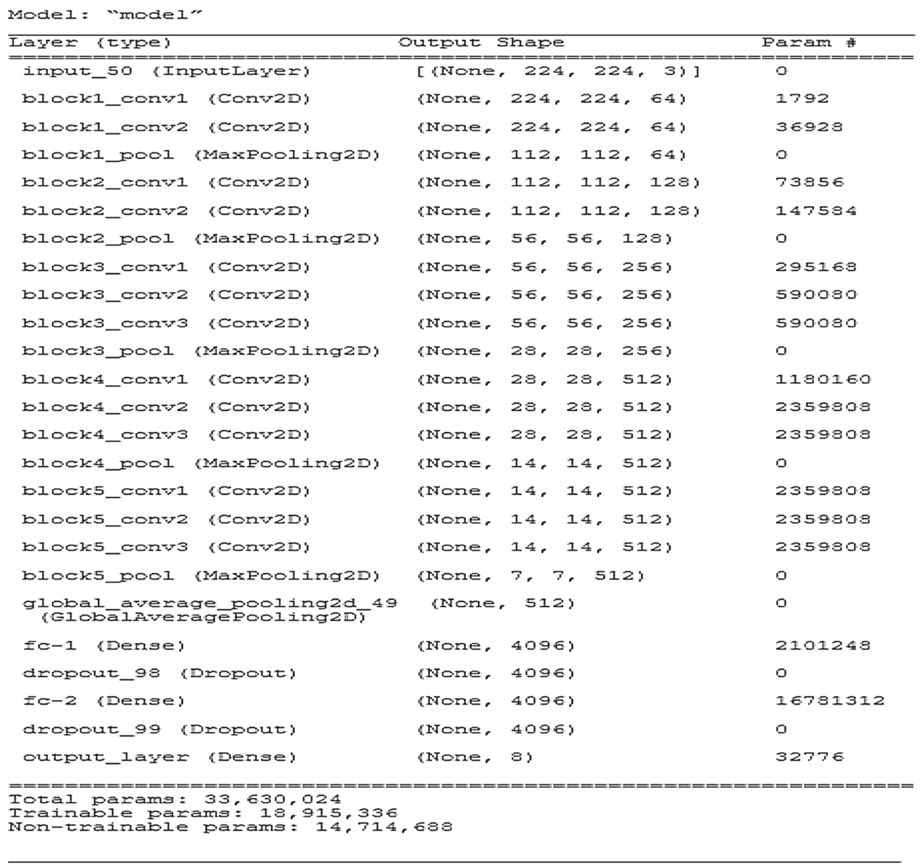


Fig. 12 Loss Graph

forms like Kuchipudi, Manipuri, kathak, Odissi, kathakali, sattriya, Bharatanatyam and kathak. It gives better classification accuracy compared to CNN.

6 Comparative analysis and discussion

In this section the proffered method is compared with the state of art results on different datasets. And also conducted statistical analysis on each dataset.


```
Model: "model"
_________________________________________________________________
Layer (type)                    Output Shape                Param #
=================================================================
 input_50 (InputLayer)          [(None, 224, 224, 3)]       0
 block1_conv1 (Conv2D)          (None, 224, 224, 64)        1792
 block1_conv2 (Conv2D)          (None, 224, 224, 64)        36928
 block1_pool (MaxPooling2D)     (None, 112, 112, 64)        0
 block2_conv1 (Conv2D)          (None, 112, 112, 128)       73856
 block2_conv2 (Conv2D)          (None, 112, 112, 128)       147584
 block2_pool (MaxPooling2D)     (None, 56, 56, 128)         0
 block3_conv1 (Conv2D)          (None, 56, 56, 256)         295168
 block3_conv2 (Conv2D)          (None, 56, 56, 256)         590080
 block3_conv3 (Conv2D)          (None, 56, 56, 256)         590080
 block3_pool (MaxPooling2D)     (None, 28, 28, 256)         0
 block4_conv1 (Conv2D)          (None, 28, 28, 512)         1180160
 block4_conv2 (Conv2D)          (None, 28, 28, 512)         2359808
 block4_conv3 (Conv2D)          (None, 28, 28, 512)         2359808
 block4_pool (MaxPooling2D)     (None, 14, 14, 512)         0
 block5_conv1 (Conv2D)          (None, 14, 14, 512)         2359808
 block5_conv2 (Conv2D)          (None, 14, 14, 512)         2359808
 block5_conv3 (Conv2D)          (None, 14, 14, 512)         2359808
 block5_pool (MaxPooling2D)     (None, 7, 7, 512)           0
 global_average_pooling2d_49     (None, 512)                0
  (GlobalAveragePooling2D)
 fc-1 (Dense)                   (None, 4096)                2101248
 dropout_98 (Dropout)           (None, 4096)                0
 fc-2 (Dense)                   (None, 4096)                16781312
 dropout_99 (Dropout)           (None, 4096)                0
 output_layer (Dense)           (None, 8)                   32776

=================================================================
Total params: 33,630,024
Trainable params: 18,915,336
Non-trainable params: 14,714,688
_________________________________________________________________
```

