一种基于快速 k-近邻的最小生成树离群检测方法

朱 利 邱媛媛 于 帅 原 盛

（西安交通大学软件学院 西安 710049）

摘 要 离群检测也称异常点检测，是数据挖掘领域很有意义的热点问题之一，在很多方面都有广泛应用，如入侵行为、欺诈行为、医学上疾病前期的征兆等. 基于 k-近邻的算法能够很好的运用到大数据集上，因此在基于距离和基于密度的离群检测技术方面得到广泛应用. 然而 k-近邻算法的时间复杂度为 $O(N^2)$，随着数据集规模的增加，时间开销大大增加. 基于最小生成树的聚类算法在使用 Prim 或者 Kruskal 算法构建最小生成树时空间复杂度和时间复杂度均为 $O(N^2)$，聚类结果依赖于用户参数的选择，而且容易漏检稠密簇中的局部离群点. 针对以上问题，融合基于密度和基于聚类方法的优势，提出一种新的离群检测方法. 该方法具有以下优点：(1) 计算 k-近邻的时间复杂度为 $O(kN)(k \ll N)$；(2) 构建最小生成树的时间复杂度为 $O(N\log N)$；(3) 自适应识别聚类数目；(4) 能够检测出多种类型的离群数据. 最后通过大量实验验证了文中所提的 KDNS 算法，FkNN 算法和 ADC 算法的有效性. 实验结果表明，相对于现有算法，文中算法可以大幅度降低时间复杂度并显著提高离群检测率.

关键词 良分割对；最小生成树；k-近邻；自适应聚类；离群检测；数据挖掘

中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2017.02856

《计算机学报》

A Fast kNN-Based MST Outlier Detection Method

ZHU Li QIU Yuan-Yuan YU Shuai YUAN Sheng

(*School of Software Engineering, Xi'an Jiaotong University, Xi'an* 710049)

Abstract Outlier detection, also known as anomaly detection, is a very important foundamental research task in the field of data mining. It is mainly used for finding strange mechanism or potential danger, and aims to detecting those outliers their observations deviate so much from other observations and they are few suspicious data. Outliers, which are novel, unmoral and few, are often abandoned as noise or abnormal data. Outliers are also classified as many types, such as local, partial and so on. The techniques of outlier detection can be applied to many fields such as intrusion behavior, fraud, signs of early disease in the medical field and so on. Defining outliers by their distance to neighboring data points has been shown to be an effective non-parametric approach to outlier detection. The kNN-based algorithm could be used in big data sets efficiently, so it is widely applied for outliers detection based on distance and density. Unfortunately, the kNN-based algorithm's time complexity is $O(N^2)$, and it will be greatly increased with the size of date sets. The time complexity and space complexity of minimum spanning tree-based clustering algorithms using Prim's or Kruskal's method is $O(N^2)$, and the result of clustering depends on inputting parameters by users. Moreover, this algorithm can't detect outliers in high-density clusters. The existing MST-based algorithms become ineffective when provided with unsuitable parameters or applied to datasets which are composed of clusters with diverse shapes, sizes, and

收稿日期：2016-07-11；在线出版日期：2017-08-24. 本课题得到国家自然基金项目(61473220)、陕西省自然基金项目(S2015YFJM2129)资助. **朱 利**，男，1968 年生，博士，副教授，主要研究方向为机器学习、数字媒体理解. E-mail：zhuli@xjtu.edu.cn. **邱媛媛**，女，1988 年生，硕士研究生，主要研究方向为数据挖掘、机器学习. **于 帅**，男，1991 年生，硕士研究生，主要研究方向为推荐算法、数据挖掘. **原 盛**，男，1976 年生，博士，讲师，主要研究方向为数字媒体理解.

densities. Meanwhile, the most MST algorithms couldn't build tree dynamically, because of needing to know the distance between any two points in advance. In order to address these challenging problems, we proposed a new outliers detection method, which absorbs the advantages of distance-based method and density-based method. Firstly, this algorithm builds a split-tree to storage the information among data points. Secondly, we efficiently acquire all sets of well-separated pair decomposition on the whole dataset. Thirdly, all this algorithm partitions the input data set into several frames which are satisfy certain condition so that we can quickly obtain each point's k-nearest neighbors on the basis of the first two results. Fourthly, a minimum spinning tree is dynamically built according to the third result. In addition, we rank points which are suspected as outliers on the basis of its outlier factor by using the MST-based clustering without inputting parameter of cluster numbers manually. A new algorithm and a new metric are proposed to select the exact number of clusters and avoid insignificant clusters. And we detect all outliers at last. The time complexity of computing kNN and creating tree are $O(kN)$ and $O(N\log N)$, respectively. The experiments show that this new algorithm can detect both local outliers and global outliers without inputting the number of clusters from users. In the experiments, we use a series of real datasets and synthetic datasets to verify the efficiency and effectiveness of KDNS, FkNN and ADC proposed in this paper. The experimental results show that comparing with the previous approaches, our proposed algorithms can drastically reduce time complexity and significantly improve the rate of outlier detection.

Keywords　well separated pair; MST; kNN; auto clustering; outlier detection; data mining

《计算机学报》

1　引　言

离群检测是数据挖掘研究的一项重要任务，目的是发现在不同机制下，那些显著区别于大部分数据，且足够引起怀疑的少部分数据[1]. 这些新颖的不合常规的少部分离群数据通常被视为噪声或者异常点而被抛弃，然而在很多应用中这些离群数据可能蕴涵着重要的隐藏信息，如入侵行为、欺诈行为、医学上疾病前期的征兆等. 根据与整个数据集的差异度，离群数据分为全局离群点和局部离群点. 根据离群数据规模划分为离群点和离群簇[2]. 目前离群检测技术大致可以分为：基于统计的方法、基于近邻的方法、基于分类的方法、基于聚类的方法等[3]. 其中基于近邻的方法包括基于距离的离群点检测和基于密度的离群点检测方法.

基于距离的离群点检测方法最早由 Knorr 和 Ng 提出[4-5]，克服了基于统计的离群点检测算法中对数据的分布模型和分布参数的限制，并且适用于任何维度的特征空间. 该方法将每个数据点到其近邻数据点之间的距离大小作为离群的度量参照. 如果数据集中一个对象 t 与距离均值 u 的偏差大于或者等于用户指定一个距离阈值，则认为 t 就是一个离群点. 随后，Boriah 等人[6]提出了可以处理大数据集的一个基于 k-近邻(kNN)度量离群度的方法. 该算法的主要缺陷在于计算每一个数据点的近邻时都要扫描数据集一遍，随着数据集规模的增加，计算效率非常低. Ghoting 等人[7]做了进一步的改善，提出可以处理高维数据集的二段算法 RBRP，之后又提出了基于索引的算法提高计算效率；循环嵌套算法减少操作 I/O 次数；基于单元的算法提高离群点检测效率等[8]. 然而基于距离的离群点检测方法需要大量的距离计算，并且不能检测出局部离群点.

基于密度的离群点检测算法由 Breunig 等人[10]提出，根据数据点的邻域信息，考察数据点和其它近邻"密度"的差异性来判断该数据点是否是离群点. 这种差异性称为局部离群因子(Local Outlier Factor，LOF)，该方法可以解决局部离群点的问题. 对 LOF 的改进也提出了许多方法. Tang 等人[11]提出了基于连通性的离群因子(Connectivity-based Outlier Factor，COF). Jin 等人[12]基于逆近邻(Reverse

Nearest Neighborhood，RNN）提出离群影响因子（INFLuenced Outlierness，INFLO），使局部离群点挖掘效果进一步提高. He 等人[9]提出 CBLOF（Cluster-Based Local Outlier Factor），提高了局部离群点的检测精度. 由于 LOF 的特性，使得算法关注在一个点的离群度上，而对于整个数据集的离群簇不能检测出来.

基于聚类的方法是通过考察对象与簇之间的关系，将不属于任何簇的对象，或者属于小的偏远簇的对象作为离群点检测出来. 该方法以无监督方式检测，对许多数据类型均有效. 与其他离群点检测方法比较，基于聚类的检测方法在簇离群检测点上效果明显. 代表性的是由 Jiang 等人[13]提出的两阶段聚类离群检测算法（two-phase clustering process for outlier detection）. 算法的第一阶段利用改进的 k-means 算法将整个数据集划分成若干聚类，使得在同一簇中的数据点要么都是离群点，要么都不是. 第二阶段将第一阶段中形成的各个簇类用其质心代替，形成新的数据集，质心之间的距离作为权值，生成一棵最小生成树（Minimum Spanning Tree，MST）. 删除树中的长边，形成多个子树. 那些具有较少结点的树所对应的小簇类，被认为是离群点或离群簇. 删除长边的过程可以重复进行，直到总的簇类数目满足要求. 基于聚类的离群点检测方法缺点是检测结果的有效性高度依赖于所使用的聚类方法，这些方法对离群点检测可能不是最优的；而对于海量数据集，聚类方法通常开销很大，这可能是一个瓶颈.

