改进贝叶斯算法在未知恶意软件识别中的研究

赖英旭，杨 震
（北京工业大学 计算机学院，北京 100124）

摘 要：为改进朴素贝叶斯（naive Bayes，NB）算法在识别未知恶意代码过程中学习速度慢的缺点，在分析研究朴素贝叶斯算法、复合贝叶斯（multi-naive Bayes，MNB）算法的基础上，提出了一种改进贝叶斯（half-increment naive Bayes，HNB）算法. 算法采用特征集增量学习方式，在保证分类精度不降低的前提下，学习速度提高约30%. 实际样本测试表明，分类精度达到了96%，其中对已知恶意代码的分类精度达到99%.

关键词：分类器；数据挖掘；贝叶斯算法
中图分类号：TP 309.5 **文献标志码**：A **文章编号**：0254－0037(2011)05－0766－07

因特网的发展和普及给人们交流信息带来便捷的同时，也为恶意代码的快速蔓延提供了平台. 识别恶意代码的传统方法是特征检测法，但此方法只能检测已知恶意代码，对未知恶意代码无能为力[1]. 这是因为，1987年Cohen[2]就提出了“恶意代码通用检测方法不可判定性”的著名论断，此后Spinellis[3]证明了有界长度病毒的可靠检测是一个多项式复杂程度的非确定性问题（non-deterministic polynomial，NP）问题. 因此，人们致力于研究新方法来检测未知恶意代码并将其消除.

1995年，Kephart等[4]首次提出将人工神经网络应用于未知恶意代码检测，能识别出80%～85%的未知引导型恶意代码，同时负误报率非常低（<1%）. 这种方法后来又被Tesauro等[5]应用于检测Win32程序并得到了验证. 随着数据挖掘方法在入侵检测中取得了非常好的效果[6-7]，Schultz[8]等将数据挖掘中的RIPPER、朴素贝叶斯（naive Bayes，NB）算法、复合贝叶斯（multi-naive Bayes，MNB）算法用于检测多种类型的未知恶意代码，测试结果表明复合贝叶斯的分类效果高于其他方法. 遵循上述思想，Kolter[9]和Zhang[10-12]等在此基础上又做了大量工作. Zhang[11-12]等又首次将模糊集理论和K-近邻的方法应用于未知恶意代码检测，取得了较好的结果. 在上述算法中提取的特征基本上都采用n-gram机器码，特征数量巨大，分类器学习速度较慢.

本文在文献[8-9]的基础上做了2处改进：1）提取字符串作为特征，这样比只使用头部信息（DLL名等）作为特征有了更高的可靠性. 因为头部信息在脱壳时极易被破坏；同时由于字符串的长度不是固定长度，比采用固定长度n-gram滑动窗口选取特征更能反映程序的行为特征. 2）在分析朴素贝叶斯算法和复合贝叶斯算法[13]的基础上，对特征集进行增量学习，大大提高了分类器学习速度，同时保证精度不降低. 本文对所提出方法的有效性和准确度作了理论上的分析和计算，并且对所得到的模型进行了实验测试. 在实验中使用了2 995个实验样例，取得了96%的分类精度，其中对已知恶意代码的分类精度达到99%，这个结果和文献[8-9]的效果接近，但学习速度快.

1 构造特征集

特征集F的选取直接影响到分类器的分类精度. 特征项可以采用资源描述符、字符串和机器码3种

收稿日期：2009-04-03.
基金项目：国家重点基础研究发展计划资助项目（2007CB311100）；北京市教委基金资助项目（KM200810005030）；北京市自然科学基金资助项目（4102012）；国家自然科学基金资助项目（61001178）.
作者简介：赖英旭（1973—），女，辽宁抚顺人，副教授.

描述方式. 考虑到大部分恶意代码都经过了加压和加壳处理，以达到隐蔽目的，如果对加壳文件进行脱壳处理，会出现可执行(portable execute，PE) 文件头被破坏的现象. PE 文件头中包含着调用的应用程序编程接口(application programming interface，API) 函数名，所以如果 PE 文件头被破坏，则不能再采用资源描述符作为特征. 机器码描述方式通常采用 n-gram 滑动窗口提取出固定长度的机器码串，因为机器码没有实际意义，所以在特征过滤时没有非常好的策略. 本文采用的方法是提取样本集中的字符串作为特征集.

