第 32 卷　第 10 期　　计　算　机　工　程　　2006 年 5 月
Vol.32　№ 10　　Computer Engineering　　May 2006
·博士论文·　　文章编号：1000—3428(2006)10—0018—04　　文献标识码：A　　中图分类号：TP309.5

基于多重朴素贝叶斯算法的未知病毒检测

张波云[1,2]，殷建平[1]，蒿敬波[1]，张鼎兴[1]
(1. 国防科技大学计算机学院，长沙 410073；2. 湖南公安高等专科学校计算机系，长沙 410138)

摘　要：提出了一种基于多重朴素贝叶斯分类算法的检测方法来实现对计算机病毒的近似判别。该法可以克服病毒特征代码扫描法不能识别未知病毒的缺点。在该检测方法的基础上，设计了一个病毒检测网络模型，该模型既可以实现对已知病毒的查杀，又可以对可疑程序行为进行分析评判，最终实现对未知病毒的识别。

关键词：计算机病毒；多重朴素贝叶斯算法；信息熵；病毒检测

Unknown Computer Virus Detection Based on Multi-naive Bayes Algorithm

ZHANG Boyun[1,2], YIN Jianping[1], HAO Jingbo[1], ZHANG Dingxing[1]
(1. School of Computer Science, National University of Defense Technology, Changsha 410073;
2. Dept. of Computer Science, Hunan Public Security College, Changsha 410138)

【Abstract】 A multi-naive Bayes algorithm to detect computer virus approximately is presented in this paper. It can overcome the shortage of normal virus scanner, which could not detect unknown virus. Based on this method, a virus detect network model is also designed. This model is fit for detecting virus in the on-line system; it could detect known and unknown computer virus by analyzing the program's behavior.

【Key words】 Computer virus; Multi-naive Bayes algorithm; Information entropy; Virus detection

当前的计算机病毒检测技术主要基于特征检测法，其基本方法是提取已知病毒样本的特征，并将此特征数据添加到病毒特征库中，在病毒检测时通过搜索病毒特征库查找是否存在相匹配的病毒特征来发现病毒。这种检测办法只能用于检测已知的病毒，对于新出现的病毒的检测无能为力。为了解决这一问题，本文提出一种基于多重朴素贝叶斯分类算法的检测方法来实现对计算机病毒的近似判别。采用多重朴素贝叶斯分类算法对提取的可疑文件行为特征进行分析，并利用病毒程序与正常程序的行为特征的差异性进行分类，从而达到检测未知病毒的目的。

R.Lo 等[5]曾提出用朴素贝叶斯算法检测未知恶意代码，其算法建立在对程序的二进制代码直接分析的基础上，显然，该法对目前日渐盛行的变形病毒的检测无能为力，因为变形病毒的机器代码变化十分迅速[8]。本文的分类算法主要针对程序的执行过程中使用的系统 API 函数的调用，对各种多态与变形病毒的检测准确率比较高。

在该检测方法基础上，本文设计了一个恶意代码检测网络模型，从模型的设计和技术实现的角度进行分析，介绍了它的工作原理。该模型既可实现对已知病毒的查杀，又可对可疑文件进行分析评判，最终实现对未知病毒的识别。最后对模型的功能进行了实验测试，达到了很好的实际应用效果。

1 模型结构分析

系统结构框图如图 1 所示。系统由病毒防火墙、服务器和病毒检测服务器组成，进入系统的可疑文件首先经过系统的第一道防线病毒防火墙的检测，若没有检测出病毒，则将其复制两份：一份直接存放于 Server 中；另一份则进入基于 M-NB 的 Detect Server 中，对其行为特征进行检测，如果发现其可疑行为，判断为感染了未知病毒，Detect Server 则将信息反馈给 Server，在 Server 中对该文件进行隔离、监管或删除。由于基于程序行为的杀毒系统可能引发对系统不可预料的破坏，因此在该系统中配置了专门进行未知病毒检测的 Detect Server，它对正常服务器的性能影响极微，适用于实时在线系统中的未知病毒检测。而对于 Detect Server 自身的安全防护，可采用系统还原保护法等。系统的逻辑结构如图 2。

