# 一种融合 Kmeans 和 KNN 的网络入侵检测算法

华辉有 陈启买 刘 海 张 阳 袁沛权
(华南师范大学计算机学院 广州 510631)

**摘 要** 网络入侵检测算法是网络安全领域研究的热点和难点内容之一。目前许多算法如KNN、TCMKNN等处理的训练样本集都比较小,在处理大样本集时仍然非常耗时。因此,提出了一种适应大样本集的网络入侵检测算法(Cluster-KNN算法)。该算法分为离线数据预处理(数据索引)和在线实时分类两个阶段:离线预处理阶段建立大样本集的聚簇索引;在线实时分类阶段则利用聚簇索引搜索得到近邻,最终采用KNN算法得出分类结果。实验结果表明:与传统的KNN算法相比,Cluster-KNN算法在分类阶段具有很高的时间效率,同时在准确率、误报率和漏报率方面与其它同领域入侵检测方法相比也具有相当的优势。Cluster-KNN能够很好地区分异常和正常场景,且在线分类速度快,因而更适用于现实的网络应用环境。

**关键词** 网络入侵检测,Kmeans,KNN,KDDCUP99
**中图法分类号** TP393.08 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2016.3.030

## Hybrid Kmeans with KNN for Network Intrusion Detection Algorithm

HUA Hui-you CHEN Qi-mai LIU Hai ZHANG Yang YUAN Pei-quan
(School of Computer Science, South China Normal University, Guangzhou 510631, China)

**Abstract** Network intrusion detection algorithm is one of the hot and difficult topics in the field of network security research. At present, many algorithms like KNN and TCMKNN, which process relatively small data samples, are still very time-consuming when processing large scale date set. Therefore, this paper put forward a hybrid algorithm (Cluster-KNN), which is adaptive to large scale data set. The algorithm is divided into the offline data preprocess phase (data indexing) and the online real-time classification phase. The offline phase establishes the cluster index for the large data set. Then the online phase uses the index to search neighbors, and finally outputs the result by KNN algorithm. The experimental results show that compared with the traditional KNN algorithm, Cluster-KNN algorithm has high time efficiency in the classification phase, and it has considerable advantages as well compared to intrusion detection methods of the same field in the accuracy rate, false positive rate, false negative rate and other aspects. Cluster-KNN can clearly distinguish the abnormal and normal scenes, and it has a high online classification speed. Thus, it is more suitable for the real network application environment.

**Keywords** Network intrusion detection, Kmeans, KNN, KDDCUP99

## 1 引言

随着互联网在军事、金融、电子商务等领域的大规模应用,越来越多的主机和网络正受到各种类型的网络入侵攻击的威胁。实时的、智能的入侵检测算法能够有效地检测出网络入侵事件,从而及时地将入侵行为带来的破坏降到最低。文献[1-5]将近邻算法思想应用到网络入侵检测中,分别提出了KNN、TCMKNN、LVQ-KNN、LBKNN及NSM-KNN等入侵检测方法,但这些算法所处理的训练样本集都比较小,在处理大样本集时仍然非常耗时。随着许多新的网络入侵方式的产生和更新换代,势必要不断地将许多新的样本添加到训练集中,样本集变得越来越大,与此同时算法处理的时间也会大大增加,从而很难满足入侵检测实时性的要求。因此,本文研究在大样本集的条件下一种融合Kmeans和KNN的入侵检测算法:通过Kmeans算法建立大样本集的聚簇索引,提高近邻搜索速度,从而提高算法的实时分类速度;通过KNN算法对待分类样本的$K$个近邻进行加权投票,最终得出分类结果。

本文第2节分别阐述Kmeans和KNN算法的主要思想及其在网络入侵检测领域的应用;第3节介绍融合Kmeans和KNN的网络入侵检测算法(Cluster-KNN),详细描述了该算法的思路和步骤;第4节针对KDDCUP99数据集,采用Cluster-KNN算法和其他一些相关算法进行若干项对比实验,并分析实验结果;最后对本文工作进行总结和展望。

到稿日期:2015-01-19 返修日期:2015-04-18 本文受广东省教育部产学研结合项目(2009B090300326),华南师范大学研究生科研创新基金项目资助。

