Clustering Algorithm Combining CPSO with K-Means

Chunqin Gu [1, a], Qian Tao [2, b]

[1]Department of Information Science, Zhongkai University of Agriculture and Engineering, Guangzhou, 510225, China

[2] Department of Computer Science, Guangdong University of Education, Guangzhou, China

[a]email: guchunqin@gmail.com, [b]email:tao66@139.com(Corresponding author)

Keywords: K-Means; Clustering; Particle Swarm Optimization; Chaotic

Abstract. A clustering algorithm combining particle swarm optimization (CPSO) with K-Means (KM-CPSO) is proposed, which features better search efficiency than K-Means, PSO and CPSO. The K-Means algorithms cannot guarantee convergence to global optima and suffer in local optimal cluster center because they are sensitive to initial cluster centers. Chaotic particle swarm optimization (CPSO) can find global optimal solution; meanwhile K-Means can achieve local optima. The CPSO-KM algorithm utilizes both global search capability of CPSO and local search capability of K-Means. CPSO-KM algorithm has been tested with two synthetic datasets and three classical data sets from UCI. Experimental results show better performance of the CPSO-KM as compared to K-Means, PSO and CPSO.

Introduction

Clustering is a common unsupervised learning method, which partitions a group of objects (instances) into groups (clusters) such that objects in the same cluster are similar to each other and dissimilar to the objects in other clusters. K-Means [1] algorithm partitions the groups of given objects into k clusters based on a distance metric. The K-Means algorithm is easy to implement and very efficient. The main drawback of the K-Means algorithm is that the clustering result is sensitive to the initial clusters centers and may converge to the local optima [2].

In recent years, swarm intelligent algorithms has been combined with K-Means and applied on many clustering problems, because of the ability of global search of the swarm intelligent algorithms, such as GA[3], PSO[4], ACO[5]. The combination of swarm intelligent algorithms and K-Means algorithm can take advantage of both global search ability of swarm intelligent algorithms and local search ability of K-Means algorithm. CPSO[6] has the advantage of more precise global search ability and more fast convergence speed in our previous literature. The CPSO is used to obtain better clusters centers for initial clusters centers, then K-Means algorithm is used based on the initial clusters centers found by CPSO.

The rest of the paper is organized as follows. Section 2 describes the model of clustering problems. Section 3 presents our clustering algorithm combining CPSO with K-Means (CPSO-KM). Section 4 illustrates experimental results. Finally, Section 5 makes conclusion.

Model of clustering problem

The clustering problem is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. Given a data objects set $DS=\{\mathrm{X}_1,\mathrm{X}_2,...,\mathrm{X}_N\}$, where $X_j=(x_j^1,x_j^2,...,x_j^L)$, L is the dimension of a data object, a clustering problem tries to find a K-partition of DS, $C=\{C_1,C_2,...,C_K\}$, such that the similarity of the data objects in the same cluster is maximum and the difference of the data objects between different cluster centroid is maximum. The objective function of clustering problem is evaluated based on the sum of squared error (SSE), which is defined as

$$\min \sum_{i=1}^{K} \sum_{\forall x_j \in C_i} D^2(\mathrm{x}_j, \overline{C_i}), \overline{C_i} = \frac{1}{|C_i|} \sum_{\forall x_j \in C_i} x_j \tag{1}$$

$$\bigcup_{i=1}^{K} C_i = DS, \quad C_i \neq \phi \tag{2}$$

$$C_i \cap C_j = \phi, \qquad i \neq j \quad and \quad i, j \in \{1, 2, ..., K\} \tag{3}$$

where D(x, y) denotes the Euclidean distance between x and y, $\overline{C_i}$ is the mean of data objects in the cluster C_i.

Clustering Algorithm Combining CPSO with K-Means

In this paper, we develop a hybrid algorithm for solving clustering problems, whose flowchart is shown in Fig.1. It can be observed that CPSO is combined into the clustering algorithm. Detailed implementations of the algorithm are proposed and described in this section.

A. Particle Swarm Optimization

Particle swarm optimization (PSO), which was developed by Kennedy and Eberhart in 1995[7,8], is a population-based swarm intelligence algorithm. The PSO simulates the social behavior of birds flocking and fish schooling. It has been successfully applied to many practical optimization problems. In PSO, each particle i represents a candidate solution in the solution space of D dimensions, which has two vectors: a position vector $X_i = [x_i^1, x_i^2, ..., x_i^D]$ and a velocity vector $V_i = [v_i^1, v_i^2, ..., v_i^D]$.

