# 论文信息

论文题目‘DILOF：EFFective and Memory Efficient Local Outlier Detection in Data Streams’

论文作者：Gyoung S.Na,Donghyun Kim,Hwanjo Yu

来源：KDD2018

# 相关介绍

在流数据异常检测领域，有三大类异常检测方法：

1、 distribution-based：这类方法目的是研究基于流数据的概率模型，但是这种模型需要太多的先验知识，如分布性等，而现实情况中是无法转却知道一种数据的分布特征的，最常见的假设都是基于数据分布为正态分布，而这种假设会严重造成异常数据精测精度的降低。

2、 Clustering-based:构建集群来为底层数据分布建模。然后，根据算法，形成小簇的数据点或远离簇形中心的数据点被认为是离群点。然而，设计这种算法的主要目的是建立一个聚类模型，而不是检测离群值，这种算法通常对异常单点效果较好，在异常情况复杂的数据点效果一般。

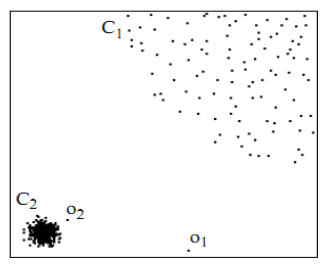
3、Distance-based:通过计算一个点相对于数据集中其他点的距离来检测异常值,在不知道数据分布或者数据特征情况都了解不足的情况下可以选择这种方法，因此，基于距离的方法在近些年使用最为广泛。

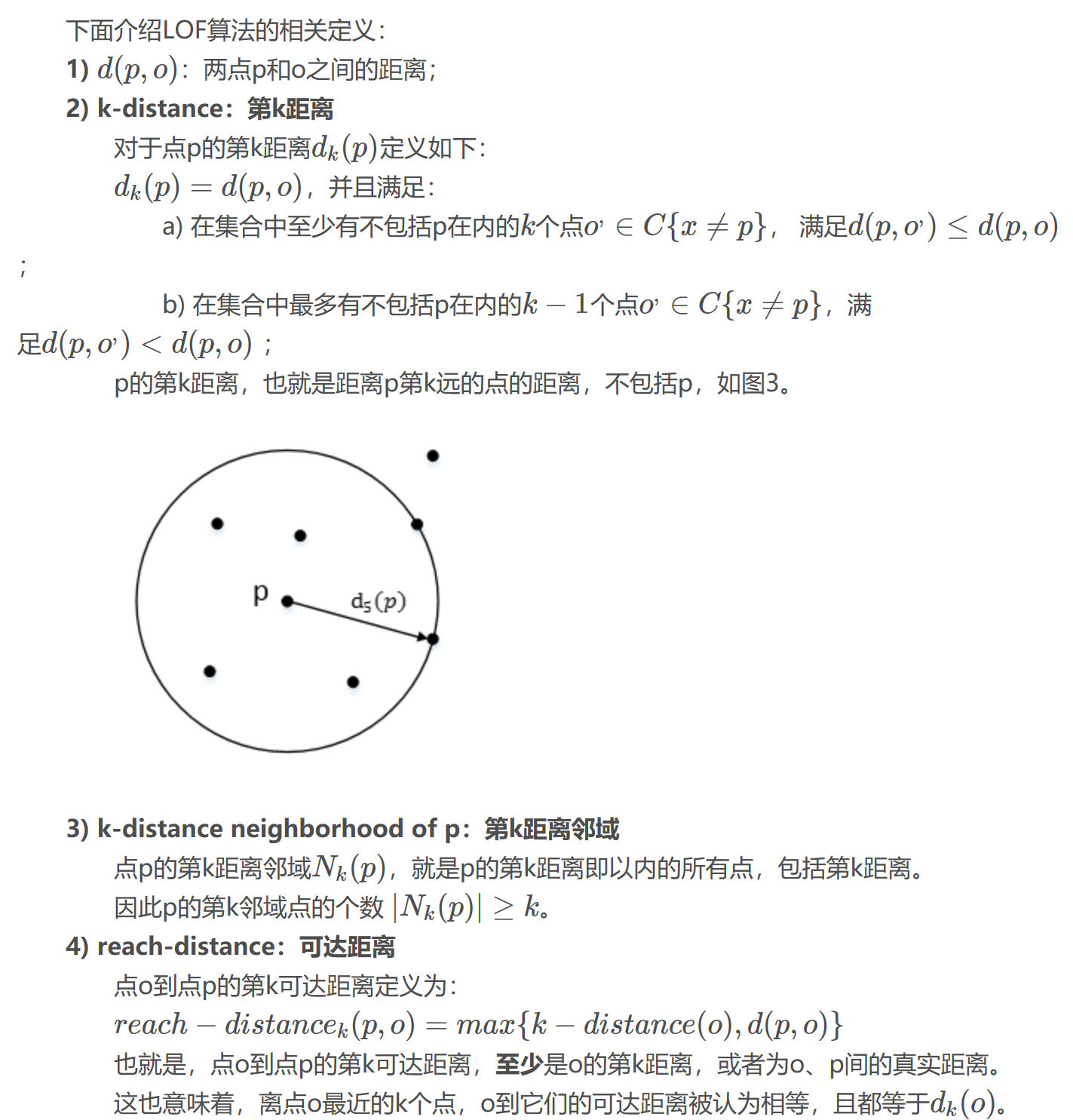
还有一些利用RNN、一维CNN、DNN等方法检测异常数据的方法，属于极少数，暂不做解释。

