文章编号: 1001-0920 (2016) 12-2122-09 DOI: 10.13195/j.kzyjc.2015.1488

基于多代表点的大规模数据模糊聚类算法

陈爱国, 王士同

(江南大学 数字媒体学院，江苏 无锡 214122)

摘 要: 针对传统模糊聚类在大规模数据场景下, 由于内存的限制不能一次装载所有数据, 以及在通过聚类捕捉数据的潜在结构和描述各个类时仅使用单个代表点存在信息量不足的问题, 提出一种基于多代表点的大规模数据模糊聚类算法. 该算法通过对大规模数据进行分块, 在对每个数据块进行聚类时使用多个代表点描述捕捉到的数据的潜在结构和各个类信息, 并通过考虑代表点与代表点之间在聚类过程中的约束关系, 提高最后聚类结果的精度. 在模拟数据集和真实数据集上的 3 组实验验证了所提出算法的有效性.

关键词: 大规模数据；模糊聚类；增量式聚类；多代表点

中图分类号: TP391 **文献标志码:** A

Fuzzy clustering algorithm based on multiple medoids for large-scale data

CHEN Ai-guo, *WANG Shi-tong*

(School of Digital Media，Jiangnan University，Wuxi 214122，China. Correspondent：CHEN Ai-guo，E-mail: agchen@jiangnan.edu.cn)

Abstract: For the problem that the traditional fuzzy clustering is not able to load all the data at a time because of the limited memory in the application scenario for large-scale data, and using a single medoid is insufficient to capture the underlying structure of data and describe each cluster, a fuzzy clustering algorithm based on multiple medoids for large-scale data is presented. The algorithm handles data chunk by chunk, and uses multiple medoids to represent the underlying data structure and each cluster information in one chunk, and the pairwise constraints from the relationship between two identified medoids are taken into account. These mechanisms improve the accuracy of the final clustering results. The effectiveness of the proposed algorithm is verified by three sets of experiments on a simulated dataset and two real datasets.

Keywords: large-scale data；fuzzy clustering；incremental clustering；multiple medoids

0 引 言

随着计算机网络技术的快速发展, 各种信息正以前所未有的速度高速增长, 如何从大量的数据中提取有效信息是当前大规模数据应用的关键点, 也是难点[1-3]. 聚类是一种行之有效的数据分析工具, 能够从纷繁复杂的数据中发现隐藏在其内部的潜在结构, 在无监督的情况下将数据按照其性质的亲疏程度划分到具有相似性质组成的多个类中[4-5]. 目前, 聚类已经广泛应用于数据分析、模式识别、图像处理、市场研究等领域[6-10].

在大规模数据应用场景下, 由于计算机内存的限制, 要处理的数据不能一次性全部载入, 传统的聚类算法也因此失效. 针对此问题, 各种特定于大规模数据的聚类算法相继被研究. 文献 [11] 提出了基于随机抽样的 CLARA 聚类算法; 文献 [12] 提出了一种基于平衡树结构的增量式聚类算法 BIRCH; 文献 [13] 提出了基于谱聚类的增量式聚类算法. 上述聚类算法都是硬聚类方法, 即一个数据要么属于该类, 要么不属于该类, 没有其他情况存在. 这与现实生活的实际情况不大一致, 使用模糊聚类中的隶属度来表示一个数据属于所有类的程度更符合现实情况[14]. 针对大规模数据的模糊聚类近年来也得到了广泛的研究. SPFCM 算法[15]和 OFCM 算法[16]是在经典的模糊 C 均值 (FCM) 算法[17]的基础上扩展到大规模数据应用场景下的两个算法. spkFCM 算法和 okFCM 算法[18]是上面两个算法的核化版本. 上述模糊聚类算法处理的都

收稿日期: 2015-12-02；**修回日期:** 2016-03-18.

基金项目: 国家自然科学基金项目(61272210)；江苏省杰出青年基金项目(BK20140001)；江苏省自然科学基金项目(BK20130155).

作者简介: 陈爱国 (1975−), 男, 讲师, 博士, 从事模式识别与机器学习的研究；王士同 (1964−), 男, 教授, 博士生导师, 从事人工智能和机器学习等研究.

是对象型的大规模数据, 目前针对关系型大规模数据的模糊聚类算法主要有 OFCMD 算法和 HOFCMD 算法[19]. 这两种算法也都是将要处理的数据进行分块, 每个块中的各个类都使用单个代表点来表示. 然而, 使用单个代表点来呈现一个类的潜在结构往往是不够的[20], 如果使用更多的代表点来表示一个类, 则可以更充分地呈现类的内部结构信息.

受此思想的启发, 本文提出一种新的基于多代表点的针对大规模数据的模糊聚类算法. 该算法在 OFCMD 算法和 HOFCMD 算法的基础上, 使用多代表点并同时考虑这样一个事实: 属于同一个类的多个代表点, 在下一轮聚类时, 聚类为相同类的概率应该最大. 将这样的事实用代表点对之间的约束来表示, 并引入到目标函数中, 形成本文所提出的算法. 在模拟数据集和真实数据集上的 3 组实验验证了所提出算法的有效性.

1　大规模数据模糊聚类算法

FCM 算法是由 Dunn[21]首次提出, 后由Bezdek[17]正式提出的一种聚类算法. FCM 算法能够运行的前提是: 所有的数据要能一次性地全部载入内存. 当面对大规模数据时, 因为机器内存的限制, 要处理的数据无法进行一次性的全部装载, 所以需要对 FCM 算法进行进一步研究, 使其能适合大规模数据的处理需求. SPFCM 算法和 OFCM 算法是在 FCM 算法的基础上扩展到大规模数据应用场景的两个模糊聚类算法.

