doi: 10.3969/j.issn.0490-6756.2019.04.012

基于图卷积网络的恶意代码聚类

刘 凯[1], 方 勇[2], 张 磊[2], 左 政[1], 刘 亮[2]
(1. 四川大学电子信息学院, 成都 610065; 2. 四川大学网络空间安全学院, 成都 610065)

摘 要: 许多新型恶意代码往往是攻击者在已有的恶意代码基础上修改而来,因此对恶意代码的家族同源性分析有助于研究恶意代码的演化趋势和溯源. 本文从恶意代码的 API 调用图入手,结合图卷积网络(GCN),设计了恶意代码的相似度计算和家族聚类模型. 首先,利用反汇编工具提取了恶意代码的 API 调用,并对 API 函数进行属性标注. 然后,根据 API 对恶意代码家族的贡献度,选取关键 API 函数并构建恶意代码 API 调用图. 使用 GCN 和卷积神经网络(CNN)作为恶意代码的相似度计算模型,以 API 调用图作为模型输入计算恶意代码之间的相似度. 最后,使用 DBSCAN 聚类算法对恶意代码进行家族聚类. 实验结果表明,本文提出的方法可以达到 87.3%的聚类准确率,能够有效地对恶意代码进行家族聚类.

关键词: 恶意代码; 图卷积网络; 聚类; API 调用图; 卷积神经网络
中图分类号: TP391.1 **文献标识码**: A **文章编号**: 0490-6756(2019)04-0654-07

Malware clustering based on graph convolutional networks

LIU Kai[1], *FANG Yong*[2], *ZHANG Lei*[2], *ZUO Zheng*[1], *LIU Liang*[2]
(1. College of Electronics and Information Engineering, Sichuan University, Chengdu 610065, China;
2. College of Cybersecurity, Sichuan University, Chengdu 610065, China)

Abstract: Many new types of malwares are often modified by attackers based on the existing malwares. Therefore, family homology analysis of malwares can help to study of evolutionary trend and traceability of malwares. In this paper, starting from API call graphs of malwares and combined with Graph Convolutional Networks (GCN), we proposed a similarity calculation and family clustering model for malwares. Firstly, we extract API call graphs of malwares with disassembly tools and the attribution of the API functions in the graphs are labeled. Then, we select key API functions by its contribution to the malware families and the API call graphs of malwares are constructed. We use GCN and Convolutional Neural Networks (CNN) as the model of the malware similarity calculation which the inputs are the API call graphs. Finally, we use DBSCAN algorithm to cluster malwares. The experimental results show that the proposed method can achieve 87.3% accuracy and can effectively cluster malware families.

Keywords: Malicious code; GCN; Clustering; API call graph; CNN

1 引 言

在信息安全领域,恶意代码的威胁是一个无法回避的话题. 在攻防对抗的过程中,信息系统的安全性逐渐提高,随之而来的是攻击者的攻击手段也更加隐蔽与复杂,新型的恶意代码层出不穷. 出于

收稿日期: 2019-01-09
基金项目: 国家重点研发计划基金(2017YFB0802904)
作者简介: 刘凯(1994—), 男, 硕士生, 研究方向为信息系统安全. E-mail: 13969107503@163.com
通讯作者: 刘亮. E-mail: liangzhai118@163.com

开发成本的考虑，攻击者往往在已有的恶意代码基础上进行修改来生成新的恶意代码而不是重新开发，这类恶意代码被认为来源于一个家族. 这类恶意代码往往存在许多共同点，比如执行流程，代码风格具有相似性，或者具有相同的代码片段和关键技术. 分析恶意代码的家族特征，有助于研究恶意代码的演化趋势，也有助于发现新型的恶意代码，并对其进行溯源.

当前恶意代码的自动化分析分为动态和静态两大类，比如 API 序列，纹理指纹[1]，API 调用图[2]，反汇编指令等. 相比与动态分析，静态分析不需要程序执行，分析速度快. 而其中 API 调用图是程序的典型的静态特征. 目前图相似性计算的主流方法是子图同构，但由于子图同构是 NP 问题，所以随着图规模的增大，计算复杂度会呈指数级增长. 为解决此问题，许多研究者将深度学习引入到了图领域.

