PBO: A New Pollination Based Optimization Algorithm and its Application to Paddy Disease Detection

IJIMSR, Vol. 1, No. 1, (2023) 23.1.1.005

Shakti Kumar
SDGI Global University
Ghaziabad 201015, UP, India

Ajay Singh
Allgeier Engineering GmbH
München, Germany

Rahul Sharma
Government Degree College
Kathua, J&K, India

***Amar Singh**
School of Computer Applications
Lovely Professional University
Phagwara-144411, Jalandhar Punjab, India
amar.23318@lpu.co.in

***Corresponding author**

Received 10th August 2023; Accepted 1st September 2023
KEYWORDS: ACO, BBO, BAT, DSR, SVM, GA-CNN, BBBC-CNN, PBO, Plant Disease Detection

PBO: A New Pollination Based Optimization Algorithm and its Application to Paddy Disease Detection

Shakti Kumar
SDGI Global University
Ghaziabad 201015, UP, India

Ajay Singh
Allgeier Engineering GmbH
München, Germany

Rahul Sharma
Government Degree College
Kathua, J&K, India

Amar Singh
School of Computer Applications
Lovely Professional University
Phagwara-144411, Jalandhar Punjab, India
amar.23318@lpu.co.in

ABSTRACT

This paper presents a pollination based optimization (PBO) algorithm. PBO is a bio-inspired, multi-population global optimization algorithm capable of generating high accuracy solutions to complex problems. The plants have been observed to optimize their resource expenditure on fragrance, floral display, nectar production and pollen to attract pollinating agents such as insects, bees, flies, bats, birds, etc. Subject to pollination success, plants increase or decrease their total resource cost on fragrance, superior nectar content, pollen and floral display. If the reproductive success is better, plants decrease their investment. In case the reproductive success is below average, plants increase their investment on resources affecting pollination. This increases the number of pollinators and their re-visitation causing the reproductive success to go up. The proposed PBO algorithm was evaluated on the 80 test functions of CEC 2021 test suite, and the performance was compared with 8 recent algorithms. The algorithm performed exceptionally well, leading in 41 of the 80 functions of the test bench. The paper further, demonstrates the application of the proposed algorithm to evolve an optimized CNN architecture for the paddy plant disease detection from the paddy leaf dataset. The paddy leaf dataset has 5932 infected images indicating various diseases. The PBO based approach with 99.37% accuracy outperformed KNN, SVM, Decision Tree, Random Forest, GA-CNN and BBBC-CNN based algorithms.

KEYWORDS: ACO, BBO, BAT, DSR, SVM, GA-CNN, BBBC-CNN, PBO, Plant Disease Detection

1. BACKGROUND

OPTIMIZATION is an instinct inherent in human beings, animals, plants and trees. In their search for food, ants find the shortest path between their colony and location of food, through pheromone laying [1]. According to biogeography, as survival tactics, birds maintain an optimal number on an island [2], fireflies optimize their hunt and reproductive success through their flashing behavior [3], honey bees waggle dance to communicate the distance and direction of flower patches to other bees of the hive to maximize their nectar collection [4-5]. The human beings have a tendency to create a mathematical discipline for anything that is abstract, significant and general. Searching and optimization are so significant to us that these have become one of the important and established branches of the mathematics. This paper proposes a pollination-based global optimization algorithm named PBO. In the pollination process pollen is transferred from the anther (male part), to the stigma (reproductive organ). There are two forms of pollination namely biotic and abiotic. In the biotic pollination insects and bees act as primary pollen carriers. Biotic pollination accounts for approximately 90% of the total pollination. On the other hand, water and wind act as pollen carriers in abiotic pollination.

For plants, a cost-based pollination model was proposed by Thakkar et al. [6]. This model was further improved by B. Sriram et al. [7]. The above model does not adequately represent the inherent randomness of the pollination process. A careful look at the pollination process reveals that neither it is totally random, nor it is purely deterministic. This process is a guided one with considerable randomness. In the pollination process pollinators arrive randomly, and their repeated visits are guided by plant investment in the nectar, floral display, pollen and fragrance. Another pollination-based algorithm namely FPA was proposed by Yang She et al. [8].

This paper presents a new pollination success based, bio-inspired, multi-population, optimization algorithm named PBO, that is conceptually very different from FPA. Section 2 builds the PBO concept. Section 3 develops the formal algorithm based on the inspiration derived from the plants. In the Section 4 we evaluate and discuss the performance of PBO on the 80 functions of the CEC 2021 test suite and compares it with 8 of the leading algorithms. Section 5 describes its application to the identification of an optimized CNN for the disease detection in paddy plants. Section 6 concludes the paper.

2. PBO CONCEPT

For plant species to survive, these need to produce their next generation by producing their seeds via pollination. Pollinators forage for pollen and nectar. The plant flowers through their fragrance and floral display attract pollinators to provide them with quality nectar and pollen. These pollinators then carry pollen from one flower to other and thus, the pollination begins. This interdependence of the two on each other leads to successful pollination. The pollination process gets completed in a specific time frame during the pollination season. The relationship between pollination success and resource control could be explained as follows:

(a) If the pollination success is proceeding at the normal rate, the plants invest average resource on all cost factors.

(b) If the pollination success is observed to be below (above) normal rate, plants increase (decrease) the investment in different components (floral display, fragrance, pollen and nectar) randomly, to attain optimal pollination with minimal total investment/cost. Increased investment on resources/ cost components results into attracting a greater number of pollinators with increased visitation leading to successful pollination with high probability.

© If plants allow maximum expenditure/ investment for every cost component i.e., floral display in terms of brightness of flowers, nectar sweetness, fragrance and pollen quantity then plants are likely to achieve maximal pollination success with high probability at the highest cost in that particular season.