SPFCM 算法和 OFCM 算法针对大规模数据都使用了分块方法, 而且在聚类过程中都使用了权重 FCM (wFCM) 机制. 在 SPFCM 算法中, 每个块的中心点通过将前一块所获取的带权重中心点信息与当前数据块合并后聚类所得. 最后一块聚类出的中心点即为整个大规模数据的中心点, 每个块中各类中心点的权重为

$$w_k = \sum_{i=1}^{N_b+m} z_{ki} w_i,\ 1 \leqslant k \leqslant C. \tag{1}$$

其中: w_k 为第 k 类的中心点权重, N_b 为第 b 块的数据个数, m 为前一数据块的中心点数目, C 为分类数, z_{ki} 为第 i 个数据点属于 k 类的隶属度, w_i 为第 i 个数据点的权重. 在 OFCM 算法中, 每个块的中心点通过每个块的数据分别获得, 整个大规模数据的中心点通过对所有块的带权重中心点组成的数据单独进行聚类获得. 每个块中各类中心点的权重为

$$w_k = \sum_{i=1}^{N_b} z_{ki} w_i,$$

$$1 \leqslant k \leqslant C,\ w_i = 1,\ 1 \leqslant i \leqslant N_b. \tag{2}$$

以上两种基于中心点的模糊聚类算法都能处理对象型大规模数据集, 但当数据为关系型时, 上述两种算法则无法处理.

HOFCMD 算法和 OFCMD 算法是既能处理大规模数据又能处理关系型数据的两种模糊聚类算法. HOFCMD 算法和 OFCMD 算法与 SPFCM 算法和 OFCM 算法相比, 最大的区别是对聚类结果使用不同的表示方式. 前两个算法使用类中真实存在的代表点表示该类, 后两个算法使用类中数据的均值即中心点表示该类, 这个中心点并不真实存在. 使用真实存在的代表点表示聚类信息具有更好的解释性.

HOFCMD 算法和 OFCMD 算法对大规模数据的处理方式与 SPFCM 算法和 OFCM 算法相类似. 其中的不同是, HOFCMD 算法和 OFCMD 算法使用 wFCMd (weighted fuzzy c medoids) 处理代表点的权重, 目标函数为

$$J_{\mathrm{wFCMd}} = \sum_{k=1}^{C}\sum_{i=1}^{N} z_{ki}^m w_i \|\boldsymbol{x}_i - \boldsymbol{v}_k\|^2. \tag{3}$$

其中: z_{ki} 为数据 i 属于 k 类的隶属度, C 为分类数, N 为数据总个数, m 为模糊度指数, w_i 为数据 i 的权重, $\boldsymbol{v}_k$ 为第 k 类的代表点, $\|\boldsymbol{x}_i - \boldsymbol{v}_k\|^2$ 为数据 i 与 k 类的代表点之间的距离.

以上 4 种方法都是只使用单个中心点或代表点来表示一个类信息, 正如引言中所讨论的, 使用单个中心点或代表点来表示一个类提供的信息往往是不足够的, 所以本文提出一个新的具有多代表的模糊聚类算法.

2　基于多代表点的大规模数据模糊聚类算法

针对目前大规模数据模糊聚类算法仅使用单个代表点, 对聚类出来的类信息表示不够充分的缺点[20], 本文在基于单个代表点的大规模数据模糊聚类算法基础上, 提出新的基于多代表点的大规模数据模糊聚类算法 (MMFCA). 该算法的目标函数为

$$J = \sum_{k=1}^{C}\sum_{i=1}^{N}\sum_{j=1}^{N} z_{ki} w_{kj} d_{ij} + \frac{\alpha}{2}\sum_{k=1}^{C}\sum_{i=1}^{N} z_{ki}^2 + \frac{\beta}{2}\sum_{k=1}^{C}\sum_{j=1}^{N} w_{kj}^2 - \frac{\gamma}{2}\sum_{\boldsymbol{x}_m,\boldsymbol{x}_n\in\Omega}\sum_{k=1}^{C}(z_{km}^2 + z_{kn}^2). \tag{4}$$

$$\text{s.t.}\ \sum_{k=1}^{C} z_{ki} = 1,\ i = 1, 2, \cdots, N; \tag{5}$$

$$z_{ki} \geqslant 0,\ i = 1, 2, \cdots, N,\ k = 1, 2, \cdots, C; \tag{6}$$

$$\sum_{j=1}^{N} w_{kj} = 1,\ k = 1, 2, \cdots, C; \tag{7}$$

$$w_{kj} \geqslant 0,\ k = 1, 2, \cdots, C,\ j = 1, 2, \cdots, N. \tag{8}$$

其中: z_{ki} 为第 i 个数据点属于第 k 类的隶属度; w_{kj} 为第 j 个数据点作为第 k 类代表点的权重; d_{ij} 为第 i 个数据点和第 j 个数据点的距离度量; $\boldsymbol{x}_m$、$\boldsymbol{x}_n$ 为前一次聚类所产生的属于同一类的最具代表性的两个数据点. 式(4)的目标函数由4项组成, 第1项是考虑数据点的隶属度和权重的距离度量总和, 第2项和第3项分别是隶属度和权重的两个惩罚项, 第4项是考虑代表点对的约束关系. 该项的加入基于这样的考虑: 前一轮聚类产生属于同一个类最具代表性的两个点, 在下一轮聚类中作为同类的两个代表点的概率应该尽可能地大, 即原属于同一个类权重最大的两个代表点, 在下一轮聚类时其两个代表点的隶属度平方和应尽可能地大. 集合 Ω 中的代表点约束对通过计算每个类中样本点作为代表点的权重来获得, 本文中选取每个类中代表点权重最大的两个数据点作为该约束对.

该算法的任务是在给定集合 Ω 和满足约束(5)~(8)的情况下, 最小化目标函数. 通过采用拉格朗日条件极值的优化理论和Karush-Kuhn-Tucker(KKT)[22]条件获得目标函数的最优解.

2.1 算法优化

为了求得式(4)在有约束条件下的极值, 可以利用拉格朗日乘子法进行求解. 在考虑约束情况下, 拉格朗日函数如下:

$$\begin{aligned}L=&\sum_{k=1}^{C}\sum_{i=1}^{N}\sum_{j=1}^{N}z_{ki}w_{kj}d_{ij}+\frac{\alpha}{2}\sum_{k=1}^{C}\sum_{i=1}^{N}z_{ki}^2+\\&\frac{\beta}{2}\sum_{k=1}^{C}\sum_{j=1}^{N}w_{kj}^2-\frac{\gamma}{2}\sum_{\boldsymbol{x}_m,\boldsymbol{x}_n\in\Omega}\sum_{k=1}^{C}(z_{km}^2+z_{kn}^2)+\\&\sum_{i=1}^{N}\lambda_i\Big(\sum_{k=1}^{C}z_{ki}-1\Big)+\sum_{k=1}^{C}\eta_k\Big(\sum_{j=1}^{N}w_{kj}-1\Big)+\\&\sum_{k=1}^{C}\sum_{i=1}^{N}\varphi_{ki}z_{ki}+\sum_{k=1}^{C}\sum_{j=1}^{N}\phi_{kj}w_{kj},\end{aligned}\tag{9}$$