深度学习在计算机视觉，自然语言处理，语音处理等领域已经取得了显著的效果，也逐步应用于信息安全领域，比如流量识别[3]和攻击检测[4]. 但诸如图片，语音等都是欧式数据. 与其相对应，在科学研究中，还存在许多非欧式空间的数据，比如分子结构，3D 图像，程序 API 调用图等. 传统的卷积神经网络，递归神经网络无法直接用于处理此类数据. 研究者将深度学习推广到了非欧氏空间，统称为几何深度学习. 具体到图的处理，图卷积神经网络(GCN)[5]已经应用于节点分类，连接预测和图相似性计算等领域.

1988 年，Goleberg 等人提出了计算机病毒家族发生树理论，这是最早的恶意代码同源性分析理论[6]. 之后恶意代码的同源性分析方法被陆续提出. 目前，恶意代码同源性分析方法主要分为动态分析和静态分析两大类别. 静态分析技术是在不运行恶意代码的前提下，根据恶意代码的反汇编代码，来判定恶意代码间的相似性，进而分析不同恶意代码间是否具有相同的家族特征. 动态分析技术是根据恶意代码在运行过程中的行为，操作等等来判定恶意代码间的相似性和家族同源性.

Karim 等人使用 N-Grams 和 N-Perm 作为特征，然后利用 TF-IDF 系数计算恶意代码间的相似度[7]. 2017 年 Searles 等人使用 SPGK 算法计算恶意代码函数调用图之间的相似度，他们根据每个函数的汇编指令类型分布，对函数的属性进行了向量化的标注[8]. Zhu 等人在 2018 年首先将 GCN 引入了恶意代码分析领域. 他们提取了恶意代码的调用图，并使用 GCN 对恶意代码进行家族分类[9].

本文在恶意代码 API 调用图的基础上，借助 GCN 计算 API 调用图之间的相似度作为恶意代码间的相似度，最后使用 DBSCAN 算法对恶意代码进行聚类. 本文的主要贡献如下：(1) 提取恶意代码的 API 调用图，并定义了图的节点即 API 的属性；(2) 提出了将图卷积网络应用于恶意代码分析领域，设计了恶意代码的相似度计算模型，并在此基础上对恶意代码进行家族聚类；(3) 在公开的数据集上对所提模型进行了验证，达到了 87.3% 的聚类准确率，优于 Opcode N-Gram 方法.

2　基于图卷积网络的聚类模型

本文所提出的分析框架如图 1 所示，包括反汇编，API 提取，关键 API 选取，相似度计算和家族聚类. 通过反汇编工具提取 API 后，对 API 进行属性标注，并根据 API 对恶意代码家族的贡献度选取关键的 API 构建规模相同的 API 调用图. 然后使用 GCN 和 CNN 来构建图的相似度计算模型，使用 GCN 对调用图进行图嵌入，将其映射为低维向量，由 CNN 计算并输出相似度. 最后使用 DBSCAN 算法对恶意代码进行家族聚类.

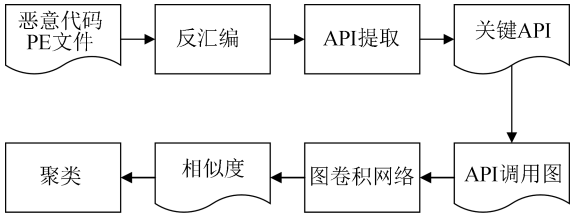

图 1　聚类模型框架

Fig. 1　The frame of the clustering model

2.1　API 调用图

从图的角度来看，API 调用图是有向图，但由于 GCN 无法处理有向图，所以将其简化为无向图. API 调用图包含两个元素，节点的集合和边的集合，可以用如下公式表示.

$$G=(V|E) \tag{1}$$

其中，G 为 API 调用图；V 为有限的节点集合；E 为边的集合，即为每个节点的连接关系. 如图 2 所示，从恶意代码的角度来看，API 调用图是恶意代码在执行过程中，采用图结构对 API 调用执行的连接关系进行表示的数据结构. 该方式能够更加全面地保留恶意代码各 API 执行情况以及 API 表

示的整体特征信息. 因此,在本文中,将恶意代码的二进制文件转换成 API 调用图,采用图结构表示恶意代码的特征信息.

