高技术通讯 2022 年 第32卷 第7期:674-686

doi:10.3772/j.issn.1002-0470.2022.07.002

FNOD:基于近邻差波动因子的离群点检测算法①

张忠平②*,**,*** 邓 禹③* 刘伟雄* 张玉停*

(*燕山大学信息科学与工程学院 秦皇岛 066004)

(**河北省计算机虚拟技术与系统集成重点实验室 秦皇岛 066004)

(***河北省软件工程重点实验室 秦皇岛 066004)

摘 要 针对现存离群点检测算法和剪枝方法存在算法精确度较低和剪枝程度小的问题,提出了一种基于近邻差波动因子的离群点检测方法。该方法首先依据离群点的相互k近邻(MUN)点数远小于参数k这一特点,提出了一种基于近邻关系的剪枝方法;然后提出近邻差的概念来刻画数据对象与其邻居点的分布特征,在变化的参数k下,离群点和正常点的近邻差的变化不同;最后采用近邻差波动衡量每个数据点的离群程度,进而检测出离群点。人工数据集和真实数据集下的实验结果表明,该算法能够有效且较为全面地检测出离群点。

关键词 数据挖掘;离群点;剪枝;相互k近邻(MUN);近邻差波动因子

0 引 言

离群点是数据集中偏离大部分数据的数据,由于偏离其他数据太多,这些数据的偏离可能并非由随机因素产生,而是产生于完全不同的机制[1]。离群点检测是一个长期存在于数据挖掘领域的基本主题,其旨在消除噪音和干扰或发现潜在的、有价值的信息。离群点检测在欺诈检测[2]、垃圾邮件检测[3]、交通异常检测[4]、入侵检测[5-6]、医疗与公共卫生安全监测[7]等领域有着广阔的应用前景。

目前,国内外许多学者在离群点检测领域的研究十分活跃,提出了大量优秀的离群点检测方法,这些方法大致可分为基于分布的、基于距离的、基于密度的、基于聚类的和基于深度的5种[8]。近年来也有一些新方法被提出,文献[9]和文献[10]通过构建k近邻(k nearest neighbors, KNN)图,利用图的标签传播和随机游走的方法检测离群点。

早期基于分布的方法通过判断数据点偏离正常分布模式的程度判断离群点[11],但该方法不适用于高维数据集,特别是数据集数据分布未知的情况。

为了改进基于分布方法的缺陷,文献[12]提出了一种基于距离的离群点检测方法。该方法通过计算数据点到其k近邻点的平均距离,选取平均距离最大的n个点作为离群点,但由于该方法使用的是全局阈值,未考虑局部环境的影响,因此难以检测局部离群点。

为解决基于距离方法的问题,文献[13]定义了局部离群因子(local outlier factor, LOF)算法,该方法将数据点与其k近邻点平均距离的倒数看作局部密度因子,密度因子越小,该点是离群点的概率越高。LOF算法作为经典的局部离群点检测算法已被广泛研究,但存在精确度较低、参数敏感等问题,很多研究学者都对此算法提出了改进。INFLO(influence outlierness)算法[14]、NOF(natural outlier factor)算法[15]分别在局部密度因子中引入了反向k近邻

① 国家自然科学基金面上项目(61972334),河北省创新能力提升计划(20557640D)和四达铁路智能图像工件识别(x2021134)项目资助。

② 男,1972年生,博士,教授;研究方向:大数据,数据挖掘,半结构化数据;E-mail:zpzhang@ysu.edu.cn。

③ 通信作者,E-mail:962058456@qq.com。

(收稿日期:2021-05-26)

(reverse k nearest neighborhood, RNN)和相互 k 近邻(mutal k neighborhood, MUN)提高了算法的性能。RDOS(reverse shared outlier density)算法[16]将局部密度因子替换为效果更好的核密度,同时引入反向 k 近邻、相互 k 近邻和共享 k 近邻提高了检测精度。文献[17]提出的 DDPOS(density and distance parameters outlier factor scores)算法利用反向 k 近邻来排除掉部分边界点的干扰,类似的算法还有 RDOF(relative density outlier factor)[18] 和 LCDO(local correlation dimension outlier)[19]。

文献[20]引入反向近邻关系,提出了一种将 k 近邻点和反向 k 近邻点数的比值作为离群因子的快速离群点检测方法。文献[21]提出了一种基于不稳定因子的离群点检测方法 INS(INStability factor),通过计算虚拟 k 近邻点的稳定性判断离群点。以上方法都需要遍历数据集中所有点,然后再对离群因子进行排序来确定离群点,但是此类方法对计算位于聚类区域的正常点是没有必要的。

为解决上述问题,文献[22]通过 DBSCAN(density-based spatial clustering of applications with noise)聚类的方法剪枝位于聚类中心的正常点,只计算保留下的可疑点,减少了后续离群点检测算法的计算量。文献[23]提出了一种基于累积权熵空间聚类(subspace outlier detection cumulative holoentropy, SODCH)的离群点检测算法,利用 k-means 对子空间进行聚类,根据累积权熵检测离群点,大幅提高了算法处理数据的力度。文献[24]利用相互 k 近邻关系提出了一种无参数的聚类方法,减少了算法对参数 k 的依赖。文献[25]提出了一种基于聚类因子和相互密度的离群点检测算法,通过引入相互 k 近邻关系提高了算法的精确度。

通过对上述算法的总结可以看出,很多改进算法都在原有的离群因子中引入了更复杂的近邻关系,但没有改变离群因子的计算方法,且未考虑变化的参数 k 对离群点检测的影响。随着数据分布越来越复杂,这些离群点检测算法效果不够理想,因此,本文提出一种基于变化参数 k 的离群因子,为离群点检测提供了新的研究方向。基于聚类的剪枝方法对数据聚类程度要求较高,不能适应多种数据集,因此提出一种新的剪枝方法,也是一个新的研究方向。综上所述,本文引入相互 k 近邻关系,提出了一种基于近邻差波动因子的离群点检测算法(outlier detection algorithm based on fluctuation of nearest neighbor difference factor,FNOD)。该算法引入相互 k 近邻关系,首先提出一种近邻剪枝算法(neighbor pruning factor algorithm,NPF)剪枝正常点,保留离群度较高的可疑点,降低运算量,提升后续离群点检测算法的精确度;其次设计迭代减小的参数 k,随着 k 值减小,离群点的相互 k 近邻变化程度要小于正常点;最后利用数据点相互 k 近邻变化的波动程度刻画每个点的离群程度。

本文其余部分安排如下。第 1 节介绍相关工作,关于相关算法的各种定义。第 2 节对本文提出的算法进行阐述。第 3 节通过实验对所提算法的实效性进行验证。第 4 节对本文做出总结。

1 相关工作

下面简要介绍相互 k 近邻数搜索算法。

1.1 相关定义

定义 1 k 距离(k _ distance)[15]。数据点 p 的 k 距离表示点 p 和其第 k 个近邻点 o 之间的欧氏距离,用 $k\_dist(p)$ 表示。点 p 至多存在 $(k-1)$ 个 o' 点满足 $dist(p, o') \leqslant dist(p, o)$。

定义 2 k 近邻[15]。点 p 把点 q 看作 k 近邻点。数据点 p 的 k 近邻集合表示为 $KNN(p) \subseteq D$,要求点 q 满足 $dist(p, q) \leqslant k\_dist(p)$,定义为式(1)。

$$KNN(p) = \{q \in D \mid dist(p,q) \leqslant k\_dist(p)\} \tag{1}$$

其中,D 表示数据集,p、q、o 为数据集 D 中的数据点,$dist(p,q)$ 表示点 p、q 之间的欧式距离,k 为正整数。

定义 3 反向 k 近邻[15]。数据点 p 被点 q 看作 k 近邻点,数据点 p 的反向 k 近邻集合用 $RNN(p) \subseteq D$ 表示,定义为式(2)。

