一种基于 BIRCH的异常检测技术*

黄 斌[1] 史 亮[2] 陈德礼[1]
(1.莆田学院 电子信息工程系,福建 莆田 351100; 2.厦门大学 软件学院,福建 厦门 361005)

〔摘要〕 文章针对 KNN存在的复杂度过高的问题,提出应用把 BIRCH算法的层次聚类思想近似地计算 weight的 BirchOut算法,以降低其复杂度,同时利用孤立点挖掘的思想做异常检测.通过在 KDD99数据集上的实验,我们验证了算法的有效性.

〔关键词〕 BIRCH算法; BIRCHOUT算法; 异常检测; 入侵检测
〔文章编号〕 1672-2027(2008)04-0055-04 〔中图分类号〕 TP391 〔文献标识码〕 A

0 引言

异常检测是当前入侵检测研究领域的热点,主要是通过检测用户的异常行为来发现入侵事件.异常检测的优点就是能够发现未知的攻击类型,但同时也存在误报率高、建立正常行为模型不易等问题[1].通过分析我们认为,异常检测所存在的这些问题主要是由于训练样本集的不完备性所导致,由于在实际中,我们无法获得包含所有正常行为的训练集合,因此依据这种不完备的样本所建立的正常行为模型仅是真正正常行为模型的一个子集,而以此为标准进行异常检测就不可避免地会产生误报.如果我们在进行异常检测时,能够绕过正常行为样本的不完备性这一问题,利用入侵行为和正常行为存在的差异进行入侵检测,是否有可能有利于解决当前异常检测误报率高的问题呢? 基于上述考虑,我们尝试将孤立点挖掘算法应用于入侵检测中,这里我们主要的依据是以下两个认识:

* 正常行为和异常行为存在明显差别;
* 在现实应用中,异常行为的数量要远低于正常行为的数量.

从这两个认识可以看出:入侵行为相当于数据集中的孤立点,因此入侵检测的问题可以转换为孤立点挖掘的问题.

1 孤立点挖掘

孤立点挖掘是数据挖掘的任务之一,它在许多不同的领域中都有实际的应用.孤立点挖掘可以描述如下:"给定一个 n个数据点或对象的集合,及预期的孤立点数目 k,发现与剩余的数据相比是显著相异的、异常的或不一致的头 k个对象"[2].很多数据挖掘算法把孤立点当作噪声加以排除,以提高结果的准确性.但是由于"一个人的噪声可能是另一个人的信号"[2],因此孤立点本身可能是非常重要的.孤立点挖掘可以被用于许多领域,比如电信和信用卡欺骗的检测等.

孤立点挖掘通常分为三类:基于分布的方法、基于距离的方法和基于偏离的方法.在本文中,我们采用的是基于距离的 KNN(K-Nearest Neighbour)方法,所谓 weight是用来衡量一个数据是孤立性的度量,它被定义为离该点最近的 k个邻居的距离之和[3].从权值的定义上可以看出:如果要精确地计算出权值,我们需要计算每两个点之间的距离,然后对每个点选出最近的 k个点,计算复杂度为 $O(n^2)$,这在大数据集中是不可行的,因此,我们通过近似算法来减少计算量.下面我们介绍一种基于 BIRCH的算法来近似地计算 KNN,以降低复杂度.

* 收稿日期: 2008-09-14
基金项目:福建省自然基金(2008F50602);福建省省青年人才项目(2008F3101).
作者简介:黄 斌(1981-),男,福建莆田人,硕士,莆田学院电子信息工程系助教,从事数据挖掘、入侵检测研究.

2 BirchOut算法

BirchOut算法是一种先使用BIRCH算法聚类,然后通过算法生成的CF树来近似地计算weight的KNN算法.

所谓聚类又称无指导的学习,是一个将数据库中的数据划分成具有一定意义的子聚类,使得不同子聚类中的数据尽可能相异,而同一子聚类中的数据尽可能相同的过程.迄今为止,人们提出了许多数据聚类的算法,聚类算法一般分为分割和分层两种.分割聚类算法通过优化一个评价函数把数据集分割为 K 个部分(K 为聚类个数),如 K-means[1]算法、K-medoids算法、CLARANS[2]算法;分层聚类是由不同层次的分割聚类组成,层次之具有嵌套的关系间的分割,其中的代表有 BIRCH[3]算法、DBSCAN算法和 CURE算法.在面向大型数据库和超大型数据库方面,目前主要采用了 BIRCH CURE等算法[4].

2.1 BIRCH与 CF-Tree

BIRCH是一个综合的层次聚类方法.它引入了两个概念:聚类特征和聚类特征树(CF),它们用于概括聚类描述.这些结构辅助聚类方法在大型数据库中取得高的速度和可伸缩性.BIRCH方法对增量或动态聚类也非常有效.所谓聚类特征(CF)是一个三元组,给出对象子聚类的信息汇总描述.