其中 λ_i、η_k、φ_{ki}、ϕ_{kj} 为拉格朗日乘子. 为了获得 z_{ki} 最优解, 给出其相关KKT条件如下:

$$\frac{\partial L}{\partial z_{ki}}=0,\tag{10}$$

$$\varphi_{ki}\geqslant 0,\tag{11}$$

$$\varphi_{ki}z_{ki}=0.\tag{12}$$

由式(9)和(10)得到

$$z_{ki}=\frac{1}{\gamma-\alpha}\Big(\lambda_i+\varphi_{ki}+\sum_{j=1}^{N}w_{kj}d_{ij}\Big).\tag{13}$$

由式(5)和(13)得到

$$\lambda_i=\frac{1}{C}\Big(\gamma-\alpha-\sum_{k=1}^{C}\varphi_{ki}-\sum_{k=1}^{C}\sum_{j=1}^{N}w_{kj}d_{ij}\Big).\tag{14}$$

将式(14)代入(13)可得

$$z_{ki}=\frac{1}{C}+\frac{1}{\gamma-\alpha}\Big[\varphi_{ki}+\pi_{ki}-\frac{1}{C}\sum_{h=1}^{C}(\varphi_{hi}+\pi_{hi})\Big],\tag{15}$$

其中 $\pi_{ki}=\sum_{j=1}^{N}w_{kj}d_{ij}$.

由式(11)可知 $\varphi_{ki}\geqslant 0$. 下面分两种情况进行讨论:

1) 当 $\varphi_{ki}=0$ 时, 式(15)变为

$$z_{ki}=\frac{1}{C}+\frac{1}{\gamma-\alpha}\Big(\pi_{ki}-\frac{1}{C}\sum_{h=1}^{C}\pi_{hi}\Big).$$

由式(6)和(12)可知

$$z_{ki}=\frac{1}{C}+\frac{1}{\gamma-\alpha}\Big(\pi_{ki}-\frac{1}{C}\sum_{h=1}^{C}\pi_{hi}\Big)\geqslant 0.$$

2) 当 $\varphi_{ki}>0$ 时, 由式(12)可知 $z_{ki}=0$.

根据上述两种情况的讨论可知, 隶属度的值也分为对应的两种情况. 为了进一步推导, 根据隶属度值将类别分成两个子集: 隶属度等于0的类别 z^- 和隶属度大于零的类别 z^+, 分别为

$$z^-=\{k:z_{ki}=0\},\ z^+=\{k:z_{ki}>0\}.$$

因为当 $k\in z^-$ 时, $z_{ki}=0$, 所以仅需要推导当 $k\in z^+$ 时 z_{ki} 的迭代公式即可. 当 $k\in z^+$ 时, $z_{ki}>0$, 由式(12)可知 $\varphi_{ki}=0$. 将式(15)中的类别分成两个子类后, 改写为

$$z_{ki}=\frac{1}{C}+\frac{1}{\gamma-\alpha}\Big[\pi_{ki}-\frac{1}{C}\Big(\sum_{h\in z^+}\pi_{hi}+\sum_{h\in z^-}\Big(\varphi_{hi}+\pi_{hi}\Big)\Big)\Big].\tag{16}$$

当 $k\in z^-$ 时, $z_{ki}=0$, 由式(15)可得

$$\varphi_{ki}+\pi_{ki}=\frac{1}{C}\Big[\alpha-\gamma+\sum_{h=1}^{C}\Big(\varphi_{hi}+\pi_{hi}\Big)\Big].\tag{17}$$

由式(17)可以看出, $\varphi_{ki}+\pi_{ki}$ 的值独立于 k, 这意味着当 $k\in z^-$ 时, 这些值是相等的, 所以式(17)可以进一步改写成

$$\varphi_{ki}+\pi_{ki}=\frac{1}{C}\Big[\alpha-\gamma+\sum_{h\in z^+}\pi_{hi}+|z^-|\Big(\varphi_{ki}+\pi_{ki}\Big)\Big].$$

推导可得

$$\varphi_{ki}+\pi_{ki}=\frac{1}{|z^+|}\Big(\alpha-\gamma+\sum_{h\in z^+}\pi_{hi}\Big).\tag{18}$$

将式(18)代入(16)得到

$$z_{ki}=\frac{1}{|z^+|}-\frac{1}{\alpha-\gamma}\Big(\pi_{ki}-\frac{1}{|z^+|}\sum_{h\in z^+}\pi_{hi}\Big).\tag{19}$$

最终得到 z_{ki} 的迭代公式

$$z_{ki}=\begin{cases}0,\ k\in z^-;\\ \dfrac{1}{|z^+|}-\dfrac{1}{\alpha-\gamma}\Big(\pi_{ki}-\dfrac{1}{|z^+|}\sum\limits_{h\in z^+}\pi_{hi}\Big),\ k\in z^+.\end{cases}\tag{20}$$

为了获得 w_{kj} 的最优解, 给出其相关 KKT 条件如下:

$$\frac{\partial L}{\partial w_{kj}}=0,\tag{21}$$

$$\phi_{kj}\geqslant 0,\tag{22}$$

$$\phi_{kj}w_{kj}=0.\tag{23}$$

由式 (9) 和 (21) 得到

$$\sum_{i=1}^{N}z_{ki}d_{ij}+\beta w_{kj}+\eta_k+\phi_{kj}=0.\tag{24}$$

推导式 (24) 得到

$$w_{kj}=-\frac{1}{\beta}\Big(\eta_k+\phi_{kj}+\sum_{i=1}^{N}z_{ki}d_{ij}\Big).\tag{25}$$

由式 (7) 和 (25) 得到

$$\eta_k=-\frac{1}{N}\Big(\beta+\sum_{j=1}^{N}\phi_{kj}+\sum_{j=1}^{N}\sum_{i=1}^{N}z_{ki}d_{ij}\Big).\tag{26}$$

