(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 109670310 B
(45)授权公告日 2023.04.18

(21)申请号 201910081131.X
(22)申请日 2019.01.28
(65)同一申请的已公布的文献号
申请公布号 CN 109670310 A
(43)申请公布日 2019.04.23
(73)专利权人 杭州师范大学
地址 310015 浙江省杭州市余杭区余杭塘路2318号
(72)发明人 刘雪娇 罗娟 胡芷琦
(74)专利代理机构 杭州杭诚专利事务所有限公司 33109
专利代理师 尉伟敏
(51)Int.Cl.
G06F 21/56(2013.01)
G06F 18/23213(2023.01)

审查员 吴佳兴

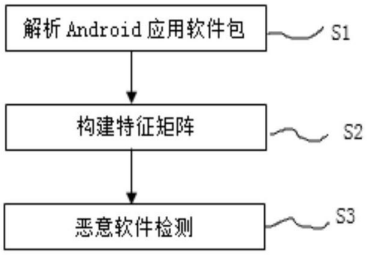
权利要求书2页 说明书6页 附图1页

(54)发明名称

一种基于半监督K-Means聚类算法的Android恶意软件检测方法

(57)摘要

本发明公开了一种基于半监督K-Means聚类算法的Android恶意软件检测方法，所述方法包括，步骤S1：解析Android应用软件包，选择适量有标签的样本和两倍数量的无标签的样本，使用解压缩工具打开这些样本的Android应用软件包，得到权限集合P1、P2、P3；步骤S2：构建特征矩阵，通过信息增益算法统计每个样本的权限集合P1、P2、P3的评分结果为s1、s2、s3，选择适量有标签的样本构建成特征矩阵FN和FM；步骤S3：恶意软件检测：运用半监督k-means算法对特征矩阵FN和特征矩阵FM中的样本进行恶意软件的检测，本技术方案使用大量的无标签样本，同时使用尽量少的有标签样本，来进行分类，对恶意软件检测具有较高的准确性。



CN 109670310 B

1.一种基于半监督K-Means聚类算法的Android恶意软件检测方法，其特征在于，所述方法包括：

步骤S1、解析Android应用软件包：选择适量有标签的样本和两倍数量的无标签的样本，使用解压缩工具打开这些样本的Android应用软件包，得到classes.dex文件和AndroidManifest.xml文件，解析AndroidManifest.xml文件提取每个样本的权限的特征集合P1，反编译classes.dex文件提取调用的API，构建每个样本API对应的权限的特征集合P2，通过每个样本的权限集合P1和P2得到该样本的非过度申请的权限集合P3；

步骤S2、构建特征矩阵：通过信息增益算法得到不同权限的属性评分结果，统计每个样本的权限集合P1、P2、P3的评分结果为s1、s2、s3，选择适量有标签的样本构建成特征矩阵FN，

选择两倍数量的无标签的样本构建成特征矩阵FM；

步骤S3、恶意软件检测：运用半监督k-means算法对特征矩阵FN和特征矩阵FM中的样本进行恶意软件的检测。

2.根据权利要求1所述的一种基于半监督K-Means聚类算法的Android恶意软件检测方法，其特征是所述步骤S1中获取每个样本的权限特征集合的过程包括：

步骤S11、反编译classes.dex文件获取源代码，遍历所有源代码得到API集合，API集合对照公开的权限-函数映射关系表获得其对应的权限集合P2；

步骤S12、根据AndroidMainfest.xml文件解析得到的权限集合P1和权限集合P2得到非过度申请的权限集合P3，其中P3＝P1∩P2。

3.根据权利要求1所述的一种基于半监督K-Means聚类算法的Android恶意软件检测方法，其特征是所述步骤S2中构建特征矩阵的过程包括：

步骤S21、通过信息增益算法处理多个非恶意软件和恶意软件得到不同权限的属性评分结果；定义权限变量Y为恶意软件中的权限，一共有n种权限，在恶意软件中每一种权限存在的概率为$p(y_i)$，在非恶意软件中每一种权限存在的概率为$q(y_i)$，计算Y的信息量为：$H(Y)=-\sum_{i=1}^{n}p(y_i)\ln\frac{p(y_i)}{q(y_i)}$，计算权限y的信息增益为IG(y)＝H(Y)-H(Y|y)，IG(y)越大代表该权限对恶意软件和非恶意软件的区分度越大，将其作为权限y的属性评分结果；

