Choosing DBSCAN Parameters Automatically using Differential Evolution

Amin Karami
Computer Architecture Department (DAC)
Universitat Politécnica de Catalunya (UPC)
Jordi Girona 1-3, Campus Nord, Barcelona, Spain
amin@ac.upc.edu

Ronnie Johansson
Informatics Research Center (IRC)
University of Skövde
P.O. Box 408, SE-541 28 Skövde, Sweden
ronnie.johansson@his.se

ABSTRACT

Over the last several years, DBSCAN (Density-Based Spatial Clustering of Applications with Noise) has been widely applied in many areas of science due to its simplicity, robustness against noise (outlier) and ability to discover clusters of arbitrary shapes. However, DBSCAN algorithm requires two initial input parameters, namely Eps (the radius of the cluster) and MinPts (the minimum data objects required inside the cluster) which both have a significant influence on the clustering results. Hence, DBSCAN is sensitive to its input parameters and it is hard to determine them a priori. This paper presents an efficient and effective hybrid clustering method, named BDE-DBSCAN, that combines Binary Differential Evolution and DBSCAN algorithm to simultaneously quickly and automatically specify appropriate parameter values for Eps and MinPts. Since the Eps parameter can largely degrades the efficiency of the DBSCAN algorithm, the combination of an analytical way for estimating Eps and Tournament Selection (TS) method is also employed. Experimental results indicate the proposed method is precise in determining appropriate input parameters of DBSCAN algorithm.

General Terms:

Artificial Intelligence, Metaheuristics

Keywords:

Clustering Analysis, DBSCAN, Differential Evolution, Tournament Selection

1. INTRODUCTION

Clustering is a fundamental and popular data analysis technique. It can be considered as an unsupervised classification to find a number of groups of similar data in a data set. Clustering techniques are applied in many fields such as image processing, pattern recognition, machine learning, information retrieval and so on [1]. There are a large number of clustering algorithms in the literature. They can usually be classified into the four categories: partitioning, hierarchical, density-based and grid-based methods [2].
DBSCAN (Density-Based Spatial Clustering of Applications with Noise), introduced by Ester et al. [3], is a density-based clustering. Initially, the DBSCAN clustering algorithm was proposed for clustering spatial data. But it has been popular rapidly and applied in different fields of science [4, 5, 6, 7, 8]. DBSCAN clusters data objects based on the density. Clusters are regarded as regions in which the objects are dense, and which are separated by regions of low object density or noise. It can discover clusters of arbitrary shape as well as to distinguish noise [9]. DBSCAN requires two input parameters, Eps (the radius of the cluster) and MinPts (the minimum data objects required inside the cluster). In spite of its advantages, the original DBSCAN algorithm suffers by some drawbacks: (1) it is not easy to determine proper values for Eps and MinPts, (2) the computational complexity without special structure is $O(n^2)$, but if a spatial index is used, the complexity can be reduced to $O(nlogn)$ [10], and (3) it fails when the border objects of two clusters are relatively close [11], or when there are multi-density and connected clusters.
There have been many efforts to mitigate the drawbacks of DBSCAN clustering algorithm. Jiang et al. [12] present a new hybrid method based on partitioning-based DBSCAN and ant clustering to improve memory usage in DBSCAN. The GMDBSCAN algorithm [13] based on spatial index and grid technique has been proposed to improve clustering in multi-density data sets. Xue-yong et al. [14] propose a density-based algorithm for intrusion detection using a method for calculating the distance and the merging process. Shah [10] gives a new detection method to discover clusters that exist within a cluster. Tepwankul and Maneewongwattana [15] present a metric which specifically measures the density quality of DBSCAN clustering in order to give better clustering quality. Edla and Jana [16] resolve the quadratic computational complexity drawback of DBSCAN by using the prototypes produced from a squared error clustering method such as K-means. Tran et al. [11] present a modified version of the DBSCAN algorithm to solve instability of DBSCAN when detecting border objects of adjacent clusters.
There are also some research to solve difficulties in finding appropriate input parameters. Darong and Peng [17] combine the grid partition technique and DBSCAN to automatically generate input parameters. The efficiency of this method has not been evaluated against noise and various data sets with different densities. This method also requires input parameters for grid partitioning. Smiti and Elouedi [18] combine Gaussian-Means (GM) and DBSCAN algorithm to determine the input parameters in DBSCAN. However, GM provides circular cluster shape not density-based clusters, and it is not strong against noise (outlier). It still needs input

parameter for the Gaussian distribution. These existing algorithms and techniques have their own drawbacks and limitations which leads to a bad clustering.

Recently, many research have combined clustering algorithms with optimization and meta-heuristic algorithms to improve the results of clustering. For instance, Simulated Annealing [19], Particle Swarm Optimization [20, 21, 22, 23], Tabu Search [24, 25], Harmony Search [26, 27, 28], Bees algorithm [29, 30, 31], and Ant Colony Optimization [32, 33]. However, there is no research to solve the problem of automatically choosing input parameters in DBSCAN algorithm. In this paper, we propose a new hybrid DBSCAN algorithm combining with an optimization algorithm as Binary Differential Evolution (BDE) to choose efficiency very well suited DBSCAN input parameters (BDE-DBSCAN). Since a slightly different setting (changing MinPts and Eps values) in DBSCAN may lead to total different clusters of a data set [34], an optimization procedure would fit the requirements (finding the optimal combination of MinPts and Eps) for a good clustering. Because an optimization algorithm is related to an optimal choice of process decisions that satisfy definite constraints and make an optimization criterion (performance or cost index) maximize or minimize [35, 36]. Since the Eps parameter can largely degrades the efficiency of the DBSCAN algorithm [37], the combination of an analytical way for estimating Eps and Tournament Selection (TS) method is employed. The TS [38] method can create more diverse Eps values until an appropriate combination of MinPts and Eps values be selected.

The rest of the paper is organized as follows. Section 2 describes the DBSCAN clustering algorithm and its characteristics. In Section 3, Differential Evolution algorithm is described. Section 4 presents the proposed hybrid clustering algorithm BDE-DBSCAN in detail. Experimental results and discussions are explained in Section 5. Finally, Section 6 draws conclusions.

2. DBSCAN: A DENSITY-BASED CLUSTERING

Density-based clustering discovers clusters as regions where the objects of the regions are dense. The clusters are separated from each other by low-density regions [2]. Density-based algorithms have considerable advantages over partitional and hierarchical clustering algorithms. It can define clusters of arbitrary shapes as well as effectively identify noise points. DBSCAN [3] is a density based algorithm which discovers clusters with arbitrary shape. However, it requires the specification of two input parameters which are hard to guess [18]. The input parameters are the radius of the cluster (Eps) and minimum required points inside the cluster (MinPts). The time complexity of DBSCAN algorithm is $O(n^2)$, but it can be reduced to $O(n \cdot logn)$ by building some special data structures. The basic idea in DBSCAN algorithm is as follows [3]:

Definition 1 (*Eps-neighborhood of an object*): The Eps-neighborhood of an object *p*, denoted by *Eps(p)* in a set of objects *D*, is defined by

$Eps(p) = \{q \in D | distance(p, q) \leq Eps\}$

Definition 2 (*Directly density-reachable*): An object *p* is directly density-reachable from an object *q* wrt. Eps and MinPts, in the set of objects *D*, if two conditions are satisfied:

(1) $p \in Eps(q)$

(2) $|Eps(q)| \geq MinPts$

Definition 3 (*Core object & border object*): An object is core object if it satisfies condition 2 of Definition 2, and a border object is not a core object itself but is density-reachable from another core object (see Definition 4).

