Asian Journal of Computer and Information Systems (ISSN: 2321 – 5658)
Volume 03 – Issue 01, February 2015

Comparative Analysis of Genetic Algorithm and Particle Swam Optimization: An Application in Precision Agriculture

Oluleye Babatunde[1,2*], Leisa Armstrong[1], Jinsong Leng[3], and Dean Diepeveen[4]

[1]School of Computer and Security Science, Edith Cowan University, Perth, WA, Australia

[2]Department of Information and Communication Technology, Osun State University, Osogbo, Nigeria

[3]Security Research Institute, Edith Cowan University, Perth, WA, Australia

[4]Department of Agriculture and Food, South Perth, 6067, WA, Australia

[*]*Corresponding author's email: hezecomp [AT] yahoo.com*

ABSTRACT— ***This article details the exploration and application of Genetic Algorithm (GA) and Particle Swam Optimization (PSO) for the wrapper-based feature selection. Particularly a comparative study is carried out, examining the performances of both GA and PSO with respect to classification accuracy of some classifiers. 112 features were extracted features from set of images found in the Flavia dataset (a publicly available dataset). The extracted features are Zernike Moments (ZM), Fourier Descriptors (FD), Legendre Moments (LM), Hu's Moments (Hu7M), Texture Properties (TP), Geometrical Properties (GP), and Colour features (CF). The main contribution of this article includes the comparison of two major optimization techniques, i.e., GA and PSO, and the development of a GA-based feature selector using a novel fitness function which enabled the GA to obtain a combinatorial set of feature giving rise to optimal accuracy. The effectiveness of these manifold projection techniques were tested on Probabilistic Neural Networks (PNN), k Nearest Neighbour (kNN) and Multilayer Perceptron (MLP). The experimental analysis demonstrates the classification accuracy with GA-based approach outperforming that with PSO-based method.***

Keywords— Genetic Algorithm, Particle Swam Optimization, Feature selection, Precision Agriculture.

1. INTRODUCTION

High dimensional feature set could pose a great difficulty to pattern or image recognition systems. This is known as "the curse of dimensionality ". In other words, too many features often require intensive computation and also reduce the classification accuracy of the recognition system since some of the features may be redundant and non-informative [6, 7]. For the wrapper-based approaches, different combinatorial set of features should be obtained in order to keep the best combination to achieve optimal accuracy. In this work, both GA-based and PSO-based feature selection (a subspace or manifold projection techniques) are used to optimize the initial setting of the concerned classifiers, so as to obtain the 'optimal' subset of features. The performance of the concerned learning machines is further evaluated with the features selected from both GA and PSO.

A Feature Subset Selection (FSS) is an operator Fs or a map from an ***m-dimensional feature*** *space (input space) to* ***n-dimensional*** *feature space (output) given in mapping,*

$$Fs : R^{r\times m} \rightarrow R^{r\times n} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

where $m \geq n$ and $m,n \in Z^+$, $R^{r\times m}$ is any database or matrix containing the original feature set having r instances or observation, $R^{r\times n}$ is the reduced feature set containing r observations in the subset selection. This is further illustrated in Figure 1.

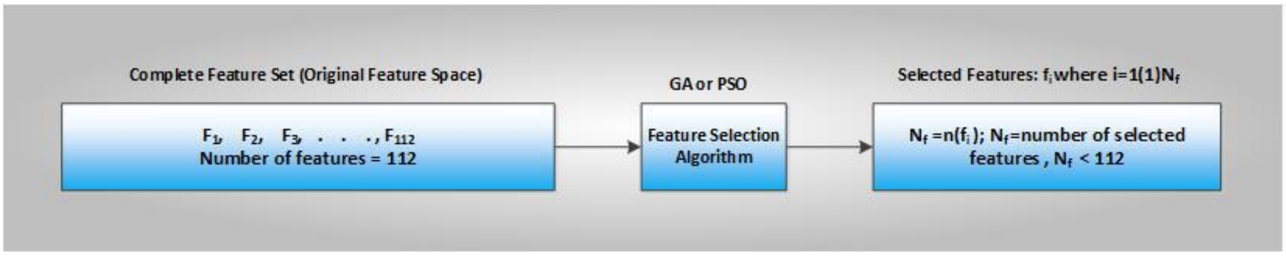

Complete Feature Set (Original Feature Space)
F1, F2, F3, . . ., F112
Number of features = 112
GA or PSO
Feature Selection Algorithm
Selected Features: fi where i=1(1)Nf
Nf =n(fi); Nf=number of selected features , Nf < 112

Figure 1: Illustrative diagram on Feature selection

The map F_S in Equation (1) can be any linear or nonlinear. There are three FS techniques, categorized as follow:

(a) Filtering-based approach
(b) Wrapper-based approach
(c) Embedded hybrid approach

In filtering-based approach, some evaluation function is used independently of the classifier in selecting feature subsets. Examples of functions used in these methods are: metric measure, information gain, dependency and consistency. The simplest of these methods is **best individual features**, where a function is used to rank individual features and the highest ranked p features are selected. The low scoring features are removed. The major advantage of this method is that it is computationally simple and fast and it is also carried out independently of the classification algorithm.

In Wrapper-based approach, feature subset selection is done through evaluation of each candidate subset with estimation obtained from the classifier or learning algorithm [9]. Wrapper Algorithms interact with the learning algorithm and model feature dependencies. However, the effectiveness of FS is dependent on the classifiers being selected. Wrapper-based FS is the choice for this study as it guarantees better accuracies.

In the embedded hybrid approaches, the search function is built into the learning or classification algorithms [10, 11]. That means classifier is seen as a composite functional. In other words, the feature space is fed into the classifier and the feature selector component part of the classifier is invoked first before the classification is done.

2. RELATED WOK

The following table taken from [15], summarizes types, examples, advantages and disadvantages of feature selection methods.