步骤S22、根据步骤S21中得到的不同权限的属性评分结果，运用公式S＝∑IG(y)(y∈P)，计算每个样本的权限集合P1、P2以及P3的属性评分结果分别为s1、s2、s3；

步骤S23、选取m个有标签样本，构建有标签样本的特征向量V1，V1＝(s1,s2,s3,type)，type＝0代表该样本为恶意软件，type＝1代表该样本为非恶意软件，将有标签的样本集合集制成一个m×4特征矩阵FN；所述FN的表达式如下：

$$FN=\begin{bmatrix}a_{11} & a_{12} & a_{13} & a_{14}\\ \vdots & \vdots & \vdots & \vdots\\ a_{m1} & a_{m2} & a_{m3} & a_{m4}\end{bmatrix}$$

步骤S24、选n个无标签样本，构建无标签样本的特征向量V2，V2＝(s1,s2,s3)，将无签的样本集合集制成一个n×3特征矩阵FM；所述FM表达式如下：

$$FM=\begin{bmatrix}a_{11} & a_{12} & a_{13}\\ \vdots & \vdots & \vdots\\ a_{n1} & a_{n2} & a_{n3}\end{bmatrix}。$$

4.根据权利要求1所述的一种基于半监督K-Means聚类算法的Android恶意软件检测方法，其特征是所述步骤S3中运用半监督k-means算法进行非恶意软件和恶意软件的检测过程包括：

步骤S31、根据特征矩阵FN将m个有标签样本表示成三维空间中的点$\{x^{(1)},...,x^{(m)}\}$，其中$x^{(i)}=(a_{i1},a_{i2},a_{i3})$，样本$x^{(i)}$的类别表示成$c^{(i)}=a_{i4}$，所有恶意样本构成恶意样本簇，所有非恶意样本构成非恶意样本簇，通过计算质心的方式分别得出恶意样本簇和非恶意样本簇的一个估计值，并将其作为初始聚类中心；

其中计算类别j初始聚类中心μ_j的公式是：

$$\mu_j=\frac{\sum_{i=1}^{m}1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m}1\{c^{(i)}=j\}}$$

步骤S32、根据特征矩阵FM将n个无标签样本表示成三维空间中的点$\{x^{(m+1)},...,x^{(m+n)}\}$，其中$x^{(i)}=(a_{i1},a_{i2},a_{i3})$，样本$x^{(i)}$的类别表示成$c^{(i)}=-1$，定义索引$Index^{(i)}=-1$，定义标志位flag＝true；

步骤S33、聚类过程：初始flag＝false，对于每一个样本$x^{(i)}$计算其索引$Index^{(i)}=argmn_j||x^{(i)}-\mu_j||^2$，如果$Index^{(i)}=c^{(i)}$，改变标志位flag＝true，重置样本$x^{(i)}$的类别$c^{(i)}=Index^{(i)}$；

步骤S34、如果一次聚类结束后，标志位flag＝true，对于恶意样本簇和非恶意样本簇，重新计算它们的聚类中心，重复聚类过程；

其中计算类别j聚类中心μ_j的公式是：

$$\mu_j=\frac{\sum_{i=1}^{m+n}1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m+n}1\{c^{(i)}=j\}}$$