Definition 4 (*Density-reachable*): An object *p* is density reachable from an object *q* wrt. Eps and MinPts if there is a chain of objects $p_1, ..., p_n, p_1 = q, p_n = p$ such that $p_i + 1$ is directly density-reachable from p_i.

Definition 5 (*Density-connected*): An object *p* is density-connected to an object *q*, if there is an object $o \in D$ such that both, *p* and *q* are density-reachable from *o*.

Definition 6 (*Cluster*): Let *D* be a data set of objects. A cluster *C* wrt. Eps and MinPts is a non-empty subset of *D* satisfying the following conditions:

(1) $\forall p, q$, if $p \in C$ and *q* is density-reachable from *p* wrt. Eps and MinPts, then $q \in C$ (Maximality).

(2) $\forall p, q \in C$, *p* is density-connected to *q* wrt. Eps and MinPts (Connectivity).

Definition 7 (*Noise*): Let $C_1, ..., C_k$ be clusters of the data set *D* wrt. Eps and MinPts. Then the noise is the set of objects in the data set *D* not belonging to any cluster C_i, i.e. noise=$\{p \in D | \forall i : p \notin C_i\}$, $i = 1, 2, .., k$.

The steps of DBSCAN clustering algorithm is summarized as follows [6, 7]:

Function *DBSCAN(Dataset D, Eps, MinPts)*
1: Select an arbitrary object P in D;
2: Retrieve all objects *density-reachable* from P by arbitrary/random Eps and MinPts values;
3: **if** P *is a core object* **then** a cluster is formed;
4: **if** P *is a border object* **then** no objects are density reachable from P and DBSCAN visit the next object of the data set;
5: **else** assign P to *noise* object;
6: Continue the process (from step 1) until all of the objects have been processed.
end

Algorithm 1: Pseudo-code of the DBSCAN

3. DIFFERENTIAL EVOLUTION ALGORITHM

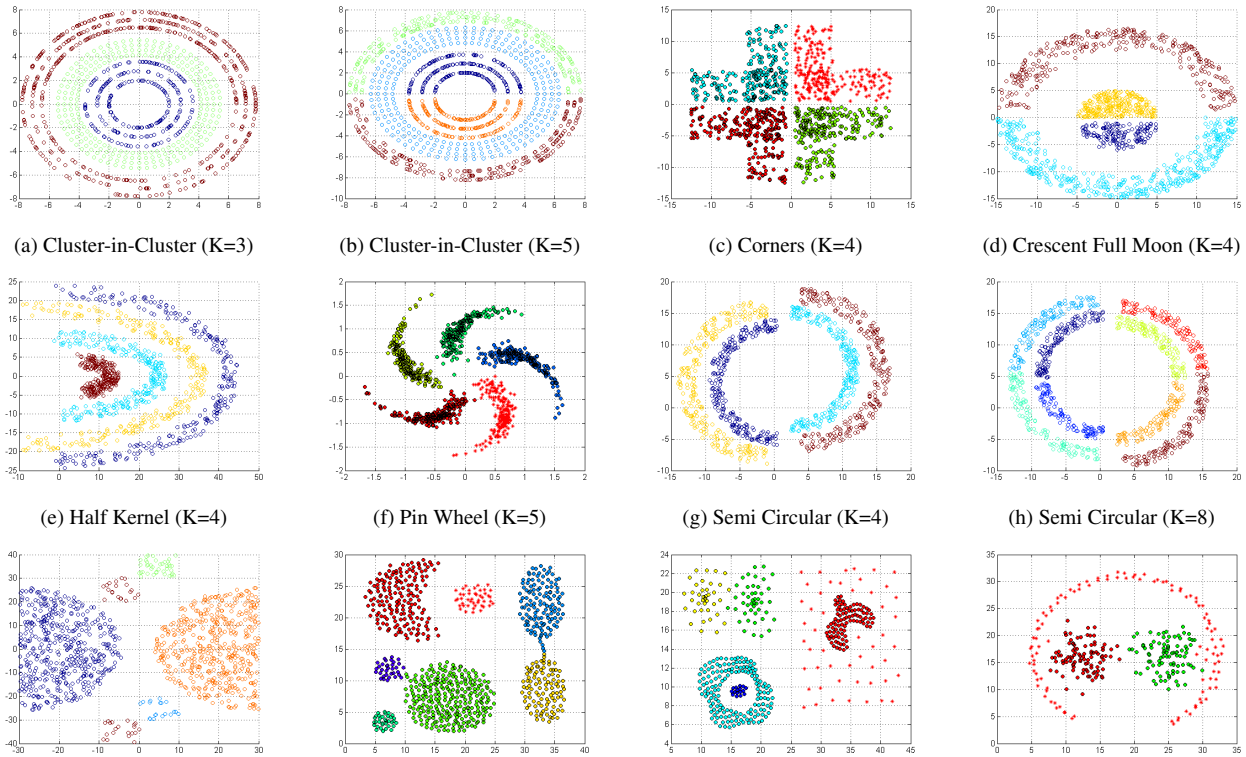
Differential Evolution (DE) Algorithm is a new evolutionary computational method for global optimization over continuous spaces proposed by Storn in 1997 [39]. Differential Evolution is similar to the overall structure of the genetic algorithm [40, 41]. DE consists of three basic operators: mutation, crossover and selection. Mutation is the most important operator in the performance of the DE algorithm because it generates new elements for the population, which may contain the optimum solution of the objective function [42, 43]. The DE algorithm can be summarized as follows [44]:

(1) Initialization: This step creates arbitrary initial population in *n* dimension space as follows:

$$x_i(j) = x_j^l + rand(0, 1) \cdot x_j^u \quad (1)$$

Where $x_i(j)$ denotes the *j*th variable of the *i*th individual, and x_j^l and x_j^u are the lower and upper constraints. $rand(0, 1)$ represents a uniformly distributed random value within [0 1].

(2) Mutation: DE randomly selects two population vectors $x_{p2}, x_{p3} (p2 \neq p3)$ which must be different from each other, then uses the difference between the individuals x_{p2}, x_{p3} by



(a) Cluster-in-Cluster (K=3) (b) Cluster-in-Cluster (K=5) (c) Corners (K=4) (d) Crescent Full Moon (K=4)

(e) Half Kernel (K=4) (f) Pin Wheel (K=5) (g) Semi Circular (K=4) (h) Semi Circular (K=8)

(i) Outlier (K=5) (j) Aggregation (K=7) (k) Compound (K=6) (l) Pathbased (K=3)

Fig. 1. The twelve applied data sets

scaling factor η (usually set the value within [0.5 1] [42]) to mutate x_{p1} by equation:

$$h_i = x_{p1} + \eta(x_{p2} - x_{p3}) \quad (2)$$

(3) Crossover: The new individual is generated by recombining x_i and h_i as represented below:

$$U_i(j) = \left\{ \begin{matrix} h_i(j),\ if\ rand(0,1) \leq CR\ or\ j = j_{rand} \\ x_i(j), otherwise \end{matrix} \right\} \quad (3)$$

Where $rand(0,1)$ is a random number within [0 1], j_{rand} is a randomly chosen index to ensure that the train vector U_i does not duplicate x_i, and the CR is crossover rate.