During the evolutionary process, the velocity vector and the position vector of particle i on dimension d at iteration t+1 are updated as

$$\begin{aligned} v_i^d(t+1) = \omega v_i^d(t) + c_1 \cdot r_1 \cdot (pBest_i^d(t) - x_i^d(t)) \\ + c_2 \cdot r_2 \cdot (gBest^d(t) - x_i^d(t)) \end{aligned} \tag{4}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{5}$$

where d=1,2,…,D represents each dimension of the search space, ω is inertia weight, c_1 and c_2 are cognitive learning coefficient and social learning coefficient, respectively, r_1 and r_2 are two uniform random numbers in the range of [0, 1], $pBest_i^d(t)$ is the position on dimension d with the best fitness found up to the tth iteration for the particle i, $gBest_i^d(t)$ is the best position on dimension d found by the whole particle swarm.

The inertia weight ω in (4) is usually updated as

$$\omega = \omega_{max} - (\omega_{max} - \omega_{min}) \times \mathrm{g}/\mathrm{g}_{max},$$

where ω_{max} and ω_{min} are initial and final weight, and set to 0.9 and 0.4, respectively[9]. g is the current evolutionary generation number, g_{max} is the maximum number of generations. c_1 and c_2 are set 1.8, respectively. During the evolutionary process, the velocity of each particle on dimension d is restricted to the range of $\left[-vmax^d, vmax^d\right], vmax^d \in \Re^+$. The maximum velocity $vmax^d$ is usually set to 20% of the search range [10].

To find good initial cluster center, the cluster centers should be encoded into particle's representation. We employ a representation that each particle is characterized by a D=Num_Attri*$Num\_Centroids$ dimensional real number vector. Num_Attri is the number of attribution of instances in the clustering problems. $Num\_Centroids$ is the number of the centroids of instances in the clustering problems. Thus the particle is encoded as a sequence of cluster centroids. The ith particle in the swarm is in the form of

$$X_i = [x_i^{1,1}, x_i^{1,2}, ..., x_i^{1,Num\_Attri}, x_i^{2,1}, x_i^{2,2}, ..., x_i^{2,Num\_Attri}, ..., x_i^{Num\_Centroids,1}, x_i^{Num\_Centroids,2}, ..., x_i^{Num\_Centroids,Num\_Attri}] \tag{6}$$

where $x_i^{j,k}$ is the kth attribution value of the jth centroid, $j = 1, 2, ..., Num\_Centroids$, $k = 1, 2, ..., Num\_Attri$, Fig.1 shows the particle representation of the ith particle in a 2-D space.

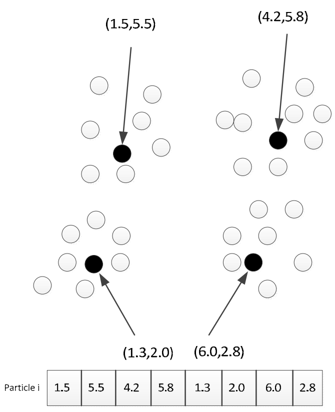

Fig.1. Particle representation in a 2-D space

In the initialization, the position of all particles are randomly generated, with $(x_i^{j,1}, x_i^{j,2}, ..., x_i^{j,Num\_Attri})$ is the attribution value of the *j*th centroid($j = 1, 2, \ldots, K$). The velocity of each particle is also randomly generated, while the maximum velocity of each dimension is set to be 20% of the search range.

B. K-Means

The K-means algorithm is a straightforward and widely used clustering algorithm. It is simple to implement and run, relatively fast. The K-Means algorithm partitions a given dataset into a user-specified number of clusters.

The standard K-Means algorithm is described as follows:

(1) Initialize the K cluster centroid $M = \{m_1, m_2, ..., m_K\}$ randomly.

(2) Assign each data point x_j to the closest cluster based on Euclidean distance. The distance between data point x_j to the closest cluster centroid m_k is defined as

$$D(x_j, m_k) = \sqrt{\|x_j - m_k\|^2} \tag{7}$$

(3) Recalculate the means of *K* cluster centers, and obtain new *K* centers.

(4) Repeat step 2 and 3 until the average change in centroid vectors is less than a predefined value.

C. Chaotic PSO algorithm(CPSO) and fitness calculation

Otherwise, during the evolutionary process, the PSO may trap into a local optima solution, then the exploration performance will not be improved. CPSO introduces chaotic mapping with certainty and stochastic property into PSO in order to improve the global convergence ability [11]. The chaotic mapping method is the chaotic logistic sequence, which is defined as follows:

$$x(t+1) = r \cdot x(t) \cdot (1 - x(t)), r \in N, x(0) \in [0,1] \tag{8}$$

where r is the control parameter, y is a variable, r = 4, $x \notin \{0, 0.25, 0.5, 0.75, 1\}$ and t = 0, 1, 2, ….

