Data Clustering Based on Particle Swarm Optimization with Neighborhood Search and Cauchy Mutation

Dang Cong Tran[1,2] and Zhijian Wu[1]

[1] State Key Laboratory of Software Engineering, Computer School, Wuhan University, Wuhan 430072, China
[2] Vietnam Academy of Science and Technology, Hanoi, Vietnam
trandangcong@gmail.com, zhijianwu@whu.edu.cn

Abstract. K-means is one of the most popular clustering algorithm, it has been successfully applied in solving many practical clustering problems, however there exist some drawbacks such as local optimal convergence and sensitivity to initial points. In this paper, a new approach based on enhanced particle swarm optimization (PSO) is presented (denoted CMPNS), in which PSO is enhanced by new neighborhood search strategy and Cauchy mutation operation. Experimental results on fourteen used artificial and real-world datasets show that the proposed method outperforms than that of some other data clustering algorithms in terms of accuracy and convergence speed.

Keywords: data clustering, K-means, particle swarm optimization.

1 Introduction

Data clustering is the process of identifying natural groupings or clusters, within multidimensional data, based on some similarity measure. The K-means clustering algorithm was developed by J.A. Hartigan [1] which is one of the most popular and widely used clustering techniques because it is easy to implement and very efficient, with linear time complexity. However, its main drawbacks are that it converges to arbitrary local optima as well as at local maxima and saddle points and that it cannot deal well with non-spherical shaped clusters [2]. The performance of the K-means algorithm depends on the initial choice of the cluster centers. In order to tackle the drawback of initialization, in [3] a method called K-means++ was presented, where a new initial method was presented. An alternative approach is applying evolutionary algorithms (EAs) in clustering, yielding EA-based clustering algorithms. Unlike K-means clustering, they simultaneously optimize a population of candidate solutions, which give them the ability to escape from local optima. Various EA-based clustering algorithms have been developed, including genetic algorithms, differential evolution, ant colony optimization, artificial bee colony, and particle swarm optimization [4–6].

C.K. Loo et al. (Eds.): ICONIP 2014, Part II, LNCS 8835, pp. 151–159, 2014.

Remain of this paper is structured as follows: some preliminaries of K-means and PSO algorithms are briefly reviewed in Section 2. The proposed CMPNS algorithm will be described in Section 3. The benchmark datasets, parameters setting and results will be demonstrated in Section 4. Finally, in Section 5 the conclusions will be drawn.

2 Preliminaries

2.1 K-means Clustering Algorithm

In partitioning clustering problems, we need to divide a set of N objects into K clusters. Let $O(o_1, o_2, ..., o_N)$ be the set of N objects of data set. Each object has D features, and each feature is quantified with a real-value. Let $S_{N\times D}$ be the feature data matrix. It has N rows and D columns. Each row S_i presents a data vector and s_{ij} corresponds to the jth feature of ith data vector (i=1,2,...,N, j=,1,2,...,D). Let $C = (C_1, C_2, ..., C_K)$ be the K clusters. Then $C_i \neq \phi,\ C_j \cap C_i \neq \phi,\ \cup_{j=1}^{K} C_i = \mathrm{O},\ i, j = 1, 2, ..., K,\ i \neq j$. The goal of clustering algorithm is to find such a C that makes the objects in the same clusters are as similar as possible while other objects in the different clusters as dissimilar, which can be measured by some criterions.

K-means clustering [1] groups data vectors into a pre-specified number of clusters, based on Euclidean distance as similarity measure. The classical K-means algorithm is summarized as follows:

Step 1. Randomly choose K cluster centroids from N objects.

Step 2. For each data vector, assign the vector to the cluster with the closest centroid, where the distance to the centroid is determined by Eq. (1).

$$d(S_i, Z_j) = \sqrt{\sum_{p=1}^{D} (S_{ip} - Z_{jp})^2} \tag{1}$$

Step 3. Recalculate the cluster centroids, using Eq. (2) as follows:

$$Z_j = \frac{1}{N_{C_j}} \sum_{\forall S_p \in C_j} S_p \tag{2}$$

where N_{C_j} is the number of data vectors in cluster j and C_j is the subset of data vectors that form cluster j, return *Step 2* if stopping criterion is not satisfied.