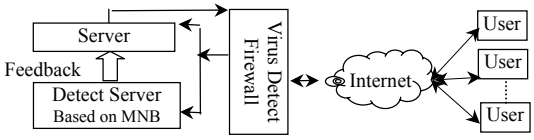

图 1 系统结构

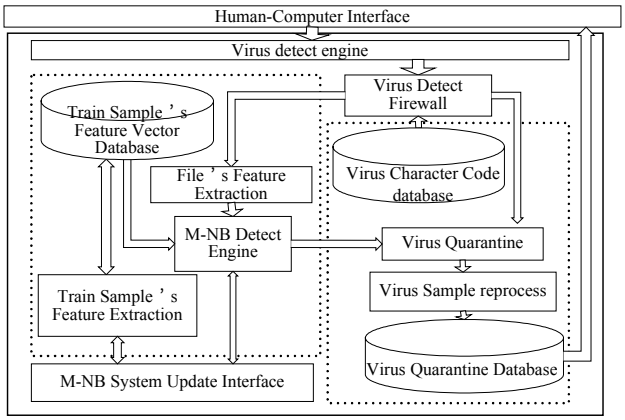

图 2 基于多重贝叶斯分类算法的病毒检测系统逻辑流程

基金项目：国家自然科学基金资助项目(60373023)；湖南省自然科学基金资助项目(04JJ6032)；湖南省教育厅优秀青年基金资助项目
作者简介：张波云（1972－），男，博士生，主研方向：网络与信息安全；殷建平，教授、博导；蒿敬波、张鼎兴，博士生
收稿日期：2005-07-03　**E-mail：**hnjxzby@yahoo.com.cn

Detect Server 中的病毒检测引擎采用 M-NB 算法实现。

(1) 系统首先进行自我训练，采用特征提取工具对样本空间 U 中的训练样本程序进行特征提取，并保存于训练样本特征库中以备将来学习和检测病毒之用。

(2) 当外来文件进入系统时，先经过病毒防火墙的第一次测试，若发现其已染有病毒，则在 Server 中对其隔离，保存入病毒隔离库中；若没发现病毒，则在 Detect Server 中继续调用 M-NB 检测引擎测试。将可疑程序特征提取出来以获取被检测文件的特征矢量，然后用 M-NB 算法对可疑程序进行病毒判定。若判断为病毒，则将信息反馈给 Server，在 Server 中对病毒文件进行隔离、监管或删除，也可以将其发送给相关专家进行进一步精确分析。与此同时把新病毒样本添加到训练样本空间，重复上述过程进行新一轮的学习。

2 算法数学模型

贝叶斯推理提供推理的一种概率手段，显然这对于病毒的近似判定十分合适。贝叶斯推理的基石是贝叶斯公式：

$$P(h \mid D) = \frac{p(D \mid h)P(h)}{P(D)} \tag{1}$$

贝叶斯分类器应用的学习任务中，每个实例 x 可由属性值的合取描述，而目标函数 $f(x)$从某有限集合 V 中取值。学习器被提供一系列关于目标函数的训练样本以及新实例(描述为特征向量$<a_1,a_2,...,a_n>$)，然后要求预测新实例的目标值（即分类）。分类器在给定描述实例的属性值下计算最可能的目标值V_{map}：

$$V_{map} = \arg\max_{v_j \in V} P(v_j \mid a_1, a_2, ..., a_n) \tag{2}$$

用贝叶斯公式重写，得

$$V_{map} = \arg\max_{v_j \in V} P(a_1, a_2, ..., a_n \mid v_j)P(v_j) \tag{3}$$

朴素贝叶斯分类器基于一个简单的假定：在给定目标值时属性值之间相互条件独立，故有：

$$P(a_1, a_2, ..., a_n \mid v_j) = \prod_i P(a_i \mid v_j) \tag{4}$$

代入式(3)得

$$V_{NB} = \arg\max_{v_j \in V} P(v_j)\prod_i P(a_i \mid v_j) \tag{5}$$

上式即是朴素贝叶斯分类器所使用的方法。

因为上述方法的计算开销很大，所以使用多个贝叶斯分类器，对样本的特征向量分成多个子集分别计算，然后用一个多重贝叶斯算法对多个朴素贝叶斯分类器获得的结果进行综合求值：

$$P_{NB}(V \mid F) = \prod_{i=1}^{|NB|} P_{NB_i}(V \mid F) / P_{NB_i}(V) \tag{6}$$

