文献引用格式：蒋永康，吴越，邹福泰 . 基于图像矢量的恶意代码分类模型 [J]. 通信技术，2018，51（12）：2953-2959.

JIANG Yong-kang,WU Yue,ZOU Fu-tai.An Image-based Malware Classification Model[J]. Communications Technology,2018,51(12):2953-2959.

doi:10.3969/j.issn.1002-0802.2018.12.028

基于图像矢量的恶意代码分类模型 *

蒋永康，吴　越，邹福泰

（上海交通大学 网络空间安全学院，上海 200240）

摘　要：近年来，恶意软件呈爆发性增长，而主流的恶意代码分析大都依赖复杂的特征提取和融合技术。因此，研究了一种新颖的基于图像矢量的恶意代码分类模型，将恶意代码的汇编指令映射为图像矢量，实现恶意代码分类问题向图像分类问题的转化。同时，借鉴语句分类问题的思想，构建结构简单、理论可解释性强的深度学习网络。模型在微软 BIG2015 数据集上取得了 97.87% 的交叉验证准确率，虽略低于冠军模型的结果，但实现了显著的性能提升。

关键词：恶意代码；图像矢量；分类；深度学习

中图分类号：TP309.5　　**文献标志码：**A　　**文章编号：**1002-0802(2018)-12-2953-07

An Image-based Malware Classification Model

JIANG Yong-kang, WU Yue, ZOU Fu-tai

(School of Cyberspace Security, Shanghai Jiao Tong University, Shanghai 200240, China)

Abstract: In recent years, malware has experienced explosive growth, while mainstream malicious code analysis relies heavily on complex feature extraction and fusion techniques. Therefore, a novel image vector based malicious code classification model is studied. It maps the assembly instructions of malicious code into image vectors, and realizes the conversion of malicious code classification problems to image classification problems. At the same time, drawing on the idea of sentence classification problem, construct a deep learning network with simple structure and strong theoretical explanation. The model achieved a cross-validation accuracy of 97.87% on the Microsoft malware BIG2015 dataset, which was slightly lower than the results of the champion model, but achieved significant performance improvements.

Key words: malware; image-vector; classification; deep learning

0　引　言

如今，随着多态、变异和反分析技术的发展，恶意软件在数量和质量上都呈现爆发性增长。文献 [1] 提到，海量数据的恶意性检测任务正逐渐成为当下反病毒引擎的主要挑战之一。文献 [2] 表明，尽管攻击者开发移动平台恶意软件的兴趣与日俱增，但 windows 依然是被攻击的主要平台。2017 年第一季度检测到的 4.8 千万个恶意样本中，来自 windows 平台的占到 77.2%。这表明传统的基于特征匹配和行为分析的恶意代码分析技术已经很难满足海量样本的恶意检测任务。如何实现大规模数据的

*　**收稿日期：**2018-08-18；**修回日期：**2018-11-12　　Received date:2018-08-18;Revised date:2018-11-12

基金项目：国家重点研发计划（No.2017YFB0802300）

Foundation Item: National Key Research and Development Program of China(No.2017YFB0802300)

万方数据

高效分析，已经成为恶意代码研究的主要课题之一。

因此，本文主要研究基于图像矢量的恶意代码分类模型在大规模 PE（Portable Executable）格式恶意代码分类任务中的应用和效果。第 1 章介绍恶意代码的矢量化技术；第 2 章介绍构建的深度学习模型；第 3 章给出模型在微软数据集上的实验结果；第 4 章分析模型的意义，给出进一步的研究方向。

1 恶意代码矢量化

恶意代码矢量化是一种将恶意代码样本映射为图像矢量的编码技术。恶意代码矢量化的最终目标是用一张全局唯一的图像矢量来表征恶意代码，从而将恶意代码的检测问题转化为图像的分类问题。恶意代码矢量化的核心在于编码源、编码长度和编码量的选择。

1.1 Nataraj 矢量化

2011 年，Nataraj[3] 提出了基于二进制文件的恶意代码矢量化方法。Nataraj 矢量化选择恶意代码二进制文件作为编码源，将 8-bit 二进制映射为 8-bit 整型数值。通常来说，编码量为整个 PE 文件或者 PE 文件中包含可执行代码的 .text 节。图 1 为 Nataraj 矢量化的具体步骤。

