A Kernel-Based Membrane Clustering Algorithm

Jinyu Yang[1], Ru Chen[1], Guozhou Zhang[1], Hong Peng[1], Jun Wang[2], and Agustín Riscos-Núñez[3]

[1] School of Computer and Software Engineering, Xihua University, Chengdu 610039, Sichuan, China
ph.xhu@hotmail.com
[2] School of Electrical and Information Engineering, Xihua University, Chengdu 610039, Sichuan, China
[3] Research Group of Natural Computing, Department of Computer Science and Artificial Intelligence, University of Seville, 41012 Sevilla, Spain

Abstract. The existing membrane clustering algorithms may fail to handle the data sets with non-spherical cluster boundaries. To overcome the shortcoming, this paper introduces kernel methods into membrane clustering algorithms and proposes a kernel-based membrane clustering algorithm, KMCA. By using non-linear kernel function, samples in original data space are mapped to data points in a high-dimension feature space, and the data points are clustered by membrane clustering algorithms. Therefore, a data clustering problem is formalized as a kernel clustering problem. In KMCA algorithm, a tissue-like P system is designed to determine the optimal cluster centers for the kernel clustering problem. Due to the use of non-linear kernel function, the proposed KMCA algorithm can well deal with the data sets with non-spherical cluster boundaries. The proposed KMCA algorithm is evaluated on nine benchmark data sets and is compared with four existing clustering algorithms.

1 Introduction

Membrane computing, introduced by Păun [1], was inspired by the structure and functioning of living cells and their cooperation in tissues, organs, and biological neural networks [2]. Membrane computing is a class of distributed parallel computing models, known as P systems or membrane systems. In the past, a variety of variants of P systems have been proposed [3–11], and they have been applied to real-world problems, for example, robots [12,13], image processing [14–17], signal processing [18–20], fault diagnosis [21–25], ecology and system biology [26–28].

Clustering is a class of machine learning techniques, which is the task of finding natural partitioning within a data set such that patterns within the same cluster are more similar than those within different clusters. Membrane clustering algorithms (MCA) are a kind of partitioning clustering algorithms realized in

the framework of membrane computing. In recent years, a number of membrane clustering algorithms have been developed. Zhao et al. [29] discussed an improved clustering algorithm that used a cell-like membrane system to realize classical k-medoids algorithm. In Peng et al. [30], an evolution-communication membrane system has been used to propose a fuzzy cluster approach, called Fuzzy-MC. Two automatic membrane clustering algorithms were discussed [31,32], where an object representation with control bits and a membrane system with active membranes, respectively, were used to realize the corresponding automatic clustering mechanisms. Peng et al. [33] presented a multiobjective fuzzy clustering approach based on tissue-like membrane systems. The experimental results on a lot of benchmark datasets have shown that compared to the existing clustering algorithms, membrane clustering algorithms offer a more competitive approach due to three advantages: good clustering performance, better convergence and stronger robustness. However, the existing membrane clustering algorithms have a shortcoming: their cluster boundaries are spherical. Therefore, these membrane clustering algorithms may suffer a low clustering quality for the data sets with non-spherical cluster boundaries.

To overcome the critical shortcoming, this paper introduces kernel methods [34] into membrane clustering algorithms and proposes a kernel-based membrane clustering algorithm, called KMCA. Based on the principle of kernel methods, data samples are mapped to a high-dimensional feature space by a non-linear kernel function, and then KMCA algorithm is realized in the high-dimensional feature space. Due to the use of nonlinear kernel function, KMCA algorithms have a non-spherical cluster boundary. Meanwhile, KMCA algorithms can hold the advantages of membrane clustering algorithms, for example, good clustering performance, better convergence and stronger robustness, even if for the data sets with non-spherical cluster boundaries.

The remainder of this paper is organized as follows. Section 2 discusses in detail the proposed kernel-based membrane clustering algorithm. Experimental results are provided in Sect. 3. Conclusions are given in Sect. 4.

2 KMCA Algorithms

2.1 Kernel Clustering Problems

Let $X = \{x_1, x_2, \ldots, x_n\}$ be a data set of n data points in R^d, where $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, $i = 1, 2, \ldots, n$. The data set X is partitioned into k clusters, $C_1, C_2, \ldots, C_k$. Denote by $z_1, z_2, \ldots, z_k$ the centers of the k clusters, respectively. In classical k-means algorithm, the objective function to be optimized is as follows:

$$J_m(z_1, z_2, \ldots, z_k) = \sum_{i=1}^{k} \sum_{x_j \in C_i} ||x_j - z_i||^2 \tag{1}$$

Based on the principle of kernel methods, data set X is mapped into a high-dimension feature space by a nonlinear function $\phi(x)$.

