Clustering using Levy Flight Cuckoo Search

J. Senthilnath^{a1}, Vipul Das^{b2}, S.N. Omkar^{a3}, V. Mani^{a4}

^{a} Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India

^{b} Department of Information technology, National Institute of Technology, Karnataka, India

{^{1}snrj@aero.iisc.ernet.in; ^{2}vipulramdas@gmail.com; ^{3}omkar@aero.iisc.ernet.in;
^{4}mani@aero.iisc.ernet.in}

Abstract. In this paper, a comparative study is carried using three nature-inspired algorithms namely Genetic Algorithm (GA), Particle Swarm Optimization (PSO) and Cuckoo Search (CS) on clustering problem. Cuckoo search is used with levy flight. The heavy-tail property of levy flight is exploited here. These algorithms are used on three standard benchmark datasets and one real-time multi-spectral satellite dataset. The results are tabulated and analysed using various techniques. Finally we conclude that under the given set of parameters, cuckoo search works efficiently for majority of the dataset and levy flight plays an important role.

Keywords: Genetic algorithm, Particle swarm optimization, Cuckoo search, Levy flight, Clustering.

1 Introduction

Clustering is an unsupervised learning method where objects with closer resemblance are grouped together to form a cluster based on a similarity measure. The objective of clustering is to minimize intra-cluster distance while inter-cluster distance is maximized [1]. Clustering has various applications which include data analysis, machine learning, image analysis and other engineering applications.

Clustering can be classified into two types: hierarchical and partition. In hierarchical clustering, objects belong to more than one cluster forming a hierarchical pattern. Hierarchical clustering is carried out by splitting and merging the dataset. In splitting the number of cluster centres generated would be greater than the number of classes while merging is to group the dataset to exact number of classes. In partition clustering, objects are clustered into disjoint groups without forming a hierarchy. In both methods, similarity measure is used to generate cluster centres.

Previously, the most popularly used and tested partition based algorithm is k-means clustering. The main disadvantage of k-means clustering is convergence to

J. C. Bansal et al. (eds.), *Proceedings of Seventh International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA 2012),* Advances in Intelligent Systems and Computing 202, DOI: 10.1007/978-81-322-1041-2_6, © Springer India 2013

the local minima [2]. In literature, nature inspired algorithms are used effectively in clustering problem as it converges to global minima [2, 3]. These algorithms are based on the exploration and exploitation behaviour observed in nature and is effectively used in optimization problems.

In this paper, a comparative performance study is carried out based on the results obtained using three nature inspired algorithms namely genetic algorithm (GA), particle swarm optimization (PSO) and cuckoo search algorithm (CS) on clustering problem. The standard benchmark clustering data used in our study are the same that is available in the (UCI machine learning repository) literature [4] and a real-time multi-spectral satellite image for crop type classification. Xin-She *et.al* [5] has implemented and analyzed CS algorithm by comparing with GA and PSO using standard benchmark functions. In their study, CS algorithm is used with levy flight and is found to be performing better compared to the other two methods. In literature, CS has been used without levy distribution for clustering problem on satellite image [3]. In our study, we use CS with levy flight as used in [5], on clustering data set by comparing with GA and PSO. The important property of levy flight is it makes sure that the whole search space is covered, which is due to the heavy-tailed property of levy distribution [6-10]. In our study, we split the data into training and testing samples. The cluster centres are determined using the algorithms on the training dataset and the testing dataset is used to determine the classification error percentage (CEP).

The remaining sections are in the following order: in section 2 the problem formulation for clustering is discussed, in section 3 a brief discussion of the algorithms is presented, in section 4 and section 5 we discuss analysis of the results obtained and conclusion respectively..

