doi :10.3969/j.issn.1671-1122.2016.11.006

基于多窗口机制的聚类异常检测算法

何明亮[1]，陈泽茂[1]，左进[2]

（1. 海军工程大学信息安全系，湖北武汉 430033；2. 91428 部队，浙江宁波 315000）

摘　要：文章通过分析单窗口聚类异常检测算法的不足，综合利用权值、相似度和局部密度等概念对单窗口检测出的潜在异常点进行归属查找和异常合并，设计了一种基于多窗口机制的数据流异常检测算法。该算法首先在单个窗口内用改进的 *K*-means 聚类算法对预处理之后的数据流进行初步聚类检测，将每个窗口聚类的结果分为正常簇集合和潜在异常点集合。然后对单窗口检测结果进行二次判断。针对单窗口检测的潜在异常点，利用相似度原理进行正常类簇的归属查找，排除异常误判；利用局部密度等概念，对剩下的潜在异常点进行异常合并，再次排除可能的正常点。最后利用时间权值，综合多个数据流窗口的检测结果得出最终异常数据。仿真实验表明，相较于单窗口数据流异常检测算法，该算法提高了数据流的异常检测率，减少了异常误判，在检测率和误报率方面更具优势。

关键词：单窗口；多窗口；数据流；异常检测

中图分类号：TP309　　**文献标识码：**A　　**文章编号：**1671-1122（2016）11-0033-07

中文引用格式：何明亮，陈泽茂，左进 . 基于多窗口机制的聚类异常检测算法 [J]. 信息网络安全，2016（11）：33-39.

英文引用格式：HE Mingliang, CHEN Zemao, ZUO Jin. Cluster Anomaly Detection Algorithm Based on Multi-windows Mechanism[J]. Netinfo Security, 2016 (11) : 33-39.

Cluster Anomaly Detection Algorithm Based on Multi-windows Mechanism

HE Mingliang[1], CHEN Zemao[1], ZUO Jin[2]

(1. *Information Security Department*, *Naval University of Engineering*, *Wuhan Hubei* 430033, *China*; 2. 91428 *Troops of PLA*, *Ningbo Zhejiang* 315000, *China*)

Abstract: This paper analyses the weaknesses of cluster anomaly detection algorithm based on single-window, takes advantage of weigh value, similarity, local density and other concepts to conduct affiliation search and abnormal merging on potential abnormal point obtained by single-window algorithm. Moreover, a dataflow anomaly detection algorithm based on multi-window mechanism is designed. This algorithm firstly conducts primary cluster detection to preprocessed dataflow with improved *K*-means cluster algorithm in single window and then conduct second judge to the results. For the potential abnormal point detected by single-window algorithm, similarity principle is adopted to conduct normal cluster affiliation search to exclude misjudges, other conceptions like local density is adopted to conduct abnormal merging to the rest potential abnormal points to exclude normal points again. Lastly, the time weigh value is used to obtain final abnormal data comprehensively from the detection results of several dataflow windows. The simulation shows that this algorithm has advantage over single-window cluster anomaly detection algorithm on detection rate and misjudge rate.

Key words: single window; multi-windows; data flow; anomaly detection

收稿日期：2016-07-01

基金项目：湖北省自然科学基金 [2015CF867]

作者简介：何明亮（1988—），男，湖北，硕士研究生，主要研究方向为信息安全；陈泽茂（1975—），男，福建，教授，博士，主要研究方向为网络安全；左进（1989—），男，四川，硕士，主要研究方向为信息安全。

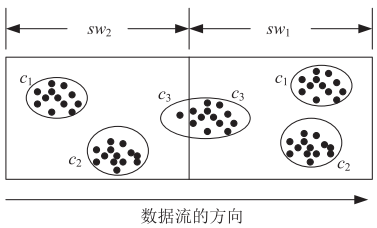
通信作者：何明亮 626946621@qq.com

0 引言

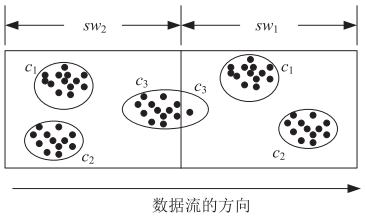
由于数据流的潜在无限性，对数据流进行全部保存再进行聚类异常分析几乎是不可能的，因此一些学者提出了窗口的概念。窗口是将传统数据挖掘应用于数据流领域的关键技术[1,2]。众多数据流聚类算法对数据流的处理与分析大多数都是基于窗口进行的，如著名的 Clustream 算法、Storm 算法[3]，以及如文献 [4–6] 中提出的各类算法。窗口的目标是对高速、海量的数据流进行约减，在一定的范围内对其分析处理。基于窗口概念的数据流聚类算法可以简化数据流，使对无限的数据流进行有限的分析成为可能。但人为地对数据流进行窗口分隔会使聚类结果有失整体性，不能很好地对整个数据流进行顺承性分析[6]。而且在单窗口内处理的结果始终都是一种阶段性的结论，从数据流的整体来看有失准确性，在网络数据流的异常检测中容易造成较大误差[7]。