For the pollination process to be successful, pollinators need to be attracted and they must repeatedly visit flower patches. Plants achieve this by investing their resources in four major components namely floral display, flower fragrance, pollen and nectar. These four investment/cost components are as depicted in Table 1; form the decision variables for the optimization model. Though, pollination success is a function of these variables as these aid in luring the pollinators yet the visitation by the pollinators is random. Hence, the pollination success has a lot of randomness. Thus, an investment vector consists of 'n' elements/ decision variables. For a plant, investment vector consists of above said 4 components. The count for decision variables is problem-specific and shall vary from problem to problem.

In the PBO, patches represent populations of candidate solutions, with each patch consisting of a fixed number of flowering plants. Each plant has an associated investment vector 'I' consisting of investment/cost components. As stated earlier each of the investment component is a decision variable influencing the total investment as well as pollination success for the plant. The fitness of a plant (pollination success) is the function of investment vector.

Table 1: Formulation of an Investment Vector

No. of investment/cost components = No. of decision variables Display Cost = Sum of display costs of all the flowers of a plant Fragrance Cost = Sum of Fragrance costs of all the flowers of a plant Nectar Cost = Sum of Nectar Costs of all the flowers of a plant					
	I = Investment Vector = One individual				Remarks
Investment vector of Plant No. 1 = I_1	x_{11}	x_{12}	x_{13}	x_{14}	Flowers of "P" Plants constitute One Patch or One Population
Investment vector of Plant No. 2 = I_2	x_{21}	x_{22}	x_{23}	x_{24}	
---------	---	---	---	---	
Investment vector of Plant No. p = I_p	x_{p1}	x_{p2}	x_{p3}	x_{p4}	
The fitness function and the total the cost of an investment vector "I" are the functions of its decision variables					

Let for an investment vector I = {x1, x2, …, xn} pollination success 'S' be defined as given below:

$S = f(I)$ (1)

The optimization problem can now be stated as follows:

Search a specific vector I* = {x1, x2, …, xn} from amongst all the possible candidate solutions for which

$S^* = f(I^*)$

is the optimal value of 'S';

subject to the constraint that:

Bounds on decision variables are not violated.

In the beginning algorithm creates 'N' patches, each consisting of 'p' investment vectors (candidate solutions). Each investment vector consists of 'n' investment components or decision variables. All investment vectors are created randomly, respecting the bounds.

As soon as the initial set of 'N' patches is created, algorithm evaluate pollination success (fitness) of each of the plants. From amongst all the patches of investment vectors we record the global best investment vector that yields the best level of pollination.

The algorithm generates new investment vectors for every vector of the current patch as given in Algorithm 1. These are then subjected to the combination and mutation operations. Collective pollination behavior of all the patches is modeled by 'Combination' operation. It has been observed that whenever one of the flowering patches attains a specific level of success, simultaneously all of its neighboring patches also achieve almost the same level of pollination success. The algorithm models this process by moving each of the decision variables of the investment vectors of a patch towards global best investment vector. For good results, this movement is performed with a high probability between 0.7 to 0.95. When the PBO begins search in the current patch we call it a local search. It continues with the local search for every patch. After a given fraction of the maximum number of iterations the algorithm incorporates the global search; moving towards global optima for a limited fraction of iterations e.g. let us say the total number of iterations (seasons) is max_iterations, then up to the first 20% of max_iterations we do not apply combination operator (only local search). From 20% to 25% of max iterations, we apply the combination operator as follows:

$$I_{i,j} = (I_{i,j} + I_{gbest_{1,j}})/2 \quad \ldots (2)$$

Here, subscript 'j', indicates jth decision variable of ith candidate solution.

The algorithm again carries out a local search without a combination operation from iteration number greater than 25% of the max_iterations to iteration number less than or equal to 45% of the max_iterations; between the iteration number greater than 45% to 50% of the max_iterations, algorithm carries out global search using a combination operation. This cycle of local search followed by global search is repeated.

The algorithm combines the two sets of patches, i.e., the current patch and the newly created patch. This combined patch has twice the number of investment vectors. Constraints/ bound violation if any are checked and corrected. Following this step algorithm evaluates the fitness (pollination success) of each of the investment vectors of the combined patches and retains each patch with 'p' best fit investment vectors. The algorithm then updates global best I_{gbest} (if needed) and records its corresponding fitness values.

Since, the pollination is a seasonal process, the algorithm runs for a given number of seasons (iterations). On meeting the termination criterion, the algorithm stops with I_{gbest} as the optimal solution vector and the corresponding fitness value as the optimal fitness value for the given total cost.

Natural calamities such as abnormal temperatures, storms, rains or damage to plants due to outbreak of certain diseases may affect the pollination success adversely. PBO models such effects using a 'mutation' operator that is similar to the one used in GAs. Mutation operator is applied with a low probability on each of the components of all the investment vectors

Fig 1: Algorithm 1- Pollination Based Optimization (PBO) Algorithm

```
begin
    for patch_num = 1 : N do
        randomly generate a candidate solution matrix P_current (patch_number) of size p×n; respecting the
        bounds;
        evaluate pollination success(fitness) of all the investment vectors of "P_current (patch_number)"
    end for
    record global best investment vector I_gbest and its fitness value
    P_current = pop.patch(pach_num).iv
    α_patch_num = α0

    while season < max_seasons do
        for patch_num = 1: N do
            α_patch_num = α_(patch_num-1)(1 − (patch_num − 1) × (0.01))                    … (3)
            for j = 1: p do
              for k = 1: n do
                change(j,k) = [(−α_patch_num) + (2 * α_patch_num) * (rand(1))] * (1 − (season / max_seasons)) … (4)
                P_new(j, k) = P_current(j, k) + change(j, k)      … (5)
              end
            end
            if global_flag == true
            (/*with a given probability Pc, combine each decision variable of P_new with the corresponding
            element of global best investment vector as follows/*)
                for j = 1: p do
                  for k = 1: n (number of cost components) do
                      generate a random number "r" between 0 to 1
                        if r < Pc
                              P_new(j, k) = [{( P_new(j, k) + I_gbest (1, k) }/2]
                        endif
                  end for
                 end for
            end if
            with a given probability Pm, mutate cost components of each of the P_new as follows:
            for j = 1: p do
                for k = 1: n do
                    generate a random number "r" between 0 to 1.
                    if r < Pm
                       Max possible value of (P_new (j, k) − current value of P_new (j, k))
                    endif
                 end
            end

            combine matrix "P_current" and "P_new" to another matrix "P_combined" with "2p" rows
            check bounds violations of cost component of "P_combined"; correct if needed
            evaluate pollination success (fitness) of all investment vectors of (rows) "P_combined"
            select best "p" investment vectors from "P_combined" and save these in "P_new2"
            update global best investment vector I_gbest and its fitness value; if needed
            pop.patch(patch_num).iv = P_new2

        end for patch_num
        season = season + 1
    end while
    display global best investment vector I_gbest and its fitness value
end
```

