

文章编号:1007-130X(2007)09-0019-04

# 基于 SVM 的计算机病毒检测系统 A SVM-Based Computer Virus Detection System

张波云<sup>1,2</sup>,殷建平<sup>1</sup>,蒿敬波<sup>1</sup>

ZHANG Bo-yun<sup>1,2</sup>, YIN Jian-ping<sup>1</sup>, HAO Jing-bo<sup>1</sup>

(1. 国防科技大学计算机学院, 湖南 长沙 410073; 2. 湖南公安高等专科学校计算机系, 湖南 长沙 410138)

(1. School of Computer Science, National University of Defense Technology, Changsha 410073;

2. Department of Computer Science, Hunan Public Security College, Changsha 410138, China)

**摘 要:**自从第一例计算机病毒被发现以来,特征码法一直是病毒检测的基本方法。但是,病毒的复杂化和变形病毒的出现,限制了该法的有效应用。本文提出一种基于支持 SVM 的通用病毒智能检测方法,通过支持 SVM 算法的应用,使得检测系统在小样本的情形下仍具有良好的泛化能力。然后,以系统 API 函数调用执行迹为例,测试了该法的检测性能,并将实验结果与其他检测方法进行了比较。实验表明,API 函数调用序列在区分正常与恶意 PE 格式程序文件上有很好的辨别力,发现基于支持 SVM 的病毒检测系统所需要的先验知识小于其他方法。而且,当检测性能相当时,系统的训练时间将会缩短。

**Abstract:** Since the first computer virus was found, scanning detection has been used as a primary method in virus detection systems. As viruses become more complex and sophisticated, the scanning detection method is no longer able to detect the various forms of malicious code effectively. We explore the idea of automatically detecting viruses based on Support Vector Machine (SVM) and not strictly dependent on certain viruses. By utilizing SVM, the generalizing ability of virus detection systems is still good when the sample size is small. An experiment using the system API function call trace is given to illustrate the performance of this method. Finally, the comparison of detection abilities between the above detection method and others is given. Evidence shows that the sequences of the operating system API function calls executed by the running programs are a good discriminator between benign and malicious PE files, the detection system based on SVM needs less priori knowledge than other methods, and can shorten the training time under the same detection performance condition.

**关键词:** 计算机病毒; 支持向量机; 病毒检测

**Key words:** computer virus; support vector machine; virus detection

**中图分类号:** TP309.5

**文献标识码:** A

## 1 引言

传统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,对于新出现病毒的检测无能为力<sup>[1]</sup>。

长期以来,为了解决上述问题,各反病毒研究机构都在

努力探讨病毒检测的智能方法。在 IBM 病毒研究中心,曾经成功地将神经网络用于判断引导型病毒<sup>[2]</sup>。Schultz M 等<sup>[3]</sup>曾提出用数据挖掘的算法如朴素贝叶斯算法等检测未知恶意代码,其算法建立在对程序静态分析的基础上,他们直接选用程序的机器码(用 hex dump 获取)、程序中的 ASCII 字符串以及对 PE 程序的静态分析而取得的 API 引用序列作为样本程序的特征向量,各算法以此为基础进行学习与分类。以上方法都需要大量或者是完备的数据集才能达到比较理想的检测性能,并且训练时间较长。在现实环境中一般较难获得大量的恶意代码,那么在小样本情况

• 收稿日期:2006-03-03; 修订日期:2006-07-13

基金项目:国家自然科学基金资助项目(60373023); 湖南省教育厅青年基金资助项目(05B072)

作者简介:张波云(1972~),男,湖南永州人,博士生,研究方向为网络与信息安全;殷建平,教授,博士生导师,研究方向为算法设计与分析、人工智能、模式识别、信息安全等;蒿敬波,博士生,研究方向为网络与信息安全。

通讯地址:410073 湖南省长沙市国防科技大学计算机学院博士生队; Tel:13873198965; E-mail:hnjxzy@yahoo.com.cn

Address: Doctoral Brigade, School of Computer Science, National University of Defense Technology, Changsha, Hunan 410073, P. R. China

下如何实现高效的检测呢?针对该问题,本文提出一种基于支持向量机(Support Vector Machine,简称 SVM)的病毒检测模型,检测算法采用的特征为程序执行过程中使用的系统 API 函数的调用迹,检测系统在虚拟机中监视程序的行为并进行分析,能有效地检测未知病毒和各种多态与变形病毒,而且该法所需训练样本量也较少。