**华辉有**(1990—),男,硕士,CCF会员,主要研究方向为数据挖掘、网络安全,E-mail:benfenghua@foxmail.com;**陈启买**(1965—),男,教授,主要研究方向为数据挖掘、数据仓库;**刘 海**(1974—),男,副教授,主要研究方向为语义Web、数据挖掘、个性化推荐;**张 阳**(1991—),女,硕士,主要研究方向为云计算、个性化推荐;**袁沛权**(1991—),男,硕士,主要研究方向为软件工程、个性化推荐。

## 2 Kmeans 和 KNN 算法

### 2.1 Kmeans 算法

Kmeans 算法是一种经典的聚类算法。该算法主要是将样本按照相似度聚集到 $K$ 个聚簇当中，最终簇内相似度高，簇间相似度低。主要过程如下：首先随机初始化 $k$ 个样本作为 $K$ 个簇初始质心；然后将所有样本指派到最相似的簇，重新计算每个簇的质心；重复上述指派过程直到停止条件被满足。通常采用平方误差准则及最大迭代次数限制作为停止条件。最大迭代次数条件与数据集的大小和聚集程度相关，平方误差定义如式(1)所示。

$$E=\sum_{i=1}^{K}\sum_{p\in C_i}(p-c_i)^2 \tag{1}$$

其中，$p$ 是样本空间中的样本，$c_i$ 是簇 $C_i$(Cluster)的质心(centroid)。

一般地，平方误差达到最小时，聚簇结果满足条件 T1：

$$\forall p\in PC, distance(p, getCluster(p))\leqslant \min_{0<i\leqslant M}\{distance(p,C_i)\} \tag{T1}$$

其中，$p$ 表示样本，$PC$ 表示样本集合，$distance()$表示样本与聚簇中心的距离，$getCluster()$表示样本所属聚簇中心，$M$ 表示聚簇个数，$C_i$ 表示第 $i$ 个聚簇。

文献[6,7]等将 Kmeans 算法应用到网络入侵检测当中，采用无监督的聚类方法，对 KDD Cup99 数据集进行聚类。分类待测样本时，考察待测样本所属聚类中的样本的类型和比例，最终确定待测样本的类型。文献[8,9]则将基于 Kmeans 聚类的算法应用到数据划分(Data Partitioning)中，将大数据集划分到一系列的聚簇中，然后将聚簇质心作为其中样本的索引。这样加快了样本的搜索效率，在处理大数据集时具有很大的时间优势。

### 2.2 KNN 算法

KNN 算法是一种经典的分类算法。该算法主要是从待分类样本最近的 $K$ 个样本中得出其分类结果。主要过程是：将所有训练样本存放在 $n$ 维模式空间中；对于待分类样本，搜索该模式空间，根据样本邻近性找出最近的 $K$ 个训练样本，作为其 K 近邻；最后根据 K 近邻采用某种投票策略预测待分类样本类型。其中，样本的邻近性主要有两种度量方式：欧几里得距离和余弦相似度，计算公式分别如式(2)和式(3)所示。

$$distance(P_1,P_2)=\sqrt{\sum_{i=1}^{n}(p_{1i}-p_{2i})^2} \tag{2}$$

$$Cos(P_1,P_2)=\frac{\sum_{k=1}^{n}p_{1k}p_{2k}}{\sqrt{\sum_{k=1}^{n}p_{1k}^2}\sqrt{\sum_{k=1}^{n}p_{2k}^2}} \tag{3}$$

作为一种懒惰型学习算法，KNN 系列算法在模式学习阶段只需要存储训练样本，耗费时间可以忽略不计。而在实时分类阶段，则需要遍历整个模式空间，最终得出待测样本的类型，如果模式空间过大，则每次分类耗费时间增大，将不能满足实时分类的需求。这是 KNN 这类懒惰型学习算法相比其他积极学习算法固有的缺陷之一。

文献[1-5]将 KNN 系列算法应用到网络入侵检测中，分别采用传统 KNN 和改进决策方式、近邻性度量方式以及优化特征向量权重的 KNN 算法。这些算法的正确率均达到 97%以上，但是它们采用的数据集都是 $10^4$ 量级大小的，在处理大数据集(大于 $10^6$ 量级)时存在时间瓶颈，未解决 KNN 算法的上述缺陷。