Compute the radius of chaotic searching region to avoid destroy the excellent solution. The radius can be adaptively adjusted by the distance between *pBest* and *gBest* and variance.

The chaotic searching time is defined as

$$\sigma(pbest, gbest) = \frac{1}{PNum} \sum_{i=1}^{PNum} \sqrt{\sum_{j=1}^{d} (pbest(particle_{i,j}) - gbest_j)^2} \le \varepsilon_1, \tag{9}$$

where ε_1 is the threshold of the premature convergence. As described in Section II, the objective of clustering problem is the minimizing of the SSE.

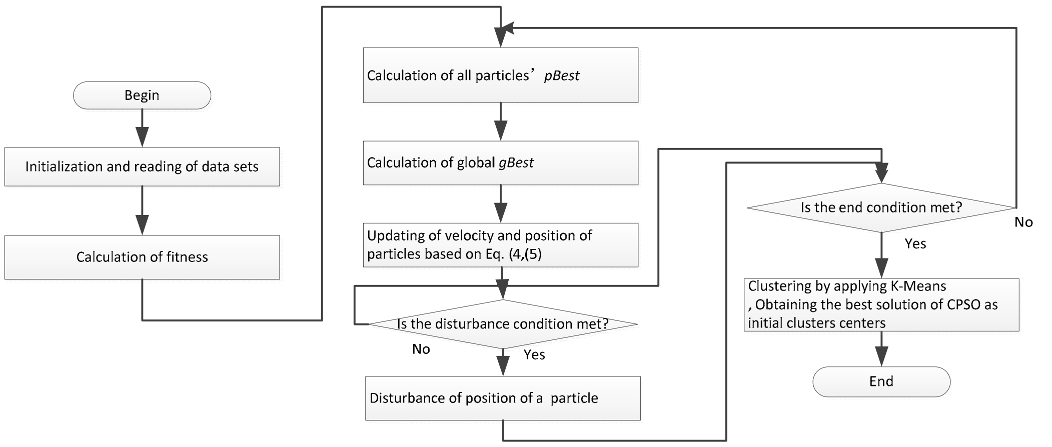
Begin
Initialization and reading of data sets
Calculation of fitness
Calculation of all particles' pBest
Calculation of global gBest
Updating of velocity and position of particles based on Eq. (4,(5)
Is the disturbance condition met?
No
Yes
Disturbance of position of a particle
Is the end condition met?
No
Yes
Clustering by applying K-Means , Obtaining the best solution of CPSO as initial clusters centers
End

Fig. 2. Flowchart of CPSO-KM

Experiments and comparisons

In this section, seven data sets are employed to validate the proposed CPSO-KM clustering algorithm and to compare the CPSO-KM with conventional K-Means, PSO and CPSO.

A. Data sets and algorithm configuration

The five data sets include three synthetic data sets (SynSet1, SynSet2) and three real data sets. The three synthetic data sets are generated based on normal distribution with mean vector μ and covariance matrix $\sum$. The parameters for the three synthetic data sets are shown in Table Ⅰ. The synthetic data sets are relatively simple, and the real number and centers position of their clusters are known in advance, so experiments on these data sets can reveal the virtues and defects of the algorithms. Fig. 3-4 graphically presents the two synthetic data sets.

Table Ⅰ. The parameters for two synthetic data sets

Data Sets	No. of clusters	No. of dimensions	Size of clusters	Parameters of clusters
SynSet1	3	2	50	$\mu_1=[10,10],\Sigma_1=0.4$ $\mu_2=[7,7],\Sigma_2=0.4$ $\mu_3=[3,3],\Sigma_3=0.4$
SynSet2	4	3	100	$\mu_1=[30,30,30],\Sigma_1=4$ $\mu_2=[20,20,20],\Sigma_2=4$ $\mu_3=[14,14,14],\Sigma_3=4$ $\mu_4=[5,5,5],\Sigma_4=4$