2 Problem Formulation

The clustering is done based on unsupervised learning. Here the data is divided into training set and testing set. The training set data is used to generate the cluster centres. The aim of clustering is to minimize the objective function [2].

$$f(k) = \sum_{k=1}^{K} \sum_{i=1}^{n_k} (x_i - c_k)^2 \qquad 1$$

where $k=1,2,...K$ is the number of clusters, x_i , $i=1,2,...n_k$ are the patterns in the k^{th} cluster, c_k is centre of the k^{th} cluster. Here the cluster centres are represented by

$$c_k = \frac{1}{n_k} \sum_{i=1}^{n_k} x_i \qquad 2$$

In this study, the nature-inspired algorithms are used to find the cluster centers from the training data set. This is done by placing each object to their respective cluster centers using the distance measure. The testing data set is used to calculate percentage error using classification matrix.

3 Methodology

This section gives brief introduction about the algorithms used in our study, the way it has been applied for clustering problem and also the pseudo-code for the algorithms are discussed.

3.1 Genetic algorithm

This algorithm is based on the natural selection process seen in nature [11, 12]. The best fit organism of the current generation carries on the genes to the next generation. The concept of genetic operators (cross-over and mutation) is included in the algorithm wherein a change in the gene structure is introduced that produces an entirely different trait. The main idea behind genetic algorithm is the operators used namely reproduction, crossover and mutation.

This algorithm takes a predetermined number of random solutions (population) in the search space called chromosomes. Here the convergence criterion is used to terminate the algorithm. At each iteration the chromosomes are made to crossover using single point crossover and the fitness of each chromosomes is calculated using

$$f_i = f(x_i) \quad i=1,2,...,n \tag{3}$$

where $f(x_i)$ is the fitness function given by Eq. 1 considering the clusters individually and n is the population size.

The fittest chromosomes (solutions) among the entire population are considered for the next generation (iteration). At any random point the chromosomes undergo mutation based on the mutation rate. The fitness is calculated and the best solutions carryon till termination criteria is reached. Thus the cluster centres are generated using the training data set.

Pseudo-code

```
1. Initialize population of $n$ chromosomes
2. Repeat till stopping criteria
        a) Calculate fitness using Eq. 3
        b) Apply elitism by sorting the fitness value of the population
        c) Retain the best fit solutions (reproduction)
        d) Crossover the adjacent chromosomes at a random position using single point crossover
        e) Mutate randomly selected point within a chromosome
3. Cluster centre will be the best fit solution from the population
```

3.2 Particle Swarm Optimization

This is a population based method which iteratively improves the solution by moving the solutions closer to the optimal solution. Here each particle moves towards the optimal solution with a velocity v_i at each iteration. Eventually all particles converge to an optimal position [13].

Initially n particles are created and randomly distributed in the search space. The fitness of each particle is evaluated using Eq.3 and Eq.1, considering the classes individually. All the particles are made to move one step towards the fittest particle (global best solution) as well as towards its personal best position with a velocity v_i given by

$$v_i(t+1)=w*v_i(t)+b_p*rand*(p_i-c_i)+b_g*rand*(g-c_i) \quad 4$$

where p_i is the personal best position of the particle, c_i is the current position of the particle, g is the global best of the entire particle, w is the inertial constant, b_p is the personal best constant and b_g is the global best constant, $i=1, 2,..., n$. Each particle moves using

$$c_i(t+1)=c_i(t)+v_i \quad 5$$

The fitness of each particle is calculated and the personal best position and the global best are determined. This process is repeated until stopping criteria is met. The global best position will be the cluster centre to the given data set.

Pseudo-code

```
1. Initialize $n$ particles
2. Repeat till stopping criteria met
   a) Calculate fitness of each particle using Eq.3
   b) global best position is the best fit particle
   c) move all the particles towards the global best position using Eq.4 and Eq.5
   d) for each particle if (fitness of current position < fitness of personal best) then personalbest = current position
   e) update personal best position for each particle
   f) global best fitness value is retained
3. Cluster centre is the global best position
```

3.3 Cuckoo Search

This algorithm is based on the breeding pattern of parasitic cuckoos [3, 5, 14]. Some species of cuckoo namely ani and Guira lay their eggs in the nest of other birds. The possibility of occurrence of such act leads to i) the host birds' eggs being destroyed by the cuckoo itself or the cuckoo chick upon hatching; ii) the host birds may realise the presence of a foreign egg in its nest and may throw away these eggs or abandon the nest altogether and build a new nest elsewhere [5].