## 3 融合 Kmeans 和 KNN 的网络入侵检测算法

目前，在网络入侵检测算法研究中，较为热门的方向是通过融合并改进若干传统的算法，提高检测的正确率、降低误检率和漏检率。例如文献[10-14]分别将 Kmeans 与 Naïve Bayes、SVM、OneR 等算法进行融合，Kmeans 用于对训练数据进行聚类、提取特征及归约等预处理，Naïve Bayes、SVM、OneR 等则用于产生分类结果。文献[15,17]则对 KPCA、PSO、SVM 等算法进行了融合，KPCA 用于特征选择，PSO 用于分类器参数的选取和优化，SVM 则用于构建分类器。文献[16]则融合无监督和有监督的机器学习方法，提出了多层次的混合入侵检测方法，根据不同入侵类型的特点分层次地采用有监督和无监督的算法识别对应的类型。

本文则结合第 2 节中的 Kmeans 聚类算法和 KNN 近邻算法，提出了融合算法——Cluster-KNN 算法。如图 1 所示，算法主要分为两个阶段：离线数据预处理(数据索引)阶段和在线实时分类阶段。主要思路如下：首先运用基于 Kmeans 聚类的数据划分算法(A1-1)，将大数据集划分成若干聚簇 Clusters，并计算聚簇中心 centroids；然后运用近邻搜索算法(A1-2)从聚簇 Clusters 中获取对应的 K 近邻；最后对 K 近邻用基于 KNN 的近邻决策算法(A2)得出最终的分类结果。

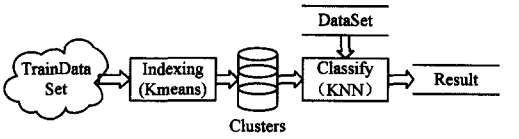


图 1 Cluster-KNN 算法流程

### 3.1 离线数据预处理阶段

本阶段的主要工作是对大数据集进行划分索引(partitioning indexing)。常用的数据索引方式主要有数据划分(Data partition)和空间划分(Space partition)两种[9]。本文采用的基于 Kmeans 的划分算法属于数据划分，最终将大数据集划分成大小均衡的若干个聚簇，聚簇满足条件(T1)。主要过程如算法 A1-1 所示。

**算法 A1-1** Data partioning indexing using Kmeans

输入：训练集 DTrain，聚簇 K 初始值，聚簇最大约束 Vmax，聚簇最小约束 Vmin，最大迭代次数 $T_{max}$

输出：聚簇样本集 Clusters[]，聚簇中心 centroids[]

```
DataPartition(DTrain,K,Vmax,Vmin,Tmax)
Begin
1. init centroids[K],t=0;//初始化聚集中心
2. Clusters[]=Kmeans(DTrain,centroids[]),t++;//一次聚集到满足条件(T1)
3. while(t<Tmax&&∃Clusters[i].count∉(Vmin,Vmax))
4. if(Clusters[i].count>Vmax)//聚簇 Clusters[i]中样本数量大于 Vmax
     Generate Clusters[j] whose centroid centroids[j] is the item in
     Clusters[i] which is farest from centroids[i];//形成新聚簇
   End if
5. If(Clusters[i].count<Vmin)
     Delete Clusters[i] and add its items to other clusters;//删除小聚簇
   End if
6. Clusters[]=Kmeans(DTrain,centroids[]);t++;//一次聚集到满足条件(T1)
   End while
7. Save Clusters[] and centroids[] //保存聚簇 Clusters[]及其聚簇中
```

心 centroids[]
End DataPartion

运用算法 A1-1 得到聚簇索引后，获取 K 近邻的方式是：选择满足条件的聚簇作为候选聚簇，然后再从候选聚簇中选取样本的 $K$ 个近邻。具体过程如算法 A1-2 所示。其中候选聚簇的选择过程如下：①首先选择最近聚簇作为第一个候选聚簇 $C_1$；②其他聚簇 $C_i$ 如果满足条件(T2)中的任意一条则加入到候选聚簇集合中。

$$distance(inst,C_i)\leqslant \max Radio(C_i)$$

$$distance(inst,C_i)\leqslant 1/2(distance(C_1,C_i))$$

$$0\leqslant distance(inst,C_i)-\max Radio(C_i)\leqslant distance(inst,C_1)-\max Radio(C_1) \quad (T2)$$