2.2 Particle Swarm Optimization

Each particle in PSO [7, 8] has a velocity vector (V) and a position vector (X). PSO remembers both the best position found by all particles and the best positions found by each particle in the search process. For a search problem in D-dimensional space, a particle represents a potential solution. The velocity and position of particle are updated according to Eqs. (3) and (4).

$$v_{ij} = w \cdot v_{ij} + c_1 \cdot rand1_{ij} \cdot (pbest_{ij} - x_{ij}) + c_2 \cdot rand2_{ij} \cdot (gbest_j - x_{ij}) \tag{3}$$

$$x_{ij} = x_{ij} + v_{ij} \tag{4}$$

where the particles index i =1, 2, ..., NP, NP is the population size, x_i is the position of the ith particle, v_i represents the velocity of ith particle, $pbest_i$ is the best previous position yielding the best fitness value for the ith particle, and $gbest$ is the global best particle found by all particles so far, $rand1_{ij}$ and $rand2_{ij}$ are two random numbers independently generated within the range of [0, 1], c_1 and c_2 are two learning factors which control the influence of the social and cognitive components, w is the inertia factor. The inertia weight w in Eq. (3) was introduced by Y. Shi et al. [9], a w linearly decreasing with the iterative generations was proposed as Eq. (5).

$$w_k = w_0 - \frac{(w_0 - w_1) \cdot k}{Max\_Gen} \tag{5}$$

where k is the kth generation index, w_0 and w_1 are maximum and minimum inertia weight value, respectively.

3 Proposed Method

To improve the performance of K-means over the drawbacks and enhance the algorithm in terms of convergence speed and accuracy, in this paper we present a new approach based on improved PSO, where PSO is introduced into K-means.

3.1 Neighborhood Search

By employment of local neighborhood search and global neighborhood search strategies with ring topology and radius is equal to 2, H. Wang et al. [10] proposed DNSPSO approach to enhance PSO algorithm, in which a local neighborhood search (LNS) and global search (GNS) strategies were proposed. To improve the exploitation ability of the local search strategy, the best particle of local neighbour is employed to generate the trial particle LNS. The neighborhood of a particle P_i, a trial particle $L_i = (LX_i, LV_i)$ is generated by Eqs. (6, 7).

$$LX_i = r_1 \cdot X_i + r_2 \cdot (pbest_i - X_i) + r_3 \cdot nbest_i \tag{6}$$

$$LV_i = V_i \tag{7}$$

where X_c and X_d are the position vectors of two random particles in the k-neighborhood radius of P_i, $c, d \in [i-k, i+k] \wedge c \neq d \neq i$, r_1, r_2 and r_3 are three uniform random numbers within (0,1), and $r_1 + r_2 + r_3 = 1$, and $nbest_i$ is the best particle of X_i neighborhood.

Besides the LNS, a global neighborhood search (GNS) strategy is proposed to enhance the ability of exploration. When searching the neighborhood of a particle P_i, another trial particle $G_i = (GX_i, GV_i)$ is generated by Eqs. (8, 9).

$$GX_i = r_4 \cdot X_i + r_5 \cdot gbest + r_6 \cdot (X_e - X_f) \tag{8}$$

$$GV_i = V_i \tag{9}$$

where X_e and X_f are the position vectors of two random particles chosen for the entire swarm, $e, f \in [1, NP] \wedge e \neq f \neq i$, r_4, r_5 and r_6 are three uniform random numbers within (0, 1), and $r_4 + r_5 + r_6 = 1$.

3.2 Diversity Mechnism

Like DNSPSO [10], the diversity mechanism was employed, where for each particle $P_i(t)$ a new particle $P_i(t+1)$ is generated by the PSO's velocity and position updating equations. By recombining $P_i(t)$ and $P_i(t+1)$, a trial particle $TP_i(t+1) = (TX_i(t+1), TV_i(t+1))$ is generated as follows:

$$TX_{ij}(t+1) = \begin{cases} X_{ij}(t+1) & \text{if } rand_j(0,1) < P_r \\ X_{ij}(t) & \text{otherwise} \end{cases} \tag{10}$$

$$TV_{ij}(t+1) = V_{ij}(t+1) \tag{11}$$

where P_r is a user-defined value of greedy selection probability. After recombination, a greedy selection is used as follows:

$$P_i(t+1) = \begin{cases} TP_i(t+1) & \text{if } f\left(TP_i(t+1)\right) < f\left(P_i(t+1)\right) \\ P_i(t+1) & \text{otherwise} \end{cases} \tag{12}$$

3.3 Cauchy Mutation

Aim to improve the convergence speed, in each iteration the global best particle is mutated by Cauchy distribution function [11] as follows:

$$gbest_j = gbest_j + Cauchy() \tag{13}$$

3.4 Reinitialization

Similar to the scout bee of artificial bee colony (ABC) [12], particle is reinitialized randomly if the number of relative *pbest* fitness not changed is more than the pre-defined number (called *limit*, in this case the particle may be trapped into local optima). By this technique, the exploration ability of the algorithm can be enhanced.