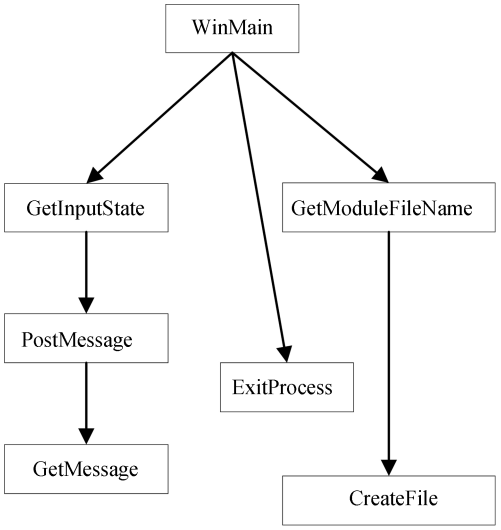

图 2 API 调用图
Fig. 2 API call graph

本文使用 IDAPython 脚本自动提取程序的 API 调用图. 通过 IDA 可以得到恶意代码的反汇编信息. 采用递归方式遍历恶意代码的函数调用,选取跳转指令和函数调用指令. 根据 IDA 的反编译信息,确定跳转目的地址是否为函数起始地址,即被调用函数. 被调用函数分为两类,内部函数和导入函数,若是内部函数则进入函数内部,递归查找并记录所有导入函数. 除此之外,恶意代码为了躲避静态检测,经常动态获取 API 的地址,间接调用 API. 通过对获取 API 相关函数的匹配,字符串匹配和交叉引用跟踪,对恶意代码中的动态调用的 API 进行了分析.

构建 API 调用图之后,恶意代码表示为 API 间的连接关系,每个节点代表一个 API. 之后,提取每个函数的特征作为节点的属性. 换而言之,本文要构建的 API 调用图是带节点特征的图. 每个节点所代表的 API 有以下几个属性。

(1) API 类型. 得到的导入函数未必为系统 API 函数,也可能是自定义函数或者第三方 API 函数,通过与预定义的 Windows 动态链接库(dll)列表比较,确定 API 函数是否为系统 API 函数.

(2) 是否为文档化 API. 在 Windows 系统中,存在未文档化的 API,比如 ntdll. dll 中的导出函数便为未文档化的 API.

(3) 函数字符集类别. Windows 中的部分 API 存在字符串类型的参数,而这类 API 一般有 Unicode 字符集与多字节字符集两种类型.

2.2 关键 API 选取

由于本文所使用的相似度计算模型只支持相同规模的图作为输入,即图的节点数量需要固定. 该相似度计算模型包含全连接层,全连接层的神经元个数是固定的,并且与输入向量的维数相对应,所以要对关键 API 进行选取,以便构建相同规模的 API 调用图. 关键 API 是指在恶意代码中比较能体现恶意代码特征的 API 函数,本文使用贡献度进行衡量.

首先,定义 API 函数在恶意代码家族中出现的加权频率为 $\mathrm{AF}(m_i, f_j)$,其计算方法如式(2)所示.

$$\mathrm{AF}(m_i,f_j)=\frac{\mathrm{NS}(m_i,f_j)}{\mathrm{NS}(f_j)}\times\frac{\mathrm{NA}(m_i,f_j)}{\mathrm{NA}(f_j)} \quad (2)$$

其中,$\mathrm{NS}(m_i, f_j)$表示恶意代码家族 f_j 中包含 API 函数 m_i 的样本数量,$\mathrm{NS}(f_j)$表示恶意代码家族 f_j 的样本总数,$\mathrm{NA}(m_i, f_j)$为恶意代码家族调用 API 函数的次数,$\mathrm{NA}(f_j)$为恶意代码家族调用所有 API 函数的总次数. 由此,则 API 函数对某个恶意代码家族的贡献度可根据式(3)计算.

$$C(m_i,f_j)=\left\{\frac{\mathrm{AF}(m_i,f_j)}{\sum_{f_j\in F}\mathrm{AF}(m_i,f_j)}\times\lg\frac{|F|}{FA}\right. \quad (3)$$

其中,$|F|$表示所有恶意代码家族集合 F 的数量,而 FA 为调用了 API 函数 m_i 的恶意代码家族的数量. 通过式(3),本文可以计算出每个 API 对每个恶意家族的贡献度,对每个恶意代码的 API 进行贡献度排序,并根据所有恶意代码 API 调用图的规模最小值,确定关键 API 的数量,进而重新构建 API 调用图.