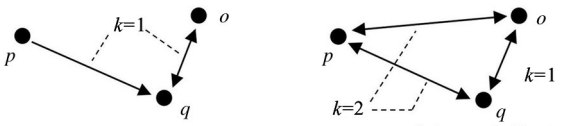
$$RNN(p) = \{q \in D \mid p \in KNN(q)\} \tag{2}$$

定义 4 相互 k 近邻。数据点 p 的相互 k 近邻用 $MUN(p) \subseteq D$ 表示,要求在当前 k 值下,点 p 是点

q 的 k 近邻,同时点 p 也是点 q 的反向 k 近邻,定义为式(3)。

$$MUN(p) = \{q \in D \mid q \in (KNN(p) \cap RNN(q))\} \tag{3}$$

相互 k 近邻建立在 k 近邻的基础上,要求两点互相将对方看作 k 近邻,受距离和参数 k 的影响,当一个点距离其他点越远时,其越不容易被其他点选为 k 近邻点,也就很难形成相互 k 近邻关系。图 1 中单向箭头表示 k 近邻,双向箭头表示相互 k 近邻。$k = 1$ 时,点 q 是点 p 的 k 近邻,由于点 q 与点 o 的距离小于点 q 与点 p 的距离,所以点 q 的 k 近邻是点 o, 同理点 o 的 k 近邻点是点 q, 因此点 q 与点 o 形成相互近邻关系,未与点 p 形成相互近邻关系。若使点 p 与点 o 和 q 形成相互近邻关系,只有扩大参数 k 值,使点 p 成为点 q 和 o 的 k 近邻点。如图 1(b)中 $k=2$ 时所示,点 p 成为点 o 和 q 的第 2 个k近邻点,此时点p形成了相互k近邻关系,所以相互 k 近邻关系受到距离和参数 k 的影响。



(a) k=1 时点 p 近邻关系　(b) k=2 时点 p 近邻关系

图 1　相互 k 近邻关系

图 2 显示了不同 k 值下,数据点相互 k 近邻的变化情况。图中虚线单向箭头代表 k 近邻关系,双向箭头代表相互 k 近邻关系(点 o 和 r 存在一条双向箭头),点 p 为离群点。

由于图中离群点 p 远离其他点,且参数 k 较小,所以点 $[o,s,r]$ 是点 p 的 k 近邻点,但点 p 不是点 $[o,s,r]$ 的 k 近邻点,因此点 $[o,s,r]$ 不会与点 p 形成相互近邻关系。$k=4$ 时点 p 仅与点 q 形成了相互近邻关系,而其他点均有 3 或 4 个相互 k 近邻点。

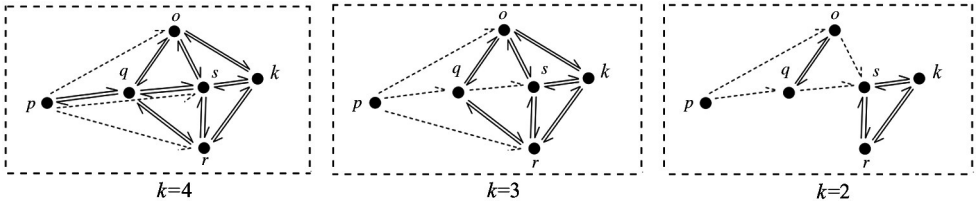


图 2　参数 k 变化对相互 k 近邻关系的影响

表 1 统计了图 2 中不同参数 k 下,部分数据点相互 k 近邻点的变化。当参数 k 递减时,数据点的 k 近邻点减少,所以数据点的相互 k 近邻点也会减少,相互近邻点开始发生变化。由于离群点 p 仅有一个相互 k 近邻点,因此随着参数 k 下降,最多发生一次相互 k 近邻变化。但位于聚类区域的点 $[q,o,s]$, 由于其相互 k 近邻点个数接近 k 近邻点个数,伴随 k 值下降,这些点会有多次相互 k 近邻变化。

根据图 2 和表 1 的描述可以总结出:(1)在某一参数 k 下,离群点的相互 k 近邻点个数要小于 k 近邻点个数;(2)在变化的近邻参数 k 下,离群点的相互 k 近邻点数相对稳定,不易发生变化。本文根据(1)设计一种基于相互 k 近邻关系的近邻剪枝方法,排除正常点;根据(2)设计一种在变化参数 k 下基于数据点相互 k 近邻波动的离群点检测算法。

表 1　数据点相互 k 近邻点个数

参数 k	p	q	o	s
$k=4$	1	4	4	4
$k=3$	0	2	3	3
$k=2$	0	1	1	2

1.2　相互 k 近邻数统计算法

根据 1.1 节描述的相关定义,统计相互 k 近邻数量的算法如算法 1 所示,首先根据输入的近邻参数 k 计算每个点的 k 近邻和反 k 近邻;再根据相互 k 近邻的定义,搜索每个数据点的相互 k 近邻;最后统计每个数据点的相互 k 近邻的数量作为输出。

在相互近邻数搜索(MUNn-Searching)算法中,$KNN(i)$代表数据点 i 的 k 近邻点集合,$RNN(i)$代表数据点 i 的反向 k 近邻集合,$MUN(i)$代表数据点

算法 1 相互近邻数搜索(MUNn-Searching)算法

输入:数据集 D,近邻参数 k;

输出:每个数据点相互近邻数量 $SUM(i)$。

MUNn-Searching(D, k)

BEGIN

1. 初始化 $KNN(i)=\varnothing, RNN(i)=\varnothing, MUN(i)=\varnothing$
2. 搜索数据集中每个点 i 的 k 近邻点,即 $KNN(i)$
3. FOR D 中每一个数据点 j, $i \neq j$
4. IF $i \in KNN(j)$
5. $RNN(i) = =j$
6. $MUN(i)=KNN(i) \cup RNN(i)$
7. ELSE CONTINUE
8. $SUM(i)=MUN(i)$ 中数据点的数量
9. 输出 $SUM(i)$

END

i 的相互 k 近邻点集合。MUNn-Searching 算法的时间复杂度源于利用 KD 树计算 k 近邻点的过程,时间复杂度为 $O(n \cdot \log n)$,n 为数据集中数据点的个数。

2 近邻差波动算法

2.1 近邻剪枝算法思想

离群点检测算法中对聚类区域中正常点的计算是没有必要的,为减少此类点的计算,本文利用相互 k 近邻提出了一种近邻剪枝算法,能够剪枝大部分聚类区域的正常点,减少后续离群点检测算法的计算量。

根据定义 2 k 近邻、定义 4 相互 k 近邻和图 2可知,由于离群的数据点距离聚类区域的数据点较远,难以被聚类区域的点看作 k 近邻点,因此相互 k 近邻点的数量远远小于 k 近邻点数量,反之位于聚类区域点的相互 k 近邻点数量接近 k 近邻点数量。因此,聚类区域点的 k 近邻点数与相互 k 近邻点数的比值接近 1,稀疏区域点的 k 近邻点数与相互 k 近邻点数的比值远远大于 1。据此特点可以使大部分正常点被剪枝。