Fig. 13 CNN-HPSGW Layered Architecture for ICD

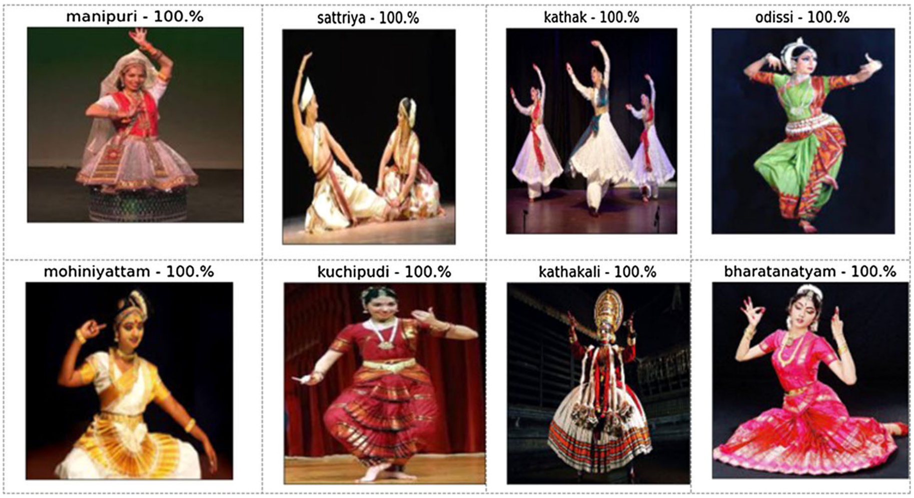


Fig. 14 ICD Classification on Test data using CNN-HPSGW

6.1 Evaluation metrics

The metrics used to evaluate the performance of the proposed model are Accuracy, Sensitivity, Specificity, F1-Score, Cohen/Kappa Score, Precision and Matthew Score. True positive, true negative, false positive, and false negative are represented by TP, TN, FP, and FN, respectively. Figure 15 depicts the performance analysis (PA) on three datasets MNIST, CIFAR and ICD with HPSGW by auto tunning of hyper parameters to get the optimal CNN architecture. It results accuracy score 99.4% on MNIST, 91.1% on CIFAR and ICD results 97.3%. Figure 16 depicts the performance results on three datasets MNIST, CIFAR and ICD without applying auto tunning. It results accuracy score 96.4% on MNIST, 87.6% on CIFAR and ICD results 95.5%. Figure 17 depicts the convergence curve which shows the fitness