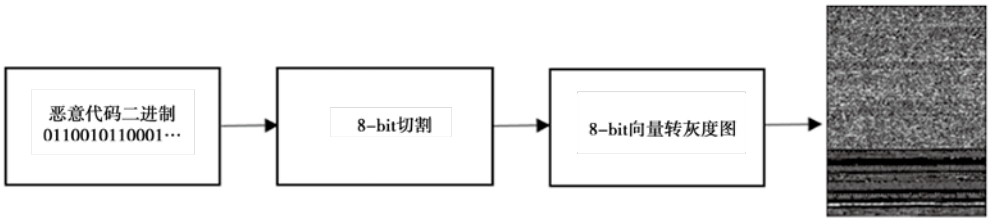

图 1 Nataraj 矢量化编码方法

Nataraj 矢量化是恶意代码分析技术的一种新思路，打开了基于图像矢量的恶意代码分析的大门。文献 [4-6] 中构建的恶意代码分析模型都将 Nataraj 矢量化作为重要的输入特征。然而，加壳和混淆技术能使恶意代码的二进制文件发生显著变化，直接导致输出的图像矢量产生巨大差异，使得 Nataraj 矢量化面对加壳和混淆技术时鲁棒性很差。同时，文献 [4] 也表明，在 Nataraj 矢量化下，不同恶意代码家族的图像矢量也有可能十分相似。因此，Nataraj 矢量化对于攻击者来说十分脆弱。

1.2 Andrew 矢量化

2015 年的黑帽大会上，Andrew[7] 提出了另一种基于反汇编文件的恶意代码矢量化思路。Andrew 矢量化选取反汇编十六进制机器码作为编码源，将 4-bit 的十六进制机器码映射成 4×8-bit 的整数值，再进行填充。如图 2 所示，Andrew 矢量化具有很好的视觉可解释性，图像矢量的每一行对应一条机器码。

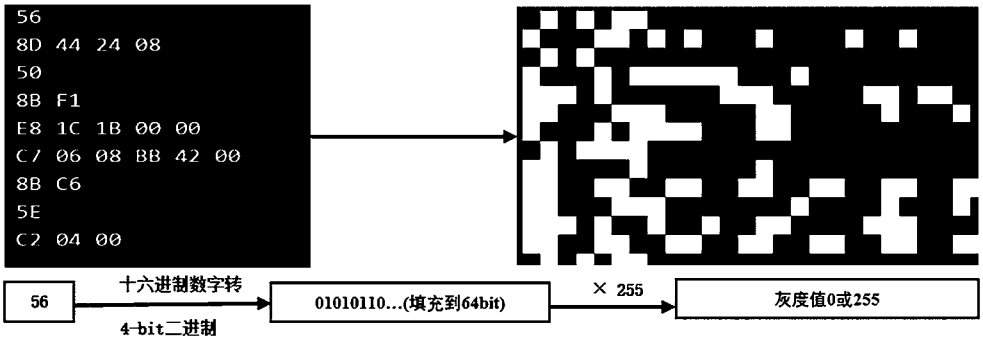

图 2 Andrew 矢量化编码方法

Andrew 矢量化不仅编码了恶意代码的机器码信息，还通过填充保留了汇编指令的空间信息。因此，Andrew 矢量化对于加壳和混淆技术具有较好的鲁棒性。遗憾的是，Andrew 并没有给出详细的分析和具体的深度学习模型。因此，本文将详细阐述 Andrew 矢量化中编码长度、编码量选择问题，并给出具体的深度学习模型。

1.3 64-bit 矢量填充

Andrew 矢量化中的一个关键问题在于，为什么选择 64-bit 矢量填充。

一方面，如图 3 所示的 Intel 64 和 IA-32 架构指令编码格式 [8] 规定指令的最大长度为 15 Bytes。更一般地，指令长度不会超过 11 Bytes。