These are the processes in nature that this algorithm inculcates. The basic assumptions made are: 1) At a time each cuckoo lays one egg and dumps it into randomly chosen nest; 2) The best nest with high quality eggs will carry over to the next generation; 3) Each nest contains only one egg and the number of host nests are fixed and; 4) The probability that the host bird discovers the cuckoo egg is $p_{a.}$. This implies that the fraction p_a of n nests is replaced by new nests (with new random solutions) [5].

Each nest represents a solution and a cuckoo egg represents a new solution. The aim is to use the new and potentially better solutions (cuckoo eggs). An initial population of host nest is generated randomly. The algorithm runs till the convergence is reached. At each iteration a cuckoo is selected at random using levy flight as given [5]

$$x_i(t+1)=x_i(t) + \alpha*L \qquad 6$$

where α is the step-size, L is a value from the Levy distribution, $i=1,2,...,n$, n is the number of nests considered. The fitness of the cuckoo is calculated using Eq.3 and Eq.1, considering the classes individually.

Choose a random nest from the given population of nests and evaluate its fitness from Eq.6. If the fitness of the new solution is better than the older one then replace the older one with the new one. A fraction p_a of the total number of nests is replaced by new nests with new random solution. The best nests with the fittest egg (solution) are carried-on to the next generation.

This is continued till the termination criteria is reached and the best nest with fittest egg is taken as the optimal value. Thus the cluster centres can be generated using this optimal value.

Pseudo-code

```
1. Initialise $n$ nests
2. Repeat till stopping criteria is met
   a) Randomly select a cuckoo using levy flight using Eq.6
   b) Calculate its fitness using Eq.3 ($F_c$)
   c) Randomly select a nest
   d) Calculate its fitness using Eq.3 ($F_n$)
   e) If ($F_c < F_n$) then Replace the nest with the cuckoo
   f) A fraction $p_a$ of nest are replaced by new nests
   g) Calculate fitness and keep best nests
   h) Store the best nest as optimal fitness value
3. Cluster centre will be the best nest position
```

4 Results and discussion

In this section the results and the performance evaluation are discussed. The specifications of the clustering data used in this study are given in Table 1. The training data are randomly picked from the dataset for vehicle dataset and glass dataset. The training data for image segmentation dataset are as in the UCI repository.

Table 1. Specifications of the clustering dataset used

Dataset	Total data	Training data	Test data	Attributes	Classes
Image segmentation	2310	210	2100	19	7
Vehicle	846	635	211	18	4
Glass	214	162	52	9	6*
Crop Type	5416	2601	2815	4	6

*Glass dataset has 7 classes. The data for the fourth class is unavailable.

The performance measures used in this paper are classification error percentage [2], Statistical significance test [4], Receiver operating characteristic [15, 16] and time complexity analyses.

4.1. Classification error percentage

The result of application of the algorithms on clustering data is given in terms of classification error percentage. This is the measure of misclassification of the given dataset using the particular algorithm. Let n be the total number of elements in the dataset and m be the number of elements misclassified after finding out the cluster centre using the above algorithms, then classification error percentage is given by

$$CEP = \frac{m}{n} * 100 \tag{7}$$

Table 2. Classification error percentage

Dataset \ Algorithms	GA	PSO	CS
Image segmentation	32.6857	32.45716	30.56188
Vehicle	61.61138	60.18956	58.76636
Glass	61.15386	55.76924	45.76926
Crop type	19.3677	20.0710	20.0355

The algorithms are run five times and the average of the results is as shown in Table 2. The values are obtained using the testing dataset. The parameters such as the maximum generation and the number of initial random solution are kept the same for all the algorithms. Each algorithm is run till it converged to a point with a tolerance of 0.01. In GA, the best 40% of the parent generation and the best 60% of the offspring generation are carried on to the next generation. In PSO, the inertial constant (w), the personal best constant (b_p) and the global best constant (b_g) are all set to 1. In CS algorithm, the probability factor p_a is set to 0.25.

