

A Self Organizing Map Based Multi-objective Framework for Automatic Evolution of Clusters

Naveen Saini¹(✉), Shubham Chourasia², Sriparna Saha¹,
and Pushpak Bhattacharyya¹

¹ Indian Institute of Technology, Patna, India

naveen.pcs16@iitp.ac.in, nsaini1988@gmail.com

² Sikkim Manipal Institute of Technology, East Sikkim, India

Abstract. The current paper reports about the development of an automatic clustering technique which builds upon the search capability of a self-organizing multi-objective differential evolutionary approach. The algorithm utilizes new search operators which are developed after considering the neighborhood relationships of solutions of a population extracted using a self organizing map (SOM). Variable number of cluster centers are encoded in different solutions of the population which are evolved using the new search operators of differential evolution to automatically determine the number of clusters. Two cluster validity indices capturing different goodness measures of partitioning are used as objective functions. The effectiveness of the proposed framework namely, self organizing map based multi-objective (MO) clustering technique (SMEA_clust) is shown for automatically partitioning four artificial and four real-life data sets in comparison with a multi-objective differential evolution based clustering technique (similar to our proposed approach but without using SOM concept), two recent multi-objective clustering based techniques, VAMOSA and MOCK. Results are further validated using statistical significance tests.

Keywords: Self organizing map · Unsupervised classification · Clustering · Evolutionary algorithm · Cluster validity measures

1 Introduction

Clustering [1], an unsupervised approach of pattern classification, aims to partition the given data set into various K-groups where members belonging to a particular group should have similar properties. Here the value of K may or may not be known *a priori*. Traditional approaches [2] of determining the appropriate number of clusters mainly apply different clustering algorithms like K-means [1], hierarchical [1] approaches for different values of K and then select the optimal partitioning based on the value of a cluster validity index, measuring the goodness of a partitioning. Most of the existing clustering algorithms implicitly optimize a single cluster quality measure capturing different cluster properties

like compactness, separation, connectivity, density or symmetry. But in real-life, clusters of a given data set possess different structures which can not be detected with the use of a single cluster quality measure. Thus the optimization of multiple cluster quality measures simultaneously using the search capacity of a multiobjective optimization algorithm [3,4] became popular in recent years. This leads to the development of some multi-objective based clustering algorithms [5]. Moreover automatic optimization of cluster validity indices also guides the algorithm to automatically determine the number of clusters. Most of the existing multi-objective clustering approaches utilize different evolutionary techniques (EAs) like genetic algorithm (GA) [6], differential evolution (DE) [7] or particle swarm optimization (PSO) [8] as the underlying optimization strategy.

In [9], Handl et al. developed a multi-objective automatic clustering algorithm, MOCK. But the application of MOCK is limited in determining only some well-separated or hyper-spherical shaped clusters. It does not perform well in detecting overlapping clusters. Moreover the complexity of MOCK increases linearly with the increase in the number of data points. In recent years some symmetry based multiobjective clustering techniques are proposed in [10,11] utilizing AMOSA [3] as the underlying optimization technique. Suresh et al. [7] have proposed some multi-objective differential evolution based clustering techniques and have shown that differential evolution can serve as a promising backbone for devising multi-objective clustering techniques. In general it was shown in the literature that DE based approaches converge faster compared to other EA based techniques [7]. Motivated by this, current paper also exploits the search capability of DE in proposing an automatic clustering technique in collaboration with self organizing map (SOM) [12].

Self organizing map (SOM) is an unsupervised learning method of artificial neural network to map high dimensional input space to low dimensional output space preserving the topological structure of the input data. In recent years researchers are working towards integrating SOM with evolutionary algorithms to develop some new optimization techniques like SMOEA/D [13], SMEA [14]. Motivated by this in the current paper we have developed a self organizing evolutionary algorithm based clustering technique where at each generation, a new solution is generated using the neighborhood relationship identified by SOM trained on solutions present in the current population as done in SMEA [14]. For a given solution, only the neighboring solutions in the topographical map identified by SOM are participated in the genetic operations of DE to generate high quality solutions. The proposed clustering approach simultaneously optimizes two cluster validity indices, PBM index [15] and Silhouette score [11] for the purpose of automatic determination of the number of clusters.