上式中 NB_i 指的是朴素贝叶斯分类器，$|NB|$ 显其数目，NB 代表多重联合贝叶斯分类器。$P_{NB}(V \mid F)$ 和 $P_{NB}(V)$ 是给定分类的先验概率。

经过上述修正后，计算待测样本 x 类别 V_{NB} 的公式为

$$V_{NB} = \arg\max_{v_j \in V}(P_{NB}(V) \times P_{NB}(V \mid F)) \tag{7}$$

学习中的离散目标分类函数为：$f : A^n \to V$，其中 V 是有限集合 $\{v_1, v_2, ..., v_s\}$，即各不同分类集。

3 未知病毒检测

3.1 病毒特征提取

病毒与一般程序的区别在于执行了一些特殊的动作来破坏系统。从外在表现来看，如木马程序常进行如下的操作：对 win.ini 文件夹特定项的修改；对 system.ini 文件的特定项的修改；对注册表特定键值的修改，文件关联；端口异常与此端口上的数据流量；硬盘数据共享；远程文件操作；远程抓屏操作；计算机关闭与启动；鼠标键盘的操纵；远程执行可执行程序；消息发送；进程管理；更改服务；文件加壳；修改注册表；文件打开文件复制；文件修改等。

表 1 特征向量

序号	程序行为	相关 API 调用	危险度	动态链接库
1	堆操作	RtlFreeHeap；RtlAllocateHeap	☆☆	NTDLL.DLL
2	动态库加载与释放	LoadLibraryA；FreeLibrary；GetModuleHandleA；GetModuleFileNameA	☆	KERNEL32.dll
3	API 地址获取	GetProcAddress	☆	
4	进程操作	OpenProcess；CloseHandle；CreateRemoteThread；ExitThread；WaitForSingleObject；CreateThread	☆☆	
5	内存读写	VirtualAlloc；GetProcessHeap；VirtualAllocEx；WriteProcessMemory；VirtualFreeEx； OpenMutexA；CreateMutexA； VirtualProtect；VirtualFree；HeapFree	☆☆	
6	读注册表	RegCloseKey；RegEnumKeyExA；RegOpenKeyExA；RegQueryValueExA；RegNotifyChangeKeyValue；	☆	
7	写注册表	RegSetValueExA；RegCreateKeyExA ；RegDeleteKeyA	☆☆	
8	程序执行	WinExec；CreateProcess	☆☆	
9	文件读 & 创建	CreateFileA；ReadFile；OpenFile	☆	
10	文件写	WriteFile；WriteFileEx	☆☆	
11	文件删除	DeleteFileA	☆☆☆	
12	文件移动	MoveFileA；MoveFileExA；	☆☆	
13	变更属性权限	SetFileAttributesA；SetFileTime	☆☆	
14	文件搜索	FindClose；FindFirstFileA；FindNextFileA；FindResourceA	☆☆☆	
15	目录搜索	GetWindowsDirectoryA；GetSystemDirectoryA；SetCurrentDirectoryA；CreateDirectoryA	☆☆☆	
16	目录删除	RemoveDirectoryA	☆☆☆	
17	磁盘操作	GetDiskFreeSpaceA；GetDriveTypeA	☆	
18	时间操作	GetTickCount；GetSystemTime；GetTickCount；GetLocalTime	☆	
19	系统重启	Exitwindows;ExitwindowsEx;AbortSystemShutdown;InitialteSystemShutdown	☆☆	
20	加密与解密	CryptAcquireContextA;CryptGenKey;CryptDestroyKey;CryptImportKey;CryptExportKey;CryptEncrypt;CryptDecrypt;CryptReleaseContext;	☆☆☆	ADVAPI32 .dll
21	远程通信	WSAStartu;Socket;Connect;Recv;Send;Bind;Listen;Accept;Gethostname;Gethostbyname;Closesocket;WSACleanup	☆☆	WSOCK32.dll
22	进程搜索	EnumProcesses;EnumProcessModules;GetModuleBaseNameA	☆☆	PSAPI.dll
23	注册表操作 2	SHDeleteValueA；SHGetValueA；SHSetValueA	☆☆	SHLWAPI.DLL