3. PROPOSED ALGORITHM

A. Nomenclature

α0 :	A problem specific constant, usually between 1 to 100
Patch :	Set of flowering plants
S:	Pollination success (fitness) corresponding to an investment vector
N:	Total number of patches
n:	Number of cost components of an Investment Vector (number of decision variables)
Pcurrent:	Matrix of size p×n of investment vectors corresponding to current Patch
p:	Number of investment vectors. It equals number of plants in the patch
Pc:	Combination probability (0.7 to 0.95)
Pm:	Mutation probability (usually less than 0.1)
global_flag:	is set for a given number of iterations (about 5%) after a preset amount of iterations for local search (about 20% of maximum iterations)
max_seasons:	Number of maximum Seasons for which algorithm should run (termination criterion)
I_{gbest} :	Optimally fit investment vector of all the populations evaluated so far

The proposed algorithm is given in figure 1.

4. PERFORMANCE OF THE PROPOSED ALGORITHM

To validate the effectiveness of the PBO we implemented it in MATLAB. We evaluated its performance on 80 functions of the CEC-2021 test suite, using a core i7@2.8 GHz with a 16 GB RAM-based laptop operating on Windows-11 platform. We compared the performance of PBO with 8 other leading algorithms namely L-SHADE-OrdRW [10], MadDE [11], RB_IPOP_CMAES_PPMF [12], NL-SHADE-RSP [13], MLS-LSHADE [14], DEDMNA [15], J21 [16] and SOMA-CLP [17]. For performance analysis, we considered all functions with 10 dimensions. 25 trial runs were conducted for each of the test functions. Mean error of the 25 runs was used as the comparison metric. We evaluated the performance of all the 9 algorithms. Function wise performance of all the competing algorithms is placed in Table 2.

Comparative performance of all the 9 algorithms, including the proposed PBO algorithm, is presented in Table 3. The comparative performance is in terms of the number of functions on which an algorithm delivered the best-performance. Looking at Table 3 one could observe that amongst the 9 competing algorithms, PBO algorithm tops the chart, achieving the best performance in 41 of the 80 functions of the CEC-2021 test suite. In 3 of these 41 functions none of the competing algorithms could match the performance record of PBO. For other 38 of the 41 functions, its performance is the best but it was also matched by a few other algorithms.

Algorithm L-SHADE-OrdRW [10] bags the second place with the best performance on the 41 of the test functions. Out of these 41 functions, it gave unique best performance only on 2 of the 80 benchmark functions, which was lesser than PBO.

Table 3: Comparative Performance on CEC-2021 Test Suite

Algorithm	A	B	C	Rank
PBO	41	3	38	1
L-SHADE-OrdRW	41	2	39	2
MadDE	40	2	38	3
RB_IPOP_CMAES_PPMF	31	8	23	4
NL-SHADE-RSP	21	4	17	5
MLS-LSHADE	20	0	20	6
DEDMNA	18	3	15	7
J21	15	1	14	8
SOMA-CLP	7	0	7	9

A = Number of functions for which the best performance is recorded, B = Number. of functions for which (Unequalled) best performance was recorded, C = No. of functions for which the best performance is recorded but is equalled by other competing algorithms also.

MadDE [11] stands at the number 3 position by achieving the best performance over 40 out of 80 functions. Out of these 40 functions MadDE algorithm delivered an unmatched performance for the 2 functions only and the best performance over 38 other functions, those were matched by the performance of some other algorithms also. RB_IPOP_CMAES_PPMF was placed at the number 4 position as it gave the best performance over 31 benchmark functions.

5. PBO APPLICATION TO RICE DISEASE DETECTION

Rice is a staple food crop for a large portion of the world's population and plays a vital role in global food security. However, rice plants are vulnerable to a number of diseases, which can significantly reduce crop yield and cause farmers to lose money. Early disease detection limits the spread of the disease and boosts agricultural productivity. Manual examination and laboratory testing on plant samples (using techniques including chemical analyses, genetic analyses, and biochemical approaches) are the traditional ways of detecting diseases. Applying computer vision algorithms to diagnose plant diseases has the potential to provide faster and precise diagnosis with fewer computing resources, hence reducing the spread of diseases and boosting crop production. Due to the complexity of real-world datasets, automated computer-based disease diagnosis is a challenging and complex task. As a result, effective image classification methods based on soft computing are required for the accurate and scalable identification of plant diseases.

Selecting the optimal hyperparameter combination in Convolution Neural Networks (CNNs) can be difficult because so many distinct combinations could potentially be used. It is difficult to guarantee that a specific set of hyperparameters would produce the best results for a task. The other option is trial and error selection. But the trial and error selection of hyperparameters is time and resource consuming. Consequently, there arises a need for an automated system that evolves the optimal CNN hyperparameters for a given situation. Additionally, when building a CNN, one must select the number and structure of layers, the number of filters, the size of the filters, the stride, padding, the pooling type, the activation function, the number of neurons in the fully connected layers, the optimizer, etc. The network architecture and hyperparameters need to be carefully designed in order for it to learn the features of the training data successfully. This section presents a PBO algorithm based approach to evolve an optimized CNN model with optimal hyper-parameters. The proposed method successfully identified a lightweight CNN model from the given training dataset [18].

