计算机科学
COMPUTER SCIENCE

基于共享最近邻的自适应密度峰值聚类算法

王心耕, 杜韬, 周劲, 陈迪, 仵匀政

引用本文

王心耕, 杜韬, 周劲, 陈迪, 仵匀政. 基于共享最近邻的自适应密度峰值聚类算法[J]. 计算机科学, 2024, 51(8): 97-105.

WANG Xingeng, DU Tao, ZHOU Jin, CHEN Di, WU Yunzheng. Adaptive Density Peak Clustering Algorithm Based on Shared Nearest Neighbor [J]. Computer Science, 2024, 51(8): 97-105.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230500226

基于共享最近邻的自适应密度峰值聚类算法

王心耕[1]　杜　韬[1,2]　周　劲[1,2]　陈　迪[1]　仵匀政[1]
1 济南大学信息科学与工程学院　济南 250024
2 山东省网络环境智能计算技术重点实验室　济南 250024
(1054493071@qq.com)

摘　要　密度峰值聚类算法(DPC)是一种简单高效的无监督聚类算法,该算法虽能自动发现簇中心,实现任意形状数据的高效聚类,但依然存在一些缺陷。针对密度峰值聚类算法在定义相关度量值时未考虑数据的位置信息、聚类中心数目需要人工预先设定且分配样本点时易出现连锁反应这3个缺陷,提出一种基于共享最近邻的自适应密度峰值聚类算法。首先,利用共享最近邻重新定义局部密度等度量值,充分考虑了数据分布的局部特点,使样本点的空间分布特征得以更好地体现;其次,通过引入密度衰减现象让样本点自动聚集成微簇,实现了簇个数自适应确定和簇中心自适应选取;最后,提出一种两阶段的分配方法,先将微簇合并形成簇的主干部分,再用上一步分配好的簇主干指导剩余点的分配,避免了链式反应的发生。在二维合成数据集以及UCI数据集上的实现表明,相较于经典的密度峰值聚类算法及近年来对其提出的改进算法,在大多数情况下,所提算法表现出更优异的性能。

关键词:共享最近邻;密度峰值聚类;分配策略;聚类中心;密度衰减
中图分类号　TP391

Adaptive Density Peak Clustering Algorithm Based on Shared Nearest Neighbor

WANG Xingeng[1],DU Tao[1,2],ZHOU Jin[1,2],CHEN Di[1] and WU Yunzheng[1]
1 College of Information Science and Engineering,University of Jinan,Jinan 250024,China
2 Shandong Provincial Key Laboratory of Network Based Intelligent Computing,Jinan 250024,China

Abstract　Density peak clustering algorithm(DPC) is a simple and efficient unsupervised clustering algorithm. Although the algorithm can automatically discover cluster centers and realize efficient clustering of arbitrary shape data,it still has some defects. Aiming at the three defects of density peak clustering algorithm,which does not consider the location information of data when defining the correlation value,the number of clustering centers needs to be set manually in advance,and the chain reaction is easy to occur when distributing sample points,an adaptive density peak clustering algorithm based on shared nearest neighbor is proposed. Firstly,the shared nearest neighbor is used to redefine the local density and other measures,and the local characteristics of data distribution are fully considered,so that the spatial distribution characteristics of sample points can be better reflected. Secondly,by introducing the phenomenon of density attenuation,the sample points are automatically gathered into micro-clusters,which realizes the adaptive determination of cluster number and the adaptive selection of cluster center. Finally,a two-stage distribution method is proposed,in which the micro-clusters are merged to form the backbone of the cluster,and then the backbone of the cluster allocated in the previous step guides the distribution of the remaining points,avoiding the occurrence of chain reactions. The implementation on two dimensional composite datasets and UCI datasets shows that this algorithm has better performance in most cases than the classical density peak clustering algorithm and its improved algorithms in recent years.

Keywords　Shared nearest neighbor,Density peak clustering,Allocation strategy,Cluster center,Density decay

1　引言

聚类是将一组数据分组的过程。通过聚类算法将相似的数据划分到同一簇,将不相似的数据划分到不同的簇,从而挖掘出数据的潜在信息[1]。聚类作为一种不需要先验知识的无监督学习方法,在工业、生物、统计学、信息检索等领域都有着广泛的应用[2]。传统的聚类算法大致可以分为5种:基于划分的聚类[3-4]、基于网格的聚类[5-6]、基于密度的聚类[7-8]、基于

到稿日期:2023-05-31　返修日期:2023-10-18
基金项目:国家自然科学基金(62273164);山东省自然科学基金联合基金(ZR2020LZH009)
This work was supported by the National Natural Science Foundation of China(62273164) and Joint Fund of Natural Science Foundation of Shandong Province,China(ZR2020LZH009).
通信作者:杜韬(ise_dut@ujn.edu.cn)

层次的聚类[9]、基于模型的聚类[10-11]。基于划分的聚类算法通过不断迭代重新划分每一个数据点，从而将它们分配到更合适的簇中。*K*-Means 作为典型的基于划分的聚类算法，在球形数据集上取得了良好的效果[12]。但其具有 k 值需要人工预先设定、聚类结果受初始聚类中心的影响大、无法处理任意形状的数据、对离群点敏感这 4 个缺点。基于密度的聚类算法认为簇由相邻的高密度点组成，并被低密度点分隔开。DBSCAN 算法作为一种具有代表性的基于密度的聚类算法，可以发现任意形状的簇，并且聚类结果受噪声点和离群点的影响较小[13]。然而，DBSCAN 算法在处理数据集中密度差异很大的簇和高维数据时，得到的结果往往不尽人意。

2014 年，Rodriguez 等在 *Science* 上发表的一篇文献中提出了一种新颖的基于密度的聚类算法[14]——密度峰值聚类算法(Density Peak Clustering，DPC)。他们认为聚类中心有以下两个特点：1)聚类中心本身密度大，被密度均不超过它的邻居包围；2)聚类中心与其他密度更大的数据点之间的“距离”相对更大。基于以上两个特点，DPC 定义了局部密度和上级点距离作为数据点的相似性度量。DPC 算法的优点在于可以迅速发现聚类中心且分配过程无需迭代，近些年被广泛应用到各个领域。然而，DPC 算法仍存在一些缺陷，如 DPC 算法需要人工预先指定聚类中心的个数，而在很多应用场景下人们无法事先知道样本究竟可分为几类；DPC 算法在定义局部密度时未考虑样本的几何特征；DPC 算法的单一分配策略易产生“连锁效应”，即一个局部密度较大的样本发生分配错误，会导致以该点为上级点的所有样本点发生同样的分配错误。为了弥补 DPC 算法的上述缺陷，近年来有许多学者提出了不同的改进思路[15-17]。这些优化算法主要集中在局部密度的重新定义、聚类中心的选取和分配策略的优化 3 个方向。