定义 5 近邻剪枝因子(NPF)。$NPF(p)$ 是数据点 p 在某一 k 值下 k 近邻个数和相互 k 近邻点个数的比值,定义为式(4)。

$$NPF(p)=\frac{|KNN_n(p)|+1}{|MUN_n(p)|+1} \tag{4}$$

式中,$KNN_n(p)$ 表示数据点的 k 近邻点个数,等于近邻参数 k, $MUN_n(p)$ 表示数据点 p 的相互 k 近邻点个数。将分母 $MUN_n(p)$ 加 1 是为了防止出现分母为 0,即数据点 p 没有相互 k 近邻点的特殊情况,分子 $KNN_n(p)$ 加 1 是为了降低 $MUN_n(p)$ 加 1 带来的影响。

2.2 近邻剪枝算法

根据 2.1 节中描述及相关定义,近邻剪枝算法如算法 2 所示,该算法首先根据输入的参数 k 统计数据点 k 近邻点和相互 k 近邻点个数,随后计算数据点的近邻剪枝因子 NPF,最后对所有数据点按 NPF 值降序排序,保留最大的前 pt 个数据点作为可疑点。

算法 2 NPF 算法

输入:数据集 D, 近邻参数 k, 剪枝阈值 pt;

输出:pt 个可疑点。

$NPF(D, k, pt)$

BEGIN

1. 初始化 $KNN_n(p)=\varnothing, MUN_n(p)=\varnothing$
2. 根据算法 1 计算每个数据点相互 k 近邻点数,即 $MUN_n(i)$
3. FOR D 中每一个数据点 i
4. 根据式(4)计算 $NPF(i)$
5. 按照 $NPF(i)$ 值,对所有数据点降序排序
6. 输出前 pt 个可疑点

END

剪枝后保留的可疑点的相互近邻数很少,能够减少相互近邻随参数 k 变化而变化的次数,有利于提高基于近邻差波动的离群点检测算法精度。NPF 算法的时间复杂度主要来源于利用 KD 树计算 k 近邻点的过程,时间复杂度为 $O(n \cdot \log n)$,n 为数据集中数据点的个数。相比基于聚类的剪枝算法[22],具有剪枝程度大、速度快的优点。

2.3 近邻差波动因子算法思想

本文提出了一种基于近邻差波动因子的离群点检测方法 FNOD,算法思想源于相互 k 近邻关系。聚类区域的点容易形成相互 k 近邻,且相互 k 近邻易随参数 k 减小而减少,相反,由于离群点远离聚类

区域,相互 k 近邻点很少,所以相互 k 近邻不易随 k 值减小而减少。

图 3 所示的数据集中,双向虚线代表相互 k 近邻关系,A 代表一个小聚类,点 B 为局部离群点,点 C、D、E 为全局离群点。虚线表示在当前 k 值下存在的相互 k 近邻关系。对比发现在 $k=[10,9,\cdots,2]$ 过程中全局离群点 C、D、E 的相互 k 近邻点不会变化;$k=[1,2,\cdots,5]$ 过程中局部离群点 B 的相互 k 近邻点不会变化;聚类 A 中的点在 k 每次变化后都会发生变化。离群程度越高的点越不易发生相互近邻的变化,本文据此设置一个迭代减小的参数 k,随着 k 值减小,数据点相互 k 近邻点发生变化,离群程度越高的点,其相互 k 近邻点数越少,甚至为 0,不易变化。本文设计用波动的方式描述数据点相互近邻变化程度,波动越小的点是离群点的概率越大。

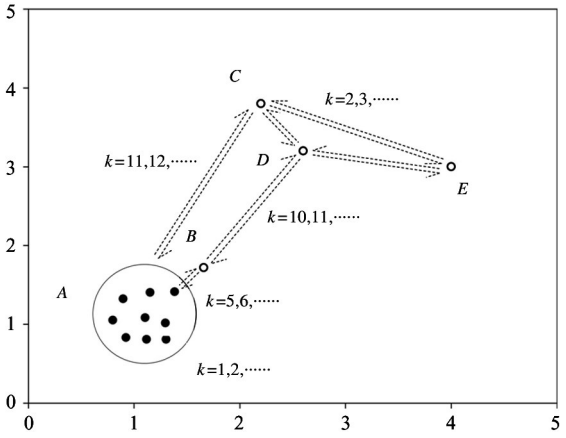


图 3 FNOD 算法思想示意图

2.4 相关定义

本节详细介绍 FNOD 算法及其相关定义,其中,D 表示数据集;p 代表数据集中的数据点;i 代表参数 k 的迭代次数;k_i 代表第 i 次迭代时的 k 值;$MUN_i(p)$ 表示数据点 p 在第 i 次迭代时的相互 k 近邻点个数;n 为正整数表示初始 k 值。

定义 6 近邻差 $d(p)$。$d_i(p)$ 是数据点 p 第 i 次迭代后相互 k 近邻点数的差,用来描述相互 k 近邻的变化程度,定义为式(5)。式中 $MUN_{i-1}(p)$ 和 $MUN_i(p)$ 分别为第 $i-1$ 和第 i 次参数 k 变化后,数据点 p 的相互 k 近邻点个数。n 为数据点个数,本文将一个 $d_i(p)$ 看作一次波动。

$$d_i(p) = | MUN_{i-1}(p) - MUN_i(p),\ i \in (1,n-1) | \tag{5}$$

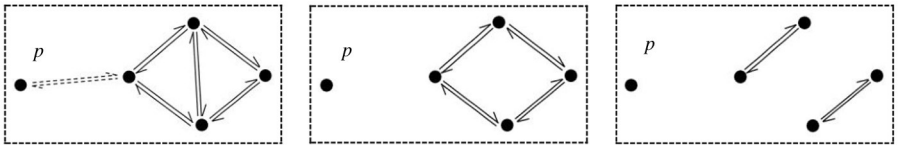
定义 7 稳态 ST(steady)。稳态表示数据点在相邻两次 k 值变化过程中近邻波动差 $d_i(p)=0$ 的状态。

本文定义 $ST=0$,是因离群点的相互 k 近邻点较少,因此在近邻参数 k 下降过程中,离群点的近邻差不容易发生变化,即 $d_i(p)=0$,相反正常点会随着参数 k 的变化而变化。统计数据点在整个参数 k 值迭代过程中 $d_i(p)$ 每次的变化情况是否贴近 $d_i(p)=0$,因此设置整体稳态 $ST=0$。

定义 8 近邻下降率 α。参数 k 的下降速度定义为近邻下降率。定义为式(6)。

$$k = k - \alpha i (k \geqslant 1) \tag{6}$$

近邻下降率 α 直接决定算法迭代的次数,α 值越大,k 值减小的速度越快,算法迭代的次数越少,但会降低离群点检测的精确度。如图 4 所示,$k=3$ 时点 p 的相互近邻点数为 1;$k=2$ 时,点 p 的相互 k 近邻点减少到 0;$k=1$ 时,点 p 的相互 k 近邻点仍为 0。因此 k 值从 3 到 2 形成了一次波动,从 2 到 1 是一个稳态。稳态有利于增加离群程度,而波动会降低离群程度。若参数 k 直接从 3 降到 1,会丢失一个稳态,降低离群程度,因此本文设置近邻下降率 $\alpha=1$。