5.1 Proposed Approach

As referred to CNNs, hyper-parameters are the variables those control as to how the network is trained and how its structure is set up. CNN architecture makes use of a large number of hyper-parameters [19-20]. Domain/Technical knowledge is necessary to select the optimal hyper-parameters manually. It is a tedious, time-consuming technique that relies on trial and error [21]. This section demonstrates the application of a PBO based approach to evolve an optimal number of convolution layers of the CNN model along with the optimal hyper-parameters. This has shortened the design time drastically. The approach begins with a single convolution layer skeleton CNN as shown in Figure 2, and randomly generates populations of investment vectors, where each investment vector represents CNN hyper-parameters. The fitness function evaluates the CNN model's test accuracy for the rice-plant disease classification.

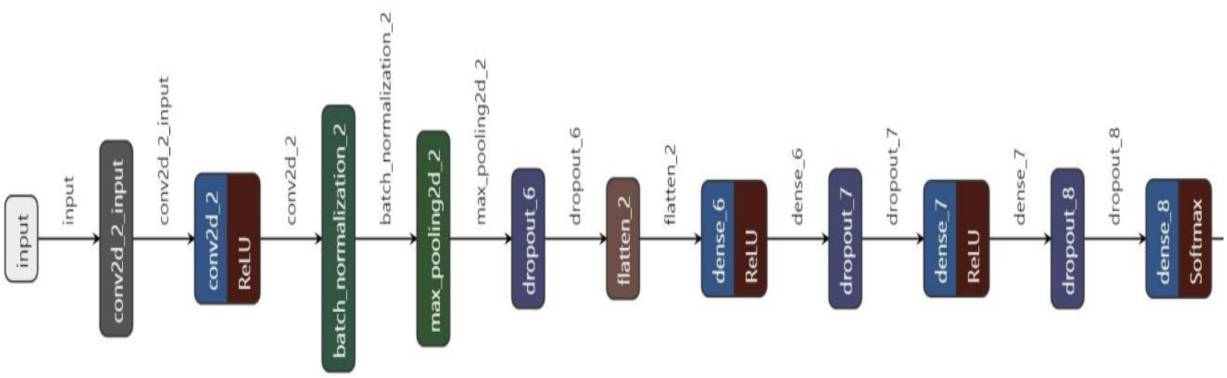

Fig 2. CNN model structure used for evolution

Table 4 displays the ranges of hyper-parameters considered for optimization.

Parameter	Range
Convolution Layers	1-10
Filters	1-64
Filter Size	1-10
Neurons in fully connected layers	32-1024
Batch size	8-512 (multiple of 2)
Epochs	1-20
CNN model	SGD, Adadelta, Adam, Adagrad, RMSprop, Ftrl Nadam, and Adamax.

The proposed approach automatically adds a convolution layer in the CNN model when necessary. With the addition of a new hidden layer, the number of hyper-parameters increases. As a result, a variable-length investment vector encoding scheme is employed. The hyper-parameters are encoded in the investment vector, as shown in Figure 3.

1	No. of filters of C_1	Filter size of C_1	Neurons in FC Layer 1	Neurons in FC Layer 2	Batch Size	Epochs	Model Optimizer

2	No. of filters of C_1	Filter size of C_1	No. of filters of C_2	Filter size of C_2	Neurons in FC Layer 1	Neurons in FC Layer 2	Batch Size	Epochs	Model Optimizer

3	No. of filters of C_1	Filter size of C_1	No. of filters of C_2	Filter size of C_2	No. of filters of C_3	Filter size of C_3	Neurons in FC Layer 1	Neurons in FC Layer 2	Batch Size	Epochs	Model Optimizer

n	No. of filters of C_1	Filter size of C_1	...	...	No. of filters of C_n	Filter size of C_n	Neurons in FC Layer 1	Neurons in FC Layer 2	Batch Size	Epochs	Model Optimizer

Fig. 3. Variable length investment vector structure used for the optimization

The investment vector structure represents the number of filters, convolutional layers, filter size, neurons in fully connected layers, batch size, epochs, and optimizer to be applied. The classification accuracy for each of the individual is then obtained. The CNN architecture with the best accuracy is considered the fittest. Our objective is to evolve the best-performing CNN architecture along with tuned hyper-parameters.

Figure 4 shows the proposed approach to evolve CNN architecture. It begins with randomly generated 'N' patches of investment vectors (candidate solutions), each of size 'p×n'. As given in the algorithm 1, it evaluates the fitness of each individual of every population using the CNN model. Thereafter, current population Pcurrent is modified to obtain 'N' new populations (P_{new}). Under stated condition and with a given probability Pc, P_{new} is combined with global best investment vector. With the given probability P_m, The current population is then mutated. The algorithm then checks the bound violation for each decision variable of every population of P_{new}. The algorithm combines the P_{new} with $P_{current}$, evaluates the combined population and selects best fit 'p' investment vectors for each of the 'N' populations as P_{new2}. The global best investment vector is continually updated as and when needed. P_{new2} is then saved as current patch $P_{current}$. Finally, if the stopping criteria are satisfied, the proposed approach outputs the structure of CNN model along with the optimized hyper-parameters represented by the global best. If the stopping criteria are not met and the given number of iterations are not over the algorithm goes for next iteration; if the number of iterations are over but the condition of maximum limit of layers is yet not satisfied, we modify the CNN architecture by adding a new convolution layer, reset iteration count to zero, and generate new 'N' populations for the evolution of newly obtained CNN architecture.