将式 (26) 代入 (25) 得到

$$w_{kj}=\frac{1}{N}-\frac{1}{\beta}\Big[\phi_{kj}+\omega_{kj}-\frac{1}{N}\sum_{l=1}^{N}\Big(\phi_{kl}+\omega_{kl}\Big)\Big],\tag{27}$$

其中

$$\omega_{kj}=\sum_{i=1}^{N}z_{kj}d_{ij}.\tag{28}$$

由式 (22) 可知 $\phi_{kj}\geqslant 0$. 下面分两种情况进行讨论:

1) 当 $\phi_{kj}=0$, 式 (27) 变为

$$w_{kj}=\frac{1}{N}-\frac{1}{\beta}\Big[\omega_{kj}-\frac{1}{N}\sum_{l=1}^{N}\omega_{kl}\Big].$$

由式 (8) 和 (23) 可知

$$w_{kj}=\frac{1}{N}-\frac{1}{\beta}\Big[\omega_{kj}-\frac{1}{N}\sum_{l=1}^{N}\omega_{kl}\Big]\geqslant 0.$$

2) 当 $\phi_{kj}>0$, 由式 (23) 可知 $w_{kj}=0$.

根据上述两种情况的讨论可知, 权重值 w_{kj} 分为对应的两种情况. 为了进一步推导, 根据权重值 w_{kj} 将数据点分成两个子集: 权重 w_{kj} 等于 0 的数据点子集 w^- 和权重 w_{kj} 大于 0 的数据点子集 w^+, 分别为

$$w^-=\{j:w_{kj}=0\},\ w^+=\{j:w_{kj}>0\}.$$

当 $j\in w^-$ 时, $w_{kj}=0$, 所以仅需推导当 $j\in w^+$ 时 w_{kj} 的迭代公式即可. 当 $j\in w^+$ 时, $w_{kj}>0$, 由式 (23) 可知 $\phi_{kj}=0$. 将式 (27) 根据权重值分成两个子类后, 改写为

$$w_{kj}=\frac{1}{N}-\frac{1}{\beta}\Big[\omega_{kj}-\frac{1}{N}\Big(\sum_{l\in w^+}\omega_{kl}+\sum_{l\in w^-}\Big(\phi_{kl}+\omega_{kl}\Big)\Big)\Big].\tag{29}$$

当 $j\in w^-$ 时, $w_{kj}=0$, 由式 (27) 可得

$$\phi_{kj}+\omega_{kj}=\frac{1}{N}\Big[\beta+\sum_{l=1}^{N}\Big(\phi_{kl}+\omega_{kl}\Big)\Big].\tag{30}$$

由式 (30) 可以看出, $\phi_{kj}+\omega_{kj}$ 的值独立于 j, 这意味着当 $j\in w^-$ 时, 这些值是相等的, 所以式 (30) 可以进一步改写为

$$\phi_{kj}+\omega_{kj}=\frac{1}{N}\Big[\beta+\sum_{l\in w^+}\omega_{kl}+|w^-|\Big(\phi_{kj}+\omega_{kj}\Big)\Big].$$

进一步推导可得

$$\phi_{kj}+\omega_{kj}=\frac{1}{|w^+|}\Big(\beta+\sum_{l\in w^+}\omega_{kl}\Big).\tag{31}$$

将式 (31) 代入 (29) 得到

$$w_{kj}=\frac{1}{|w^+|}-\frac{1}{\beta}\Big(\omega_{kj}-\frac{1}{|w^+|}\sum_{l\in w^+}\omega_{kl}\Big).\tag{32}$$

最终得到 w_{kj} 的迭代公式

$$w_{kj}=\begin{cases}0,\ j\in w^-;\\ \dfrac{1}{|w^+|}-\dfrac{1}{\beta}\Big(\omega_{kj}-\dfrac{1}{|w^+|}\sum\limits_{l\in w^+}\omega_{kl}\Big),\ j\in w^+.\end{cases}\tag{33}$$

2.2 算法步骤

根据上述 MMFCA 算法的推导过程和得到的迭代公式, 给出 MMFCA 算法的具体步骤如下.

输入: 各块的距离矩阵 $\boldsymbol{D}^b_{N_b\times N_b}$, 聚类数 C, 表示每个类信息所使用的代表点数 t, 参数 α、β、γ 和终止阈值 ε;

输出: 数据所属类别标识向量 $\boldsymbol{q}$.

Step 1: 初始化代表点集合 $S=\varnothing$, 代表点对约束集合 $\Omega=\varnothing$.

Step 2: for $b=1,2,\cdots,B$, do.

Step 3: 使用过程 Compute_z_w 计算当 $\gamma=0$ 时, 块的隶属度矩阵 $\boldsymbol{Z}_b$ 和块的权重矩阵 $\boldsymbol{W}_b$.

Step 4: for $k=1,2,\cdots,C$, do.

Step 5: 设置当前循环次数计数器 $l=0$, 集合 $A=1,2,\cdots,N_b$.

Step 6: repeat.

Step 7: 更新循环次数计数器 $l=l+1$.

Step 8: 根据 $h=\arg\max\limits_{j\in A}w^b_{kj}$ 获得最具代表性的点的索引.

Step 9: 保存最具代表性的点到代表点集合中, 即 $S = S\bigcup\{\boldsymbol{x}_h\}$.

Step 10: 从集合 A 中删除最具代表性的点的索引, 即 $A = A\backslash h$.

Step 11: until 获取到 t 个最具代表性的点, 即 $l = t$.

Step 12: 获取当前类中最具代表性的两个点, 形成代表点约束对 (m_1, m_2).

Step 13: 将代表点约束对 (m_1, m_2) 加入代表点约束对集合 Ω.

Step 14: end for.

Step 15: end for.

Step 16: 将前面步骤获取的代表点集合 S 形成距离矩阵 $\boldsymbol{R}_{|S|\times|S|}$ 和代表点对约束集合 Ω, 使用过程 Compute_z_w 计算当 $\gamma > 0$ 时, 隶属度矩阵 $\boldsymbol{Z}$ 和权重矩阵 $\boldsymbol{W}$.

Step 17: 设置最终的代表点集合 $S^f = \varnothing$.

Step 18: for $k = 1, 2, \cdots, C$, do.

Step 19: 根据 $h = \underset{1\leqslant j\leqslant|S|}{\arg\max} w_{kj}$ 获取权重最大的点作为最终类的代表点.