Type	Advantage	Disadvantage	Examples
(1a) Univariate Filter	It's fast, scalable and Independent of the learning algorithm	It ignores feaure depencies and interaction with the learning algorithm	Chi-square, Euclidean distance, t-test, Information gain, Gain ratio etc
(1b)Multivariate Filter	It models feature dependencies, it's independent of the learning algorithm, it has better computational complexity than wrapper method	It's slower and less scalable than univariate techniques. It ignores interaction with the learning algorithms	Correlation-based feature selection(CBFS), Markov Blanket filter (MBF), Fast Correlation-Based Feature Selection (FCBF)
(2a) DeterministicWrapper	It's simple. It interact with the learning algorithm. It models feature dependencies. It's less computationally intensive than stochastic methods.	It has risk of over fitting. It's more to getting stuck in a local optimum than the stochastic algorithms	Sequential Forward Search(SFS), Sequential Backward Elimination (SBE), **plus** q **minus** r, Beam search
(2b) Stochastic Wrapper	It's less prone to local optima. It interacts with the learning algorithm. It models feature dependencies.	It's computationally intensive. It's dependent on the learning algorithm. It has higher risk of overfitting than the deterministic type	Simulated Annealing (SA), Randomized Hill Climbing, Genetic Algorithms (GA), Estimation of distribution algorithms
(3) Embedded	It has better computational complexity and better interaction with the classifier than Wrapper	It's a classifier-dependent algorithm	Decision tress, Weighted Naive Bayes, feature

	methods. It models feature dependencies		selection using SVM weight vectors, Sparse regression, LASSO, Random Multinomial Logit (RMNL), Regularized Random Forest Memetic Algorithm, Autoencoding networks with a bottleneck-layer and many other machine learning methods applying a pruning step.

3 COMPLETE DATASET (FEATURE SPACE)

The *Flavia dataset (a publicly available dataset) is utilized for the comparative study in this work* [16]. The Flavia dataset is composed of a set of highly constrained leaf images taken against a white background and without any stem. Each class of leaves (with red, green and blue channels) has 50 to 77 leaf samples having resolution of 1600 x 1200 pixels (see Figure 2). *The original features extracted from the raw plant leaf images* comprise of ZMs, FDs, Legendre Moments, Hu 7 Moments, texture, geometrical properties and Colour Features. The variables Fi; i = 1, 2, 3, ... 112, in Table 2 and Table 3 represent the original features needed for this work. Thus the feature space of this work is a $R^{r \times m}$ matrix, where r, (number of observations)= 1907 and m, (number of attributes or futures required) = 112.

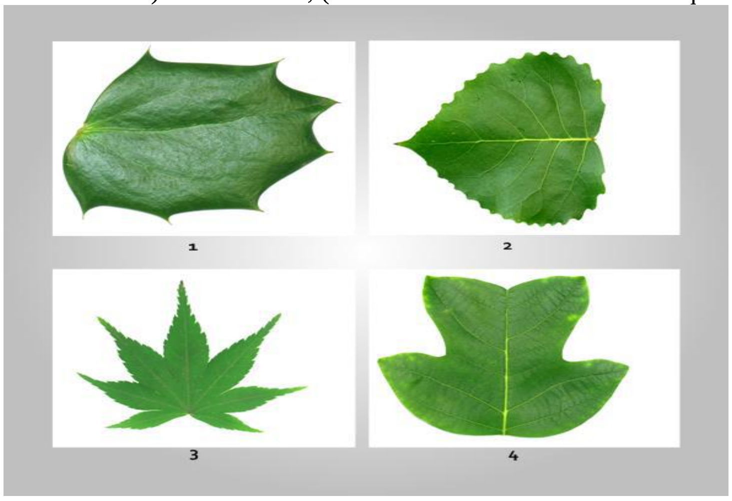


Figure 2: Four selected samples from the Flavia dataset

Table 2: 112 Features in the extracted from the Flavia dataset

SN	Descriptor	Feature Index	Descriptor Cardinality
1	Zernike Moments (ZM)	$F_{01}, F_{02}, \ldots, F_{20}$	20
2	Lengendre Moments(LM)	$F_{21}, F_{22}, \ldots, F_{40}$	20
3	Hu 7 Moments (Hu7M)	$F_{41}, F_{42}, \ldots, F_{47}$	7
4	Texture Features (TF)	$F_{48}, F_{49}, \ldots, F_{69}$	22
5	Geometric Features (GF)	$F_{70}, F_{71}, \ldots, F_{79}$	10
6	Fourier Descriptors (FD)	$F_{80}, F_{81}, \ldots, F_{100}$	21
7	Colour features (CF)	$F_{101}, F_{102}, \ldots F_{112}$	12

Table 3: Variable representation for the features in Table 2.

Observation	**(Features)**
	F_1 F_2 F_3 ....F_{112}
Image1	$X_{1,1}$ $X_{1,2}$ $X_{1,3,}$ … ....$X_{1,112}$
Image2	$X_{2,1}$ $X_{2,2}$ $X_{2,3,}$ .... $X_{2,112}$
Image3	$X_{3,1}$ $X_{3,2}$ $X_{3,3,}$ .... $X_{3,112}$
…	....
…	....
…	....
Image1907	$X_{1907,1}$ $X_{1907,2}$ $X_{1907,3,}$ … $X_{1907,112}$