Similar to any other MOO based approach our proposed clustering approach, SMEA_clust, also produces a set of solutions at the end on the final Pareto optimal front. A single best solution is selected using an external cluster validity index namely Adjusted Rand Index (ARI) [7]. The supremacy of the proposed approach is shown on automatically partitioning four artificial and four real-life data sets of varying complexities. The obtained partitioning results are

compared with some existing MOO based clustering techniques like MOCK [9] and VAMOSa [10]. In a part of the paper, in order to show the adequacy of SOM based genetic operators, experiments are also performed with normal genetic operators in association with MOO based clustering approach developed in the paper, namely MEA_clust. Results clearly show the strength of using SOM in mating pool generation process of the proposed MOO based clustering approach. This clearly illustrates that participation of nearest neighbors of a solution in generating the mating pool helps in exploring the search space in an efficient way. Furthermore in order to validate the obtained results, some statistical tests guided by [16] are also conducted to show the superiority of our algorithm.

2 Methodology

In the current work we have proposed a self organizing evolutionary algorithm based clustering technique. The key attributes of the current work are the following: (i) a recently developed self organizing multi-objective evolutionary algorithm [14] is utilized as the underlying optimization technique. Here neighborhood information extracted by a SOM for a given solution is used during genetic operations. This helps in exploring the search-space steadily. (ii) optimization of two cluster validity indices, PBM index [15] and Silhouette score [11], simultaneously enables the proposed clustering approach to automatically determine the appropriate partitioning from a given data set. A flow-chart of the proposed approach is shown in Fig. 2 and the basic operations are described below.

2.1 Self Organizing Map

Self Organizing Map or SOM developed by Kohonen [12] is a special type of artificial neural network which makes use of unsupervised learning to produce a low-dimensional mapping of input space of the training samples. Low-dimensional space (output space) consists of regular, usually 2-D grid of neurons (or map units or prototype vectors). Neurons in low dimensional space have *position vectors* and *weight vectors* associated with them.



Fig. 1. Traditional approach for SOM training. Here η_0 : initial learning rate, σ_0 : initial neighborhood size, η and σ are continuously decreasing functions at each iteration, i_{max} is the maximum number of iterations for SOM training

Main objective of SOM is to create a topographic preserving map in the output space such that different regions of the output space respond similarly