(a) k=3 时相互 k 近邻　(b) k=2 时相互 k 近邻　(c) k=1 时相互 k 近邻

图 4 近邻下降率 α 对稳态的影响

此外设置 $k=1$ 时停止迭代,此时每个数据点只有 1 个 k 近邻点,每个数据点至多只有 1 个相互 k

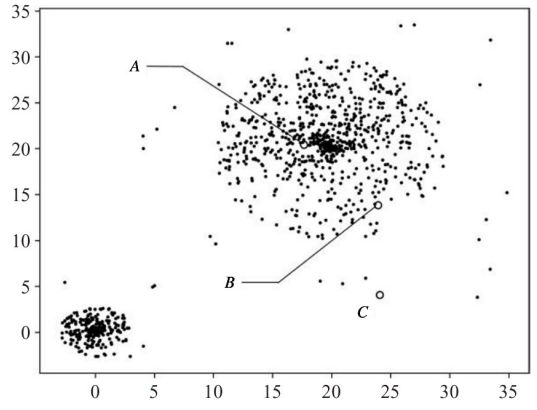
近邻点,减少相互 k 近邻点数由 1 到 0 带来的必然波动。

定义 9 近邻差波动因子 $FNOD(p)$ 描述数据点 p 随 k 值下降,点 p 的近邻波动差 $d(p)$ 的变化情况,参考统计学中的标准差,定义为式(7)。

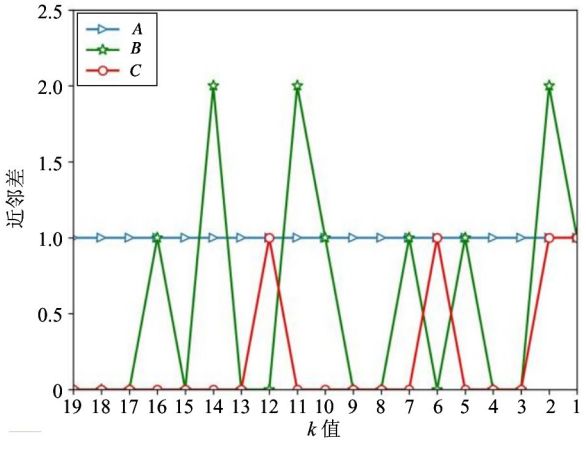
$$FNOD(p) = \sqrt{\frac{\sum_{i=1}^{n-1}(d_i(p) - ST)^2}{n-1}} \tag{7}$$

标准差描述的是各个阶段的数据相对均值的波动情况,本文中要求统计数据点相对于稳态的波动情况,因此,用稳态 $ST=0$ 替换均值。本文设置初始参数 k,同时参数 k 按近邻下降率 $\alpha=1$ 迭代,直到 $k=1$,共迭代 $n-1$ 次。$FNOD(p)$ 波动值越小,说明此点是离群点的概率越高。

在图 5 (a)所示的人工数据集中,验证了随着参数 k 减小,不同类型的点的近邻差波动不同。图中数据点 A 和 B 为正常点、C 为离群点,它们的近邻差波动如图 5(b)所示,图中横坐标轴为参数 k,纵坐标轴为近邻波动差,根据近邻波动因子,定义 $ST=0$ 为横坐标轴。计算图中折线相对于横坐标轴的波动情况,可得 $FNOD(A)=0.95$、$FNOD(B)=1.0$、$FNOD(C)=0.46$。点 C 的近邻波动因子远小于其余两点,因此点 C 为离群点。通过观察图 5(b)也可以看出,离群点 C 的波动程度很小,只有 4 次 $d(p)=1$ 的波动,相反正常点 B 有 18 次 $d(p)=1$,点 A 拥有多次 $d(p)=1$ 和 $d(p)=2$ 的波动。



(a) 正常数据和离群数据样本



(b) 点 C、B、A 近邻差波动图

图 5 正常数据和离群数据样本

2.5 FNOD 算法描述

本文提出的一种基于近邻差波动的离群点检测算法 FNOD,输入数据集 D 和参数 k,输出离群点。首先对数据集进行剪枝,排除位于聚类区域的正常点,留下可疑点;随后在迭代减小的参数 k 下,计算可疑点的近邻波动因子,按从小到大排序,前 n 个点即为离群点。FNOD 算法代码如算法 3 所示。

算法 3 FNOD 算法

输入:数据集 D,参数 k;

输出:离群点集合 $S_{outlier}$。

FNOD(D, k)

BEGIN

1. 初始化 $\alpha=1$,$ST=0$,$FNOD(O_{doubt})=\varnothing$, $S_{outlier}=\varnothing$
2. 利用算法 2 对数据集 D 进行剪枝,留下可疑点集合 O_{doubt}
3. FOR 每个数据点 $p \in O_{doubt}$
4. While $k \geqslant 1$
5. 利用算法 1 计算 p 点的相互近邻数 $MUN(p_k)$
6. $MUN(p)=MUN(p) \cup MUN(p_k)$
7. $k=k-\alpha$
8. 根据式(5)计算近邻差 $d_i(p)$
9. 根据式(7)计算点的近邻波动因子 $FNOD(p)$
10. $FNOD(O_{doubt})=FNOD(O_{doubt}) \cup FNOD(p)$
11. $FNOD(O_{doubt})$ 中的点升序排序
12. $S_{outlier}=FNOD(O_{doubt})$ 中的前 n 个点
13. 输出 $S_{outlier}$

END

算法 3 中 O_{doubt} 存放剪枝后保留的可疑点,$MUN(p_k)$ 为当前 k 值下,数据点 p 的相互 k 近邻点数,集合 $MUN(p)$ 存放点 p 每次 k 值变化后的相互近邻数,集合 $FNOD(O_{doubt})$ 存放所有可疑点的近邻

差波动因子,算法最终输出波动最小的前 n 个点作为离群点。FNOD 算法的时间复杂度主要来源于两方面:(1)使用 KD 树寻找相互 k 近邻点个数,时间复杂度为 $O(n \cdot \log n)$,n 为剪枝后可疑点个数;(2)参数 k 迭代减小,时间复杂度为 $O(k)$。算法总体的时间复杂度为 $O(k \cdot n \cdot \log n)$。

3 实验评估

为检测 FNOD 算法的性能,在人工数据集和真实数据集下进行实验,同时将 FNOD 算法与较为流行的——利用其他近邻关系的 LOF[13]、INFLO[14]、NOF[20]、NOF2[15]算法和同样迭代 k 值的 INS[21]算法进行对比。FNOD 算法和 INS 算法的 k 值分别对应初始 k 值和截止 k 值。实验环境如表 2 所示,实验结果使用 Python 进行可视化处理。

表 2 实验环境

软硬件环境	参数
处理器	Inter Core i7-9750H 2.60 GHz
内存	8 GB
硬盘	1 TB
编译语言	Python 3.8.1
开发工具	Pycharm 2020.2.3
可视化工具	Python 3.8.1

3.1 实验指标

在大多数实际应用中,相比大量存在的正常数据,含有异常值的数据显得非常稀有,数据分布存在极端不平衡现象。因此,常见的准确率等指标不能客观评价算法性能。为评估离群点检测的性能,针对人工数据集,采用精确率(precision,Pr)指标进行评估。精确率定义如式(8)所示。

$$Pr = \frac{TP}{TP + FP} \tag{8}$$

式中,TP 表示算法检测到的真实离群点数量,FP 表示算法错误地将正常点判断为离群点的数量。精确率数值越大,算法性能越好。

针对真实数据集,采用 $F1$ 曲线和接受者操作特性曲线(receiver operating characteristic curve,ROC)的曲线下面积(area under curve,AUC)进行评估。$F1$ 值是衡量二分类模型的一种常用指标,其同时兼顾了精确率和召回率两个指标。$F1$ 值越接近 1,算法性能越好。$F1$ 值定义如式(9)所示。