步骤S35、如果一次聚类结束后，flag＝false，代表聚类结果不再发生变化，$\{c^{(1)},...,c^{(m)}\}$为特征矩阵FN的样本的最终的聚类结果，可用于计算该算法检测的准确率，$\{c^{(m+1)},...,c^{(m+n)}\}$为特征矩阵FM的样本的最终的聚类结果，$c^{(i)}=0$代表待检测的无标签样本$x^{(i)}$所代表的Android应用软件包的检测结果为恶意软件；$c^{(i)}=1$代表待检测的无标签样本$x^{(i)}$所代表的Android应用软件包的检测结果为非恶意软件。

一种基于半监督K-Means聚类算法的Android恶意软件检测方法

技术领域

[0001] 本发明涉及恶意软件检测及网络安全技术领域，尤其涉及一种基于半监督K-Means聚类算法的Android恶意软件检测方法。

背景技术

[0002] 由于系统开源性、免费性的特点，Android系统作为主流的移动操作系统成为了恶意软件攻击的主要目标。从2017年新增恶意软件分布来看资费消耗类型的恶意样本占比已达到 3/4，说明恶意软件依然是以推销广告、消耗流量、增加手机用户的流量资费等来谋取经济利益，用户的安全得不到保障，所以研究Android恶意软件检测方法仍然是很有必要的。

[0003] Android恶意代码分析研究目前主要有两种方法，即静态分析和动态检测。静态分析是指通过分析程序代码来判断程序行为；动态分析是指在严格控制的环境下执行应用程序，尽可能的触发软件的全部行为并记录，以检测应用程序是否包含恶意行为。目前，关于Android恶意应用程序的检测方法较多，但总体而言，其大致是围绕权限机制和API的调用这两大方面来进行展开的；权限控制着应用软件对资源的访问，但程序员在APP的开发过程中存在过度申请权限的行为，这加大了恶意软件检测的难度。应用程序可以通过对API的调用在一定程度上反映出一款应用程序真实的行为特点，因此可以运用于Android恶意软件检测。

[0004] 在静态分析中主要运用机器学习的方法来进行数据挖掘，主要分为两类，有监督学习和无监督学习，有监督学习主要分为分类模型和预测模型，无监督学习主要分为关联分析和聚类分析。分类算法主要有k近邻算法，朴素贝叶斯、决策树归纳、随机森林、支持向量机SVM、遗传算法等，这些分类算法在Android恶意软件检测中频繁运用。现在研究中的Android恶意软件检测方法大多是有监督学习的算法，但是有监督学习的算法需要大量的恶意Android软件作为训练样本，由于恶意软件更新换代很快，收集足够多的最新型恶意软件的代价是昂贵的。无监督学习的算法因为具有太多不确定性，所有很少用于检测Android恶意软件。用K-Means聚类算法检测Android恶意软件易受初始聚类中心影响，当初始聚类中心选取合适时，准确率高，初始聚类中心选取不合适时，准确率低。2018年《Android恶意软件检测方法研究综述》表明，Android恶意软件检测经过多年努力，已经取得了一定的成果，研究结果都表示机器学习算法在Android恶意软件检测的研究中起着积极的作用，但随着新型恶意软件的增多，恶意软件样本库需要不断更新，能够检测出现有恶意软件的同时还能够应对不断出现的新型恶意软件的成果还未曾出现。

发明内容

[0005] 本发明的目的在于提供一种基于半监督K-Means聚类算法的Android恶意软件检测方法，以解决K-Means聚类算法检测Android恶意软件易受初始聚类中心影响的问题。运

用半监督 K-Means聚类算法进行Android恶意软件检测，首先通过约束种子即有标签的样本集合确定初始聚类中心，然后运用kmeans聚类算法根据欧式距离对无标签的样本集合进行聚类，直到每个类别中的样本不再发生变化，使用大量的无标签样本，同时使用尽量少的有标签样本，来进行分类，对恶意软件检测具有较高的准确性。