(4) Selection: Greedy algorithm is used to select the better one between the trial individual U_i and the parent vector x_i for the next generation using a fitness/cost function f:

$$x_i = \left\{ \begin{matrix} U_i,\ if\ f(U_i) \geq f(x_i) \\ x_i, otherwise \end{matrix} \right\} \quad (4)$$

The algorithm of DE in pseudo-code is shown in algorithm 2.

4. BDE-DBSCAN: BINARY DIFFERENTIAL EVOLUTION-BASED DBSCAN ALGORITHM

BDE-DBSCAN adopts the binary coding scheme and each individual (MinPts parameter) is represented by a bit string. In summary, the procedure of BDE-DBSCAN can be stated as follows:

Function *DE*
  Initialize the population P by Eq. 1;
1 **repeat**
    for $i \leftarrow 1$ **to** nPop *over* $P(i)$ **do**
      Execute Mutation operator by Eq. 2;
      Execute Crossover operator by Eq. 3;
      Execute Selection operator by Eq. 4 using a fitness/cost function;
    end
2 **until** *stopping condition*;
end

Algorithm 2: Pseudo-code of the DE

Step 1: Set control parameters and initialize the binary-coded populations randomly;
Step 2: Calculate the best fitness function value in terms of the purity metric (see section 5.2) for each individual and keep the maximum purity as Best Solution;
Step 3: Generate the mutant individuals according to Eq. 2;
Step 4: Generate the new trial individual by using the crossover operator in Eq. 3;
Step 5: Apply limit over real-coded individuals by Eq. 6;
Step 6: Evaluate the target individual and the corresponding trial individual, and choose the better one based on the maximum purity (fitness function) to survive into the next generation. Also, keep the best one into the Best Solution;
Step 7: If the terminal conditions are met, terminate the iteration;

Function *BDE-DBSCAN(data set: input Matrix)*
; `/Declare DE parameters`
MaxIt ← *Maximum iteration of DE algorithm*;
nPop ← *Number of individuals*;
nVar ← *The number of decision variables*;
Population ← *[Pop_1 Pop_2 ... Pop_{nPop}], e.g., if $MinPts_i = 26$ and* nVar $= 8$ *then* $Pop_i = [00011010]$;
; `/Define an array for storing global information`
BestSol ← *Store the best solution (MinPts, Eps, Purity, nCluster (number of clusters), Class) in each iteration based on the maximum fitness (purity) value (Eq. 8). Initially,* BestSol =*null*;
; `/Define the population structure`
Population (i).Position ← *The position/point of each individual in the search space, $i = 1, 2, 3, ..., nPop$*;
Population (i).Cost ← *The purity value of each individual*;
Population (i).Sol ← *Store the solution of each individual including MinPts, Eps, Purity, nCluster and Class*;
for $i \leftarrow 1$ **to** nPop **do**
Population (i).Position = *randi([0 1], nVar); generate the random initial position value (called MinPts) by a binary coding (0 or 1) based on the number of nVar*;
[Population (i).Cost Population (i).Sol] = `Fitness Function(`Population *(i).Position*`)`;
end
; `/DE Main Loop`
for $it \leftarrow 1$ **to** MaxIt **do**
for $i \leftarrow 1$ **to** nPop **do**
Execute Mutation operator (Eq. 2) by an uniform distribution of scaling factor in Eq. 7;
Execute Crossover operator (Eq. 3);
Apply position limits over Population (i) by Eq. 6.;
Execute Selection operator (Eq. 4) by `Fitness Function(`Population *(i).Position*`)`;
; `/Update Best Solution of each population` i
if Population *(i).Cost* $>$ BestSol.*Cost* **then**
BestSol =Population (i);
else if *(*Population *(i).Cost* $==$ BestSol.*Cost) AND (*Population *(i).Sol.nCluster* $<$ BestSol.*Sol.nCluster)* **then**
BestSol =Population (i);
end
if BestSol.*Cost(it)==1* **then** Exit from algorithm; `/* purity==100% */`
end
end
Function *Fitness Function(MinPts)*
Calculate an analytical formula of estimating neighborhood radius (Eps) for DBSCAN by Eq. 5;
Run Tournament Selection (TS) method to select an Eps value by high probability (purity value) among stored Eps;
Run standalone DBSCAN ($MinPts, Eps$) algorithm;
Run purity function over DBSCAN results by Eq. 8;
return *MinPts, Eps, Purity and nCluster*;
end

Algorithm 3: Pseudo-code of the BDE-DBSCAN algorithm

otherwise go to step 2.
Generally, the evolution process terminates if the maximum generation is reached or the minimum/maximum fitness value is satisfied. The detailed process of BDE-DBSCAN is described in the algorithm 3. There are some definitions over the proposed algorithm as follows:

Definition 8 (*Eps Parameter*):
Since the Eps parameter can largely degrades the efficiency of the DBSCAN algorithm [37], the combination of an analytical way for estimating Eps and Tournament Selection (TS) method is employed. The TS [38] method based on its specifications creates more diverse Eps values until an appropriate combination of MinPts and Eps values be selected. The Eps parameter can be calculated by an analytical way as [45]:

$$Eps = \left(\frac{\left(\prod_{i=1}^{max(x)-min(x)} i\right) * k * \gamma(0.5 * n + 1)}{m * \sqrt{\pi^n}}\right)^{1/n} \quad (5)$$

Where x is data matrix by m-objects and n-variables, k is the number of objects in a neighborhood of an object, and γ interpolates the factorial function. In each iteration, the Eps parameter is calculated and compared to the stored Eps values from previous iterations through tournament selection method. At first, this initial Eps value has high probability by default in order to be selected in the tournament among others. It might be a very well suited selection of Eps and MinPts.

Definition 9 (*Tournament Selection*):
TS [38] is widely used selection strategy in evolutionary algorithms. It employs to select the Eps value from a population of

stored Eps values based on the fitness function (purity). The better fitness value of each Eps gets more chance to be selected. The winner of each tournament is selected for running the DBSCAN algorithm by the generated population (MinPts).

Definition 10 (*Apply limit over real-coded vectors*):
Since the standard operators generate real-coded vectors not bit strings, a simple control is proposed to values get either 0 or 1:

$$Binary - coded = min(max(Real - coded, 0), 1) \quad (6)$$

Definition 11 (*Uniform Distributions of Scaling Factor*):
For scaling factor in mutation operator, a random number with the size of $nVar$ bit strings is generated from the continuous uniform distributions with lower and upper bound specified by β_{min} and β_{max}, respectively:

$$\eta = unifrnd(\beta_{min}, \beta_{max}, nVar) \quad (7)$$

4.1 Time Complexity

The time complexity of the BDE-DBSCAN is $O((Iteration \cdot Population) \cdot (n \cdot logn))$, where $Iteration$ is the number of Differential Evolution iterations, $Population$ is the number of individuals, $n \cdot logn$ is the time complexity of the DBSCAN algorithm, and n is the number of data objects in the data set.

