文章编号：2095-2163(2021)11-0194-04 中图分类号：TP301.6 文献标志码：A

基于 KD 树的 k-means 聚类算法优化

薛丁文，李建中
（哈尔滨工业大学 计算机科学与技术学院，哈尔滨 150001）

摘 要：作为模式识别最基本的分类方法之一，聚类在各个科学领域的数据分析中都扮演着重要的角色。然而随着大数据的出现，聚类分析在前沿发展中不断地面临着计算复杂度和计算成本等新的问题和挑战。通过研究 k-means 聚类算法的时间复杂度 $O(nk)$，针对迭代过程中大量的最近邻计算和其特殊场景，引入 KD 树作为索引，提出了基于单 KD 树的近似近邻算法和基于多 KD 树的交叉搜索算法。将 k-means 聚类算法的时间复杂度降为 $O(n\log k)$，并通过实验验证，基于多树的交叉搜索算法具有与 k-means 聚类算法相当的聚类质量。

关键词：聚类分析；k-means 聚类；KD 树；近似近邻

Optimization of k-means clustering algorithm based on KD-tree

XUE Dingwen, LI Jianzhong
(Department of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China)

【Abstract】 As one of the most basic classification methods for pattern recognition, clustering plays an important role in data analysis in various scientific fields. However, with the emergence of big data, clustering analysis continues to face new problems and challenges in frontier development such as computing complexity and computational cost. By studying the time complexity $O(nk)$ of the k-means clustering algorithm, we introduce the KD-tree as an index for the large number of nearest neighbor calculations, which scenario is special, in the iterative process, and propose approximate nearest neighbor search algorithms based on a single KD-tree or multiple KD-trees. The algorithms reduce the time complexity of the k-means clustering algorithm to $O(n\log k)$. It is verified by experiments that the algorithm based on multiple KD-trees has the comparable clustering quality with the k-means clustering algorithm.

【Key words】 clustering analysis; k-means clustering; KD-tree; approximate nearest neighbor

0 引 言

聚类分析是将物理或抽象对象的集合分成由类似的对象组成的多个类的过程。虽然聚类分析至今已有六十多年的发展历史，其间涌现出许多经典的聚类算法，但是聚类分析现在依然是数据挖掘、计算机视觉、医学数据分析等诸多研究领域的有力工具。目前，聚类研究的分支方向为数很多[1]，按方法论来分，有基于划分的聚类、基于密度的聚类和基于图的模型的聚类等。随着大数据的出现，聚类分析在前沿发展中不断面临着计算复杂度等新的问题和挑战[2]。

现今对于大数据聚类分析的算法研究，在不同的分支方向都有所突破和提升。其中，大数据聚类算法关注的焦点，是以最小化降低聚类质量为代价，提高算法的可扩展性与执行速度。结合当前最新研究结果[3-7]，大数据聚类算法的研究主要分为两个方向：一是优化算法本身的时间复杂度；二是通过并行环境计算。

在基于划分的聚类方面，现今主要的挑战是，当数据规模 n 很大或者聚类中心 k 很多时，进行的迭代次数过多，从而导致计算成本很高。现有的成果[8-9]中主要关注参数 n 的问题，而对 k 的关注度很低。因此本文针对 k 的问题，提出了基于 KD 树的近似近邻搜索来降低 k-means 算法的时间复杂度，并通过实验验证具有很好的聚类质量。

1 k-means 算法的近邻问题与特殊场景

在此先给出 k-means 聚类算法的流程。

对于任意数据集 X 和中心个数 k：

(1)选择 k 个点为初始质心，并记为 Q；

(2)对于每个点 $x \in X$，在 Q 中计算出距离 x 最近的点 q，将 x 加入 q 对应的集合；

(3)对于每个 $q \in Q$，计算集合的质心并更新

作者简介：薛丁文(1995-)，男，博士研究生，主要研究方向：海量数据聚类分析；李建中(1950-)，男，教授，博士生导师，主要研究方向：海量数据计算、无线传感网络。

收稿日期：2021-05-18

Q；

(4)重复(2)和(3)直到终止条件，返回 Q。

k-means 算法的时间复杂度为 $O(nk)$，是因为在每轮迭代中，对于每个点 x，都要和当前的 k 个中心计算距离，并选出最近邻的中心。这个过程就是不断重复的最近邻计算。

将这个过程形式化定义：给定集合 P 和查询点 q，在 P 中计算出和 q 距离最近的点，即 $\min\limits_{p\in P} d(q,p)$。

针对最近邻计算的问题，直观想法就是通过预处理的索引结构来减少距离计算的次数。然而不同于常规的最近邻计算问题，在 k-means 算法的场景时，集合 P 是变化的。这就要求建立索引的时间复杂度不能太高，同时可以降低比较次数。如果不能通过索引快速得到精确的最近点，那么也应该是近似的最近点。考虑到预处理的时间复杂度，在此选用 KD 树。