4 GENETIC ALGORITHM (GA)

Genetic Algorithms (GA) can be defined as population-based and algorithmic search heuristic methods that mimic natural evolution process of man [2, 4, 5, 12, 14]. GA iteratively employ the use of one population of chromosomes (solution candidates) to get a new population using a method of natural selection combined with genetic functionals such as crossover and mutation (in the similitude of Charles Darwin evolution principle of reproduction, genetic recombination, and the survival of the fittest). In comparative terminology to human genetics, chromosomes are the bit strings, gene is the feature, allele is the feature value, locus is the bit position, genotype is the encoded string, and phenotype is the decoded genotype [13]. The fitness of each chromosome is evaluated using a function commonly referred to as objective function or fitness function. In other words, the fitness function (objective function) reports numerical values which are used in ranking the chromosomes in the population. The fitness function used for both GA and PSO is given as Equation 2. The detailed description of the GA can be found in the companion paper [3].

$$f = \frac{\alpha}{N_O - N_S} \quad \text{.......................................(2)}$$

where

α = **kNN-based classification error** ; N_O = **Cardinality of the original feature set** ; N_S = **Cardinality of the selected features.**

Table 3: GA Configuration

GA Parameter	Value
Population size	120
Genomelength	112
Population type	bitstrings
Fitness Function	f in Equation (2)
Number of Generations	300
Crossover	Arithmetic Crossover
Crossover Probability	0.8
Mutation	Uniform Mutation
Mutation Probability	0.1
Selection scheme	Tournament of size 2
EliteCount	2

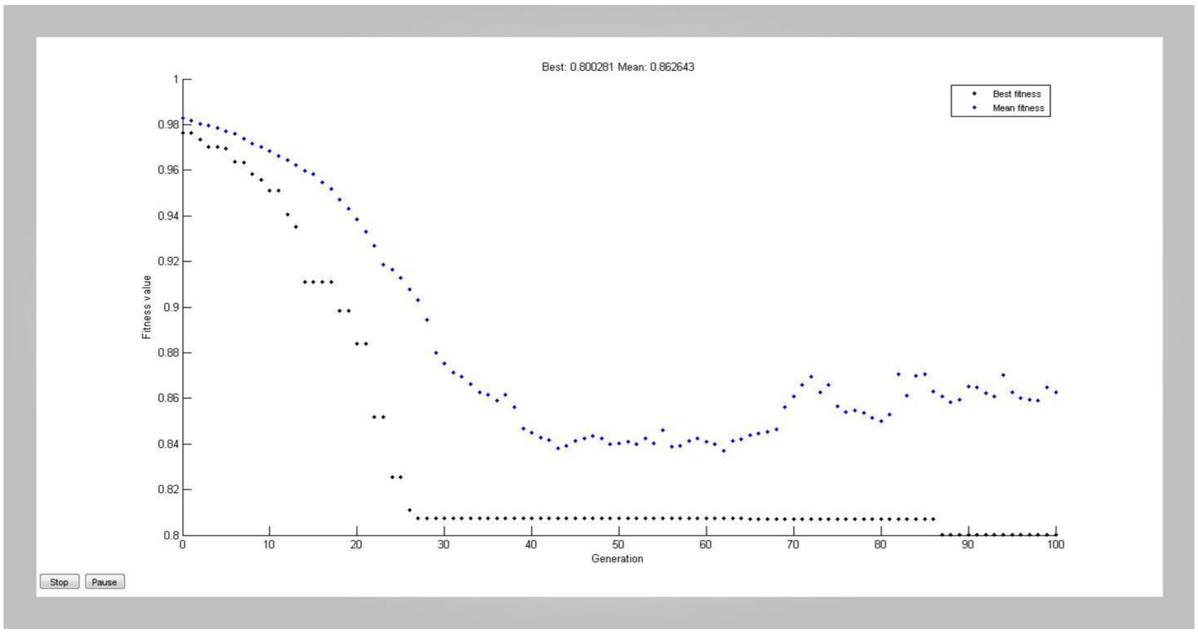


Figure 3: GA Simulation diagram on the 112 feature space

The features generated by GA are those indexed by the following positional vectors:

FeatVect1 = GA_based_Features = [3, 6, 8, 21, 56, 71, 75, 78, 101, 102, 103, 110, 111,112]