不管是二进制可执行病毒、脚本病毒还是宏病毒，它们都是一种程序，需要调用操作系统提供的各种功能函数才能达到传播自身和破坏系统的目的。因此可以将下列信息作为待检测程序的行为特征：(1) 程序使用的动态链接库（DLL）文件；(2)程序在所使用的 DLL 中调用的 API 函数[6]；(3)程序的机器代码序列或源代码等。

在试验过程中，主要检测程序用到的 API 函数，选取的特征属性如表 1，并对不同的程序的行为特征划分为 3 个危险级别，即一般、较严重、严重，用☆、☆☆、☆☆☆表示。

然后，我们编写了相应的特征提取工具，对待检测程序的上述特征进行获取。

3.2 病毒检测引擎

通过搜集大量的同类程序可执行代码可以构成样本空间 U，利用现有的病毒检测工具对每一个程序，准确地归类为两种类型：病毒程序和普通正常程序。将样本空间划分为训练集 $S_{training}$ 和测试集 S_{test} 两个集合（训练集和测试集为样本空间中两个不相交的子集 $S_{training} \cap S_{test} = \varnothing$）。另将训练集划分为病毒程序集 S_{virus} 和普通程序集 S_{normal}，且 $S_{virus} \cap S_{normal} = \varnothing$，$S_{virus} \cup S_{normal} = S_{training}$。然后根据病毒类型构造相应的特征提取工具对程序进行特征提取，并应用 M-NB 算法对训练集进行分类。同时，当新的病毒样本添加到样本空间后。可以重复上述过程进行新一轮的学习。

计算机病毒检测是一个二值分类问题，即病毒与非病毒两类。对于样本空间中的每一个样本程序 x，均可以从中提取出感兴趣的一组特征集 F，定义 C 为分类集，即{正常，病毒}，令 C_1 表示正常，C_2 表示病毒，它是一个随机变量。我们的目标是：在获得给定样本程序文件中的特征集 F 后，判别出该样本是正常程序或是病毒程序的概率，即计算机出 P(C|F)。用朴素贝叶斯公式计算为

$$P(C \mid F) = \frac{P(F \mid C) \times P(C)}{P(F)} \tag{8}$$

因为特征向量 F 中又包含有分量 $F_1, F_2, ..., F_n$（本文中 n=88），所以上式可以写成

$$P(C \mid F) = \frac{\prod_{i=1}^{n} P(F_i \mid C) \times P(C)}{\prod_{j=1}^{n} P(F_j)} \tag{9}$$

最后根据 P(C|F)的最大值来确定 F 最大可能的类。即待测样本 S 的类别 C_s 为

$$C_s = \arg\max_{C}(P(C) \prod_{j=1}^{n} P(F_j \mid C)) \tag{10}$$

因为上述方法的计算量很大，所需 RAM 开销量也较大，所以为改善性能，在此根据各个特征量的“危险度”不同，将所有的特征量分成 3 类，对每一类分别使用一个朴素贝叶斯分类器进行分类，最后用一个多重贝叶斯算法将 3 个分类器计算所得的待测样本所属类别的概率综合求值，作为样本程序分类的最终依据。

$$P_{NB}(C \mid F) = \prod_{i=1}^{|NB|} P_{NB_i}(C \mid F) / P_{NB_i}(C) \tag{11}$$

上式中，NB_i 指的是朴素贝叶斯分类器（本文为 3 个，即 $|NB| = 3$），NB 代表多重联合贝叶斯分类器。$P_{NB}(C \mid F)$ 和 $P_{NB}(C)$ 是给定分类的先验概率。

经过上述修正后，计算待测样本 x 类别 C_x 的公式为

$$C_x = \arg\max_{C}(P_{NB}(C) \times P_{NB}(C \mid F)) \tag{12}$$

像其它的数据挖掘算法一样，贝叶斯分类器也需产生规则集以预测新的实例类别。朴素贝叶斯发现的分类规则是基于统计方法得到的，示例如表 2。当新的实例加入样本空间后，就可进入新一轮的学习，更新分类，如此往复。