[0006] 为实现上述技术目的，本发明提供的一种技术方案是，一种基于半监督K-Means聚类算法的Android恶意软件检测方法，所述方法包括：

[0007] 步骤S1、解析Android应用软件包：选择适量有标签的样本和两倍数量的无标签的样本，使用解压缩工具打开这些样本的Android应用软件包，得到classes.dex文件和AndroidManifest.xml文件，解析AndroidManifest.xml文件提取每个样本的权限的特征集合 P1，反编译classes.dex文件提取调用的API，构建每个样本API对应的权限的特征集合P2，

[0008] 通过每个样本的权限集合P1和P2得到该样本的非过度申请的权限集合P3；

[0009] 步骤S2、构建特征矩阵：通过信息增益算法得到不同权限的属性评分结果，统计每个样本的权限集合P1、P2、P3的评分结果为s1、s2、s3，选择适量有标签的样本构建成特征矩阵FN，

[0010] 选择两倍数量的无标签的样本构建成特征矩阵FM；

[0011] 步骤S3、恶意软件检测：运用半监督k-means算法对特征矩阵FN和特征矩阵FM中的样本进行恶意软件的检测。

[0012] 所述步骤S1中获取每个样本的权限特征集合的过程包括：

[0013] 步骤S11、反编译classes.dex文件获取源代码，遍历所有源代码得到API集合，API集合对照公开的权限-函数映射关系表获得其对应的权限集合P2；

[0014] 步骤S12、根据AndroidMainfest.xml文件解析得到的权限集合P1和权限集合P2得到非过度申请的权限集合P3，其中P3＝P1∩P2；

[0015] 所述步骤S2中构建特征矩阵的过程包括：

[0016] 步骤S21、通过信息增益算法处理多个非恶意软件和恶意软件得到不同权限的属性评分结果；定义权限变量Y为恶意软件中的权限，一共有n种权限，在恶意软件中每一种权限存在的概率为$P(y_i)$，在非恶意软件中每一种权限存在的概率为$q(y_i)$，计算Y的信息量为：$H(Y)=-\sum_{i=1}^{n}p(y_i)\ln\frac{p(y_i)}{q(y_i)}$，计算权限y的信息增益为IG(y)＝H(Y)-H(Y|y)，IG(y)越大代表该权限对恶意软件和非恶意软件的区分度越大，将其作为权限y的属性评分结果；

[0017] 步骤S22、根据步骤S21中得到的不同权限的属性评分结果，运用公式S＝∑IG(y)(y∈P)，计算每个样本的权限集合P1、P2以及P3的属性评分结果分别为s1、s2、s3；

[0018] 步骤S23、选取m个有标签样本，构建有标签样本的特征向量V1，V1＝(s1，s2，s3，type)， type＝0代表该样本为恶意软件，type＝1代表该样本为非恶意软件，将有标签的样本集合集制成一个m×4特征矩阵FN：所述FN的表达式如下：

[0019] $$FN=\begin{bmatrix}a_{11} & a_{12} & a_{13} & a_{14}\\ \vdots & \vdots & \vdots & \vdots\\ a_{m1} & a_{m2} & a_{m3} & a_{m4}\end{bmatrix}$$

[0020] 步骤S24、选n个无标签样本，构建无标签样本的特征向量V2，V2＝(s1，s2，s3)，将无签的样本集合集制成一个n×3特征矩阵FM；所述FM表达式如下：

[0021] $$FM=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & a_{n3} \end{bmatrix}$$。

[0022] 所述步骤S3中运用半监督k-means算法进行非恶意软件和恶意软件的检测过程包括:步骤S31、根据特征矩阵FN将m个有标签样本表示成三维空间中的点$\{x^{(1)},...,x^{(m)}\}$,其中 $x^{(i)}=(a_{i1},a_{i2},a_{i3})$,样本$x^{(i)}$的类别表示成$c^{(i)}=a_{i4}$,所有恶意样本构成恶意样本簇,所有非恶意样本构成非恶意样本簇,通过计算质心的方式分别得出恶意样本簇和非恶意样本簇的一个估计值,并将其作为初始聚类中心;

[0023] 其中计算类别j初始聚类中心μ_j的公式是:

[0024] $$\mu_j=\frac{\sum_{i=1}^{m}1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m}1\{c^{(i)}=j\}}$$

[0025] 步骤S32、根据特征矩阵FM将n个无标签样本表示成三维空间中的点$\{x^{(m+1)},...,x^{(m+n)}\}$,其中$x^{(i)}=(a_{i1},a_{i2},a_{i3})$,样本$x^{(i)}$的类别表示成$c^{(i)}=-1$,定义索引$Index^{(i)}=-1$,定义标志位flag=true;

[0026] 步骤S33、聚类过程:初始flag=false,对于每一个样本$x^{(i)}$计算其索引 $Index^{(i)}=argmin_j||x^{(i)}-\mu_j||^2$,如果$Index^{(i)}=c^{(i)}$,改变标志位flag=true,重置样本$x^{(i)}$的类别$c^{(i)}=Index^{(i)}$;

[0027] 步骤S34、如果一次聚类结束后,标志位flag=true,对于恶意样本簇和非恶意样本簇,重新计算它们的聚类中心,重复聚类过程;

[0028] 其中计算类别j聚类中心μ_j的公式是:

[0029] $$\mu_j=\frac{\sum_{i=1}^{m+n}1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m+n}1\{c^{(i)}=j\}}$$

[0030] 步骤S35、如果一次聚类结束后,flag=false,代表聚类结果不再发生变化,$\{c^{(1)},....,c^{(m)}\}$为特征矩阵FN的样本的最终的聚类结果,可用于计算该算法检测的准确率,$\{c^{(m+1)},....,c^{(m+n)}\}$为特征矩阵FM的样本的最终的聚类结果,$c^{(i)}=0$代表待检测的无标签样本$x^{(i)}$所代表的 Android应用软件包的检测结果为恶意软件;$c^{(i)}=1$代表待检测的无标签样本$x^{(i)}$所代表的 Android应用软件包的检测结果为非恶意软件。

[0031] 本发明的有益效果:本发明技术方案运用半监督K-Means聚类算法进行Android恶意软件检测,首先通过约束种子即有标签的样本集合确定初始聚类中心,然后运用kmeans聚类算法根据欧式距离对无标签的样本集合进行聚类,直到每个类别中的样本不再发生变化,使用大量的无标签样本,同时使用尽量少的有标签样本,来进行分类,对恶意软件检测具有较高的准确性。

附图说明

[0032] 图1为本发明的一种基于半监督K-Means聚类算法的Android恶意软件检测方法的方法流程图。

[0033] 图2本发明的一种基于半监督K-Means聚类算法的Android恶意软件检测方法的聚类算法流程图。

具体实施方式

[0034] 为使本发明的目的、技术方案以及优点更加清楚明白，下面结合附图和实施例对本发明作进一步详细说明，应当理解的是，此处所描述的具体实施方式仅是本发明的一种最佳实施例，仅用以解释本发明，并不限定本发明的保护范围，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0035] 实施例：如图1所示，一种基于半监督K-Means聚类算法的Android恶意软件检测方法，所述方法包括：

[0036] 步骤S1、解析Android应用软件包：选择适量有标签的样本和两倍数量的无标签的样本，使用解压缩工具打开这些样本的Android应用软件包，得到classes.dex文件和AndroidManifest.xml文件，解析AndroidManifest.xml文件提取每个样本的权限的特征集合P1，反编译classes.dex

[0037] 文件提取调用的API，构建每个样本API对应的权限的特征集合P2，通过每个样本的权限集合P1和P2得到该样本的非过度申请的权限集合P3。

[0038] 步骤S2、构建特征矩阵：通过信息增益算法得到不同权限的属性评分结果，统计每个样本的权限集合P1、P2、P3的评分结果为s1、s2、s3，选择适量有标签的样本构建成特征矩阵 FN，选择两倍数量的无标签的样本构建成特征矩阵FM；