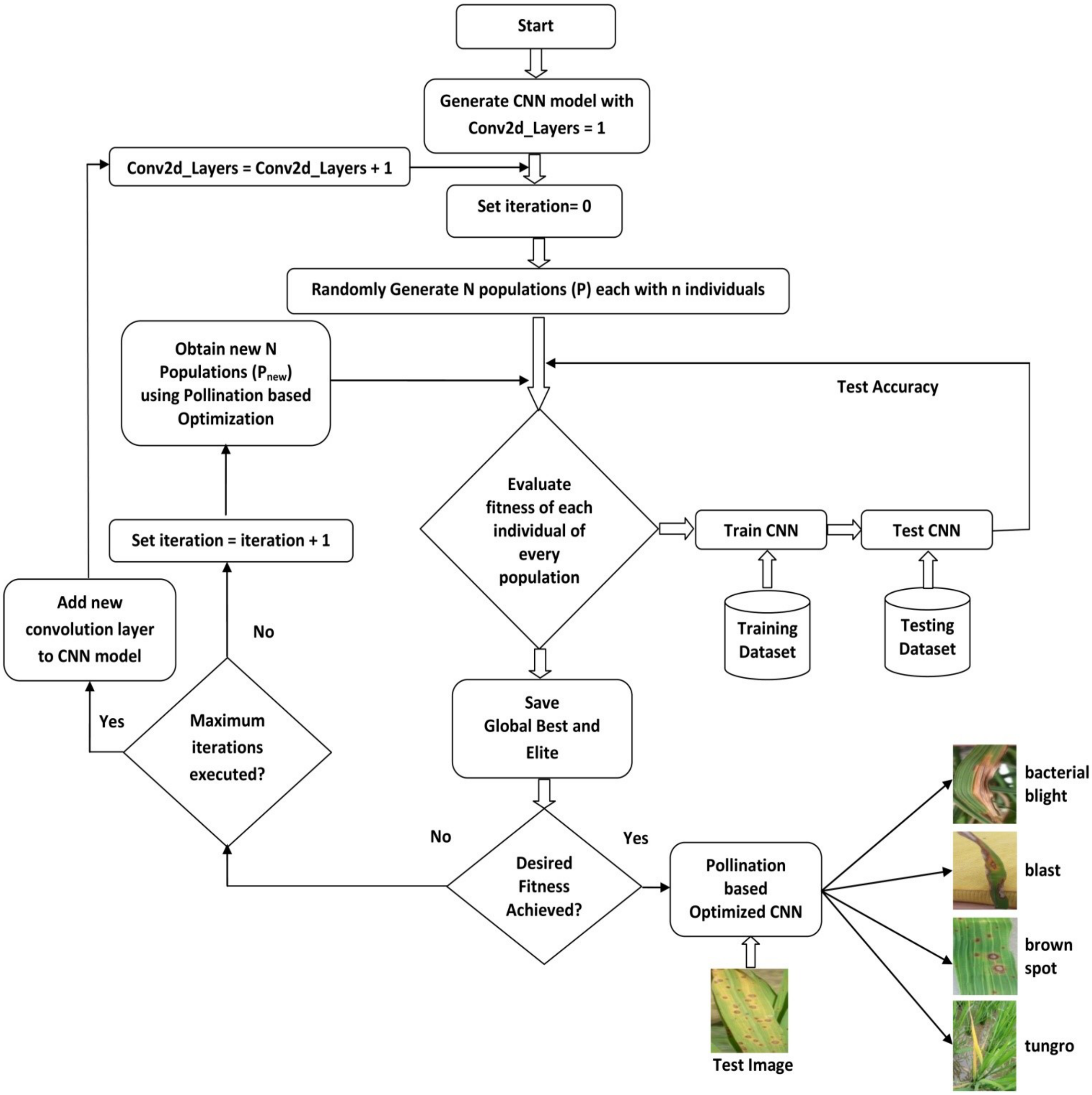

Fig. 4. PBO based approach for evolving CNN model and hyper-parameters

The proposed approach's termination conditions are greater than 98% classification accuracy or no gain in model accuracy with a new convolution layer.

5.2 Results and Discussion

As shown in Fig. 4, we first optimize single convolution (Conv2d) layer CNN using pollination-based optimization. The best accuracy of 94.625% was recorded while executing the 29th iteration. The stopping criteria were unsatisfied, so a new hidden (Conv2d) layer was included in the architecture of CNN, and the new optimization cycle began. A similar approach was applied to CNNs with three and four convolution layers. Figure 5 presents the record of the algorithm progress for the CNN with three layers. The plot shows generation (seasons) versus accuracy values across the iterations. With three convolutional layers, the CNN model achieved an accuracy of 99.37%. Thus, The PBO Based approach successfully evolved a three Conv2d layer CNN architecture and optimal hyper-parameters for classification of rice plant disease.

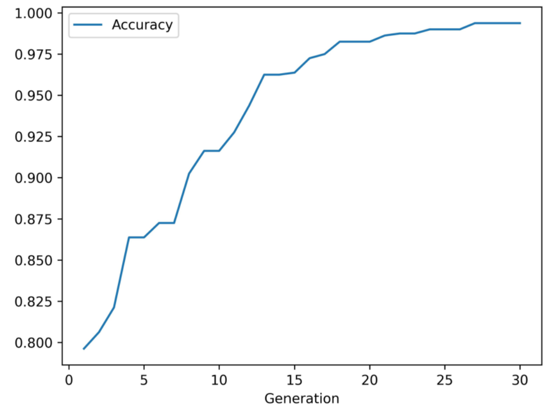

Fig. 5. Generations versus Accuracy (seasons) three-layer CNN

Table 5. Different layers of CNN optimized using PBO

Conv2d layers in CNN	Accuracy
1	94.625%
2	96.875%
3	99.375%

As shown in Table 5 and Figure 5, proposed PBO approach achieved 99.375% accuracy. We also observed that the CNN architecture with 4 convolution layers produced 97.625% accuracy. The optimal three-layer CNN model hyperparameters are presented in Figure 6.

No. of Convolution Layers	No. of filters of C_1	Filter Size of C_1	No. of filters of C_2	Filter Size of C_2	No. of filters of C_3	Filter Size of C_3	Neurons in FC Layer 1	Neurons in FC Layer 2	Batch Size	Epochs	Model Optimizer
03	45	1	18	1	64	10	586	838	206	20	Adam

Fig. 6. Pollination-based optimized CNN hyper-parameter

We evaluated the performance of proposed approach using confusion matrix. The performance results of the PBO based approach are shown in Figure 7.