其中，$distance()$ 如条件式(T1)定义，$\max Radio()$ 表示聚簇中离聚簇中心最大的距离。

**算法 A1-2** k-NNSearch

输入：样本 inst，聚簇集 Clusters[]，近邻个数 K
输出：K 个近邻样本 insts
NNSearch(Clusters[],centroids[],inst,K)
Begin
1. init CanSet[],MaxHeap;
//初始化候选聚簇集 CanSet[]，大根堆 MaxHeap
2. foreach Clusters[i] in Clusters[]
//找出最近的聚簇
If(centroids[i] is nearest to inst) add Clusters[i] to CanSet[],delete Clusters[i] from Clusters[]
End if End for
3. foreach Clusters[i] in Clusters[]
//获取候选聚簇集
If(Clusters[i] satisfies condition (T2)) add Clusters[i] to CanSet[],delete Clusters[i] from Clusters[]
End if End for
4. foreach Clusters[i] in CanSet[]
//从候选聚簇集中获取 K 近邻
Foreach(inst1 in Clusters[i]
5. if(distance(inst1,inst)<MaxHeap. first) add inst1 to MaxHeap;
End if
6. if(MaxHeap. size>K) remove MaxHeap. first;End if
End for End for
7. return MaxHeap. Items//返回大根堆中的元素
End NNSearch

### 3.2 在线实时分类阶段

本阶段主要是完成在线实时的样本分类，对实时性和准确率有很高要求。对于待测样本的 $K$ 个近邻，统计其中的类型及其数量，最终取数量值最大的那个类型作为分类结果输出。具体如算法 A2 所示。

**算法 A2** KNNClassify

输入：样本 inst，聚簇 cluster，近邻个数 K
输出：inst 的分类
KNNClassify(inst,Clusters[],centroids[],K)
Begin
1. init classMap[Class];//初始化(类型，数量)对
2. KNN[]=NNSearch(Clusters[],centroids[],inst,K);
3. foreach(inst1 in KNN[]) classMap[inst1. class]++;
End for
4. Find the max classMap[result];
5. return result;
End KNNClassify

### 3.3 算法复杂度分析

上述 3 个算法的时间和空间复杂度如表 1 所列。

表 1 Cluster-KNN 算法的时间和空间复杂度

| 算法 | A1-1 | A1-2 | A2 |
|---|---|---|---|
| 时间复杂度 | T1=O(t*n*c) | T2=O(m*n/c) | T3=T2+O(k) |
| 空间复杂度 | S1=O(n+c) | S2=O(n+c) | S3=O(n+c) |

其中，$t$ 是聚簇迭代总次数，$n$ 是训练样本的数量，$c$ 是聚簇的数量($c\ll n$)，$m$ 是候选聚簇数量，$K$ 是近邻数量。

当 $m/c\ll 1$ 时，近邻搜索的时间复杂度远小于传统的 KNN 算法的时间复杂度 $O(n)$。另外 Cluster-KNN 算法两个阶段均需要存储所有的训练样本，因此空间复杂度均为 $O(n+c)$，这与标准 KNN 算法的空间复杂度 $O(n)$ 近似。

## 4 实验及分析

### 4.1 实验环境和评估标准

本文采用网络入侵检测算法研究领域的基准数据集 KDD Cup 1999，其中有约 490 万条数据，包含 4 类共 36 种攻击类型，每条数据有 1 个类别标签和 41 个属性，其中名词型属性 9 个、连续型属性 32 个。文献[18-20]对数据集进行了详细的分析和说明。本文实验环境是 i5 3. 3GHz 4 core CPU、6GB DDR3 内存的 Windows Server 2008 虚拟机。程序采用单线程实现，只用到了 CPU 的一个核心运行计算过程。