字符串提取方法会提取出很多无意义的字符串，例如“aaaaaaaa”、“bhuhttt”等. 这种无意义字符串的数量是巨大的，直接影响着分类器的学习速度，也易使数据库过于庞大，所以需要选择一种恰当的方法把这些无意义的字符串过滤掉.

本文选用了计算机字典进行过滤，字典内容包含 7 336 个条目: 计算机常用英语单词，包括常用计算机英语缩写; 特别的计算机单词，如“QQ”、“msn”; 加入“. dll”等字符串，以避免动态链接库名称字符串遗漏.

2　改进贝叶斯算法

为了提高朴素贝叶斯算法的学习速度，本文提出了一种改进贝叶斯算法(half-increment naive Bayes，HNB) ，采用特征集增量学习方法，分类器根据新增特征项自动增量更新分类规则，从而提高了学习速度，实现了对新恶意代码的自动识别.

朴素贝叶斯的定义如下.

定义 1(朴素贝叶斯分类)　假设样本程序包含了一个特征集 F，定义 C 为 M(恶意代码) 类或 B(正常程序) 类中的任意一类，通过式(1) 可预测出 F 属于某类的概率 $P(C|F)$.

$$P(C|F) = \frac{P(F|C) \times P(C)}{P(F)} \tag{1}$$

根据定义 1 可知，分类器的准确度主要取决于 $P(F|C)$，$P(F|C)$ 是从已有的样本中分析得到的概率. $P(F|C)$ 来自于大量的样本，如果样本空间足够大，由此得出的概率 $P(C|F)$ 也越准确. 但当样本空间非常大时，计算 $P(F|C)$ 非常耗时，尤其是当增加新的样本后，需要重新计算 $P(F|C)$，影响了分类器的学习速度. 下面给出一种可以提高学习速度的改进贝叶斯算法.

2.1　改进贝叶斯算法

引理 1　从样本集中前 n 个样本中获得的 $P(F^{(n)}|C)$ 与第 $n+1$ 样本的 $P(X^{(n+1)}|C)$ 叠加可以得到前 $n+1$ 个样本空间的 $P(F^{(n+1)}|C)$，即 $P(F_i^{(n+1)}|C_j) = \dfrac{P(F_i^{(n)}|C_j) \times n_a}{n_a+1} + P(X_i^{(n+1)}|C_j)$，其中 $P(F^{(n)}|C)$ 是利用前 n 个样本得到的事件 C 发生条件下的事件 F 发生的条件概率，$P(X^{(n+1)}|C)$ 是利用第 $n+1$ 个样本得到的事件 C 发生条件下的事件 F 发生的条件概率，$P(F^{(n+1)}|C)$ 是利用前 $n+1$ 个样本得到的事件 C 发生条件下的事件 F 发生的条件概率，n_a 为满足 $C=C_j$ 条件的样本数量.

证明: 假设前 n 个样本形成的特征集中包含有 a 个特征，即 $F^{(n)} = \{F_1, F_2, \cdots, F_a\}$，则

$$P(F^{(n)}|C) = \prod_{i=1}^{a} P(F_i|C_j),\ j=1,2 \tag{2}$$

$$P(F_i|C_j) = \frac{P(F_iC_j)}{P(C_j)} = \frac{\text{count}(F=F_i \wedge C=C_j)}{\text{count}(C=C_j)} \tag{3}$$

这里，$\text{count}(F=F_i \wedge C=C_j)$ 是满足 $F=F_i$ 和 $C=C_j$ 条件的样本数量，$\text{count}(C=C_j)$ 是满足 $C=C_j$ 条件的样本数量.

如果在样本集中满足 $C=C_j$ 条件的样本数量为 n_a，则条件概率 $P(F^{(n)}|C)$ 为

$$P(F_i^{(n)}|C_j) = \frac{\text{count}((F=F_i^{(n)}) \wedge (C=C_j))}{n_a} \tag{4}$$

此时,假设属于 $C=C_j$ 的第 $(n+1)$ 个样本被加入,则会出现下述 3 种情况.