1 单窗口聚类异常检测缺陷分析

单窗口聚类异常检测缺陷的情况如图 1、图 2、图 3 所示。假设在数据流的方向上依次截取两个滑动窗口 sw_1、sw_2 的数据流，采用聚类算法进行聚类，产生 3 个类簇分别为 c_1、c_2、c_3。如图 1 a）所示，滑动窗口 sw_2 的簇 c_3 中只包含 1 个数据点，因此如果只在一个窗口内进行判断，则 c_3 为异常簇，c_3 的数据点判断为异常点。但事实上，滑动窗口 sw_2 中的 c_3 是滑动窗口 sw_1 中 c_3 后到达的点，隶属于滑动窗口 sw_1 中的簇 c_3 因为人为的窗口划分造成错误的判断。如果将滑动窗口 sw_1、sw_2 中的簇 c_3 进行合并分析则可消除误差。同理，在图 1 b）中，滑动窗口 sw_1 中的簇 c_3 如果只在滑动窗口 sw_1 中进行分析，则会判断为异常。滑动窗口 sw_1 中的 c_3 属于先到达的数据点，在 sw_2 中的大部分数据点到达之前进行聚类分析造成前后脱节，形成误差。如果能够将滑动窗口 sw_1、sw_2 的聚类结果进行关联分析，则可消除误差。



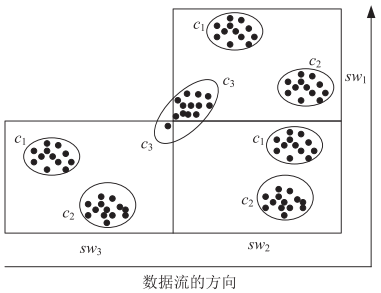
a）滞后误差



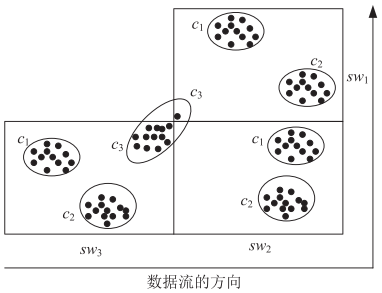
b）超前误差

图1 单窗口聚类缺陷一

同理，图 2 和图 3 中的簇 c_3 也会因为人为窗口划分而产生异常误判。



a）混合误差一



b）混合误差二

图2 单窗口聚类缺陷二

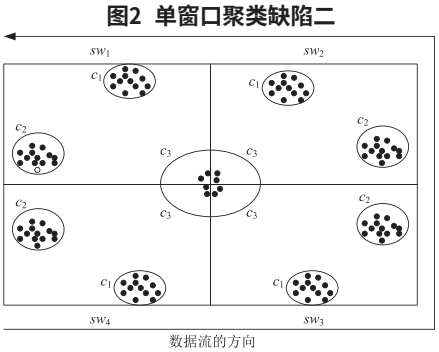


图3 单窗口聚类缺陷三

数据流的特点决定了对其处理与计算的方式必须是区间型的。基于窗口的数据流处理模型作为数据区间的一种方式，有其独特的优点。但人为地将数据流进行窗口划分，在单独的窗口内进行数据流聚类，却可能产生聚类结果误差。因此研究多个窗口的连贯性聚类检测具有非常重要的意义。

2 基于多窗口机制的聚类异常检测算法

将数据流限定在单个窗口内处理产生的结果具有很

大局限性，因此本文考虑将多个窗口聚类的结果进行连贯性综合分析，提出了基于多窗口机制的聚类异常检测算法（Cluster Anomaly Detection Algorithm Based on Multi-windows Mechanism，MWCluAD）用于数据流的异常检测。

MWCluAD 算法主要有以下两个过程：

1）在单个窗口内用聚类算法对预处理之后的数据流进行初步聚类检测。将每个窗口聚类的结果分为正常簇集合 N 与潜在异常点集合 U。在集合 N 与集合 U 生成的同时，赋予其随时间衰减的权值 w_0，并在每一个滑动窗口聚类结束时，更新集合 N 和集合 U 内所有簇和点的权值，权值随时间衰减 [8]。对于权值 $w_0<w_\triangle$ 的簇和点进行删除（$w_\triangle$ 为权值阈值）。集合 U 中被删除的点即为最终异常点。