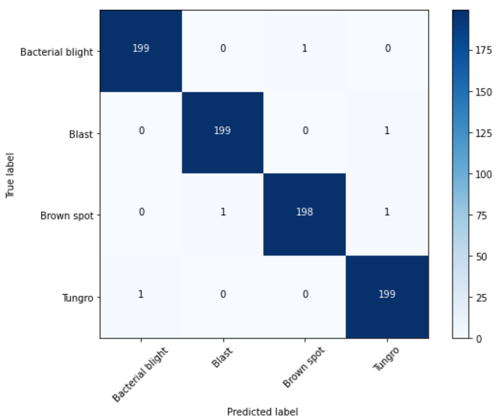

Fig. 7. Confusion Matrix of pollination based Optimized CNN

The confusion matrix yielded true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values. The performance of the disease detection approach is validated using the accuracy, sensitivity, specificity, precision, and F1-score performance measures.

A comparison of the proposed approach with other cutting-edge image classification techniques, such as a genetic algorithm (GA)-based CNN, and a big-bang, big-crunch optimised CNN is shown in Table 6. The PBO based optimized CNN performed better than other classifiers.

Table 6. Performance Comparison of Proposed approach with existing image classification approaches

Model	Accuracy	Precision	Recall	F1-Score
SVM	92.54%	92.65	92.69	92.6
KNN	70.30%	73.56	70.77	69.76
Decision Tree	88.51%	88.36	88.08	88.17
Random Forest	95.3%	95.82	95.92	95.86
GA-CNN	96.37	96.74	96.88	96.72
BB-BC CNN	98.70%	98.701	98.75	98.70
PBO-CNN	99.375	99.37	99.375	99.374

6. CONCLUSIONS

This paper proposed a novel multi-population based, bio-inspired, global optimization algorithm called PBO Algorithm. The cost optimization behaviour of flowering plants inspired the algorithm. Plants optimise resource spending on pollen production, floral display, floral fragrance, and nectar production based on pollination success. The performance of the PBO algorithm was tested on 10-dimensional 80 functions of the CEC 2021 test suite. We compared the performance of PBO with the 8 existing algorithms. The proposed PBO algorithm performed best on 41 of the 80 functions of the CEC-2021 test bench, followed by L-SHADE-OrdRW. MAdDE ranked third with best performance in 40 functions.

Further, we tested the PBO algorithm on rice plant disease detection problem. The PBO algorithm was applied to evolve the structure of CNN from the training dataset. We observed that the PBO performed extremely well. The proposed PBO-CNN approach efficiently identified rice diseases. We compared the performance of the proposed approach with the existing 8 machine learning approaches. The comparison results show that the proposed PBO-CNN-based approach outperformed all the other competing machine learning based approaches including GA-CNN and BBBC-CNN approaches.

Table 2: Performance of Proposed algorithm on CEC-2021 Benchmark Functions

Function	**F1**	**F2**	**F3**	**F4**	**F5**	**F6**	**F7**	**F8**	**F9**	**F10**
					Basic					
DEDMNA	0.0000000	0.0000000	2.1800000	0.1280000	0.0000000	0.0035600	0.0005680	0.0000000	0.0000000	48.0000000
MadDE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
RB_IPOP_CMAES_PPMF	0.0000000	0.2670000	9.2300000	1.0200000	37.8000000	1.4300000	7.5000000	0.0000000	0.0000002	48.0000000
J21	0.0000000	0.0000000	5.6300000	0.2430000	0.0000000	0.0340000	0.0079700	0.0000000	0.0000000	46.4000000
NL-SHADE-RSP	0.0000000	0.0000000	0.0000000	0.0143000	0.0000000	0.0068700	0.0013800	0.0000000	0.0000000	0.0019300
SOMA-CLP	0.0000000	0.1040000	361000000.0000000	0.3600000	0.0000000	0.0311000	0.0021500	0.0000000	0.0000000	771.0000000
MLS-LSHADE	0.0000000	0.0000000	2.2500000	0.0065800	0.0000000	0.0018300	0.0000000	0.0000000	0.0000000	0.0069200
L-SHADE-OrdRW	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000041
PBO	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
					Bias					
DEDMNA	0.0000000	15.0000000	9.8200000	0.4420000	3.5800000	0.3740000	0.1570000	3.6200000	0.0000000	51.4000000
MadDE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000087	0.0000000	0.0000000	0.0000000	0.0000000
RB_IPOP_CMAES_PPMF	0.0000000	0.3060000	8.8700000	1.0100000	34.9000000	1.2000000	14.5000000	0.0000000	0.0000002	66.9000000
J21	0.0000000	16.2000000	11.6000000	0.8440000	4.3500000	1.3200000	0.3530000	15.9000000	0.0000000	51.6000000
NL-SHADE-RSP	0.0000000	5.4600000	5.0500000	0.3830000	3.3100000	0.4190000	0.1930000	42.4000000	0.0000000	48.1000000
SOMA-CLP	0.0000001	1610000.0000000	1600.0000000	0.9940000	71600000.0000000	0.4520000	0.6990000	18.3000000	0.0000000	52.7000000
MLS-LSHADE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000008	0.0000000	0.0000000	0.0000000	0.0044800
L-SHADE-OrdRW	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000019
PBO	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	51.8000000
					Shift					
DEDMNA	0.0000000	0.0000000	4.8100000	0.1560000	0.0000000	0.0053100	0.0005100	14.0000000	86.7000000	373.0000000
MadDE	0.0000000	0.0000000	10.9000000	0.1880000	0.0000000	0.0162000	0.0014200	87.9000000	93.3000000	400.0000000
RB_IPOP_CMAES_PPMF	0.0000000	275.0000000	11.2000000	1.0600000	125.0000000	50.6000000	30.4000000	97.2000000	256.0000000	400.0000000
J21	0.0000000	0.0020800	10.2000000	0.2540000	0.0000000	0.0254000	0.0056200	0.0000000	110.0000000	363.0000000
NL-SHADE-RSP	0.0000000	0.0000000	10.2000000	0.0942000	0.0000000	0.0075700	0.0020100	0.5450000	80.1000000	390.0000000
SOMA-CLP	0.0000000	0.0923000	2930000.0000000	0.3580000	0.0000079	0.0242000	0.0026500	0.7190000	152.0000000	394.0000000
MLS-LSHADE	0.0000000	0.0208000	10.1000000	0.1210000	0.0000000	0.0445000	0.0070300	62.3000000	214.0000000	387.0000000
L-SHADE-OrdRW	0.0000000	0.0437000	10.9000000	0.1890000	6.1700000	0.2710000	0.2980000	100.0000000	319.0000000	400.0000000
PBO	0.0000000	6.9500000	11.5000000	0.4740000	11.8000000	0.6730000	0.4470000	0.0000000	100.0000000	400.0000000