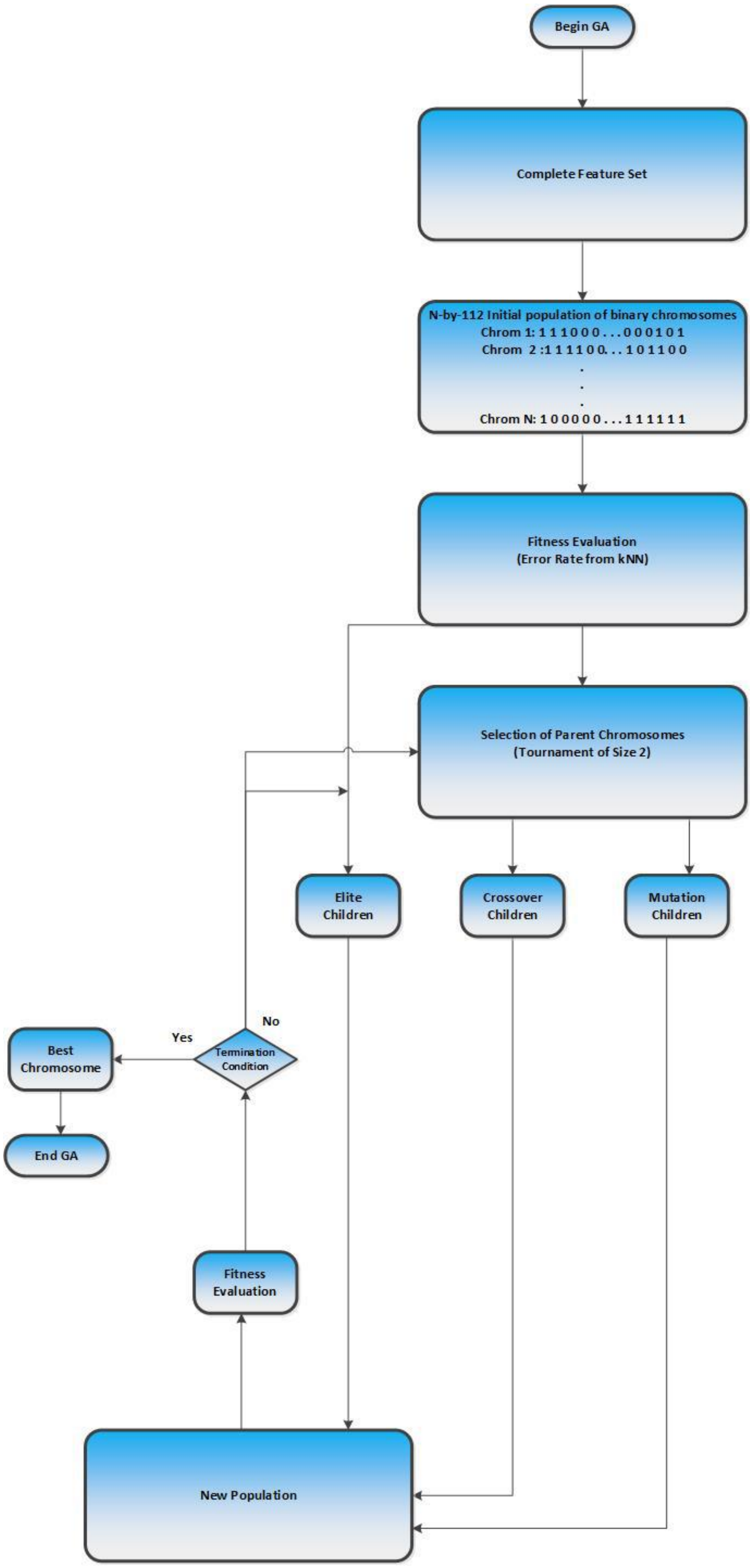


Figure 4: GA-Based Feature Selection [3].

5. PARTICLE SWAM OPTIMIZATION (PSO)

The Particle Swam Optimization (PSO) is a computational technique that optimizes a given problem through some iterative updates on some solution candidates. The PSO was first introduced by Russel C. Eberhart and James Kennedy in 1995. It is an adaptive and swarm intelligence meta-heuristic algorithm based on socio-psychological paradigm. Specifically, the development PSO was based on observation of a group of animal behaviours such as bird flocks or fish schools. Just like the GA, PSO is also a population-based method, since it represents the state of the algorithm by a population, which is modified iteratively until a stopping criteria is met. Herein, a population of individuals (solution candidates or particles) adapts by randomly going back towards previously successful regions. It is to be noted that unlike the GA, PSO do not modify the population from generation to generation, but instead keep the same population, iteratively updating the positions of the members of the population. The PSO has two main functions (or operators):

(i) **velocity update** (see Equation 3)
(ii) **position update** (see Equation 4)

During each generation each particle is accelerated towards the particles previous position, and the distance from the global best position. The new velocity is then used to calculate the next position of the particle in the search space. The process is iteratively repeated until some stopping criteria are met. Such may be minimum error. Let $f_{PSO} : R^n \to R$ be the cost function associated with the PSO-based feature selection in this work. In other words, the features in the Table 3 are to be reduced to a subfeature that minimizes classification error of our system. In this case the classification output will be the least error associated with the any of the sub-features generated by the **binary PSO** used herein. Thus the output, say **psoERROR** $\in$ R, is a single scalar (i.e **0.0009364** in Figure 6). The goal is to find a solution $x^* \in R$ such that $f_{PSO}(x^*) \le f_{PSO}(x)$ for all x (different combinatorial set of features from the original dataset) in the search space.

1. Generate a population of agents (also called particles) over a uniform distribution space.
2. Using a suitable objective function, evaluate each particle's position. The fitness function used for the PSO herein is the same as that used for the GA (see Equation 2). If the current position of a particle is better than the previous, update it.
4. Determine the best particle according to the particle's previous position.
5. Update the velocities of the particles and
6. Update the position of the particle via:

$$V_i^{k+1} = \omega V_i^k + c_1 r_{i1}^k (P_i^k - X_i^k) + c_2 r_{i2}^k (P_{pBEST}^k - X_i^k) \text{ ....(3)}$$

$$X_i^{k+1} = X_i^k + V_i^{k+1} \text{ ....(4)}$$

where
- k = iteration number = 1000
- $i = 1,2,3,\ldots,N$, N = Swarm initial population = 120
- ParticleLength = 112
- Population type : bitstrings
- Fitness function : kNN-based classification error
- Cognitive parameter c_1 = 1
- Social parameter c_2 = 1
- Initial weight ω = 0.25
- r_{i1}^k and r_{i2}^k are random numbers uniformly distributed in the interval [0, 1].