3.5 Proposed Algorithm

Firstly, particle is encoded according to Eqs. (14), (15). Each particle is a potential candidate solution for the optimal center centroids. In this case, solving data clustering problem can be seen as solving the global optimization with fitness function is the validity index of SED calculated by Eq. (16).

$$X_i = (X_{i1}, ..., X_{ij}, ..., X_{iK}) \tag{14}$$

$$V_i = (v_{i,1}, v_{i,2}, ..., v_{i,K \times D}) \tag{15}$$

Table 1. The main steps of CMPNS

```
Initialize each particle by randomly selecting from dataset;
While FEs ≤ MaxFEs do
    Update inertia weight w according to Eq. (5);
    For i = 1 to NP do
        Update the velocity and position according to Eq. (3, 4);
        Generate a new trial particle TP_i by Eqs. (10, 11);
        Select a fitter one between P_i and TP_i as the new P_i by Eq. (12);
        Update pbest and gbest;
        If f(pbest) = f(lastpbest) then monitor[i] + + else monitor[i] = 0;
    End for
    For i = 1 to NP do
        If rand(0, 1) ≤ P_ns then
            Generate a trial particle L_i according to Eqs. (6, 7);
            Generate a trial particle G_i according to Eqs. (8, 9);
            Select the best one among P_i, L_i, and G_i as the new P_i;
            Update pbest and gbest;
            If f(pbest) = f(lastpbest) then monitor[i] + + else monitor[i] = 0;
        End if
    End for
    Mutate gbest according to Eq. (13);
    If monitor[k] ≥ max(monitor[j], j = 1, .., NP) and monitor[k] ≥ limit then
        Reinitialize the kth particle from random K distinct data objects of dataset;
End while
```

where D-dimensional vector $X_{ij} = (x_{i,1j}, x_{i,2j}, ..., X_{i,Dj})$ represents the $j$cluster centroid of ith particle.

L. Kaufman et al. [13] suggested that Sum of Euclid Distance (SED) is better than Mean Squared Error (MSE) for measuring cluster analysis results. In this paper we also use SED, which is calculated by Eq. (16), is used as the fitness function.

$$SED = \sum_{j=1}^{K} \sum_{S_i \in C_j} \|S_i - X_j\| \tag{16}$$

The main steps of the proposed algorithm are listed in Table 1, where *NP* is the population size, K is the number of clusters, *lastpbest* records the last fitness values of *pbest*. $monitor[i]$ records the successive number of iterations where the fitness values of $pbest_i$ does not change. *FEs* is the number of fitness evaluations, and *MaxFEs* is the maximum number of fitness evaluations. P_{ns} is the probability to implement the neighborhood search strategy, *limit* is the pre-defined number. The fitness function is SED function calculated by Eq. (16).

3.6 Measure Criterions

Two metrics were used in our experiments, the first measure is the fitness value, the sum of Euclid distance SED, as defined in Eq. (16). The second metric is the clustering accuracy, which is the percentage of the objects that are correctly

recovered in a clustering result (called classification accuracy percentage CAP) defined in Eq. (17).

$$CAP = 100 \times \frac{\#\text{of correctly classified examples}}{\text{size of test data set}} \tag{17}$$

4 Experimental Results

To evaluate the performance of the proposed algorithm, fourteen benchmark datasets including four artificial datasets and ten real-world datasets were used. In addition, four data clustering algorithms K-means[1], K-means++[3], KPSO[4], and PSOK[5] were compared to the proposed algorithm in terms of fitness value SED and accuracy CAP.

Table 2. The main properties of artificial datasets

Data set	Size	Features	No of clusters	Data set	Size	Features	No of clusters
Dataset1	400	3	4	Dataset3	300	2	6
Dataset2	250	2	5	Dataset4	500	2	10

4.1 Benchmark Datasets

The details of properties of artificial datasets are described in Table 2 [14], the properties of ten real-world datasets Iris, Wine, Glass, Ecoli, Liver disorder, Vowel, Vowel 2, Pima, WDBC, and CMC can be found in [15].