2）对单窗口检测的结果进行二次判断。该过程主要分为两步。第一步查找潜在异常点的归属。对集合 U 中未被删除的潜在异常点与集合 N 内存活簇进行相似度判断，若相似度 $d \leqslant \overline{\overline{d}}(\overline{\overline{d}}$ 为此时集合 N 中所有簇内点到中心点的距离均值），则将数据点归并到相应集合 N 中的正常簇，同时更新簇的权值。第二步合并潜在异常点。对通过单窗口检测之后剩下的集合 U 内的数据点再次合并，若成功合并，则该簇为正常簇，将合并成功的正常簇加入到集合 N 中，同时更新其权值；若合并不成功，则将其继续保存在集合 U 中。

MWCluAD 的核心思想是将单窗口内检测的结果进行二次判断，从而避免因人为窗口划分造成结果的误判。而检测结果的二次判断主要通过潜在异常点的归属查找与合并这两步来完成。

MWCluAD 算法框架如图 4 所示。

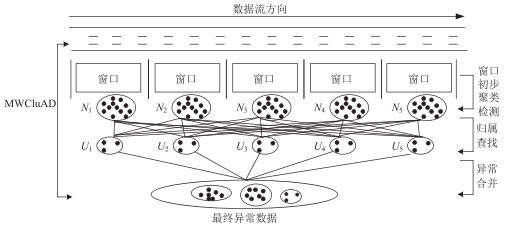


图4 MWCluAD算法框架

算法内容主要由以下两部分组成：

1）窗口初步聚类检测。完成单个窗口内的异常检测。

2）多窗口间的异常二次判断。主要包括潜在异常点的归属查找和异常合并，消除异常误判，降低误报率。

2.1 数据预处理

网络数据流的抽样样本为网络数据包。将数据包统计为网络连接记录仍然不能直接用于数据挖掘，需要对其进行预处理。预处理过程主要包括特征属性项的选取、属性值的数值化和属性值的标准化。

1）特征属性项的选取

鉴别并选取关键属性项作为数据流挖掘算法的输入对数据分析意义重大，不仅可以降低算法的复杂度和所需的存储空间，还可以提高算法的准确率。以 KDD99 数据集为例，文献 [9] 根据 PFRM 算法（基于效能等级的重要特征排序算法）针对数据集中每一条网络连接记录的 41 个特征属性，筛选出了对应于不同网络攻击行为的重要特征属性子集，如表 1 所示。表 1 中的数字对应于 KDD99 数据集中特征属性项的 1~41 编号值。

表1 PFRM算法重要特征属性子集列表

攻击类型	重要特征属性子集
Normal	2、3、4、5、6、10、12、23、29、32、33、34、36
DoS	23、24、25、26、36、38、39
Probing	2、4、5、23、24、33
R2L	1、3、32
U2R	1、2、4、5、12、29、34

综合得出 PFRM 算法可选择的特征属性项个数为 19，即特征属性项集 $F1$={1,2,3,4,5,6,10,12,23,24,25,26,29,32,33,34,36,38,39}。文献 [10,11] 利用 RS 粗糙集理论对数据集的属性进行约简，并与 SVDF、LGP、MARS 算法进行比较，选出了 6 个最为重要的特征属性项。各算法选择的重要特征属性项如表 2 所示。

表2 RS、SVDF、LGP、MARS算法重要特征属性子集列表

算法	重要特征属性子集
RS	3、4、5、24、32、33
SVDF	2、4、5、23、24、33
LGP	3、5、12、27、31、35
MARS	5、24、27、33、34、35

考虑到 RS 选择的特征属性子集能够很好地判断入侵，且特征属性项的个数较小，容易实现，本文采用了 RS 算法对数据集属性约简筛选出的特征属性项子集 $F2$={3,4,5,24,32,33}。

2）属性值的数值化

在网络连接记录中的所有特征属性中，还包含一些非数值数据，如 flag、Protocol_type、service 等属性值是字符串类型的数据。为了能够对这些数据进行运算，需要将这些字符串变为数值型。flag（连接正常或错误的状态）属性取值有 S0、S1、S2、S3、SF、SH、OTH、REJ、RSTO、RSTOSO、RSTR 共 11 个，可分别将其转换为对应的 1~11

的整数；Protocol_type（协议类型）属性的取值 icmp、tcp、udp 可对应为整数 1~3；其他协议类型一律对应为 4；service（目标主机的网络服务类型）属性共有 70 种取值，可分别对应于整数 1~70。

3）属性值的标准化

大部分的数据流挖掘算法根据相似度对算法的输入即特征属性项进行挖掘分析，将相似度小的数据聚为一类，将相似度大的数据分开。而相似度对特征属性项的值域范围是非常敏感的。例如，采用欧式距离对如下两组数据进行相似度计算