[0039] 步骤S3、恶意软件检测：运用半监督k-means算法对特征矩阵FN和特征矩阵FM中的样本进行恶意软件的检测。

[0040] 所述步骤S1中获取每个样本的权限特征集合的过程包括：

[0041] 步骤S11、反编译classes.dex文件获取源代码，遍历所有源代码得到API集合，API集合对照公开的权限-函数映射关系表获得其对应的权限集合P2；

[0042] 步骤S12、根据AndroidMainfest.xml文件解析得到的权限集合P1和权限集合P2得到非过度申请的权限集合P3，其中P3＝P1∩P2；

[0043] 本实施例中，设有标签样本1如下：

[0044] P1＝{ACCESS_NETWORK_STATE,ADD_SYSTEM_SERVICE,READ_CONTACTS,READ_SM S,SEND_SMS,WRITE_SMS,RECEIVE_SMS,INTERNET,WRITE_APN_SETTINGS,ADD_SYS TEM_SERVICE,CHANGE_NETWORK_STATE,CALL_PHONE,MODIFY_PHONE_STATE,RE AD_PHONE_STATE,com.android.launcher.permission.INSTALL_SHORTCUT,com.android.launcher.permission.UNINSTALL_SHORTCUT,MODIFY_AUDIO_SETTINGS,PROCESS_OUTGOIN G_CALLS,CHANGE_WIFI_STATE,ACCESS_WIFI_STATE,GET_TASKS,RECEIVE_BOOT_C OMPLETED,WAKE_LOCK,com.android.browser.permission.READ_HISTORY_BOOKMARKS, com.android.browser.permission.WRITE_HISTORY_BOOKMARKS}；

[0045] 反编译classes.dex文件获取源代码，遍历所有源代码得到API集合，API集合对照公开的权限-函数映射关系表获得其对应的权限集合P2；

[0046] P2＝{CHANGE_NETWORK_STATE,CONNECTIVITY_INTERNAL,INTERACT_ACROSS_USERS,BACKUP,INTERACT_ACROSS_USERS_FULL,DUMP,INTERNET,BLUETOOTH_ADMI N,WRITE_SMS}；

[0047] 根据AndroidMainfest.xml文件解析得到的权限集合P1和权限集合P2得到非过度申请的权限集合P3，其中P3＝P1∩P2；

[0048] 则P3＝{CHANGE_NETWORK_STATE,INTERNET,WRITE_SMS}。

[0049] 如图2所示,所述步骤S2中构建特征矩阵的过程包括:

[0050] 步骤S21、通过信息增益算法处理多个非恶意软件和恶意软件得到不同权限的属性评分结果;定义权限变量Y为恶意软件中的权限,一共有n种权限,在恶意软件中每一种权限存在的概率为$p(y_i)$,在非恶意软件中每一种权限存在的概率为$q(y_i)$,计算Y的信息量为:

[0051] $H(Y)=-\sum_{i=1}^{n}p(y_i)\ln\frac{p(y_i)}{q(y_i)}$,计算权限y的信息增益为IG(y)＝H(Y)-H(Y|y),IG(y)越大代表该权限对恶意软件和非恶意软件的区分度越大,将其作为权限y的属性评分结果;

[0052] 步骤S22、根据步骤S21中得到的不同权限的属性评分结果,运用公式S＝∑IG(y)(y∈P),计算每个样本的权限集合P1、P2以及P3的属性评分结果分别为s1、s2、s3;

[0053] 步骤S23、选取m个有标签样本,构建有标签样本的特征向量V1,V1＝(s1,s2,s3,type), type＝0代表该样本为恶意软件,type＝1代表该样本为非恶意软件,将有标签的样本集合集制成一个m×4特征矩阵FN;所述FN的表达式如下:

[0054] $$FN=\begin{bmatrix}a_{11} & a_{12} & a_{13} & a_{14}\\ \vdots & \vdots & \vdots & \vdots\\ a_{m1} & a_{m2} & a_{m3} & a_{m4}\end{bmatrix}$$