1.1 问题和挑战

可以看出，基于距离、基于密度和基于聚类的离群点检测方法均存在各自的优势和缺陷，而对于分布复杂的数据集中离群点的检测，使用单一的离群点检测算法有一定的局限性. 如何能够将这些离群点检测方法进行融合，在克服各自方法的缺陷下，保留各自的优势，进行各种方法的融合并提出一种高效的离群点检测算法. 这种新的检测方法，不仅能够解决分布复杂，离群点类型多样，规模较大的数据集中的离群数据的前提下，而且在一定程度上能够提高时间效率和空间效率. 但由于问题固有的难度较多，目前还未出现在此思想下较为完善的解决办法. 本文主要面临的问题总结如下：

（1）如何检测出多种类型的离群点. 离群数据的类型可以划分为：全局离群点、局部离群点和离群簇. 如图 1 所示.

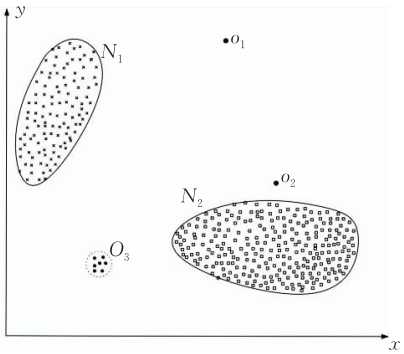

图 1 离群点和离群簇的分布

图 1 中，o_1，o_2，O_3 分别表示全局离群点，局部离群点和离群簇. 基于距离的离群检测方法将距离作为主要的度量标准，善于发现 o_1 这类具有远离其他数据点特征并且具有较大的距离权值的全局离群点；然而对于 o_2 这类点，虽然不具有远离其他数据点特征（即距离权值远小于 o_1），但是 o_2 的距离值相对于 N_2 中的数据点，o_2 的距离值又较大的局部离群点，检测效果并不好. 基于密度的离群点检测方法在基于距离的离群检测方法基础上，结合每个数据点的邻域信息，通过度量数据点的 LOF，很好的检测出局部离群点. 但是对于 O_3 中的数据点，内部点之间的 LOF 差异值均很小，不会被作为局部离群点检测出来；而相对于 N_1 和 N_2 中的数据点，O_3 中数据点的距离值均比较大，在对离群点的个数设置合理的情况下，才可能被当做全局离群点检测出来，由于对参数十分敏感，因此不能很好的用于实际问题. 基于聚类的离群点检测方法会将 O_3 作为一个数据点数目远小于正常簇（如 N_1，N_2）中包含的数据点数目的离群簇检测出来. 在实际的应用中，数据集分布复杂，包含有多种类型的离群点，使用一种离群点检测方法很难达到理想的检测效果，所以融合各种方法的优势，克服每一种方法存在的缺陷，建立新的更好的数据结构用来关联数据点之间的信息，设计新的方法计算数据点之间距离信息，具有实际的意义和价值.

（2）如何提高离群点检测方法的查询效率. 在处理大数据集的问题上，查询数据点的 kNN 是一种普遍适用的方法，目前的文献中，其计算方法上的时间开销一般为 $O(N^2)$，一些算法致力于通过索引机制来改善查询效率，如 R 树、M 树等. 文献[14]通过使用 R 树进行范围查找，可以快速的计算离群点，kNN 的时间开销减小到 $O(N\log N)$. 建立有效的索引机制，提高查询效率，使得时间开销再次降低到

《计算机学报》

$O(kN)(k \ll N)$，目前是一种挑战.

(3) 如何有效的度量离群因子. 离群点检测算法主要从数据点的“差异度”来衡量一个数据点是否可以被选为候选离群点，然后再根据离群点的规模(一般是数据集的 5%～10%)从候选离群点中选出“差异度”最大的 top-n[15-16] 个点作为最终的离群点. 这种“差异度”的计算方法直接影响到离群数据的检测结果. Zhang 等人[17] 讨论并使用基于距离的计算方法，Kim 等人[18] 认为一个数据点的“差异度”应该与其近邻点联系在一起，使用基于密度的计算方法. Huang 等人[19] 指出基于密度的计算方法是建立在数据点和其近邻点具有相似密度的基础上，但是对于一些数据点，其近邻分布在不同簇类中，会导致错误的检测结果. 同时对于一些特殊分布的数据集，例如数据集的分布为一条直线，即使数据点之间的距离相等，但是根据数据点的位置不同密度也应存在差异. 本文经过实验验证，该方法对于边界点离群检测结果并不好，如图 2 所示. 本文结合数据点在其近邻的“亲密度”排名，定义新的离群因子的计算公式，获得离群点.

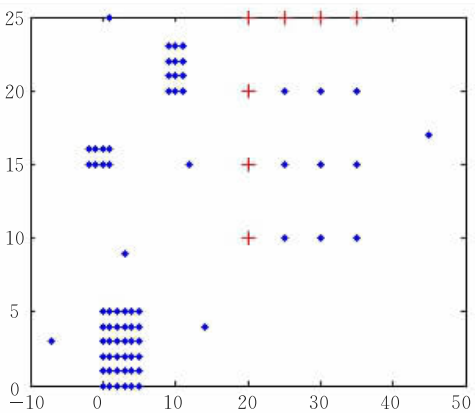

图 2　kNN 分布在不同簇类中的数据集分布

(4) 如何自适应的确定聚类数目. 基于聚类的离群点检测方法，簇类数目的确定是作为聚类算法的输入值事先设定好的. 然而在实际的应用中，很难明确原始数据集的簇类数目，因此本文进行改进，用边阈值和簇的分裂阈值来决定簇类的个数，尽量减少人工干涉的次数，自适应地根据边值的波动差异来判断簇类个数.

1.2　主要贡献

本文主要针对复杂分布的数据集中具有多种类型的离群点问题，提出了一种新的数据结构来关联数据点之间的信息，并在此基础上提高算法中数据点近邻的查询效率，设计新的离群因子公式，自适应的判断簇类数目，有效地检测出离群点. 具体的，本文的贡献总结如下：

(1) 提出一种新的基于 kd 树的数据结构算法：KDNS(KD-tree based New Struct)算法. 该算法首先将数据集在维度上进行划分，建立平分树，同时计算出所有的良分割数据集对，为后续数据点近邻的查询和聚类算法做准备.

(2) 基于 KDNS 数据结构算法，提出了一种快速查找 k 近邻的算法：FkNN(Fast k-Nearest Neighbors)算法. 该方法分为两个部分：① 筛除每个数据点 i 的非 k-近邻，利用良分割对的性质，选出一个数据集合 A_i，使得被操作的这个数据点 i 是集合 A_i 中的所有数据点的 k-近邻之一；② 将 1 中每个数据点对应的 A_i 集合进行处理，选出数据点的 k-近邻.

(3) 提出一种新的计算离群因子的方法：基于距离比值的加权排序离群因子：DRWR(Distance Ratio-based Weighted Rank)，据此检测出离群点.

(4) 提出一种新的自适应的判断簇类数目算法：ADC(Adaptive Detecting Clusters)算法. 该算法用边阈值和簇的分裂阈值来决定簇类的个数，尽量减少人工干涉的次数，自适应的根据边值的波动差异来判断簇类个数.

(5) 分别使用真实数据集和人工合成数据集进行实验，验证了本文所提出的 KDNS，FkNN 和 ADC 算法的有效性.

本文第 2 节概述离群点检测的相关工作；第 3 节详细描述本文所提离群点检测算法思想以及算法的实现过程；第 4 节分析算法复杂度；第 5 节给出实验结果和分析，第 6 节对全文进行总结.

2　相关工作

离群数据根据离群数据规模划分为离群点和离群簇；根据所处数据集的数据类型划分为矢量离群点、序列离群点、轨迹离群点和图形离群点等[20]. 除非另有说明，本文中探讨的所有离群检测算法是关于低维度的类矢量数据集中离群数据的检测.

基于距离的离群点检测算法主要使用欧几里得距离，k 维数据点 x，y 之间的距离计算公式为

$$dist(x,y)=[(|x_1-y_1|)^P+(|x_2-y_2|)^P+\cdots+(|x_k-y_k|)^P].$$

但是对于高维数据集，随着数据规模的增加，时间开

《计算机学报》

销会急剧增加.目前大部分的基于距离的离群点检测方法使用两种度量基准:局部邻域和 kNN.局部邻域是指数据点 p 是数据集 S 中关于参数 k,r 的一个离群点,称为 $DB(k,r)$-离群点,满足到 p 的距离不大于 r 的数据点个数不超过 k. kNN 是指对于参数 k 和 n,如果数据点 p 到其第 k 个近邻的距离小于不超过 $n-1$ 个点的对应的距离(即这最多 $n-1$ 个点各自的第 k 近邻距离),那么 p 为离群点.基于距离的离群检测方法可以概括为:嵌套循环算法[21-22]、基于索引机制[23]和基于划分的算法,近期 Sugiyama 等人[24]提出了基于采样的方法.根据不同的应用场合,还出现了许多针对特定应用背景的离群点检测算法,如数据流上的离群点检测[25],关于时间类型的数据上离群点检测等[26].

基于密度的离群点检测算法相对于基于距离的离群点检测算法具有更复杂的度量机制.因为它不仅考察数据点的局部密度,还需要考察该数据点近邻的局部密度,也就是数据点的局部密度与其近邻局部密度平均值的比值.目前局部离群点检测算法主要在于对离群因子的优化上(邻域的确定方法和离群因子的计算方法).最近 Mathew 等人[27]提出了新的 LOF 计算方法.