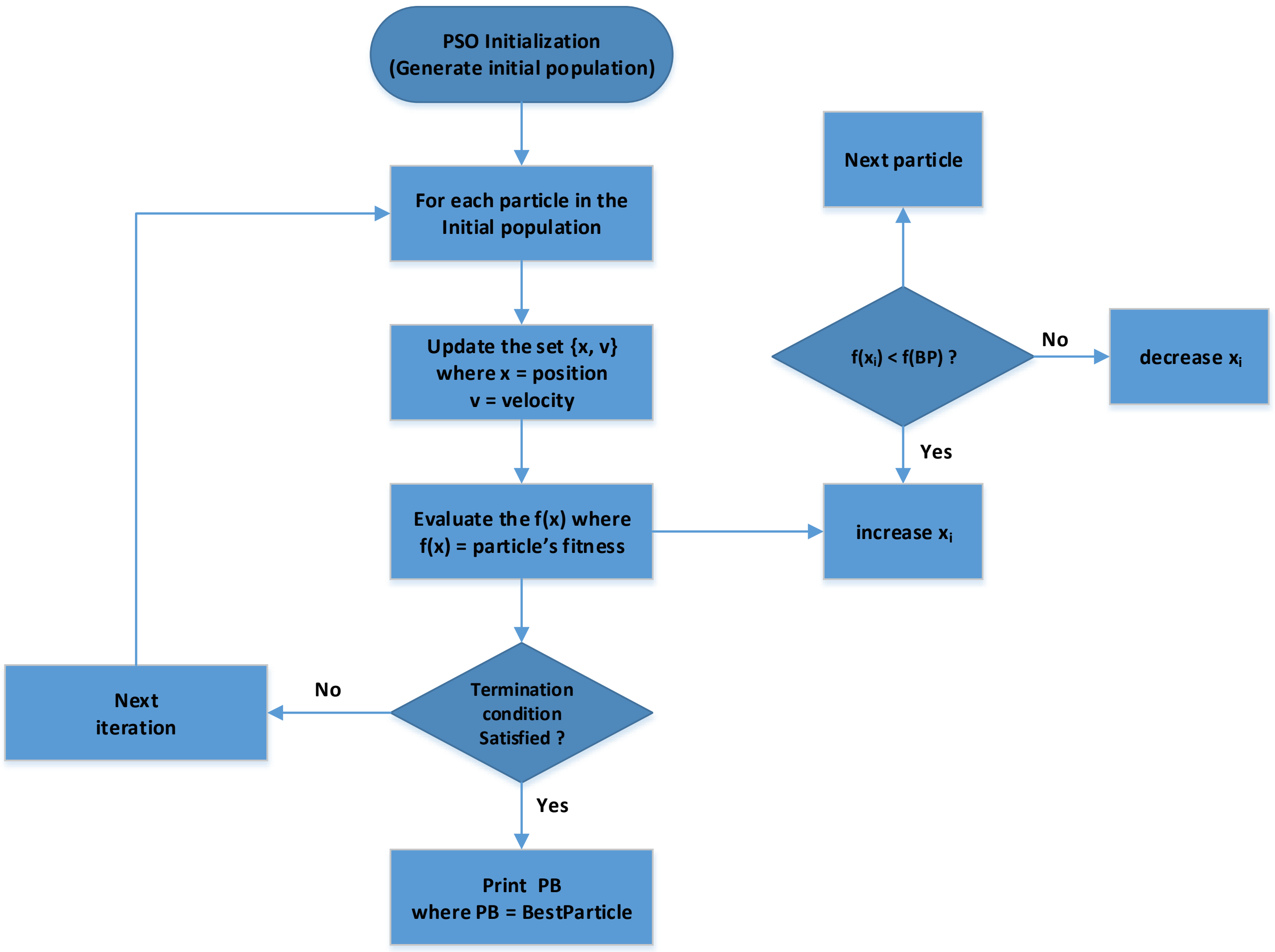

PSO Initialization (Generate initial population)
For each particle in the Initial population
Update the set {x, v} where x = position v = velocity
Evaluate the f(x) where f(x) = particle's fitness
Termination condition Satisfied ?
No
Next iteration
Yes
Print PB where PB = BestParticle
increase x_i
f(x_i) < f(BP) ?
Yes
No
decrease x_i
Next particle

Figure 5: Overview of Particle Swarm Optimization (PSO).

The following enumerated points are valid for explaining the underlying PSO used in this work:

1. **Original number of features:** The candidate solution representing all features (shown in Table 3) in the original feature space is by a $\mathbf{R}^{112 \times 1}$ matrix given as:

OriginalString = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1].

This is also called a particle but this is a special particle as it captures all the features represented in the original feature space. For this special particle, n = 112.

2. **BPSO Particle:** A particle in the PSO is represented as **n-bit** string, where n is the number of available features associated with the solution candidate in the feature space. For example if the candidates population is a **PopulationSize x 112** matrix of binary strings which can be generated as follows:

```
function PopBits = PopulationFunctionPSO
PopulationSize = 120;
CandidateLength = 112;
RD1 = rand; % To generated binary chromosomes by comparison
PopBits = rand (PopulationSize, CandidateLength) > RD1;
End
```

Any row from PopBits is a solution candidate. For example suppose a given row is given as :

PSOCandidate =
[1 1 0 0 0 1 0 1 0 0 0 0 1 1 1 0 1 1 0 1 0 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 1
1 1 0 1 0 1 0 1 0 0 1 1 0 1 1 0 0 0 0 1 1 1 1 0 1 1 0 1 1 1 1 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 1 1 0 0 0 0 1 0 1 1 0 0]

3. **Binary digit** {**0,1**}: In each binary string, **"1"** represents selected features, while **"0"** represents unselected features.

4. **How to obtain the features related to a particular candidate:** Using the **PSOCandidate** above, we code as follows to obtain the associated features:

PSOCandidate = find (**PSOCandidate** ==1); which gives the following feature index:

PSOCandidateFeatureIndex = [1 2 6 8 13 14 15 17 18 20 29 30 34 37 43 44 45 49 53 54 55 57 59 61 64 65 67 68 73 74 75 76 77 79 80 82 83 84 85 86 91 98 101 102 107 109 110]

The new dataset associated with **PSOCandidate** will now be given as:

NewDataset = OldDataset (: , [1 2 6 8 13 14 15 17 18 20 29 30 34 37 43 44 45 49 53 54 55 57 59 61 64 65 67 68 73 74 75 76 77 79 80 82 83 84 85 86 91 98 101 102 107 109 110])

The features generated by PSO are those indexed by the following positional vectors:

FeatVect2 = PSO_based_Features = [1, 4, 8, 18, 19, 71, 74, 78, 100, 101, 102, 103, 110, 111]

The FeatVect2 is just a variable holding the indexes of the features selected by the PSO.