万方数据

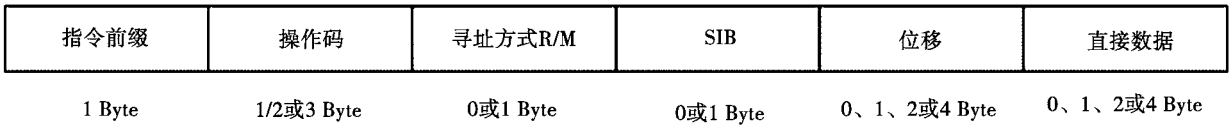

图 3　Intel 64 和 IA-32 架构指令格式

另一方面，通过对微软恶意代码数据集（BIG2015）[9] 的研究发现，99% 的恶意样本的指令长度都不超过 64 bit，结果如图 4所示。

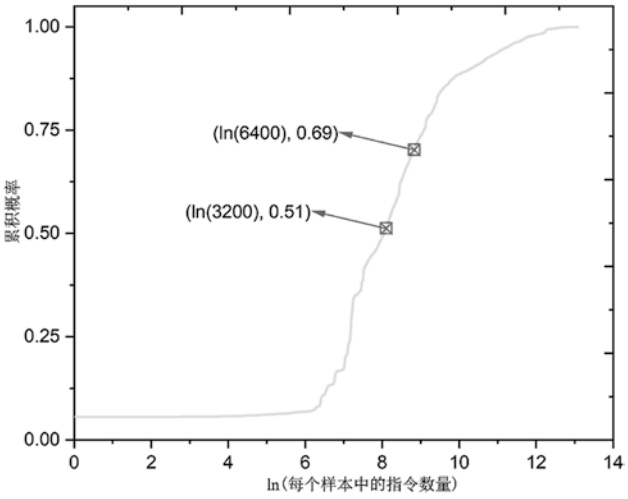

图 4　BIG2015 指令长度累积概率分布

因此，Andrew 矢量化选择 64-bit 矢量填充，以最大程度地保留恶意代码的指令信息。

1.4　编码量的选择

为了加速矢量化过程和深度学习网络的训练时间，需要在保证模型准确率的前提下，选择一个合适的、较小的编码量。

研究 BIG2015 时发现，不同恶意代码反汇编文件包含的指令数量差异很大，如图 5 所示，51% 的样本包含的指令少于 3 200 条。结合深度学习中图片大小的选择经验，3 200 是一个合理的编码指令数量。

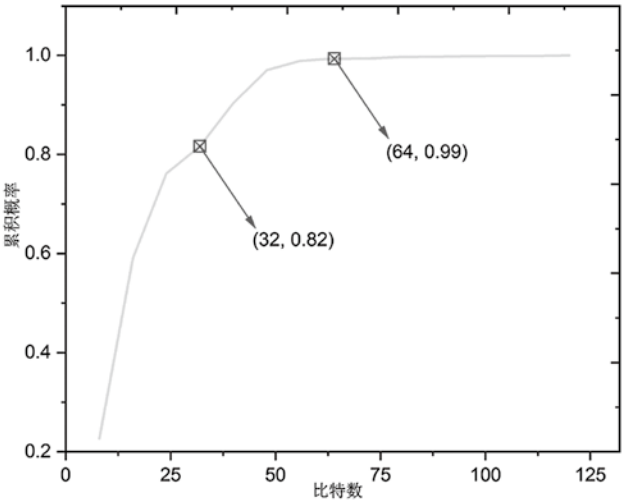

图 5　BIG2015 指令数量累积概率分布

2　深度学习模型

2014 年，Kim[10] 提出了一个适用于语句分类的单层卷积神经网络（CNN）架构模型。受 Kim 研究的启发，本文提出了如图 6 所示的基于图像矢量的恶意代码分类模型架构。