2.3 相似度计算

本文使用 GCN 与 CNN 构建如图 3 所示的相似度计算模型,为 Siamese 双路神经网络[10],分为节点嵌入,图嵌入和相似度计算三个阶段. 节点嵌入层接受图结构作为输入,将图中的节点映射为低维向量. 在图嵌入层将使用节点嵌入表达图嵌入,最后使用 CNN 计算图之间的相似度. 由于图一般是变长的空间结构,不具有平移不变性,无法直接用神经网络处理,往往需要对图进行嵌入. 图嵌入即使用低维,稠密的向量表示图中的节点,进而表示图. 图结构中两个点的共享邻近点越多,则两个图的嵌入向量距离越小. 换而言之,嵌入向量的欧几里得距离是具有实际意义的,体现了节点间的关

联关系. 通过图嵌入可以提取图的空间特征，将复杂的图矩阵映射为低维向量，进而可以使用机器学习算法与神经网络处理图数据.

2.3.1　节点嵌入　在图处理中，常常使用邻接矩阵存储图结构数据，对于无向图，邻接矩阵是关于对角线对称的对称矩阵，式(4)为图 2 所对应的邻接矩阵. 定义图的邻接矩阵为 A，则 A_{ij} 表示第 i 个节点与第 j 个节点有无连接，1 表示两个节点有连接，0 表示两个节点无连接.

$$\begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \tag{4}$$

定义图的节点度为 D. 节点的度表示和节点相关联的边的条数. 图 2 的节点度表示如式(5)所示.

$$(3 \quad 2 \quad 2 \quad 1 \quad 1 \quad 2 \quad 1) \tag{5}$$

图卷积网络输入为的输入包括邻接矩阵和节点特征矩阵 X，每个神经网络层可以表示总结为一个非线性函数(6). $H^{(0)}=X$ 而 $H^{(l+1)}=z$，其中 z 是第 l 层的输出.

$$H^{(l+1)}=f(H^{(l)},A) \tag{6}$$

由式(6)可以将图卷积网络总结为多个图卷积层的堆叠，即分层传播模型，每一层的非线性函数可以定义为式(7).

$$f(H^{(l)},A)=\sigma(\hat{D}^{-1/2}\hat{A}\hat{D}^{1/2}H^{(l)}W^{(l)}) \tag{7}$$

令 $\tilde{A}=A+I_N$，则 $\hat{A}=D^{-1/2}\tilde{A}D^{-1/2}$ 为归一化矩阵，其任意行相加为 1. I_N 为单位矩阵，与 I_N 相加意味着加入自连接. 加入自连接的原因是，直接与邻接矩阵 A 相乘，仅考虑了所有相邻节点的特征向量，并没有考虑其自身. 对邻接矩阵归一化的原因是，若直接与 A 相乘会改变特征向量的原有占比. 归一化意味着取相邻节点特征的平均值，从而解决此问题.

本文采用双层 GCN 模型，双层 GCN 相当于考虑节点的二阶邻近节点. 将各层 GCN 相连后可以得到式(8)所示的节点嵌入计算式.

$$Z=\tilde{A}\mathrm{softmax}(\hat{A})\mathrm{ReLU}(\hat{A}XW_0W_1) \tag{8}$$

其中，W_0 和 W_1 为对应 GCN 层的权重系数矩阵，需要通过训练模型得到. 特征矩阵 X 是一个 $N\times M$ 的矩阵；N 是节点的数量；M 是节点特征的维数. 节点特征的定义在章节 2.1 进行了阐述，每个节点的特征为三维向量，其表示方法为 one-hot 编码.