Let $Y = \{\phi(x_1), \phi(x_2), \ldots, \phi(x_n)\}$ be the mapped data set. In the feature space, the distance between $\phi(x_i)$ and $\phi(x_j)$ can be computed as follows:

$$d(\phi(x_i), \phi(x_j)) = ||\phi(x_i) - \phi(x_j)|| = \sqrt{K(x_i, x_i) - 2K(x_i, x_j) + K(x_j, x_j)} \quad (2)$$

where $K(x_i, x_j)$ is called kernel function. The widely used kernel functions are: linear kernel function, polynomial kernel function, Gaussian kernel function and sigmoid kernel function. Gaussian kernel function is defined by

$$K(x_i, x_j) = \exp\left(-\frac{||x_i - x_j||^2}{2\sigma^2}\right) \quad (3)$$

where $\sigma > 0$ is the width of Gaussian kernel. Therefore, kernel clustering problem can be viewed as the following optimization problem:

$$\min_{z_1, z_2, \ldots, z_k} J(z_1, z_2, \ldots, z_k) = \sum_{i=1}^{k} \sum_{x_j \in C_i} \left(K(x_j, x_j) - 2K(x_j, z_i) + K(z_i, z_i)\right) \quad (4)$$

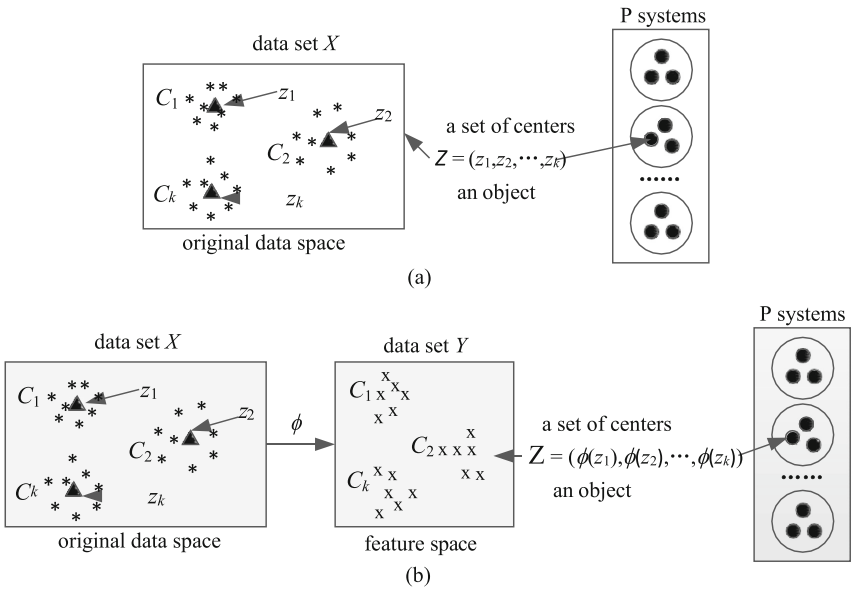

Fig. 1. (a) MCA algorithm; (b) Proposed KMCA algorithm

2.2 Basic Idea

Figure 1(a) shows the principle of the existing membrane clustering algorithms (MCA), where an object in P systems is used to denote a set of candidate

cluster centers. Based on the objective function (1), a P system is considered to determine the optimal cluster centers. However, a shortcoming that MCA algorithm suffers from is that it may fail to deal with the data sets with non-spherical cluster boundaries.

To overcome the shortcoming, this paper presents a kernel-based membrane clustering algorithm, KMCA. In KMCA, data set X in original data space is mapped to a high-dimension feature space by a non-linear map ϕ, shown in Fig. 1(b). Assume that Y is the mapped data set in the feature space, i.e., $Y = \{\phi(x_1), \phi(x_2), \ldots, \phi(x_n)\}$. KMCA algorithm will design a tissue-like P system to solve the kernel clustering problem (3), i.e., determining an optimal set of cluster centers. Although cluster boundaries obtained by tissue-like P system are still spherical in the feature space, KMCA algorithm can well handle the data sets with non-spherical cluster boundaries due to using nonlinear mapping ϕ that is expressed implicitly by kernel function.