针对局部密度定义所存在的问题，Du 等[18]提出了 DPC-KNN-PCA 算法，通过引入 K 近邻将局部密度的计算范围缩小，在减少计算成本的同时也充分考虑了样本分布的局部特征，并引入 PCA 降维算法改善了 DPC 在高维数据集上表现不佳的问题。Jiang 等[19]同样引入了 K 近邻思想使中心点和非中心点的差异增大，进而降低了聚类中心被错误选取的概率。Liu 等将共享最近邻的概念引入 DPC 算法中，提出了一种基于共享近邻密度峰值聚类算法(SNN-DPC)，该算法在许多数据集上都展现出优异的聚类效果，相较于之前的算法在准确率上也有较大的提升，但聚类中心的个数仍需预先设定[20]。

针对聚类中心选取所存在的问题，Chen 等[21]提出了基于自适应域密度的聚类算法(DADC)，该算法先设置一个阈值大于实际情况的初始聚类中心点，再通过一种聚类融合度模型来评估相邻簇的聚类融合度，从而将符合条件的碎片簇合并。Liu 等[22]提出了一种自适应密度峰值聚类算法(AD-PC-KNN)，引进了一种新的方法来自动选取初始聚类中心，之后，若选出的聚类中心之间是密度可达的，则将它们所在的簇融合。该算法只需调试邻居的数量 K，在一定程度上实现了自适应。Zhang 等[23]将密度衰减现象引入 DPC 算法，提出了 DGDPC 算法。该算法通过密度衰减现象将样本分成许多密度衰减图，从而避免了通过决策图选取聚类中心的问题，但该算法由于最大最近邻距离的定义问题而对离群点非常敏感，无法处理含有噪声的数据集。

针对分配策略所存在的问题，Xie 等[24]提出了 FKNN-DPC 算法，该算法使用二阶段分配方式代替 DPC 的一步分配法，先从中心点开始进行最近邻广度优先搜索来进行分配操作，再通过近邻加权技术分配离群点和第一次未完成分配的样本点。Sun 等[25]提出了 NADPC 算法，该算法先引入相互邻域概念来重新定义局部密度，再根据点的互邻度优化样本点的分配。Guo 等[26]在 2022 年提出了 DPC-CE 算法，通过图的连通性估计策略来估计局部中心之间的连通性信息，避免了“多米诺效应”。

为了解决 DPC 存在的问题或对改进方案进行结合与完善，本文提出了一种基于共享最近邻的自适应密度峰值聚类算法(Adaptive Density Peak Clustering Algorithm Based on Shared Nearest Neighbor，ADPC-SNN)。首先，引入共享邻居来重新定义样本点的相似性度量，既考虑了样本的局部信息，又缩小了计算空间。然后引入密度衰减现象来遍历样本点的 K 近邻，从而形成数量大于真实簇数量的微簇。最后提出了一种两阶段的分配方式，先通过合并相似微簇来分配一部分样本点，再用已分配的样本点指导完成剩余点的分配，得到最终的聚类结果。该算法克服了 DPC 算法聚类前需要预先设定聚类个数的缺点，也不再需要通过决策图手动选择聚类中心，实现了簇个数自适应确定和簇中心自适应选取，并且缓解了链式反应的问题。为了验证 ADPC-SNN 算法的优越性，本文在实验部分对不同的聚类算法在二维合成数据集和真实数据集上(UCI 数据库)得到的聚类结果进行对比分析。实验结果表明，在大部分情况下，ADPC-SNN 算法的聚类结果优于 *K*-Means，DPC 等经典聚类算法以及 FKNN-DPC，SNN-DPC，DGDPC 等近年来对 DPC 进行改进的算法。

2 相关工作

本章简要介绍 DPC 算法的基本概念，然后对其缺点和缺点产生的原因进行分析，最后介绍共享最近邻思想。

2.1 密度峰值聚类算法(DPC)

2.1.1 DPC 算法的基本思想

密度峰值聚类算法的核心思想是：1)聚类中心点的局部密度大于其周围点的局部密度；2)不同聚类中心点之间的距离相对较远。基于以上两种思想，DPC 定义了局部密度 ρ 和上级点距离 δ 作为相似性度量。假设存在数据集 $X=\{x_1, x_2, x_3, \cdots, x_n\}$，样本点 x_i 的局部密度 ρ_i 有式(1)中的截断距离方法和式(2)中的高斯核函数方法这两种定义方式：

$$\rho_i=\sum_{j=1}^{n}\chi(d_{ij}-d_c),\chi=\begin{cases}1, & x\leqslant 0\\ 0, & \text{otherwise}\end{cases} \tag{1}$$

$$\rho_i=\sum_{j=1,j\neq i}^{n}\exp\left[-\left(\frac{d_{ij}}{d_c}\right)^2\right] \tag{2}$$

其中，d_{ij} 表示数据点 x_i 和 x_j 之间的欧氏距离；d_c 为截断距离，需要人工预先设定。对于大数据集，DPC 算法一般使用截断距离方法来计算局部密度，对于小数据集则使用高斯核函数方法来计算局部密度。

对于数据集中的任意一个样本点 x_i，DPC 算法在局部密度大于 x_i 的点中选取一个距离样本 x_i 最近的点定义为 x_i 的上级点，其上级点距离 δ_i 定义为 x_i 与其上级点之间的欧氏距离，使用式(3)来进行计算：

$$\delta_i=\begin{cases}\min(d_{ij}), & \text{if } \rho_i\neq\rho_{\max}\ \&\ \rho_i<\rho_j\\ \max(d_{ij}), & \text{otherwise}\end{cases} \tag{3}$$

DPC 算法以局部密度 ρ 为横坐标、上级点距离 δ 为纵坐标画出决策图来辅助选取聚类中心。根据 DPC 的两种核心思想，同时具有高 ρ 值和高 δ 值的样本点被人工确定为聚类中心，一般分布于决策图的右上方。当出现通过决策图无法判断出聚类中心的情况时，DPC 算法会计算决策值 γ，选择 γ 值大的点作为聚类中心。任意样本点 x_i 的决策值 γ_i 的定义如式(4)所示：

$$\gamma_i=\rho_i\times\delta_i \tag{4}$$

在通过决策图选取出聚类中心后，DPC 采用一步分配的方式，将剩余点直接分配给其上级点所属的簇中，若其上级点也尚未分配，则依次向上查找，从而得到最终的聚类结果。

2.1.2 DPC 算法的缺点

DPC 算法对密度分布不均匀的数据集的聚类效果较差，并且其通过决策图人工选取聚类中心的方法使 DPC 算法在很多情况下难以选出正确的聚类中心。如图 1 所示，在 Jain 数据集中，我们可以很明显地分辨出该数据集有两个簇：左上角的倒 U 型为一簇，右下角的 U 型为另一簇。然而无论 DPC 算法局部密度选择哪一种算式，都会选取到错误的聚类中心。为了探究产生错误的原因，我们分别做出决策图和 γ-决策图，可以发现无论在哪个图中，都会有两个样本点因较高的局部密度 ρ 和较大的上级点距离 δ 而与其他样本点产生明显的间隔。而这两个点都是右下高密度簇中的样本点，左上低密度簇的真实聚类中心在决策图中和剩余点混在一起，造成了聚类中心的错误选取。事实上，只要数据集中存在密度分布不均匀的簇，即使预先设定好聚类中心的个数，DPC 算法也很难通过决策图找出正确的聚类中心。