Rotation										
DEDMNA	0.0000000	16.3000000	11.6000000	0.4370000	12.0000000	0.1760000	0.1880000	54.8000000	83.3000000	368.0000000
MadDE	0.0000000	12.3000000	13.6000000	0.3510000	1.1000000	0.3260000	0.2330000	96.0000000	90.0000000	398.0000000
RB_IPOP_CMAES_PPMF	0.0000000	498.0000000	11.4000000	0.7860000	114.0000000	46.8000000	75.0000000	96.1000000	205.0000000	418.0000000
J21	0.0000000	21.6000000	11.7000000	0.7960000	1.7300000	0.5170000	1.3700000	9.3100000	90.0000000	318.0000000
NL-SHADE-RSP	0.0000000	12.1000000	13.3000000	0.1300000	6.2000000	0.2970000	0.0783000	23.9000000	75.2000000	398.0000000
SOMA-CLP	0.0000002	4340000.0000000	13.8000000	0.2150000	1770000.0000000	0.3330000	0.1960000	33.5000000	191.0000000	399.0000000
MLS-LSHADE	0.0000000	18.7000000	13.1000000	0.4010000	6.3300000	0.6390000	0.4380000	65.5000000	100.0000000	388.0000000
L-SHADE-OrdRW	0.0000000	5.2000000	11.8000000	0.3360000	29.0000000	0.5810000	2.2900000	100.0000000	290.0000000	428.0000000
PBO	15.6000000	15.1000000	10.7000000	0.5390000	77.9000000	1.3200000	1.3600000	11.4000000	0.0003790	398.0000000
Bias and Shift										
DEDMNA	0.0000000	0.0000000	2.1400000	0.1280000	0.0000000	0.0035600	0.0004750	0.0000000	0.0000000	48.0000000
MadDE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
RB_IPOP_CMAES_PPMF	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
J21	0.0000000	0.0000000	4.7100000	0.2390000	0.0000000	0.0327000	0.0198000	0.0000000	0.0000000	46.4000000
NL-SHADE-RSP	0.0000000	0.0000000	0.0000000	0.0134000	0.0000000	0.0074800	0.0011200	0.0000000	0.0000000	0.0020700
SOMA-CLP	0.0000000	0.1190000	34600000.0000000	0.3080000	0.0000000	0.0284000	0.0029500	0.0000000	0.0000000	49800000.0000000
MLS-LSHADE	0.0000000	0.0000000	2.3200000	0.0032900	0.0000000	0.0001530	0.0000000	0.0000000	0.0000000	0.0072100
L-SHADE-OrdRW	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
PBO	0.0000000	0.3120000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	48.0000000
Bias and Rotation										
DEDMNA	0.0000000	15.0000000	9.8200000	0.4420000	3.5800000	0.3740000	0.1570000	3.6200000	0.0000000	51.4000000
MadDE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000315	0.0000000	0.0000000	0.0000000	0.0000000
RB_IPOP_CMAES_PPMF	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
J21	0.0000000	16.2000000	11.3000000	0.8390000	4.3400000	1.3300000	0.3410000	12.4000000	0.0000000	51.6000000
NL-SHADE-RSP	0.0000000	6.4600000	5.2700000	0.4300000	2.1000000	0.4230000	0.2130000	44.1000000	0.0000000	51.7000000
SOMA-CLP	0.0000001	24100000.0000000	15600000.0000000	1.0800000	71900000.0000000	0.4620000	0.3070000	37.7000000	0.0000000	52.1000000
MLS-LSHADE	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000009	0.0000000	0.0000000	0.0000000	0.0042900
L-SHADE-OrdRW	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000001	0.0000000	0.0000000	0.0000000	0.0000014
PBO	0.0000000	0.3120000	0.0000000	0.3250000	0.0000000	0.4490000	0.0000000	0.0000000	0.0000000	51.6000000