4.2. Statistical significance test

Statistical significance is done to ascertain that the results are obtained consistently. No matter where the initial random solutions are picked up from, they would always converge to the global optimum position (cluster centre). This would imply that an algorithm which performed better than the other algorithms will always perform better when run under similar initial conditions. In this study a binomial test is conducted [3] between CS and GA and also CS and PSO based on the result obtained on image segmentation dataset.

Assume the test is carried between CS and GA. Here the total number of test-runs is *N*, i.e., the result of CS and GA differ in *N* places. Let *S* (success) is the number of times CS gave correct result and *F* (failure) is the number times GA gave correct result. Now, calculating the *p-value* (probability of *S* successes out of *N* trials) using the binomial distribution as

$$P = \sum_{j=S}^{N} {}^{N}C_{j} * p^{j} * q^{N-j} \qquad 8$$

Here *p* and *q* are the probability that the algorithms CS and GA will succeed. Let *p* and *q* value be set to 0.5, assuming each algorithm to behave the same. The results of comparison of CS with GA and CS with PSO are as shown in Table 3. With a low value of *P*, we can say that cuckoo search gives better result than GA and PSO, the chance has nothing to do with the better performance of CS algorithm.

Table 3. Binomial Test on image segmentation dataset

	N	S	F	P
GA	255	153	102	$8.44e^{-04}$
PSO	62	35	27	0.1871
CS	-	-	-	-

4.3. Receiver Operating Characteristics

Receiver operating characteristics [15, 16] are used to evaluate the performance of a binary classifier. An experiment will have actual values and prediction values. If the prediction value is *P* and the actual value is also *P*, then it is called true positive (TP). If prediction value is *P* and the actual value is *N*, then it is called false positive (FP). Likewise, true negative (TN) if prediction value is *N* and actual value is *N* and false negative (FN) when the prediction value is *N* and actual value is *P*. The above can be shown using a 2×2 contingency matrix given in Table 4.

With the above statements, we define three parameters namely – sensitivity or true positive rate (TPR), false positive rate (FPR) and accuracy (ACC) given by

$$TPR = \frac{TP}{TP + FN} \qquad 9$$

$$FPR = \frac{FP}{FP + TN} \tag*{10}$$

$$ACC = \frac{TP + TN}{TP + FN + FP + TN} \tag*{11}$$

Sensitivity defines how many correct positive results occur among all the positive samples available during the test i.e., in our case the number of elements that have been correctly clustered amongst all the elements that belonged to the particular class. FPR defines how many incorrect positive results occur among all negative samples available during the test i.e., the number of misclassified elements amongst all the other elements that does not belong to the particular class. Accuracy defines how many samples have been correctly classified to their respective classes.

Table 4. ROC Contingency matrix

		Predicted value	
		True	False
Actual Value	True	True Positive	False Negative
	False	False Positive	True Negative

In our case, we analyse on the image segmentation dataset. We give an example of the analyses using cuckoo search. The classification matrix obtained after applying cuckoo search algorithm on image segmentation data is given in Table 5.

Table 5. Classification matrix of image segmentation dataset using CS algorithm

	Class1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
Class 1	126	0	91	14	69	0	0
Class 2	0	294	0	6	0	0	0
Class 3	60	1	171	14	54	0	0
Class 4	69	12	10	183	9	14	3
Class 5	30	0	50	21	195	0	4
Class 6	9	0	9	0	0	249	33
Class 7	0	0	17	0	41	4	238

In the above representation, row indicates the class the element belongs to and the column indicates the class the elements are classified into after using the cluster centre based on the CS algorithm. The principal diagonal elements represent correctly classified elements. Consider class 1, from Table 4, we have TP=126, FN= 174, FP=168, TN=1632. From this data, we calculate the true positive rate, false positive rate and the accuracy of the given algorithm on class 1 of the given clustering dataset. From Eq. 9, Eq. 10 and Eq. 11, TPR is 0.4200, FPR is 0.0933 and ACC is 0.8371. This implies that 42% of what actually belonged to class 1 was correctly classified and 9% of the data which did not belong to class 1 were added to class 1. The overall efficiency of the algorithm with respect to class 1 is 83%. Similarly the ROC analyses for all the classes of image segmentation dataset for the above three algorithms are given in Table 6.