CF项:(N, LS, SS),其中 N 表示子类中点的数目,LS 是 N 个点的线性和,SS 是数据点的平方和,从统计学的观点来看,聚类特征是对给定子聚类的统计汇总:子聚类的0阶矩,1阶矩和2阶矩.它记录了计算聚类和有效利用存储的关键度量,因为它汇总了关于子聚类的信息,而不是存储所有的对象[4].图1给出一个 CF树的结构示例,根据 CF-Tree的定义,树中的非叶子节点包含孩子节点,他们存储了其孩子节点的 CF值的总和,即包含了其孩子节点的聚类特征信息,其中结点的每个聚类特征和它指向子节点的指针组成条目(Entry). CF-Tree包含两个参数:分支因子 B 和阈值 T.分支因子 B 限定了每个非叶子节点最多含有的孩子的个数,即每个非叶子节点最多包含 B 个条目;T 限定了存在叶子节点的簇 的最大半径(或直径),这两个参数影响了最终产生的树的大小,另外,每个叶子节点至多包含 L 个 CF向量,同一叶子节点中的所有数据点都满足阈值 T[5].

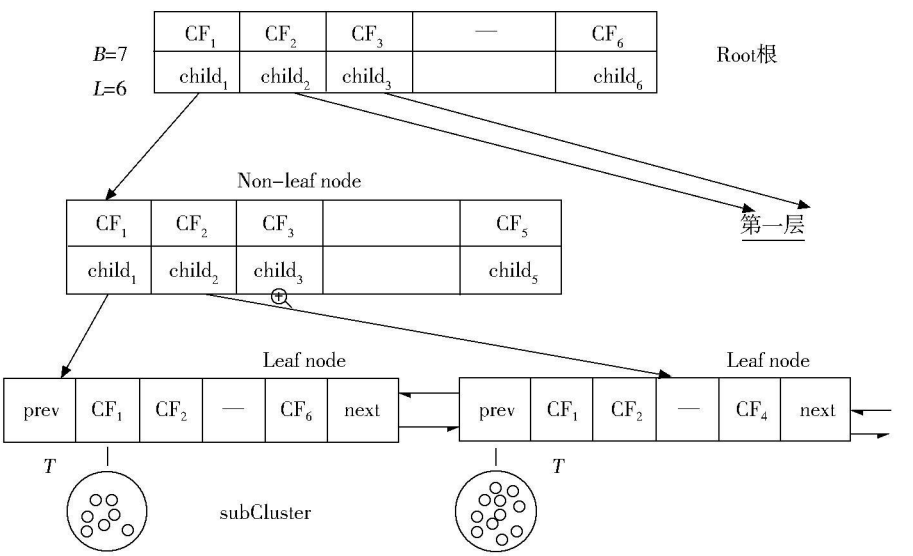

图1 CF树结构

Fig. 1 Construction of CF tree

BIRCH算法利用 CF-Tree结构对数据集进行聚类,算法主要分为两个阶段:阶段一,对整个数据集进行扫描,建立一棵初始化的聚集特征树(CF-Tree),尽可能地包含所有属性的信息;阶段二,用聚集特征代替原有数据集进行聚类.在第一阶段,聚集特征树是随着对象一个一个地加入而自动形成的:一个对象被放入那个离它最近的叶子节点(簇)中去.如果放入以后这个簇的半径大于阈值 T 的话,那么这个叶节点就会被分割(Split).新对象插入以后,关于这个对象的信息向根节点传递.插入过程类似于 $B+$ 树构建中的插入和节点分裂[5].

由于CF树的层次性,我们可以近似地认为:在同一个子聚类内,数据之间的距离是最近的,利用这一点,我们可以用近似算法来避开直接精确计算 weight的复杂度.

2.2 数据预处理

在运用聚类算法之前,因为属性值之间由于采用不同的度量单位,其差别可能很大,造成对数据间距离

的影响也不同 ,为了减少这种差别带来的影响 ,我们预先对数据进行标准化.

$$x_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j}$$

其中 ,均值 $\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}$,标准偏差 $s_j = \frac{1}{n-1}\sum_{i=1}^{n} (x_{ij} - \bar{x}_j)^2$.经过标准化变换后 ,各特征属性的均值为 0,方差为 1.

2.3 算法描述

BirchOut算法的整体流程见图 2所示 .其中计算 weight的方法可以描述如下:

设 K 表示点 p 的最近邻居集 ,X 表示候选的子聚类

1) $K = \varnothing$;

2) c= 根结点;

3) X= {c的所有直接孩子结点};

4) c= X 中离数据点 p 最近的 CF项;

5)如果 $cn + |K| <= k$;

i) $K = K + \{c$子聚类中的所有数据};

ii)如果 $|K| = k$,退出;

iii) $X = X - c$;

iv)重复 4;

6)否则 ,重复 3;

7)计算 p 与 K 中的所有数据的距离之和.