第一组：{(1,1,2,3),(2,2,3,2)}

第二组：{(180,340,320,120),(280,240,420,220)}

第一组中两个数据的相似度为

$$D_1=\sqrt{(2-1)^2+(2-1)^2+(3-2)^2+(2-3)^2}=2$$

第二组中两个数据的相似度为

$$D_2=\sqrt{(280-180)^2+(240-340)^2+(420-320)^2+(220-120)^2}=200$$

如果算法以数值 3 为相似度的度量标准，则根据得到的结果，第一组应该归为一类，第二组应该被划分开。但事实上，第二组两个数据之间的相对距离与第一组两个数据之间的相对距离等同，直接用特征属性的值进行计算势必造成很大误差，必须对数据进行标准化。

对于包含 m 个特征属性项的 L 个数据的数据集 $DS_j[i]$，可通过公式（1）、公式（2）、公式（3）将其转换到新的标准化空间 $NEW\_DS_j[i]$。

$$NEW\_DS_j[i]=\frac{DS_j[i]-mean\_vector[i]}{std\_vector[i]} \quad \cdots\cdots\cdots\cdots (1)$$

$$mean\_vector[i]=\frac{1}{L}\sum_{j=1}^{L}DS_j[i] \quad \cdots\cdots\cdots\cdots (2)$$

$$std\_vector[i]=\sqrt{\frac{1}{L-1}\sum_{j=1}^{L}(DS_j[i]-mean\_vector[i])^2} \quad \cdots\cdots\cdots (3)$$

其中，$mean\_vector[i]$ 和 $std\_vector[i]$ 分别是数据集 DS 中第 i 个特征属性项的均值和标准方差，$j=1,2,\cdots,L$，$i=1,2,\cdots,m$。这样便可将数据集中不同特征属性项由其初始空间转换到标准空间，消除不同值域范围对挖掘算法的影响。

2.2 MWCluAD的窗口初步检测

对于单个窗口内的异常检测，本文采用聚类算法中应用最为广泛的基础算法——K-means 算法[12]。K-means 算法的最大特点是能简单高效地处理大规模数据集，适合对窗口内的数据流进行快速聚类。但 K-means 算法本身有许多不足之处，并不能直接应用[13-15]。本文通过分析该算法的特点，设计了一种基于改进 K-means 的聚类异常检测算法，用于 MWCluAD 的窗口初步检测。算法描述如下：

算法 1 基于改进 *K*-means 的异常检测算法

输入：d 维数据集，$X=\{x_1,x_2,\cdots,x_n\}$，聚类个数 K，聚类收敛精度 ε，最近邻个数 t。

输出：K 个类簇中心 $C=\{c_1,\cdots,c_j,\cdots,c_K\}$，数据 x_i 所属类簇标签 L，异常点集合 UD。

1）令初始聚类准则函数值 $J_0=0$，每个数据点 x_i 的异常度 $Abn_{x_i}=0$（$i=1,2,\cdots,n$）。

2）对于 R^d 空间上的数据集 $X=\{x_1,\cdots,x_i,\cdots,x_n\}$ 中的每一个数据点 x_i，求出其紧密性 $Tigh(x_i)=(\sum_{x_j\in G_t(x_i)}D(x_i,x_j))^{-1}$，其中，$G_t(x_i)$ 为 x_i 的 t 个最近邻数据点集合。

3）删除 X 所有紧密性小于 $\frac{1}{n}\sum_{x_i\in X}Tigh(x_i)$ 的稀疏数据点，得到密集数据点集合 X'。

4）在 X' 中，取密集性最大者 $Tigh_{\max}(x_i)$ 为第一个初始聚类中心 c_1，取距离 c_1 最远的数据点作为第二个初始聚类中心 c_2，第 m（$3\leqslant m\leqslant K$）个初始聚类中心 c_m 为满足 $\max(\min(D(x_i,c_1),D(x_i,c_2),\cdots,D(x_i,c_{m-2})))(i=1,2,\cdots,n)$ 的数据点 x_i，$x_i\in X'$，直至得到最终 K 个初始聚类中心，分别代表 K 个类簇 CLU_j（$j=1,2,\cdots,K$）。

5）计算 X 的所有数据点与各个聚类中心的欧式距离 $D(x_i,c_j)=\sqrt{\sum_{h=1}^{d}(x_{ih}-c_{jh})^2}$，其中，$i=1,2,\cdots,n$，$j=1,2,\cdots,K$。若 c_j 使得 $D(x_i,c_j)=\min(D(x_i,c_j))$，则将点 x_i 划分到 c_j 所代表的簇，即 $L_{x_i}=CLU_j$。

6）重新计算各类簇的聚类中心 $c_j'=\frac{1}{m_j}\sum_{L_{x_i}=CLU_j}x_i$，其中，$m_j$ 是 c_j 所代表的簇拥有的数据点总数。