Step 20: 将最终类的代表点保存到最终代表点集合 S^f 中.

Step 21: end for.

Step 22: 根据 $q_j = \underset{1\leqslant k\leqslant C}{\arg\min}\|\boldsymbol{x}_j - \boldsymbol{s}_k\|^2$, 最终获得每个数据的类别.

块距离矩阵 $\boldsymbol{D}^b_{N_b\times N_b}$ 表示第 b 块的距离矩阵, N_b 表示第 b 块的数据个数, B 表示总的分块数, $|S|$ 表示集合 S 中的元素个数.

Compute_z_w 过程的具体步骤如下.

输入: 距离矩阵 $\boldsymbol{D}$, 分类数 C, 参数 α、β、γ 和终止阈值 ε;

输出: 隶属度矩阵 $\boldsymbol{Z}$ 和权重矩阵 $\boldsymbol{W}$.

Step 1: 初始化隶属度矩阵 $\boldsymbol{Z}^0$, 设置当前步骤 $l = 0$.

Step 2: 更新当前步骤 $l = l + 1$, 使用式 (33) 更新权重矩阵 $\boldsymbol{W}^l$, 使用式 (20) 更新隶属度矩阵 $\boldsymbol{Z}^l$.

Step 3: $\|\boldsymbol{Z}^l - \boldsymbol{Z}^{l-1}\| < \varepsilon$, 则算法结束, 否则转至 Step 2.

在 Compute_z_w 过程的 Step 2 中需要使用 w^+ 和 w^- 两个子集以及 z^+ 和 z^- 两个子集. 确定 w^+ 和 w^- 两个子集的具体步骤如下.

输入: 隶属度权重矩阵 $\boldsymbol{Z}$, 距离矩阵 $\boldsymbol{D}$, 参数 β;

输出: 子集 w^+ 和 w^-.

Step 1: 初始化子集 $w_0^+ = \varnothing$, $w_0^- = 1, 2, \cdots, N$, 设置当前步骤 $l = 0$.

Step 2: 更新当前步骤 $l = l + 1$, 根据

$$s = \underset{j\in w_{l-1}^-}{\arg\min}\left\{\sum_{i=1}^{N} z_{ki}d_{ij}\right\}$$

获取满足条件的数据点 s, 更新当前步骤的子集

$$w_l^+ = w_{l-1}^+ + \{s\},\ w_l^- = w_{l-1}^- - \{s\}.$$

Step 3: 根据式 (33) 计算 w_{kh} 的值, 其中

$$h = \underset{j\in w_l^+}{\arg\max}\left\{\sum_{i=1}^{N} z_{ki}d_{ij}\right\}.$$

如果 $w_{kh} > 0$, 则转至 Step 2, 否则设置 $w^+ = w_{l-1}^+$, $w^- = w_{l-1}^-$, 结束算法.

确定 z^+ 和 z^- 两个子集的具体步骤如下.

输入: 权重矩阵 $\boldsymbol{W}$, 距离矩阵 $\boldsymbol{D}$, 参数 α、γ, 代表点对约束集合 Ω;

输出: 子集 z^+ 和 z^-.

Step 1: 初始化子集 $z_0^+ = \varnothing$, $z_0^- = 1, 2, \cdots, C$, 设置当前步骤 $l = 0$.

Step 2: 更新当前步骤 $l = l + 1$, 根据

$$s = \underset{k\in z_{l-1}^-}{\arg\min}\left\{\sum_{j=1}^{N} w_{kj}d_{ij}\right\}$$

获取满足条件的类别 s, 更新当前步骤的子集

$$z_l^+ = z_{l-1}^+ + \{s\},\ z_l^- = z_{l-1}^- - \{s\}.$$

Step 3: 根据式 (20) 计算 z_{hi} 的值, 其中

$$h = \underset{k\in z_l^+}{\arg\max}\left\{\sum_{j=1}^{N} w_{kj}d_{ij}\right\}.$$

如果 $z_{hi} > 0$, 则转至 Step 2, 否则设置 $z^+ = z_{l-1}^+$, $z^- = z_{l-1}^-$, 结束算法.

MMFCA 算法的时间复杂度为

$$O(B(N_b^2 + tC) + (|S|^2 + C)).$$

该时间复杂度由 $O(B(N_b^2 + tC)$ 和 $O(|S|^2 + C)$ 两部分组成. 第 1 部分表示处理 B 个数据块所对应的时间花销, 第 2 部分表示处理由所有分块数据得出的代表点组成的新数据块的时间花销. 在实际应用中, N_b^2 往往远大于 tC, $|S|^2$ 往往远大于 C, 所以整个算法的时间复杂度可以近似地表示为 $O(BN_b^2 + |S|^2)$.

3 实验分析

3.1 实验设置

为了验证本文所提出 MMFCA 算法的有效性, 本节使用一组模拟数据集和两种的常见真实数据集进行实验, 同时, 选择两个基于单代表点的大规模数据模糊聚类算法 OFCMD 和 HOFCMD 进行聚类性能对比.

实验采用归一化互信息 (NMI)[23]和芮氏指标

(RI)[24]对实验结果进行评价. 两种评价指标的取值范围均为[0,1], 取值越大代表对应算法的聚类性能越好.

MMFCA 算法中参数α、β、γ的取值分别为0.1、4、0.05, 所有算法的终止阈值都设置为$\varepsilon = 10^{-5}$, HOFCMD 算法和OFCMD 算法中的参数按照文献的推荐, 设置模糊指数$m = 1.7$. 所涉及的代表点的初始化采用与文献[19]相同的策略. 对于MMFCA 算法中涉及到的隶属度初始化, 本文采用如下方法.

输入: 距离矩阵$\boldsymbol{D}_{N\times N}$, 聚类数$C$;

输出: 隶属度矩阵$\boldsymbol{Z}_{C\times N}$.

Step 1: 通过$h = \arg\min\limits_{1\leqslant j\leqslant N}\sum\limits_{i=1}^{N}\|\boldsymbol{x}_j - \boldsymbol{x}_i\|^2$计算第一代表点的索引.

Step 2: 将代表点放入代表点集合$M = \{\boldsymbol{x}_h\}$, 设置类计数器$l = 1$.

Step 3: 设置当前类的代表点的隶属度值$z_{lh} = 1$, 设置当前类的非代表点的隶属度值$z_{lj} = 0,\ j = 1, 2, \cdots, N,\ j \neq h$.