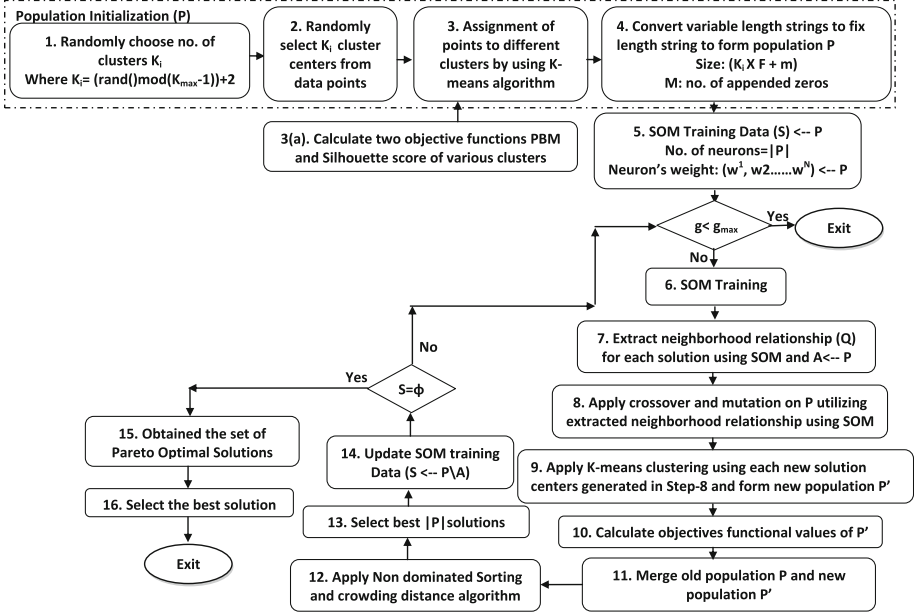


Fig. 2. SMEA based proposed architecture for multi-objective clustering. Here, g_{max} is the maximum number of generation, F is no. of features in data

to certain input patterns. Learning of SOM starts by assigning each neuron a weight vector, a randomly chosen data point from the available training samples. At each iteration, the weight vectors of the neurons are updated by using the training samples closer to them. In this work, we have utilized the traditional learning algorithm [17] as shown in Fig. 1 for training of SOM using neighborhood relationships among neurons which returns the weight vectors of different neurons.

2.2 String Representation and Population Initialization

In SMEA based clustering, the strings (solutions or chromosomes) are made up of real numbers which represent the coordinates of cluster centers. Our algorithm attempts to find the appropriate number of cluster centers that represent the partitioning of the data. Thus the number of cluster centers present in different chromosomes varies over a range, 2 to \sqrt{N} , where N is the total number of data points. For i th chromosome a random number (K_i) in the range $[2, \sqrt{N}]$ is generated and then K_i number of data points are randomly selected from the data set which are considered as the initial cluster centers. But as these chromosomes further take part in training of SOM to generate the neighborhood information, we have converted the variable length chromosomes to some fixed length vectors. Reason for keeping the fix length for all solutions is that weight vector corresponding to a neuron of SOM must be of same length as input

vector (here solutions are the input vectors). Maximum length of the string can be $(K * d + m)$, where K is the number of clusters present in a solution, d is the number of features in the data, m is the number of appended zeros which can vary between '0' to $(K * d - 2 * d)$ because of the constraint of having minimum two clusters. This means string length should be $\sqrt{N} * d$. For an example, if a particular string encodes the centers of 3 clusters in 2-d space and $N = 16$, then string will be represented as $\langle 2.31.47.612.92.13.40.00.0 \rangle$ which encodes three cluster centers (2.3, 1.4), (7.6, 12.9) and (2.1, 3.4), where m is 2. This set of strings with varying number of clusters will form the initial population.

2.3 Euclidean Distance Calculation and Weight Updation

The vectors (strings) present in the population further take part in SOM training to find the neighborhood of all these solutions. As string length is fixed in the population, during Euclidean distance calculation between training vector and weight vector of neuron, only that much length of the vector is considered which has the maximum number of appended zeros. For example, let one vector be (a, b, c, d, 0, 0) and second vector be (a, b, 0, 0, 0, 0) then during distance calculation or weight updation, consider the length of vector (or number of features) as 2 and consider the remaining features as zero because second vector has maximum number of appended zeros.

2.4 Objective Functions

In this paper, two well known internal cluster validity measures, PBM (Pakhira-Bandyopadhyay-Maulik) index [15] and Silhouette score [11] are used as objective functions which need to be simultaneously optimized. These two objective functions measure the compactness and the separation of the partitionings in term of the Euclidean distance in two different ways and should be maximized.

Algorithm 1. MatingPool_Construction($M, P, H, \beta, \overline{x_{current}}$)

- 1: Find out winning neuron 'h' mapped by solution $\overline{x_{current}}$ based on minimum Euclidean distance.
- 2: Sort h th row of M in ascending order and keep the sorted indices in I .

$$Q = \begin{cases} \cup_{k=1}^H \{\overline{x^k}\} & \text{if } \text{rand}() < \beta, \text{ and } k < H \\ P & \text{otherwise} \end{cases}$$

Where $\text{rand}()$ is a random number generated between 0 and 1. $\overline{x^k}$ is the k th neuron present in list I .