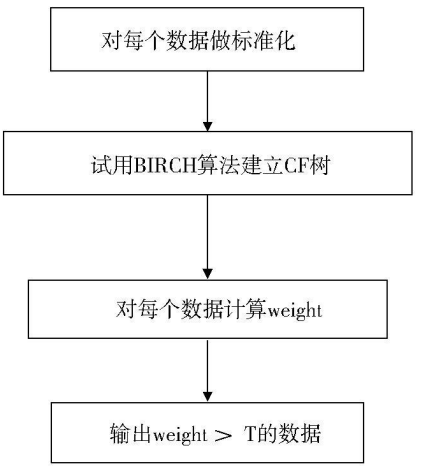

图 2　算法整体流程

Fig. 2　Process of algorithm

3 实验

KDDCup99[6]的数据集中包含了 41条属性 ,在本文中我们参考了文献 [7]中的结果 ,选取了如下属性 dst_host_count, dst_host_srv_count, dst_host_same_srv_rate, count, dst_host_same_src_port_rate, protocol_type, srv_count, same_srv_rate.除了 protocol_type以外都是数值属性 ,把 protocol_type转化为三个二元属性: “is_tcp” ,“is_udp” 和“is_icmp” ,即如果 protocol_type= tcp,转化后 is_tcp= 1, is_udp= 0, is_icmp= 0.

在基于 KNN的算法中的有 2个参数: 邻居数 k 和阈值 T.在实验中 ,k的取值 ,因为入侵行为的数量在整个数据集中所占的比例在 1% 左右 ,为了避免入侵行为自相似 ,所以 k取整个数据集数量的 2%～ 3% (在实验中一般取 1 200).在我们的实验中 ,孤立点的定义就是: 所有 $w > T$ 的数据点 ,而参数 T 该如何取值? 我们通过实验的方法来选择一个较为合适的 T.

表 1　检测率与参数 T 的关系

Table 1　The relationship between detection rate and parameters T

T	检测率 %	误报率 %
4 000	99.49	3.95
4 500	99.49	2.64
5 000	99.49	2.30
5 500	98.34	2.02
6 000	89.24	1.69

从表 1中可以看出: 当 T= 5 000时 ,检测率和误报率都比较理想 (分别为 99.49%和 2.3%),之后的实验都设 T= 5 000.

在下面的实验中 ,我们将分别抽取 KDD99的 4大类攻击: DOS(拒绝服务攻击) R2L(从远程计算机进行非授权的访问) U2R(非授权得到超级用户权限或运行超级用户函数)和 Probing(扫描或者对其他系统漏洞的探测) ,来测试算法的检测率和误报率 ,见表 2

表2 各种类型攻击检测率和误报率

Table 2 Result of detection for various types of attack

类型	BirchOut		K-Mean	
	检测率%	误报率%	检测率%	误报率%
DOS	99.49	2.30		
Probe	70.78	2.44		
R2L	33.33	3.37		
U2R	16.67	3.46		

4 结论

本文针对目前异常检测技术所存在的问题,基于对入侵行为的两点认识,将孤立点挖掘技术引入到异常检测领域.在具体实现上,本文提出了一种基于BIRCH的孤立点挖掘算法BirchOut,该算法可以高效近似地计算weight,适合对入侵进行检测分析.我们将算法应用在KDDCup99数据集检测分析,取得不错的实验结果.

参考文献:

[1] 戴英侠,连一峰,王 航.系统安全与入侵检测[M].北京:清华大学出版社,2002

[2] Han Jiawei, Micheline Kamber. Data mining concepts and techniques[M]. Morgan: Kaufmann Publishers, 2000

[3] Angiulli F, Pizzuti C. Outlier mining in large high-dimensional data sets[J]. IEEE Transactions on Knowledge and Data Engineering, 2005, 17(2): 203-205

[4] Zhang R, Ramakrishnan R, Livny M. BIRCH: An efficient clustering method for every large databases[C]. Montreal, Canada: Proc. of the International Conference Management of Data, 1996: 101-114

[5] 邵峰晶,张 斌,于忠清.多阀值BIRCH聚类算法及其应用[J].计算机工程与应用,2004(12): 177-179

[6] KDD99 Cup dataset. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. 1999

[7] Wing W Y NG, Rocky K C Chang, Daniel S. Young. Dimensionality reduction for denial of service detection problems using RBFNN Output[C]. Xi'an: The 2nd Int'l Cnnf on Machine Learning and Cybernetics, 2003: 198-202

An Anomaly Detection Method Based on Birch Algorithm

Huang Bin[1] **Shi Liang**[2] **Chen Deli**[1]

(1. Electronic Information Engineering Department, Putian University, Putian 351100;
2. Software School, Xiamen University, Xiamen 361005, China)

〔**Abstract**〕 A new algorithm called BirchOut is proposed to reduce the complexity of KNN calculation, which uses the idea of hiberarchy clustering, presents an anomaly detection method by using an outlier detection method. We apply this technique on KDD99 data set and get satisfactory results.

〔**Key words**〕 BIRCH algorithm; BIRCHOUT algorithm; anomaly detection; intrusion eletection