2.3 Algorithm Implementation

The proposed KMCA algorithm is based on a tissue-like P system, which is designed to solve the kernel clustering problem (3). The tissue-like P system consists of q cells, and is defined as follows

$$\Pi = (O, w_1, \ldots, w_q, R_1, \ldots, R_q, R', i_0) \tag{5}$$

where

(1) O is a finite non-empty alphabet (of objects);
(2) $w_i (1 \leq i \leq q)$ is finite set of objects initially present in cell i;
(3) $R_i (1 \leq i \leq q)$ is finite set of evolution rules in cell i;
(4) R' is finite set of communication rules of the form $(i, u/v, 0)$, which denotes communication rule between cell i and the environment, $i = 1, 2, \ldots, q$;
(5) i_0 indicates the output region of the system.

In what follows, we describe in detail several components of the tissue-like P system.

(1) ***Object presentation***

The role of the designed tissue-like P system is to search the optimal cluster centers, so its each object is used to express a group of candidate cluster centers. Let $Z = \{z_1, z_2, \ldots, z_k\}$ is a set of candidate cluster centers, and each center z_i is a d-dimension vector, $z_i = (z_{i1}, z_{i2}, \ldots, z_{id}) \in R^d$, $i = 1, 2, \ldots, k$. Thus, each object in cells can be formally expressed by

$$Z = (z_{11}, z_{12}, \ldots, z_{1d}, z_{21}, z_{22}, \ldots, z_{2d}, \ldots, z_{k1}, z_{k2}, \ldots, z_{kd}) \tag{6}$$

In the designed tissue-like P system, each cell has a best object, and denote by Z^i_{best} the best object in ith cell, $i = 1, 2, \ldots, k$. There is an object in the environment, Z_{best}, which is the best object in entire system.

(2) ***Evaluation mechanism***

In the designed tissue-like P system, an improved velocity-position model is used to evolve the objects in cells. The improved velocity-position model can be described as follows.

$$\begin{cases} V_i = w \cdot Z_i + c_1 r_1 (P_i - Z_i) + c_2 r_2 (Z_{best}^i - Z_i) + c_3 r_3 (Z_{best} - Z_i) \\ Z_i = Z_i + V_i \end{cases} \tag{7}$$

where P_i is the best position of object Z_i found so far, w is inertia weight, c_1, c_2, c_3 are learning factors, and r_1, r_2, r_3 are three random real numbers in [0,1]. In the implementation, the following decreasing strategy of inertia weight is used:

$$w = w_{max} - (w_{max} - w_{min}) t / t_{max} \tag{8}$$

where $w_{max} = 0.9$, $w_{min} = 0.4$, and t_{max} is maximum computing step number (or maximum iteration number).

(3) ***Communication mechanism***

The tissue-like P system uses communication mechanism to achieve the exchange and sharing of objects between each cell and the environment. The communication mechanism usually is provided by communication rules. In the tissue-like P system, the used communication rule is described as follows:

$$< i, Z_{best}^i / Z_{best}, 0 >, i = 1, 2, \ldots, q \tag{9}$$

The communication rule indicates that Z_{best}^i in cell i is transported into the environment to update its best object, and Z_{best} in the environment is transported into cell i for object evolution at next step.

(4) ***Halting and output***

The designed tissue-like P system adopts a simple halting condition, namely, maximum computing step number. The tissue-like P system will continue to execute until the halting condition is reached, thus, the system halts. When the system halts, the best object stored in the environment ($i_0 = 0$) is regarded as final computing result, namely, the determined optimal cluster centers.

3 Experimental Results

3.1 Data Sets

In order to evaluate the proposed KMCA algorithm, nine widely used benchmark data sets from UCI [35] machine learning repository were used in experiments, shown in Table 1. Simulation experiment is implemented in python [36] on a Microsoft Window computer.

Table 1. The data sets used in experiments

Data set	Number of data points (n)	Data dimension (d)	Number of clusters (k)
Wine	178	13	3
Lung cancer	32	56	3
Seeds	210	7	3
Lenses	24	4	3
Hayes-roth	132	5	3
Dermatology	366	33	6
Iris	150	4	3
Leuk	72	40	3
Zoo	101	16	7

3.2 The Compared Methods

In experiments, the proposed KMCA algorithm was compared with four existing clustering algorithms, which are illustrated as follows.