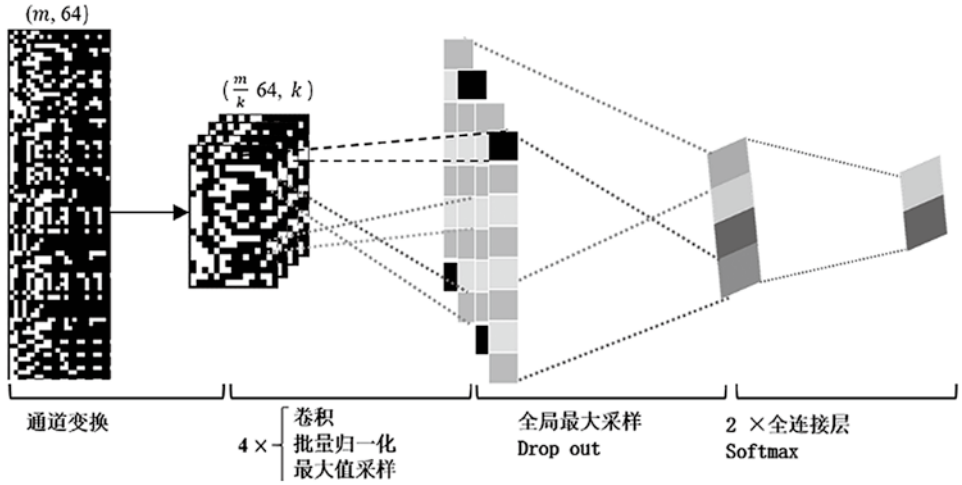

图 6　基于图像矢量的恶意代码分类模型架构

万方数据

模型输入矢量化的恶意代码图像，大小为 $(m,64)$，其中 m 代表编码的指令数量。通道变换模块输出大小为 $\left(\frac{m}{k},64,k\right)$ 的图像矢量，其中 k 代表通道数，是待确定的超参数。下面将以 k=1 为例，给出模型的详细描述。

首先，记 $S_j \in R^{64}$ 为 64 维图像矢量，对应 m 中的第 j 条指令，则恶意样本 X_i 可以表示为：

$$X_i=[S_1,S_2,\cdots,S_m] \quad (1)$$

记每个卷积层的过滤器为 $fr \in R^{hq}$，其中 h、q 表示过滤器的尺寸，每次卷积窗口移动都将 h 条指令的 q/64 矢量映射成新的特征矢量。例如，记 $c_{i,t}$ 表示由指令 $X_{i:i+h-1}(t{:}t+q-1)$ 经过滤器移动到位置 t 得到的新特征矢量，则：

$$c_{i,t}=w \cdot X_{i:i+h-1}(t{:}t+q-1)+b \quad (2)$$

其中 $b \in R$ 为偏置项参数。

因此，当一个行卷积操作完成时，一个新的特征矢量便产生了：

$$c_i=[c_{i,1},c_{i,2},\cdots,c_{i,64-q+1}] \quad (3)$$

当一层卷积的所有操作完成时，便得到了一个新的图像特征矢量：

$$c=[c_1,c_2,\cdots,c_{m-h+1}] \quad (4)$$

对 c 进行批量归一化（Batch Normalization）。批量归一化允许深度学习模型使用较大的学习率进行训练，能一定程度上抗过拟合，对于加速模型的训练具有显著的意义[11]。mini-batch 上的批量归一化算法的详细过程如下：

输入：每个 mini-batch 上的 c 值 $\varphi=\{c_{1\cdots n}\}$

1：需要学习的参数 γ，β

输出：$e_i=BN_{\beta,\gamma}(c_i)$

2：计算 mini-batch 均值 $\mu \leftarrow \frac{1}{n}\sum_{i=1}^{n} c_i$

3：计算 mini-batch 方差 $\sigma_\varphi^2 \leftarrow \sum_{i=1}^{n}\left(c_i-\mu_\varphi\right)^2$

4：标准化 $c_i^{'} \leftarrow \frac{c_i-\mu_\varphi}{\sqrt{\sigma^2+\varepsilon}}$

5：缩放与位移 $e_i \leftarrow \gamma c_i^{'}+\beta \equiv BN_{\beta,\gamma}(c_i)$

批量归一化后，对得到的矢量进行激活和最大值采样，得到新的特征矢量：

$$\boldsymbol{e}=maxpool(f[e_1,e_2,\cdots,e_{m-h+1}]) \quad (5)$$

其中 f 为非线性函数。

至此，已经描述了模型中单个 CNN 模块的特征提取过程。模型堆叠四个 CNN 模块进行抽象特征提取，因此全局最大采样（Global Max Pooling）层的输入为：

$$e=[e_1,e_2,\cdots,e_g] \quad (6)$$

其中 g 由具体的 CNN 模块参数决定。为了进一步降低模型输出的特征向量维度，并同时保留重要特征，对整个 CNN 模块输出进行全局最大采样，输出恶意代码的抽象特征矢量 $\hat{e}$ =max{e}。