本文采用正确率、误报率、漏报率和总耗费时间作为评估指标。定义如式(4)—式(7)所示。

$$\text{正确率：}CR=\frac{TP+TN}{TP+TN+FP+FN} \quad (4)$$

$$\text{误报率：}FR=\frac{FN}{TP+TN+FP+FN} \quad (5)$$

$$\text{漏报率：}MR=\frac{FP}{TP+TN+FP+FN} \quad (6)$$

$$\text{总耗费时间：}T_{all}=T_{train}+n\times T_{classify} \quad (7)$$

其中，$TP$ 表示分类为正常样本的正常样本数量，$TN$ 表示分类为异常样本的异常样本数量，$FN$ 表示分类为异常样本的正常样本数量，$FP$ 表示分类为正常样本的异常样本数量，$T_{train}$ 表示训练时间，$T_{classify}$ 表示分类每个样本的时间，$n$ 表示样本数量。

在实际网络环境中，“正常报为异常”是可以接受的，而“异常报为正常”是不可接受的。因此算法的选用往往对漏报率和正确率要求较为严格，而误报率则只要达到可接受的水平即可，也就是说在误报率可接受的的情况下，漏报率低、准确率高的算法更有优势。

另外，一次攻击行为持续时间往往不长，需要检测技术具备高实时性，在线分类阶段的时间效率非常重要，因此考察总耗费时间 $T_{all}$ 及其对分类样本数的增量 $T_{classify}$、$T_{all}$ 和 $T_{classify}$ 越小越好。

### 4.2 数据预处理

对于名词型属性如协议类型、服务类型等，按每个取值在取值空间中的频率标准化到[0,1]区间中；然后再对所有属性进行 Z-Score 中心化处理。处理方式如式(8)所示。

$$newValue=\frac{oldValue-mean}{varience} \quad (8)$$

其中，$newValue$ 是处理后的属性值，$oldValue$ 是原属性值，$mean$ 是属性均值，$varience$ 是属性标准差。

数据集经过去重后得到约 108 万条数据，按十折交叉验

证的方法将该数据划分成 10 份。每一份共 107385 个样本，其中正常样本 81281 个，异常样本 26104 个。10 份数据分别作为测试集，其余作为训练集，进行 10 次实验。实验结果如 4.3 节所述。

### 4.3 结果分析

(1)Cluster-KNN 算法的正确率、误报率、漏报率分析

表 2 所列的是 Cluster-KNN 算法进行十折交叉验证的实验结果。其中，$K$ 分别取 1、11、21、31，数值后的“+”号表示四舍五入中的舍去，其它的则表示末尾进入高位(表 3 同)。实验中最高正确率接近 100%(D2：$K=1$，D9：$k=1$)，最低正确率稍低于 99.87%(D10：$k=31$)，误报率和漏报率均不超过 0.09%，最低则接近于 0.00%。

表 2 Cluster-KNN 算法十折交叉验证结果

| | 1 | | | 11 | | | 21 | | | 31 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | CR(%) | FR(%) | MR(%) | CR(%) | FR(%) | MR(%) | CR(%) | FR(%) | MR(%) | CR(%) | FR(%) | MR(%) |
| D1 | 99.96 | 0.02 | 0.03 | 99.91 | 0.04 | 0.05+ | 99.89 | 0.04+ | 0.07+ | 99.88 | 0.05+ | 0.07+ |
| D2 | 100.00 | 0.00+ | 0.00+ | 99.93 | 0.03 | 0.05 | 99.90+ | 0.04 | 0.06+ | 99.88 | 0.05 | 0.07+ |
| D3 | 99.98 | 0.02 | 0.01 | 99.93+ | 0.04 | 0.03+ | 99.91 | 0.04+ | 0.05 | 99.89+ | 0.05 | 0.06 |
| D4 | 99.96 | 0.02 | 0.03 | 99.92 | 0.04 | 0.05 | 99.89+ | 0.04+ | 0.07 | 99.87+ | 0.05+ | 0.08 |
| D5 | 99.98 | 0.01+ | 0.01+ | 99.92 | 0.03+ | 0.05+ | 99.89+ | 0.04+ | 0.07 | 99.87+ | 0.05 | 0.08 |
| D6 | 99.97 | 0.01+ | 0.02+ | 99.93+ | 0.03+ | 0.04 | 99.90+ | 0.04+ | 0.06 | 99.89 | 0.05+ | 0.06 |
| D7 | 99.94 | 0.03 | 0.04 | 99.92+ | 0.04 | 0.04+ | 99.90+ | 0.04+ | 0.06+ | 99.88+ | 0.07+ | 0.05 |
| D8 | 99.99 | 0.01 | 0.00+ | 99.92+ | 0.03 | 0.05 | 99.90 | 0.04 | 0.06+ | 99.88 | 0.05 | 0.07+ |
| D9 | 100.00 | 0.00+ | 0.00+ | 99.92+ | 0.04 | 0.04 | 99.90 | 0.04+ | 0.06 | 99.88 | 0.05+ | 0.07 |
| D10 | 99.99 | 0.01 | 0.01 | 99.92 | 0.03 | 0.05+ | 99.90 | 0.03+ | 0.07+ | 99.87 | 0.05 | 0.09 |
| AVG | 99.97+ | 0.01+ | 0.01+ | 99.92 + | 0.03+ | 0.05 | 99.90 | 0.04+ | 0.06+ | 99.88 | 0.05+ | 0.07 |