(1) K-means (KM): a classical k-means algorithm.
(2) K-means+PSO (KM+PSO) [37]: a k-means algorithm optimized by particle swarm optimization (PSO).
(3) Kernel k-means (KKM) [38]: a kernel-based k-means algorithm.
(4) Kernel k-means+PSO (KKM+PSO) [39]: a kernel-based k-means algorithm optimized by particle swarm optimization (PSO).

3.3 Parameter Setting

For the proposed KMCA algorithm, parameters of tissue P system are assigned: the number of cells $q = 3$, the number of objects in each cell $m = 100$, and maximum number of iterations is 100. In KM+PSO and KKM+PSO, population size is $m = 100$, maximum number of iterations is 100, and $c_1 = c_2 = 2.0, w = 1$. In KMCA, KKM and KKM+PSO, Gaussian function is used as kernel function, and the same parameter δ is used for each data set, but the different parameters δ are used in the different data sets: $\delta = 0.5$ for Wine, $\delta = 2.5$ for Lung cancer and Leuk, $\delta = 0.07$ for Iris, $\delta = 9$ for Zoo, $\delta = 0.13$ for Seeds, $\delta = 1$ for Lenses and Dermatology, and $\delta = 0.1$ for Hayes-roth.

3.4 Performance Measures

To measure the clustering quality of these clustering algorithms, two internal indexes and three external indexes were used in experiments. Internal index is used to indicate the clustering effect of data, while external index illustrates the accuracy of data clustering. In external index, n_{ij} denotes the number of samples that are actually in jth class but are classified into i class. $n_{i+} = \sum_j n_{ij}$ denotes the number of samples that are classified into i class in experiment. $n_{+j} = \sum_i n_{ij}$ denotes the number of samples that are classified into j class in the actual situation. The five indexes are as follows.

(1) Silhouette index (SI) [40]
Suppose that a denotes the average kernelized distance between a point and other points from the same cluster, and b is minimum average kernelized distance between a point and the points from other clusters. $SI \in [-1, 1]$ is the average of Silhouette widths of all samples. Generally, the higher SI means the better clustering quality. The Silhouette widths is defined by $SI = \frac{b-a}{\max\{a,b\}}$.

(2) CS measure [41]
CS measure can effectively measure the clusters with different densities and sizes. Generally, the smaller CS has the better clustering partition, i.e., better clustering effect. using a Gaussian Kernelized distance measure and transforming to the high dimensional feature space, CS measure is defined as follows.

$$CS = \frac{\sum_{i=1}^{k}[\frac{1}{n_i}\sum_{x_i \in C_i} \max_{x_q \in C_i}\{2(1 - K(x_i - x_q))\}]}{\sum_{i=1}^{k} \min_{j \in \{1,\ldots,k\}, j \neq i}\{2(1 - K(m_i - m_j))\}} \tag{10}$$

where $m_i = \sum_{x_j \in C_j} \frac{x_j}{n_j}$.

(3) Accuracy [42]
The clustering accuracy is defined by $p = \frac{\sum_i n_{ii}}{n}$.

(4) Adjusted rand index (ARI) [43]
$ARI \in [-1, 1]$. Generally, larger ARI value means that the clustering result is the more consistent with the actual situation. ARI index is defined by

$$ARI = \frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i+}}{2} \cdot \sum_j \binom{n_{+j}}{2}\right] / \binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i+}}{2} + \sum_j \binom{n_{+j}}{2}\right] - \left[\sum_i \binom{n_{i+}}{2} \cdot \sum_j \binom{n_{+j}}{2}\right] / \binom{n}{2}} \tag{11}$$

(5) Adjusted mutual index (AMI) [44]
Adjusted mutual index is defined by

$$AMI = \frac{MI - E(MI)}{\max(H(U), H(V)) - E(MI)} \tag{12}$$

where $MI(U, V) = \sum_{i=1}^{|U|}\sum_{j=1}^{|V|} p(i,j)\log(\frac{p(i,j)}{p(i)p^{'}(j)})$, $p(i,j) = \frac{|U_i \cap V_j|}{N}$, $H(U) = \sum_{i=1}^{|U|} p(i)\log p(i)$, $p(i) = \frac{|U_i|}{N}$ and $H(V) = \sum_{j=1}^{|V|} p^{'}(j)\log p^{'}(j)$, $p^{'}(j) = \frac{|V_j|}{N}$.