最终，抽象特征矢量 $\hat{e}$ 经过两个全连接层和一个 softmax 层输出结果：

$$y=[y_1,y_2,\cdots,y_n] \quad (7)$$

其中 y_i 表示恶意样本属于家族 i 的概率，n 表示恶意样本的家族数量。

基于图像矢量的恶意代码分类模型利用上述方法对恶意代码的指令矢量进行层层变换，提取出高纬度抽象特征，从而实现恶意代码的分类。

3 实验结果与分析

3.1 数据集

BIG2015 数据集包含 9 个恶意家族的 21 741 个样本，其中 10 868 个样本为带标签的训练集，其他为不带标签的测试集。训练集中，每一个样本包含一个 20 字符的哈希 ID 和一个整数值得家族标签，分别为 Ramnit(F1)、Lollipop(F2)、Kelihos ver3(F3)、Vundo(F4)、Simda(F5)、Tracur(F6)、Kelihos、ver1(F7)、Obfuscator.ACY(F8) 和 Gatak(F9)。每个恶意样本包含两个文件，分别为十六进制表示的、去除 PE 头的二进制文件和反汇编工具 IDA 生成的包含恶意样本机器码、汇编指令等的元数据文件。

因为 BIG2015 中只有训练集带有标签，所以选取训练集中的恶意样本作为模型验证的基准，其分布如图 7 所示。

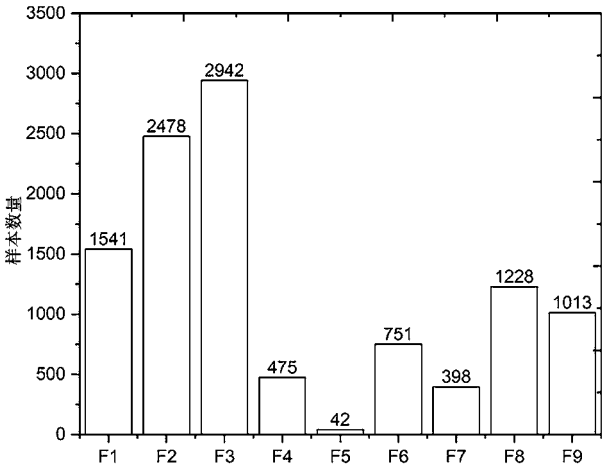

图 7 BIG2015 训练集样本家族信息分布

万方数据

3.2　模型实现

实验使用的平台信息如表 1 所示，具体的模型参数见表 2。

表 1　实验平台信息

平台	内容
硬件	GeForce GTX 1080 @ 8GB
	E5-2360 v3 @ 2.4GHz
	32 GB memory
软件	Centos 7.4
	Python3.6.5
	Tensorflow-gpu 1.8.0
	Keras 2.2.0
	Scikit-learn 0.19.1
	Numpy 1.14.3

表 2　基于图像矢量的恶意代码分类模型参数

模块	层名	参数
Input	input	(6400,64,1)
4 × CNN_Block	conv2d	(32,64,96,128)@8 × 2
	batch_normal	*m*=0.99,*ep*=0.001
	activation	relu
	max_pool	8 × 2,stride 2 × 1
Global_Pooling	global_mp	–
	drop_out	0.5
2 × Fully_Con	dense	256
	batch_normal	*m*=0.99,*ep*=0.001
	activation	relu
Softmax	dense	9

3.3　评估标准

模型使用交叉熵损失函数，定义如下：

$$loss=\frac{1}{M}\sum_{i}^{M}\sum_{j}^{N}Y_{ij}\log y_{ij} \tag{8}$$

其中 M 表示 mini-batch 的样本数量，N 表示恶意家族的数量。Y 为样本标签值，如果样本 i 在家族 j 中，则 Y_{ij}=1；反之，Y_{ij}=0。y 为模型的预测输出，y_{ij} 代表样本 i 在家族 j 中的概率。