表 2 贝叶斯算法发现的分类规则示例

P（“RegCreateKey”|正常）=110/373
P（“RegCreateKey”|病毒）=134/159
P（“FindNextFileA”|正常）=76/373
P（“FindNextFileA”|病毒）=142/159

实验中使用的详细算法如下：

（1）训练算法

□Input: 训练样本集

□Output: 分类规则

□procedure:

1)对训练集中全部样本的特征进行提取

2)统计各特征量出现的频率

3)return classification rules

（2）分类算法

□input: 一个要分类的查询实例 x_q

□output:类别 C

□procedure:

1)提取出特征向量

2)计算：

$C_x = \arg\max_{C}(P_{NB}(C) \times P_{NB}(C \mid F))$

3)return c_x,式中 $c_x \in$ {正常,病毒}

（3）自学习算法

□Input: train sample set L,test sample set T

□Output: classification rules

□Procedure:

While (T!=Φ)

{ For each $x_j \in T$

{call 分类算法，对 x_j 分类，并将其加入到样本空间的相应集合中；

call 训练算法，更新分类规则}

return classification rules}

4 实验分析

用于实验的样本数据如表 4 所示。样本空间中样本总数为 632，分为正常程序与染毒程序。正常程序从操作系统平台中选取，我们选用的 Windows 2000 Server 首次安装后，机器中的全部 PE 文件共 423 个。另外通过各种途径收集 PE 格式病毒程序 209 个。为获得样本空间中程序的特征向量，我们编写了 API 调用跟踪器，可以实现 Windows 2000 Server 环境下的全部 API 函数调用的拦截。

表 4 用于实验的样本数据

	样本空间	训练集	测试集
正常程序	423	373	50
染毒程序	209	159	50
合计	632	532	100

实验目标主要是对 M-NB 查病毒引擎的错误率进行测试。我们将错误类型分为 2 种，即：(1) 将正常程序判断为病毒，称为 False Negative；(2) 将病毒程序判断为正常程序，称为 False Positive。

用测试集（共 532 个文件）作为训练数据先对分类器进行训练，然后对测试集中的文件（共 100 个）进行分类测试，并进行自学习。然后用全体数据集作为训练数据，测试集从该样本空间中随机选出不等数目的文件而获得，结果见图 3。

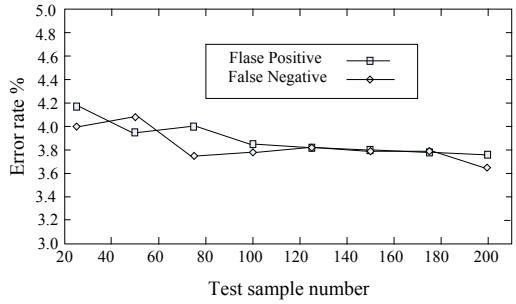

图 3 测试样本数与错误率的关系

从实验结果知，分类器用较少训练集中的样本测试与用较多训练样本进行测试获得的精度几乎相等，这对于计算机病毒样本较难获得的情况下检测未知病毒非常有用。

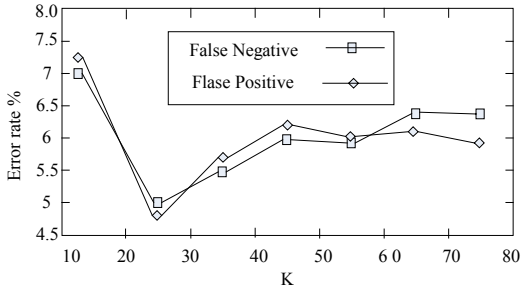

图 4 基于 K 最近邻算法的检测系统性能

我们曾利用基于实例学习的 K-最近邻算法对本组样本空间中的数据进行过测试[7]，结果如图 4 所示，基于 KNN 算法的检测系统在 K=25 时的 False Negative error rate 最小，约为 4.8%。通过对这 2 种不同的算法结果的比较，我们发现使用多重贝叶斯分类算法得到的 False Negative error rate 最小约为 3.65%，效果稍好。但 KNN 算法开销很小，而 M-NB 算法的开销则大得多。在实际应用中这两方面都要兼顾。

参考文献

1 Cohen F. Computer Viruses: Theory and Experiments [J]. Computers & Security, 1987, 6(1): 22-35.

2 Spinellis D. Reliable Identification of Bounded-length Viruses is NP-Complete [J]. IEEE Transactions on Information Theory, 2003, 49(1): 280-284.