7）在形成的 K 个类簇中，若属于该簇的数据点与该聚类中心距离大于平均距离，即 $D(x_i,c_j)\geqslant\frac{1}{m_j}\sum_{L_{x_i}=CLU_j}D(x_i,c_j)$，其中，$m_j$ 是 c_j 所代表的簇拥有的数据点总数，则 $Abn_{x_i}=Abn_{x_i}+1$。

8）若 $Abn_{x_i}\geqslant 3$，则判断 x_i 为异常点，将其从数据集 X 中剔除，并入集合 UD 中。

9）判断聚类准则函数 $J=\sum_{j=1}^{K}\sum_{L_{x_i}=w_j}D^2(x-c_j)$ 是否满足收敛条件 $|J'-J|\leqslant\varepsilon$，若不满足，则转到步骤 5）继续迭代；若满足收敛条件，则算法结束，输出 C、L 和 UD。其中，J 是上次迭代聚类准则函数值，J' 是本次聚类准则函数值。

2.3 MWCluAD的异常二次判断

2.3.1 相关概念与定义

定义 1 潜在异常点集合 *U* 与正常簇集合 *N*

潜在异常点集合 $U=\{UD,c|_{n_c<(1/q)K^{-1}\sum_1^K n_i}\}$；正常簇集合 $N=\{c|_{n_c\geqslant(1/q)K^{-1}\sum_1^K n_i}\}$。其中，*UD* 为单个窗口应用改进 *K*–means 初次检测到的异常点集合；*c* 为单个窗口聚类产生的类簇，数量为 *K*；n_i 为第 *i* 个类簇内数据点个数；*q* 为异常临界值，若类簇 *c* 的 n_i 小于所有簇内元素个数均值的 1/*q*，即判断 *c* 内的数据点为潜在异常点 [16,17]。

利用改进 *K*–means 异常检测算法对单个窗口数据处理完成之后，将处理结果分为潜在异常点集合 *U* 和正常簇集合 *N*。将规模较小的类簇和 *UD* 归入 *U*，规模较大的类簇归入 *N*。

定义 2 类簇中心点坐标 *CF*

$$CF=(m_1,m_2,m_3,\cdots,m_d,n,t,\bar{d},w) \quad \cdots\cdots(4)$$

公式（4）中，$m_1,m_2,m_3,\cdots,m_d$ 为类簇 *c* 中心点具体的 *d* 维属性坐标值，*d* 取 *c* 内所有数据点属性坐标的均值；*n* 为类簇 *c* 拥有的数据点个数；*t* 为类簇 *c* 生成的时间戳，用于计算 *c* 的权值；*w* 为类簇 *c* 的权值，初值为 1，随时间进行衰减，当 $w\geqslant w_\triangle$ 时，意味着类簇 *c* 为存活状态，参与运算；$\bar{d}$ 为类簇 *c* 内所有数据点到簇中心的距离平均值，简称簇内平均距离，即

$$\bar{d}=n^{-1}\sum_{i=1}^{n}\sqrt{(x_{i1}-m_1)^2+(x_{i2}-m_2)^2+\cdots+(x_{id}-m_d)^2} \quad \cdots\cdots(5)$$

对于 *N* 中任意一个类簇 *c*，算法并不关心其中具体的数据点，所以不必耗费大量的内存对其进行处理，只用类簇 *c* 的中心点坐标 *CF* 进行标识，用 *CF* 参与运算即可。

定义 3 潜在异常点坐标 *df*

$$df=(m_1,m_2,m_3,\cdots,m_d,t,w) \quad \cdots\cdots(6)$$

公式（6）中，$m_1,m_2,m_3,\cdots,m_d$ 为潜在异常点具体的 *d* 维属性坐标值；*t* 为潜在异常点生成的时间戳，用于计算权值；*w* 为潜在异常点的权值，初值为 1，随时间进行衰减，当 $w\geqslant w_\triangle$ 时，意味着潜在异常点为存活状态，参与运算。

U 中的每个数据点都属于潜在异常对象，是算法的运算主体，用其坐标 *df* 进行标识，参与运算。

定义 4 权值衰减函数 w_{t_2}

$$w_{t_2}=w_{t_1}e^{-\lambda(t_2-t_1)} \quad \cdots\cdots(7)$$

其中，w_{t_2} 为 t_2 时刻的权值，w_{t_1} 为 t_1 时刻的权值。

将各类簇与潜在异常点的权重值设置为随时间衰减的函数，可以降低历史结果对当前结果分析的重要程度，体现近期数据的重要性，同时对其进行状态标记，即存活或消亡，决定是否参与运算。