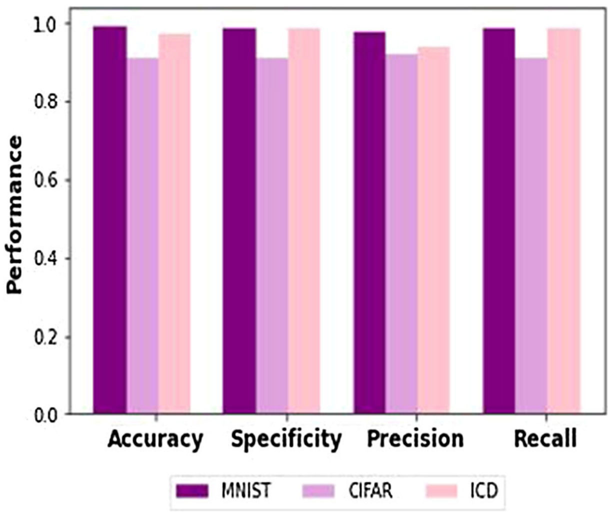


Fig. 15 PA HPSGW

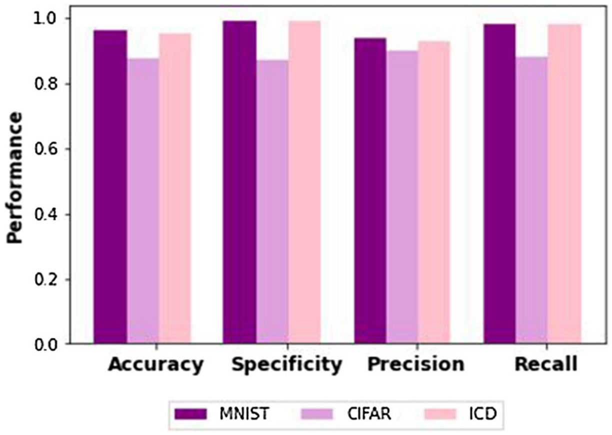


Fig. 16 PA without HPSGW

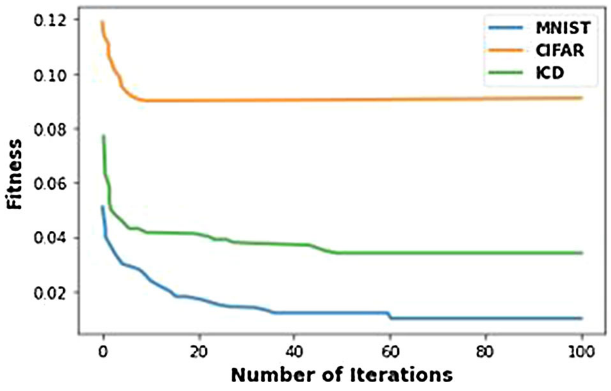


Fig. 17 Convergence Graph

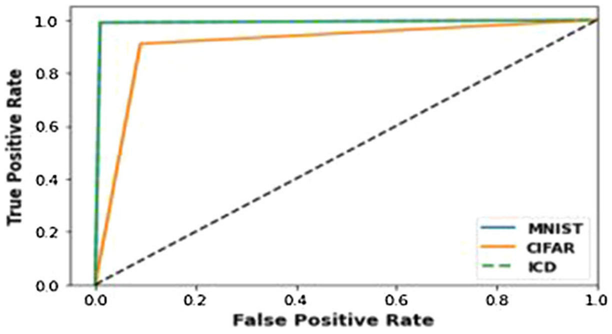


Fig. 18 ROC Curve

value for 100 iterations. The best global score for MNIST dataset, CIFAR Dataset and ICD Dataset is at 10th iteration. Figure 18 shows the ROC curve for 3 datasets using HPSGW.

6.2 Statistical analysis

To find the level of significance between the two proposed architectures and to find which architecture is better a nonparametric test, Wilcoxon rank test is used [41]. It compares the performance of two architectures with and without HPSGW by considering the accuracy values of both the architectures. The process of this test is done as.

- Assume Null Hypothesis(H_0): Both architectures yield the same accuracy that is H_0: $\mu_0 = \mu_1$.
- Alternative Hypothesis(H_1): CNN -HPSGW is greater than CNN-WITHOUTHPSGW that is $\mu_1 > \mu_0$.
- A confidence level of 95% with $\alpha = 0.05$ and n = 13 (Sample Size).
- Find the T^+, T^- and p-value for each dataset if it is smaller than α (0.05) then reject the Null Hypothesis (H_0) and concluded that Optimized CNN is better than CNN without optimized.

From Table 3 we observe that the Wilcoxon Rank test results p-value which is less than

Table 3 Statistical Analysis

Dataset	Statistical test	*p*-value
MNIST	Wilcoxon rank	0.0018
CIFAR	Wilcoxon rank	0.0396
ICD	Wilcoxon rank	0.0457

the 95% of level of significance 0.05 for 3 datasets MNIST, CIFAR, ICD. So, we reject the Null Hypothesis (H_0) and concluded that auto tuned CNN with HPSGW is better than CNN without optimization.

6.3 State-of-art comparison

We compare the results of the optimization methodologies given in this study to the state-of-the-art research, in which CNN models are applied in the ICD dataset, to get further proof about their effectiveness. The results in Table 4 indicate the authors' best classification values, which are explained in detail below.

[38] proposed the "identification and classification of Indian Classical Dance images using Deep Learning Convolution Neural Network (CNN)" which results 78.8% accuracy. [39] proposed a classification of different dance forms into 8 classes is attempted using a deep convolutional neural network (DCNN) model using backbone as ResNet50, which results the classification accuracy of 91.1%. [40] has been proposed a transfer-leaning based CNN Model for ICD classification. This paper used VGG-16 and VGG-19 pre-trained models for ICD classification. VGG-19 results best accuracy of 91.7%. Compared to state-of-art analysis our model CNN-HPSGW performs best results 97.3% for ICD Classification. This optimized CNN improves the accuracy rate by 5.6%.