Step 4: 设置类计数器$l = l + 1$.

Step 5: 通过$h = \arg\max\limits_{1\leqslant j\leqslant N; x_j\notin M}\min\limits_{1\leqslant k\leqslant |M|}\|\boldsymbol{x}_j - \boldsymbol{m}_k\|$计算当前类的代表点索引. 其中: $\boldsymbol{m}_k$为集合M中的第k个元素, $|M|$表示集合M的元素个数.

Step 6: 将代表点放入代表点集合$M = M + \{\boldsymbol{x}_h\}$.

Step 7: 设置当前类的代表点的隶属度值$z_{lh} = 1$, 设置当前类的非代表点的隶属度值$z_{lj} = 0,\ j = 1, 2, \cdots, N,\ j \neq h$.

Step 8: 重复执行Step 4 ~ Step 7, 直到当前的类计数器$l = C$.

实验使用的硬件环境为Intel I7-5600U 2.60 GHz 8 G RAM, 操作系统为Windows 8 64 位, 软件环境为Matlab R2012b. 为了使3 个算法的结果具有可比性, 在对数据进行随机分块时, 每次运行中3 个算法使用相同的随机分块, 实验结果所列数据都是在运行10 次后求均值所得.

3.2　模拟数据集实验和结果分析

在模拟数据集2D15[25]上验证所提出MMFCA 算法的有效性, 数据集含有5 000 个共15 类的2 维数据, 不同类的数据在分布上有部分重合, 对应的数据分布如图1 所示.

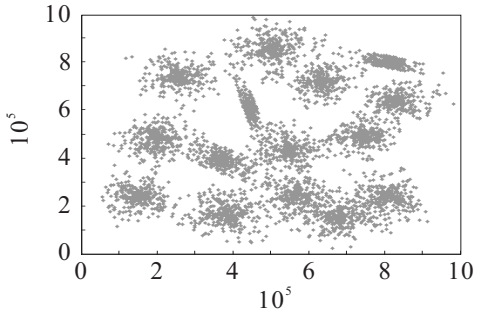

图 1　2D15 数据集数据分布

为了考察算法在不同分块大小下的聚类性能, 采用占数据总量的1%、5%、10%、15%、20% 和25% 六种数据块大小进行实验. 3 种算法在不同分块大小下的实验结果如表1 和表2 所示.

表 1　3 种算法在2D15 数据集上的NMI 性能对比

Chunk size/%	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
1	0.894 3	0.020 6	0.843 4	0.020 6	0.900 3	0.012 1
5	0.884 2	0.021 3	0.894 2	0.031 2	0.916 1	0.020 0
10	0.909 2	0.020 1	0.894 2	0.027 8	0.930 6	0.014 5
15	0.933 4	0.012 4	0.892 4	0.026 1	0.940 3	0.012 8
20	0.939 4	0.013 0	0.902 5	0.014 6	0.946 9	0.011 0
25	0.944 6	0.008 7	0.884 3	0.024 1	0.946 7	5.966 5e-04

表 2　3 种算法在2D15 数据集上的RI 性能对比

Chunk size/%	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
1	0.976 5	0.006 0	0.963 2	0.007 4	0.978 2	0.003 9
5	0.972 4	0.007 4	0.976 0	0.010 7	0.981 8	0.006 9
10	0.979 7	0.006 4	0.975 8	0.009 0	0.986 5	0.005 1
15	0.986 9	0.005 0	0.975 3	0.008 0	0.990 3	0.004 4
20	0.989 6	0.004 8	0.979 2	0.004 5	0.992 4	1.579 5e-04
25	0.991 9	0.002 1	0.974 1	0.007 4	0.992 4	1.568 0e-04

由表 1 可见:

1) 3 个算法随着块的数据量的增大, 其 NMI 均值都呈现逐渐增大的趋势, 主要原因是该模拟数据集总数据量不大, 通过分块, 每类对应的数据量不够充足, 聚类出不高的 NMI 均值, 当分块数据量增大时, 每块中各类的数据量随之增大, 用来聚类的信息更加充分, 对应的 NMI 均值也将随之提高.

2) OFCMD 算法大多数情况取得比 HOFCMD 算法更高的 NMI 均值, 主要是由于 HOFCMD 算法将前一块获得的代表点作为历史数据与当前块的数据结合在一起进行当前块代表点的获取, 该机制弱化了历史数据对最终整个数据集代表点求解的影响. OFCMD 算法将所有块的代表点一起组成一个新的块数据, 近似均等地作用于最终代表点的产生, 这种机制的不同使得 OFCMD 算法比 HOFCMD 算法在多数情况下取得更好的 NMI 评价值.

3) 6 种不同分块大小的情况下, MMFCA 算法得到的 NMI 均值比 HOFCMD 算法和 OFCMD 算法都要高, 其相应的 NMI 标准方差却比 2 个算法的 NMI 标准方差都要小. 这主要是由于在 MMFCA 算法中, 每个类使用多代表点来表示, 可获得更加丰富的类信息, 同时, 通过将代表点对之间的约束关系引入聚类过程, 进一步增强了最终的聚类效果.

由表 2 可见:

1) 3 个算法随着分块数据量的增大在 RI 均值上同样呈现出增大的变化趋势, 该变化趋势与表 1 所观察到的 NMI 均值变化趋势具有相同的原因.

2) 由于 HOFCMD 算法与 OFCMD 算法运行机制的差异, OFCMD 算法的 RI 均值在大多数情况下要超过 HOFCMD 算法.

3) 无论分块大小如何变化, MMFCA 算法所获得的 RI 均值都要高于其他 2 个对比算法, 其原因同样是由于多代表点和代表点之间的约束引入到 MMFCA 算法中所产生的结果.

3.3 手写数字数据集实验和结果分析

第 2 组实验在真实数据集 MNIST[26] 上验证 MMFCA 算法在不同分块大小下的有效性, 并与基于单代表点的 OFCMD 算法和 HOFCMD 算法进行性能对比. MNIST 是手写数字数据集, 包含 0～9 共 10 类手写的图片数据, 每张图片为 28×28 分辨率的灰度图, 每个像素的灰度值大小位于 0～255 之间. MNIST 数据集包含 60 000 张训练图片和 10 000 张测试图片. 本文将所有训练图片和所有测试图片合并组成实验所用数据. OFCMD 算法、HOFCMD 算法和 MMFCA 算法在进行该组实验时, 设置总数据量为 70 000, 类别数为 10, 维数为 784. 由于内存的限制, 本组实验的块的大小分别采用占总数据量的 1%、2%、4%、6%、8% 和 10% 进行选取, 实验结果如表 3 和表 4 所示.