定义 5 权值更新问题 在每一轮算法迭代结束后，对各类簇的权值大小重新计算。

1）点与类簇的合并

在本算法中，将 *U* 和 *N* 中的正常类簇和潜在异常点均赋予其权值 *w*。在运算过程中，如果 *U* 中有 *h* 个潜在异常点进行归属查找，归属到 *N* 的某个正常簇中，则此正常簇的权值更新公式如公式（8）所示。

$$w_{t_2}'=(n+h)(nw_{t_1}e^{-\lambda(t_2-t_1)}+\sum_{i=1}^{h}w_i) \quad \cdots\cdots(8)$$

其中，w_{t_1} 为此正常簇在上一次更新权值时的权值；w_i 为第 *i* 个潜在异常点的权值；*n* 为当前正常簇内元素个数。

从公式（8）可以看出，潜在异常点归属之后，该归属簇的新权值等于原来簇内每个数据点的权值与新加入数据点权值的平均。

2）点与点的合并

如果 *U* 中有 *h* 个潜在异常点合并成功，产生了新的正常簇，则新生正常簇的权值更新公式如公式（9）所示。

$$w_{新簇}=h^{-1}\sum_{i=1}^{h}w_i \quad \cdots\cdots(9)$$

其中，w_i 为第 *i* 个潜在异常点的权值。由公式（9）可以看出，新生正常簇的权值为所有数据点的权值平均。

2.3.2 潜在异常点归属查找

1）潜在异常点归属查找过程

潜在异常点归属查找过程就是对潜在异常点集合 *U* 和正常类簇集合 *N* 中所有存活点（数据对象的权值 $w\geqslant w_\triangle$）和类簇进行归属匹配。在 MWCluAD 中，权值的思想贯穿始终。由于内存的限制与历史结果影响力的衰减，赋予数据对象权值能够使其不断消亡与更新，实现整个算法的更替，使算法具有高效性。潜在异常点的归属查找过程如图 5 所示。

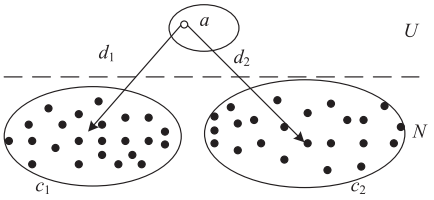


图5 潜在异常点的归属查找过程

图 5 中，假设在 t 时刻 U 中的潜在异常点为 a，N 包含两个正常类簇 c_1 和 c_2。数据点 a 的潜在异常点归属查找过程如下：首先，计算数据点与类簇之间的相似度；其次，计算各类簇的平均距离均值；最后，比较数据点相似度与类簇的簇内平均距离，若相似度小于等于某个平均距离，则将该数据点划归为该类簇，若相似度大于所有类簇的簇内平均距离，则数据点不加入任何簇。

2）潜在异常点归属查找算法

算法 2 潜在异常点归属查找算法

输入：潜在异常点集合 U、正常簇集合 N、权值阈值 $w_{\triangle}$。

输出：潜在异常点集合 U^*、正常簇集合 N^*、最终异常点集合 OS。

（1）当一个窗口聚类完成后，首先更新 U 与 N 中所有正常类簇和潜在异常点的权值 $w_{t_2}=w_{t_1}e^{-\lambda(t_2-t_1)}$，并删除 $w<w_{\triangle}$ 的数据对象，并入 OS；

（2）计算 U 中数据点与 N 中所有簇的相似度 $S(a,c)=(\sum_{i=1}^{d}(m_{ai}-m_{ci})^2)^{1/2}$；

（3）计算 N 中当前存活簇簇内平均距离均值 $\overline{\overline{d}}=k^{-1}\sum_{j=1}^{k}\overline{d_j}$；

（4）令 $d=S_{\min}(x,c)$，若 $d\leqslant\overline{\overline{d}}$，则将对应的潜在异常点并入与其最相似的正常簇 c 中，同时更新吸纳数据点之后 c 的 CF 与权值 $w'_{t_2}=(n+h)(nw_{t_1}e^{-\lambda(t_2-t_1)}+\sum_{i=1}^{h}w_i)$；若 $d>\overline{\overline{d}}$，不做任何处理，保持该数据点的潜在异常点身份，保留在 U 中，等待下一步处理。

2.3.3 潜在异常点合并

1）潜在异常点合并过程

潜在异常点合并过程就是将所有的潜在异常点合并成可能的正常类簇的过程。如图 6 所示，首先在潜在异常点集合中找出 n 个聚类中心；然后以该 n 个点为圆心，各类簇的簇内平均距离均值为半径画圆，若该圆内的数据点个数满足一定数量，则将该区域范围内的数据点合并为一个正常类簇。