Table 5. depicts the CNN auto tuning with various metaheuristic algorithms on MNIST dataset. Compared to state of art CNN-HPSGW shows better performance with an accuracy of 99.10%.

Table 6 depicts the CNN auto tuning with various metaheuristic algorithms on CIFAR dataset. Compared to state of art CNN-HPSGW shows better performance with an accuracy of 91.1%

Table 4 State-of-the-Art Comparison ICD

Author Reference	Accuracy (%)
CNN [38]	78.8
DCNN-Resnet50 [39]	91.1
CNN-VGG-19 [40]	91.7
CNN-HPSGW(Ours)	97.3

Table 5 State-of-the-Art Comparison MNIST

Model Reference	Accuracy (%)
CNN-PSO [22]	95.08
CNN-MAA [43]	98.75
Genetic DCNN [10]	99.3
CNN-SA [42]	97.35
CNN-DE [42]	97.32
CNN-HS [42]	96.77
CNN-HPSGW(Ours)	99.4

Table 6 State-of-the-Art Comparison CIFAR

Model	Accuracy (%)
CNN-PSO [44]	80.5
MPSO-CNN [45]	87.4
Genetic DCNN [10]	89.23
CNN-MAA [43]	96.53
CNN-HPSGW(Ours)	91.1

6.4 Computational complexity

The computational COMPLEXITY of the proposed HPSGW algorithm is O(mn^2). Where n is the population size and m is the no. of selected features. Compared to manual tune architectures auto tuned architectures takes less execution time. The time complexity will be reduced.

7 Conclusion and future enhancement

In this work we proffered CNN-HPSGW to optimize the CNN architecture performance and being applied on ICD classification. The proposals covered the no. of hidden layers, the size of the filter used in each hidden layer, the no. of filters, and the batch size. According to the findings of the experiments, the CNN-HPSGWO enhances the network with few parameters. Overall accuracy achieved by three datasets were: for MNIST dataset it results 99.4%, for CIFAR dataset results 91.1% and for ICD dataset 97.3%. The no. of hidden layers, the filter size utilized in each hidden layer, the no. of filters, and the batch size were all optimized in this study. The findings show how important it is to use optimization methods to determine the best parameters for convolutional neural network architectures. Finally concluded that CNN-HPSGW gives better performance than without optimization for any type of classification problems. Compared to earlier methods for ICD Classification the proffered method gives 5.6% improved accuracy. In future study, we will enhance the CNN-HPSGW auto tuning by include some other hyper parameters in the optimization process, such as training size, training rate, regularization rate, and activation functions. Additionally, other computational algorithms such as the monarch butterfly optimization (MBO), earthworm optimization algorithm (EWA), elephant herding optimization (EHO), moth search (MS) algorithm, slime mould algorithm (SMA), hunger games search (HGS), Runge-Kutta optimizer (RUN), colony predation algorithm (CPA), and Harris hawks optimization can be used to improve the CNN architecture (HHO). This approach can be applied for building CNN for Indian classical dances.

Author contributions All authors contributed to the study conception and design. Mrs Challapalli Jhansi Rani has defined the problem of the work and it was verified and refined with the help of other coauthor Dr. Nagaraju Devarakonda. The organization of chapters and design of the proposed work, implementation and results section is carried out by Mrs Challapalli Jhansi Rani and Dr. Nagaraju Devarakonda. All authors read and approved the final manuscript. The authors confirm contribution to the paper as follows: study conception and Design: Mrs.Challapalli Jhansi Rani, Dr. Nagaraju Devarakonda Data Collection: Mrs.Challapalli Jhansi Rani Analysis and Interpretation of Results: Mrs.Challapalli Jhansi Rani Draft Manuscript Preparation: Mrs.Challapalli Jhansi Rani, Dr. Nagaraju Devarakonda All authors reviewed the results and approved the final version of the manuscript.

Funding No funding was received for conducing this study.

Declarations

Conflict of interest On behalf of all authors the corresponding author states that there is no conflict of Interest.

Competing interests The authors declare no competing interests.

Consent to participate All Authors consented for participating in this study.