Table 1. Characteristics of data sets considered

Data set	classes	Size
Cluster-in-Cluster	3	1006 (250,256,500)
Cluster-in-Cluster	5	1056 (230,256,220,230,220)
Corners	4	1000 (250,250,250,250)
Crescent Full Moon	4	1200 (150,450,300,300)
Half Kernel	4	1000 (250,250,200,200)
Pin Wheel	5	1000 (200,200,200,200,200)
Semi Circular	4	1000 (250,250,250,250)
Semi Circular	8	1000 (130,120,118,132,115,135,124,126)
Outlier	5	1020 (460,20,40,460,40)
Aggregation	7	788 (45,170,102,273,34,130,34)
Compound	6	399 (50,92,38,45,158,16)
Pathbased	3	300 (110,97,93)

5. EXPERIMENTAL RESULTS AND DISCUSSIONS

5.1 Data sets

The nine 2D artificial data sets as *Cluster-inside-Cluster (K=3 and 5), Corners, Crescent Full Moon, Half Kernel, Pinwheel, Semi Circular (K=4 and 8), Outlier*, as well as Aggregation [46], Compound [47], and Pathbased [48] data sets are applied. Applied data sets consist of different unified densities such as, clusters inside clusters, multi-density, connected clusters, and well-separated densities. It allows to evaluate the effectiveness and efficiency of the BDE-DBSCAN algorithm over different shapes of data sets. These data sets are summarized in Table 1 and depicted in Fig. 1. All the experiments were performed on Intel Pentium (R) CPU 2.13 GHz with 3 GB RAM on the platform Microsoft Windows 7. We have implemented BDE-DBSCAN algorithm and artificial data sets using MATLAB software.

5.2 Performance Metric

Clustering (unsupervised classification) is usually more difficult to evaluate than a supervised approach (e.g., counting the number of

Table 2. BDE-DBSCAN with Tournament Selection

Iteration	Purity (%)	No. of Cluster	MinPts	Eps
Cluster-in-Cluster (K=3)				
1	49.70	1	274	0.48047
21	75.15	2	13	1.0004
30	92.05	5	7	0.73406
91	97.81	50	3	0.48056
133	99.10	60	2	0.48056
205	99.92	38	4	0.48056
Cluster-in-Cluster (K=5)				
1	22.145	1	121	2.8659
4	98.356	38	3	0.45127
118	99.394	42	1	0.45127
Corners				
1	53	2	760	0.89093
26	99.70	5	7	1.1763
143	100	5	2	1.1763
Crescent Full Moon				
1	62.50	2	35	2.1013
35	87.50	4	10	1.5743
45	99.75	5	5	1.1132
53	99.75	4	14	1.1132
73	99.81	10	1	1.1132
82	99.92	9	4	1.1132
Half Kernel				
1	25	1	150	11.4776
102	51.80	4	6	2.2955
108	94.20	58	2	1.3253
144	96.80	29	3	1.6232
279	97.60	35	2	1.6232
357	99.01	48	1	1.6232
Pin Wheel				
1	40.20	3	14	0.21768
17	79.50	5	7	0.15393
96	98.10	6	5	0.13009
109	98.30	8	4	0.11636
337	99	13	1	0.11636
375	99.03	11	1	0.13009
Semi Circular (K=4)				
1	78.50	9	4	1.0542
103	98.80	40	2	0.74543
124	99.60	14	3	0.91297
209	99.90	16	1	0.91297
Semi Circular (K=8)				
1	26.70	2	13	1.8214
7	40.80	4	5	1.1296
20	66.70	12	3	0.8749
96	99.40	36	2	0.7144
180	99.45	30	1	0.7445
182	99.50	21	3	0.7445
Outlier				
1	94.12	3	21	5.5989
19	98.43	6	7	3.2325
44	98.92	14	3	2.1162
70	99.91	7	5	2.732
Aggregation				
1	82.741	6	4	1.2079
2	82.741	5	19	2.6326
84	91.497	23	2	0.85414
315	92.513	32	1	0.85414
Compound				
1	84.211	4	7	1.8423
56	96.992	6	4	1.3926
89	97.74	54	3	1.45
126	98.25	56	1	1.5
Pathbased				
1	37.667	2	6	2.252
9	86	7	3	1.5924
11	93.667	56	1	0.91938
33	95	22	2	1.3002
40	98.333	31	1	1.3002

errors or true detection) [49]. We evaluate effectiveness and accuracy of BDE-DBSCAN on applied data sets (Table 1) using *Purity* criterion. The purity metric measures the goodness of formed clusters. It is also fitness function in the proposed algorithm. The purity determines the frequency of the most common category/class into each cluster:

$$Purity = \frac{1}{n}\sum_{q=1}^{k}\max_{1\leq j\leq l} n_q^j \quad (8)$$

Where, n is the total number of samples; l is the number of categories, n_q^j is the number of samples in cluster q that belongs to the original class $j(1 \leq j \leq l)$. A large purity (close to 100%) is desired for a good clustering.

5.3 Analysis

The experiments on each data sets were repeated 10 times independently to find the optimal results. Table 2 summarizes the results of BDE-DBSCAN algorithm over twelve data sets based on any improvement in the purity and in the number of clusters. Some data sets such as Cluster-inside-Cluster (K=5), Half Kernel, Semi Circular (K=8), and Pathbased have initialized with a low purity. After several iterations they reached to the optimal results. In contrast, some data sets such as Semi Circular (K=4), Outlier, Aggregation and Compound have initialized with high purity, then reached to the optimal results. In the most data sets, a same Eps value with different MinPts values lead to the optimal accuracy (maximum purity). For instance, in cluster-in-cluster (k=3) Eps=0.48056 by MinPts=3, 2 and 4 satisfy accuracy criterion by 97.81%, 99.1% and 99.9%, respectively. In corners, Eps=1.1763 by MinPts=7 and 2 satisfy the accuracy by 99.7% and 100%. Also, crescent full moon data set by Eps=1.1132 and MinPts=5, 14, 1 and 4, half kernel by Eps=1.6232 and MinPts=3, 2 and 1, pin wheel by Eps=0.13009 and MinPts=5 and 1, semi-circular (k=4) by Eps=0.91297 and MinPts=3 and 1, and semi-circular (k=8) by Eps=0.7445 and MinPts=1 and 3 perform optimal accuracy results. Table 3 shows BDE-DBSCAN without TS. The experimental results demonstrate that BDE-DBSCAN with TS performs better than without TS. It can be concluded that the combination of an analytical way for estimating Eps and TS method has considerably decreased the MinPts sensitivity to Eps value. Hence, this procedure leads to find the optimal results.