- 3: **return** Q for solution $\overline{x_{current}}$
-

Algorithm 2. $y = \text{Generate}(\overline{Q}, CR, MP, \overline{x_{current}})$

- 1: Randomly select two parents $\overline{x^1}$ and $\overline{x^2}$ from mating pool Q of solution $\overline{x_{current}}$ such that $\overline{x^1} \neq \overline{x^2} \neq \overline{x_{current}}$
- 2: Generate a trial solution $\overline{y'}$ using the following

$$\text{if rand}() \leq CR, \text{ then } y'_i = x_{current_i} + F \times (x^1_i - x^2_i), \text{ Otherwise } y'_i = x_{current_i}$$

$$i = 1 \dots, k$$

- 3: Here only the non-zero values (say k) of $\overline{x_{current}}$ are updated with the help of corresponding k values of x_1 and x_2 keeping remaining values from (k+1) to n unchanged.
- 4: Repair the trial solution to generate $\overline{y''}$

$$\text{if } y'_i < a_i, \text{ then } y''_i = a_i, \text{ elseif } y'_i > b_i, \text{ then } y''_i = b_i, \text{ Otherwise } , y''_i = y'_i$$

Where, $i = 1, 2, \dots, k$ and a_i, b_i are the lower and upper boundaries of variable $x_{current_i}$.

- 5: Mutate the trial solution by using the following Equation to generate \overline{y}
 - (i) if $0 \leq MP < 0.6$: ▷ Perform normal mutation

$$\text{if rand}() \leq p_m, \text{ then } y_i = y''_i + \delta_i \times (b_i - a_i), \text{ otherwise } y_i = y''_i$$

where $i = (1, 2, \dots, k)$, $r = \text{rand}()$ is a random number between 0 and 1, and

$$\delta_i = \begin{cases} \left[2r + (1 - 2r) \left(\frac{b_i - y''_i}{b_i - a_i} \right)^{\eta_m + 1} \right]^{\frac{1}{\eta_m + 1}} - 1, & \text{if } r < 0.5 \\ 1 - \left[2 - 2r + (1 - 2r) \left(\frac{y''_i - a_i}{b_i - a_i} \right)^{\eta_m + 1} \right]^{\frac{1}{\eta_m + 1}} & \text{Otherwise} \end{cases}$$

- (ii) if $0.6 \leq MP < 0.8$: ▷ Perform insert mutation
Pick a random sample from the data set and add that starting from (k+1) th position of the string.
 - (iii) if $0.8 \leq MP \leq 1.0$: ▷ Perform delete mutation
Randomly select a cluster center and delete that from the given string.

6: **return** the new solution \overline{y}

2.5 Extracting Neighborhood Relationship

The neighborhood relationship of current solution is established with the use of SOM which identifies the solutions closer to the current solution. This phenomenon is known as mating pool construction. Mating pool constructed is denoted as Q shown in Fig. 2. These neighboring (closer) solutions can mate to generate a new solution. Steps of constructing mating pool Q for current solution $\overline{x_{current}} \in P$ are shown in Algorithm 1 [14]. In the algorithm we have used the following parameters, M: distance matrix created using position vectors of neurons, P: the population containing solutions $(\overline{x^1}, \overline{x^2}, \dots, \overline{x^n})$, H: mating pool size, β : threshold probability of selecting the closer solution and $\overline{x_{current}}$: current solution for which we want to generate mating pool.

2.6 Crossover and Mutation Operations

Mating pool generated in the previous step is participated in crossover and mutation operation. Two more changes over traditional DE are incorporated to generate the trial solution \overline{y}' and new solution y . Firstly only non-zeros values (say k) of the current solution (excluding appended zeros) are used to generate the trial solution. Second change was made in the mutation operation. A new solution is generated from the current solution $\overline{x_{current}}$ by using any of the three types of mutation operations - normal mutation, insert mutation and delete mutation to explore the search space of solutions efficiently. Any of these three types of mutation operations is selected based on some random probability lying within a range as similar to [10]. The detailed algorithm [14] on genetic operations with changes incorporated is discussed in Algorithm 2. In the algorithm, F and CR are the two control parameters of the DE operator; p_m is the threshold mutation probability for mutation of each component of \overline{y}'' in case of normal mutation; CR is the crossover probability to generate the trial solution; MP is the mutation probability of current solution deciding which type of mutation should be performed and η_m denotes the distribution index of mutation.