3.5 Experimental Results

In the experiments, the proposed KMCA algorithm and four compared algorithms were executed on nine data sets. Since these clustering algorithms contain some stochastic/random factors, they have been independently executed 20

times on each data set. For each performance measure and each data set, averages and standard deviations of the results obtained by these algorithms were computed, respectively. The average value indicates the average performance of each algorithm, while standard deviation reflects the robustness of the algorithm.

Tables 2, 3, 4 and 5 give the averages and standard deviations of 20 times for the five algorithms in terms of five performance measures, respectively.

Table 2. Comparison results of the proposed and compared algorithms in terms of SI and CS indexes

Data sets	Internal index	KKM	KKM+PSO	KMCA
Wine	SI	0.2808(0.0008)	0.2826(0.0005)	**0.2827(0.0003)**
	CS	1.3376(0.0047)	1.288(0.0044)	**1.2842(0.0018)**
Lung cancer	SI	0.0662(0.0478)	0.1059(0.0132)	**0.1156(0.0005)**
	CS	3.0802(0.9787)	2.743(0.1793)	**2.5796(0.0901)**
Seeds	SI	0.0685(0.0047)	0.069(0.0001)	**0.06928(0.0001)**
	CS	**0.9834**(0.0726)	1.0(2.4e-05)	1.0(**1.1e-05**)
Lense	SI	0.3536(0.0338)	0.3924(0.0)	**0.3924(0.0)**
	CS	1.151(0.102)	1.1337(0.0)	**1.1337(0.0)**
Hayes-roth	SI	**0.0112**(0.0009)	0.0103(0.002)	0.0107(**0.0008**)
	CS	1.0(**0.0**)	**0.9373**(0.1364)	1.0(3e-06)
Dermatology	SI	0.1927(0.031)	0.2233(**0.0073**)	**0.2245**(0.0087)
	CS	1.5852(0.4746)	**1.2882(0.0769)**	1.3109(0.0942)
Iris	SI	0.0610(0.0064)	0.0655(0.0016)	**0.0656(0.0004)**
	CS	**0.8833**(0.1590)	1.0001(4e-05)	1.0(**0.0**)
Leuk	SI	0.0715(0.0118)	0.0791(0.0002)	**0.0792(0.0001)**
	CS	**1.0286**(0.1728)	1.0375(0.0026)	1.0373(**0.0025**)
Zoo	SI	0.4498(0.0662)	0.4787(0.0277)	**0.5029(0.0256)**
	CS	1.5972(0.3879)	1.4062(0.1259)	**1.3673(0.1183)**

Table 2 shows comparison results of the five clustering algorithms in terms of SI and CS indexes on nine data sets. For SI index, apart from Hayes-roth, KMCA can achieve the best performance on each of eight data sets. Meanwhile, compared with KKM and KKM+PSO, KMCA has the smallest standard deviations on eight data sets except Dermatology. For CS index, KMCA has the best performance on Wine, Lung cancer, Lense and Zoo, and achieves the smallest standard deviations on seven sets except Hayes-roth and Dermatology. The comparison results indicate that KMCA has the better performance compared with KKM and KKM+PSO, however, the advantage is not obvious based on results on CS index.

Tables 3, 4 and 5 provide the comparison results of five clustering algorithms on nine data sets in terms of three external indexes, including clustering accuracy, adjusted rand index and adjusted mutual index. It can be obviously observed from Tables 3, 4 and 5 that compared with other four clustering algorithms,

Table 3. Comparison results of the proposed and compared algorithms in terms of clustering accuracy

Data sets	KM	KM+PSO	KKM	KKM+PSO	KMCA
Wine	0.8533(0.1706)	0.9648(0.0039)	0.9477(0.0073)	0.9654(0.0036)	**0.9657(0.0016)**
Lung cancer	0.4781(0.0648)	0.5625(0.0453)	0.4984(0.0659)	0.5641(0.0413)	**0.5828(0.0247)**
Seeds	0.8021(0.1461)	0.886(0.004)	0.8762(0.0503)	0.895(0.0046)	**0.8967(0.0034)**
Lense	0.5833(0.152)	0.5375(0.107)	0.5896(0.1105)	0.8063(0.0622)	**0.8312(0.0596)**
Hayes-roth	0.4492(0.0585)	0.4095(0.0692)	0.4564(0.0807)	0.4966(0.0618)	**0.5087(0.0525)**
Dermatology	0.6396(0.1521)	0.7963(**0.071**)	0.6334(0.1573)	0.8004(0.0748)	**0.8344**(0.082)
Iris	0.7977(0.1153)	0.88(**0.0**)	0.7993(0.1100)	0.8907(0.0129)	**0.894**(0.0029)
Leuk	0.9153(0.1094)	0.9583(**0.0**)	0.8708(0.1394)	0.9708(0.0041)	**0.9715**(0.0030)
Zoo	0.7475(0.076)	0.7401(0.0769)	0.7223(0.0988)	0.7554(0.0717)	**0.7728(0.065)**