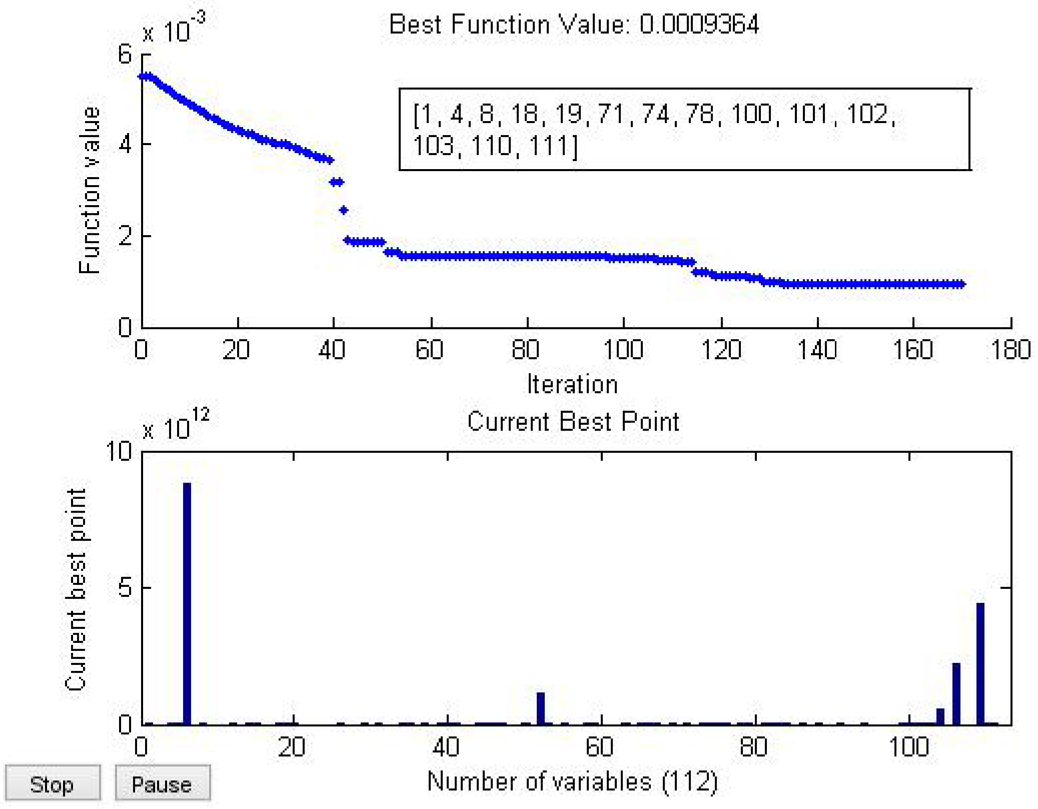


Figure 6: Simulation result for PSO: The features selected by the PSO were those indexed by the vector [1, 4, 8, 18, 19, 71, 74, 78, 100, 101, 102, 103, 110, 111] in the dataset. There is a very high correlation and relationship between these features and those selected by the GA.

6. EXPERIMENTAL FEATURE ANALYSIS

The computing platform for the entire experiments (both feature selections and image classification) was **MATLAB** (**MAT**rix **LAB**oratory). A image processing tool built using MATLAB is shown in Figure 10. The same number of features was generated by both GA and PSO. This mostly probably, may be due to the same fitness function used in both algorithms. (see Equation 2). Out of the 14 features selected by the GA and PSO, 8 were similar (see Figure 9). This implies the features common to both are very discriminating. These features were **zernike moments**, **texture properties**, and **colour features**. Based on the indices selected by both GA and PSO, the corresponding features were then derived and tested on ***radial basis network*** (**RBF**), ***general regression neural network*** (**GRNN**) and ***multi-layer perceptron*** (**MLP**). A 10-fold cross validation (10 fold CV) was used to partition the feature space into the following was partitioned into training data and test data as {1717, 1716, 1716, 1716, 1716, 1716, 1716, 1716, 1717, 1717} and {190, 191, 191, 191, 191, 191, 191, 191, 190, 190} respectively. The visual diagram for each of the fold is shown below.