References

1. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778
2. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks, In: Advances in Neural Information Processing Systems, pp. 1097–1105
3. Hemanth JD, Deperlioglu O, Kose U (2020) An enhanced diabetic retinopathy detection and classification approach using deep convolutional neural network. Neural Comput Appl 32:707–721
4. Li P, Li J, Wang G (2018) Application of convolutional neural network in natural language processing. IEEE Access, pp. 64–70
5. Lee W-Y, Park S-M, Sim K-B (2018) Optimal hyperparameter tuning of convolutional neural networks based on the parameter setting-free harmony search algorithm. Optik 172:359–367
6. Gülcü A, KUs Z (2020) Hyper-parameter selection in convolutional neural networks using microcanonical optimization algorithm. IEEE Access 8:52528–52540
7. Wang B, Sun Y, Xue B, Zhang M (2018) A hybrid differential evolution approach to designing deep convolutional neural networks for image classification. In: Proceedings of the Australasian Joint Conference on Artificial Intelligence, Wellington, New Zealand, 11–14 December 2018; Springer: Cham, Switzerland, pp. 237–250
8. Sun Y, Xue B, Zhang M, Yen GG (2020) Evolving deep convolutional neural networks for image classification. IEEE Trans Evol Comput 24:394–407
9. Yanan S, Yen GG, Yi Z (2019) Evolving unsupervised deep neural networks for learning meaningful representations. IEEE Trans. Evol. Comput. 23:89–103
10. Ma B, Li X, Xia Y, Zhang Y (2020) Autonomous deep learning: a genetic DCNN designer for image classification. Neurocomputing 379:152–216
11. Baldominos A, Saez Y, Isasi P (2018) Evolutionary convolutional neural networks: an application to handwriting recognition. Neurocomputing 283:38–52
12. Fregoso J, Claudia IG, Martinez GE (2021) Optimization of convolutional neural networks architectures using PSO for sign language recognition. Axioms 10:139
13. Mohakud R, Rajashree D (2021) Designing a grey wolf optimization based hyper-parameter optimized convolutional neural network classifier for skin cancer detection.

14. 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
15. 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
16. 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
17. Singh N, Singh SB (2017) Hybrid algorithm of particle swarm optimization and grey wolf optimizer for improving convergence performance. Vol 2017, Article ID 2030489, https://doi.org/10.1155/2017/2030489
18. Sathyanarayana E, Krishna P (2021) Hybrid particle swarm and grey wolf optimization algorithm for IoT intrusion detection system. Int J Intell Eng Syst, 14(4)
19. Goel T, Murugan R, Mirjalili S, Chakrabartty DK (2021) OptCoNet: an optimized convolutional neural network for an automatic diagnosis of COVID-19. Appl Intell 51:1351–1366. https://doi.org/10.1007/978-981-16-6285-058
20. Kumaran N, Vadivel A, Saravana Kumar S (2018) Recognition of human actions using CNN-GWO: a novel modeling of CNN for enhancement of classification performance. Multimed Tools Appl 77:23115–23147
21. Hayder MA, Tony H, Naz EI (2015) Hybrid algorithm for the optimization of training convolutional neural network. IJACSA Vol. 6, No. 10
22. Syulistyo AR, Purnomo DMJ, Rachmadi MF, Wibowo A (2016) Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN) DOI: https://doi.org/10.21609/jiki.v9i1.366
23. Wang GG, Lu M, Dong YQ, Zhao XJ (2016) **Zhao** Self-adaptive extreme learning machine. Neural Comput Appl 27(2):291–303. https://doi.org/10.1007/s00521-015-1874-3
24. Wang Y, Qiao X, Wang GG (2022) Architecture evolution of convolutional neural network using monarch butterfly optimization. J Ambient Intell Humanized Comput. https://doi.org/10.1007/s12652-022-03766-4
25. Yi JH, Wang J, Wang GG (2016) Improved probabilistic neural networks with self-Adaptive strategies for transformer fault diagnosis problem. Adv Mech Eng Jan https://doi.org/10.1177/1687814015624832
26. Cui Z, Xue F, Cai X, Cao Y, Wang G, Chen J (2018) Detection of malicious code variants based on deep learning. IEEE Trans Industr Inf 14:3187–3196. https://doi.org/10.1109/TII.2018.2822680
27. Braik M, Hammouri A, Atwan J, Al-Betar M, Awadallah M (2022) White shark optimizer: a novel bio-inspired meta-heuristic algorithm for global optimization problems. Knowl-Based Syst 243:108457. https://doi.org/10.1016/j.knosys.2022.108457
28. Abdollahzadeh B, Gharehchopogh FS, Mirjalili S (2021) African vultures optimization algorithm: a new nature-inspired metaheuristic algorithm for global optimization problems. Comput Ind Eng 158:107408
29. Abdollahzadeh B, GharehchopoghSeyedali SFM (2021) Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. Int J Intell Syst. https://doi.org/10.1002/int.22535
30. Askari Q, Younas I, Saeed M (2020) Political optimizer: a novel socio-inspired meta-heuristic for global optimization. Knowl-Based Syst 195:10570
31. Chou J-S, Nguyen N-M (2020) FBI inspired meta-optimization. Appl Soft Comput 93:106339. https://doi.org/10.1016/j.asoc.2020.106339
32. Chou JS, Truong DN (2021) A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. Appl Math Comput. https://doi.org/10.1016/j.amc.2020.125535
33. Mohammadi-Balani A, Nayeri MD, Azar A, Taghizadeh-Yazdi M (2021) Golden eagle optimizer: a nature-inspired metaheuristic algorithm. Comput Ind Eng. https://doi.org/10.1016/j.cie.2020.107050
34. Noel MM; Muthiah-Nakarajan V, Geraldine Bessie A, Advait Sanjay T (2021–11–30). A new biologically inspired global optimization algorithm based on firebug reproductive swarming behaviour. Exp Syst Appl. 183: 115408. doi:https://doi.org/10.1016/j.eswa.2021.115408
35. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gen Comput Syst 97:849–872
36. Kennedy J, Eberhart R (1995) Particle swarm optimization," In: Proceedings of the IEEE International Conference on Neural Networks, pp. 1942–1948, Perth, Australia
37. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69(2014):46–61
38. Naik A, Supriya MH (2020) Classification of indian classical dance images using convolution neural network. Int Conf Commun Signal Process (ICCSP) 2020:1245–1249
39. Jain N, Bansal V, Virmani D, Gupta V, Salas-Morera L, Garcia-Hernandez L (2021) An enhanced deep convolutional neural network for classifying indian classical dance forms. Appl Sci 11:6253. https://doi.org/10.3390/app11146253