1) 对于 $F^{(n)}$ 中的特征集,且未在第$(n+1)$ 个样本出现的特征,则有

$$P(F_i^{(n+1)}\mid C_j)=\frac{\text{count}((F=F_i^{(n)})\wedge(C=C_j))}{n_a+1}=\frac{P(F_i^{(n)}\mid C_j)\times n_a}{n_a+1} \tag{5}$$

2) 对于 $F^{(n)}$ 中的特征集,且在第$(n+1)$ 个样本出现的特征,则有

$$P(F_i^{(n+1)}\mid C_j)=\frac{\text{count}((F=F_i^{(n)})\wedge(C=C_j))+1}{n_a+1}=$$

$$\frac{P(F_i^{(n)}\mid C_j)\times n_a}{n_a+1}+\frac{1}{n_a+1} \tag{6}$$

3) 在第$(n+1)$ 个样本中有 b 个特征没有在特征集 $F^{(n)}$ 中出现过,那么这些字符串将被作为特征加入到特征集中,也就是 $F^{(n+1)}=\{F_1,F_2,\cdots,F_a,F_{a+1},\cdots,F_{a+b}\}$. 对于这些新的特征,条件概率 $P(F_i^{(n)}\mid C_j)=0$,$a<i\leqslant a+b$,所以

$$P(F_i^{(n+1)}\mid C_j)=\frac{1}{n_a+1},\quad a<i\leqslant a+b \tag{7}$$

重写式(5) ~(7) 可得到

$$P(F_i^{(n+1)}\mid C_j)=\frac{P(F_i^{(n)}\mid C_j)\times n_a}{n_a+1}+P(X_i^{(n+1)}\mid C_j) \tag{8}$$

因此,从样本集中前 n 个样本中获得的 $P(F^{(n)}\mid C)$ 与第$(n+1)$ 个样本的 $P(X^{(n+1)}\mid C)$ 叠加可以得到前$(n+1)$ 个样本空间的 $P(F^{(n+1)}\mid C)$.

引理 2 针对同样的样本集和同样的字符串提取及过滤算法,朴素贝叶斯算法和改进贝叶斯算法能获得相同的特征集,即 $F_{\mathrm{N}}=F_{\mathrm{H}}$. 这里 F_{N} 是朴素贝叶斯算法获得的特征集,F_{H}是改进贝叶斯算法获得的特征集.

证明: 假设前 n 个样本形成的特征集中包含有 a 个特征,针对这些样本集和前面所述字符串提取及过滤算法,朴素贝叶斯算法获得的特征集为 $F_{\mathrm{N}}=\{F_1,F_2,\cdots,F_a\}$.

改进贝叶斯算法获得的特征集中的特征项是随着样本的增加而增加的. 假设在第 1 个样本中通过字符串提取方法提取出 S_1个字符串,经过特征过滤后得到 k_1 个特征项,$F_{\mathrm{H}}^{(1)}=\{F_1,F_2,\cdots,F_{k_1}\}$. 第 2 个样本中通过字符串提取方法提取出 S_2 个字符串,经过特征过滤后得到 k_2 个不同的特征项,$F_{\mathrm{H}}^{(2)}=\{F_1,F_2,\cdots,F_{k_1},F_{k_1+1},\cdots,F_{k_1+k_2}\}$. 当所有的样本被提取完成后,特征集中包含根据 n 个样本且按上面所述字符串提取和过滤算法后得到的特征项,$F_{\mathrm{H}}^{(n)}=\{F_1,F_2,\cdots,F_{k_1},F_{k_1+1},\cdots,F_{k_1+k_2},\cdots,F_a\}$. 所以

$$F_{\mathrm{N}}=F_{\mathrm{H}}. \tag{9}$$

根据引理 1 和 2,得到定理 1.

定理 1 改进贝叶斯分类器与朴素贝叶斯分类器具有相等的预测分类概率,即 $C_{\mathrm{N}}=C_{\mathrm{H}}$,其中 C_{N} 为利用朴素贝叶斯分类器得到的预测分类最大概率,C_{H} 为利用改进贝叶斯分类器得到的预测分类最大概率.