下面介绍几种常用的离群因子计算方法:

Breunig 等人[10]提出了局部离群因子 LOF(Local Outlier Factor)概念.给定一个整数参数 k,数据对象 p 的 LOF 被定义为

$$LOF_k(p)=\frac{\sum_{o\in N_k(p)}\frac{lrd_k(o)}{lrd_k(p)}}{|N_k(p)|},$$

其中 $N_k(p)$ 表示 p 的 k-近邻,$lrd_k(p)$ 为 p 的局部可达密度,被定义为 p 到 k-近邻内各对象平均可达距离的倒数:

$$lrd_k(p)=|N_k(p)|\Big/\sum_{o\in N_k(p)}reach\text{-}dist_k(p,o),$$

其中 p 到 o 的可达距离 $reach\text{-}dist$ 定义为

$$reach\text{-}dist=\max\{dist_k(p),dist(p,o)\}.$$

以上可以看出,p 的密度越小,并且(或)其近邻的密度越大,LOF 的值就越高.

Tang 等人[11]提出了 LOF 算法的一种变种,称之为 COF(Connectivity Outlier Factor),定义为

$$COF_k(p)=[A_{N_k(p)}(p)]\left[\frac{\sum_{o\in N_k(p)}A_{N_k(p)}(o)}{|N_k(p)|}\right]^{-1},$$

其中 $A_{N_k(p)}(p)$ 表示 p 与 k-近邻之间边长度的加权和,定义为

$$A_{N_k(p)}(p)=\frac{2}{k}\sum_{i=1}^{k-1}\frac{k-i}{k-1}l(e_1),$$

其中 $l(e_1)$ 表示边 e_1 的长度,e_i 定义为

$$e_i=d(o,p_{i+1})$$
$$=d(\{p_1,\cdots,p_i\},\{p_{i+1},\cdots,p_k\}),o\in\{p_1,\cdots,p_i\}.$$

以上可以看出,p 距离 k-近邻越远,COF 值大.

基于聚类的离群点检测方法是通过聚类后发现离群点,同时能够很好的检测出离群簇.离群点可以看做聚类挖掘的副产物,所以通过对聚类算法的进一步改进也可以用于离群点挖掘. Shi 等人[28]提出一种检测离群点和离群簇的算法,即 COID.传统的聚类算法,比如 k-means 算法,将数据进行聚类依赖于某些"中心点",对于非常规的簇类边界点效果并不好.基于图论的方法,典型的基于最小生成树聚类算法很好的对不规则边界的数据集进行聚类.经过多年众学者的研究,构建 MST 算法日臻成熟,时间复杂度由原来的 $O(N^2)$(例如 Prime 算法和 Kruscal 算法)降低到 $O(E\alpha(E,V))$[29]. March 等人[30]提出一种基于 kd 树和覆盖树的双树结构,来构建 MST. Wang 等人[39]利用分治思想提出了一种快速的基于 MST 的聚类算法,之后 Zhong 等人[31]在此思想上进行改进.与本文所研究的问题最为接近的工作有文献[32],该文主要基于 k-近邻构建不相似性度量矩阵,在此基础上建立 MST.由于该算法时间复杂度为 $O(N^2)$,因此在处理大数据集时有较高的时间代价.

目前还出现了关于聚类的算法,比如基于同步的[33]和基于压缩的方法[34],由于具有时间性能高的特点,已经运用于离群点检测[35].

良分割分离技术(Well-Separated Pair Decomposition,WSPD)最先由 Callahan 等人[36]提出,并对其应用进行了理论描述.不久提出一种在数据集上进行良分割成对分解获取稀疏图的方法[37],由于关于基于良分割对的方法在目前文献中很少出现,本文对基本概念稍作详细介绍.良分割对 WSP(Well-Separated Pair)定义如下.

定义 1(WSP). 以 c 为中心,r 为半径的球体,可以表示集合 $B=\{p\in\mathbb{R}^d:dist_2(c,p)\leq r\}$.给定一个分割阈值 $s>0$,如果数据集合 A 和 B 所在的最小矩形框 $R(A)$ 和 $R(B)$ 能够被半径为 r 的 d 维球体 S_a 和 S_b 分别包含,并且两个球体之间的距离不小于 sr,那么称集合 A 和 B 是良分离的.

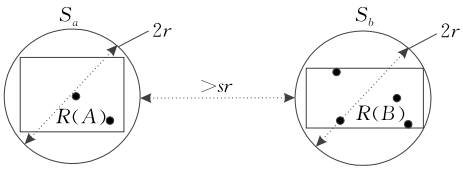

图 3　WSP 示意图

如果集合 A 和 B 关于一个很大的分离系数 s-良分离，那么：

(1) A 和 B 之间所有的两点间距离都是相近的；

(2) A 或 B 中两点之间的距离远远小于 A 和 B 之间的距离.

总之，目前存在的基于距离、基于密度和基于聚类的离群点检测方法均存在一些不足，不能够满足具有多种离群数据类型的离群检测.

3　数据存储方式

在离群点检测之前，建立有效的数据结构，进行数据存储，可以提高计算效率. 本文提出的 KDNS 是一种新型的数据结构方法，该方法继承了 KD 树的索引机制，并提高了数据间的关联性. 算法大体上包括两个步骤：(1) 将数据集进行划分，建立一棵平分树，树的分支代表某一维度上所有数据的平均值，左结点代表该维度上值小于等于平均值的数据集合，右结点代表该维度上值大于平均值的数据集合(3.1 节)；(2) 从平分树中获取 WSPD 集合(3.2 节).

3.1　构建平分树

关于数据集 S 的平分树 $T(S)$ 相关定义如下.

(1) 如果 $n=1$，$T(S)$ 由 S 中唯一的一个点构成；

(2) 如果 $n\geq 2$，v 包含 S 的矩形框定义为

$$R(S)=\prod_{i=1}^{d}[l_i,r_i],$$

其中，i 是数据的某一维度，l_i，r_i 分别表示维度 i 上的最小值和最大值，满足 r_i-l_i 值最大. 算法 1 描述了平分树构建过程.

算法 1.　splitTree( ).

输入：本地存储的所有数据点集合，记为 S

　　数据集的维度数，记为 D

输出：平分树的存储结构

1. 创建根节点 root，保存 S 和 $R(S)$；
2. 取 S 上第 i 维，i 满足 $r_i-l_i=\max(r_j-l_j)(j\in D)$；
3. 取 S 在第 i 维上的中值 $\mathrm{mid}=(l_i+r_i)/2$；沿着第 i 维的方向，以值 mid 为分割线，将数据集 S 划分为两个子集 $S(left)$，$S(right)$，分别是 root 根节点的左子树和右子树，其中 S_w：$S(left)=\{p\in S: p_i\leq(l_i+r_i)/2\}$，$S(right)=S\backslash S(left)$；
4. IF 子集 $S(left)$ 数据点的个数大于 1，THEN
5. 　$D1=\{j\in D-i\}$；
6. 　递归调用 splitTree($S(left)$，$D1$)；
7. ELSE
8. 　作为叶节点返回；
9. ENDIF
10. IF 子集 $S(right)$ 数据点的个数大于 1，THEN
11. 　$D2=\{j\in D-i\}$；
12. 　递归调用 splitTree($S(right)$，$D2$)；
13. Else
14. 　$S(right)$ 作为叶节点返回；
15. ENDIF

平分树的叶节点存储真实的数据点，内部结点存储该结点下所有孩子的信息以及能够包围该结点所在子树中所有数据点的最小矩形框.

3.2　获取 WSPD 集合

给定数据集 S 的平分树 $T=T(S)$，对于 T 中的任一结点 u，S_u 表示根节点为 u 的子树中所有叶子节点的数据集合. 对于 $\mathbb{R}^d$ 中的任意子集 X，$L_{\max}(R(X))$ 表示 X 的矩形框 $R(X)$ 的最长边.

s 表示分割系数，利用 T 计算 S 的 WSPD 过程如下：

(1) 初始化一个空队列 Q，对于树 T 中的每个内部结点 u 执行如下操作：将 u 的两个孩子结点和 w 组成一对数据集对 (v,w) 插入到 Q 队列中；

(2) 重复执行下列操作，直到队列 Q 为空：将队列中第一对 (v,w) 出列并从 Q 中删除. 如果 S_v 和 S_w 关于 S 是良分离，则输出 $\{S_v, S_w\}$. 否则，假设 $L_{\max}(R(S_v))\leq L_{\max}(R(S_w))$，将 w 的两个孩子结点 w_1，w_2 与 v 组成结点对 (v,w_1)，(v,w_2)，插入到队列 Q 中.

算法 2 描述了判断两个数据集是否为 WSP 的过程，算法 3 描述了获取 WSPD 集合的过程.

算法 2.　WSP( ).