Table 6. ROC analyses for image segmentation data using CS, GA and PSO

Class	CS			GA			PSO		
	TPR	FPR	ACC	TPR	FPR	ACC	TPR	FPR	ACC
1	42%	9.3%	83%	73%	21%	77%	46%	21%	77%
2	98%	0.7%	99%	99%	0.8%	99%	98%	0.8%	99%
3	57%	9.8%	85%	6%	0.3%	86%	46%	0.3%	86%
4	61%	3.0%	92%	56%	2.8%	91%	61%	2.8%	91%
5	65%	9.6%	86%	63%	9.7%	86%	67%	9.7%	86%
6	83%	1.0%	96%	87%	2.1%	96%	84%	2.1%	96%
7	79%	2.2%	95%	82%	1.2%	96%	78%	1.2%	96%

4.4. Time complexity analysis

The pseudo-codes of the algorithms are discussed in section 3. The time complexity of each algorithm can be derived from the pseudo-code. The time complexity analysis gives us an insight into the complexity of calculation involved in the algorithm, in order to know the time taken to produce the output. The time complexities of the algorithms are given in Table 7.

Table 7. Time complexity

Algorithm	Time complexity
GA	*O(clnum*gen*(comp_fit + sort_inb + m))*
PSO	*O(clnum*gen*(comp_fit * m))*
CS	*O(clnum*gen*(comp_fit * m))*

The algorithm is run till the stopping condition is met which in this case is till the solutions converge to a point with a tolerance of 0.01. Let the total number of iterations be *gen* and the number of clusters is *clnum.* Thus the total number of outer iterations is *clnum* gen.* Let *m* be the population size and *n* be the number of fitness evaluation to generate each cluster center. Thus in each iteration, fitness is calculated with a time complexity of *O(n)*. Let this *O(n)* be called *comp_fit*. In GA, additional operation is performend by sorting the population using *m* fitness values. This is done using a Matlab inbuilt function. Let the complexity of this function be *sort_inb*. Crossover and mutation takes *(m/2)* and *m* run respectively. Thus in each iteration, the overall operations executed will be of the order (*comp_fit* + *sort_inb* + *m/2* + *m* + *C*). Thus the overall order is (*clnum*gen**(*comp_fit* + *sort_inb* + *m/2* + *m* + *C*)). Thus the time complexity of GA used in this paper is *O(clnum*gen**(*comp_fit* + *sort_inb* + *m*)). Similarly for PSO and CS, *clnum, gen*, *m* and *comp_fit* implies the same as in GA.

The algorithms are run on a system with core i-5 processor, 4 GB memory on Matlab version 7.12.0.635. The execution time in secs taken by these algorithms to converge to the solution on glass dataset is given in Table 8.

Table 8. Time taken by the algorithms on glass dataset (in seconds)

Algorithms	Trial 1	Trial 2	Trial 3	Trial 4	Trial 5
GA	47.8299	62.7020	58.2859	33.6453	41.2319
PSO	760.5055	661.8087	1051.3	676.1566	695.6928
CS	163.95	147.8284	141.5073	159.0653	141.6662

5 Conclusion and Discussions

In this paper, we have implemented and analyzed three nature inspired techniques for clustering problem. Here we observe that the average classification error percentage of clustering dataset using cuckoo search with levy flight algorithm is less than GA and PSO for the benchmark problems and is at par with GA and PSO for crop type dataset. The statistical significance test proves that the cuckoo search was not better by chance. The obtained *p*-value being very small implies that the cuckoo search is better than GA and PSO with a high confidence level. The ROC analyses further gives us an insight into the efficiency of cuckoo search.