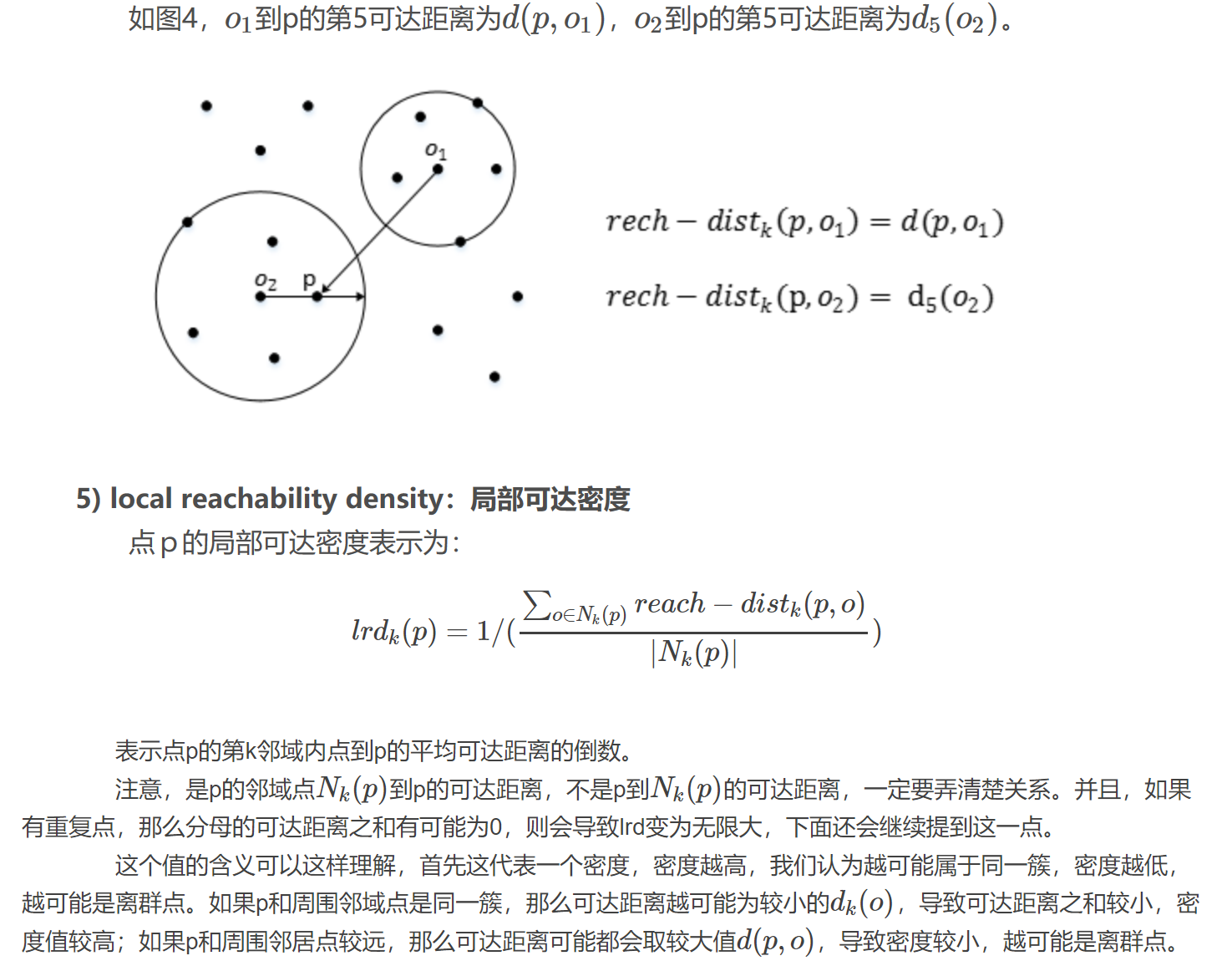
# 预备知识

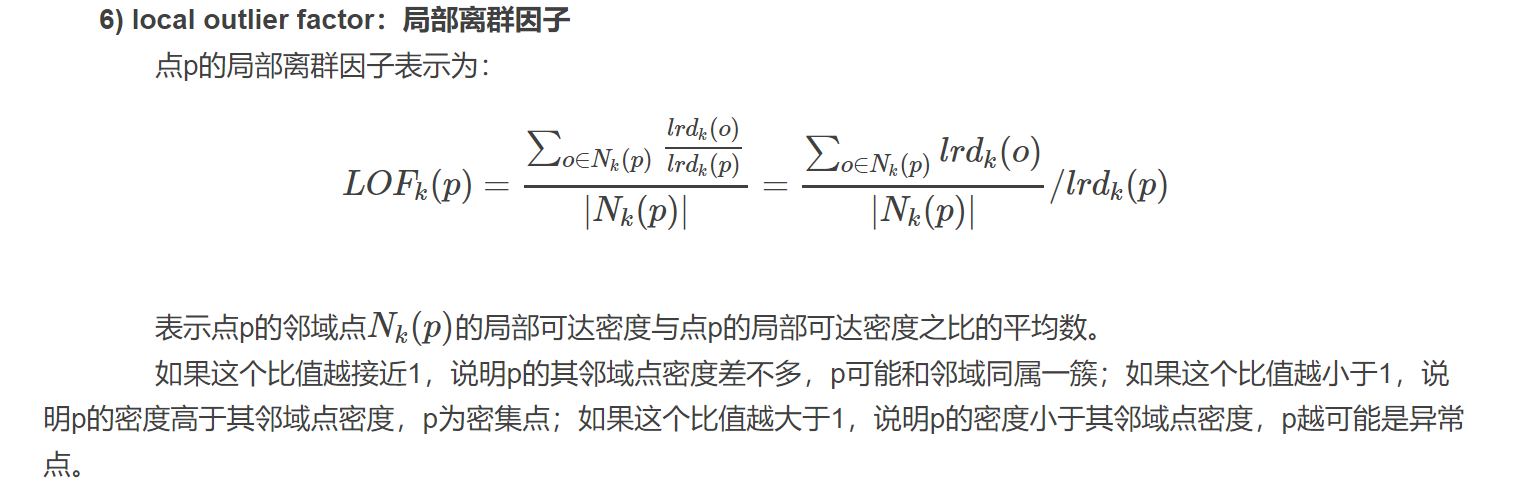
**一、局部异常因子算法-Local Outlier Factor(LOF)**

用视觉直观的感受一下，如图2，对于C1集合的点，整体间距，密度，分散情况较为均匀一致，可以认为是同一簇；对于C2集合的点，同样可认为是一簇。o1、o2点相对孤立，可以认为是异常点或离散点。现在的问题是，如何实现算法的通用性，可以满足C1和C2这种密度分散情况迥异的集合的异常点识别。LOF可以实现我们的目标。