输入：平分树中的节点，记为 u

　　平分树中的节点，记为 v

输出：u 和 v 是否为 WSP 的判断结果

1. 取结点 u 中矩形框 $R(u)$；
2. 取结点 v 中矩形框 $R(v)$；
3. IF $R(u)$ 和 $R(v)$ 满足定义一，THEN
4. 　取 P_u，满足 $P_u=\{p, p\in S\cap p\in u\}$；
5. 　取 P_v，满足 $P_v=\{p, p\in S\cap p\in v\}$；
6. 　输出 (P_uP_v)；
7. ELSE
8. 　IF $R(u)>R(v)$，THEN
9. 　　递归调用 WSP($u(left)$，v) $\cup$ WSP($u(right)$，v)；
10. 　Else

《计算机学报》

11. 递归调用 WSP(u,$v(left)$)$\cup$WSP(u,$v(right)$);
12. EndIF
13. EndIF

算法 3. WSPD( ).

输入:本地存储的所有数据点组成的集合,记为 S
数据集的维度数,记为 D
分割系数,记为 s

输出:数据集 S 中所有的 WSPD 组成的集合

1. T=splitTree(S,D);
2. FOR 每一个内部结点 $u\in T$,DO
3. $WP=WP\cup$WSP($u(left)$,$u(right)$);
4. END FOR
5. 输出 WP.

4 基于快速 k-近邻的 MST 离群检测方法描述

结合上文的研究分析,提出快速基于 k-近邻的 MST 的离群检测方法.整个算法的步骤如下:

(1)根据数据集的数据维度信息,将整个数据集进行平分,建立一棵平分二叉树;

(2)给定分离系数 s,利用平分树计算 WSPD 集合,并对集合进行最短连接,将连接后的值作为权值,构建一棵近似 MST;

(3)根据 WSPD 集合计算所有点的 k-近邻;

(4)程序自动确定删除长边数目.如果删除长边后新形成的两棵子树的总权值大于原来树的总权值,删除操作不进行.如果小于原来树的总权值,则进行删除长边操作.此处可以检测出离群簇和全局离群点;

(5)对已经形成的聚类结果,判断每个聚类中数据点的离群因子,将离群度最大的前 m 个数据点作为离群点.此处可以检测出局部离群点.

4.1 建立最小生成树

4.1.1 WSPD 的最短连接

以定义 1 为例,假设一对 WSP:$\{A,B\}$,令 S_a,S_b分别表示包含集合 A,B 的最小圆,(x_1,y_1),(x_2,y_2)分别表示 S_a,S_b的中心点.图 4 显示连接示意图,$\{A,B\}$的最短连接过程如下:

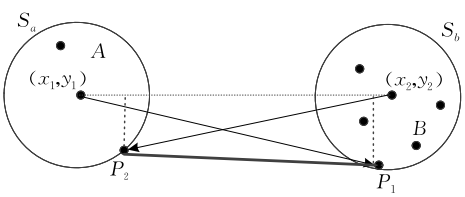

图 4 WSPD 最短连接示意图

(1)定义向量 $\boldsymbol{v}=(x_2-x_1,y_2-y_1)$;

(2)对于 A 中每一数据点(x_A,y_A),向量 $\boldsymbol{w}_A=(x_A-x_2,y_A-y_2)$;

(3)距离 $d_A=\boldsymbol{v}\cdot w_A=(v_x\cdot w_{A_x}+v_y\cdot w_{A_y})$表示点$(x_A,y_A)$在 $\boldsymbol{v}$ 上的投影.令数据点 p_1满足 $d_{p_1}=\min\{d\mid d\in d_A\}$;

(4)同理,对于 B 中每一个数据点(x_B,y_B)构成的向量 $\boldsymbol{w}_B=(x_B-x_1,y_B-y_1)$在 $\boldsymbol{v}$ 上投影,距离最短的数据点为 p_2;

(5)距离 $d=dis(p_1,p_2)$表示$\{A,B\}$的最短连接.

定义 2(AEMST 连通图). 给定一个有 n 个点的集合 S,假设 WP 是 S 的 WSPD 集合,$WP=\{\{A_1,B_1\},\cdots,\{A_m,B_m\}\}$,选择两点 $a_i\in A_i$,$b_i\in B_i$,使得(a_i,b_i)在$\{A_i,B_i\}$中是距离最近的两个点.连接 WP 集合中所有的$\{A_i,B_i\}$,那么构成的连通图称为 AEMST(Approximate Euclidean Minimum Spanning Tree)[38].

4.1.2 构建 EMST

构建 EMST 的过程描述如下:

(1)对 WP 中的所有$\{A_i,B_i\}$,进行最短连接,形成一个 WSPD 连通图 AEMST=$G(V,E)$;

(2)将 $G(V,E)$作为 Kruscal 算法的输入参数,生成 EMST(Euclidean Minimum Spanning Tree).

4.2 获取 k-近邻

定义 3(p 的 k-近邻). 对于数据点 p,$N(p,k)$表示距离(欧氏距离)p 最近的 k 个数据点集合.

定义 4(α-frame), 给定数据集 S,O_S表示矩形框 $R(S)$的中心.将以 O_S为顶点的区域划分成固定数目的扇形域,使得每个扇形域中的任意两个数据点 a,a'与 O_S构成的线段$\overline{O_Sa}$和$\overline{O_Sa'}$之间的夹角$\angle aO_Sa'$小于$\alpha=\dfrac{s}{s+1}$(其中 $s>1$).这些扇形域称为 α-frame.

定理 1. 假设$R_l(p)$表示某一扇形域 l 中所有数据点集合,其中 $l\in\alpha$-frame,p 表示 S 中任意一个数据点,a 和 b 是$R_l(p)$中的任意两个不相同的数据点,那么有 $d(a,b)<\max(d(a,O_S),d(b,O_S))$.

证明. 由于扇形域 l 角度小于 $\pi/3$,根据三角形定理,$d(a,b)$不可能是 a,b 和 O_S三个点构成的三角形中的最长边. 证毕.

定义 5. 令$\{A,B\}$是 $S\otimes S$ 关于分离系数 $s>1$ 的一对 WSPD,如果 B 中存一个数据点 b 是 A 中的数据点 a 的 k-近邻,那么 A 的数据点数$|A|\leq K$.

定理 2. $\{A,B\}$是良分离的,假设 C 是以 O_S为顶点的一个扇形域,a 和 b 是 C 中的任意两个不相同

《计算机学报》

的数据点，满足 $\angle aO_Sb < s/s+1(s>1)$，令 $A \cap C = \{a_1, a_2, \cdots, a_l\}$，并对这些点按照与 O_S 之间的距离大小进行排序. 那么，对于点 $a_i(i>K)$，其 k-近邻不会存在于 B 中.

证明.　如果 B 中存在一个数据点是 a_i 的 k-近邻之一，由定义 5 可知 $|A| \le K$，即 $i \le k$，这与条件 $i>k$ 相矛盾，故假设不成立.　　证毕.

基于以上定义，计算良分离数据集对计算数据点的 k-近邻过程如下：

(1) 对于结点 B，候选集 $X(B) = \bigcup_{i=1}^{q} A'_i$，$A'_i$ 满足：$\{A'_i, B\} \in \mathrm{WSP}$，$|A'_i| \le K$，$A' \le_i \cap A'_j = \varnothing (i \ne j)$；

(2) 对于结点 B，创建关于 O_B 的 α-frame，记为 ζ，划分 $X(B)$ 成若干扇形域 $X_C = X(B) \cap C, C \in \zeta$；

(3) 获取 X_C 中离 O_B 最近的前 k 个数据点，形成数据集 X'_C；

(4) 令 $N(B) = \bigcup_{C \in \zeta} X'_C$，那么对于 $N(B)$ 中的所有数据点 $p \in S$，p 的 k-近邻之一一定包含在结点 B 中；

(5) B 结点的两个孩子节点 B_1 和 B_2 的候选集 $X(B_1)$ 和 $X(B_2)$ 增加了数据集 $N(B)$；

(6) 算法依次从树 T 的根节点开始往下遍历，最终得到树 T 中每个叶子结点 q 的对应集合 $N(q)$，$q \in S$. 那么 $N(q)$ 中所有数据点的 k-近邻之一一定是叶子结点 q，也就是数据点 q；

(7) 计算出集合 S 中所有数据点 p 的近邻集 $N'(p)$，$N'(p)$ 中前 k 个离 p 最近的点组成 p 的 k-近邻.

获取 k-近邻的算法描述如下.

算法 4.　FKnn().

输入：本地存储的所有数据点组成的集合，记为 S
　　数据集的维度数，记为 D
　　参数 K，表示每个数据点近邻数
输出：所有数据点的 K 近邻集

```
1.  获取 S 的平分树 T = splitTree(S, D);
2.  获取 S 的 WSPD 集合 WP = WSPD(S, s, D);
3.  FOR T 中内一个内部结点 B, DO
4.      FOR WP 中每一对集合, DO
5.          IF {A'_i, B} ∈ WP 并且 |A'_i| ≤ K
6.              X(B) = X(B) ∪ A'_i;
7.          END IF
8.      END FOR
9.      X_C = α-frame(X(B), s);
10.     N(B) = getTopK(X_C, O_B);
11.     N(B) = N(B) ∪ N(B.parent);
12.     IF B 是叶子结点
13.         p_knn = N(B) 中前 k 个离 B 最近的结点集合;
14.     END IF
15.     P_knn = P_knn ∪ p_knn;
16. END FOR
17. 输出 P_knn.
```

4.3　基于快速 k-近邻的 MST 局部离群因子

在上文关于离群度算法分析的基础上，本文提出一种改进的离群度计算方法，步骤如下所示：

(1) 给定数据集 S，p 是 S 中任意数据点，$N_k(p)$ 表示对象 p 的 k-近邻. 基于距离比值的加权排序离群因子 DRWR 被定义为

$$DRWR_k(p) = \frac{\sum_{q \in N_k(p)} rto_q(p)}{|N_k(p)|} \times w_k(p),$$

式中 rto 代表对象 p 到其 k-近邻的距离与 k-近邻平均距离的比值，$w_k(p)$ 表示对象 p 到 k-近邻的距离排名的权值和.