In cuckoo search, the levy flight factor plays a major role here. The fact that levy flights are heavy-tailed is used here. This helps in covering the output domain efficiently. Looking into the time complexity measure, we see that GA has one additional computation compared to the other two i.e., sorting of the population (solutions) according to the fitness values. But this takes negligible time as the number of agents or the population size is only 20. Thus GA takes less time as expected but CS takes a far lesser time compared to PSO. This can be attributed to the fact that CS algorithm uses levy flight. Thus we can clearly observe that the heavy-tailed property of levy flights helps to converge to the solution fast thereby increasing the efficiency.

References

[1] Anitha, E.S., Akilandeswar, J., Sathiyabhama, B. : A survey on partition clustering algorithms. International Journal of enterprise and computing and business systems. 1 (1), 1 – 14 (2011)

[2] Senthilnath, J., Omkar, S. N., Mani, V.: Clustering using firefly algorithm – Performance study. Swarm and Evolutionary Computation. 1 (3), 164-171 (2011)

[3] Suresh, S., Sundararajan, N., Saratchandran, P.: A sequential multi-category classifier using radial basis function networks. Neurocomputing. 71, 1345-1358 (2008)

[4] Samiksha, G., Arpitha, S., Punam, B.: Cuckoo search clustering algorithm: a novel strategy of biomimicry. World Congress on Information and Communication Technologies. IEEE proceedings (2011)

[5] Xin-She, Y., Suash, D.: Cuckoo search via levy flight. World Congress on Nature and Biologically Inspired Algorithms. IEEE publication. 210-214 (2009)

[6] Viswanathan, G.M., Afanasyev, V., Sergey, V.B., Shlomo, H., Da Luz, M.G.E., Raposo, E.P., Eugene, S.H.: Levy flight in random searches. Physica A 282, 1-12 (2000)

[7] Viswanathan, G.M., Bartumeus, F., Sergey V.B., Catalan, J., Fulco, U.L., Shlomo, H., Da Luz, M.G.E., Lyra, M.L., Raposo, E.P., Eugene, S.H.: Levy flights in Biological systems. Physica A 314, 208-213 (2002)

[8] Peter, I., Ilya, P.: Levy flights: transitions and meta-stability. Journal of Physics A: Mathematical and General. J. Phys. A: Math. Gen. 39 L237–L246 (2006). doi: 10.1088/0305-4470/39/15/L01

[9] Ilya P.: Cooling down Levy flight. Journal of Physics A: Mathematical and Theoretical. J. Phys. A: Math. Theor. 40 12299–12313 (2007). doi: 10.1088/1751-8113/40/41/003

[10] John P.N.: Stable Distributions – Models for Heavy Tailed Data. chapter 1. Processed (2009)

[11] Yannis, M., Magdalene, M., Michael, D., Nikolaos, M., Constantin, Z.: A hybrid stochastic genetic–GRASP algorithm for clustering analysis. Oper Res Int J 8:33–46 (2008). doi: 10.1007/s12351-008-0004-8

[12] Whitley, L. D.: A genetic algorithm tutorial, Statist. Comput. 4:65–85 (1994)

[13] De Falcao, I., Dello Ciappo, A., Tarantino, E.: Facing classification problems with Particle swarm optimization – Applied Soft Computing 7. 652-658 (2007)

[14] Walton, S., Hassan, O., Morgan, K., Brown, M.R.: Modified cuckoo search: A new gradient free optimization algorithm- Chaos, Solition and Fractals 44. 710-718 (2011)

[15] Christopher D.B., Herbert T.D.: Receiver operating characteristics curves and related decision measures: A tutorial. Chemometrics and Intelligent Laboratory Systems 80. 24-38 (2006)

[16] Tom F.: ROC Graphs: Notes and Practical Considerations for researchers. HP Laboratories, MS 1143, 1501 Page Mill Road, Palo Alto, CA 94304 (2004)