Table 4. Comparison results of the proposed and compared algorithms in terms of adjusted rand index (ARI)

Data sets	KM	KM+PSO	KKM	KKM+PSO	KMCA
Wine	0.7102(0.2504)	0.8946(0.0122)	0.8462(0.0206)	0.8962(0.0112)	**0.8972(0.0051)**
Lungcancer	0.0682(0.0870)	0.1193(0.0458)	0.0822(0.0765)	0.1187(0.0454)	**0.1282(0.0176)**
Seeds	0.6092(0.1673)	0.694(0.0125)	0.6873(0.0478)	0.7142(0.0105)	**0.7181(0.0079)**
Lenses	0.1953(0.2004)	0.1059(0.15)	0.1513(0.13)	0.5415(0.0674)	**0.5686(0.0646)**
Hayes-roth	0.0539(**0.0422**)	0.0073(0.0642)	0.078(0.0803)	0.103(0.0711)	**0.1258**(0.056)
Dermatology	0.5742(0.2084)	0.7617(**0.0713**)	0.5759(0.1821)	0.7695(0.0737)	**0.7975**(0.0846)
Iris	0.6331(0.0968)	0.7021(**0.0**)	0.6302(0.0956)	0.7205(0.0279)	**0.7291**(0.0064)
Leuk	0.8232(0.1505)	0.8803(**0.0**)	0.7661(0.1823)	0.9148(0.0115)	**0.9167**(0.0083)
Zoo	0.7211(0.1017)	0.5864(0.1186)	0.6972(0.1284)	0.6994(0.0744)	**0.73(0.0696)**

Table 5. Comparison results of the proposed and compared algorithms in terms of adjusted mutual index (AMI)

Data sets	KM	KM+PSO	KKM	KKM+PSO	KMCA
Wine	0.6990(0.2361)	0.8696(0.0104)	0.8242(0.0205)	0.8709(0.0101)	**0.8716(0.0050)**
Lungcancer	0.1006(0.1141)	0.1814(0.0458)	0.0904(0.0917)	0.1752(0.054)	**0.1907(0.017)**
Seeds	0.5762(0.1628)	0.6619(0.009)	0.654(0.0493)	0.6777(0.0086)	**0.6812(0.0066)**
Lenses	0.2761(0.164)	0.2104(0.1661)	0.2291(**0.1171**)	0.4424(0.1246)	**0.4925**(0.1194)
Hayes-roth	0.0563(**0.0496**)	0.004(0.0638)	0.0945(0.088)	0.125(0.0816)	**0.1494**(0.0591)
Dermatology	0.6952(0.185)	0.8811(0.021)	0.7114(0.1363)	0.8835(**0.0192**)	**0.8892**(0.0224)
Iris	0.6395(0.0788)	0.7050(**0.0**)	0.6113(0.1214)	0.7137(0.0202)	**0.7233**(0.0044)
Leuk	0.7955(0.1502)	0.8542(**0.0**)	0.7450(0.1723)	0.8878(0.0112)	**0.8896**(0.0081)
Zoo	0.7409(0.0549)	0.7194(0.0667)	0.7426(0.0649)	0.7679(0.0417)	**0.7939(0.0289)**

KMCA achieves the best performances on nine data sets. Moreover, KMCA also has the smallest standard deviations on seven data sets except Iris and Leuk.

In summary, KMCA can achieve a good clustering performance and also is a robust clustering algorithm.