$rto_q(p)$ 和 $w_k(p)$ 分别被定义为

$$rto_q(p) = \frac{d(p,q)}{\frac{1}{k}\sum_{o \in N_k(q)} d(q,o)},$$

其中 $rto_q(p) = \frac{rank_q(p)}{k}$，$rank_q(p)$ 表示对于 $o \in C(q)$ 和 $d(p,q)$ 在关于 $d(q,o)$ 的集合中的排序位置 loc. $o \in C(q)$ 表示对象 q 所在簇类的集合.

(2) 令 $D_o = \{p \in D \mid O_k(p) \le O_{\max}\}$，其中 $O_{\max}$ 的选取使得 D_o 的大小是 D 的 β 倍，本文将 β 设置为 75%. 将 $DRWR_k(p)$ 标准化为

$$Z_k(p) = \frac{1}{S_k}(DRWR_k(p) - \bar{O}_k),$$

其中 $\bar{O}_k = \frac{1}{|D_o|}\sum_{p \in D} O_k(p)$，$S_k^2 = \frac{1}{|D_o|-1}\sum_{p \in D}(O_k(p) - \bar{O}_k)^2$.

离群因子算法描述如下.

算法 5.　LOF().

输入：本地存储的所有数据点组成的集合，记为 S
　　所有数据点的 K 近邻集
输出：数据集的离群因子集合

```
1. FOR each point p in S, DO
2.     获取 p 的离群因子 DRWR_k(p);
3.     将 DRWR_k(p) 降入离群因子集合 DRWR_k(S);
4. END FOR
5. 输出离群因子集合 DRWR_k(D).
```

4.4　自适应获取基于 MST 的聚类数目

聚类的目的是使同簇类中的对象相似度尽可能高而不同簇类中的对象相似度尽可能低. 基于此，本文用最小生成树中边权值大小度量相似度，自动判断聚类数目. 过程如下：

(1)假设 $T(V,E)$ 是关于数据集 S 的无向加权生成树,树中的边权值表示树中两点之间的距离,V 代表数据点集合,E 代表两个数据点之间的距离集合.树的分裂权值总和 SWS 定义为

$$SWS_T=\frac{\sum_{q\in V,p\in V} l_{path(q,p)}}{\sum e_{path(q,p)}},$$

式中 $path(q,p)$ 表示对象 p 到对象 q 所有边(即 p 到 q 的路径),$l_{path(q,p)}$ 表示从对象 p 到对象 q 所有边权值的总和.$e_{path(q,p)}$ 表示从对象 p 到对象 q 的边个数;

(2)删除 T 中的一条边,T 分裂成两颗子树 T_1 和 T_2,如果 $SWS_{T_1}+SWS_{T_2}<SWS_T$,表示树 T 可以分裂成两棵子树,两棵子树中的数据点分别组成两个类簇;

(3)对 T 中的大于某个阈值的所有长边进行遍历,找出去掉任一长边后使得树总权值最小的长边;然后将该长边进行删除;

(4)对形成的新子树,递归进行(2)、(3)操作,直到树的总权值的大小变化幅度小于某一阈值为止;

(5)最后,删除边的总数加一对应形成的聚类数目.

算法 6. AutoClusters().

输入:平分树,记为 T
阈值,记为 $minC$
阈值,记为 $differ$
输出:簇类集合

```
FOR 树 T 中的每个节点 e, DO
    IF |e|>minC, THEN
      numC++;
      SWS_T=SWS(T);
      {T1,T2}=RemoveEdge(e,T);
      SWS_T1=SWS(T1);
      SWS_T2=SWS(T2);
      IF (SWS_T1+SWS_T2)-differ≤10^-6, THEN
         BREAK;
      ELSE
        differ=SWS_T1+SWS_T2;
        IF SWS_T1+SWS_T2<SWS_T, THEN
           C=C ∪ splitCluster(T1,numC,differ);
        ENDIF
    ENDIF
  ENDIF
END FOR
输出簇类集合 C.
```

4.5 基于快速 k-近邻的 MST 离群点检测

通过自适应聚类算法删除长边后,形成若干类簇,判断每个类簇中数据点的个数,将数据点个数明显小于其他簇的类簇作为离群点或者离群簇输出(本文中将数量小于 $\alpha\times|S|/K$ 的类簇做为离群点/簇,其中 $|S|$ 表示数据集 S 的数据总数,K 表示簇类数量,$\alpha=0.1$).

通过离群度算法计算出每个类簇数据点的离群度,将拥有较大离群度的数据点作为候选离群点.算法描述如下.

算法 7. OutlierDetec().

输入:簇类集合,记为 C
基于距离比值的加权排序离群因子
阈值,记为 $minC$
输出:离群点集合

```
FOR C 中每一个簇 cluster, THEN
    IF cluster.size <minC, THEN
      O=O∪cluster.point;
    ENDIF
    FOR each point p in cluster, THEN
      IF p 的离群因子与 mean 比值最大,THEN
        O=O∪p;
      ENDIF
    END FOR
END FOR
输出离群点集合 O.
```

说明:其中的 $mean$ 表示数据点所在的簇类中所有点离群度的平均值,$minC$ 表示离群点的个数,即 $\alpha*|S|/K$.

5 算法复杂度分析

5.1 WSPD 算法时间复杂度

根据平分树的特点,构建平分树的时间复杂度为 $O(N\log N)$.

引理 1. 构建WSPD的时间复杂度为$O(N\log N)$.

证明. 建立一棵平分树的时间复杂度为 $O(N\log N)$.任一内部结点 u 调用函数 WSP($L(u)$,$R(u)$)时间为 $O(|P_u|)$.假设 Q_d 表示到深度 d 的所有内部结点集合.Q_d 中所有结点调用函数 WSP 所花费的总时间为 $O(\sum_{v\in Q_d}|P_v|)=O(n)$.根据平分树的特点,树的高度最大为 $O(\log N)$,因此总的时间复杂度为 $O(N\log N)$. 证毕.

5.2 构建 EMST 算法时间复杂度

由算法 3 可知,计算 G 的时间复杂度为 $O(E)$,E 表示 WSPD 的对数.Kruskal 算法时间复杂度为 $O(E\log N)$.将 G 作为 Kruskal 算法的输入,EMST 的算法时间复杂度为 $O(N\log N)$.

《计算机学报》

5.3　快速 k-近邻算法时间复杂度

引理 2.　快速 k-近邻算法时间复杂度为 $O(kN)$.

证明.　由于 WSPD 只有 $O(N)$ 对，每一对中提供的数据点数目最多为 k，候选集 $X(B)$ 的总大小为 $O(kN)$. 算法从树 T 的根节点开始遍历，后续结点的候选集增加来自于父节点的 $O(kN)$ 数目的数据集. 当遍历到只包含一个数据点的叶节点时，总大小为 $O(kN)$. 因此，快速 k-近邻算法的时间复杂度为 $O(kN)$.　　证毕.

5.4　基于快速 k-近邻的 MST 局部离群因子算法时间复杂度

根据算法 5 知，每个点的离群因子的复杂度为 $O(k^2)$，整个数据集的离群因子计算算法时间复杂度为 $O(k^2N)$.

5.5　基于快速 k-近邻的 MST 离群点检测时间复杂度

算法 6 是从树中删除长边形成聚类的操作时间复杂度为 $O(E)$，E 表示近似最小生成树的边数目. 算法 7 的时间复杂度为 $O(N)$.

6　实验结果和分析

算法用 JAVA 语言编写，在 Eclipse 下编译，在一台配置 Intel(R)Core(TM) i5、3.1 GHz CPU、4 GB 内存的 PC 机上运行. 本文提出的基于快速 k-近邻的 MST 离群检测方法，简称 FKMOD(Fast K-nearest neighbor and MST-based Outliers Detection).

6.1　MST 和聚类结果比较和分析

为了验证算法自动确定聚类数目和检测离群数据的有效性，在合成数据集上进行测试分析，并与 Prim 算法所获得的 MST 结果和聚类结果进行比较.

(1) Data1 数据集

Data1 有 473 个数据点，由 5 个簇组成. 该数据集的主要特征是右上角的 1 个大的稀疏簇类中包含了 3 个小的稠密簇类. 数据集有 6 个离群点. 实验结果如图 5、图 6 所示.