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{9}$$

ROC 是以真阳性率(TPR)为纵坐标,假阳性率(FPR)为横坐标绘制的曲线,ROC 曲线描绘了两者的相对权衡。真阳性率和假阳性率的定义如式(10)、(11)所示。AUC 可以直观地展现出算法的性能,值取 0 到 1 之间,越大的 AUC 值意味着算法有更大的概率将离群点排在正常点之前,所以 AUC 值越大越好,小于 0.5 意味着算法不可用。

$$TPR = \frac{TP}{TP + FN} \tag{10}$$

$$FPR = \frac{FP}{FP + TN} \tag{11}$$

3.2 人工数据集

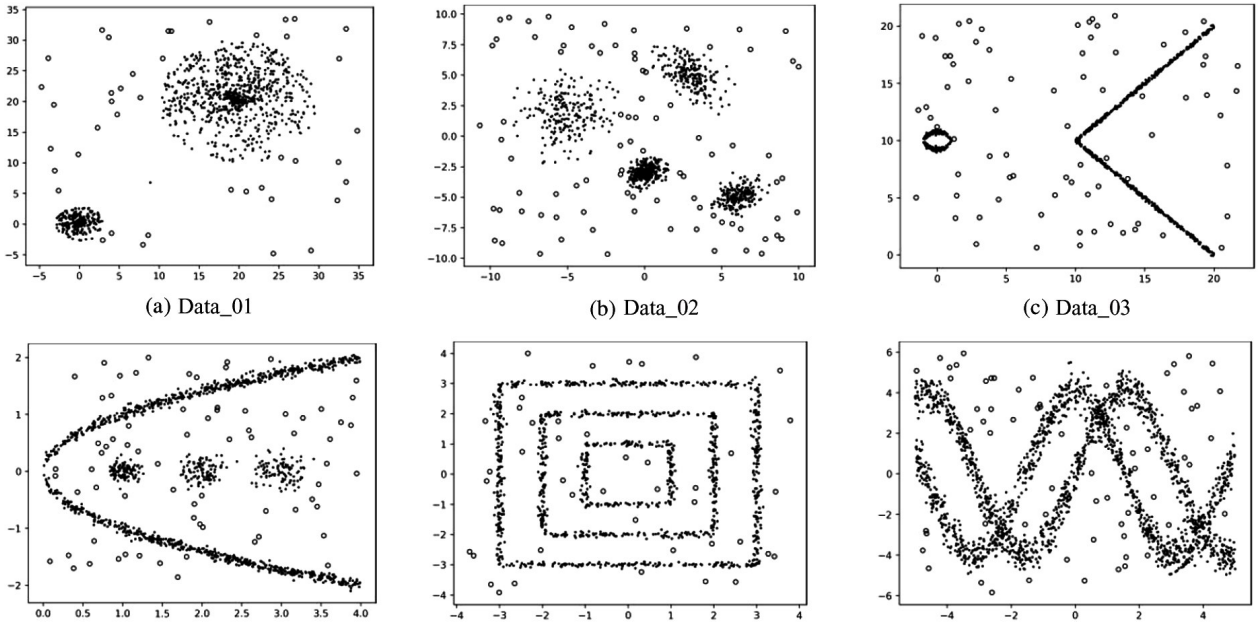
为测试本文算法在各种复杂数据分布下的性能,实验在图 6 所示的 6 种二维人工数据集 Data_01 ~ Data_06 中进行,图 6 中“o”代表离群点。每个数据集的属性特征如表 3 所示。

表 3 人工数据集特征

数据集	数据点数量	离群点数量	离群点占比
Data_01	1043	43	4.1%
Data_02	1000	85	8.5%
Data_03	876	77	8.8%
Data_04	1372	72	5.2%
Data_05	1256	43	3.4%
Data_06	2042	64	3.1%

图 7 展示了本文 FNOD 算法和其余 5 个算法在 6 个人工数据集上精确率随 k 值变化的过程,横坐标为 k 值,纵坐标为精确率 Pr。

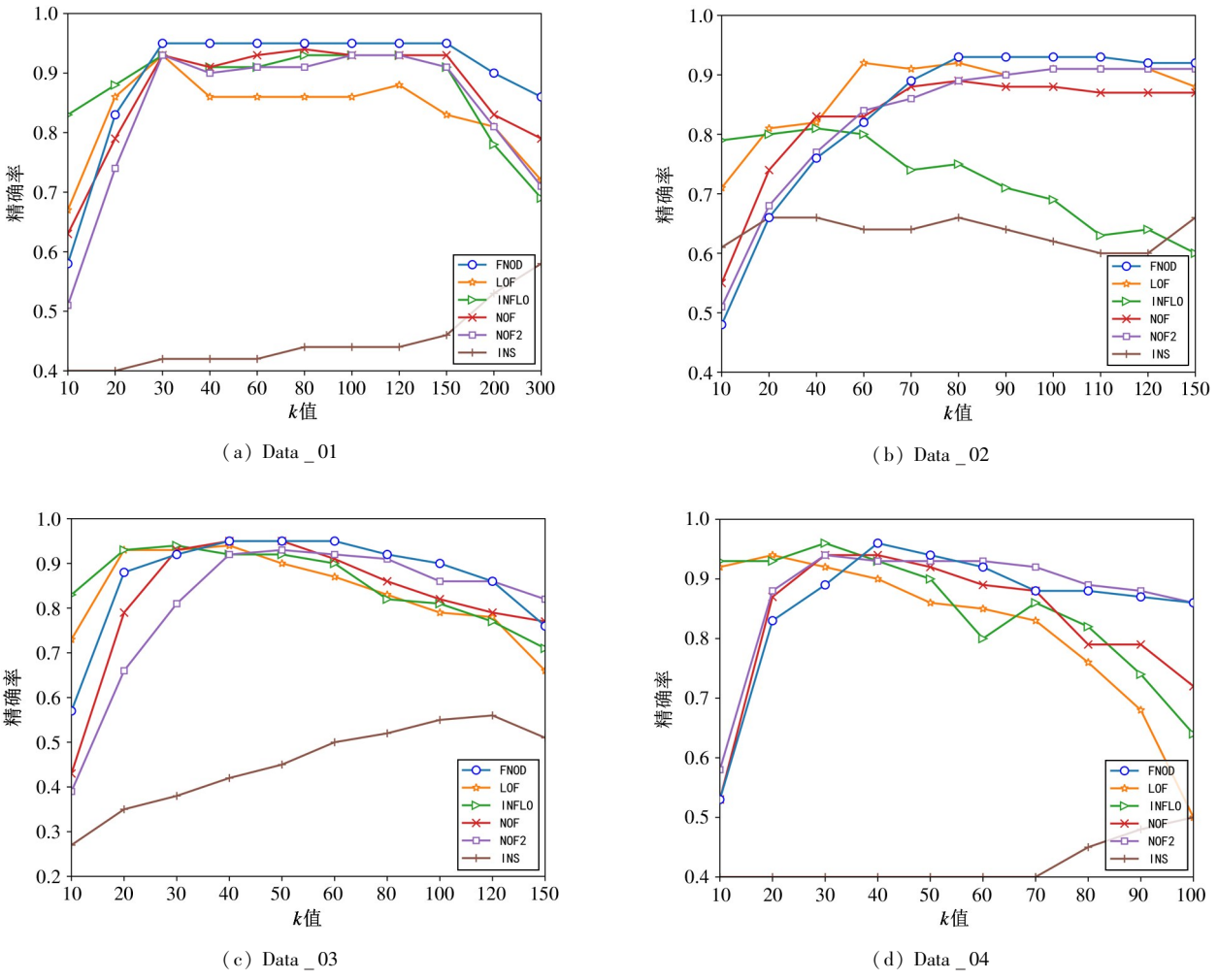
随着参数 k 不断增大,FNOD 算法的精准率不断上升,最终大于或等于其余算法。特别是在 Data_01、Data_05、Data_06 中,FNOD 算法的峰值超出了其余算法的峰值,分别达到 0.95、0.98、0.86。此外 FNOD 算法在 Data_02、Data_03、Data_06 数据集中,曲线变化程度接近且高于同样使用相互k近