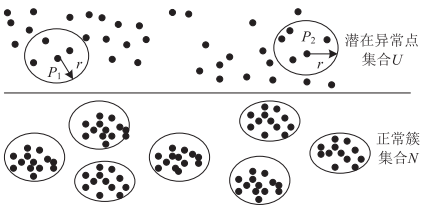


图6 潜在异常点的合并

2）潜在异常点合并算法

算法 3 潜在异常点合并算法

输入：潜在异常点集合 U^*、正常簇集合 N^*。

输出：潜在异常集合 U^{**}、正常簇集合 N^{**}。

（1）计算 N^* 中所有存活簇的簇内元素个数平均数 $\bar{n}$、所有存活簇的簇内平均距离均值 $\overline{\overline{d}}$，$\bar{n}=k^{-1}\sum_{1}^{k}n_i$，$\overline{\overline{d}}=k^{-1}\sum_{j=1}^{k}\overline{d_j}$。

（2）利用 AP 算法搜索出 U^* 中可能的类代表点 $P_1,P_2,\cdots,P_n$。

（3）在 U^* 中，计算以类代表点 $P_1,P_2,\cdots,P_n$ 为中心，以 $r=\overline{\overline{d}}$ 为半径的邻域范围内的数据点个数 n。如果 $n\geqslant(1/q)k^{-1}\sum_{1}^{k}n_i$，则将该区域范围内的数据点成功合并为一个正常类簇，并入 N^* 中，更新该新类簇的权值（$w_{新簇}=h^{-1}\sum_{i=1}^{h}w_i$）；否则，合并失败，合并结束。

3 实验仿真及分析

对于数据流的异常检测，本文针对单个窗口检测的缺陷提出了基于多个窗口联合检测的方法。本次实验主要完成两种窗口机制在异常检测方面的性能比较。

所有的算法均是通过 VC++6.0 编程实现，并结合 MATLAB 7.1 进行了实验数据的仿真测试。实验运行的机器为 i5-2430M，2.4 GHz 的 CPU，64 位的 Win 7 操作系统，2.0 GB 内存。

实验数据来源于 UCI 机器学习数据库 [13]。实验选取了不同数量规模和维度的 4 种数据集，包括 Iris 数据集、Ecoli 数据集、Yeast 数据集和 KDD99 数据集。4 种不同数据集的描述如表 3 所示。

表3 实验数据集列表

数据集种类	数据对象个数	属性个数	分类数
Iris 数据集	150	4	3
Ecoli 数据集	336	7	8
Yeast 数据集	1484	8	10
KDD99 数据集	81452	41	5

1）改进 K-mean 算法与原 K-mean 算法

图 7 是改进 K-mean 算法与原 K-means 算法对不同数据集聚类结果准确率的对比。由于改进 K-mean 算法在选取初始聚类中心时，尽量避免了噪声点，更加靠近真实的聚类中心，使算法在聚类效果上更佳。改进 K-mean 算法在 Iris 和 Ecoli 数据集上的聚类准确率比原 K-mean 算法提高了近 10%；在 Yeast 数据集上的聚类准确率提高了近 25%。通过对比说明改进 K-mean 算法对大数据集更加具有优势。

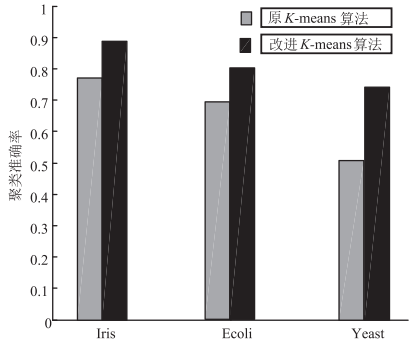


图7 改进*K*-means算法与原*K*-means算法聚类准确率比较

2）MWCluAD 多窗口与单窗口异常检测算法

实验首先在 Iris、Ecoli、Yeast 数据集中加入一定比例的人工异常数据，用于测试两种算法的异常检测性能；然后用真实的网络数据 KDD99 数据集进行检测对比[18,19]，结果如表 4、图 8 所示。在 Iris、Ecoli、Yeast 数据集的平均用时方面，多窗口异常检测算法比单窗口异常检测算法平均用时多 50 ms（因为多窗口算法多了二次判断过程）；而检测率提高了 4%。在误检率方面，由于多窗口异常检测算法对单窗口异常检测结果进行了潜在异常点的归属查找和异常合并，排除了因人为窗口划分产生的异常误判，提高了检测的质量，误检率远远低于单窗口异常检测算法。在对真实的高维网络数据 KDD99 数据集进行异常检测时，多窗口异常检测算法在时间效率方面稍微落后一些，在检测率和误检率方面，性能优于单窗口异常检测算法，说明多窗口异常检测算法在异常检测性能方面更具优势。