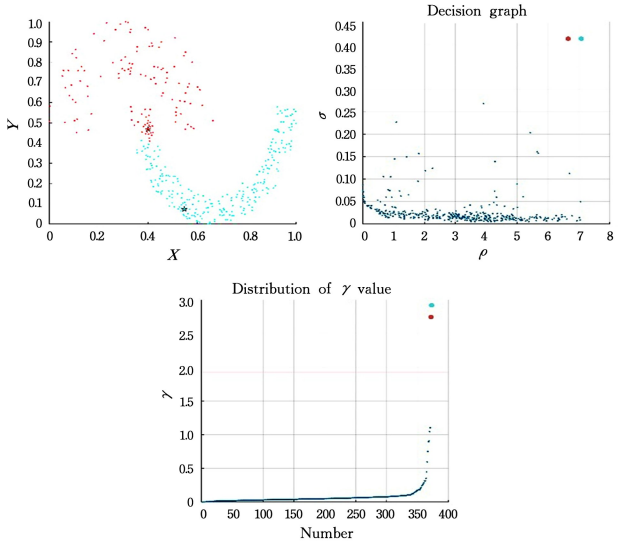


图 1 Jain 数据集决策图及聚类结果

Fig. 1 Decision graph and clustering results of Jain dataset

DPC 算法的另一个明显的缺陷是分配过程中的连锁反应问题。以 PathBased 数据集举例，如图 2(a)所示，我们能很容易地分辨出该数据集有 3 个簇：外面的圆环一簇，圆环里面包着两个小团簇。在 PathBased 数据集上，DPC 算法正确地选出了聚类中心，但在后续的分配环节出现了错误：环两侧的样本点被错误分配到中间两个团簇中。造成这种现象的原因是在圆环左右两侧，密度最大的样本点的上级点是内部团簇中的点，导致该点被错误分配，然后以该点为上级点的所有点都跟着被错误分配到了内部的簇中，从而产生了灾难性的聚类结果。由此可见，DPC 算法的一步分配策略虽然简单高效，但也存在着非常大的隐患。

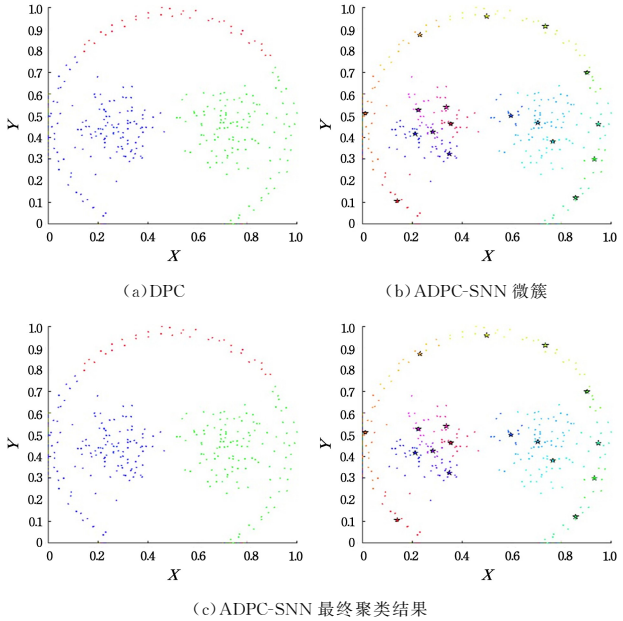


图 2 DPC 在 PathBased 数据集上的聚类结果

Fig. 2 Clustering results of DPC on PathBased dataset

2.2 共享最近邻

共享最近邻(Shared Nearest Neighbor，SNN)是基于 K 近邻算法的改进。K 邻近(KNN)的思想是：如果在特征空间中，距离一个样本点最近的 K 个样本点大多属于同一类别，则该样本点也属于这一类别。SNN 的基本思想是，如果两个点有更多共同的邻居，则认为它们更相似。

定义 1(共享邻居集) 对于数据集 X 中的任意点 i 和 j，其共享邻居集定义如下：

$$SNN(i,j)=KNN(i)_k\cap KNN(j)_k \tag{5}$$

3 基于共享最近邻的自适应密度峰值聚类算法

为弥补 DPC 算法及其优化算法的缺陷，本文对样本点的度量值和剩余点的分配方式进行改进，提出了基于共享最近邻的自适应密度峰值聚类算法，该算法在弥补上述缺陷的同时，摆脱了 DPC 算法对预设聚类个数和人工选取聚类中心的依赖，实现了簇个数自适应确定和簇中心自适应选取。

3.1 基于共享最近邻定义点的度量值

DPC 算法在一些复杂数据集上可能不会产生令人满意的结果，因为 DPC 算法是直接计算点之间的距离和密度。

然而，一个点的大多数邻居通常仍然属于同一个簇，这一事实可用于改进 DPC 算法的度量值，我们用共享最近邻算法重新定义了 DPC 算法的度量值。

定义 2（相似度） 基于两个点有更多共同的邻居则认为它们更相似的思想，我们定义了相似度，对于数据集 X 中的任意点 i 和 j，其相似度定义如下：

$$Sim(i,j)=\begin{cases}\dfrac{|SNN(i,j)|^2}{\sum\limits_{p\in SNN(i,j)}(d_{ip}+d_{jp})}, & \text{if } i,j\in SNN(i,j)\\ 0, & \text{otherwise}\end{cases} \tag{6}$$

其中，d_{ij} 是点 i 和 j 的欧氏距离，只有当点 i 和点 j 出现在彼此的 K 近邻中时才会计算相似度，否则点 i 和点 j 之间的相似度为 0。不难发现，两个点的共享邻居数量越多，两个点的相似度越高。两个点的共享邻居距离这两个点越近，两个点的相似度越大。通过同时检查两个点的共享邻居和周围邻域的密度，SNN 相似度可以更好地适应各种环境。

定义 3（局部密度） 对于数据集 X 中的任意点 i，$L(i)=\{x_1,x_2,\cdots,x_k\}$ 为与样本点 i 相似度最高的 k 个样本点的集合。我们把与点 i 相似度最高的 k 个点的相似度之和定义为点 i 的局部密度。

$$\rho_i=\sum_{j\in L(i)}Sim(i,j) \tag{7}$$

定义 4（上级点距离） 设点 i 是数据集 X 中的任意样本点，点 j 为满足局部密度大于点 i 的样本点，计算 i 和 j 的距离乘以两点分别到各自 K 近邻的距离之和，取其中最小值作为点 i 的上级点距离，其对应的样本点即为点 i 的上级点，公式定义如下：

$$\delta_i=\min_{j:\rho_j>\rho_i}\left[d_{ij}\left(\sum_{p\in KNN(i)}d_{ip}+\sum_{q\in KNN(j)}d_{jq}\right)\right] \tag{8}$$