同时，模型使用准确率（accuracy）、精确率（precision）、召回率（recall）和 f1_socre 来进行模型性能的评估，其定义如下。

记 S 为数据集中的样本数量，i 表示 S 中的第 i 个样本，y 表示预测值，Y 表示真是值，$l(x)$ 为指示函数，则：

$$accuracy=\frac{1}{|S|}\sum_{i=0}^{|S|-1}l(Y_i=y_i) \tag{9}$$

其次，定义：

$$P(A,B)\equiv\frac{|A\cap B|}{|A|},A\neq\varnothing \tag{10}$$

$$R(A,B)\equiv\frac{|A\cap B|}{|B|},B\neq\varnothing \tag{11}$$

记 η 为 S 的子集，则：

$$precision=\frac{1}{|S|}\sum_{\eta\in S}P(y_\eta,Y_\eta) \tag{12}$$

$$recall=\frac{1}{|S|}\sum_{\eta\in S}R(y_\eta,Y_\eta) \tag{13}$$

$$f1\_score=\frac{2\cdot precision\cdot recall}{precision+recall} \tag{14}$$

综上，*accuracy* 反映模型分类正确的样本占总样本的比例；*precision* 体现的是模型不将一个负样本标记为正原本的能力，*recall* 反映的是模型找到所有正样本的能力，*f1_score* 是两者的加权体现。

3.4　结果与分析

实验中，使用 10-fold 交叉验证方法对模型进行评估，结果如表 3 所示。结果表明，基于 Andrew 图像矢量的恶意代码分类模型在 BIG2015 训练集上，能实现 97.87% 的准确率和 0.094 的损失，其他详细性能评价指标详见表 3。平均来看，模型的训练时间为 1.7 h，训练好的模型检测 1 024 个样本的时间为 5.11 s。

表 3　BIG2015 10-fold 交叉验证结果（*k*=1，*m*=3 200）

序号	准确率 /（%）	精确率 /（%）	召回率 /（%）	*f1_score*/（%）	损失率	训练时间 /h	预测时间 /s
0	97.92	98.13	97.92	97.95	0.076	1.81	5.67
1	97.53	97.88	97.54	97.60	0.109	1.82	4.55
2	97.82	98.01	97.82	97.86	0.099	1.82	4.67
3	98.11	98.28	98.11	98.13	0.089	1.82	4.75
4	98.39	98.54	98.39	98.42	0.087	1.64	5.01
5	97.92	98.08	97.92	97.95	0.092	1.82	5.01
6	97.82	97.93	97.82	97.83	0.105	1.35	5.20
7	97.82	98.02	97.82	97.83	0.097	1.65	5.46
8	98.11	98.44	98.11	98.16	0.082	1.64	5.46
9	97.25	97.53	97.25	97.27	0.101	1.64	5.35
平均值	97.87	98.08	97.87	97.90	0.094	1.70	5.11

万方数据

与相关研究对比来看，如表 4 所示（预测时间为 1 024 个样本的检测时间），本文提出的基于图像矢量的恶意代码分类模型的准确率只比文献 [12] 和文献 [4] 中模型的能达到的准确率略低，主要原因在于后两者都是基于复杂、耗时的特征提取和融合技术。因此，本文提出模型的预处理时间、训练时间和预测时间相较于文献 [12] 和文献 [4] 都是成倍减少的。同时，相较于文献 [13-14] 的方法，本文的模型在准确率上也有明显优势。

表 4 相关研究工作比较

方法	准确率 /（%）	预处理时间 /h	训练时间 /h	预测时间 /s
(2015)BIG2015 Winner Solution[12]	99.83	72.00	1.0	13 649
(2016)Novel Features Extraction[4]	99.77	21.86	—	4 096
(2016)Random Forest[13]	95.62	—	—	—
(2017)Lempel-Ziv Jaccard Distance with kNN[14]	97.10	1.35	—	—
基于图像矢量的恶意代码分类模型	97.87	0.23	1.70	5.11