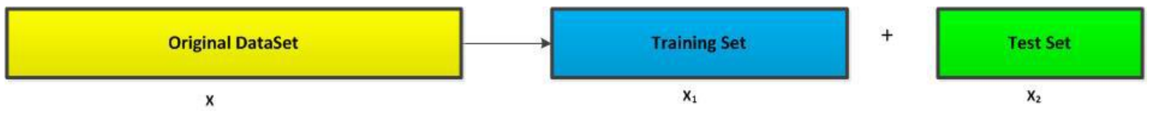


Figure 7: Splitting of Original DataSet into Training Set and Test Set

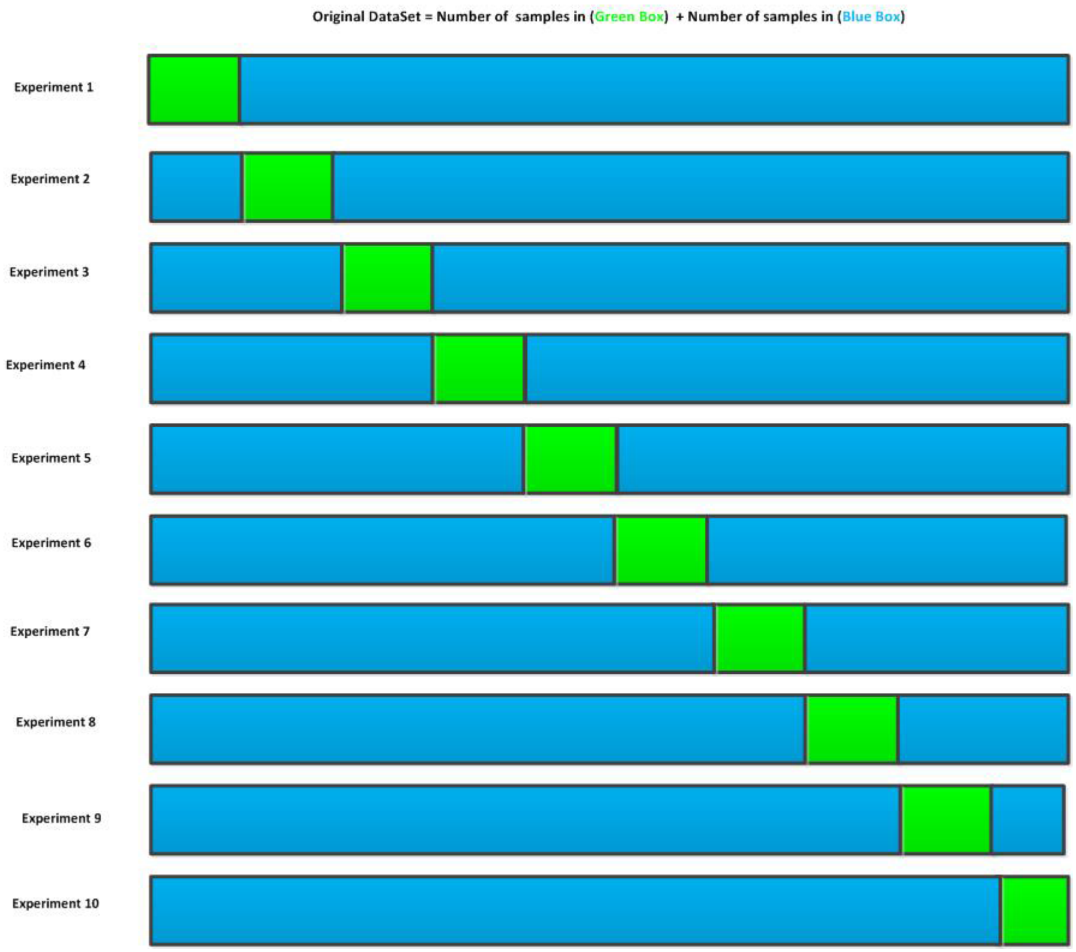


Figure 8: Visual Representation of 10-Fold Cross Validation Experiments. The 10-Fold CV runs for 10 iteration, computing the classification accuracy for each fold , storing the accuracies and finally computing the average of these accuracies.

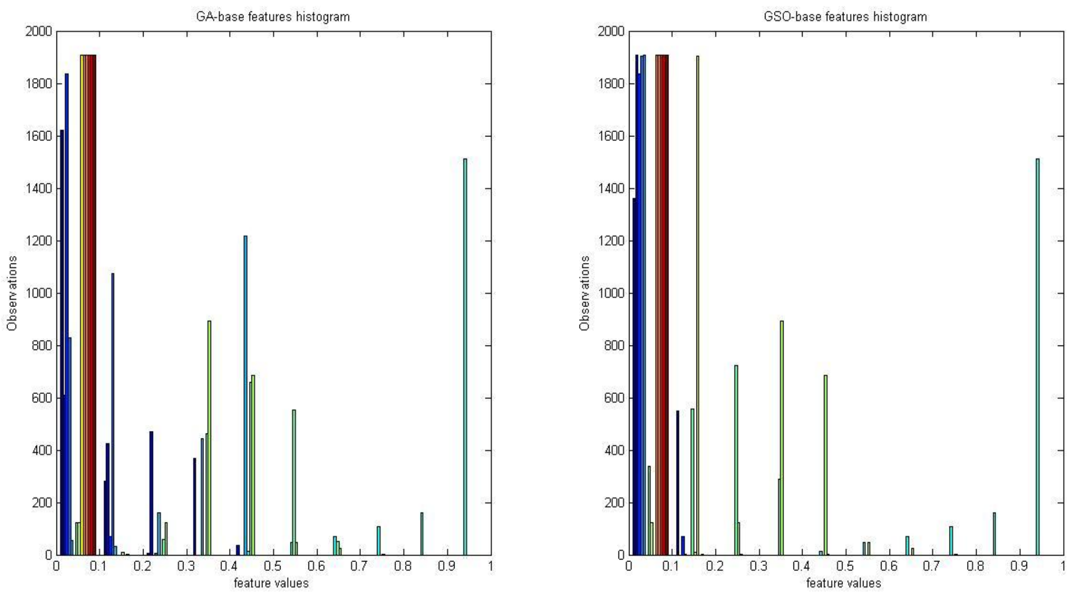
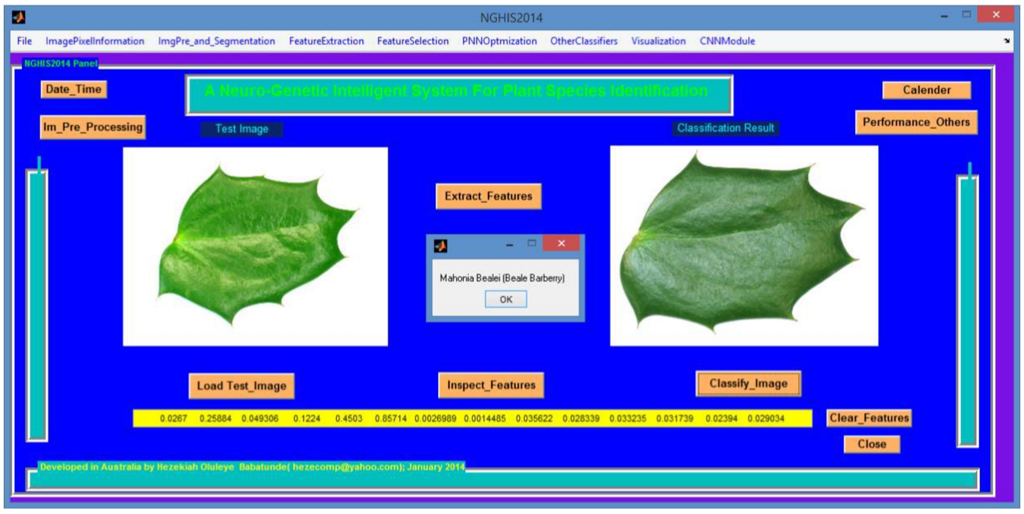


Figure 9: Histogram of both GA-based and PSO-based features. A careful look at the these two histograms shows the similarity between the features selected by both GA and PSO.