3 Tesauro G J, Kephart J O, Sorkin G B. Neural Networks for Computer Virus Recognition [J]. IEEE Expert, 1996,11(4): 5-6.

4 Mitchell T M. Machine Learning [M]. New York: McGraw-Hill, 1997.

5 Lo R, Levitt K, Olssonn R. MFC: A Malicious Code Filter [J]. Computer & Security, 1995,14 (6): 541-566.

6 Kruglinski D J, Wingo S, Shepherd G. Programming Visual C++ [M]. Washington: Microsoft Press, 1998.

7 张波云, 殷建平, 张鼎兴等. 基于 K-最近邻算法的未知病毒检测 [J]. 计算机工程与应用, 2005, 41(6): 7-10.

8 祝　恩, 殷建平, 蔡志平等. 计算机病毒自动变形机理的分析 [J]. 计算机工程与科学, 2002,24(6):14-17.

(上接第 17 页)

位数据、1 位奇偶校验。UART 位串中数据位占 72.7%，但若要考虑一次传送信息，由于位串间还可能存在若干传号，因此实际的传输效率小于 60%。为保证数据传输质量与快速性，对每个字节进行校验的同时，在自定义高层协议中应尽量减少附加信息以提高有效传输率 br。这里

$$br = Br \times L_v /(L_v + L_a) \tag{21}$$

其中 L_v 为有效数据长度，L_a 为附加数据长度。在计算附加数据长度时除了帧中的附加数据外，还应包括 UART 位串中非数据位及位串间的若干传号。

4.2 传输线的“忙闲”程度衡量

衡量传输线的“忙闲”程度，即单位时间内平均有多少比特在通信线上传输，这里用单位时间内的比特吞吐率。它是指节点单位时间内发送/接收的比特数目，它不仅与波特率有关，还与 UART 的硬件特性和 CPU 特性相关。

令某次通信所传送信息所用的时间为 t_c，而前处理与后处理所用时间为 t_p，则比特吞吐率为

$$\xi = B_r \frac{t_c}{t_c + t_p} \tag{22}$$

要提高比特吞吐率必须减小 $t_c + t_p$，增大 t_c。

4.3 数据理解率

数据理解率是衡量节点通信效果和应用系统固件特性的重要参数，表示单位时间可以理解多少 UART 位串。令节点收到能代表完整意义的 l 个位串所需时间为 t_r，理解这些位串时间为 t_u，则数据理解率为

$$\eta = l /(t_r + t_u) \tag{23}$$

η 值越大，节点的整体固件特性越好。

4.4 DLL 层的时态特性分析

对于 M/S 方式主要是分析在 LL 层协议一定的情况下主节点访问从节点的时间和轮询从节点 1 周所用的时间,分析的目的是为了考查 M/S 系统能否满足现场对通信的要求,如实时性、是否影响主节点软件的操作等。对于令牌方式的系统则要分析 DLL 层协议中通信参数间的关系，及在满足通信要求下应该使用的报文调度方式以增加对实时节点的响应能力。对于多主对等方式则要利用随机理论和排队论对 LL 层协议规定的“忙闲”检测、冲突检测、冲突解决策略进行评估，并研究负载对通信性能的影响。

参考文献

1 范晓烨, 王　豪, 张道中. 伺服控制系统的计算机实现[J]. 计算机工程, 2004, 30(11): 174-176.

2 Lian F L, Moyne J R, Tibury D M. Performance Evaluation of Control Networks: Ethernet, ControlNet, DeviceNet[J]. IEEE Control Systems Magazine, 2001, 21(1): 66-83.

3 吴军辉, 林开颜, 徐立鸿. RS485 总线通信避障及其多主发送的研究[J]. 测控技术, 2002, 21(8): 41-43.

4 吴忻生, 陈　安, 胡跃明等. 一种实现载波监听多点接入/冲突检测的多主 RS485 总线[J]. 电子技术应用, 2004, 20(2): 48-50.

5 Elmenreich W, Delvai M. Time-triggered Communication with UARTs[C]. 4th IEEE International Workshop on Factory Communication Systems, Vasteas, Sweden, 2002.