4.2 Parametric Settings

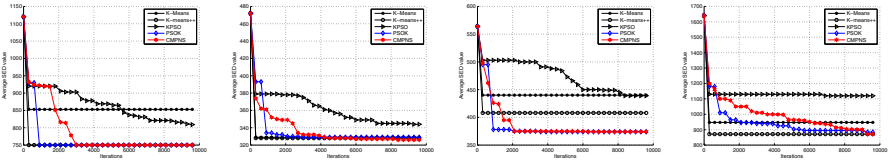
In this test, the parameters of four other competitive algorithms K-means, K-means++, KPSO, PSOK are set according to their experiments. For the sake of fair comparison, the population size *NP=100*. The maximal number of fitness evaluations *MaxFEs* was set to 10e+04 for all algorithms, all algorithms were run on each of the 14 datasets over 25 times and their mean value of SED, accuracy percentage. For CMPNS, other parameters were empirically set as follows: $w_0 = 0.9$, $w_1 = 0.4$, $c_1 = c_2 = 1.49$, $P_r = 0.9$, $P_{ns} = 0.6$, $limit = 50$.

4.3 Comparison of Results

The results of SED are shown in Tables 3, where the best values are written in bold. The results in Table 3 indicate that the proposed CMPNS algorithm has the best results of SED on 12 of 14 datasets, two other datasets of Vowel2 and Ecoli belong to K-means and K-means++, respectively. In order to compare the performance of multiple algorithms on the test suite, we conduct Friedman test [16], the highest ranking belongs to CMPNS, namely the ranks of K-means,

Table 3. Comparison of SED results

Data set	K-means	K-means++	KPSO	PSOK	CMPNS
Dataset1	8.5182e+02	7.4998e+02	8.1271e+02	7.4997e+02	**7.4961e+02**
Dataset2	3.2838e+02	3.2816e+02	4.1106e+02	3.2841e+02	**3.2644e+02**
Dataset3	4.4943e+02	4.2890e+02	4.4795e+02	3.7449e+02	**3.7361e+02**
Dataset4	9.4805e+02	8.7136e+02	1.1241e+03	8.8793e+02	**8.6534e+02**
Iris	1.0502e+02	9.8663e+01	1.0685e+02	9.7272e+01	**9.6691e+01**
Wine	1.6838e+04	1.7339e+04	1.7078e+04	1.6364e+04	**1.6299e+04**
Glass	2.2470e+02	2.3202e+02	2.4546e+02	2.1866e+02	**2.1773e+02**
Ecoli	6.4785e+01	**6.3604e+01**	6.7063e+01	6.4673e+01	6.4697e+01
Liver dis	1.0213e+04	1.0222e+04	1.0262e+04	9.8829e+03	**9.8519e+03**
Vowel	1.5306e+05	1.5304e+05	1.7413e+05	1.5119e+05	**1.5069e+05**
Vowel2	**7.0912e+02**	7.0980e+02	8.6285e+02	7.2348e+02	7.1489e+02
Pima	5.2072e+04	5.2072e+04	5.0867e+04	4.7832e+04	**4.7564e+04**
WDBC	1.5295e+05	1.5295e+05	1.5215e+05	1.4985e+05	**1.4953e+05**
CMC	5.5133e+03	5.5142e+03	6.1808e+03	5.5140e+03	**5.5103e+03**



(a) Dataset1 (b) Dataset2 (c) Dataset3 (d) Dataset4

Fig. 1. The convergence curves on artificial datasets

Table 4. Comparison of results of CAP on artificial datasets (in percentage)

Data set	K-means	K-means++	KPSO	PSOK	CMPNS
Dataset1	96.25±9.16	**100.00±0**	**100.00±0**	**100.00±0**	**100.00±0**
Dataset2	94.00±0	94.68±1.09	85.44±5.07	95.80±1.62	**96.46±1.50**
Dataset3	89.17±9.79	92.50±8.51	97.95±4.04	**100.00±0**	**100.00±**
Dataset4	89.13±4.81	91.01±5.38	92.31±4.84	93.76±5.23	**94.85±4.28**
Iris	82.27±10.48	87.83±4.99	88.30±2.98	89.33±0.43	**89.97±0.15**
Wine	69.97±0.62	69.24±1.18	71.29±1.03	70.84±0.25	**71.52±0.32**
Glass	56.80±2.51	55.33±3.26	59.30±0.86	59.37±3.40	**60.09±2.81**
Ecoli	80.51±2.76	**81.49±1.99**	78.81±3.01	81.12±2.20	80.79±3.03
Liver	**57.97±0**	**57.97±0**	**57.97±0**	**57.97±0**	**57.97±0**
Vowel	58.29±2.78	58.86±2.21	57.65±2.68	58.27±1.66	**59.39±2.70**
Vowel2	37.43±2.36	36.66±1.83	34.99±2.72	36.94±1.96	**37.44±1.93**
Pima	65.10±0	65.10±0	65.10±0	**66.02±0**	**66.02±0**
WDBC	85.41±0	85.41±0	86.29±0.79	86.41±0.38	**86.820**
CMC	45.34±0.40	45.11±0.38	44.95±0.91	45.26±0.38	**45.58±0.13**