(a) Data_01　(b) Data_02　(c) Data_03

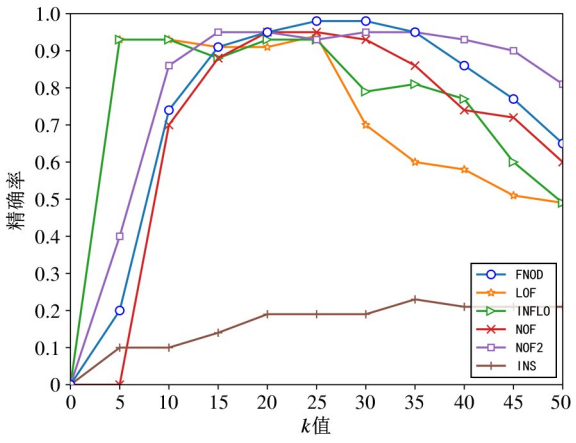
(d) Data_04　(e) Data_05　(f) Data_06

图 6　6 种人工数据集分布

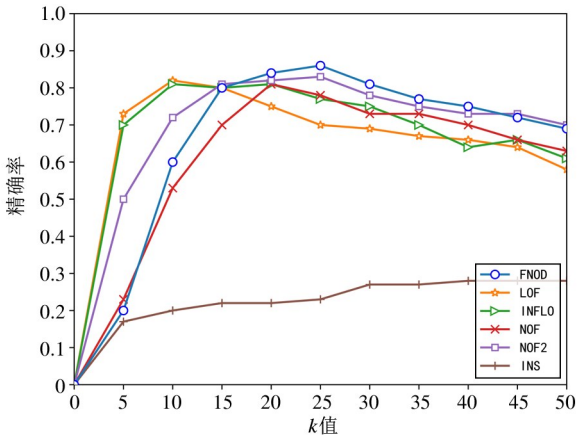


(a) Data _ 01　(b) Data _ 02

(c) Data _ 03　(d) Data _ 04



（e）Data _ 05



（f）Data _ 06

图 7　人工数据集下不同算法的精确率变化曲线

邻关系的 NOF2 算法。随着 k 值不断增大，实验中所有算法的精确率有所下降，但本文 FNOD 算法的下降程度要低于其余算法，特别是在 Data _ 01 数据集中初始参数 k 从 150 增加到 300 的过程中 LOF、INFLO、NOF 和 NOF2 的下降率分别为 0.05、0.11、0.07 和 0.10，而 FNOD 的下降率为 0.04，这是因为随着参数 k 增大，利用距离或密度的方法会不断获取一个较远的近邻点，这些方法依赖 k 近邻点的距离计算离群因子，因此不断加入距离较大的 k 近邻点不利于算法的精确度，而本文 FNOD 算法用数量计算离群程度，受到的影响较小。

由于 INS 算法相比于其余算法需要更大的 k 值，所以图 7 中没有体现 INS 算法的最佳精确率，将 INS 算法的最佳精确率放置在表 4 中。

表 4　人工数据集精确率

数据集	FNOD	LOF	INFLO	NOF	NOF2	INS
Data _ 01	**0.95**	0.93	0.93	**0.94**	0.93	0.86
Data _ 02	**0.93**	0.92	0.81	0.88	**0.92**	0.65
Data _ 03	**0.95**	0.94	0.94	**0.95**	0.93	0.93
Data _ 04	**0.96**	0.94	**0.96**	0.94	0.93	0.83
Data _ 05	**0.98**	0.94	0.93	0.95	**0.95**	0.88
Data _ 06	**0.86**	0.82	0.81	0.81	**0.83**	0.78

通过总结图 7 和表 4 的数据，FNOD 算法在 Data _ 02、Data _ 01、Data _ 03、Data _ 06 表现最优，Data _ 04 和 Data _ 05 数据集中数据聚类不明显，所以表现略低于 NOF2 算法。由于 FNOD 算法需要统计多次参数 k 变化下的数据，当初始 k 值取到 10 和 20 时，算法只能获得 9 和 19 次的波动数据，因此 FNOD 算法在初始 k 值较小的情况下精确率较低。随着初始 k 值增加，FNOD 算法获取波动的数据增多，算法的精确度提高。

3.3　真实数据集

为较为全面地检测FNOD算法的真实性能，本文实验选择在 6 个真实数据集 Ecoli、Lonoshpere、Lymphography、PenDigits、Stamps 和 Wdbc 中进行，6 个真实数据集均来自 UCI 数据库，表 5 展示了真实数据集的数据特征。

表 5　真实数据集特征

真实数据集	数据点数量	维数	离群点数量	离群点占比
Ecoli	168	7	25	14.8%
Lonoshpere	351	32	126	35.8%
Lymphography	148	47	6	4.0%
PenDigits	1641	17	20	1.2%
Stamps	315	10	6	1.9%
Wdbc	390	30	27	6.9%

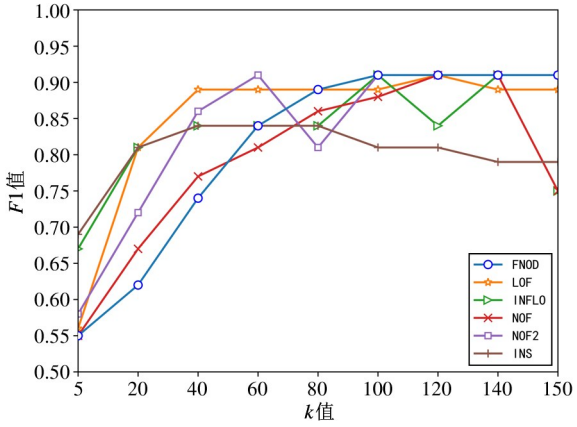
表 6 展示了真实数据集下 FNOD 算法和其余 5 个算法的 AUC 值，同时标注出每个数据集中排名前 2 的算法。FNOD 算法在每个真实数据集上的 AUC 值均属于表现最优的前 2 个算法。特别是在 Ecoli 和 PenDigits 中，FNOD 的 AUC 值达到了 0.96，超过

同样使用相互近邻关系的 NOF2 算法的 0.91 和使用变化 k 值的 INS 算法的 0.78。在数据量较大的 PenDigits 和离群点数较多的 Lonoshpere 中,FNOD 算法同样保持了较好的检测效果。面对维数较高的数据集 Lymphography 时,FNOD 算法表现不佳,仅为 0.80,但也接近其余几种算法的最佳值。整体上看 FNOD 算法 AUC 值大于同样使用相互 k 近邻的 NOF2 算法。