## 2 系统框架

系统结构框图如图 1 所示。系统由病毒防火墙、应用服务器、虚拟机和病毒检测服务器组成。该方法是对病毒行为进行跟踪进而获取其特征,而病毒对真实机器的破坏不可预料,故特别设置一台虚拟机对程序行为进行监控以规避风险。该虚拟机基于 VMware Workstation<sup>[1]</sup> 实现, VMware 公司开发的该软件能在一台实体机器上同时运行多种操作系统与应用程序。这些操作系统与应用程序共用硬件装置,但在逻辑上各自独立运行,互不干扰。VMware 的虚拟层映射实体的硬件资源到本身的虚拟机器资源,每个虚拟机器都有各自的 CPU、内存、硬盘、I/O 设备、BIOS 等,虚拟机器完全等同于一个标准的 x86 计算机。

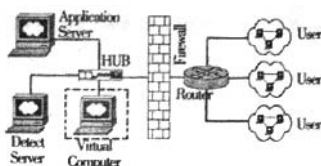


图 1 系统结构

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:一份拷贝直接存放于应用服务器中;一份拷贝则进入虚拟机中,对其行为进行监控,跟踪其 API 调用序列,然后将获得的信息发送到基于 SVM 的检测服务器中,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,检测服务器则将信息反馈给应用服务器,然后在应用服务器中对该文件进行隔离、监管或删除。

## 3 病毒检测引擎

### 3.1 基本定义与假设

**定义 1** 计算机病毒是一种在计算机系统运行过程中非法获得控制权,把自身精确拷贝或有修改地拷贝到其他程序体内的程序。计算机病毒由主控模块、传染模块、触发模块及破坏模块组成,其本质是复制和非授权的程序加载。

**假设 1** 恶意代码运行时与操作系统交互。

大多数的程序行为如访问网络或文件服务均会与操作系统交互,但有的恶意行为如拒绝服务、破坏数据等不与操作系统交互,此类恶意行为为本系统不能检测。

**假设 2** 恶意代码通过 API 函数与操作系统交互。

在 Windows 平台中,恶意代码可以通过 Win32 API 或 Windows NT native API function 与操作系统交互。本系统目前仅能检测前者,而只要做简单的扩展也可以检测后者。

**假设 3** 染毒程序执行时,其传染模块均会运行,而其破坏模块不一定会运行。

**假设 4** 可执行程序的系统调用序列在运行时可以被有效地监控。

上述定义与假设是本文研究工作的前提。

### 3.2 样本提取

**恶意程序与一般程序的区别主要在于执行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。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。New Mexico 大学的 Forrest 教授<sup>[5]</sup>曾对 Unix 中的系统调用序列在入侵检测中的分类作用进行了深入的研究。受此启发,在我们的实验中以程序运行过程中调用到的 API 函数为研究对象。

#### 3.2.1 窗口尺寸

首先,我们在虚拟机中对样本库中的程序进行行为监控,跟踪每个程序的 API 调用过程,可以得到大量的系统调用迹。每个执行迹数据文件为程序的系统 API 调用函数在系统 API 调用名称列表(mapping file)中的索引值,如‘35’代表‘openfile( )’,如图 2b 所示。该数据已是数字序列,然后用长度为  $k$  的窗口在程序执行迹上滑动来得到这个执行迹的系统调用短序列,如图 2c 所示。

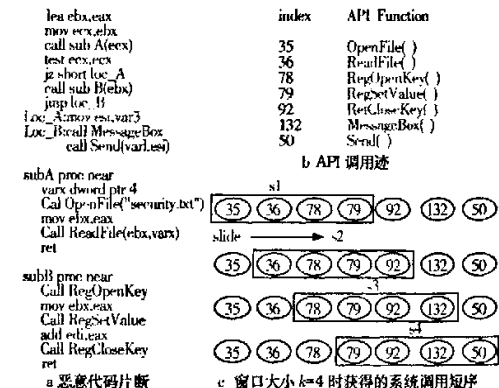


图 2 API 调用序列

系统调用短序列反映了进程执行过程中系统调用之间的次序关系。那么,窗口大小(即短迹的长度)选取何值是最合适的?如果选取的短序列长度为 1,就丢掉了系统调用的次序信息,而长度太大,就丢掉了系统执行的局部信息状况,无法正确反映正常和异常情况下的局部序列调用状况。Hofmeyr S A<sup>[6]</sup>从实验中得出结论:当窗口大于 30 时,从调用序列中得不到对程序行为判定有用的信息。根据 Lee W<sup>[7]</sup>的研究,从条件熵和计算成本的综合角度研究数据长度的选择情况,认为最合适的系统调用短序列长度为 6~7。