2　KD 树的简单介绍与构建算法

2.1　KD 树的简单介绍

KD 树解决的是在多维空间为数据集建立索引的问题，在一维空间时会变成二叉搜索树(BST)。二维空间的 KD 树示例如图 1 所示。

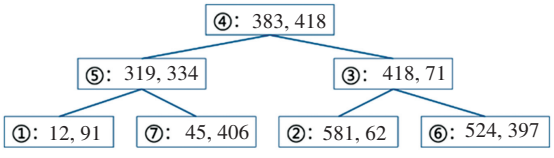

图 1　二维空间的 KD 树

Fig. 1　KD-tree in 2-dimensional space

对于 KD 树的每个节点来说，其左子树的点在某个维度的值都不超过该点在该维度的值；其右子树的点在某个维度的值都不低于该点在该维度的值。按树的深度会循环遍历每个维度。

以图 1 为例：对于根节点④，左子树点的横坐标都比点④的横坐标小，平面来看都在点④的左侧；对于节点⑤，左子树点的纵坐标都比点⑤的纵坐标小，平面来看都在点⑤的下侧。

通常根节点的选择和维度的方差有关系。在计算出每个维度的方差后，根据最大方差对应的维度对集合 P 作排序，然后选择中点为根节点。树的深度为 $\lceil \log_2 |P| \rceil$。

2.2　KD 树的构建算法

在多维空间时，维度的计算顺序可以有所调整。若记原有的维度序列为 $D=[1\cdots d]$，即 $x=[x_1\cdots x_d]$；维度序列的某种排列为 $RD=[i_1\cdots i_d]$。

KD 树的递归构建算法如下：

对于任意的集合 P 和维度序列 RD：

$Build(P,RD,depth)$：

(1)如果 $|P|=1$，创建叶节点并返回，递归终止；

(2)根据第 $RD[depth\%|RD|]$ 个维度的值，对集合 P 重新原地(升序)排序；

(3)记 mid 为 P 的中点，根据 mid 创建节点 v，并将集合 $P-\{mid\}$ 分为 $P1$ 和 $P2$；其中 $P1$ 包含 mid 左边的点，$P2$ 包含 mid 右边的点；

(4) $v.left=Build(P1,RD,depth+1)$；

(5) $v.right=Build(P2,RD,depth+1)$；

(6)返回节点 v。

KD 树的根节点为 $RT=Build(P,RD,0)$。

3　基于 KD 树的 k-means 聚类算法优化

3.1　基于单 KD 树的近似近邻查询

考虑到 k-means 算法的实际场景，随着收敛过程的进行，聚类中心 P 是不断分散并趋于稳定的。在这种情况下，如果允许少量的点被分类错误，对聚类中心的影响并不大。也就是说，可以在 KD 树上快速计算出近似的最近邻，以微小的误差代价来加快算法的执行时间。

如图 2 所示，通过查询点 q，从 KD 树的根节点出发，搜索到叶节点的路径为 $P=7\rightarrow13\rightarrow2\rightarrow9$。如果在 P 中计算 q 的最近邻，则为点 9。

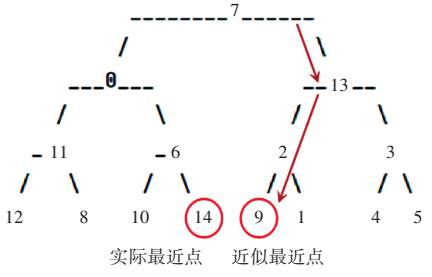

图 2　KD 树的树形搜索

Fig. 2　Search on KD-tree

如图 3 所示，近似的最近点 9 相比于精确的最近点 14，误差很小。

在此给出 KD 树的近似近邻查询算法。

对于 KD 树 RT 和维度序列 RD，给定查询 q：

$Search(RD,RT,q)$：

(1) $depth=0, PATH=\phi$；

(2) $while(RT\neq NULL)$：

①更新路径 PATH=PATH+ $\{RT\}$;
②比较维度 $i_d = RD[depth\%|RD|]$;
③如果 $q[i_d] < RT[i_d]$, $RT = RT.left$;
否则 $RT = RT.right$;
④更新 $depth = depth + 1$;
(3)返回 PATH。