其中，局部密度最高的样本点的 δ 值单独定义为其他样本点中最大的 δ 值：

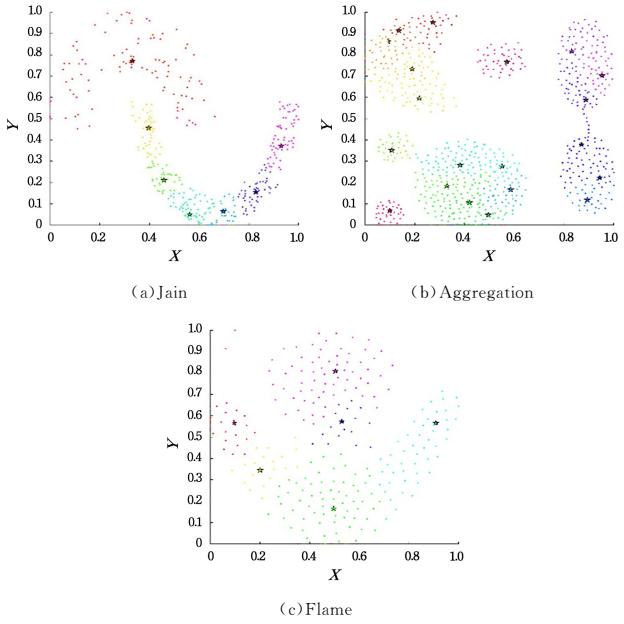
$$\delta_i=\max_{j\in(X-i)}(\delta_j) \tag{9}$$

如果只看式(8)的前半部分，则上级点距离定义和 DPC 算法一致，更新后的上级点距离因子不仅取决于两个点之间的距离，还考虑了两个点各自的邻域信息。

3.2 基于密度衰减形成微簇实现聚类中心自适应选取

如果一组事物有一定的数量，并且该事物随着其与中心的距离的增大而逐渐减少，那么这些事物应该被视为一个整体，这种现象被称为衰减现象。本文将自然中的衰减现象和聚类算法相结合，定义了密度衰减点和密度衰减集。

定义 5（密度衰减点和密度衰减集） 如果点 P_i 和点 P_j 满足：存在一个路径 $P_1=P_i,\cdots,P_n=P_j$，如果每一个 P_k $(1\leqslant k\leqslant n)$ 都满足 $\rho_k>\rho_{(k+1)}$，且 $P_{(k+1)}$ 是 P_k 的 K 近邻，则称点 P_j 是点 P_i 的密度衰减点。点 P_i 的所有密度衰减点的集合被定义为点 P_i 的密度衰减集。可以将已经形成的密度衰减集看成一个个自下而上聚集而成的微簇。图 3 为 ADPC-SNN 算法分别在 Jain，Aggregation，Flame 这 3 个数据集上产生的微簇结果。如图 3 所示，ADPC-SNN 算法产生的微簇个数一般要大于最终结果的簇个数。



(a) Jain (b) Aggregation

(c) Flame

图 3 ADPC-SNN 算法在 3 个数据集上产生的微簇结果

Fig. 3 Microcluster results generated by ADPC-SNN algorithm on three datasets

3.3 基于微簇合并的二阶段分配法

由于 DPC 的一步分配方式容易产生一步错步步错的链式反应，因此本文提出了一种两阶段分配策略以代替一阶段分配策略。第一阶段将微簇合并形成簇主干，并将簇主干上的样本点分配给对应的簇，第二阶段对未分配的剩余点采用和传统 DPC 一样的分配方式，即将剩余点分配给对应的上级点。由于第一步已经将一部分点分配完毕，这些点在第二步分配过程中会起到指导作用，从而有效避免“多米诺效应”。下面将对微簇的合并过程以及相关定义进行详细介绍。

定义 6（交点集） 对于任意两个微簇 C_u 和 C_n，其交点集 I_n^u 的定义如下：

$$I_n^u=\{i\mid i\in C_u\bigcap C_n\} \tag{10}$$

定义 7（连接点） 对于任意两个微簇 C_u 和 C_n，如果点 i 是交点集 I_n^u 中的点且满足以下条件，则称点 i 为微簇 C_u 和 C_n 的连接点：

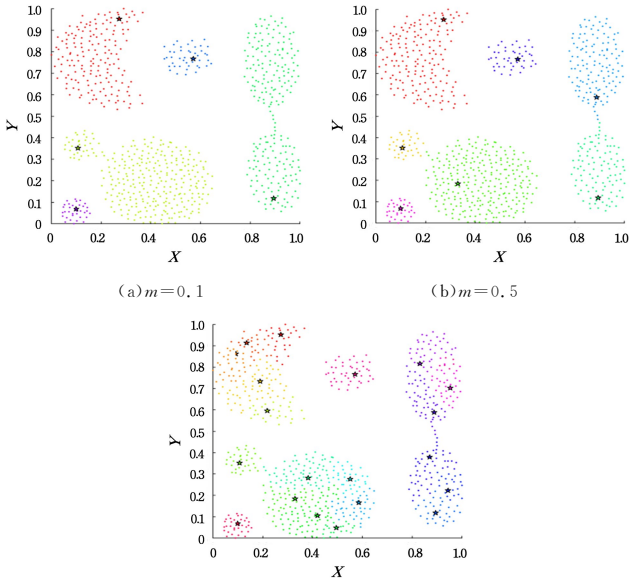
1) C_u 中至少存在 $m\cdot|C_u|$ 个密度小于点 i 的点。

2) C_n 中至少存在 $m\cdot|C_n|$ 个密度小于点 i 的点。

其中 m 为合并阈值，取值范围为 0～1。图 4 为 ADPC-SNN 算法设置不同的 m 值在 Aggregation 数据集上得到的聚类结果图。如图所示，m 越大微簇越难合并，最终聚类结果中簇个数就越多；m 越小越容易合并，最终聚类结果中簇个数就越少。算法将存在连接点的两个微簇进行合并，合并完成后将未满足合并条件的交集点的标签清空，使其不属于任何一个簇，至此第一步分配结束。已被分配的点的分配结果将作为簇主干指导第二步分配。第二步的分配方式和 DPC 相同，由于已经形成簇的雏形，剩余点不需查找多次即可找到其归属簇，这在一定程度上减弱了“多米诺效应”的影响。

如图 2(b)所示，ADPC-SNN 算法在 PathBased 数据集上自动汇聚成了 18 个微簇，经过两步分配策略后，18 个微簇合

并为了图 2(c)中的 3 个簇，与图 2(a)中 DPC 的聚类结果相比可以看到，ADPC-SNN 算法将 DPC 算法没有正确分配的点分配到了正确的簇中。



(a)$m=0.1$ (b)$m=0.5$ (c)$m=1$

图 4 ADPC-SNN 算法设置不同的 m 值时在 Aggregation 数据集上的聚类结果

Fig. 4 Clustering results of ADPC-SNN algorithm on Aggregation dataset with different m values

3.4 算法步骤

ADPC-SNN 算法的具体步骤如算法 1 所示。

算法 1 ADPC-SNN 算法

输入：数据集 X，合并阈值 m，样本近邻数 k

输出：最终的聚类结果

1. 对数据进行预处理并归一化
2. 计算样本点之间的欧氏距离
3. 根据式(5)和式(6)计算共享邻居集和相似度
4. 利用式(7)计算样本点的局部密度，利用式(8)和式(9)计算样本点的距离
5. 从数据集 X 的尚未分配的点中选取密度最大的点，并生成这个点的密度衰减集，将该点的密度衰减集中所有的点分配给该点，形成一个新的微簇
6. 重复 Step5 直到所有的样本点都分配完毕
7. 将存在连接点的微簇进行合并
8. 将交点集中剩余的所有样本点设置为不属于任何一个簇，然后根据其上级点重新分配这些点
9. 撤销点数小于数据集总数 1%的簇，并将其中的点根据其上级点重新分配
10. 输出最终聚类结果