由表 3 和表 4 可见:

1) 3 个算法随着分块数据量的增大, 均未出现如表 1 和表 2 所示的 NMI 均值和 RI 均值逐渐增大的趋势, 这主要是因为该组实验的真实数据量较大, 在分别按占总数据量的 1%、5%、10%、15%、20%、25% 进行数据分块时, 每块中的各类数据均已充足, 所以在充足的数据上进行聚类时, OFCMD 算法、HOFCMD 算法和 MMFCA 算法都没有出现如表 1 和表 2 中随着分块数据量的增加, NMI 均值和 RI 均值呈现有规律地逐步增大的趋势.

表 3 3 种算法在 MNIST 数据集上的 NMI 性能对比

Chunk size / %	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
1	0.214 6	0.019 9	0.117 9	0.034 4	0.350 8	0.016 3
2	0.223 8	0.019 3	0.106 5	0.048 1	0.343 2	0.019 0
4	0.250 7	0.034 7	0.102 4	0.085 5	0.361 3	0.031 1
6	0.248 6	0.031 3	0.113 3	0.061 1	0.363 7	0.023 7
8	0.233 4	0.030 1	0.120 6	0.058 5	0.347 8	0.029 7
10	0.224 7	0.034 7	0.108 4	0.038 1	0.361 2	0.031 2

表 4 3 种算法在 MNIST 数据集上的 RI 性能对比

Chunk size / %	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
1	0.833 6	0.010 4	0.539 9	0.034 1	0.860 1	0.010 0
2	0.831 5	0.018 5	0.495 2	0.056 5	0.854 7	0.017 4
4	0.858 6	0.011 2	0.482 7	0.092 2	0.860 9	0.010 7
6	0.850 1	0.013 5	0.497 0	0.085 0	0.855 8	0.012 8
8	0.840 8	0.005 1	0.526 5	0.072 3	0.861 5	0.004 6
10	0.840 3	0.013 2	0.483 9	0.086 6	0.856 2	0.012 1

2) OFCMD 算法在所有不同分块大小的情况下, 均取得了比 HOFCMD 算法更高的 NMI 均值和 RI 均值, 而且其对应的 NMI 标准方差值和 RI 标准方差值也均小于 HOFCMD 算法. 与模拟实验原因相同, HOFCMD 算法运行机制弱化了历史信息对最终代表点产生的影响, 导致最终 HOFCMD 算法产生的 NMI 均值和 RI 均值要比 OFCMD 算法产生的 NMI 均值和 RI 均值差.

3) MMFCA 算法在 6 种不同分块大小的情况下, NMI 均值和 RI 均值都大于另外 2 种算法, 而且对应的标准方差值也都比另外 2 种对比算法小. 其原因与模拟实验的分析相同, 因为多代表点对约束机制的引入使得最终的聚类评价指标 NMI 的值和 RI 的值都优于另外 2 种使用单代表点的算法, 这进一步从真实数据集上验证了 MMFCA 算法的有效性.

3.4　入侵检测数据集实验和结果分析

选择真实的 KDD CUP99[27]网络入侵检测数据集作为第 3 组实验数据. 数据集共含有 4 898 431 个样本, 每个样本包含 41 维数据, 其中 9 个为离散型, 其他为连续型. 通过对离散型数据进行二值化处理后, 每个样本含有 122 维数据. KDD CUP 99 将攻击类型分为 4 类, 进一步划分为 39 个小类. 实验将数据聚类成 2 类: 一类是标注为 Normal 的数据, 另一类是标注为其他类型的数据. 由于内存的限制, 实验采用占总数据量的 0.04%、0.06%、0.08%、0.10%、0.12% 和 0.14% 作为 6 种分块大小, 测试在不同分块大小情况下, MMFCA 算法的有效性, 实验结果如表 5 和表 6 所示.

表 5　3 种算法在 KDD 数据集上的 NMI 性能对比

Chunk size / %	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
0.04	0.466 3	0.016 7	0.393 1	0.025 8	0.549 6	0.015 1
0.06	0.471 6	0.020 1	0.327 1	0.039 4	0.530 7	0.019 6
0.08	0.511 6	0.042 6	0.289 3	0.086 7	0.558 7	0.032 9
0.10	0.490 2	0.034 1	0.368 3	0.065 2	0.559 6	0.028 2
0.12	0.488 8	0.025 8	0.413 0	0.057 3	0.547 8	0.015 7
0.14	0.484 7	0.021 9	0.345 2	0.079 8	0.542 3	0.019 4

表 6　3 种算法在 KDD 数据集上的 RI 性能对比

Chunk size / %	OFCMD		HOFCMD		MMFCA	
	Mean	Std	Mean	Std	Mean	Std
0.04	0.768 6	0.019 6	0.711 9	0.041 2	0.785 9	0.014 3
0.06	0.764 5	0.020 4	0.604 0	0.057 3	0.779 6	0.018 7
0.08	0.784 4	0.021 1	0.698 7	0.089 6	0.786 8	0.019 6
0.10	0.780 5	0.014 1	0.758 4	0.087 8	0.783 7	0.012 4
0.12	0.776 1	0.015 7	0.793 8	0.076 5	0.790 5	0.013 9
0.14	0.779 6	0.016 7	0.742 3	0.087 0	0.781 2	0.014 8

由表 5 和表 6 可见, MMFCA 算法在 NMI 和 RI 两个评价指标上都明显高于其他 2 种比较算法. 这是因为 MMIFCA 算法的两种机制的引入: 一是使用了多个代表点来表示聚类出来的各个类, 表示的类信息更加丰富, 最终对整个大规模数据的代表点的精确提取起到了促进作用; 二是通过将多代表点对之间的约束关系应用到聚类过程中, 提高了最终聚类的性能. 该实验再次验证了 MMIFCA 算法通过两种机制的使用能有效地提升聚类的精度.