证明:

1) 朴素贝叶斯

假设前 n 个样本形成的特征集中包含有 a 个特征,$F_{\mathrm{N}}=\{F_1,F_2,\cdots,F_a\}$,且特征项是相互独立的,则

$$P(F_{\mathrm{N}}\mid C)=\prod_{i=1}^{a}P(F_i\mid C_j),\ j=1,2 \tag{10}$$

2) 改进贝叶斯算法

假设在第 1 个样本中通过字符串提取方法提取出 S_1 个字符串,经过特征过滤后得到 k_1 个特征项,$F_{\mathrm{H}}^{(1)}=\{F_1,F_2,\cdots,F_{k_1}\}$,那么

$$P(F_{\mathrm{H}}^{(1)}\mid C)=\prod_{i=1}^{k_1}P(F_i\mid C_j),\ j=1,2 \tag{11}$$

如果第 2 个样本中通过字符串提取方法提取出 S_2 个字符串，经过特征过滤后得到 k_2 个不同的特征项，$F_{\mathrm{H}}^{(2)}=\{F_1,F_2,\cdots,F_{k_1},F_{k_1+1},\cdots,F_{k_1+k_2}\}$，那么

$$P(F_{\mathrm{H}}^{(2)}\mid C)=P(F_{\mathrm{H}}^{(1)}\mid C)\times\prod_{i=k_1+1}^{k_1+k_2}P(F_i\mid C_j)=\prod_{i=1}^{k_1+k_2}P(F_i\mid C_j) \tag{12}$$

根据引理 2，对于所有的样本，$F_{\mathrm{H}}^{(n)}=\{F_1,F_2,\cdots,F_{k_1},F_{k_1+1},\cdots,F_{k_1+k_2},\cdots,F_a\}$，那么

$$P(F_{\mathrm{H}}^{(n)}\mid C)=\prod_{i=1}^{a}P(F_i\mid C_j),j=1,2 \tag{13}$$

所以 $P(F_{\mathrm{N}}\mid C)=P(F_{\mathrm{H}}\mid C)$，也即

$$C_{\mathrm{N}}=C_{\mathrm{H}} \tag{14}$$

2.2　复杂性分析

朴素贝叶斯和改进贝叶斯算法的耗时主要用于: 从样本程序中提取字符串，并用字典进行过滤，也就是数据预处理; 计算 $P(F\mid C)$.

2 种算法第 1 部分的时间是相同的，因为样本程序的数量相同，提取方法相同，过滤方法也相同，主要是第 2 部分的计算时间不同. 对于改进算法来说，假设有 n 个样本，其中在第 1 个样本中通过字符串提取方法提取出 S_1 个字符串，经过特征过滤后得到 k_1 个特征项，第 2 个样本有 S_2 个字符串，确定这些字符串是否出现在特征集中的耗时为: $T_{\mathrm{H}_2}=O(S_2\times k_1)$，如果其中有 k_2 个不同的特征项，则特征集将更新为包含 (k_1+k_2) 个特征项. 以此类推，对于 n 个样本利用改进贝叶斯算法的总耗时为

$$T_{\mathrm{H}}=O(S_2\times k_1+S_3\times(k_1+k_2)+\cdots+S_n\times(k_1+k_2+\cdots+k_r)) \tag{15}$$

而对于朴素贝叶斯算法，假设特征项为 K 个，n 个样本中提取出的字符串分别为 $S_1,S_2,\cdots,S_n$，利用朴素贝叶斯算法计算$P(F\mid C)$ 的耗时为

$$T_{\mathrm{N}}=O(K\times(S_1+S_2+\cdots+S_n)) \tag{16}$$

根据引理 2 可知 $F_{\mathrm{N}}=F_{\mathrm{H}}$，所以 $K=k_1+k_2+\cdots+k_r$.

根据式 (15)，有

$$T_{\mathrm{H}}=O(K\times(S_2+S_3+\cdots+S_n)-S_2\times(k_2+\cdots+k_r)-\cdots-S_{n-1}\times k_r) \tag{17}$$

如果 $k_2=k_3=\cdots=k_r=0$，或 $K=k_1$，则 $T_{\mathrm{H}}=T_{\mathrm{N}}$，2 种算法的耗时相同. 但恶意代码中通常包含木马、蠕虫、后门等多种类型的文件，一个样本中不可能包含全部的特征项，所以 $k_1<K$，$T_{\mathrm{H}}<T_{\mathrm{N}}$. 由此可以得到定理 2.