3.5 算法时间复杂度分析

设置数据集的样本规模为 n，近邻个数为 k，微簇个数为 c。算法的各个部分的时间复杂度分析如下：

1)计算样本点的欧氏距离，时间复杂度为 $O(n^2)$。

2)计算样本点的相似度，时间复杂度为 $O(n^2)$。

3)计算样本点的局部密度，时间复杂度为 $O(n^2)$。

4)计算样本点的上级点距离，时间复杂度为 $O(kn)$。

5)计算密度衰减集并形成微簇，时间复杂度为 $O(n^2)$。

6)将每个微簇中的点按局部密度从小到大排序，时间复杂度为 $O(n^2)$。

7)找出连接点，时间复杂度为 $O(n)$。

8)将存在连接点的微簇合并，时间复杂度为 $O(c^2)$。

9)将交点集中的剩余点和样本点过少的点按上级点重新分配，时间复杂度为 $O(cn)$。

由于 $c \ll n$，综上所述，算法的时间复杂度为 $O(n^2)$，与 DPC 算法的时间复杂度量级相同。本文在 Jain 数据集上进行了实际耗时实验，DPC 算法耗时 0.299 s，ADPC-SNN 算法耗时 0.311 s，符合时间复杂度预期。

4 实验结果与分析

4.1 实验环境与数据

本文选用 8 个人工数据集和 8 个 UCI 数据集对本文提出的 ADPC-SNN 算法进行性能测试，进而验证算法的有效性。表 1 和表 2 列出了这些数据集的详细信息。本文采用 K-Means 算法[12]、DPC 算法[14]、FKNN-DPC 算法[24]、SNN-DPC 算法[20]、DGDPC 算法[23]进行对比实验。实验环境为 AMD Ryzen 7 5800H with Radeon Graphics CPU@ 3.20 GHz，16 GB RAM，Windows 64bit 操作系统，使用 Matlab R2017a 软件进行编译。

表 1 人工数据集

Table 1 Synthetic datasets

数据集名称	数据规模	属性数	簇数
Aggregation	788	2	7
Jain	373	2	2
Flame	240	2	2
S2	5000	2	15
A3	7500	2	50
R15	600	2	15
Compound	399	2	6
Pathbased	300	2	3

表 2 UCI 数据集

Table 2 UCI datasets

数据集名称	数据规模	属性数	簇数
Iris	150	4	3
E-coli	336	7	8
Libras movement	360	90	15
Ionosphere	351	33	2
Dermatology	366	33	6
Segmentation	2310	19	7
Statlog(Heart)	270	13	2
Glass	214	9	6

4.2 评价指标

评价指标可以有效反映出聚类算法的性能优劣。本文采用调整互信息(Adjusted Mutual Information，AMI)、调整兰德系数(Adjusted Rand Index，ARI)和 Fowlkes-Mallows 指数(Fowlkes-Mallows index，FMI)3 个评价指标进行对比，其中 AMI 和 FMI 的取值范围为[0，1]，ARI 的取值范围为[−1，1]。

4.3 参数设置

为了更客观地反映各种算法的实际效果，我们对每种

算法进行参数调整，从而确保能反映出算法的最佳性能。DPC 算法的作者提供了一条经验法则，即 dc 的值在总样本数的 1%～2%时算法效果最佳，我们将范围扩大至 0%～10%，步长为 0.1%，以求获取算法的最佳结果。DGDPC 算法的 dc 取值范围也设置为 0%～10%，步长同样设置为 0.1%，m 的取值范围在 0～1 之间，步长为 0.1。对于 ADPC-SNN 算法、SNN-DPC 算法和 FKNN-DPC 算法，由于这 3 个算法都有邻居个数这一参数，因此我们将邻居个数取值范围统一设置为[4,100]。在大多数数据集中，ADPC-SNN 算法邻居数 K 的最优参数在数据总数的 1%～4%之间。ADPC-SNN 算法的合并阈值取值范围设置为[0,1]，步长为 0.1。K-Means 算法需要簇的个数 K，我们将其设置为簇的实际数量。

4.4 人工数据集实验结果分析

表 3 列出了 6 种不同的聚类算法在 8 个人工数据集上的评价指标。表中'Arg-'表示对应算法在该数据集上的最优参数，每个数据集上的最优结果用加粗字体表示。由表 3 可得，ADPC-SNN 算法在 8 个人工数据集上都取得了最优结果，其中在 Aggregation，S2，A3，Compound，Pathbased 这 5 个数据集上，ADPC-SNN 算法的 3 个评价指标均领先其他算法。其次是 DGDPC 算法，其在 3 个数据集上得到了最优结果，DPC 和 SNN-DPC 算法分别在两个数据集上表现最优，表现最差的是 FKNN-DPC 和 K-Means 算法，仅在一个数据集上最优。