4　结　论

本文针对传统模糊聚类算法在大规模数据场景下由于内存限制不能一次装载所有数据和使用单代表点表示每个类的潜在结构时存在信息量不足的问题. 对大规模数据进行分块, 在对每块进行分别聚类时使用多代表点表示各个类的内部结构, 并在不同轮聚类时考虑代表点对的关系: 上一轮聚类产生的权重大的代表点对, 在下一轮聚类时作为代表点的概率应该最大. 通过分块机制和对代表点对约束的考虑, 提出了应用于大规模数据场景下的基于多代表点的模糊聚类 MMFCA 算法. 通过 MMFCA 算法在模拟数据集、手写数字数据集和入侵检测数据集上的实验结果和与另外 2 种使用单代表点的 OFCMD 算法和 HOFCMD 算法进行性能对比和分析, 显示了所提出算法的有效性和优越性.

本文的创新点主要有两个: 一是通过使用多代表点来表示聚类结果中的每个类的内部结构信息, 使提供的类的内部结构信息更加丰富: 二是通过在聚类过程中考虑不同轮聚类时代表点对之间的约束关系, 提高了聚类精度. 此外, 本文算法也存在缺陷, 如计算复杂度高、耗时大、目标函数中的系数如何确定问题,

这也是该算法下一步的研究方向.

参考文献(References)

[1] Slaoui S C, Lamari Y. Clustering of large data based on the relational analysis[C]. Intelligent Systems and Computer Vision. Fez: IEEE, 2015: 1-7.

[2] Sheela G, Bharat T. A survey of big data in social media using data mining techniques[C]. Int Conf on Advanced Computing and Communication Systems. Coimbatore: IEEE, 2015: 1-6.

[3] Ben A, Ben H, Alimi A M. Survey on clustering methods: Towards fuzzy clustering for big data[C]. The 6th Int Conf of Soft Computing and Pattern Recognition. Tunis: IEEE, 2014: 331-336.

[4] Rui X, Wunsch D. Survey of clustering algorithms[J]. IEEE Trans on Neural Networks, 2005, 16(3): 645-678.

[5] Nisha Kaur P J. A survey of clustering techniques and algorithms[C]. The 2nd Int Conf on Computing for Sustainable Global Development. New Delhi: IEEE, 2015: 304-307.

[6] Koonsanit K, Jaruskulchai C, Eiumnoh A. Parameter-free K-means clustering algorithm for satellite imagery application[C]. Int Conf on Information Science and Applications. Suwon: IEEE, 2014: 1-6.

[7] Yong Y, Trouve A. A non-linear K-means algorithm and its application to unsupervised clustering[C]. The 6th Int Conf on Signal Processing. Beijing: IEEE, 2002, 2: 1146-1149.

[8] Wu B, Zhang Y, Hu B G, et al. Constrained clustering and its application to face clustering in videos[C]. IEEE Conf on Computer Vision and Pattern Recognition. Portland: IEEE, 2013: 3507-3514.

[9] Yang G, Xu O, Liang Z. Fuzzy clustering application in medical image segmentation[C]. The 6th Int Conf on Computer Science & Education. Singapore: IEEE, 2011: 826-829.

[10] Liang W Y. Apply rough set theory into the information extraction the application of the clustering[C]. The 5th Int Joint Conf on INC, IMS and IDC. Seoul: IEEE, 2009: 262-266.

[11] Kaufman L, Rousseeuw P J. Finding groups in data: An introduction to cluster analysis[M]. New York: Wiley-Interscience, 2009: 126-163.

[12] Livny T. Birch: An efficient data clustering method for very large databases[C]. SIGMOD'96 Proceedings of the 1996 Acm Sigmod Int Conf on Management of Data. New York: ACM, 1996, 25(2): 103-114.

[13] Kong T T, Tian Y, Shen H. A fast incremental spectral clustering for large data sets[C]. The 12th Int Conf on Parallel and Distributed Computing, Applications and Technologies. Gwangju: IEEE, 2011: 1-5.

[14] Suganya R, Shanthi R. Fuzzy C-means algorithm — A review[J]. Int J of Scientific and Research Publications, 2012, 2(11): 1-3.

[15] Hore P, Hall L, Goldgof D. Single pass fuzzy c-means[C]. IEEE Int Fuzzy Systems Conf. London: IEEE, 2007: 1-7.

[16] Hore P, Hall L, Goldgof D, et al. Online fuzzy c-means[C]. Annual Meeting of the North American Fuzzy Information Processing Society. New York: IEEE, 2008: 1-5.

[17] Bezdek J C, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm[J]. Computers & Geosciences, 1984, 10(2): 191-203.

[18] Havens T, Bezdek J, Leckie C, et al. Fuzzy c-means algorithms for very large data[J]. IEEE Trans on Fuzzy System, 2012, 20(6): 1130-1146.

[19] Labroche N. New incremental fuzzy c medoids clustering algorithms[C]. 2010 Annual Meeting of the North American Fuzzy Information Processing Society. Toronto: IEEE, 2010: 1-6.

[20] Mei J, Chen L. Fuzzy clustering with weighted medoids for relational data[J]. Pattern Recognition, 2010, 43(5): 1964-1974.

[21] Dunn J C. A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters[J]. J of Cybernetics, 1973, 3(3): 32-57.

[22] Boyd S, Vandenberghe L. Convex optimization[M]. Cambridge: Cambridge University Press, 2004: 244.

[23] Jing L, Ng K M, Huang Z. An entropy weighting K-means algorithm for subspace clustering of high-dimensional sparse data[J]. IEEE Trans on Knowledge and Data Engineering, 2007, 19(8): 1026-1041.

[24] Liu J, Motammed J, Carter J, et al. Distance-based clustering of CGH data[J]. Bioinformatics, 2006, 22(16): 1971-1978.

[25] Zhang Z, Havens T C. Scalable approximation of kernel fuzzy c-means[C]. IEEE Int Conf on Big Data. Silicon Valley: IEEE, 2013: 161-168.

[26] Kussul E, Baidyk T. Improved method of handwritten digit recognition tested on MNIST database[J]. Image and Vision Computing, 2004, 22(12): 971-981.

[27] Mohammad K S, Shams N. Analysis of KDD CUP 99 dataset using clustering based data mining[J]. Int J of Database Theory and Application, 2013, 6(5): 23-34.

(责任编辑：郑晓蕾)