### 3.2.2 训练程序集中的异常调用短序样本获取

SVM的参数是通过训练得到的,所以需要获得两类训练样本,即正常短序列样本和异常短序列样本。

用长度为 $k$ 的滑动窗口对正常程序的系统调用执行迹进行扫描,可以得到正常的系统调用短序列样本。将这些样本保存于正常样本数据库中以供后用。一般该数据库中记录条数最多为 $|\Sigma|^k$ ,其中 $\Sigma$ 为系统调用集, $|\Sigma|$ 为系统API函数数目, $k$ 为窗口大小。实际应用中因为可以将冗余记录删除,其规模会小得多。

当用长度为 $k$ 的滑动窗口对恶意代码的执行迹进行扫描时,会得到一组既有正常短序列又有异常短序列的系统调用短序列列表。由于恶意的非法活动只占程序执行的小部分,所以异常短序列只占全部执行迹的很小部分。当获取恶意代码调用短迹后,其中有一部分与正常样本库中的记录匹配,可以将它们删除。而对不匹配的短序列,我们用汉明距离来测量其与正常样本的相异程度。对于两条短序列 $i$ 与 $j$ ,它们之间的汉明距离记为 $d(i, j)$ 。对于每一条新的序列 $i$ ,定义最短汉明距离为 $d_{\min}(i) = \min\{d(i, j), \forall \text{ 正常序列 } j\}$ 。 $d_{\min}(i)$ 的值表达了序列与正常模式的偏差程度,该值与短迹的长度无关。

最后,对于不匹配的序列 $i$ ,根据其 $d_{\min}(i)$ 与某个预定的阈值 $D$ 进行比较以判定其是否为异常,即若 $d_{\min}(i) \geq D$ ,式中 $1 \leq D \leq k$ 为阈值,则认为序列 $i$ 为异常序列。据此可以获得异常短序样本集。得到 $l$ 个样本后,可将所有的短序列样本记为 $(x_1, y_1), \dots, (x_l, y_l) \in R_k \times \{\pm 1\}$ 的形式,其中正常样本的标签为+1;异常样本为-1。

### 3.3 检测方法

本文的检测方法基于SVM来实现。SVM的最大特点是根据Vapnik<sup>[8]</sup>结构风险最小化原则,尽量提高学习机的泛化能力,即由有限的训练集样本得到小的误差仍然能够保证对独立的测试集保持小的误差。另外,由于支持向量算法是一个凸优化问题,其局部最优解一定是全局最优解。这是其他学习算法所不及的。将SVM应用到计算机病毒检测中,可以保证在先验知识不足的情况下,SVM有较好的分类正确率。这在较难获得大量病毒样本的情况下十分有利。

病毒检测可以看成是一个二分类的问题,对于分类问题,SVM算法根据区域中的样本计算该区域的决策曲面,由该曲面确定该区域中样本的类别。这里我们主要讨论计算决策面和对样本分类的方法。

#### 3.3.1 线性 SVM

样本 $x$ 为 $k$ 维向量,在某区域内的 $l$ 个样本所属类别为 $(x_1, y_1), \dots, (x_l, y_l) \in R_k \times \{\pm 1\}$ 。若超平面:

$$w \cdot x + b = 0 \quad (1)$$

能将样本分为两类,其中 $\cdot$ 表示向量的点积。最佳的超平面应使两类样本到超平面的距离为最大。显然,式(1)中的 $w$ 和 $b$ 乘以系数后仍能满足方程。不失一般性,对于所有的样本 $x_i$ , $|w \cdot x_i + b|$ 的最小值为1,则样本与此最佳超平面的最小距离为 $|w \cdot x_i + b| / \|w\| = 1 / \|w\|$ 。最佳超平面应满足约束:

$$y_i[w \cdot x_i + b] \geq 1, i = 1, \dots, l \quad (2)$$

$w$ 和 $b$ 的优化条件应该是使两类样本到超平面最小距离之和 $2 / \|w\|$ 最大。另外,考虑到可能存在一些样本不能被超平面正确分类,因此引入松弛变量:

$$\xi_i \geq 0, i = 1, \dots, l \quad (3)$$

问题变成在式(2)和式(3)条件下最小化

$$\frac{1}{2} \|w\|^2 + C \cdot \sum_{i=1}^l \xi_i \quad (4)$$

式中的 $C$ 为一个正常数,式(4)中第1项使样本到超平面的距离尽量大,从而提高泛化能力,第2项则使误差尽量小。利用Lagrange乘子法,可以把式(4)变成对偶形式,从而有:

$$\begin{aligned} \max W(a) &= \sum_{i=1}^l a - \frac{1}{2} \sum_{i,j=1}^l a_i y_i a_j y_j (x_i \cdot x_j) \\ \text{s.t. } \sum_{i=1}^l a_i y_i &= 0, a_i \in [0, C], i = 1, \dots, l \end{aligned} \quad (5)$$

以及

$$w = \sum_{i=1}^l a_i y_i x_i \quad (6)$$

可以证明,在此优化问题的解中有一部分 $a_i$ 不为0,它们所对应的训练样本完全确定了这个超平面,因此称其为支持向量。按照优化理论的Kuhn-Tucher定理,在鞍点,对偶变量与约束变量的乘积为0,从而求得超平面的另一个参数 $b$ 满足:

$$y_i(w \cdot x_i + b) = 1 \quad (7)$$

对于未知属类的向量 $x$ ,可以采用线性判决函数:

$$f(x) = \text{sgn}(w \cdot x + b) \quad (8)$$

来判定其所属类别。综合式(6),得到:

$$f(x) = \text{sgn}\left(\sum_{i=1}^l a_i y_i (x_i \cdot x) + b\right) \quad (9)$$

#### 3.3.2 非线性 SVM

在线性SVM训练算法中,数据以点积的形式 $(x_i \cdot x_j)$ 出现。现在用非线性映射把输入空间映射到某一特征空间,记为: $\Psi: R_k \rightarrow H$ 。如果存在一种核函数 $K$ ,使得:

$$K(x_i, x_j) = (\Psi(x_i) \cdot \Psi(x_j)) \quad (10)$$

可以在特征空间中进行许多计算,而不需要知道具体的映射 $\Psi$ 。

现在可用核函数代替线性SVM的点积形式,式(5)的对偶规划可变为:

$$\begin{aligned} \max W(a) &= \sum_{i=1}^l a - \frac{1}{2} \sum_{i,j=1}^l a_i y_i a_j y_j K(x_i, x_j) \\ \text{s.t. } \sum_{i=1}^l a_i y_i &= 0, a_i \in [0, C], i = 1, \dots, l \end{aligned} \quad (11)$$

非线性SVM的判决函数为:

$$f(x) = \text{sgn}\left(\sum_{i=1}^l a_i y_i K(x_i, x) + b\right) \quad (12)$$

本文提出的病毒检测系统使用的是有监督的学习算法,故首先将训练用的API调用短序列样本做标记,然后对SVM分类器进行学习训练以获得其参数。对于一个待检测程序文件,跟踪其API调用后同样可获得其调用序列,按给定的窗口大小 $k$ 将其分割成调用短序列,将它们输入SVM分类器,然后根据式(12)可以得到判决向量,即可以得到被检测程序的每一API调用短序列是否异常的标

记。

由于我们获得训练样本时所得到的正常短序列样本并不完备,导致在基于正常短序列上获得的异常短序列样本中可能包含正常的短序列,从而使得 SVM 的分类器产生一些分类错误。并且,SVM 分类器本身也具有不精确性,所以需要设定某些判断规则来提高整个检测系统的性能。我们在检测器中设置了一个决策模块,它根据一个被检测程序文件中的异常 API 调用短序列的数目对其是否染毒进行判断。首先,选定一个阈值,然后对于待检测程序执行迹的短序列进行分类。如果异常系统调用短序列的数目超过阈值,则判定被检测程序文件已染病毒;反之,判定为正常。该阈值可由训练数据进行实验确定。

4 实验结果

用于实验的样本程序文件如表 1 所示。数据集中程序文件总数为 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,本文选用的是 Windows 2000 Server,首次安装后,机器中的全部 PE 格式文件共 423 个。病毒样本程序为当前广泛流行的 Windows 平台下 PE 格式病毒,从网站 <http://vx.netlux.org> 及 <http://www.cs.columbia.edu/ids/mef/> 获得,共计 209 个。经过预处理后得到训练用短序数据和测试用短序数据的分布情况如表 2 所示。