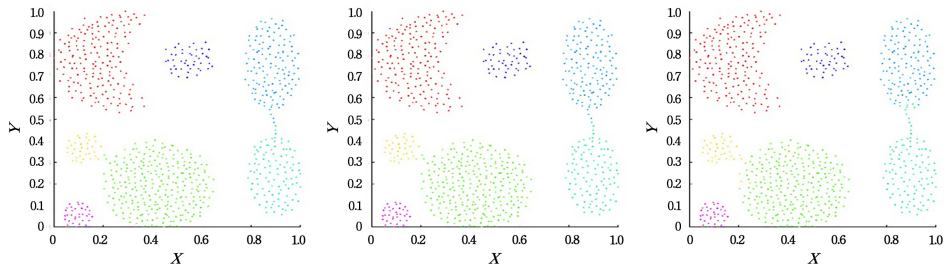
表 3 6 种算法在 8 个人工数据集上的聚类结果

Table 3 Clustering results of six algorithms on eight synthetic datasets

Algorithm	Aggregation				Jain				Flame				S2			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
ADPC-SNN	**0.9955**	**0.9978**	**0.9983**	20/0.5	**1**	**1**	**1**	12/0.1	**1**	**1**	**1**	17/0.5	0.9435	0.9345	0.9388	85/1
DGDPC	0.9922	0.9956	0.9966	5/0.5	**1**	**1**	**1**	0.1/5	**1**	**1**	**1**	0.5/5	**0.9485**	**0.9399**	**0.9439**	1/1
DPC	0.9921	0.9956	0.9966	4	0.6183	0.7146	0.8819	0.9	**1**	**1**	**1**	2.8	0.9437	0.9352	0.9395	1.5
SNN-DPC	0.9500	0.9594	0.9681	15	**1**	**1**	**1**	12	0.8975	0.9502	0.9768	5	0.9386	0.9264	0.9313	35
FKNN-DPC	0.9775	0.9855	0.9886	20	0.0562	0.1318	0.6430	10	**1**	**1**	**1**	6	0.9180	0.8889	0.8963	22
K-Means	0.7935	0.7300	0.7884	7	0.4916	0.5767	0.8200	2	0.3863	0.4534	0.7364	2	0.9461	0.9379	0.9420	15
Algorithm	A3				R15				Compound				Pathbased			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
ADPC-SNN	**0.9904**	**0.9862**	**0.9865**	89/1	**0.9938**	**0.9928**	**0.9933**	20/0.8	**0.9282**	**0.9683**	**0.9761**	8/0.5	**0.9301**	**0.9493**	**0.9662**	10/0.3
DGDPC	0.9869	0.9798	0.9802	1/1	**0.9938**	**0.9928**	**0.9933**	0.5/5	0.7871	0.8078	0.8649	0.8/3	0.6814	0.7291	0.8143	0.3/4
DPC	0.9869	0.9802	0.9806	0.6	**0.9938**	**0.9928**	**0.9933**	0.6	0.7754	0.5910	0.6876	3.8	0.5212	0.4717	0.6664	3.8
SNN-DPC	0.9890	0.9826	0.9829	43	**0.9938**	**0.9928**	**0.9933**	10	0.8434	0.8407	0.8798	4	0.9001	0.9294	0.9529	9
FKNN-DPC	0.9774	0.9627	0.9634	22	0.9907	0.9892	0.9899	27	0.8052	0.8526	0.8895	8	0.8344	0.8744	0.9165	9
K-Means	0.9378	0.8371	0.8425	50	**0.9938**	**0.9928**	**0.9933**	15	0.7402	0.6115	0.7018	6	0.5098	0.4613	0.6617	3

为了更直观地对比不同算法在数据集上的聚类情况，本文将聚类结果可视化，并与 DPC 算法和 SNN-DPC 算法进行对比。图 5—图 12 为这 3 种算法在这 8 个人工数据集上的聚类效果图。Aggregation 数据集有 7 个簇，其中左侧两个簇中间是相连的。如图 5 所示，3 种算法都将绝大多数的样本点分配正确，SNN-DPC 算法把左侧相连部分的点全都分配给了下方的簇。相较于 SNN-DPC 算法，ADPC-SNN 和 DPC 算法在连接部分明显处理得更好。Jain 数据集由两个密度相差较大的 U 形簇组成，如图 6 所示，ADPC-SNN 和 SNN-DPC 算法都可以将全部点正确分配，DPC 算法在密度差异大的数据集上的表现不是很理想，其将下侧簇的部分点错误分配给了上侧簇。Flame 数据集样本点分布均匀，分为上下两簇，中间有小部分连接，ADPC-SNN 和 DPC 算法都可以将这些连接处的点正确分配，SNN-DPC 存在小部分的分配错误。S2 和 A3 均为规模较大的数据集，分别有 5000 个样本点和 7500 个样本点，由图 8 和图 9 可得，3 种算法针对大规模数据集都有不错的聚类效果。图 10 为 R15 数据集的聚类结果，3 种算法均可准确地完成聚类。Compound 数据集有 6 个簇，每个簇都形状各异，如图 11 所示，DPC 算法将左下角的圆环簇和团簇上下聚类为 3 个簇，将右侧两个簇合并为一个簇，聚类效果最不理想。SNN-DPC 算法没有正确地将右侧的散点簇和密集簇分开，而是将这些点错误分成了上下两个簇。ADPC-SNN 算法将密度最小的散点簇分为了上下两个簇，剩下的 5 个簇的样本点均分配至正确的簇中。ADPC-SNN 算法不用事先指定聚类个数，因此相较于 DPC 和 SNN-DPC 算法，虽然其将数据集多分出来一个簇，但却取得了明显更优的聚类效果。

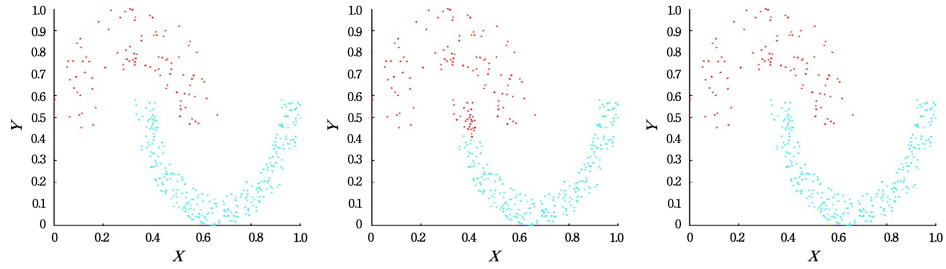
Pathbased 数据集的聚类结果如图 12 所示，ADPC-SNN 和 SNN-DPC 算法都能较好地完成聚类，而 DPC 算法将环簇两侧的点分配给了中间两个团簇。



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 5 3 种算法在 Aggregation 数据集上的聚类结果

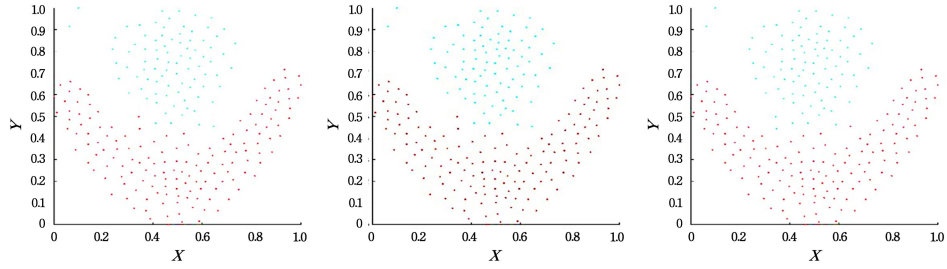
Fig. 5 Clustering results of three algorithms on Aggregation dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 6 3 种算法在 Jain 数据集上的聚类结果

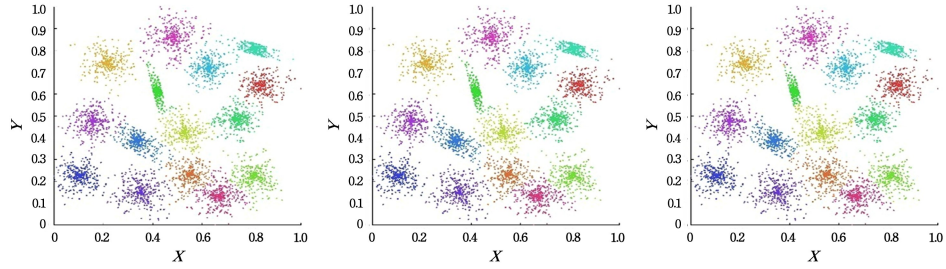
Fig. 6 Clustering results of three algorithms on Jain dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 7 3 种算法在 Flame 数据集上的聚类结果