K-means++, KPSO, PSOK, and CMPNS are 3.29, 3.07, 3.18, 2.36, and 1.29, respectively. The representative convergence curves of artificial datasets are illustrated in the Fig. 1.

The CAP results of CAP average of 25 times on each of all datasets are listed in Tables 4, where the best results be written in bold. The results in Tables show that the proposed CMPNS algorithm has the best accuracy percentage in majority of benchmark datasets, only on Ecoli dataset the best result belongs to K-means++ algorithm.

5 Conclusions

In this study, we propose a new data clustering approach in order to improve K-means algorithm by enhanced PSO algorithm. Aiming to overcome the shortcoming of K-means, enhanced PSO approach by employing the proposed neighborhood search strategy and combining with diversity mechanism, Cauchy mutation operation, and reinitialization was introduced into K-means. The results obtained from testing on fourteen benchmark datasets including artificial and real-world datasets the proposed CMPNS algorithm is also good at data clustering in compared with some data clustering algorithms. So that, CMPNS can be an alternative for solving data clustering problems and other relevant problems.

Acknowledgments. This work was supported by the National Natural Science Foundation of China (No.: 61070008 and 61364025).

References

1. Hartigan, J.A.: Clustering algorithms, 1st edn. Wiley, New York (1975)
2. Selim, S.Z., Ismail, M.A.: K-means type algorithms: a generalized convergence theorem and characterization of local optimality. IEEE Trans. Patfern Anal. Mach. Intell. 6, 81–87 (1984)
3. Arthur, Vassilvitskii: K-means++: The advantages of careful seeding. In: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2007), pp. 1027–1035 (2007)
4. Merwe, D., Engelbrecht, A.: Data clustering using particle swarm optimization. In: Proceedings of the Congress on Evolutionary Computation 2003 (CEC 2003), pp. 215–220 (2003)
5. Neshat, M., Yazdi, S.F., et al.: A New Cooperative Algorithm Based on PSO and K-Means for Data Clustering. Journal of Computer Science 8(2), 188–194 (2012)
6. Kao, Y., Lee, S.: Combining K-means and Particle Swarm Optimization for Dynamic Data Clustering Problems. In: Proceedings of IEEE International Conference on Intelligent Computing and Intelligent Systems 2009, pp. 757–761 (2009)
7. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proceedings of IEEE International Conference on Neuron Networks Conference Proceedings, Perth, Australia, pp. 1942–1948 (1995)
8. Clerc, M., Kennedy, J.: The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space. IEEE Trans on Evol. 6(1), 58–73 (2002)
9. Shi, Y., Eberhart, R.: A Modified Particle Swarm Optimizer. In: Proceedings of the 1998 Congress on Evolutionary Computation (CEC 1998), pp. 69–73 (1998)
10. Wang, H., Sun, S., Li, C., Rahnamayan, S., Pan, J.: Diversity enhanced particle swarm optimization with neighborhood search. Information Sciences 223, 119–135 (2013)
11. Wang, H., Wu, Z., Rahnamayan, S., Liu, Y., Ventresca, M.: Enhancing particle swarm optimization using generalized opposition-based learning. Information Sciences 181, 4699–4714 (2011)

12. Karaboga, D.: An idea based on honey bee swarm for numerical optimization. Technical report-TR06, Erciyes University, Engineering Faculty (2005)
13. Kaufman, L., Rousseeuw, P.: Finding Groups in Data: An Introduction to Cluster Analysis. John Wiley and Sons, New York (1990)
14. Bandyopadhyay, S.: Artificial data sets for data mining, http://www.isical.ac.in/~sanghami/data.html
15. UCI Repository of Machine Learning Databases: retrieved from the World Wide Web, http://www.ics.uci.edu/~mlearn/MLRepository.html
16. Derrac, J.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation 1, 3–18 (2011)