In order to compare the quality and performance of BDE-DBSCAN, five binary-encoded optimization algorithms, i.e., Binary Harmony Search (BHS), Binary Genetic Algorithm with Tournament Selection (BGA-TS), Binary Genetic Algorithm with Roulette Wheel (BGA-RW), Binary Bees Algorithm (BBees), and Binary Particle Swarm Optimization (BPSO) with the recommended parameter values were applied. The experiments on each algorithm were repeated 10 times independently to find the optimal parameter values. Table 4 lists the best final parameter settings of all the algorithms. The notation of control parameters in each algorithms is as follows. In Binary Harmony Search (BHS), HMS is harmony memory size, $nNew$ is number of the new harmonies, $HMCR$ is harmony memory consideration rate, PAR is pitch adjustment rate, FW is fret width (bandwidth), and FW_{damp} is fret width damp ratio. In Genetic Algorithm with Tournament Selection (GA-TS), pc is crossover percentage, nc is number of Offsprings (Parents), pm denotes mutation percentage, nm denotes number of mutants, mu denotes mutation rate. In Genetic Algorithm with Roulette Wheel (GA-RW), β denotes selection pressure parameter and other settings are same as GA-TS. In Binary Bees (BBees) algorithm, $nScoutBee$ is number of scout bees, $nBee0$ is recruited

Table 3. BDE-DBSCAN without Tournament Selection

Iteration	Purity (%)	No. of Cluster	MinPts	Eps
Cluster-in-Cluster (K=3)				
1	75.15	2	13	1.0004
39	92.048	5	7	0.73406
50	97.813	50	3	0.48056
106	98.211	39	4	0.5549
Cluster-in-Cluster (K=5)				
1	22.145	1	71	7.6892
51	66.09	13	6	0.63819
112	91.52	18	5	0.58259
127	98.356	38	3	0.45127
Corners				
1	25	1	171	5.8141
126	84.90	198	1	0.44462
136	99.30	5	5	0.99419
204	99.70	5	7	1.1763
Crescent Full Moon				
1	62.50	2	89	4.6966
13	87.333	4	7	1.3172
33	99.083	19	3	0.86229
54	99.667	7	4	0.99568
101	99.75	5	5	1.1132
Half Kernel				
1	25	1	346	17.4318
51	91.30	134	1	0.93714
67	96.80	29	3	1.6232
Pin Wheel				
1	20.40	2	20	0.26018
18	59.80	4	8	0.16455
55	96.80	9	3	0.10077
96	98.10	6	5	0.13009
Semi Circular (K=4)				
1	25	1	871	15.5562
87	99.60	14	3	.91297
Semi Circular (K=8)				
1	13.50	1	432	10.4994
20	26.70	2	10	1.5974
44	99.40	37	2	0.7144
Outlier				
1	94.118	3	23	5.8594
2	97.941	5	11	4.0522
6	99.902	8	5	2.732
Aggregation				
1	78.426	4	29	3.2525
3	82.741	5	12	2.0922
81	91.497	23	2	0.85414
Compound				
1	81.955	4	8	1.9695
5	84.211	4	7	1.8423
89	96.992	6	4	1.3926
Pathbased				
1	81.667	4	4	1.8388
9	95	22	2	1.3002

Table 4. Parameter settings of applied algorithms

Algorithm	Control Parameters
BDE-DBSCAN	$nVar$=10, $MaxIt$=500, $nPop$=25, β_{min}=0.2, β_{max}=0.8, pCR=0.25
BHS	HMS=20, $HMCR$=0.5, PAR=0.1, FW=0.02, FW_{damp}=0.995
GA-TS	$nVar$=10, $MaxIt$=500, $nPop$=25, pc=0.7, $nc = 2 * round(pc * nPop/2)$, pm=0.2, $nm = round(pm * nPop)$, mu=0.02
GA-RW	$nVar$=10, $MaxIt$=500, $nPop$=25, pc=0.8, $nc = 2 * round(pc * nPop/2)$, pm=0.25, $nm = round(pm * nPop)$, mu=0.025, β=8
BBees	$nVar$=6, $MaxIt$=500, $nScoutBee$=20, $nBee$0=round(0.3*nScoutBee), r=0.2, $rdamp$=0.98
BPSO	$nVar$=8, $MaxIt$=600, $nPop$=20, ϕ=4.1, w=0.73, $wdamp$=1 and it is linearly decreased by $w * wdamp$ in each iteration

bees scale, r is neighborhood radius, and $rdamp$ is neighborhood radius damp rate. In Binary Particle Swarm Optimization (BPSO), ϕ denotes correlation coefficient, w denotes inertia weight, and $wdamp$ denotes inertia weight damping ratio.

In order to validate the BDE-DBSCAN performance, we use some measures including purity (Eq. 8), entropy (Eq. 10), Davies-Bouldin Index (DBI) (Eq. 11) and Dunn index (Eq. 12). Entropy [50, 51, 52] is the degree to which each cluster consists of objects of a single class. For each cluster, the class distribution of the data is calculated by $p_{ij} = \frac{m_{ij}}{m_i}$, where m_i is the number of objects in cluster i, m_{ij} is the number of objects of class j in cluster i, and p_{ij} denotes the probability of a member of cluster i belongs the class j. The entropy of each cluster i is calculated using the standard formula:

$$e_i = -\sum_{j=1}^{L} p_{ij} \cdot \log_2 p_{ij} \tag{9}$$

Where L is the number of classes. The total entropy for a set of clusters is calculated by:

$$e = -\sum_{j=1}^{K} \frac{m_i}{m} \cdot e_i \tag{10}$$

Where K is the number of clusters and m is the total number of data points. The DBI [53] calculates the similarities between each cluster C and other clusters, and the highest value is assigned to C as its cluster similarity. Then DBI measures the average of similarity between each cluster. As the clusters should be compacted and separated, the lower DB means better clustering result.

$$\frac{1}{NC}\sum_{i} max_{j,j\neq i} \frac{[\frac{1}{n_i}\sum_{x\epsilon C_i} d(x,c_i) + \frac{1}{n_j}\sum_{x\epsilon C_j} d(x,c_j)]}{d(c_i,c_j)} \tag{11}$$

Where, *c: center of data set; NC: number of clusters; C_i: the i-th cluster; n_i: number of objects in C_i; c_i: center of C_i; $d(x,y)$: distance between x and y.*

Dunn index [54] is based on the minimum pairwise distance between objects in different clusters as the inter-cluster separation and the maximum diameter among all clusters as the intra-cluster compactness. The larger value of Dunn means better cluster configuration.

$$min_i\{min_j(\frac{min_{x\epsilon C_i, y\epsilon C_j} d(x,y)}{max_k\{max_{x,y\epsilon C_k} d(x,y)\}})\} \tag{12}$$

Where, *C_i: the i-th cluster; n_i: number of objects in C_i; $d(x,y)$: distance between x and y.*

Table 5 shows the comparison results over applied methods. As shown, BDE-DBSCAN outperforms BHS, BGA-TS, BGA-RW, BBees, and BPSO on almost all the tests in terms of the purity, entropy, DBI and Dunn criteria.

For the time complexity measurement, we should take into consideration two different complexities including $O(n \cdot logn)$ and $O(Iteration \cdot Population)$. It implies that $O(n \cdot logn)$ comes from DBSCAN algorithm and $O(Iteration \cdot Population)$ comes from BDE-DBSCAN. Based on the obtained results, the optimal accuracy will be reached by the number of population between 20 and 30 and the number of iteration between 200 and 500. Hence, the most influential parameter is given by n (number of the data points in considered data sets). It demonstrates that the added extra time complexity to DBSCAN algorithm can be reasonable for finding very well suited MinPts and Eps values at the same time.