(2)Cluster-KNN 算法与标准 KNN 算法的比较

在表 2 所列的十折交叉验证结果中，$K=1$ 时的实验结果均比其它取值更好，这与文献[5]的结论是一致的。因此本文取 $K=1$ 的实验结果在时间开销和实验效果两方面与标准 KNN 算法进行比较。

在时间开销方面，比较 Cluster-KNN 和标准 KNN 在离线训练阶段和在线分类阶段的总时间耗费，如图 2 所示。在训练阶段(待测样本量 $x=0$)，此时标准 KNN 算法的时间耗费趋近于 0，而 Cluster-KNN 则需要一次性耗费 93600s 建立聚簇索引；而在线分类阶段(待测样本量 $x>0$)，Cluster-KNN 的优势就表现得较为突出，只以每 10 万条 135s(2min 15s)的增速增长，但标准 KNN 却以每 10 万条 45000s(12.5h)的增速增长。显然当待测样本量累计超过 $10^6$ 时，Cluster-KNN 算法的时间总开销要远低于标准 KNN 算法。

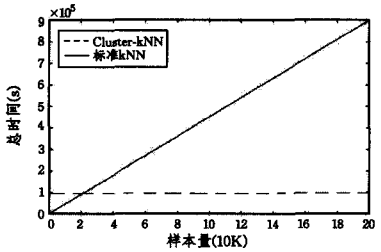


图 2 Cluster-KNN 算法和 KNN 算法总时间耗费比较($K=1$)

在准确率方面，如表 3 所列，经过数据预处理之后，标准 KNN 算法和 Cluster-KNN 算法都取得了较好的准确率、误报率和漏报率，但 Cluster-KNN 算法较标准 KNN 算法略好。

表 3 Cluster-KNN 与标准 KNN 算法的比较

| | 准确率(%) | 误报率(%) | 漏报率(%) |
|---|---|---|---|
| Cluster-KNN | 99.97+ | 0.01+ | 0.01+ |
| 标准 KNN | 99.95+ | 0.02+ | 0.03 |

(3)Cluster-KNN 算法与其它算法的比较

表 4 比较了 Cluster-KNN 算法和文献[4,5,16,21,22]中的算法，其中 MLH-IDS 是根据类别特性提出的多层次的分类算法；NSM-KNN 改进了 KNN 中的近邻性的度量方式；LVQ-KNN 提出了基于学习向量量化的 KNN 算法；TBNN 则结合聚集的思想提出了基于聚簇中心三角区域的近邻算法；而 KDDCUP99 Winner 则是 KDDCUP99 竞赛中的冠军算法。

可以看出从 1999 年至今，对入侵检测方法正确率、误报率和漏报率方面的改进一直在进行。而 Cluster-KNN 算法在这方面都比表 4 中其余 5 个算法更好，与 2013 年提出的 MLH-IDS 算法较为近似，因此 Cluster-KNN 算法在区分正常和异常情形方面也具有相当的优势。

表 4 Cluster-KNN 算法与其他算法比较

| 算法名称 | 正确率(%) | 误报率(%) | 漏报率(%) | 年份 |
|---|---|---|---|---|
| Cluster-KNN | 99.97+ | 0.01+ | 0.01+ | Now |
| MLH-IDS[16] | 99.97 | 0.01 | 0.02 | 2013 |
| NSM-KNN[5] | 93.21 | 0.32 | 6.47 | 2013 |
| LVQ-KNN[4] | 77.37 | 15.57 | 7.06 | 2012 |
| TBNN[21] | 93.87 | 5.65 | 0.48 | 2009 |
| KDDCUP99 Winner[22] | 93.3 | 0.11 | 6.59 | 1999 |