Table 4: Experimental results

S/N	Classification model	Accuracy
1	RBF + GA-based features	88.98%
2	RBF + PSO-based features	82.75%
3	GRNN + GA-based features	87.65%
4	GRNN + PSO-based features	81.95%
5	MLP + GA-based features	90.245%
6	MLP + PSO-based features	82.456%
7	RBF + Original features	82.56%
8	GRNN + Original features	80.01%
9	MLP + Original features	81.10%



A Neuro-Genetic Intelligent System For Plant Species Identification

Mahonia Bealei (Beale Barberry)

0.0267 0.25884 0.049306 0.1224 0.4503 0.85714 0.0026989 0.0014485 0.035622 0.028339 0.033235 0.031739 0.02394 0.029034

Developed in Australia by Hezekiah Oluleye Babatunde(hezecomp@yahoo.com); January 2014

Figure 10: A computer-based vision system for plant species classification

7. CONCLUSION

This works demonstrates the impact of both GA and PSO on some selected classifiers. The original features space was reduced from a **1907 x 112 matrix** of real numbers to **1907 x 14 matrix** of real numbers. In other words, both GA and PSO selected only **12.50%** of the original dataset. Nine different classification models were tested as shown in Table 4. The results herein showed that both GA and PSO-based features outperformed the original features while GA-based feature in turn outperformed the PSO-based features. The features selected by both GA and PSO are somewhat similar as shown in Figure 9. This may be due to the same fitness function used for both. However the different computational nature of the two evolutionary algorithms (GA and PSO) may be the reason why the number of features selected both algorithms are not entirely the same. Only **57% features were** similarly by both algorithms. An indication here is that both PSO and GA are good candidates for feature selections and their application in precision agriculture (***computer-based vision systems for automatic identifications of plant species***) proved useful as they were able to improve the classification accuracy of the underlying classifiers.

8. REFERENCES

[1] Bruzzone, L. and C. Persello (2010). "A novel approach to the selection of robust and invariant features for classification of hyperspectral images." Department of Information Engineering and Computer Science, University of Trento.

[2] Babatunde, O., Armstrong, L., Leng, J., & Diepeveen, D. (2014a). Application of cellular neural networks and naivebayes classifier in agriculture. AFITA 2014, 9th Conference of the Asian Federation for Information Technology in Agriculture, Australia, Perth, 6 - 9 October 2014.

[3] Babatunde, O., Armstrong, L., Leng, J., & Diepeveen, D. (2014b). A genetic algorithm-based feature selection. International Journal of Electronics Communication and Computer Engineering, 5, 889–905.

[4] Babatunde, O., Armstrong, L., Leng, J., & Diepeveen, D. (2014c). On the application of genetic probabilistic neural networks and cellular neural networks in precision agriculture. Asian Journal of Computer and Information Systems, 2(4), 90-100.

[5] Babatunde, O., Armstrong, L., Leng, J., & Diepeveen, D. (2014d). Zernike moments and genetic algorithm: Tutorial and application. British Journal of Mathematics and Computer Science., 4(15), 2217-2236.

[6] Bellman, R. (1956). Dynamic programming and Lagrange multipliers.*Proceedings of the National Academy of Sciences of the United States of America*, *42*(10), 767.

[7] Bellman, R. E., & Dreyfus, S. E. (1962). Applied dynamic programming.

[8] Bruzzone, L., & Persello, C. (2010). A novel approach to the selection of robust and invariant features for classification of hyperspectral images. Department of Information Engineering and Computer Science, University of Trento.

[9] Cordon, O., Herrera, DelJesus, M. J., & Villar, P. (2001). A multi-objective genetic algorithm for feature selection and granularity learning in fuzzy-rule based classication system. IEEE, 1253-1258.

[10] Kittler, J. (1978). Feature set search algorithms. Pattern Recognition and Signal Processing. Sijhoff an Noordhoff, the Netherlands.

[11] Kohavi, R., & John, G. (1996). Wrappers for feature subset selection. . Artificial Intelligence, special issue on relevance, 97(1-2), 273-324.

[12] Melanie, M. (1999). An introduction to genetic algorithms. A Bradford Book The MIT Press.

[13] Sivanandam, S. N., & Deepa, S. N. (2008). Introduction to genetic algorithms. Springer-Verlag , Berlin, Heidelberg.

[14] Tian, J., Hu, Q., Ma, X., & Ha, M. (2012). An improved kpca/ga-svm classication model for plant leaf disease recognition. Journal of Computational Information Systems, 18(8), 7737-7745.

[15]Yvan, S., Inaki, I., & Pedro, L. (2005). A review of feature selection techniques in bioinformatics. BIOINFORMATICS, 0, 1-10.

[16] Wu, S. G., Bao, F. S., Xu, E. Y., Wang, Y. X., Chang, Y. F., & Xiang, Q. L. (2007). A leaf recognition algorithm for plant classification using probabilistic neural network. In *Signal Processing and Information Technology, 2007 IEEE International Symposium on* (pp. 11-16). IEEE.