5.4 Accuracy Improvement

As seen in Table 2, BDE-DBSCAN shows uniformly high accuracy ranging around 99.4%-100% for all data sets except half kernel (accuracy=99.01%), pin wheel (accuracy=99.03%), aggregation (accuracy=92.513%), compound (accuracy=98.25%), and pathbased (accuracy=98.333%). Since these data sets consist of border data objects with absolutely close clusters and connected clusters, we should apply a modified DBSCAN algorithm to tackle successfully with these types of data sets. The assumption for optimal accuracy is the purity with more than 99.3%. By applying any modified version of DBSCAN, the accuracy can be considerably improved. We applied the three revised versions of DBSCAN algorithm from papers [11], [10] and [13] in *Fitness Function* section of algorithm 3. According to Table 6, applied modified DBSCAN algorithms outperform almost all the standalone DBSCAN algorithms in terms of the optimal accuracy. It shows that BDE-DBSCAN algorithm by integration of any revised DBSCAN which can tackle with multi-density data sets and close connected clusters, can provide better combination of Eps and MinPts parameters in the same time.

5.5 Discussion

This paper presented a new hybrid approach combining Binary Differential Evolution (BDE) and DBSCAN clustering algorithm called BDE-DBSCAN. This is a meta-heuristic population-based algorithm to find the optimal initial parameters (Eps and MinPts) of the DBSCAN algorithm. BDE-DBSCAN tries to find the very well suited Eps value for each individual (MinPts) in DE. To do so, a combination of an analytical way of estimating Eps and tournament selection method is employed. Tables 2 and 3 confirm that tournament selection can provide optimal results in side of an analytical way. We also applied standalone DBSCAN version for clustering purposes. However, for relatively closed clusters and densities, we applied three revised version of DBSCAN from literature to improve BDE-DBSCAN accuracy reasonable. The time complexity of BDE-DBSCAN consists of two phases: $O(n \cdot logn)$ from DBSCAN algorithm and $O(Iteration \cdot Population)$ from DE. Experimental results imply that the added time complexity to DBSCAN algorithm is reasonable for accuracy improvement.

Convergence of BDE-DBSCAN is studied for finding the optimal

Table 5. The comparison results of some optimization algorithms

Algorithm	Purity (%)	Entropy	Dunn (max)	DBI (min)	No. of Cluster	MinPts	Eps
Cluster-in-Cluster (K=3)							
BDE-DBSCAN	99.9	0.0831	1.3628	1.0442	38	4	0.4806
BHS	98.211	0.1817	0.9935	1.0761	39	4	0.5549
GA-TS	99.601	0.0926	1.3451	1.0543	71	2	0.4616
GA-RW	98.356	0.1784	1.2141	1.7033	38	3	0.4513
BBees	98.211	0.1753	0.9705	1.0772	39	4	0.5549
BPSO	98.356	0.1681	1.3012	1.0812	38	3	0.4513
Cluster-in-Cluster (K=5)							
BDE-DBSCAN	99.394	0.1217	1.4112	1.0452	42	1	0.4513
BHS	97.664	0.1663	0.9718	1.0658	25	4	0.5108
GA-TS	98.356	0.1455	1.234	1.0545	38	3	0.4513
GA-RW	98.356	0.1405	1.2802	1.0528	38	3	0.4513
BBees	98.356	0.1508	1.3384	1.0572	38	3	0.4513
BPSO	98.356	0.1527	1.3237	1.0567	38	3	0.4513
Corners							
BDE-DBSCAN	100	0	1.5	1.0401	5	2	1.1763
BHS	99.4	0.1104	1.2122	1.0619	5	6	1.0891
GA-TS	99.8	0.1009	1.2831	1.0598	5	6	1.0854
GA-RW	99.7	0.1088	1.2611	1.0518	5	7	1.1763
BBees	99.4	0.1135	1.314	1.0621	5	6	1.0891
BPSO	99.8	0.1006	1.1915	1.0584	5	6	1.0854
Crescent Full Moon							
BDE-DBSCAN	99.92	0.0724	1.4125	1.0511	9	4	1.1132
BHS	99.75	0.0962	1.2831	1.0621	5	5	1.1132
GA-TS	99.917	0.0798	1.3341	1.0602	7	5	1.1098
GA-RW	99.75	0.0944	1.3018	1.0612	5	5	1.1132
BBees	99.667	0.0933	1.2517	1.0591	7	4	0.9957
BPSO	99.75	0.0957	1.284	1.0582	5	5	1.1132
Half Kernel							
BDE-DBSCAN	99.01	0.1234	1.4981	1.0601	48	1	1.6232
BHS	94.2	0.2217	1.2183	1.1108	58	2	1.3253
GA-TS	97.6	0.1604	1.2703	1.0832	35	2	1.6232
GA-RW	96.8	0.1782	1.2219	1.0842	29	3	1.6232
BBees	91.3	0.2459	1.1418	1.1318	134	1	0.4371
BPSO	94.2	0.2228	1.2712	1.129	58	2	1.3253
Pin Wheel							
BDE-DBSCAN	99.03	0.1298	1.5018	1.0581	11	1	0.1301
BHS	98.3	0.1914	1.3211	1.071	8	4	0.1164
GA-TS	99	0.1611	1.479	1.0609	18	1	0.1029
GA-RW	98.3	0.1962	1.1927	1.0612	8	4	0.1164
BBees	98.1	0.1744	1.1981	1.0708	6	5	0.1301
BPSO	98.6	0.1644	1.1928	1.0707	7	4	0.1301
Semi Circular (K=4)							
BDE-DBSCAN	99.9	0.0795	1.4114	1.0523	16	1	0.9130
BHS	99.6	0.0997	1.2119	1.0689	14	3	0.9130
GA-TS	99.6	0.0976	0.9912	1.0671	14	3	0.9130
GA-RW	99.6	0.098	0.9988	1.0695	14	3	0.9130
BBees	98.8	0.1995	0.9217	1.0741	40	2	0.7454
BPSO	99.6	0.0982	1.3091	1.0708	14	3	0.9130
Semi Circular (K=8)							
BDE-DBSCAN	99.5	0.1114	1.3119	1.0612	21	3	0.7445
BHS	93.4	0.2217	0.8021	1.1125	140	1	0.5051
GA-TS	99.4	0.118	1.2181	1.0693	36	2	0.7144
GA-RW	99.4	0.1224	1.2094	1.0688	36	2	0.7144
BBees	93.4	0.2341	0.8192	1.1218	140	1	0.5051
BPSO	99.4	0.1151	1.1445	1.0874	36	2	0.7144
Outlier							
BDE-DBSCAN	99.91	0.0698	1.4012	1.0462	7	5	2.7320
BHS	98.922	0.1711	1.292	1.0641	14	3	2.1162
GA-TS	99.902	0.0888	1.2987	1.0514	8	5	2.7320
GA-RW	99.902	0.0987	1.3719	1.0645	10	4	2.4436
BBees	98.922	0.1823	1.3448	1.0557	14	3	2.1162
BPSO	99.902	0.0767	1.3842	1.0524	10	4	2.4436
Aggregation							
BDE-DBSCAN	92.513	0.452	2.1314	1.2543	32	1	0.8541
BHS	82.741	0.7825	1.2491	1.3294	6	4	1.2079
GA-TS	82.741	0.7654	1.8835	1.3487	5	9	1.8119
GA-RW	91.497	0.5027	1.9947	1.2948	23	2	0.8541
BBees	91.497	0.5542	1.9312	1.3128	23	2	0.8541
BPSO	91.497	0.512	1.9881	1.2891	23	2	0.8541

it is continuing on the next page ...

it comes from the previous page ...