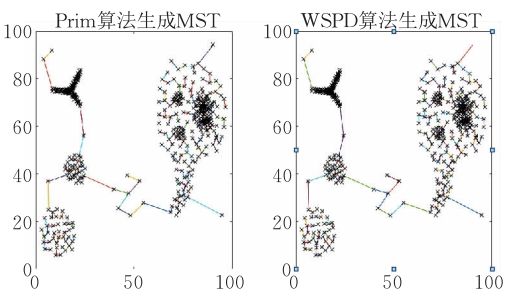

图 5　Prim 算法最小生成树和 WSPD 算法最小生成树

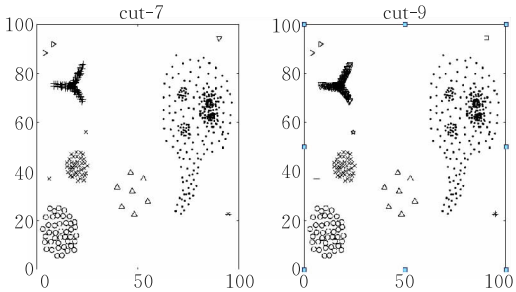

图 6　自动聚类算法减掉长边后的簇类和离群检测

图 5 显示两种算法的 MST 结果一致，说明本文中的基于 WSPD 构建的 MST 具有准确性.

图 6 中 cut-9 和 cut-7 分别表示减去 9 条长边和 7 条长边后的结果，从图中可以看出算法通过自适应聚类算法剪枝后不仅能很好的将数据集进行分类，而且能够检测出所有的全局离群点. 在 cut-9 中有 1 个局部离群点被聚在“x”类型表示的类簇中，通过本文局部离群因子的检测算法，知道该离群点在该簇类中具有最大的离群因子，因此最终将该离群因子检测出. 而在 cut-7 中有两个局部离群点被聚在“x”类型表示的类簇中，同样利用局部离群因子检测算法，取得离群因子最大的两个值，对应的数据点即为这两个局部离群点.

(2) Data2 数据集

Data2 数据集由 4 个簇组成：两个稠密高斯分布的簇，两个密度相异的均匀分布的簇. 数据集中有 7 个离群点. 实验结果如图 7、图 8 所示.

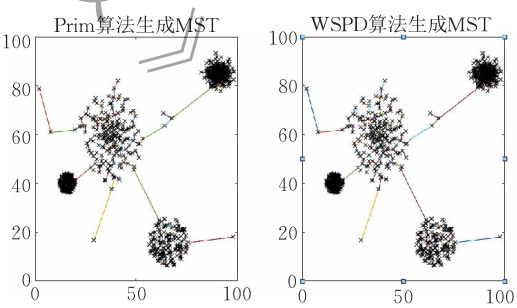

图 7　Prim 算法最小生成树和 WSPD 算法最小生成树

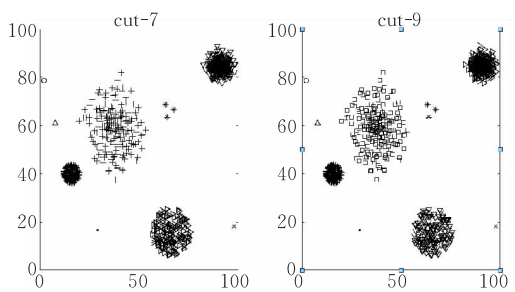

图 8　自动聚类算法减掉长边后的簇类和离群检测

《计算机学报》

图 7 显示两种算法的 MST 结果一致，说明本文中的基于 WSPD 构建的 MST 具有准确性.

图 8 中 cut-9 和 cut-7 分别表示减去 9 条长边和 7 条长边后的结果，从图中可以看出通过自适应聚类算法剪枝后 cut-9 不仅能很好的将数据集进行分类，而且能够检测出所有的全局离群点和一个离群簇. cut-7 中虽然将其中两个“+”类型的两个类合并为同一聚类，这是因为两个簇的距离值不足以被算法剪枝，但是也很好的检测出所有的全局离群点和一个离群簇.

(3) 真实数据集

为了进一步验证构建 EMST 算法的有效性，采用了文献[31]中使用的真实数据集进行分析对比，见表 1.

表 1 EMST 算法数据集特征

数据集	数据点数	维度
t4.8k	8000	2
MNIST	1000	784
ConfLongDemo	15 000	3
MiniBooNE	4000	50
Shuttle	5000	9
Drug consumption	1885	32

本文使用文献[31]中两个评判 MST 的指标：边正率和权值误率，定义如下：

$$ER_{\text{edge}}=\frac{N-|E_{\text{EMST}}|-1}{N-1},\ ER_{\text{weight}}=\frac{W_{\text{EMST}}-W_{\text{exact}}}{W_{\text{exact}}}$$

$|E_{\text{EMST}}|$表示错误边个数，$|E_{\text{EMST}}|$越小，ER_{edge}值越大，边正率越高；W_{EMST}表示 EMST 边权值总和，W_{exact}表示 MST 边权值总和，W_{EMST}越小，ER_{weight}值越小，权值误率越小. 数据集 t4.8k 和 MNIST 的实验结果如图 9 所示；数据集 ConfLongDemo 和 MiniBooNE 的实验结果如图 10；数据集 shuttle 和 wine MiniBooNE 的实验结果如图 11；从图 9、图 10 和图 11 中可以看出本文所构建的 EMST 能够到达 Prim 算法构建的 MST 的效果.

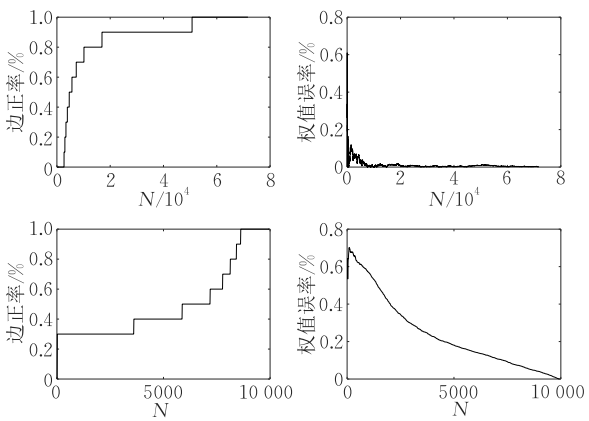

图 9 数据集 t4.8k 和 MNIST 的最小生成树实验结果

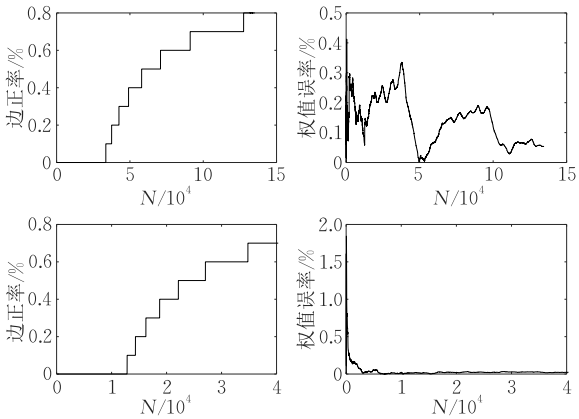

图 10 数据集 ConfLongDemo 和 MiniBooNE 的最小生成树实验结果

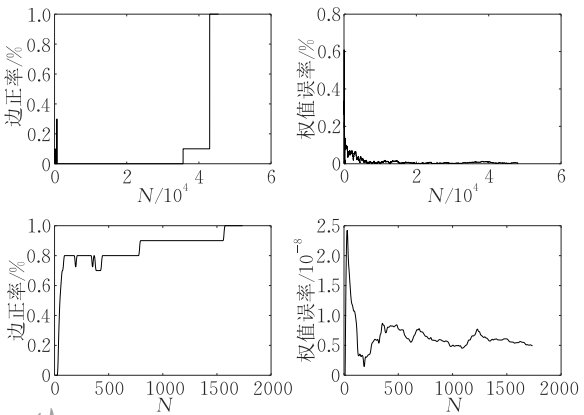

图 11 数据集 shuttle 和 wine 的最小生成树实验结果

6.2 离群点检测结果对比和分析

在真实数据集上，将本文提出的算法 FKMOD 与 LOF、COF 算法、KNMOD 算法[32]进行了比较，检验算法的时效性. 实验数据皆来自 UCI 机器学习数据库，包括 lymphography(简称 Lym)、Wholesale customers data(简称 Wcd)、Image segmentation(简称 Is)、Iris. 它们的数据特征见表 2. 为了方便离群检测实验，本文做了一些处理：(1) 从某簇中随机选出若干个点作为离群点(从第一簇中选择)与其他簇的所有数据点构成新的数据集，这些选出的离群点相对于其他簇可以认为是不同机制下产生的离群点；(2) 试图找出这 15 个离群点.

表 2 UCI 数据集特征表

数据集	实例点个数	属性数	离群数
Lym	148	18	6
Wcd	444	6	4
Is	1815	19	15
Iris	115	4	15

Wcd 数据集的按照 channle 类别可以分为 2 类，分别包含 298 和 142 个数据点，按照 Region 类别可

计算机学报

分为 3 类，分别包含 77、47、316 个数据点. 该数据集的 6 种属性取值特征见表 3.