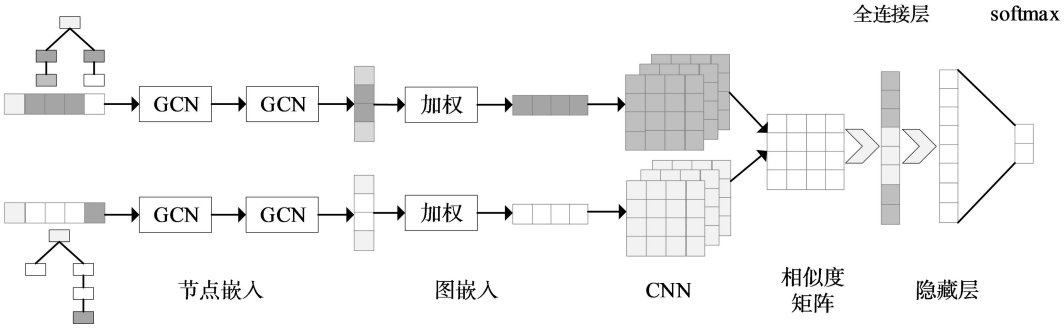

图 3　相似度计算模型

Fig. 3　The model of similarity computing

2.3.2　图嵌入　经过节点嵌入后，每个节点表示为一个向量，而图嵌入则是使用节点嵌入向量表示图. 本文借鉴 Bai 等人在 SimGNN 模型中用到的 attention 机制得到图嵌入[11]. Attention 机制源自于人在观察外界时的注意力问题，在观察某一场景时，人的注意力总是会根据自己的兴趣或者需求观察特定的局部信息，而不是按照顺序观察. Attention 机制在自然语言处理中取得了卓越的效果，可由词嵌入得到句子的嵌入表达[12]. 从某种程度上而言，这与由图结构中的节点嵌入到图嵌入具有相似性. 给定某一图结构，图中的不同节点对图的重要性是不同的，如果对节点嵌入简单求和会损失图的一部分特征，所以决定引入 attention 机制. Attention 机制本质上是计算节点嵌入的加权和，使用节点嵌入表达图嵌入. 其中与节点相关联的权重由其节点度来确定. 节点嵌入客观上表现了节

点间的关联关系，因此，使用以下 attention 机制根据节点间的关联关系来学习节点的权重.

节点嵌入阶段的输出为节点嵌入矩阵 $Z=R^{N\times D}$，其中，N 为节点的数量，也是节点嵌入矩阵的行数；D 为节点嵌入的维数. 矩阵 Z 的第 n 行，$z_n=R^D$ 是节点 n 的嵌入向量. 首先，根据嵌入矩阵计算图的全局上下文 $c\in R^D$，是每个节点嵌入的简单平均值的非线性变换：$c=\tanh\left(\frac{1}{N}\sum_{n=1}^{N}z_n\right)W_2$，其中，$W_2\in R^{D\times D}$是可学习的权重矩阵，训练模型可以得到. 通过学习权重矩阵，全局上下文 c 可以提供图的全局信息图的全局结构和特征信息，其的主要用途是为每个节点计算一个权值.

为了通过全局上下文获取节点 n 的权值，则要计算 c 和节点 n 的嵌入向量之间的内积 z_n^Tc. 节点 n 与全局上下文的内积运算可以理解为是得到节点 n 与其余节点的相关度. 由此，与全局上下文关联高的节点可以获得更高的权值. 然后使用 sigmoid 函数 $\sigma(x)=1/(1+\exp(-x))$确保节点的权值范围在(0,1)内. 最后，图嵌入 $h=R^D$ 是节点嵌入的加权和. 式(9)反映了 attention 的加权机制. Attention 机制在为不同节点赋予不同权值的同时，也避免了因遍历节点而带来的巨大计算开销.

$$h=\sum_{n=1}^{N}\sigma(z_n^T\tanh(\frac{1}{N}\sum_{m=1}^{N}z_m)W_2))z_n \tag{9}$$

2.3.3 相似度得分 经过图嵌入后，图结构表示为低维向量，作为 CNN[13]层的输入，CNN 包括两路参数相同的卷积层和池化层，池化层选择最大池化. 然后经过相似度矩阵计算相似度，相似度矩阵是该网络的一个可学习参数. 全连接层和隐藏层进行特征整合和非线性变换. Softmax 层输出两个输入之间的相似度. 相比与普通神经网络，Siamese 神经网络的输入为样本对而不是单个向本，其数学形式为一个元组 tuple(S_1, S_2, y). 其中标签 $y=0$ 表示输入样本 S_1和 S_2不属于同一类别，而 $y=1$ 表示 S_1和 S_2属于同一类别. 损失函数选用交叉熵代价函数[14]，用于计算真实类别与训练时预测类别的误差，通过反向传播优化模型的各个权重系数，从而使得模型的预测结果不断趋近于真实结果.