Algorithm	Purity (%)	Entropy	Dunn (max)	DBI (min)	No. of Cluster	MinPts	Eps
Compound							
BDE-DBSCAN	98.25	0.1761	1.4012	1.0792	56	1	1.5
BHS	96.992	0.1945	0.9871	1.0924	6	4	1.3926
GA-TS	96.992	0.1978	0.9745	1.0931	6	4	1.3926
GA-RW	96.992	0.1986	0.9712	1.0944	6	4	1.3926
BBees	92.732	0.2242	0.7731	1.1391	6	3	1.206
BPSO	97.74	0.1814	1.1044	1.0819	54	3	1.45
Pathbased							
BDE-DBSCAN	98.333	0.1746	1.2517	1.0714	31	1	1.3002
BHS	93.667	0.2167	0.8618	1.0911	56	1	0.9194
GA-TS	95	0.1876	0.9984	1.0814	22	2	1.3002
GA-RW	95	0.1894	1.1012	1.0845	22	2	1.3002
BBees	95	0.1903	1.1191	1.0823	22	2	1.3002
BPSO	95	0.1901	1.1018	1.0812	22	2	1.3002

accuracy (maximum purity) over twelve data sets with variety of shapes and densities. The performance measurement results show that the BDE-DBSCAN algorithm performs well with optimal accuracy over different data sets. Tables 2 and 6 show results for finding the optimal accuracy by very well suited Eps and MinPts, where BDE-DBSCAN performs well.

The feasibility and efficiency of BDE-DBSCAN for optimization of different examples are compared to five different algorithms. Table 5 depicts the final results, using Binary Harmony Search (BHS), Genetic Algorithm with Tournament Selection (GA-TS), Genetic Algorithm with Roulette Wheel (GA-RW), Binary Bees (BBees) algorithm, and Binary Particle Swarm Optimization (BPSO). BDE-DBSCAN provides better accuracy based on the purity, entropy, Davies-Bouldin Index (DBI) and Dunn index among others. Moreover, BDE-DBSCAN and other applied algorithms use different parameter settings to find the global results; therefore, the effect of tuning parameters on performance of the algorithms are studied. Results show BDE-DBSCAN is capable for finding globally optimal accuracy in a relatively small number of generations.

Some improvements are needed in future works. First, utilization of various cluster validity measures; second, hybridization of other metaheuristics techniques with DBSCAN, and third, evaluation of the proposed method with multi-dimensional data sets.

Table 6. The purity of revised versions of DBSCAN

Data set	Revised [11]	Revised [10]	Revised [13]
Half Kernel	99.57%	99.25%	99.33%
Pin Wheel	99.81%	99.12%	99.12%
Aggregation	97.42%	96.95%	97.93%
Compound	99.75%	99.41%	99.44%
Pathbased	99.62%	98.96%	99.18%

6. CONCLUSION

This paper proposes a novel hybrid approach consisting Binary Differential Evolution (BDE) and DBSCAN clustering algorithm as BDE-DBSCAN to choose quickly and automatically very well suited Eps and MinPts parameters for DBSCAN algorithm. BDE-DBSCAN performance is evaluated using various data sets with different densities and shapes. Moreover, BDE-DBSCAN outperforms other algorithms such as Binary Harmony Search (BHS), Binary Bees (BBees) algorithm, Binary Genetic Algorithm with Tournament Selection (GA-TS) and with Roulette Wheel (GA-RW), and Binary Particle Swarm Optimization (BPSO) in terms of the purity, entropy, Davies-Bouldin and Dunn indices. As illustrated results over applied data sets, the proposed algorithm provides optimal accuracy by the purity ranging around 99.4% and 100%.

7. REFERENCES

[1] Krista Rizman Žalik. An efficient ḱ-means clustering algorithm. *Pattern Recognitoin Letters*, 29(9):1385–1391, 2008.

[2] Tran Manh Thang and Juntae Kim. The anomaly detection by using dbscan clustering with multiple parameters. In *International Conference on Information Science and Applications (ICISA)*, pages 1–5, 2011.

[3] M. Ester, H. P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceeding of 2nd International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[4] T. Ali, S. Asghar, and N.A. Sajid. Critical analysis of dbscan variations. In *International Conference on Information and Emerging Technologies (ICIET)*, pages 1–6, 2010.

[5] Wu Ying, Yang Kai, and Zhang Jianzhong. Using dbscan clustering algorithm in spam identifying. In *2nd International Conference on Education Technology and Computer (ICETC)*, volume 1, pages V1–398–V1–402, 2010.

[6] Derya Birant and Alp Kut. St-dbscan: An algorithm for clustering spatialtemporal data. *Data & Knowledge Engineering*, 60(1):208–221, 2007.

[7] P. Viswanath and V. Suresh Babu. Rough-dbscan: A fast hybrid density based clustering method for large data sets. *Pattern Recognition Letters*, 30(16):1477–1488, 2009.

[8] Amin Karami. Data clustering for anomaly detection in content-centric networks. *International Journal of Computer Applications*, 81(7):1–8, November 2013. Published by Foundation of Computer Science, New York, USA.

[9] Guilherme Andrade, Gabriel Ramos, Daniel Madeira, Rafael Sachetto, Renato Ferreira, and Leonardo Rocha. G-dbscan: A gpu accelerated algorithm for density-based clustering. *Procedia Computer Science*, 18:369–378, 2013.

[10] G.H. Shah. An improved dbscan, a density based clustering algorithm with parameter selection for high dimensional data sets. In *Nirma University International Engineering (NUiCONE)*, pages 1–6, 2012.

[11] Thanh N. Tran, Klaudia Drab, and Michal Daszykowski. Revised dbscan algorithm to cluster data with dense adjacent clusters. *Chemometrics and Intelligent Laboratory Systems*, 120:92–96, 2013.

[12] Hua Jiang, Jing Li, Shenghe Yi, Xiangyang Wang, and Xin Hu. A new hybrid method based on partitioning-based dbscan and ant clustering. *Expert Systems with Applications*, 38(8), 2011.

[13] Chen Xiaoyun, Min Yufang, Zhao Yan, and Wang Ping. Gmdbscan: Multi-density dbscan cluster based on grid. In *IEEE International Conference on e-Business Engineering (ICEBE '08)*, pages 780–783, 2008.

[14] Li Xue-yong, Gao Guo-hong, and Sun Jia-xia. A new intrusion detection method based on improved dbscan. In *International Conference on Information Engineering (ICIE)*, volume 2, pages 117–120, 2010.

[15] A. Tepwankul and S. Maneewongwattana. U-dbscan : A density-based clustering algorithm for uncertain objects. In *26th IEEE International Conference on Data Engineering Workshops (ICDEW)*, pages 136–143, 2010.