4 结 语

本文主要研究了基于图像矢量的恶意代码分类模型在大规模 PE（Portable Executable）格式恶意代码分类任务中的应用和效果，详细阐述了 Andrew 矢量化技术的相关细节，设计和训练了基于 Andrew 矢量化的恶意代码分类的深度学习模型。模型在 BIG2015 数据集上的交叉验证结果表明，虽然模型的准确率比冠军模型（基于复杂特征提取和融合技术）的准确率略低，但是实现了显著的性能提升。同时，实验结果也表明，与其他相关研究相比，本文的模型在准确率上优势明显。

综上，基于图像矢量的恶意代码分类模型成功将恶意代码的分类问题转化为图片的分类问题，且模型具有较好的理论鲁棒性，对于大规模恶意代码的分类任务具有实际意义。然而，Andrew 矢量化使用了大量的无效填充，使得图片矢量的编码效率很低。因此，如何提高恶意代码图片矢量化的编码效率是今后进一步的研究方向。

参考文献：

[1] MICROFT.SAM Cybersecurity Engagemen Kit[EB/OL].(2017-08-10)[2018-08-05],https://assets.microsoft.com/en-nz/cybersecurity-sam-engagement-kit.pdf.

[2] AV-TEST.Security_Report_2016-2017[EB/OL].(2017-08-17)[2018-08-05],https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2016-2017.pdf.

[3] NATARAJ L,KARTHIKEYAN S,JACOB G,et al.Malware Images:Visualization and Automatic Classification[C].Proceedings of the 8th International Symposium on Visualization for Cyber Security,2011.

[4] Ahmadi M,Ulyanov D,Semenov S,et al.Novel Feature Extraction,Selection and Fusion for Effective Malware Family Classification[C].Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy,2016.

[5] Kebede T M,Djaneye-Boundjou O,Narayanan B N,et al.Classification of Malware Programs Using Autoencoders Based Deep Learning Architecture and Its Application to the Microsoft Malware Classification Challenge (BIG 2015) Dataset[C].Proceedings of the Aerospace and Electronics Conference (NAECON),2017.

[6] Kim h J.Image-based Malware Classification Using Convolutional Neural Network[M].Advances in Computer Science and Ubiquitous Computing,2017:1352-1357.

[7] Andrew Davis M W.Deep Learning on Disassembly Data[EB/OL].(2015-08-06)[2018-08-05].https://www.blackhat.com/docs/us-15/materials/us-15-Davis-Deep-Learning-On-Disassembly.pdf.

[8] INTEL.Intel® 64 and ia-32 Architectures Software Developer's Manual,Volume 2:Instruction Set Reference[EB/OL].(2016-10-11)[2018-10-05],https://www.intel.com/content/dam/www/public/us/en/documents/manuals/64-ia-32-architectures-software-developer-instruction-set-reference-manual-325383.pdf.

[9] Microsoft.Microsoft Malware Classification Challenge(BIG2015)[EB/OL].(2015-04-08)[2018-08-05].https://www.kaggle.com/c/malware-classification/data.

[10] Kim Y.Convolutional Neural Networks for Sentence Classification[Z].2014.

万方数据

[11] Ioffe S,Szegedy C.Batch Normalization:Accelerating Deep Network Training by Reducing Internal Covariate Shift[Z].2015.

[12] Xiaozhou W J L,Xueer C.Say No to Overfitting [EB/OL].(2015-05-01)[2018-08-05].https://github.com/xiaozhouwang/kaggle_Microsoft_Malware/blob/master/Saynotooverfitting.pdf.

[13] Garcia F C C,Muga I,FELIX P.Random Forest for Malware Classification[Z].2016.

[14] Raff E,Nicholas C.An Alternative to NCD for Large Sequences,Lempel-Ziv Jaccard Distance[C].Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,2017.

作者简介：

蒋永康（1996—），男，硕士，主要研究方向为恶意代码分析；

吴　越（1968—），男，博士，研究员，主要研究方向为移动互联网安全；

邹福泰（1973—），男，博士，高级工程师，主要研究方向为恶意代码分析、僵尸网络检测。

万方数据