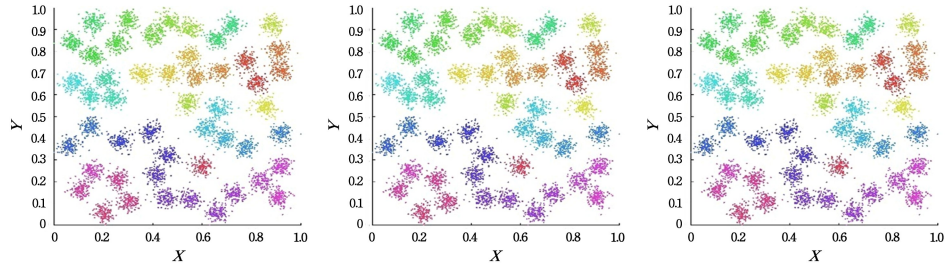
Fig. 7 Clustering results of three algorithms on Flame dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 8 3 种算法在 S2 数据集上的聚类结果

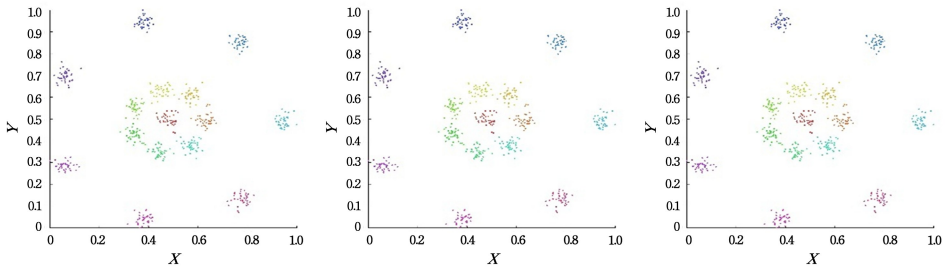
Fig. 8 Clustering results of three algorithms on S2 dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 9 3 种算法在 A3 数据集上的聚类结果

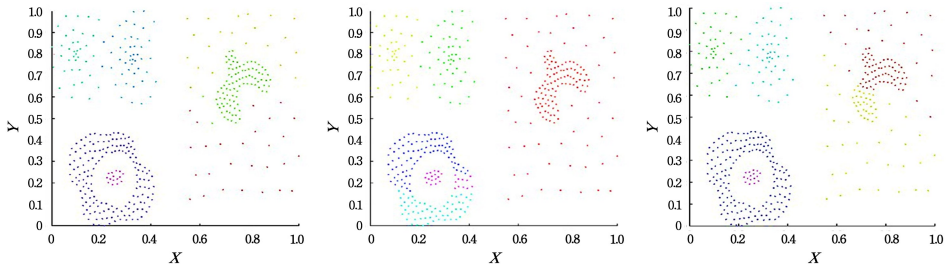
Fig. 9 Clustering results of three algorithms on A3 dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 10 3 种算法在 R15 数据集上的聚类结果

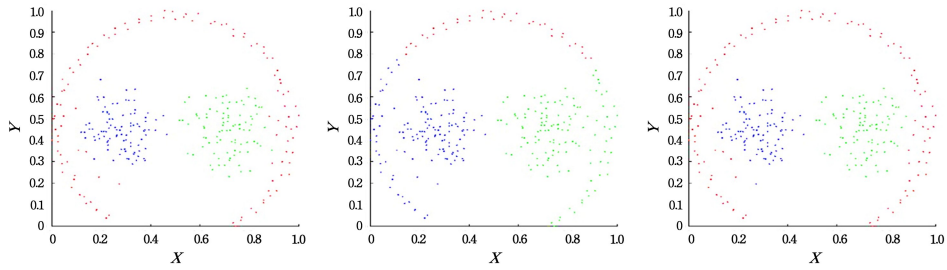
Fig. 10 Clustering results of three algorithms on R15 dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 11 3 种算法在 Compound 数据集上的聚类结果

Fig. 11 Clustering results of three algorithms on Compound dataset



(a) ADPC-SNN (b) DPC (c) SNN-DPC

图 12 3 种算法在 Pathbased 数据集上的聚类结果

Fig. 12 Clustering results of three algorithms on Pathbased dataset

上述实验结果表明 ADPC-SNN 算法在聚类中心个数需要预先设定和聚类中心需要人工选取这两个前提下，在不同形状、不同规模和不同簇数的数据集中都取得了较为优秀的聚类结果。

4.5 真实数据集实验结果分析

为了进一步测试 ADPC-SNN 算法的聚类性能，本文选取了 UCI 中的 8 个真实数据集进行对比实验，实验结果如表 4 所列，相对较好的实验结果用加粗字体表示。由表 4 可得，ADPC-SNN 算法在 E-coli, Libras movement, Ionosphere, Statlog(Heart)这 4 个数据集上的 AMI, ARI 和 FMI 3 个评价指标相较于其他 5 种算法有较为明显的优势。其次是 SNN-DPC 算法，其在 Iris 和 Dermatology 两个数据集上获得了最优结果。在 Segmentation 数据集上，ADPC-SNN 算法在 ARI 和 FMI 两个指标取得了最高值，在 AMI 上仅低于 DPC 算法。在 Glass 数据集上，SNN-DPC, DPC, ADPC-SNN 分别在 AMI, ARI, FMI 上取得了最高值。上述实验结果表明，ADPC-SNN 算法在不同形状、规模和簇数的真实数据集中也有着良好的表现。

表 4 6 种算法在 8 个 UCI 数据集上的聚类结果

Table 4 Clustering results of six algorithms on eight UCI datasets

Algorithm	Iris				E-coli				Libras movement				Ionosphere			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
ADPC-SNN	0.8264	0.8771	0.9170	12/0.7	**0.6799**	**0.7816**	**0.8450**	15/0.9	**0.5927**	**0.4223**	**0.4806**	13/0.9	**0.4526**	**0.5958**	**0.8137**	14/0.3
DGDPC	0.7427	0.6844	0.7946	1/3	0.0556	0.0380	0.5310	0.9/2	0.2796	0.1323	0.2826	1/1	0	0	0.7338	1/1
DPC	0.8606	0.8857	0.9233	0.2	0.5179	0.4365	0.5693	0.2	0.5358	0.3193	0.3717	0.3	0.1504	0.2357	0.6491	0.5
SNN-DPC	**0.9124**	**0.9222**	**0.9479**	15	0.6711	0.7547	0.8243	6	0.5834	0.3927	0.4507	11	0.3644	0.4949	0.7798	5
FKNN-DPC	0.8831	0.9038	0.9355	22	0.4755	0.5535	0.6919	9	0.4754	0.3184	0.3976	11	0.1314	0.1321	0.5841	26
K-Means	0.7331	0.7163	0.8112	3	0.5051	0.4190	0.5542	8	0.5232	0.3094	0.3612	15	0.1294	0.1776	0.6053	2
Algorithm	Dermatology				Segmentation				Statlog(Heart)				Glass			
	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-	AMI	ARI	FMI	Arg-
ADPC-SNN	0.8725	0.8629	0.8917	5/0.4	0.6839	**0.6626**	**0.7091**	**90/0.9**	**0.3042**	**0.3939**	**0.7079**	31/0.7	0.5168	0.6128	**0.7368**	30/0.9
DGDPC	0.5297	0.5019	0.6646	1/0.04	0.3994	0.2583	0.5005	1/0.5	0.2891	0.3753	0.7002	1/0.08	0	0	**0.5097**	1/1
DPC	0.7840	0.7760	0.8221	1.6	**0.6927**	0.6004	0.6730	1.5	0.2878	0.3753	0.6990	0.1	0.5641	**0.6248**	0.7180	1.6
SNN-DPC	**0.8749**	**0.8689**	**0.9021**	19	0.6725	0.5770	0.6457	7	0.2400	0.2660	0.6397	10	**0.5780**	0.6052	0.7065	14
FKNN-DPC	0.8355	0.8127	0.8504	35	0.5830	0.4367	0.5581	27	0.0458	0.0324	0.5499	8	0.1755	0.1879	0.5506	18
K-Means	0.8748	0.7426	0.7947	6	0.6102	0.5049	0.5758	7	0.2570	0.3314	0.6666	2	0.5598	0.5635	0.6726	6