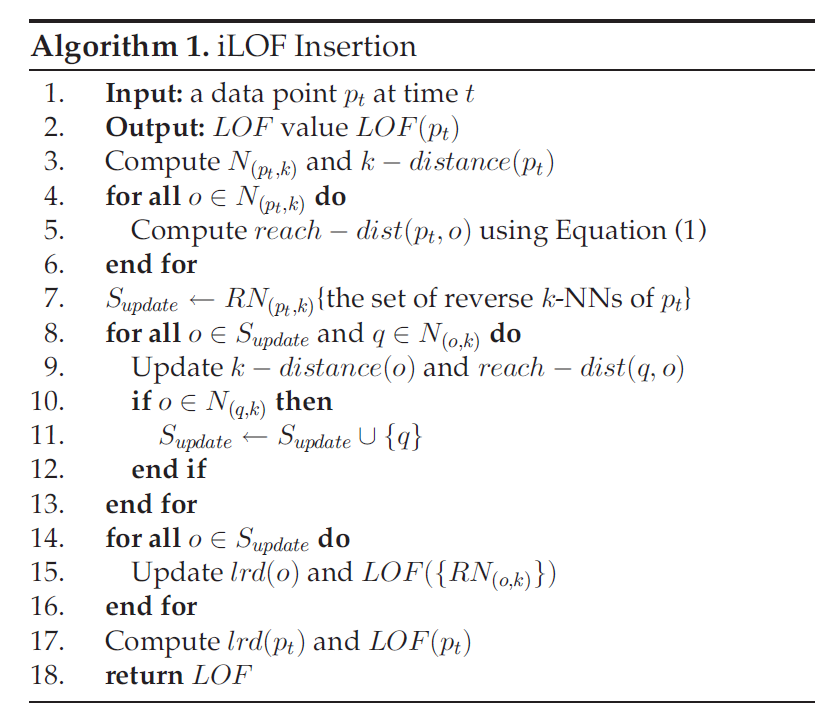


**2.iLOF算法简介**

Ilof算法核心详细见‘Incremental Local Outlier Detection for Data Streams’

Ilof算法的设计目的主要是为了降低时间复杂度。其核心思想是：在原始LOF 算法中，每当新插入一个数据点时，则所有数据集中数据的k-distance、lrd、LOF值都得重新计算，而在ilof算法中发现，每当插入一个新的数据的时候，只会对其附近的数据有影响，所以每次只需要跟新部分数据即可，不用全局数据更新，从而大大减少计算时间。