[16] Damodar Reddy Edla and Prasanta K. Jana. A prototype-based modified dbscan for gene clustering. *Procedia Technology*, 6:485–492, 2012.

[17] Huang Darong and Wang Peng. Grid-based dbscan algorithm with referential parameters. *Physics Procedia*, 24, Part B:1166–1170, 2012.

[18] A. Smiti and Z. Elouedi. Dbscan-gm: An improved clustering method based on gaussian means and dbscan techniques. In *16th International Conference on Intelligent Engineering Systems (INES)*, pages 573–578, 2012.

[19] G. P. Babu and M. N. Murty. Simulated annealing for selecting optimal initial seeds in the k-means algorithm. *Indian Journal of Pure and Applied Mathematics*, 25(1-2):85–94, 1994.

[20] Amin Karami and Manel Guerrero-Zapata. A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks. *Neurocomputing*, 2014.

[21] George E. Tsekouras. A simple and effective algorithm for implementing particle swarm optimization in rbf network's design using input-output fuzzy clustering. *Neurocomputing*, 108:36–44, 2013.

[22] Junyan Chen. Hybrid clustering algorithm based on pso with the multidimensional asynchronism and stochastic disturbance method. *Journal of Theoretical and Applied Information Technology*, 46(1):434–440, 2012.

[23] Zhenkui Pei, Xia Hua, and Jinfeng Han. The clustering algorithm based on particle swarm optimization algorithm. In *Proceedings of the 2008 International Conference on Intelligent Computation Technology and Automation - Volume 01*, ICICTA '08, pages 148–151, Washington, DC, USA, 2008. IEEE Computer Society.

[24] Li Lin and Liu Suhua. Wheat cultivar classifications based on tabu search and fuzzy c-means clustering algorithm. In *4th International Conference on Computational and Information Sciences (ICCIS)*, pages 493–496, 2012.

[25] Hong bing Xu, Hou jun Wang, and Chun-Guang Li. Fuzzy tabu search method for the clustering problem. In *Proceedings International Conference on Machine Learning and Cybernetics*, volume 2, pages 876–880, 2002.

[26] Yangyang Li, Jing Chen, Ruochen Liu, and Jianshe Wu. A spectral clustering-based adaptive hybrid multi-objective harmony search algorithm for community detection. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2012.

[27] C. Cobos, J. Andrade, W. Constain, M. Mendoza, and E. León. Web document clustering based on global-best harmony search, k-means, frequent term sets and bayesian information criterion. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2010.

[28] Liang Li, Shi bao Lu, Xue song Chu, and Guang ming Yu. A combinatorial search method based on harmony search algorithm and particle swarm optimization in slope stability analysis. In *International Conference on Computational Intelligence and Software Engineering (CiSE)*, pages 1–4, 2009.

[29] E. Hancer, C. Ozturk, and D. Karaboga. Artificial bee colony based image clustering method. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1–5, 2012.

[30] D. Karaboga, S. Okdem, and C. Ozturk. Cluster based wireless sensor network routings using artificial bee colony algorithm. In *International Conference on Autonomous and Intelligent Systems (AIS)*, pages 1–5, 2010.

[31] Y. Marinakis, M. Marinaki, and N. Matsatsinis. A hybrid discrete artificial bee colony - grasp algorithm for clustering. In *International Conference on Computers Industrial Engineering (CIE)*, pages 548–553, 2009.

[32] Bo Zhao, Zhongxiang Zhu, Enrong Mao, and Zhenghe Song. Image segmentation based on ant colony optimization and k-means clustering. In *IEEE International Conference on Automation and Logistics*, pages 459–463, 2007.

[33] Yanfang Han and Pengfei Shi. An improved ant colony algorithm for fuzzy clustering in image segmentation. *Neurocomputing*, 70(46):665–671, 2007.

[34] J. W. Han and M. Kamber. *Data Mining Concepts and Techniques, second edition*. Morgan Kaufmann Publishers, 2006.

[35] Stanisaw Sieniutycz and Jacek Jeowski. 1 - brief review of static optimization methods. In *Energy Optimization in Process Systems and Fuel Cells (Second Edition)*, pages 1–43. Elsevier, Amsterdam, second edition edition, 2013.

[36] Yang Sun, Lingbo Zhang, and Xingsheng Gu. A hybrid co-evolutionary cultural algorithm based on particle swarm optimization for solving global optimization problems. *Neurocomputing*, 98(3):76–89, 2012.

[37] Hongfang Zhou, Peng Wang, and Hongyan Li. Research on adaptive parameters determination in dbscan algorithm. *Journal of Information & Computational Science*, 9(7):1967–1973, 2012.

[38] Mohammed Azmi Al-Betar, Ahamad Tajudin Khader, Zong Woo Geem, Iyad Abu Doush, and Mohammed A. Awadallah. An analysis of selection methods in memory consideration for harmony search. *Applied Mathematics and Computation*, 219(22):10753–10767, 2013.

[39] R. Storn and K. Price. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[40] Dexuan Zou, Jianhua Wu, Liqun Gao, and Steven Li. A modified differential evolution algorithm for unconstrained optimization problems. *Neurocomputing*, 120:469–481, 2013.

[41] Marco Locatelli, Mirko Maischberger, and Fabio Schoen. Differential evolution methods based on local searches. *Computers & Operations Research*, 43:169–180, 2014.

[42] Yang Lou, Junli Li, and Yongsheng Wang. A binary-differential evolution algorithm based on ordering of individuals. In *6th International Conference on Natural Computation (ICNC)*, pages 2207–2211, 2010.

[43] Youyun Ao and Hongqin Chi. Experimental study on differential evolution strategies. In *Global Congress on Intelligent Systems*, pages 19–24, 2009.

[44] Ling Wang, Xiping Fu, Yunfei Mao, Muhammad Ilyas Menhas, and Minrui Fei. A novel modified binary differential evolution algorithm and its applications. *Neurocomputing*, 98:55–75, 2012.

[45] M. Daszykowski, B. Walczak, and D. L. Massart. Looking for natural patterns in data: Part 1. density-based approach. *Chemometrics and Intelligent Laboratory Systems*, 56:83–92, 2001.

[46] A. Gionis, H. Mannila, and P. Tsaparas. Clustering aggregation. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 1(1):1–30, 2007.

[47] C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 100(1):68–86, 1971.

[48] H. Chang and D.Y. Yeung. Robust path-based spectral clustering. *Pattern Recognition*, 41(1):191–203, 2008.

[49] Sameh A. Salem and A.K. Nandi. New assessment criteria for clustering algorithms. In *IEEE Workshop on Machine Learning for Signal Processing*, pages 285–290, 2005.

[50] Anil K. Jain and Richard C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[51] Amin Karami. Utilization and comparison of multi attribute decision making techniques to rank bayesian network options. master thesis, University of Skövde, Skövde, Sweden, 2011.

[52] Amin Karami and Ronnie Johansson. Utilization of multi attribute decision making techniques to integrate automatic and manual ranking of options. *Journal of Information Science and Engineering*, 30(2):519–534, 2014.

[53] David L. Davies and Donald W. Bouldin. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227, 1979.

[54] J.C. Dunn. Well separated clusters and optimal fuzzy partitions. *Cybernetics*, 4:95–104, 1974.