40. Biswas S, Ghildiyal A, Sharma S (2021) Classification of Indian Dance Forms using Pre-Trained Model-VGG," 2021 Sixth International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2021, pp. 278–282, doi: https://doi.org/10.1109/WiSPNET51692.2021.9419426.
41. Derrac J, García S, Molina D, Herrera F (2011) A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol Comput 1:3–18
42. Rere LM, Fanany MI, Arymurthy A (2016) Metaheuristic algorithms for convolution neural network. Comput Intell Neurosci. https://doi.org/10.1155/2016/1537325
43. Vina A, Rere LM, Mohamad Ivan F, Aniati A (2016) Optimization of convolutional neural network using microcanonical annealing algorithm. https://doi.org/10.1109/ICACSIS.2016.7872787
44. Sinha T, Verma B, Haidar A (2017) Optimization of convolutional neural network parameters for image classification. IEEE Symposium Series on Computational Intelligence (SSCI) 2017:1–7. https://doi.org/10.1109/SSCI.2017.8285338
45. Singh P, Chaudhury S, Panigrahi BK (2021) Hybrid MPSO-CNN: Multi-level Particle Swarm optimized hyperparameters of Convolutional Neural Network,Swarm and Evolutionary Computation, Vol 63,2021,100863,ISSN 2210–6502,https://doi.org/10.1016/j.swevo.2021.100863.

Jhansi Rani Challapalli received her M. Tech Degree in CSE from Guntur Engineering College Affiliated to JNTUK and M. Sc (CS) Degree from G.V.R & S College for Women, Affiliated to ANU. Currently, she is pursuing her Ph. D Degree in School of Computer Science and Engineering, VIT-AP University, Andhra Pradesh, India, as Internal Full Time Research Scholar. She worked as Assistant Professor in the Department of Computer Science and Engineering, KKR&KSR Institute of Technology and Sciences, Vinjanampadu, Vatticherukuru, Guntur, Andhra Pradesh, India. Her research areas are machine learning, deep learning, optimization techniques and video analysis. She has 15+ years of experience in teaching, and 7 technical paper publications.

Nagaraju Devarakonda is working as Associate Professor in the School of Computing Science and Engineering at VIT-AP, India. He has got totally 16 years of teaching experience in various engineering institutions in Andhra Pradesh. His research interests include data mining, machine learning, soft computing and optimization techniques.