定理 2　改进贝叶斯算法的耗时比朴素贝叶斯算法的耗时短，即 $T_{\mathrm{H}}<T_{\mathrm{N}}$.

3　实验结果

为检验字符串特征提取方法的效果和改进贝叶斯模型的分类精度，进行了如下实验: 1) 比较朴素贝叶斯、复合贝叶斯和改进贝叶斯 3 种分类器的学习时间; 2) 比较 3 种分类器的分类精度.

3.1　样本设置

从反病毒公司获得 995 个恶意代码，其中 Agobot 类 448 个，Viking 类 527 个，熊猫烧香类 20 个. 正常文件 2 000 个，为取自反病毒公司无毒服务器的系统文件. 采用 5 重交叉学习法，即样本集分 5 个子集，每个子集包含 199 个恶意代码和 400 个正常文件. 每轮实验选取 4 个样本子集用于学习，剩下的 1 个作为测试. 共重复 5 轮实验，每轮将不同的样本子集作为测试集，取 5 次结果的平均值作为最后的测试结果.

3.2 实验设置

实验时先从样本集中提取字符串,再采用字典过滤算法对已获得的字符串进行过滤,并形成学习集,然后分别采用3种算法进行测试.

对于每种算法,分类器的生成方法如下.

1) 朴素贝叶斯算法

采用5重交叉算法,每轮运算由4个样本子集形成1个学习集,根据学习集形成1个分类器.

2) 复合贝叶斯算法

每轮运算由4个样本子集形成4个学习集,并构建4个子分类器,最后的测试结果是4个子分类器结果的平均值.

3) 改进贝叶斯算法

与朴素贝叶斯算法相同,每轮运算由4个样本子集形成1个学习集,根据学习集形成1个分类器. 但不同的是,特征集的生成随着样本的增加而增加,每次以1个样本作为增量进行叠加.

3.3 分类器性能比较

3.3.1 耗时比较

图1给出了3种分类器的耗时. 从图1中可以看出,朴素贝叶斯分类器学习过程耗时随着样本的增加显著增加,改进贝叶斯分类器和复合贝叶斯分类器比朴素贝叶斯分类器有明显的优势.

由于改进贝叶斯分类器自身的特点,使得其可以支持最小单位为1个程序的增量学习方式,这样就不会导致学习过程中由于出现意外终止而全部重新学习的现象发生. 无论是复合贝叶斯分类器,还是朴素贝叶斯分类器都存在这个问题,特别是朴素贝叶斯分类器必须全部重新学习.

图2给出了改进贝叶斯分类器构建特征集时,特征项数量随着样本的增加曲线. 从图中可以看出,曲线在某个点时是非常陡峭的,表明在新学习到某一类恶意代码时,特征项的数量急剧增加; 当学习了一些样本后,曲线比较平缓,表明在学习了同一类恶意代码后,特征项数量变化不大.