**结束语** 本文针对 KNN 等懒惰型学习算法在处理大样本集时出现的时间耗费大的问题，提出了融合 Kmeans 和 KNN 的 Cluster-KNN 算法，实验结果说明 Cluster-KNN 算法是一种能够适应大样本集的算法，尤其是在线分类阶段，相比 KNN 等懒惰型学习算法具有明显的时间优势。同时和其它同领域的网络入侵检测方法相比，Cluster-KNN 算法也具有很好的正确率、误报率和漏报率，在区分异常和正常情况方面具有相当的优势。然而 Cluster-KNN 算法并未对异常情况中具体攻击类型进行识别，这将是本文的下一步工作。

## 参 考 文 献


[1] Liao Yi-hua, Vemuri V R. Using K-Nearest neighbor classifier for Intrusion detection[J]. Computers and Security, 2002, 5(21):439-448

[2] Li Yang, Fang Bin-xing, et al. Supervised Intrusion Detection Based on Active Learning and TCM-KNN Algorithm[J]. Chinese Journal of Computers, 2007, 30(8):1464-1473(in Chinese)
李洋，方滨兴，等. 基于主动学习和 TCM-KNN 方法的有指导入侵检测技术[J]. 计算机学报，2007，30(8)：1464-1473

[3] Naoum R S, Al-Sultani Z N. Learning Vector Quantization (LVQ) and k-Nearest Neighbor for Intrusion Classification[J].

World of Computer Science and Information Technology Journal,2012,3(2):105-109

[4] Jamshidi Y, Nezamabadi-pour H. A Lattice based Nearest Neighbor Classifier for Anomaly Intrusion Detection[J]. Journal of Advances in Computer Research,2013,4(4):51-60

[5] Ma Z,Kaban A. K-Nearest-Neighbours with a novel similarity measure for intrusion detection[C]//2013 13th UK Workshop on Computational Intelligence (UKCI). IEEE,2013:266-271

[6] Jianliang M,Haikun S,Ling B. The Application on Intrusion Detection Based on K-means Cluster Algorithm[C]//International Forum on Information Technology and Applications,2009(IFITA '09). IEEE,2009:150-152

[7] Li Z,Li Y,Xu L. Anomaly Intrusion Detection Method Based on K-Means Clustering Algorithm with Particle Swarm Optimization[C]//2011 International Conference on Information Technology, Computer Engineering and Management Sciences (ICM). IEEE,2011:157-161

[8] Deelers S A S. Enhancing K-Means Algorithm with Initial Cluster Centers Derived from Data Partitioning along the Data Axis with the Highest Variance[C]//Proceedings of World Academy of Science,Engineering and Technology. 2007,26:323-328

[9] Gast E,Oerlemans A,Lew M S. Very large scale nearest neighbor search: ideas, strategies and challenges[J]. International Journal of Multimedia Information Retrieval,2013,2(4):229-241

[10] Muda Z, Yassin W, Sulaiman M N, et al. Intrusion detection based on K-Means clustering and Naïve Bayes classification[C]// International Conference on Information Technology in Asia. IEEE,2011:1-6

[11] Ashok R,Lakshmi A J,Rani G D V,et al. Optimized feature selection with k-means clustered triangle SVM for Intrusion Detection[C]//Third International Conference on Advanced Computing. IEEE,2011:23-27

[12] Sharma S K,Pandey P,Tiwari S K,et al. An improved network intrusion detection technique based on k-means clustering via Naïve Bayes classification[C]//2012 International Conference on Advances in Engineering,Science and Management (ICAESM). IEEE,2012:417-422

[13] Muda Z, Yassin W, Sulaiman M N, et al. Intrusion detection based on k-means clustering and OneR classification[C]//2011 7th International Conference on Information Assurance and Security (IAS). IEEE,2011:192-197

[14] Guo C, Zhou Y, Ping Y, et al. A distance sum-based hybrid method for intrusion detection[J]. Applied Intelligence,2014,40(1):178-188