表 1 实验数据集

| 样本程序           | Sample space | Training set | Testing set |
|----------------|--------------|--------------|-------------|
| Benign file    | 423          | 50           | 373         |
| Malicious file | 209          | 50           | 159         |
| Sum            | 632          | 100          | 532         |

表 2 实验短序样本分布情况

| 训练数据集 |      | 测试数据集 |      |
|-------|------|-------|------|
| 正常程序  | 染毒程序 | 正常程序  | 染毒程序 |
| 496   | 242  | 2 766 | 876  |

对于实验数据集,我们用 SVM 算法进行学习和泛化。本文所用的分析工具是 LIBSVM<sup>[9]</sup> 软件包,实验目标主要是对查毒引擎的错误率进行测试。我们将错误类型分为两种:(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。

学习样本确定后,实验主要是选择相应的 SVM 参数:核函数和错分惩罚因子 C。本文通过对各种核函数的测试,最终选择了径向基核函数  $K(x,y)=\exp(-\|x-y\|^2/\sigma^2)$ 。对于 SVM 算法中的两个重要参数  $1/\sigma^2$  和 C,给出几种不同的组合,测试在各种组合下 SVM 的分类性能。实验中,短序列的长度  $k$  分别取 6 和 7,在判定训练文件集异常短序列时阈值  $D$  取 4,实验结果见表 3。

表 3  $k=6,7$  时 SVM 分类正确性实验结果

| C   | $\sigma^2$ | False Negative(%) |       | False Positive(%) |       |
|-----|------------|-------------------|-------|-------------------|-------|
|     |            | $k=6$             | $k=7$ | $k=6$             | $k=7$ |
| 50  | 10         | 3.21              | 4.02  | 5.66              | 7.54  |
| 100 | 1          | 4.82              | 5.63  | 6.28              | 5.66  |
| 200 | 0.5        | 6.97              | 7.50  | 10.06             | 11.32 |

我们曾在文献[10]中使用模糊模式识别(FPR)分类算法对本文样本空间中的数据进行过测试,基于 FPR 的检测

系统的 False Negative error rate 最小值约为 4.45%。本文算法得到的 False Negative error rate 最小值约为 3.21%,效果稍好。另外,本文所选用的训练样本数也少得多。比较二者实验结果知,SVM 分类器用较少训练集中的样本测试与 FPR 分类器用较多训练样本进行测试获得的精度几乎相当,这对于计算机病毒样本本较难获得的情况下检测未知病毒非常有用。

5 结束语

本文初步探讨了支持向量机在计算机病毒检测方面的应用,提出基于支持向量机的检测框架和工作原理。实验证明,该模型能非常有效地将正常和异常的系统 API 函数调用序列区分开来,只需要很少的一部分病毒程序样本数据做训练,就能得到较高的检测成功率。它对系统性能的影响很小,是一种高效低负荷的检测方法。在我们的测试床中,将计划进行其它的基于机器学习算法的恶意代码检测实验,以期对其它的恶意代码如木马、蠕虫、间谍程序等进行检测测试;并对各种方法的性能与开销进行对比、选择和优化,以用于实际的产品中。

参考文献:

[1] Spinellis D. Reliable Identification of Bounded-Length Viruses Is NP-Complete[J]. IEEE Trans on Information Theory, 2003, 49 (1): 280-284.

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

[3] Schultz M, Eskin E, Zadok E, et al. Data Mining Methods for Detection of New Malicious Executables[A]. Proc of the 2001 IEEE Symp on Security and Privacy[C]. 2001. 38-49.

[4] VMware[EB/OL]. <http://www.vmware.com>, 2003-11.

[5] Forrest S, Hofmeyr S A, Somayaji A. Computer Immunology[J]. Communications of the ACM, 1997, 40(10): 88-96.

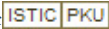
[6] Hofmeyr S A, Forrest S, Somayaji A. Intrusion Detection Using Sequences of System Calls[J]. Journal of Computer Security, 1998, 6(3): 151-180.

[7] Lee W, Dong X. Information-Theoretic Measures for Anomaly Detection[A]. Proc of the 2001 IEEE Symp on Security and Privacy[C]. 2001. 130-143.

[8] Vapnik V N. The Nature of Statistical Learning Theory [M]. New York: Springer-Verlag, 1995.

[9] Libsvm[EB/OL]. <http://www.csie.ntu.edu.tw/~cjlin>, 2004-10.