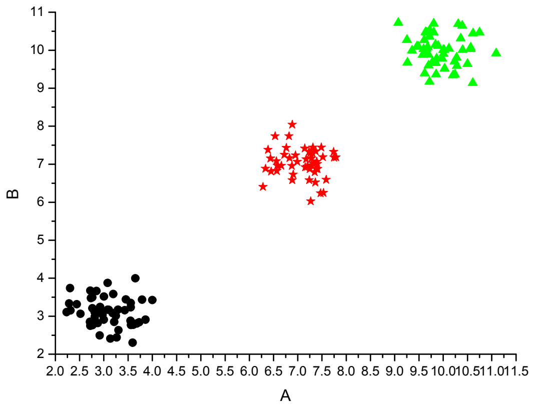

Fig. 3. Clusters of the SynSet1

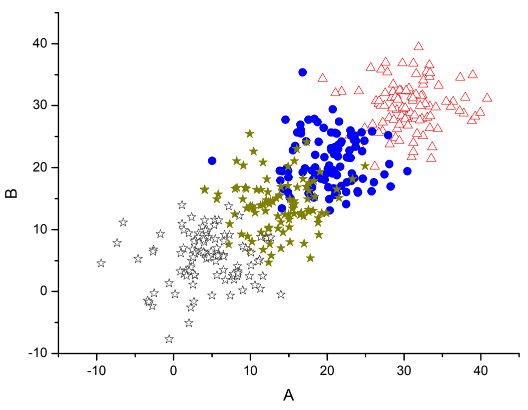

Fig. 4. Clusters of the SynSet2

The three real data sets that we considered are Iris, Haberman, Hayes-roth, which are available online [12]. These data sets are very classical and often used to examine and compare the performances of algorithms in the fields of classification.

The PSO, CPSO and CPSO-KM use the same population size of 20 and the same number of 1000 fitness evaluations (FEs) for a fair comparison. All the experiments are carried out using Visual C++ on the same machine with a Pentium(R) Dual-Core CPU 3.20 GHz, 2.0GB RAM, and Windows 7 operation system. All the data sets are independently simulated 30 times, and their mean results are compared in order to reduce the statistical errors.

B. Performance and F-Measure comparison between CPSO-KM and other algorithms

To validate the proposed CPSO-KM, we compared the CPSO-KM with K-Means, PSO and CPSO. Each data set was independently simulated 30 times for purpose of reducing statistical errors. The statistical results are shown in Table Ⅱ in terms of the mean and standard deviation of the fitness, and boldface in the table indicates the best result obtained.

Because the real partitions of the data sets considered here are already known, the performances of CPSO-KM algorithm can be evaluated by comparing the resulting cluster with the real structures in terms of external criteria. A commonly used criteria is F-Measure.

Assuming that S is a prespecified partition of data sets DS, C is the resulting clusters by applying the CPSO-KM algorithm.

(1) TP is the numbers of pairs of data points (x_i, x_j), where $x_i, x_j \in C_m, x_i, x_j \in S_n, i \neq j$.

(2) FP is the numbers of pairs of data points (x_i, x_j), where $x_i, x_j \in C_m, x_i, \in S_{n1}, x_j \in S_{n2}, i \neq j, n1 \neq n2$.

(3) TN is the numbers of pairs of data points (x_i, x_j), where $x_i \in C_{m1}, x_j \in C_{m2}, x_i \in S_{n1}, x_j \in S_{n2}, i \neq j, m1 \neq m2, n1 \neq n2$.

(4) *P* and *R* are the precision and recall of cluster *i*.

$$F-Measure = \frac{(\beta^2 + 1.0)(P \times R)}{(\beta^2 \times P) + R}, \text{ where } P = \frac{TP}{TP+FP}, \ R = \frac{TP}{TP+FN}$$

In experiments, we set $\beta = 1.0$, which weights precision and recall equally. The performance comparison between CPSO-KM and other algorithms in terms of F-Measure is illustrated in Table Ⅱ.

An interesting result is that all algorithms have most reliably achieved the minimal fitness and 1 F-Measure of SysSet1. It is shown that CPSO-KM attains better fitness than K-Means, PSO and CPSO. The CPSO-KM is also observed to obtain the smallest *SD* of the fitness and F-Measure.

TABLE Ⅱ: Performance (fitness) and F-Measure comparison of CPSO-KM with other algorithms