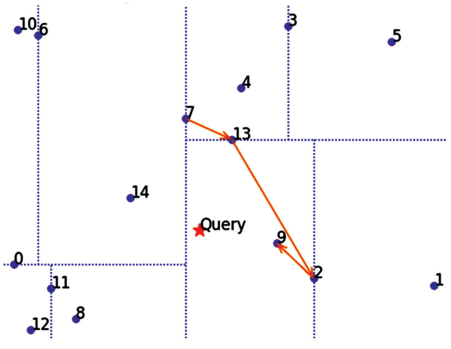

图 3 KD 树的平面路径

Fig. 3 Path of KD-tree in 2-dimensional space

在 k-means 算法的每轮迭代开始前,先根据当前的质心集合 Q, 构建出 KD 树;然后对于每个点 $x \in X$, 在 PATH 中计算出 x 的最近邻 q, 将 x 加入 q 对应的集合。

3.2 基于多 KD 树的交叉近邻搜索

可由理解 KD 树的查询比较为:查询 q 会在每个节点所对应的维度,沿着上升或下降的方向前进;维度不同,前进方向不同。因此对于不同的维度序列 RD, 得到的 KD 树会相同。同样的查询, q 经过不同的 KD 树,路径的点也不会完全相同。

如果构建多个 KD 树,合并不同树上的查询路径,则可以提高最近邻分类的正确率。

如图 4 所示,当构建两颗 KD 树时的查询路径,可以在局部搜索出最近的点,并有很高的正确率。

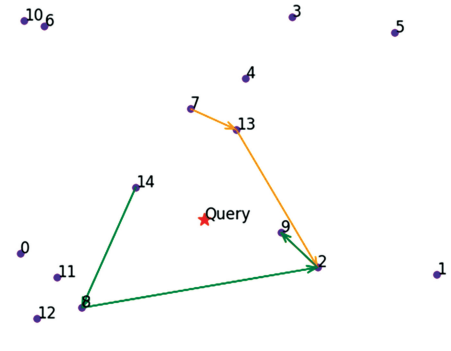

图 4 双 KD 树的交叉查询

Fig. 4 Search on two KD-trees

此时,在 k-means 算法的每轮迭代开始前,需要构建 $s(2 \leqslant s \leqslant d)$ 个 KD 树。其中, d 为数据集的维度。对于每个点 $x \in X$, 计算不同的路径 $P_1 \cdots P_S$, 在 $PATH = \bigcup_{i=1}^{s} P_i$ 中计算出 x 的最近邻 q, 将 x 加入 q 对应的集合。

4 算法实验

对于数据集 X 和集合 Q, k-means 聚类问题的量化误差为 $\mu_X(Q) = \sum_{x \in X} \min_{q \in Q} d(x,q)^2$。其中, $d(x,q)$ 为 x 到 q 的欧氏距离。

采用二维数据集 X 做实验, $|X| = 10^5$。固定迭代次数,令 X 经过 k-means 算法、单 KD 树优化算法和多 KD 树优化算法的结果为 Q_1、Q_2 和 Q_3。以 $\mu_X(Q_1)$ 为基准,计算 $\mu_X(Q_2)$ 和 $\mu_X(Q_3)$ 的相对值。

k-means 算法的运行时间见表 1。

表 1 k-means 算法的执行时间

Tab. 1 Time of the k-means algorithm

k	每轮迭代时间/s
7	5.609
15	11.291
31	22.085
63	42.639

基于单 KD 树的优化算法和基于多 KD 树的优化算法结果见表 2 和表 3。可以看出,基于 KD 树的优化算法计算时间加快很多。

表 2 基于单 KD 树的优化算法结果

Tab. 2 Results of algorithm using single KD-tree

k	$\lvert PATH \rvert$	分类正确率/%	迭代时间/s	量化误差/%
7	3	84.522	3.267	111.88
15	4	92.155	4.079	110.05
31	5	78.831	4.852	103.33
63	6	71.316	5.632	109.02

表 3 基于多 KD 树的优化算法结果

Tab. 3 Results of algorithm using multiple KD-trees

k	$\lvert PATH \rvert$	分类正确率/%	迭代时间/s	量化误差/%
7	3.92	96.331	3.671	100.03
15	6.02	97.370	5.678	101.41
31	7.79	90.509	7.152	99.08
63	10.24	89.031	8.441	99.64

$\lvert PATH \rvert$ 表示迭代过程中的平均 $PATH$ 长度,即用来计算最近邻点的个数,复杂度为 $O(\log k)$。表 3 的 $\lvert PATH \rvert$ 小于表 2 的 $\lvert PATH \rvert$ 的两倍,说明不同 KD 树的查询路径有重合的点;当待比较点的个数追加少于 $\log k$ 时,分类正确率显著提高。

根据表 3 可知, $\mu_X(Q_3) \approx \mu_X(Q_1)$, 说明基于多 KD 树的 k-means 优化算法聚类质量很高。