[10] Zhang B Y, Yin J, Hao J. Using Fuzzy Pattern Recognition to Detect Unknown Malicious Executables Code[J]. Proc of FSKD'05[C]. 2005. 629-634.

作者: 张波云, 殷建平, 蒿敬波, ZHANG Bo-yun, YIN Jian-ping, HAO Jing-bo  
作者单位: 张波云, ZHANG Bo-yun(国防科技大学计算机学院, 湖南, 长沙, 410073; 湖南公安高等专科学校计算机系, 湖南, 长沙, 410138), 殷建平, 蒿敬波, YIN Jian-ping, HAO Jing-bo(国防科技大学计算机学院, 湖南, 长沙, 410073)  
刊名: 计算机工程与科学   
英文刊名: COMPUTER ENGINEERING AND SCIENCE  
年, 卷(期): 2007, 29(9)  
被引用次数: 3次

## 参考文献(10条)

1. Spinellis D [Reliable Identification of Bounded-Length Viruses Is NP-Complete](#) 2003(01)
2. Tesauro G J; Kephart J O; Sorkin G B [Neural Networks for Computer Virus Recognition](#) [外文期刊] 1996(04)
3. Schultz M; Eskin E; Zadok E [Data Mining Methods for Detection of New Malicious Executables](#) [外文会议] 2001
4. [查看详情](#) 2003
5. Forrest S; Hofmeyr S A; Somayaji A [Computer Immunology](#) [外文期刊] 1997(10)
6. Hofmeyr S A; Forrest S; Somayaji A [Intrusion Detection Using Sequences of System Calls](#) 1998(03)
7. Lee W; Dong X [Information-Theoretic Measures for Anomaly Detection](#) [外文会议] 2001
8. Vapnik V N [The Nature of Statistical Learning Theory](#) 1995
9. Libsvm 2004
10. Zhang B Y; Yin J; Hao J [Using Fuzzy Pattern Recognition to Detect Unknown Malicious Executables](#) Code [外文会议] 2005

## 本文读者也读过(6条)

1. 张波云, 殷建平, 蒿敬波, 张鼎兴. ZHANG Boyun, YIN Jianping, GAO Jingbo, ZHANG Dingxing [基于多重朴素贝叶斯算法的未知病毒检测](#) [期刊论文]-计算机工程 2006, 32(10)
2. 张波云, 殷建平, 张鼎兴, 蒿敬波, 王树林. ZHANG Bo-yun, YIN Jian-ping, ZHANG Ding-xing, HAO Jing-bo, WANG Shu-lin [基于集成神经网络的计算机病毒检测方法](#) [期刊论文]-计算机工程与应用 2007, 43(13)
3. 朱刚, 张宁, 马良. ZHU Gang, ZHANG Ning, MA Liang [复杂网络上计算机病毒传播和控制策略研究](#) [期刊论文]-计算机应用研究 2006, 23(9)
4. 郑宇, 马蔚吟, 宋宝莲, 罗建平, ZHENG Yu, MA Wei-yin, SONG Bao-lian, LUO Jian-ping [一种基于PXE技术的计算机病毒防护方法](#) [期刊论文]-计算机与现代化 2009(4)
5. 张波云, 殷建平, 唐文胜. ZHANG Bo-yun, YIN Jian-ping, TANG Wen-sheng [一种未知病毒智能检测系统的研究与实现](#) [期刊论文]-计算机工程与设计 2006, 27(11)
6. 张波云, 殷建平, 唐文胜, 蒿敬波. ZHANG Bo-yun, YIN Jian-ping, TANG Wen-sheng, HAO Jing-bo [基于模糊模式识别的未知病毒检测](#) [期刊论文]-计算机应用 2005, 25(9)

## 引证文献(3条)

1. 王晓燕, 金聪, 谈华永 [基于Win32API和SVM的未知病毒检测方法](#) [期刊论文]-计算机工程与应用 2011(7)
2. 张敬, 姚书科 [数据挖掘在恶意程序检测中的应用](#) [期刊论文]-电脑知识与技术 2011(35)
3. 熊俊 [基于分类的未知病毒检测方法研究](#) [期刊论文]-电脑开发与应用 2012(11)

本文链接: [http://d.wanfangdata.com.cn/Periodical\\_jsjgcykx200709006.aspx](http://d.wanfangdata.com.cn/Periodical_jsjgcykx200709006.aspx)