2.7 Selection of Best Solution

Any MOO algorithm produces a large number of non-dominated solutions [4] on the final Pareto optimal set and all these solutions are equally important from optimization perspective. But sometimes, user may want to select only a single solution. Therefore, in this paper to select a single solution, we have used an external cluster validity index, *Adjusted Rand Index (ARI)* [7]. ARI utilizes the true partitioning information in its computation. The similarity between the obtained partitioning and the true partitioning is measured in ARI. Definition of ARI suggests that higher value corresponds to most perfect partitioning. Thus we have computed the ARI values for all the partitioning solutions present in the final Pareto front and the solution corresponding to the higher value of ARI is reported here.

3 Experiments and Results

The parameters of the proposed clustering approach, *SMEA_clust*, are as follows: H (mating pool size) = 5, $\eta_0 = 0.1$, $\sigma_0 = 2$, $g_{max} = 50$, $F = 0.8$, $CR = 0.8$, the probability ranges for normal, insertion and deletion mutation operators are 0 to 0.6, 0.6 to 0.8 and 0.8 to 1, respectively. As seen from Fig. 2, in Step-14, we are updating SOM's training data by excluding those solutions which are already covered in SOM training i.e., at the end of each generation, S will contain only new solutions and if no new solutions are left for training then we will exit the loop. Otherwise, the loop will continue till the maximum number of generations, g_{max} , is reached. As MOO gives a set of non-dominated Pareto

optimal solutions, therefore best solution is identified by the method discussed in Sect. 2.7. For the purpose of comparison, some other MOO clustering techniques, MEA_clust (similar to our proposed clustering approach except the use of SOM), MOCK [9] and VAMOSa [10], are also executed on the above mentioned data sets with default parameter settings.

3.1 Data Sets Used for Experiments

The proposed clustering based approach is tested on several data sets. Due to space constraint, here results are provided only for eight datasets, four of them are artificial data sets namely, AD_5.2 [10], AD_10.2 [10], Spherical_6.2 [18], Spherical_4.3 [18] and four are real-life data sets namely, Iris, Newthyroid, Wine and LiverDisorder. These real life data sets are taken from Ref. [19]. Detailed descriptions of these data sets are provided in Table 1. Note that for all the data sets used here, actual class label information is available.

3.2 Discussion of Results

In order to quantify the goodness of the obtained partitionings by different MOO based approaches, an external cluster validity index, namely *Adjusted Rand Index* [7] is utilized. But, for the purpose of comparison of results attained by different clustering approaches, corresponding values of Minkowski Score (MS) [10] for different partitionings and the corresponding number of clusters identified by different MOO based techniques are reported in Table 1. MS is an external cluster validity index to measure the quality of predicted clustering solution with respect to the given true clustering solution. Lower MS value yields better partitioning.

Table 1. Experiment results showing comparison between our proposed approach, SMEA_clust, and two existing MOO based clustering techniques, MOCK and VAMOSa, and another variant of our proposed approach, MEA_clust. Here, AC = actual number of clusters, OC = obtained number of clusters, MS = Minkowski Score, #F = No. of features, #N = No. of data points.