[0055] 步骤S24、选n个无标签样本,构建无标签样本的特征向量V2,V2＝(s1,s2,s3),将无签的样本集合集制成一个n×3特征矩阵FM;所述FM表达式如下:

[0056] $$FM=\begin{bmatrix}a_{11} & a_{12} & a_{13}\\ \vdots & \vdots & \vdots\\ a_{n1} & a_{n2} & a_{n3}\end{bmatrix}$$。

[0057] 本实施例中,标签样本1的P1属性评分结果s1为1.19279,P2属性评分结果s2为0.179746,P3属性评分结果s3为0.179746;选择4个有标签样本构成特征矩阵FN;

[0058] $$FN=\begin{bmatrix}1.19279 & 0.179746 & 0.179746 & 0\\ 0.73476 & 0.011548 & 0.006863 & 0\\ 0.740395 & 0.20717 & 0.034292 & 1\\ 1.122182 & 0.179746 & 0.011548 & 1\end{bmatrix}$$;

[0059] 选择8个无标签样本构成特征矩阵FN;

[0060] $$FM=\begin{bmatrix}0.505572 & 0.006863 & 0.006863\\ 0.542327 & 0.011548 & 0.006863\\ 1.03597 & 0.011548 & 0.006863\\ 0.717487 & 0.006863 & 0.006863\\ 0.483099 & 0.004685 & 0\\ 0.218191 & 0 & 0\\ 1.133019 & 0.20717 & 0.20249\\ 0.30244 & 0.016228 & 0.006863\end{bmatrix}$$

[0061] 所述步骤S3中运用半监督k-means算法进行非恶意软件和恶意软件的检测过程包括:

[0062] 步骤S31、根据特征矩阵FN将m个有标签样本表示成三维空间中的点$\{x^{(1)},\ldots,x^{(m)}\}$,其中$x^{(i)}=(a_{i1},a_{i2},a_{i3})$,样本$x^{(i)}$的类别表示成$c^{(i)}=a_{i4}$,所有恶意样本构成恶意样本簇,所有非恶意样本构成非恶意样本簇,通过计算质心的方式分别得出恶意样本簇和非恶意样本簇的一个估计值,并将其作为初始聚类中心;

[0063] 其中计算类别j初始聚类中心μ_j的公式是：

[0064] $$\mu_j = \frac{\sum_{i=1}^{m} 1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m} 1\{c^{(i)}=j\}};$$

[0065] 步骤S32、根据特征矩阵FM将n个无标签样本表示成三维空间中的点$\{x^{(m+1)},\ldots,x^{(m+n)}\}$，其中$x^{(i)}=(a_{i1},a_{i2},a_{i3})$，样本$x^{(i)}$的类别表示成$c^{(i)}=-1$，定义索引$Index^{(i)}=-1$，定义标志位flag＝true；

[0066] 步骤S33、聚类过程：初始flag＝false，对于每一个样本$x^{(i)}$计算其索引 $Index^{(i)}=argmin_j||x^{(i)}-\mu_j||^2$，如果$Index^{(i)}=c^{(i)}$，改变标志位flag＝true，重置样本$x^{(i)}$的类别$c^{(i)}=Index^{(i)}$；

[0067] 步骤S34、如果一次聚类结束后，标志位flag＝true，对于恶意样本簇和非恶意样本簇，重新计算它们的聚类中心，重复聚类过程；

[0068] 其中计算类别j聚类中心μ_j的公式是：

[0069] $$\mu_j = \frac{\sum_{i=1}^{m+n} 1\{c^{(i)}=j\}x^{(i)}}{\sum_{i=1}^{m+n} 1\{c^{(i)}=j\}};$$