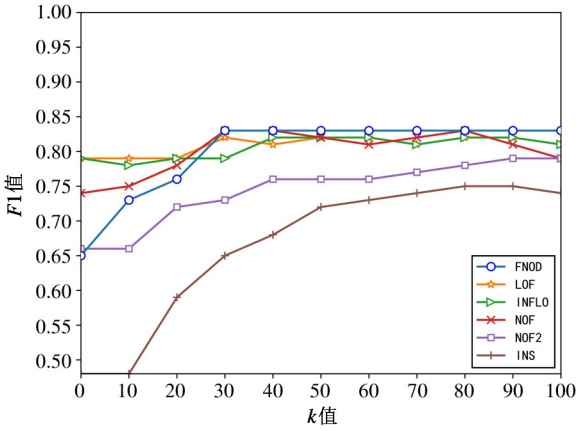
表 6　真实数据集 AUC 值

数据集	FNOD	LOF	INFLO	NOF	NOF2	INS
Ecoli	**0.96**	**0.94**	0.90	0.82	0.94	0.78
Lonoshpere	**0.86**	0.83	**0.93**	0.85	0.82	0.72
Lymphography	0.80	**0.88**	0.80	**0.90**	0.80	0.28
PenDigits	**0.96**	0.89	**0.95**	0.92	0.91	0.00
Stamps	**0.90**	0.80	**0.93**	0.85	0.87	0.00
Wdbc	**0.78**	0.77	0.69	0.73	**0.75**	0.71

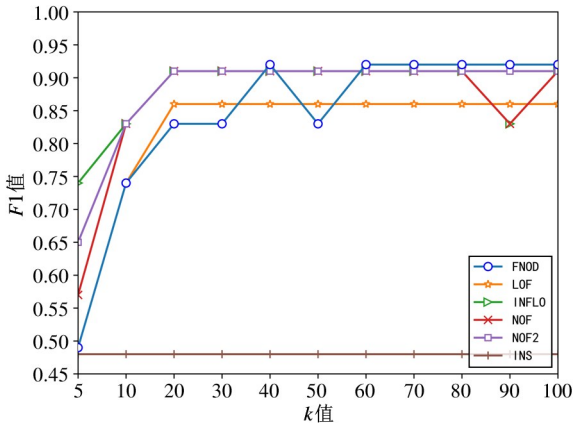
图 8 展示了各个算法 $F1$ 值随参数 k 变化的曲线。通过观察图像可以得到,在各个数据集中,除在 PenDigits 和 Stamps 下的 INS 算法 $F1$ 值为 0.5,其余每个算法在各个数据集下的 $F1$ 曲线都接近一条逐渐上升的拟合曲线,表明各算法都具备一定的二分类效果。



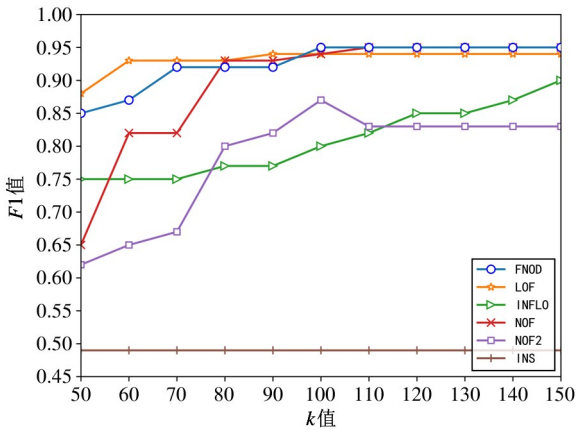
(a) Ecoli $F1$ 值变化曲线



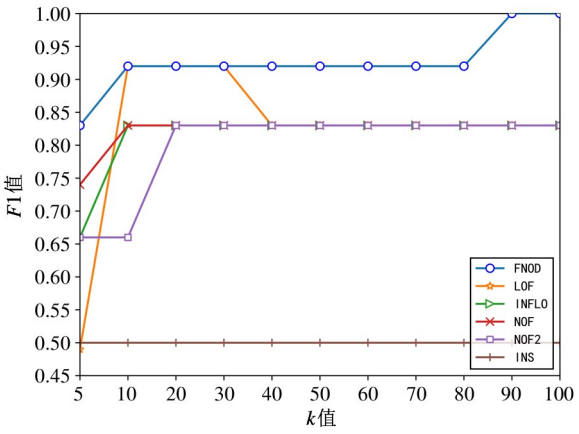
(b) Lonoshpere $F1$ 值变化曲线



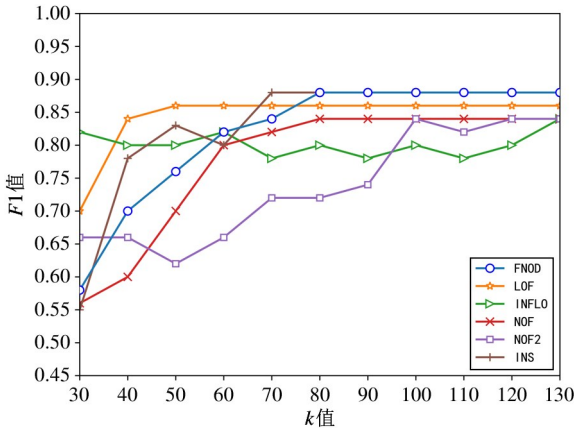
(c) Lymphography $F1$ 值变化曲线



(d) PenDigits $F1$ 值变化曲线



（e）Stamps *F*1 值变化曲线



（f）Wdbe *F*1 值变化曲线

图 8　真实数据集下不同算法的 F1 值变化曲线

真实数据集中，FNOD 算法在初始参数 k 较小时性能要低于其余 5 种算法。随着 k 值增大，FNOD 算法的 *F*1 值逐渐上升，最佳 *F*1 值大于或等于其余算法。由于 Ecoli 数据集和数据集 Lymphography 中数据数量较少，较大的初始参数 k 会接近数据点数量，从而导致数据集中离群点拥有较多的相互 k 近邻点，不利于本文算法，所以 FNOD 算法仅接近或略高于其余算法。Stamps 数据集中 FNOD 算法的 *F*1 峰值达到了 1，超出了 NOF2 算法的 0.91 和 LOF 算法的 0.92。在 Lonoshpere 数据集中，FNOD 算法和 LOF、INFLO、NOF 的指标均在 0.82 上下波动。FNOD 算法优于同样使用相互近邻关系的 NOF2 算法和利用参数 k 迭代的 INS 算法。综合上述 2 个指标，相较于其他算法，FNOD 算法在检测性能上有不同程度的优势。实验验证了本文算法能全面准确地检测到离群点。

3.4　检测效率对比

为对比 FNOD 算法和其余算法的离群点检测效率，本节统计了各算法在人工数据集和真实数据集上达到最高检测精度的时间消耗，同时标注出每个数据集中排名前 2 的算法，如表 7 和表 8 所示，表中单位为秒。

表 7、表 8 显示了各算法在不同数据集下的时间消耗。本文的 FNOD 算法在大部分数据集中都属于排名前 2 的算法，特别是在 Data _ 06 和 PenDigits 中，算法的时间消耗为 9.36 s 和 5.35 s，远低于其余算法。本文的 FNOD 算法和 INS 算法都需要计算不同 k 值下的近邻信息，算法时间消耗较大，但由于 FNOD 算法利用 NPF 算法对数据集进行了剪枝，很大程度上减少了算法关于正常点的计算，从而减少了时间消耗，所以 FNOD 算法的时间消耗远低于 INS 算法。此外，由于 Data _ 02 和 Ecoli 数据集中数据分布较为分散，NPF 算法能够剪枝的正常点较少，导致 FNOD 算法的时间消耗较大。综上所述，结合图 7、图 8 的实验结果，可以看出 FNOD 算法以较低的时间消耗保持了较高的算法性能。