2.4 DBSCAN 聚类

DBSCAN (Density-Based Spatial Clustering of Applications with Noise)是一种密度聚类方法，由 Martin Esterd 等人在 1996 年提出[15]. 相比于 k 均值聚类[16]，密度聚类[17]方法无需指定聚类的目标簇数，适用于形状不规则且具有特定演化趋势的数据簇. 在恶意代码的家族聚类研究领域，经常使用密度聚类方法. 属于同一类别的点，他们之间的距离是比较近的. 所以，对于某类别中的某个点，在特定的范围内一定有其他同类别的点存在. 所以不断在特定范围内寻找同类别的点，可以得到最终的聚类结果.

下面给出 DBSCAN 中两个基本的概念，Eps 邻域和 minPts. minPts 为形成高密度区域所需要的最少点数.

Eps 邻域：对于样本集 D 中任意点，其 Eps 邻域包含样本集中该店距离不大于半径 Eps 的数据样例，即 $N_{Eps}=\{x_i\in D\mid Dist(x_i,x_j)<Eps\}$，其中，$Dist$ 是样本 x_i，x_j 的距离. DBSCAN 的聚类步骤如下.

1) 根据 KNN 分布和数学统计分析，确定合适的全局参数 minPts 和 Eps；

2) 根据得到的 minPts 和 Eps，计算核心点；

3) 选取未被标记的点，计算能够连通的核心点，边界点，噪声点；

4) 将密度可达的所有数据点包括核心点和边界点，都放到一起，形成该类簇；

5) 重复步骤 3)和步骤 4)，直到所有的点都被标记.

3 实验设计及结果分析

3.1 实验设计

本文实验数据集通过 VirusShare 网站获得，共 8 个家族，训练集和测试集每个家族各 50 个样本，训练集用于对相似度计算模型进行训练，与测试集之间不存在重复样本. 使用开源工具 AVClass 对恶意代码的家族名进行标注. 表 1 为实验样本家族名称.

表 1 恶意代码家族名称

Tab. 1 The families of malwares

编号	家族名称
1	Bancos
2	Delf
3	Hoybar
4	Palevo
5	Ramnit
6	Sytro
7	Virut
8	Zbot

对比实验选择了 Opcode N-Gram(4-Gram)，计算恶意代码 Opcode N-gram 的 TF-IDF 系数，并使用余弦相似度作为恶意代码间的相似度度量. 对比实验同样选择 DBSCAN 聚类算法进行家族聚类.

我们使用聚类准确率和调整兰德指数[18]作为聚类效果的评价指标.

调整兰德指数是对兰德指数的修改，首先给出兰德指数的定义：给定有 n 个元素的集合，结合中有 C_2^n 个集合对，a 表示同一类元素被分到了同一个簇的对数；b 表示不同类元素被分到不同簇的对数. 则兰德指数 RI 的定义为式(10)[18].

$$RI=\frac{a+b}{c_z^n} \tag{10}$$

作为聚类评价指标，期望当聚类结果为随机值时，指标应该接近零，但是兰德指数无法满足这种情况，所以调整兰德指数 ARI 被提出，定义如式(11)所示.

$$ARI=\frac{RI-E(RI)}{\max(RI)-E(RI)} \tag{11}$$

ARI 取值范围是$[-1,1]$，值越大则聚类结果越好，与真实的结果越吻合.

3.2　实验结果

3.2.1　最佳 Eps 值　为了使得实验的得到最好的兰德指数，设置 Eps 范围为 0.1～0.94，间隔为 0.02. 根据预先的实验，发现 minPts 参数对实验效果影响较小，选定 minPts 的值为 10. 实验的兰德指数分布如图 4，其中横坐标为 Eps 值，纵坐标为调整兰德指数.

对于本文提出的方法(以下简称 GCN)与 opcode N-Gram，最佳的 Eps 值分别为 0.84 和 0.8.