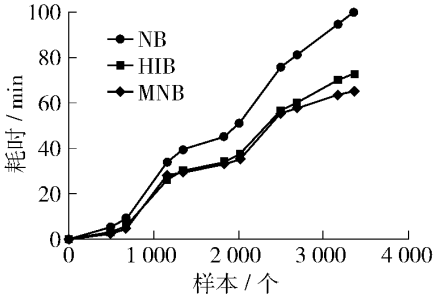

图1 3种算法的耗时

Fig. 1 Time-consuming curves of NB, MNB and HIB

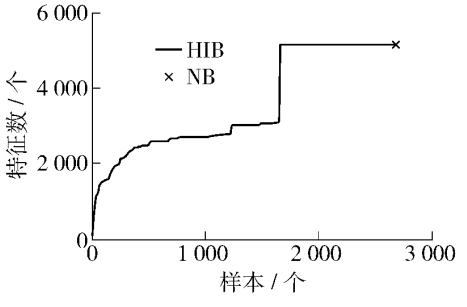

图2 改进贝叶斯算法特征项变化曲线

Fig. 2 Curve between number of feature element and number of sample

3.3.2 分类精度比较

对于分类预测结果的评价分为4种[10],即: 1) 将恶意代码分类为恶意代码,称为真正类(true positive, T_P); 2) 将正常文件分类为正常文件,称为真负类(true negative, T_N); 3) 将正常文件分类为恶意代码,称为假正类(flase positive, F_P); 4) 将恶意代码分类为正常文件,称为假负类(false negative, F_N). 评价分类器性能的指标有准确率、错报率和精度[9]. 其中:

$$\text{准确率} = \frac{T_P}{T_P + F_N},\ \text{错报率} = \frac{F_P}{T_N + F_P},\ \text{精度} = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}$$

3种分类器得到的结果采用接受者操作特性(receiver operating characteristic, ROC)曲线,曲线绘制如图3所示. 从图3可以看出,改进贝叶斯分类器的结果优于朴素贝叶斯分类器和复合贝叶斯分类器的结

果，分类精度达到了 96%，而朴素贝叶斯分类器和复合贝叶斯分类器的分类精度分别为 95% 和 93%. 从图中还可以看出，朴素贝叶斯分类器和改进贝叶斯分类器的 ROC 曲线比复合贝叶斯分类器的更陡，在错报率大于 80% 时，分类精度就已超过 90%.

表 1 给出了实验结果同经典文献结果的比较，其中，SVM 为支持向量机，J48 为一种决策树. Schultz 等[8] 给出了他们的样本集的访问地址，Henchiri 等[14] 使用了同样的样本集进行试验，我们试图得到同样的样本集，以便对试验数据进行比较，但遗憾的是网址已不可访问. 虽然这样直接比较结果意义不大，但仍然能得到同样的结论:

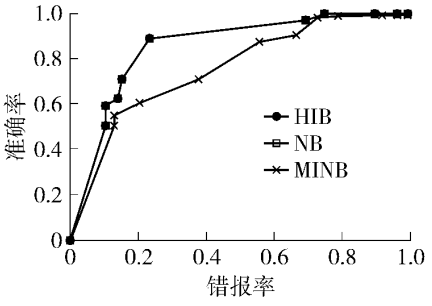

图 3　NB、MNB 和 HIB 的 ROC 曲线

Fig. 3　NB, MNB and HIB ROC

表 1　同经典文献的实验结果比较

Table 1　Experimental results using our method and traditional methods

方法	特征集	过滤方法	分类器	精度/%
本文方法	字符串	字典	HIB	96.00
Schultz[8]	字符串	频率	NB	97.11
Schultz[8]	*n*-gram	频率	MNB	96.88
Henchiri[14]	*n*-gram	频率	J48	93.65
Kolter[9]	*n*-gram	频率	NB	83.00

1) 在 Schultz 等[8] 的实验中，以字符串作为特征集得到的结果优于以机器码作为特征集得到的结果，所以字符串更宜于作为恶意软件识别的特征项.

2) Henchiri 等[14] 以 *n*-gram 机器码作为特征集得到的结果低于 Schultz 的结果，说明贝叶斯分类器更适合于恶意软件分类.

3) Kolter 等[9] 的实验虽然与上述 2 篇文献使用的样本集不一致，但仍然可以看出，尽管采用贝叶斯分类器，但由于以 *n*-gram 机器码作为特征集，因此效果很不理想.

4　结束语

本文给出了基于改进贝叶斯算法的未知恶意代码检测模型，为构建此模型，需提取恶意代码和正常程序的典型特征，并在特征集的基础上改进分类器的构造算法，以提高分类器的学习速度.

在特征提取上，对恶意代码文件和正常文件进行静态分析，提取字符串作为特征，这样比只使用头部信息(DLL 名等) 作为特征有了更高的可靠性，比采用固定长度 *n*-gram 滑动窗口选取的机器码特征更能反映程序的行为特征.

在分析研究朴素贝叶斯算法、复合贝叶斯算法的基础上，提出了改进算法，经实际样本测试表明分类精度达到了 96%，其中对已知恶意代码的分类精度达到 99%，这个结果和文献 [8-9] 的效果接近，但学习速度快.