Data set	#F	#N	AC	MOCK		VAMOSa		MEA_clust		SMEA_clust	
				OC	MS	OC	MS	OC	MS	OC	MS
AD_5.2	2	250	5	6	0.39	5	0.25	5	0.35	5	0.25
AD_10.2	2	500	10	6	1.01	10	0.43	10	0	10	0
Spherical_6.2	2	300	6	6	0	6	0	6	0	6	0
Spherical_4.3	3	400	4	4	0	4	0	4	0	4	0
Iris	4	150	3	2	0.82	2	0.80	2	0.72	3	0.67
Newthyroid	5	215	3	2	0.82	5	0.57	3	0.59	3	0.56
Wine	13	178	3	3	0.90	3	0.97	2	0.36	3	0.33
LiveDisorder	6	345	2	3	0.98	2	0.98	2	0.98	2	0.98

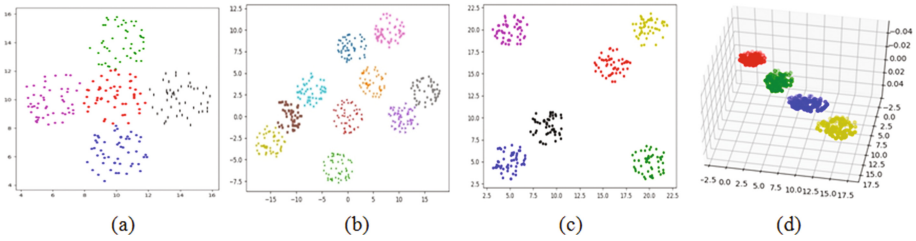


Fig. 3. Automatically clustered (a) *AD_5_2*, (b) *AD_10_2*, (c) *Spherical_6_2*, (d) *Spherical_4_3* by SMEA_clust approach

Results on Artificial Data Sets: As can be seen from Table 1, (i) for AD_5.2 data set, SMEA_clust and VAMOSa both are able to determine the appropriate number of clusters and provide the lowest MS values compared to other techniques, (ii) for AD_10.2, SMEA_clust and MEA_clust perform similarly and are better than other two MOO based techniques, MOCK and VAMOSa, (iii) for Spherical_6.2, because of the well-separated structure of the partitioning, all the clustering approaches are capable of determining the same optimal partitioning, (iv) for Spherical_4.3 again all the clustering approaches are capable of determining the optimal partitioning. Final partitionings obtained by SMEA_clust for AD_5.2, AD_10.2, Spherical_6.2 and Spherical_4.3 data sets are shown in Fig. 3(a), (b), (c) and (d) respectively.

Table 2. p-values returned by t-test comparing the performance of SMEA_clust with other MOO based clustering techniques over different data sets with respect to MS values

Data set	MOCK	VAMOSa	MEA_clust
AD_5.2	1.60815E-237	0.788494	5.34939E-153
AD_10.2	0	0	0.788494
Spherical_6.2	0.788494	0.788494	0.788494
Spherical_4.3	0.788494	0.788494	0.788494
Iris	1.12289E-255	8.12717E-215	2.53903E-51
Newthyroid	6.77E-04	0	2.73576E-21
Wine	0	0	2.73576E-21
LiverDisorder	0.788494	0.788494	0.788494

Results on Real-life Data Sets: As can be seen from Table 1, for LiveDisorder data set, VAMOSa, MEA_clust and SMEA_clust are able to determine the exact number of clusters as well as attain the same MS values. But, for other data sets, Iris, Newthyroid and Wine, our proposed algorithm performs better than other clustering techniques.

Statistical Test: To further establish the superiority of our proposed clustering algorithm, here we have conducted some statistical tests guided by [16]. Statistical t-test which provides some p-value is conducted to check whether the improvements obtained by the proposed SMEA_clust are significant or not. Minimum p-value implies that the proposed clustering algorithm is better than others. The obtained p-values (reported in Table 2) evidently support the results of Table 1.