[0070] 步骤S35、如果一次聚类结束后，flag＝false，代表聚类结果不再发生变化，$\{c^{(1)},\ldots,c^{(m)}\}$为特征矩阵FN的样本的最终的聚类结果，可用于计算该算法检测的准确率，$\{c^{(m+1)},\ldots,c^{(m+n)}\}$为特征矩阵FM的样本的最终的聚类结果，$c^{(i)}=0$代表待检测的无标签样本$x^{(i)}$所代表的 Android应用软件包的检测结果为恶意软件；$c^{(i)}=1$代表待检测的无标签样本$x^{(i)}$所代表的Android应用软件包的检测结果为非恶意软件。

[0071] 本实施例中，

[0072] $\mu_0=((0.963775,0.095647,0.0933045)$；

[0073] $\mu_1=(0.9312885,0.193458,0.02292)$；

[0074] $\{c^{(1)},\ldots,c^{12}\}=\{0,0,1,1,1,1,0,0,1,1,0,1\}$；

[0075] $\{c^{(1)},\ldots,c^{(4)}\}=\{0,0,1,1\}$；

[0076] $\{c^{(5)},\ldots,c^{(12)}\}=\{1,1,0,0,1,1,0,1\}$；

[0077] 其中通过特征矩阵FN的样本初始类别和最终的聚类结果计算该算法检测的准确率的公式是：ACC＝类别不变的样本个数/样本总个数；

[0078] 即ACC＝4/4＝100％。

[0079] 以上所述之具体实施方式为本发明一种基于半监督K-Means聚类算法的Android恶意软件检测方法的较佳实施方式，并非以此限定本发明的具体实施范围，本发明的范围包括并不限于本具体实施方式，凡依照本发明之形状、结构所作的等效变化均在本发明的保护范围内。

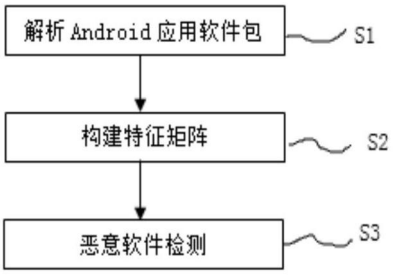

解析Android应用软件包
S1
构建特征矩阵
S2
恶意软件检测
S3

图1

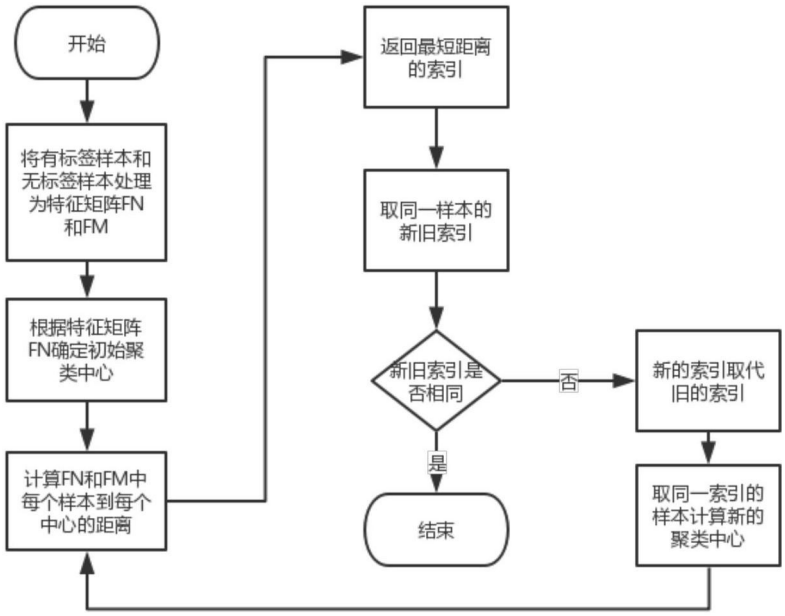

开始
将有标签样本和无标签样本处理为特征矩阵FN和FM
根据特征矩阵FN确定初始聚类中心
计算FN和FM中每个样本到每个中心的距离
返回最短距离的索引
取同一样本的新旧索引
新旧索引是否相同
否
是
新的索引取代旧的索引
取同一索引的样本计算新的聚类中心
结束

图2