4 Conclusions

This paper discussed a kernel-based membrane clustering algorithm, KMCA, based on the principle of kernel methods. The proposed KMCA algorithm is suitable to handle a challenge of the existing membrane clustering algorithms: it fails to deal with the clustering on the data sets with non-spherical cluster boundaries. In principle, samples in original data space are mapped to a high-dimension space via a non-linear kernel function, and then membrane clustering algorithm is used to cluster the mapped data sets. As usual, a tissue-like P system is designed to determine the optimal cluster centers in the high-dimension space. Based on the principle of kernel methods, all computations are completed in original data space according to the used kernel function. Therefore, although membrane clustering algorithms have spherical cluster boundary, the proposed KMCA can deal with the data sets with non-spherical cluster boundaries. Experimental results on nine benchmark data sets demonstrate the advantage over other clustering methods

Acknowledgment. This work was partially supported by the National Natural Science Foundation of China (No. 61472328), Chunhui Project Foundation of the Education Department of China (Nos. Z2016143 and Z2016148), the Innovation Fund of Postgraduate, Xihua University (No. ycjj2018184), and Research Foundation of the Education Department of Sichuan province (No. 17TD0034), China.

References

1. Păun, Gh.: Computing with membranes. J. Comput. Syst. Sci. **61**(1), 108–143 (2000)
2. Păun, Gh.: Membrane Computing: An Introduction. Springer, Berlin (2002). https://doi.org/10.1007/978-3-642-56196-2
3. Cavaliere, M.: Evolution–communication P systems. In: Păun, Gh., Rozenberg, G., Salomaa, A., Zandron, C. (eds.) WMC 2002. LNCS, vol. 2597, pp. 134–145. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-36490-0_10
4. Freund, R., Păun, Gh., Pérez-Jiménez, M.J.: Tissue-like P systems with channel-states. Theor. Comput. Sci. **330**(1), 101–116 (2005)
5. Bernardini, F., Gheorghe, M.: Population P systems. J. Univ. Comput. Sci. **10**(5), 509–539 (2004)
6. Păun, Gh., Păun, R.: Membrane computing and economics: numerical P systems. Fundam. Inform. **73**(1–2), 213–227 (2006)
7. Ciencialová, L., Csuhaj-Varjú, E., Kelemenová, A., Vaszil, G.: Variants of P colonies with very simple cell structure. Int. J. Comput. Commun. Control **IV**(3), 224–233 (2009)
8. Ionescu, M., Păun, Gh., Yokomori, T.: Spiking neural P systems. Fundam. Inform. **71**, 279–308 (2006)
9. Song, T., Pan, L., Păun, Gh.: Spiking neural P systems with rules on synapses. Theor. Comput. Sci. **529**, 82–95 (2014)
10. Peng, H., et al.: Competitive spiking neural P systems with rules on synapses. IEEE Trans. NanoBiosci. **16**(8), 888–895 (2018)