[15] Kuang F,Xu W,Zhang S. A novel hybrid KPCA and SVM with GA model for intrusion detection[J]. Applied Soft Computing,2014,18(4):178-184

[16] Gogoi P, Bhattacharyya D K, et al. MLH-IDS: A Multi-Level Hybrid Intrusion Detection Method [J]. Computer Journal,2014,57(4):602-623

[17] Xiang C,Xiao Y,Qu P,et al. Network Intrusion Detection Based on PSO-SVM[J]. TELKOMNIKA:Indonesian Journal of Electrical Engineering,2013,12(2):1052-1058

[18] Wang Jie-song,Zhang Xiao-fei. The Analysis and Pre-process of KDDCup99 Benchmark Dataset of Network Intrusion Detection [J]. Science and Technology Information,2008(15):79-80(in Chinese)
王洁松,张小飞. KDDCup99 网络入侵检测数据的分析和预处理[J]. 科技信息:科学·教研,2008(15):79-80

[19] Zhang Xin-you, Zeng Hua-shen, Jia Lei. Research of Intrusion Detection system Dataset-KDD CUP99[J]. Computer Engineering and Design. 2010,31(22):4809-4814(in Chinese)
张新有,曾华燊,贾磊. 入侵检测数据集 KDD CUP99 研究[J]. 计算机工程与设计,2010,31(22):4809-4814

[20] Wang Zhi-gang, Hu Chang-zhen, et al. Cyber Security Datasets Research advanced materials research[J]. Advanced Materials and Computer Science II,2013,4(4):191-195

[21] Tsai C,Lin C. A triangle area based nearest neighbors approach to intrusion detection[J]. Pattern Recognition,2010,43:222-229

[22] Elkan C. Results of the KDD'99 classifier learning contest [OL]. http://cseweb.ucsd.edu/users/elkan/clresults.html

(上接第 136 页)

[16] Kheir N,Wolley C. BotSuer:Suing stealthy P2P bots in network traffic through netflow analysis[M]//Cryptology and Network Security. Springer International Publishing,2013:162-178

[17] Fan Y, Xu N. A P2P Botnet Detection Method Used On-line Monitoring and Off-line Detection[J]. International Journal of Security & Its Applications,2014,8(3):87-96

[18] Amini P, Azmi R, Araghizadeh M A. Botnet Detection using NetFlow and Clustering[J]. Advances in Computer Science: an International Journal,2014,3(2):139-149

[19] Garg S,Sarje A K,Peddoju S K. Improved Detection of P2P Botnets through Network Behavior Analysis[M]//Recent Trends in Computer Networks and Distributed Systems Security. Springer Berlin Heidelberg,2014:334-345

[20] Vania J,Meniya A,Jethva H B. A Review on Botnet and Detection Technique[J]. International Journal of Computer Trends and Technology,2013,4(1):23-29

[21] Zhao Y,Xie Y,Yu F,et al. BotGraph:Large Scale Spamming Botnet Detection[C]//NSDI. 2009,9:321-334

[22] Jiang Hong-ling,Shao Xiu-li,Li Yao-fang. Online Botnet Detection Algorithm Using MapReduce[J]. Journal of Electronics & Information Technology,2013,35(7):1732-1738(in Chinese)
蒋鸿玲,邵秀丽,李耀芳. 基于 MapReduce 的僵尸网络在线检测算法[J]. 电子与信息学报,2013,35(7):1732-1738

[23] Batcher K E. Design of a massively parallel processor[J]. IEEE Transactions on Computers,1980,100(9):836-840

[24] Gropp W,Lusk E,Doss N,et al. A high-performance,portable implementation of the MPI message passing interface standard [J]. Parallel Computing,1996,22(6):789-828

[25] Geist A,Beguelin A,Dongarra J,et al. PVM:Parallel virtual machine—a users' guide and tutorial for networked parallel computing[M]. MIT press,1994

[26] Zaharia M, Chowdhury M, Franklin M J, et al. Spark: cluster computing with working sets[C]//Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing. 2010:10

[27] Dean J,Ghemawat S. MapReduce:simplified data processing on large clusters[J]. Communications of the ACM,2008,51(1):107-113

[28] Bahmani B,Moseley B,Vattani A,et al. Scalable k-means++ [J]. Proceedings of the VLDB Endowment,2012,5(7):622-633