表 3　Wcd 数据集属性取值特征

属性	最大值	最小值	平均值	标准差/方差
FRESH	3	112151	12000.30	12647/329
MILK	55	73498	27.00	7380/377
GROCERY	3	92780	7951.28	9503/163
FROZEN	25	60869	3.71.93	4854/673
DETERGENTS	3	40827	2881.49	4767/854
DELICATESSEN	3	47942	1524.87	2820/106

为了实验需要，添加 4 个离群点，取值如表 4 所示.

表 4　Wcd 数据集离群数据特征

离群点	FRESH	MILK	GROCERY	FROZEN	DETER-GENTS	DELICA-TESSEN
1	2	50	1	20	1	1
2	112200	73600	5	28	4	6
3	2	73600	93000	60900	40850	48000
4	112202	50	1	21	20000	30000

通过多次反复实验选取使实验效果较好的参数设置，参数设置如表 5.

表 5　Lof、Cof、KNMOD 和 FKMOD 算法参数设置

算法	参数	Lym	Wcd	Is	Iris
Lof	k	5	4	13	12
Cof	k	5	4	13	12
KNMOD	k	5	4	13	12
FKMOD	s	1.2	1.3	10	1.2
	k	≥3	≥3	≥3	≥3
	c	2	3	10	9

注：表中 k 为 kNN 中的 k；s 为分割因子(>1)；c 为剪边个数(一般取值范围为：离群点个数<c<离群点个数+类数).

对于任意的整数 $m\geq1$，如果 m 表示一种离群点检测模式最多检测出 O_m 个离群点时返回的最少数据数目，准确率为 $p=O_m/m$. 离群数据检测结果如表 6.

表 6　Lof、Cof 和 FKMOD 算法离群数据检测结果

数据集	Lof	Cof	KNMOD	FKMOD
Lym	0.400	0.333	0.600	0.667
Wcd	0.600	0.091	0.750	0.750
Is	0.012	0.026	0.483	0.517
Iris	0.016	0.150	0.500	0.600

由于 15 个目标离群点来自于同一个簇，易形成小簇，相对于其他的簇可看做离群簇. 实验结果显示，LOF 和 Cof 算法对于检测数据集 Lym 中的全局离群点具有较好的效果，Cof 算法对于检测具有离群簇的数据集 Wcd，Is 和 Iris 的离群点检测效果最差. KNMOD 算法在 4 个数据集上的离群检测效果均优于 Lof 算法和 Cof 算法. FKMOD 算法能够充分考虑数据集中簇类的密度差异性，分布多样性和簇类大小的不同，先通过聚类检测出离群簇，然后根据离群因子的检测出全局离群点和局部离群点. 表 6 可以看出，本文所提的算法具有最高的检测率，即获得最佳的离群检测效果. 通过 6.1 节的分析，本文算法擅长检测离群簇，全局离群点和局部离群点，在上述对比试验总表现了较好的效果.

6.3　算法的稳定性

图 12 显示了实验中所有数据集随着数据规模的变化，执行离群点检测算法总的执行时间. 图中一共有 6 条边，其中两条边显示时间复杂度为 $O(N)$ 和 $O(N\log N)$ 的算法的执行时间. 为了便于描述，EMST 表示构建基于 WSPD 的 MST 算法执行时间，AC 表示执行自适应聚类算法的时间，KNN 表示执行 k-近邻算法的时间，OD 表示基于 MST 的离群检测算法的总的执行时间.

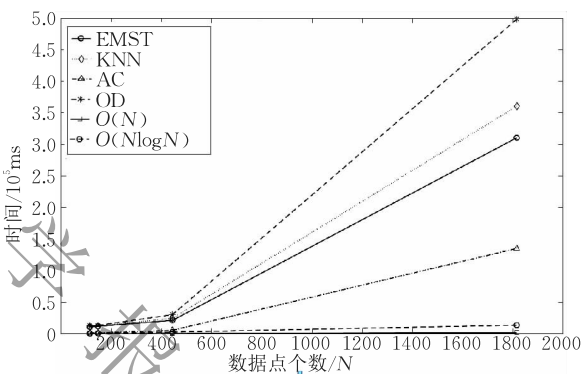

图 12　FKMOD 算法各阶段的执行时间

从图 12 可以看出，构建 EMST、执行聚类算法的时间均不超 $O(N\log N)$，执行 k-近邻算法的时间接近 $O(N\log N)$. 由此构建 EMST 的算法复杂度远小于 Prim 算法 $O(N^2)$.

6.4　时间性能

图 13 中 DB-KNN，DBMax-KNN 和 KNN 分别表示实验中所有数据集在分别使用 DB 算法、DB-Max 算法和本文所提算法(WSPD-KNN)计算 k-近邻时的时间性能对比结果. 从图中可以看出随着数据量的增加，KNN 计算 k-近邻的时间远小于其它两种算法. 图中 KNN 在 400～500 处随着数据点增加反而时间下降，是由于虽然数据点个数增多，单个数据维度大大下降导致的.

图 13 显示在数据规模较小(<500)的情况下，DB 和 DB-Max 算法具有较短的时间，但是随着数据规模的增大(>1000)，这两种算法的时间开销呈

《计算机学报》

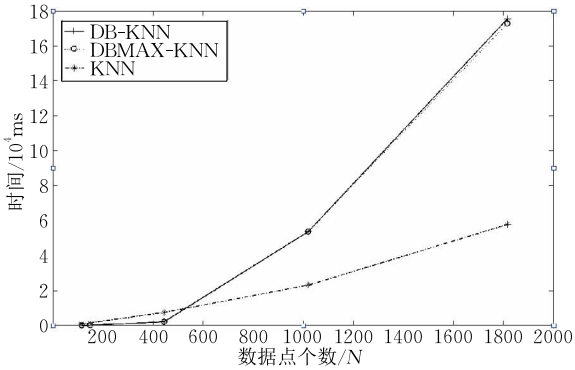

图 13 DB、DB-Max 和 KNN 算法计算 k-近邻时间对比

现指数增加，而 KNN 算法计算 k-近邻的时间最短，时间开销随数据规模的增大呈倍数增大，时间性能的优越性很好的体现了出来.

6.5 参数敏感度分析

本文涉及到的主要参数有：k-近邻的参数 k，分割因子 s，下面对真实数据集 Lym、Wcd、Is、Iris 中 3 个参数的敏感度进行分析，实验结果如图 14 所示.

由于参数 k 主要用来 k-近邻，分割因子 s 主要用来计算 wspd，因此左图中参数 k 对应的时间是指执行 k-近邻算法的时间，右图参数 s 对应的时间是指计算 wspd 的时间. 图 14 表示 k 与 s 取不同的值，对算法的时间性能影响不大.

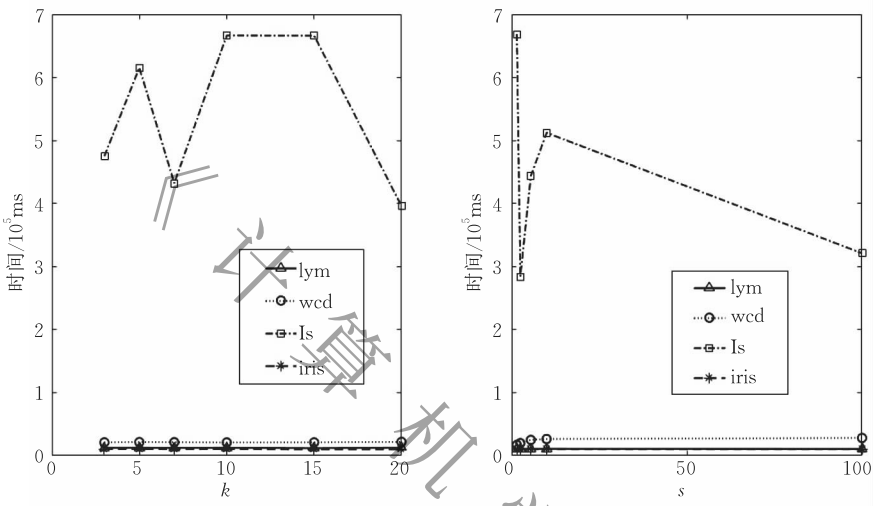

图 14 左图表示 k 取不同的值，k-近邻算法的时间；右图表示 s 取不同的值，计算 wspd 的时间

7 结束语

传统的基于 MST 的聚类算法，聚类质量受离群数据的影响很大，因此离群数据被作为噪声或者污染点被排除，然而在实际应用中这些离群数据包含着重要的信息. 本文中我们提出了一种快速基于 k-近邻最小生成树的离群检测方法，首先在原始数据集上构建平分树，计算出所有的良分离数据集对，在此基础上建立 MST，计算数据点的 k-近邻，然后通过减枝确定聚类结果，检测出全局离群点，最后通过计算局部离群因子检测出局部离群点. 实验验证了算法的有效性和实效性，尤其善于在圆形或者球形数据集上进行聚类和离群检测.

进一步的研究工作：本文算法有效性很大的依赖于 WSPD 集合，而用来存储数据的平分树类似 k-d 树的结构，由于树结构本身的特性，会在数据维度增大的时候，时间性能变得低下. 自适应的聚类算法的参数使用边阈值参数代替据类个数的参数，虽然能够根据边阈值进行剪边，但是并没有完全实现自动聚类的效果. 对于以后的研究，有以下几点值得改进：(1) 算法需要在自适应确定聚类数目时尽可能少的参数设置；(2) 提升在高维度数据集上的检测效果；(3) 算法应用于处理外存储的大规模数据集.