参考文献:

[1] MCGRAW G, MORRISETT G. Attacking malicious code: a report to the infosec research council [J]. IEEE Software, 2000, 17(5): 33-41.

[2] COHEN F. Computer viruses—theory and experiments [J]. Computers and Security, 1987, 6(1): 22-35.

[3] SPINELLIS D. Reliable identification of bounded-length viruses is NP complete [J]. IEEE Transactions on Information Theory, 2003, 49(1): 280-284.

[4] KEPHART J O, SORKIN G B, ARNOLD W C, et al. Biologically inspired defenses against computer viruses [C] // Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. San Fransisco: Publisher of Morgan Kaufmann, 1995: 985-996.

[5] TESAURO G J, KEPHART J O, SORKIN G B. Neural networks for computer virus recognition [J]. IEEE Expert, 1996, 11 (4): 5-6.

[6] LEE W, STOLFO S J, MOK K W. A data mining framework for building intrusion detection models [C] // Bob Wener. Proceedings of the 1999 IEEE Symposium on Security and Privacy. North Kansas: Technical Communication Services, 1999: 120-132.

[7] LEE W, STOLFO S J, CHAN P K. Learning patterns from UNIX processes execution traces for intrusion detection [R] // AAAI Workshop on AI Approaches to Fraud Detection and Risk Management. California: AAAI Press, 1997: 50-56.

[8] SCHULTZ M G, ESKIN E, ZADOK E, et al. Data mining methods for detection of new malicious executables [C] // Frances M Titsworth. Proceedings of 2001 IEEE Symposium on Security and Privacy. Florida: The Printing House, 2001: 38-49.

[9] KOLTER J Z, MALOOF M A. Learning to detect and classify malicious executables in the wild [J]. Journal of Machine Learning Research, 2006(7): 2721-2744.

[10] ZHANG Bo-yun, YIN Jian-ping, HAO Jing-bo. Intelligent detection computer viruses based on multiple classifiers [C] // Ubiquitous Intelligence and Computing. Heidelberg: Springer Berlin, 2007: 1181-1190.

[11] ZHANG Bo-yun, YIN Jian-ping, HAO Jing-bo. Using fuzzy pattern recognition to detect unknown malicious executables code [C] // Fuzzy Systems and Knowledge Discovery. Heidelberg: Springer Berlin, 2005: 629-634.

[12] 张波云, 殷建平, 张鼎兴, 等. 基于 *K*-最近邻算法的未知病素检测 [J]. 计算机工程与应用, 2005(6): 7-10.
ZHANG Bo-yun, YIN Jian-ping, ZHANG Ding-xing, et al. Unknown computer virus detection based on *K*-nearest neighbor algorithm [J]. Computer Engineering and Applications, 2005(6): 7-10. (in Chinese)

[13] 宫秀军, 刘少辉, 史忠植. 一种增量贝叶斯分类模型 [J]. 计算机学报, 2002, 25(6): 645-650.
GONG Xiu-jun, LIU Shao-hui, SHI Zhong-zhi. An incremental Bayes classification model [J]. Chinese Journal of Computers, 2002, 25(6): 645-650. (in Chinese)

[14] HENCHIRI O, JAPKOWICZ N. A feature selection and evaluation scheme for computer virus detecting [C] // Proceedings of the 6th International Conference on Data Mining (ICDM'06). Hong kong: ACM Press, 2006: 1-6.

Unknown Malicious Detection Based on Improved Bayes Algorithm

LAI Ying-xu, YANG Zhen

(College of Computer Science, Beijing University of Technology, Beijing 100124, China)

Abstract: The detection of unknown malicious executables is beyond the capability of many existing detection approaches. Machine learning or data mining method can identify new or unknown malicious executables with some degree of success. Bayes or improved Bayes algorithm has the detection capability of unknown malicious excutables; however, it takes more time to study. A new improved algorithm is proposed in this paper. The new classifier based on strings achieve has high detection rates and can be expected to perform as well in real-world conditions.

Key words: classifier; data mining; bayes methods

(责任编辑 吕小红)