算法中提出了插入和删除两种操作，但是插入算法这一部分效果更佳显著，而删除算法这一块自身存在较多漏洞，这里不再过多描述。算法中有太多符号意义，具体参阅上述论文。



这是其算法步骤，具体细节参考给出的论文。第7行中的RN的含义时pt在某个数据点的k邻近距离之内。

**3.milof算法略讲**

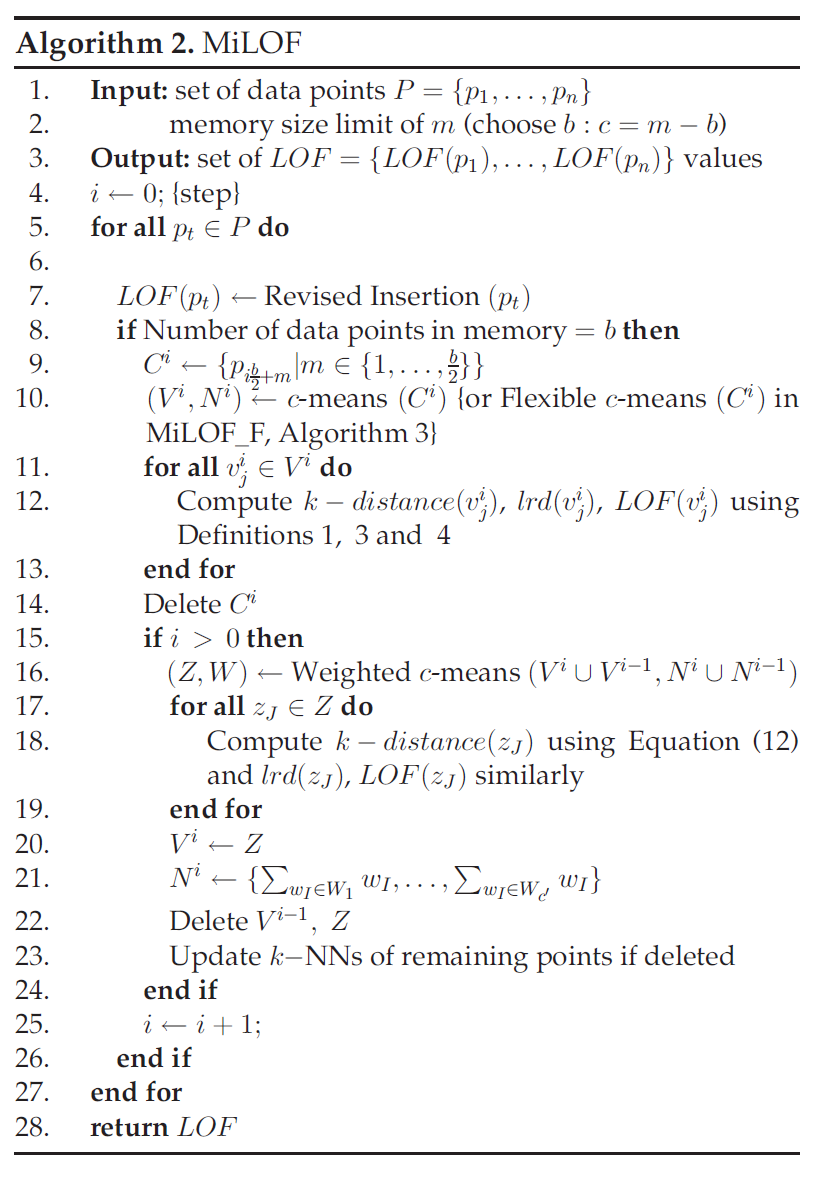
Milof算法详见于：‘Fast Memory Efficient Local Outlier Detection in data streams’这篇论文。

Milof算法算法的重要贡献点在于大大降低了内存占用问题，因为在传统的LOF或者Ilof算法中，所有的数据都需要被存储起来，并且随着新数据的插入，不断有新的数据值产生，时间过久，会造成内存的大量占用，这对于内存来说是一个巨大的挑战，尤其是本身存储容量较小的传感器这类装置。

所以Milof提出了一种基于K-means聚类来降低数据量的算法；

具体参数意义参阅上述论文。





该算法最大的贡献就是让内存占用大小永远保持在一个固定值（内存分配给该部分的大小）。

DILOF算法简单分析

论文：‘DILOF：EFFective and Memory Efficient Local Outlier Detection in Data Streams’

一、算法理解

论文中算法名称为Density summarizing incremental LOF(DILOF)，该算法主体结构是基于LOF,MILOF,ILOF三大算法改进而来，而MILOF,ILOF两种又是基于LOF算法改进而来，具体关于该三种算法介绍在上文中已有介绍。