4 Conclusions and Future Works

In the current study we have developed a new automatic multi-objective clustering approach utilizing the search capability of differential evolution. The current algorithm is a hybridization of DE and SOM where the neighborhood information identified by SOM trained on the current population of solutions is utilized for generating the mating pool which can further take part in genetic operations. The use of SOM at new solution generation helps the proposed clustering algorithm to better explore the search space of optimal partitioning. The potency of the proposed clustering technique is shown in automatically partitioning several artificial and real-life data sets of varying complexities. Results are compared with those obtained by a simple multi-objective DE based clustering technique as well as by several popular MOO based clustering techniques and it was found that the proposed approach converges faster compared to other MOO based clustering techniques.

As a future work, we would like to test the proposed approach for solving some real-life problems like text-summarization, search result clustering etc.

References

1. Jain, A.K., Dubes, R.C.: Algorithms for Clustering Data. Prentice-Hall Inc., Upper Saddle River (1988)
2. Maulik, U., Bandyopadhyay, S.: Performance evaluation of some clustering algorithms and validity indices. *IEEE Trans. Pattern Anal. Mach. Intell.* **24**(12), 1650–1654 (2002)
3. Bandyopadhyay, S., Saha, S., Maulik, U., Deb, K.: A simulated annealing-based multiobjective optimization algorithm: AMOSA. *IEEE Trans. Evol. Comput.* **12**(3), 269–283 (2008)
4. Deb, K.: *Multi-objective Optimization Using Evolutionary Algorithms*. Wiley, New York (2005)
5. Bandyopadhyay, S., Maulik, U.: Nonparametric genetic clustering: comparison of validity indices. *IEEE Trans. Syst. Man Cybern. Part C (Applications and Reviews)* **31**(1), 120–125 (2001)
6. Konak, A., Coit, D.W., Smith, A.E.: Multi-objective optimization using genetic algorithms: a tutorial. *Reliab. Eng. Syst. Saf.* **91**(9), 992–1007 (2006)
7. Suresh, K., Kundu, D., Ghosh, S., Das, S., Abraham, A.: Data clustering using multi-objective differential evolution algorithms. *Fund. Inform.* **97**(4), 381–403 (2009)

8. Kennedy, J.: Encyclopedia of machine learning. In: Saul, L., Fu, M.C. (eds.) Particle Swarm Optimization, pp. 760–766. Springer, New York (2011). doi:[10.1007/978-1-4419-1153-7_200581](https://doi.org/10.1007/978-1-4419-1153-7_200581)
9. Handl, J., Knowles, J.: An evolutionary approach to multiobjective clustering. *IEEE Trans. Evol. Comput.* **11**(1), 56–76 (2007)
10. Saha, S., Bandyopadhyay, S.: A symmetry based multiobjective clustering technique for automatic evolution of clusters. *Pattern Recogn.* **43**(3), 738–751 (2010)
11. Saha, S., Bandyopadhyay, S.: A generalized automatic clustering algorithm in a multiobjective framework. *Appl. Soft Comput.* **13**(1), 89–108 (2013)
12. Kohonen, T.: The self-organizing map. *Neurocomputing* **21**(1), 1–6 (1998)
13. Zhang, H., Zhang, X., Gao, X.Z., Song, S.: Self-organizing multiobjective optimization based on decomposition with neighborhood ensemble. *Neurocomputing* **173**, 1868–1884 (2016)
14. Zhang, H., Zhou, A., Song, S., Zhang, Q., Gao, X.Z., Zhang, J.: A self-organizing multiobjective evolutionary algorithm. *IEEE Trans. Evol. Comput.* **20**(5), 792–806 (2016)
15. Pakhira, M.K., Bandyopadhyay, S., Maulik, U.: Validity index for crisp and fuzzy clusters. *Pattern Recogn.* **37**(3), 487–501 (2004)
16. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7**(Jan), 1–30 (2006)
17. Haykin, S.S.: *Neural Networks and Learning Machines*, vol. 3. Pearson, Upper Saddle River, NJ, USA (2009)
18. Bandyopadhyay, S., Maulik, U.: Genetic clustering for automatic evolution of clusters and application to image classification. *Pattern Recogn.* **35**(6), 1197–1208 (2002)
19. Lichman, M.: UCI machine learning repository (2013)