表4 多窗口异常检测算法与单窗口异常检测算法性能比较

数据集	单窗口异常检测			多窗口异常检测		
	检测率	误检率	用时/ms	检测率	误检率	用时/ms
Iris	75.3%	16.8%	999	76.4%	5.4%	1030
Ecoli	70.2%	23.5%	1026	75.3%	9.6%	1094
Yeast	66.0%	31.4%	1786	73.7%	10.2%	1837
KDD99	52.4%	40.5%	98230	64.5%	15.7%	99005

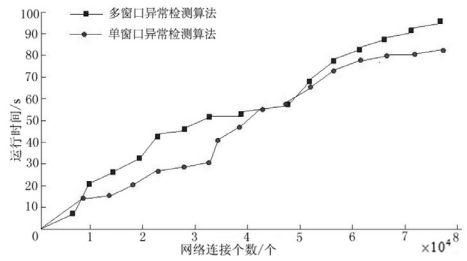


图8 多窗口异常检测算法与单窗口异常检测算法用时比较

4 结束语

本文针对数据流单窗口异常检测算法的缺陷，提出了基于多窗口机制的聚类异常检测算法——MWCluAD 算法。该算法主要包括数据的预处理、窗口的初步异常检测和异常二次判断，通过将单个窗口检测的异常结果定义为潜在异常，将其进行复检测，以达到减少误差的目的。实验仿真表明，该算法提高了数据流的异常检测率，减少了异常误判的产生。如何在真实网络环境中搭建平台与实践应用，将是下一步研究方向。（责编 潘海洋）

参考文献：

[1] SHIE B E, YU P S, TSENG V S. Efficient Algorithms for Mining Maximal High Utility Itemsets from Data Streams with Different Models[J]. Expert Systems with Applications, 2012, 39(17): 12947-12960.
[2] 吴晓平，周舟，李洪成 . Spark 框架下基于无指导学习环境的网络流量异常检测研究与实现 [J]. 信息网络安全，2016（6）：1-7.
[3] Angiulli F, Fassetti F. Detecting Distance-based Outliers in Streams of Data[C]//ACM. The 16th ACM Conference on Information and Knowledge Management, November 6-10, 2007, Lisbon, Portugal. New York: ACM, 2007: 811-820.
[4] 姚晨 . 高纬数据流的异常检测 [D]. 成都：电子科技大学，2011.
[5] 陈晓，赵晶玲 . 大数据处理中混合型聚类算法的研究与实现 [J]. 信息网络安全，2015（4）：45-49.
[6] 程军锋，王治和，刘佳，等 . 一种基于滑动窗口的一趟数据流聚类算法 [J]. 首都师范大学学报，2014，35（4）：38-40.
[7] 朱琳，刘晓东，朱参世 . 基于衰减滑动窗口数据流聚类算法研究 [J]. 计算机工程与设计，2012，33（7）：2659-2796.
[8] 张付霞，蒋朝惠 . 一种基于网格聚类的查询隐私匿名算法研究 [J]. 信息网络安全，2015（8）：53-58.
[9] 田俊锋，王惠然，刘玉玲 . 基于属性排序的入侵特征缩减方法研究 [J]. 计算机研究与发展，2006，43（S2）：565-569.
[10] BRUHA I. Pre-and Post-Processing in Machine Learning and Data Mining[J]. Machine Learning and Its Applications, 2010, 18(3): 258-266.
[11] 陈才杰 . 粗糙集理论在知识发现数据预处理中的研究与应用 [D]. 武汉：武汉理工大学，2014.
[12] MACQUEEN J. Some Methods for Classification and Analysis of Multivariate Observations[EB/OL]. https://www.researchgate.net/publication/220049032_Some_Methods_for_Classification_and_Analysis_of_MultiVariate_Observations, 2016-6-20.
[13] TZORTZIS G, LIKAS A. The Minmax k-means Clustering Algorithm[J]. Pattern Recognition, 2011, 44(4): 866-876.
[14] 蒋大宇 . 快速有效的并行二分 K 均值算法 [D]. 哈尔滨：哈尔滨工程大学，2013.
[15] 朱建宇 . K 均值算法研究及其应用 [D]. 大连：大连理工大学，2013.
[16] 李旬，徐剑，焦英楠，等 . 基于异常特征的社交网页检测技术研究 [J]. 信息网络安全，2015（5）：41-46.
[17] LI Huafu. MHUI-max: An Efficient Algorithm for Discovering High-utility Itemsets from Data Streams[J]. Journal of Information Science, 2011, 37(5): 532-545.
[18] 程转流，胡为成 . 滑动窗口模型下的概率数据流聚类 [J]. 计算机工程与应用，2011，47（4）：141-145.
[19] AHMED C F, TANBEER S K, JEONG B S, et al. Interactive Mining of High Utility Patterns over Data Streams[J]. Expert Systems with Application, 2012, 39(15): 11979-11991.