Datasets		K-Means	PSO	CPSO	CPSO-KM
SynSet1 (N=150, C=3, D=2)	Mean fitness	**48.4797**	**48.4797**	**48.4797**	**48.4797**
	Std. Dev	**0**	**0**	**0**	**0**
	F-Measure	**1**	**1**	**1**	**1**
	Std. Dev	**0**	**0**	**0**	**0**
SynSet2 (N=400, C=4, D=3)	Mean fitness	1.8643E+04	1.8642E+04	1.8642E+04	**1.8641E+04**
	Std. Dev	2.8918	2.7051	2.3612	**1.7709**
	F-Measure	0.8882	0.8882	0.8882	**0.8882**
	Std. Dev	0.0001	0.0001	0.0001	**0.0001**
Iris (N=150, C=3, D=4)	Mean fitness	92.2927	91.8262	85.4199	**78.9657**
	Std. Dev	26.8443	27.0535	20.3712	**0.2539**
	F-Measure	0.8231	0.8239	0.8266	**0.8339**
	Std. Dev	0.0121	0.0115	0.0160	**0.0198**
Haberman (N=306, C=2, D=3)	Mean fitness	3.0573E+04	3.0563E+04	3.0554E+04	**3.0546E+04**
	Std. Dev	56.8038	57.4285	54.6581	**50.1978**
	F-Measure	0.8481	0.8494	08504	**0.8480**
	Std. Dev	0.0147	0.0147	0.0149	**0.0150**
Hayes-roth (N=132, C=3, D=4)	Mean fitness	288.2780	286.0900	285.6593	**282.5963**
	Std. Dev	11.1871	8.1570	8.1873	**3.9336**
	F-Measure	0.6992	0.6093	0.6087	**0.6333**
	Std. Dev	0.0486	0.0371	0.0375	**0.0412**

Conclusion

In this paper, a CPSO-KM algorithm is proposed to solve the clustering problem, which is based on combination of the particle swarm optimization (PSO) and the K-Means algorithm. CPSO and K-Means are run sequentially and the cluster centers obtained by CPSO are the initial centers of the K-Means. The combination method solves the problem of sensitivity to the choice of initial cluster centers. The fitness experiments on three synthetic and four real data sets show that the CPSO-KM algorithm can obtain better performance than K-Means, PSO and CPSO. The experimental results also show that CPSO-KM can attain the better clusters than other algorithms in terms of F-Measure. Future work includes research into the application of the CPSO-KM algorithm to solve more demanding data sets with more complex structure, such as document clustering.

Acknowledgment

This work is supported by the Postdoctoral Science Foundation of China (No. 2013M542219), Research Fund for the Doctoral Program of Higher Education of China (No. 20120171110034), Guangdong Province Natural Science Fund Projects (No. S2012020011067), Doctoral Foundation of Guangdong University of Education(No. 2012ARF05), and Guangdong Province Colleage Specialty Construction Projects (No. 2013LYM_0062), Guangdong Provincial Key Platform and Major Scientific Research Projects of Education(2014KQNCX169, 2014KTSCX194).

References

[1] E. Forgy, Cluster analysis of multivariate data: Efficiency vs. inter-pretability of classifications, Biometrics, 21(3), 1965, 768-780

[2] S. Z. Selim, M. A. Ismail, K-means type algorithms: A generalized convergence theorem and characterization of local optimality, IEEE Transaction Pattern Analysis Machine Intelligence, 6, 1984, 6, 81-87

[3] P. Sunantasaengtong, S. Chivapreecha, Mixed K-means and GA-based weighted distance fingerprint algorithm for indoor localization system, in: Proceedings of the 2014 IEEE Region 10 Conference TENCON, 2014, 1-5

[4] M. Davodi, H. R. Modares, E. Reihani, M. Davodi, A. Sarikhani, Coherency approach by hybrid PSO, K-Means clustering method in power system, in: Proceedings of the IEEE 2nd International Power and Energy Conference, 2008, 1203-1207

[5] X. Sun, B. Zhang, L. Yang, S. Li, L. Gao, Hyperspectal image clustering using ant colony optimization(ACO) improved by K-means algorithm, in: Proceedings of the 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 2, 2010 , V2-474-V2-478

[6] Q. Tao, Z. Huang, C. Gu et al., Optimization of green agri-food supply chain network using chaotic PSO algorithm, in: Proceedings of the 2013 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), 2013, 462-467

[7] J. Kennedy, R. C. Eberhart, Particle swarm optimization, in: Proceedings of the IEEE International Conference Neural Network, 1995, 4, 1942-1948

[8] R. C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the 6th International symposium on Micromachine Human Science, 1995, 39-43

[9] Y. Shi, R. C. Eberhart, A modified particle swarm optimizer, in Proceedings IEEE World Congress Computer Intelligence, 1998, pp. 69-73.

[10] R. C. Eberhart, Y. H. Shi, Particle swarm optimization: Developments, applications and resources, in: Proceedings IEEE Congress Evolutionary Computation, 2001, 81-86.

[11] L. Zhang, Y. Zhang, Y. Yang. A Decorrelation algorithm based on chaos adaptive. Journal of Electronics & Information Technology, 2009, 31(8), 1825-1829.

[12] ftp://ftp.ics.uci.edu/pub/machine-learning-databases/