表 7　人工数据集时间消耗

数据集	FNOD/s	INFLO/s	NOF/s	NOF2/s	INS/s
Data _ 01	**2.71**	3.62	3.71	3.93	**1.44**
Data _ 02	3.76	**3.55**	**3.29**	4.00	4.91
Data _ 03	**1.70**	2.63	**2.56**	3.00	**1.85**
Data _ 04	**5.88**	6.35	**6.12**	6.82	10.38
Data _ 05	**3.95**	5.14	5.43	5.44	**2.52**
Data _ 06	**9.36**	13.91	13.96	14.04	**10.59**

表 8　真实数据集时间消耗

数据集	FNOD/s	INFLO/s	NOF/s	NOF2/s	INS/s
Ecoli	1.10	**0.12**	**0.13**	0.25	7.36
Lonoshpere	**0.39**	**0.43**	0.45	2.16	18.90
Lymphography	**0.06**	0.09	0.08	**0.48**	5.79
PenDigits	**5.35**	**9.11**	9.20	11.90	23.12
Stamps	**0.20**	**0.34**	0.37	0.51	2.83
Wdbc	**0.34**	**0.54**	0.55	2.43	10.29

4 结 论

本文分析了基于各种近邻关系的离群点检测算法和近年来较为新颖的算法。针对传统算法精确度较低且局限于获取一个参数 k 下的离群信息的问题,首先提出了一种基于相互 k 近邻关系的近邻剪枝方法,排除大部分位于聚类区域的正常点,减少了关于正常点的冗余计算,同时提高了 FNOD 算法的精确度;其次,提出了一种基于相互近邻波动的离群点检测算法,统计不同参数 k 下数据点的近邻差,根据这种差值的波动作为离群因子检测离群点。在人工数据集和真实数据集下的实验验证了该算法具有良好的离群点检测能力。但算法对初始参数 k 的选取也较为敏感,如何自适应获取初始参数 k、减少算法对参数的依赖从而提高精确率,将是未来的研究方向。

参考文献

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

[2] MALINI N, PUSHPA M. Analysis on credit card fraud identification techniques based on KNN and outlier detection[C]// The 3rd International Conference on Advances in Electrical, Piscataway, USA, 2017: 255-258

[3] 杨加, 李笑难, 张扬, 等. 基于大数据分析的校园电子邮件异常行为检测技术研究[J]. 通信学报, 2018, 39(S1): 116-123

[4] 黄超, 胡志军, 徐勇, 等. 基于视觉的车辆异常行为检测综述[J]. 模式识别与人工智能, 2020,33(3): 234-248

[5] 任家东, 刘新倩, 王倩, 等. 基于 KNN 离群点检测和随机森林的多层入侵检测方法[J]. 计算机研究与发展, 2019,56 (3): 566-575

[6] 马琳, 张莎莎, 宋姝雨, 等. 基于 SDN 的智能入侵检测系统模型与算法[J]. 高技术通讯, 2020,30(5): 533-537

[7] LIN J, KEOGH E, FU A, et al. Approximations to magic: finding unusual medical time series[C]// The 18th IEEE Symposium on Computer-Based Medical Systems, Piscataway, USA, 2005: 329-334

[8] 薛安荣, 鞠时光, 何伟华,等. 局部离群点挖掘算法研究[J]. 计算机学报, 2007,30(8):1455-1463

[9] LI Y M, WANG Y J, GUAN H T. Improve the detection of clustered outliers via outlier score propagation[C]// 2019 IEEE International Conference on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking, Piscataway, USA, 2019: 1085-1091

[10] WANG C, GAO H, LIU Z, et al. A new outlier detection model using random walk on local information graph[J]. *IEEE Access*, 2018, 6: 75531-75544

[11] HAWKINS D M. Identification of outlier[J]. *Biometries*, 1980, 37(4): 860-860

[12] KNORR E M, NG R T. Algorithms for mining distance-based outliers in large datasets[C]// Proceedings of the 24rd International Conference on Very Large Data Bases, San Francisco, USA, 1998: 392-403

[13] BREUNIG M M, KRIEGEL H P, NG R T, et al. LOF: identifying density-based local outliers[J]. *Proceedings of the* 2000 *ACM SIGMOD International Conference on Management of Data*, 2000, 29(2): 93-104

[14] JIN W, TUNG A K H, HAN J, et al. Ranking outliers using symmetric neighborhood relationship[C]// Pacific-asia Conference on Knowledge Discovery and Data Mining, Berlin, Germany, 2006: 577-593

[15] HUANG J L, ZHU Q S, YANG L J, et al. A non-parameter outlier detection algorithm based on natural neighbor [J]. *Knowledge Based Systems*, 2016, 92(15): 71-77

[16] TANG B, HE H B. A local density-based approach for outlier detection[J]. *Neurocomputing*, 2017, 241(7): 171-180

[17] ZHOU H F, LIU H J, ZHANG Y J, et al. An outlier detection algorithm based on an integrated outlier factor[J]. *Intelligent Data Analysis*, 2019, 23(5): 975-990

[18] NING J, CHEN L T, CHEN J W. Relative density-based outlier detection algorithm[C]// Proceedings of the 2nd International Conference on Computer Science and Artificial Intelligence, New York, USA, 2018: 227-231

[19] 黄添强,李凯,郭躬德. 基于局部相关维度的流形离群点检测算法[J]. 模式识别与人工智能, 2011, 24(5): 629-636

[20] 赵科平, 周水庚, 关佶红, 等. 一种新的离群数据对

象发现方法[C]//中国人工智能学会第 10 届全国学术年会论文集. 北京:北京邮电大学出版社, 2003: 470-475

[21] HA J, SEOK S, LEE J S. Robust outlier detection using the instability factor [J]. *Knowledge-Based Systems*, 2014, 63: 15-23

[22] SU S B, XIAO L M, RUAN L, et al. An efficient density-based local outlier detection approach for scattered data [J]. *IEEE Access*, 2018, 7: 1006-1020

[23] 张忠平, 房春珍. 基于累积全熵的子空间聚类离群点检测算法[J]. 计算机集成制造系统, 2015, 21(8): 2249-2256

[24] HUANG J L, ZHU Q S, YANG L J, et al. A novel outlier cluster detection algorithm without top-n parameter [J]. *Knowledge-Based Systems*, 2017, 121(1): 32-40

[25] 张忠平, 邱敬仰, 刘丛, 等. 基于聚类离群因子和相互密度的离群点检测算法[J]. 计算机集成制造系统, 2019, 25(9): 2314-2323

FNOD: outlier detection algorithm based on fluctuation of nearest neighbor difference factor

ZHANG Zhongping[*,**,***], DENG Yu[*], LIU Weixiong[*], ZHANG Yuting[*]

([*] College of Information Science and Engineering, Yanshan University, Qinhuangdao 066004)

([**] The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao 066004)

([***] The Key Laboratory of Software Engineering of Hebei Province, Qinhuangdao 066004)

Abstract

To solve the problem of low accuracy and low degree of pruning in the existing outlier detection methods, a new outlier detection algorithm based on fluctuation of nearest neighbor difference factor is proposed. First, a pruning method based on the neighbor relationship is proposed according to the feature that the number of the mutual k neighbors (MUN) of outliers is much smaller than the parameter k. Second, the concept of nearest neighbor difference is proposed to describe the distribution characteristics of data objects and their neighbors. When parameter k changes, the nearest neighbor difference of outliers and normal points will be different. Finally, the fluctuation of the nearest neighbor difference is used to measure the outlier degree of each data point, and then the outlier point is detected. The experimental results in the artificial and real data sets show that the proposed algorithm can detect outliers effectively and comprehensively.

Key words: data mining, outliers, pruning, mutual k neighbor (MUN), nearest neighbor difference fluctuation factor