致 谢 在此，我们向对本文的工作给予支持和建议的同行和评审人表示由衷的感谢！

参 考 文 献

[1] Hawkins D-M. Identification of Outliers. London, UK: Chapman and Hall, 1980

[2] Han J, Kamber M. Data mining: Concepts and techniques. Second Edition. Data Mining Concepts Models Methods & Algorithms, 2000, 5(4): 1-18

[3] Chandola V, Banerjee A, Kumar V. Anomaly detection: A survey. ACM Computing Surveys, 2009, 41(3): 75-89

[4] Knorr E-M, Ng R-T, Tucakov V. Distance-based outliers: Algorithms and applications. VLDB Journal, 2000, 8(3): 237-253

[5] Knorr E-M, Ng R-T. Algorithms for mining distance-based outliers in large datasets//Proceedings of the 24th International Conference on Very Large Data Bases. New York, USA, 1998: 392-403

[6] Boriah S, Chandola V, Kumar V. Similarity measures for categorical data: A comparative evaluation//Proceedings of the 8th SIAM International Conference on Data Mining. Atlanta, USA, 2008: 243-254

[7] Ghoting A, Parthasarathy S, Otey M-E. Fast mining of distance-based outliers in high-dimensional datasets. Data Mining & Knowledge Discovery, 2008, 16(3): 349-364

[8] Yang Yi-Dong, Sun Zhi-Hui, Zhu Yu-Quan, et al. A fast outlier detection algorithm for data streams based on dynamic grids. Journal of Software, 2006, 17(8): 1796-1803 (in Chinese)
(杨宜东，孙志挥，朱玉全等. 基于动态网格的数据流离群点快速检测算法. 软件学报，2006，17(8)：1796-1803)

[9] He Z, Xu X, Deng S. Discovering cluster-based local outliers. Pattern Recognition Letters, 2003, 24(9-10): 1641-1650

[10] Breunig M-M. LOF: Identifying density-based local outliers. ACM SIGMOD Record, 2015, 29(2): 93-104

[11] Tang J, Chen Z. Enhancing effectiveness of outlier detections for low density patterns//Proceeding of the Advances in Knowledge Discovery and Data Mining 6th Pacific Asia Conference. Taipei, China, 2002: 535-548

[12] Jin W, Tung A-K-H, Han J, Wang W. Ranking outliers using symmetric neighborhood relationship. Lecture Notes in Computer Science, 2006, 3918: 577-593

[13] Jiang M-F, Tseng S-S, Su C-M. Two-phase clustering process for outliers detection. Pattern Recognition Letters, 2001, 22(6-7): 691-700

[14] Wang Xi-Te, Shen De-Rong, Bai Mei. BOD: An efficient algorithm for distributed outlier detection. Chinese Journal of Computers, 2016, 39(1): 36-51(in Chinese)
(王习特，申德荣，白梅等. BOD：一种高效的分布式离群点检测算法. 计算机学报，2016，39(1)：36-51)

[15] Lozano E, Acuña E. Parallel algorithms for distance-based and density-based outliers//Proceedings of the 15th IEEE International Conference on Data Mining. Houston, USA, 2005: 729-732

[16] Angiulli F, Basta S, Lodi S, et al. Distributed strategies for mining outliers in large data sets. IEEE Transactions on Knowledge & Data Engineering, 2012, 99(7): 1520-1532

[17] Zhang K, Hutter M, Jin H. A new local distance-based outlier detection approach for scattered real-world data. Lecture Notes in Computer Science (LNCS), 2009, 5476: 813-822

[18] Kim S, Cho N W, Kang B, et al. Fast outlier detection for very large log data. Expert Systems with Applications, 2011, 38(8): 9587-9596

[19] Huang H, Mehrotra K, Mohan C K. Rank-based outlier detection. Journal of Statistical Computation & Simulation, 2013, 83(3): 518-531

[20] Zhang J. Advancements of outlier detection: A survey. ICST Transactions on Scalable Information Systems, 2013, 13(1-3): 1-24

[22] Bay S D, Schwabacher M. Mining distance-based outliers in near linear time with randomization and a simple pruning rule//Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington, USA, 2003: 29-38

[23] Ramaswamy S. Efficient algorithms for mining outliers from large data sets. ACM SIGMOD Record, 2000, 29(2): 427-438

[24] Sugiyama M, Borgwardt K-M. Rapid distance-based outlier detection via sampling//Proceeding of the Advances in Neural Information Processing Systems. Navada, USA, 2013: 467-475

[25] Chandore P, Chatur P. Outlier detection rechniques over streaming data in data mining: A research perspective. International Journal of Advanced Research in Computer Science and Software Engineering, 2012, 2(1): 12-16

[26] Gupta M, Gao J, Aggarwal C, et al. Outlier detection for temporal data. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(9): 2250-2267

[27] Bai M, Wang X, Xin J, et al. An efficient algorithm for distributed density-based outlier detection on big data. Neurocomputing, 2015, 181(C): 19-28

[28] Shi Y, Zhang L. COID: A cluster—outlier iterative detection approach to multi-dimensional data analysis. Knowledge and Information Systems, 2011, 28(3): 709-733

[29] Chazelle B. A minimum spanning tree algorithm with inverse-Ackermann type complexity. Journal of the ACM, 2000, 47(6): 1028-1047

[30] March W B, Ram P, Gray A G. Fast euclidean minimum spanning tree: Algorithm, analysis, and applications//Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington, USA, 2010: 603-612

[31] Zhong C, Malinen M, Miao D, et al. A fast minimum spanning tree algorithm based on K-means. Information Sciences, 2015, 295(C): 1-17

[32] Callahan P-B, Kosaraju S-R. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. Journal of the ACM, 1992, 42(1): 67-90

[33] Zhu Q, Fan X, Feng J. Outlier detection based on k-neighborhood MST//Proceedings of the 15th IEEE International Conference on Information Reuse and Integration. San Francisco, USA, 2014: 718-724

[34] Qiao Y, Wang S. Fast clustering by synchronization on large sample. Computer Engineering & Applications, 2016, 52(23): 159-166

[35] Liang Y, Li J, Guo K. Lossless compression of hyperspectral

《计算机学报》

images using hybrid context prediction. Optics Express, 2012, 20(7): 8199

[36] Shao J, Böhm C, Yang Q, et al. Synchronization based outlier detection//Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases. Barcelona, Spain, 2010: 245-260

[37] Callahan P-B, Kosaraju S-R. Faster algorithms for some geometric graph problems in higher dimensions//Proceedings of the ACM-SIAM Symposium on Discrete Algorithms. Austin, USA, 1993: 291-300

[38] Chan T M. Well-separated pair decomposition in linear time? Information Processing Letters, 2008, 107(5): 138-141

[39] Wang X, Wang X, Wilkes D M. A divide-and-conquer approach for minimum spanning tree-based clustering. IEEE Transactions on Knowledge and Data Engineering, 2009, 21(7): 945-958

ZHU Li, born in 1968, Ph. D., associate professor. His research interests include machine learning, digital media understanding.

QIU Yuan-Yuan, born in 1988, M. S. candidate. Her main interests include data mining, machine learing.

YU Shuai, born in 1991, M. S. candidate. His main interest is machine learing.

YUAN Sheng, born in 1976, Ph. D., lecturer. His main interest is digital media understanding.

《计算机学报》

Background

This research belongs to the National Science Foundation of China project—"Machine Learning Based Localization for Mobile Robot in Unknown Outdoor Environment Research"(61473220), and is also supported by the Shaanxi Province Science Foundation under Grant No. S2015YFJM2129.

Recent development of database storage technologies produces a lot of community services. One side, the amount of datasets are explosively increasing. On the other side, people are more and more desire to analyze and aquire valuable potential information from the amount of dataset. However, current traditional technologies have failed to mining potential knowledge from big datasets.

Data Mining is a top issue in many areas, such as the current database, artifical intelligence, computer science and knowledge discovery in database. Data mining is amied at finding highly valuable knowledeg from mass data. The main tasks of data mining have regression, association rule learing, classification, clustering and Outlier detection.

Amining to discover observations that deviate from other observations so much as to arouse suspicions that they are generated by a different mechanism, outlier detection has become an important data mining task.

State-of-the-art k-nearest neighbors (KNN) based outlier detection algorithms, such as many distance-based and dendity-based methods. However, there are many aspects concerning the algorithms be worth further exploration. Firstly, it has been observed that k-nearest neighbors based outlier detection methods are senxitive to the parameter k. Secondly, for large datasets, the time complexity in the search of the KNN has to be improved significantly. Finally, in high dimensional space, the notion of these outlier definitions is whether still meaningful.

To meet these challenges, in this paper, we propose a new fast kNN-based MST outlier detection method (FKNNT). With respect to other methods, the time complexity of both computing kNN and creating tree are $O(kN)$ and $O(N\log N)$, respectively. The experiments show that FKNNT algorithm can detect both local outliers and global outliers without inputting the number of clusters from users.