DILOF算法的主要解决问题有以下两个：

1：降低内存，减少内存占用量。这对于一些小容量的传感器来说及其重要，这一部分主要基于MILOF进行改进而来

2：降低时间复杂度，这一部分基本完全借鉴iLOF的插入算法。

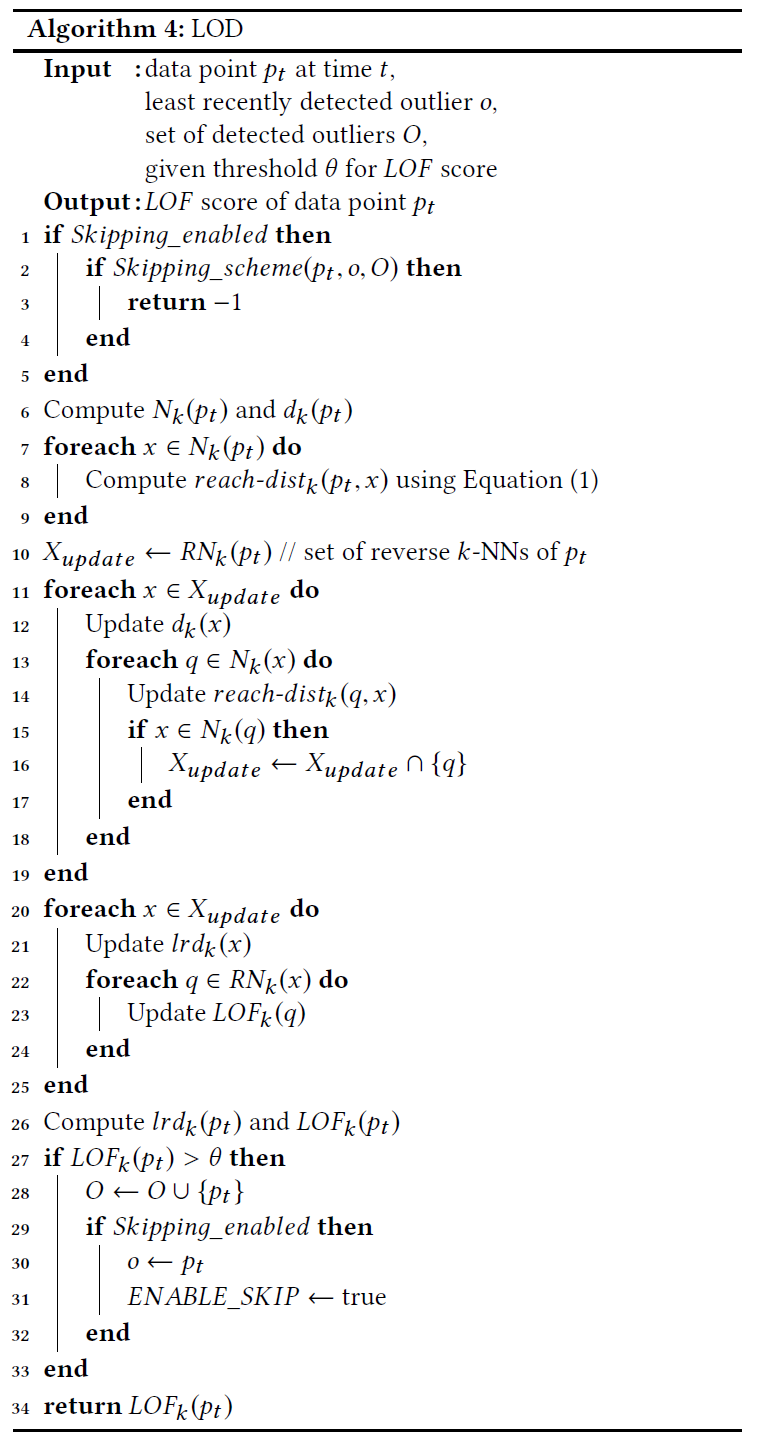
为解决这两个问题，算法在设计上分为两大部分：

（1）：detection phase

（2）：summarization phase

二、对（1）（2）两阶段算法进行详细讲解

对detection phase算法进行分析讲解：这一部分算法在iLOF的插入算法的基础上增加了一个检测连续异常数据的算法，利用新插入点于最后一个异常点的距离来判断数据异常与否，从而较好的解决了iLOF算法在连续异常数据检测方面的弱势。



对summarization phase阶段，改进了Milof算法。在缓存数据阶段没有利用K-means计算聚类中心点缓存数据，而是利用离散特性，利用离散方程构造非约束规划问题，对前M/2数据选出其中M/4个数据，从而保证数据集大小永远固定在M。这一部分也是论文的创新点所在。可以参阅‘Fast Memory Efficient Local Outlier Detection in data streams’这篇论文。