11. Peng, H., et al.: Spiking neural P systems with multiple channels. Neural Netw. **95**, 66–71 (2017)
12. Buiu, C., Vasile, C., Arsene, O.: Development of membrane controllers for mobile robots. Inf. Sci. **187**, 33–51 (2012)
13. Wang, X., et al.: Design and implementation of membrane controllers for trajectory tracking of nonholonomic wheeled mobile robots. Integr. Comput.-Aided Eng. **23**(1), 15–30 (2016)
14. Zhang, G., Gheorghe, M., Li, Y.: A membrane algorithm with quantum-inspired subalgorithms and its application to image processing. Natural Comput. **11**(4), 701–717 (2012)
15. Díaz-Pernil, D., Berciano, A., Peña-Cantillana, F., Gutiérrez-Naranjo, M.A.: Segmenting images with gradient-based edge detection using membrane computing. Pattern Recogn. Lett. **34**(8), 846–855 (2013)
16. Peng, H., Wang, J., Pérez-Jiménez, M.J.: Optimal multi-level thresholding with membrane computing. Digit. Sig. Process. **37**, 53–64 (2015)
17. Alsalibi, B., Venkat, I., Al-Betar, M.A.: A membrane-inspired bat algorithm to recognize faces in unconstrained scenarios. Eng. Appl. Artif. Intell. **64**, 242–260 (2017)
18. Zhang, G., Liu, C., Rong, H.: Analyzing radar emitter signals with membrane algorithms. Math. Comput. Model. **52**(11–12), 1997–2010 (2010)
19. Peng, H., Wang, J., Pérez-Jiménez, M.J., Riscos-Núñez, A.: The framework of P systems applied to solve optimal watermarking problem. Sig. Process. **101**, 256–265 (2014)
20. Wang, J., Shi, P., Peng, H.: Membrane computing model for IIR filter design. Inf. Sci. **329**, 164–176 (2016)
21. Xiong, G., Shi, D., Zhu, L., Duan, X.: A new approach to fault diagnosis of power systems using fuzzy reasoning spiking neural P systems. Math. Problems Eng. **2013**(1), 211–244 (2013)
22. Wang, J., Shi, P., Peng, H., Pérez-Jiménez, M.J., Wang, T.: Weighted fuzzy spiking neural P system. IEEE Trans. Fuzzy Syst. **21**(2), 209–220 (2013)
23. Wang, T., et al.: Fault diagnosis of electric power systems based on fuzzy reasoning spiking neural P systems. IEEE Trans. Power Syst. **30**(3), 1182–1194 (2015)
24. Peng, H., Wang, J., Shi, P., Pérez-Jiménez, M.J., Riscos-Núñez, A.: Fault diagnosis of power systems using fuzzy tissue-like P systems. Integr. Comput.-Aided Eng. **24**, 401–411 (2017)
25. Peng, H.: Fault diagnosis of power systems using intuitionistic fuzzy spiking neural P systems. IEEE Trans. Smart Grid **9**(5), 4777–4784 (2018)
26. Gheorghe, M., Manca, V., Romero-Campero, F.J.: Deterministic and stochastic P systems for modelling cellular processes. Natural Comput. **9**(2), 457–473 (2010)
27. García-Quismondo, M., Levin, M., Lobo-Fernández, D.: Modeling regenerative processes with membrane computing. Inf. Sci. **381**, 229–249 (2017)
28. García-Quismondo, M., Nisbet, I.C.T., Mostello, C.S., Reed, M.J.: Modeling population dynamics of roseate terns (sterna dougallii) in the Northwest Atlantic Ocean. Ecol. Model. **68**, 298–311 (2018)
29. Zhao, Y., Liu, X., Qu, J.: The K-medoids clustering algorithm by a class of P system. J. Inf. Comput. Sci. **9**(18), 5777–5790 (2012)
30. Peng, H., Wang, J., Pérez-Jiménez, M.J., Riscos-Núñez, A.: An unsupervised learning algorithm for membrane computing. Inf. Sci. **304**, 80–91 (2015)
31. Peng, H., Wang, J., Shi, P., Riscos-Núñez, A., Pérez-Jiménez, M.J.: An automatic clustering algorithm inspired by membrane computing. Pattern Recogn. Lett. **68**, 34–40 (2015)

32. Peng, H., Wang, J., Shi, P., Pérez-Jiménez, M.J., Riscos-Núñez, A.: An extended membrane system with active membrane to solve automatic fuzzy clustering problems. Int. J. Neural Syst. **26**(2), 1–17 (2016)
33. Peng, H., Shi, P., Wang, J., Riscos-Núñez, A., Pérez-Jiménez, M.J.: Multiobjective fuzzy clustering approach based on tissue-like membrane systems. Knowl.-Based Syst. **125**, 74–82 (2017)
34. Shawe-Taylor, J., Cristianini, N.: Kernel Methods for Pattern Analysis. Cambridge University Press, Cambridge (2004)
35. UCI. http://archive.ics.uci.edu/ml/datasets.html
36. Python. https://www.python.org/
37. Merwe, D.W., Engelbrecht, A.P.: Data clustering using particle swarm optimization. In: 2003 Congress on Evolutionary Computation (CEC 2003), pp. 215–220 (2003)
38. Zhang, R., Rudnicky, A.I.: A large scale clustering scheme for kernel k-means. In: Proceedings of 16th International Conference on Pattern Recognition, vol. 4, pp. 289–292 (2002)
39. Wei, X.H., Zhang, K.: An improved PSO-means clustering algorithm based on kernel methods. J. Henan Univ. Sci. Technol.: Nat. Sci. **32**(2), 41–43 (2011)
40. Rousseeuw, P.J.: Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. **20**(20), 53–65 (1987)
41. Chou, C.H., Su, M.C., Lai, E.: A new cluster validity measure and its application to image compression. Pattern Anal. Appl. **7**(2), 205–220 (2004)
42. Congalton, R.G., Green, K.: Assessing the Accuracy of Remotely Sensed Data: Principles and Practices. CRC Press, Boca Raton (2009)
43. Hubert, L., Arabie, P.: Comparing partitions. J. Classif. **2**(1), 193–218 (1985)
44. Zhang, J., Niu, Y., He, W.: Using genetic algorithm to improve fuzzy k-NN. In: International Conference on Computational Intelligence and Security, pp. 475–479 (2008)