Shift and Rotation										
DEDMNA	0.0000000	0.0000000	4.8100000	0.1560000	0.0000000	0.0053100	0.0005080	14.0000000	86.7000000	373.0000000
MadDE	0.0000000	0.0083300	10.9000000	0.1930000	0.0000000	0.0145000	0.0015500	94.0000000	90.0000000	400.0000000
RB_IPOP_CMAES_PPMF	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000
J21	0.0000000	0.0020800	9.8500000	0.2490000	0.0000000	0.0338000	0.0077200	0.0000000	104.0000000	389.0000000
NL-SHADE-RSP	0.0000000	0.0000000	10.9000000	0.1010000	0.0000000	0.0057100	0.0049000	0.0000000	76.0000000	400.0000000
SOMA-CLP	0.0000000	0.1060000	979.0000000	0.3480000	0.0000000	0.0294000	0.0029100	0.0034200	127.0000000	387.0000000
MLS-LSHADE	0.0000000	0.0312000	9.8200000	0.1580000	0.0000000	0.0324000	0.0057000	52.0000000	195.0000000	390.0000000
L-SHADE-OrdRW	0.0000000	0.0333000	10.9000000	0.1940000	6.1600000	0.3980000	0.2470000	100.0000000	319.0000000	400.0000000
PBO	0.0000000	3.5400000	2.4400000	0.3220000	0.4380000	0.6490000	0.7270000	0.0000000	100.0000000	400.0000000
Bias, Shift and Rotation										
DEDMNA	0.0000000	16.3000000	11.6000000	0.4370000	12.0000000	0.1760000	0.1880000	54.8000000	83.3000000	368.0000000
MadDE	0.0000000	21.6000000	14.0000000	0.3720000	0.9630000	0.3020000	0.1720000	91.3000000	90.0000000	398.0000000
RB_IPOP_CMAES_PPMF	0.0000000	553.0000000	10.5000000	0.8660000	93.5000000	49.7000000	121.0000000	97.1000000	172.0000000	415.0000000
J21	0.0000000	21.6000000	11.5000000	0.7950000	1.7000000	0.5170000	1.9100000	11.4000000	90.0000000	338.0000000
NL-SHADE-RSP	0.0000000	14.5000000	13.1000000	0.1380000	5.0300000	0.3010000	0.0278000	32.4000000	79.7000000	388.0000000
SOMA-CLP	0.0000001	6910000.0000000	13.5000000	0.2000000	315000000.0000000	0.2730000	0.1840000	28.1000000	263.0000000	389.0000000
MLS-LSHADE	0.0000000	18.2000000	12.6000000	0.3800000	5.1500000	0.7300000	0.3020000	70.6000000	91.2000000	384.0000000
L-SHADE-OrdRW	0.0000000	4.0600000	11.5000000	0.3470000	28.9000000	0.5750000	2.8000000	100.0000000	292.0000000	428.0000000
PBO	4.4900000	3.9500000	14.5000000	0.2290000	44.1000000	0.6770000	2.4000000	12.9000000	100.0000000	398.0000000

REFERENCES

[1] Dorigo, M. (1992). Optimization, learning and natural algorithms. PhD Thesis, Politecnico di Milano.

[2] Simon, Dan. "Biogeography-based optimization." IEEE transactions on evolutionary com putation 12(6) (2008): 702-713.

[3] Yang, X. S. (2010). "Firefly algorithm, stochastic test functions and design optimization". International journal of bio-inspired computation, 2(2), 78-84.

[4] Karaboga, D., & Basturk, B. (2007). "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm". Journal of global optimization, 39(3), 459-471.

[5] Teodorovic, D., & Dell'Orco, M. (2005). "Bee colony optimization–a cooperative learning approach to complex transportation problems". Advanced OR and AI methods in transportation, 51, 60.

[6] Thakar, J. D., Kunte, K., Chauhan, A. K., Watve, A. V., and Watve, M. G., (Aug. 2003), "Nectarless flowers: ecological correlates and evolutionary stability," Oecologia, 136(4), pp. 565–570.

[7] V., P., Sriram, B., and Watve, M. G., (Dec 2009), "The co-optimization of floral display and nectar reward," Journal of Biosciences, 34(6), pp. 963–967.

[8] Yang, X. S., Karamanoglu, M., and He, X. S., (Oct. 2013), "Flower pollination algorithm: A novel approach for multiobjective optimization," Engineering Optimization, 46(9), pp. 1222–1237.

[9] Yang, X. S., and Deb., S., (Dec. 2009), "Cuckoo search via levy flights," Proceedings of World Congress on Nature & Biologically Inspired Computing, (NaBIC 2009), Coimbatore INDIA, pp. 210–214.

[10] Mousavirad SJ, Moghadam MH, Saadatmand M, Chakrabortty RK, "An ordered and roulette-wheel-based mutation incorporated l-shade algorithm for solving cec2021 single objective numerical optimisation problems", Proceedings of the genetic and evolutionary computation conference companion, 2021, pp. 1–2

[11] Biswas, S., Saha, D., De, S., Cobb, A. D., Das, S., & Jalaian, B. A. (2021, June). Improving differential evolution through Bayesian hyperparameter optimization. IEEE Congress on evolutionary computation (CEC), 2021, pp. 832-840

[12] Warchulski, E., Arabas, J., "A new step-size adaptation rule for cma-es based on the population midpoint fitness", IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 825-831.

[13] Stanovov, V., Akhmedova, S., & Semenkin, E., "NL-SHADE-RSP algorithm with adaptive archive and selective pressure for CEC 2021 numerical optimization", IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 809-816

[14] Cuong, L. V., Bao, N. N., & Binh, H. T. T. "Technical report: A multi-start local search algorithm with l-shade for single objective bound constrained optimization", Hanoi University of Science and Technology, Vietnam, 2021.

[15] Bujok, P., Kolenovsky, P., "Differential evolution with distance-based mutation-selection applied to CEC 2021 single objective numerical optimization", IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 849-856.

[16] Brest, J., Maučec, M. S., & Bošković, B, "Self-adaptive differential evolution algorithm with population size reduction for single objective bound-constrained optimization: Algorithm j21", IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 817-824.

[17] Kadavy, T., Pluhacek, M., Viktorin, A., & Senkerik, R., "SOMA-CLP for competition on bound constrained single objective numerical optimization benchmark: a competition entry on bound constrained single objective numerical optimization at the genetic and evolutionary computation conference (GECCO), Proceedings of the genetic and evolutionary computation conference companion, 2021, pp. 11-12. DOI:10.1145/3449726.3463286

[18] Sethy, Prabira Kumar (2020), "Rice Leaf Disease Image Samples", Mendeley Data, V1, doi: 10.17632/fwcj7stb8r.1

[19] Sharma, Rahul, Singh, Amar (2021) "An Integrated Approach towards Efficient Image Classification Using Deep CNN with Transfer Learning and PCA", Advances in Technology Innovation. doi: 10.46604/aiti.2021.8538.

[20] Sharma, R., Singh, A., Kavita, Jhanjhi, N.Z., Masud, M., Jaha, E.S. and Verma, S., 2022. Plant Disease Diagnosis and Image Classification Using Deep Learning. CMC-COMPUTERS MATERIALS & CONTINUA, 71(2), pp.2125-2140.

[21] Sharma Rahul, and Amar Singh. "Big bang–big crunch-CNN: an optimized approach towards rice crop protection and disease detection. "Archives of Phytopathology and Plant Protection 55, no. 2 (2022): 143-161.