结束语 本文提出了一种基于共享最近邻的自适应密度峰值聚类算法。该算法通过引入共享最近邻解决了 DPC 算法未考虑样本局部特征的问题；通过引入密度衰减现象形成微簇，避免了聚类中心个数需预先指定和聚类中心的人工选取，减少了算法的人为主观性；通过提出一种二阶段分配法，用微簇合并形成的簇主干来指导剩余点的分配，避免了样本点分配时产生“多米诺效应”。实验结果表明，ADPC-SNN 算法可以在实现簇个数自适应确定和簇中心自适应选取的同时，使聚类效果得到大幅提升。下一步的工作重点是将该算法应用到高维数据以及流数据环境中，以便在生产环境中

进行数据分析，解决实际问题，并提高相关领域的生产效率。

参 考 文 献

[1] SUN L，QIN X Y，XU J C，et al. Density Peak Clustering Algorithm Based on K-Nearest Neighbors and Optimal Assignment Strategy[J]. Journal of Software，2022，33(4)：1390-141.

[2] CERQUITELLI T，VENTURA F，APILETTI D，et al. Enhancing manufacturing intelligence through an unsupervised data-driven methodology for cyclic industrial processes[J]. Expert Systems with Applications，2021，182(3)：115269.

[3] YANG L，CHEUNG Y M，YUAN Y T. Self-Adaptive Multiprototype-Based Competitive Learning Approach：Ak-Means-Type Algorithm for Imbalanced Data Clustering[J]. IEEE Transactions on Cybernetics，2019，51(3)：1598-1612.

[4] AHMAD A，KHAN S S. initKmix—A novel initial partition generation algorithm for clustering mixed data using k-means-based clustering[J]. Expert Systems with Applications，2020，167(2)：114149.

[5] JIANG J T，ZHENG C H. Density Peak and Grid Based Clustering for Large-scale Node Partition[J]. Journal of Chinese Mini-Micro Computer Systems，2022，43(3)：498-505.

[6] SUN L，LIANG Y Q. Improved Clustering Algorithm Fusing Grid Partition and DBSCAN[J]. Computer Engineering and Applications，2022，58(14)：73-79.

[7] HU C A，WANG J X，MAO Y M. Density-based clustering algorithm based on groups and improve gravitational search[J]. Application Research of Computers，2021，38(11)：3293-3299.

[8] GUO W J，WANG W H，ZHAO S P，et al. Density Peak Clustering with connectivity estimation[J]. Knowledge-Based Systems，2022，243(5)：108501.

[9] ZHANG T，RAMAKRISHNAN R，LIVNY M. BIRCH：an efficient data clustering method for very large databases[J]. ACM Sigmod Record，1996，25(2)：103-114.

[10] WANG R，ZHOU J，JIANG H，et al. A general transfer learning-based Gaussian mixture modelfor clustering[J]. International Journal of Fuzzy Systems，2021，23(3)：776-793.

[11] LI K，ZHANG K X. Structural α-Entropy Weighting Gaussian Mixture Model for Subspace Clustering[J]. Chinese Journal of Electronics，2022，50(3)：718-725.

[12] HARTIGAN J A，WONG M A. Algorithm AS 136：A K-Means Clustering Algorithm[J]. Journal of the Royal Statistical Society，1979，28(1)：100-108.

[13] ESTER M，KRIEGEL H P，SANDER J，et al. A density-based algorithm for discovering clusters in large spatial databases with noise[C]// Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. 1996：226-231.

[14] RODRIGUEZ A，LAIO A. Clustering by fast search and find of density peaks[J]. Science，2014，344(6191)：1492-1496.

[15] WEI L，GAO L，LI J H，et al. Traffic sub-area division method based on density peak clustering[J]. Journal of Jilin University (Engineering and Technology Edition)，2023，53(1)：124-131.

[16] WANG F Y，ZHANG D S，XIAO Y T. Density Peak Algorithm Based on Weighted Shared Nearest Neighbor and Accumulated Sequence[J]. Computer Engineering，2022，48(4)：61-69.

[17] ZHANG X Y，YUN W G. Sharing K-nearest Neighbors and Multiple Assignment PoliciesDensity Peaks Clustering Algorithm[J]. Journal of Chinese Computer Systems，2023，44(1)：75-82.

[18] DU M，DING S，JIA H. Study on density peaks clustering based on k-nearest neighbors and principal component analysis[J]. Knowledge-Based Systems，2016，99(5)：135-145.

[19] JIANG J，CHEN Y，MENG X，et al. A novel density peaks clustering algorithm based on k nearest neighbors for improving assignment process[J]. Physica A：Statistical Mechanics and Its Applications，2019，523(6)：702-713.

[20] LIU R，WANG H，YU X. Shared-nearest-neighbor-based clustering by fast search and find of density peaks[J]. Information Sciences，2018，450：200-226.

[21] CHEN J，YU P. A domain adaptive density clustering algorithm for data with varying density distribution[J]. IEEE Transactions on Knowledge and Data Engineering，2021，33(6)：2310-2321.

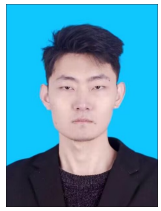
[22] LIU Y H，MA Z M，YU F. Adaptive density peak clustering based on K-nearest neighbors with aggregating strategy[J]. Knowledge-Based Systems，2017，133(10)：208-220.

[23] ZHANG Z，ZHU Q，ZHU F，et al. Density decay graph-based density peak clustering[J]. Knowledge-Based Systems，2021，224：107075.

[24] XIE J，GAO H，XIE W，et al. Robust clustering by detecting density peaks and assigning points based on fuzzy weighted K-nearest neighbors[J]. Information Sciences，2016，354(8)：19-40.

[25] SUN L，QIN X，DING W，et al. Nearest neighbors-based adaptive density peaks clustering with optimized allocation strategy[J]. Neurocomputing，2022，473：159-181.

[26] GUO W，WANG W，ZHAO S，et al. Density peak clustering with connectivity estimation[J]. Knowledge-Based Systems，2022，243：108501.



WANG Xingeng，born in 1999，postgraduate. His main research interests include data clustering and data mining.

DU Tao，born in 1979，Ph.D，associate professor. His main research interests include data clustering and data mining.

（责任编辑：何杨）