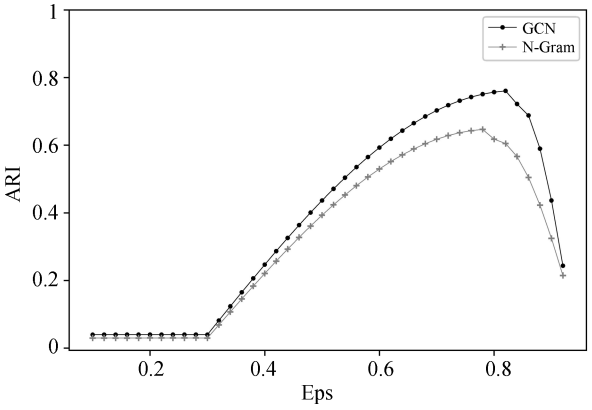

图 4　兰德指数分布

Fig. 4　The distribution of rand index

3.2.2　结果比较　表 2 为取最佳 Eps 值时，GCN 与 Opcode N-Gram 的聚类结果对比，评价指标为准确率和调整兰德指数.

表 2　聚类准确率与兰德指数

Tab. 2　Accuracy and rand index of clustering

方法	准确率(%)	兰德指数
Opcode N-Gram	78.5	0.618
GCN	87.3	0.722

由表 2 可知，本文所提出的方法聚类准确率为 87.3%，兰德指数为 0.722，均优于 Opcode N-Gram 方法，说明图卷积网络在计算恶意代码间的相似度方面具有较好的效果，适用于恶意代码的同源性分析.

4　结　论

本文提出了一种基于图卷积网络的恶意代码同源性分析模型，提取了恶意代码的 API 调用图，使用几何深度学习中的 GCN 计算恶意代码间的相似度并使用 DBSCAN 聚类算法对恶意代码进行了家族聚类，达到了 87.3%的平均聚类准确率，优于 Opcode N-Gram 方法.

本文所使用的相似度计算模型不支持有向图的处理和图的批处理，所以后面的工作中将会研究图卷积网络，构建更有效的检测模型，以进一步提升恶意代码的同源性分析效果.

参考文献：

[1]　韩晓光，曲武，姚宣霞，等. 基于纹理指纹的恶意代码变种检测方法研究 [J]. 通信学报，2017，35：125.

[2]　赵炳麟，孟曦，韩金，等. 基于图结构的恶意代码同源性分析 [J]. 通信学报，2017(s2)：86.

[3]　李勤，师维，孙界平，等. 基于卷积神经网络的网络流量识别技术研究 [J]. 四川大学学报：自然科学版，2017，54：959.

[4]　杨可心，桑永胜. 基于 BP 神经网络的 DDoS 攻击检测研究 [J]. 四川大学学报：自然科学版，2017，54：71.

[5]　Kipf T N，Welling M. Semi-supervised classification with graph convolutional networks [J/OL]. ComputSci，2016，/abs/1609.02907 [2016-09-09]. https://arxiv.org/abs/1609.02907.

[6]　Goldberg L A，Goldberg P W，Phillips C A，*et al*. Constructing computer virus phylogenies [J]. J Algorithms，1998，26：188.

[7]　Karim M E，Walenstein A，Lakhotia A，*et al*. Mal-

ware phylogeny generation using permutations of code [J]. J Comput Virol, 2005, 1: 13.

[8] Searles R, Xu L, Killian W, *et al*. Parallelization of machine learning applied to call graphs of binaries for malware detection [C]//Proceedings of the 2017 25th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP). Saint Petersburg: IEEE, 2017.

[9] Zhu R, Li C, Niu D, *et al*. Android malware detection using large-scale network representation learning [J/OL]. Comput Sci, 2018, /abs/1806.04847 [2018-06-13]. https://arxiv.org/abs/1806.04847.

[10] Chopra S, Hadsell R, LeCun Y. Learning a similarity metric discriminatively, with application to face verification [C]// 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. San Diego: IEEE, 2005.

[11] Bai Y, Ding H, Bian S, *et al*. Graph edit distance computation via graph neural networks [J]. Comput Sci, 2018, /abs/1808.05689 [2018-08-16]. https://arxiv.org/abs/1808.05689.

[12] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate [J/OL]. Comput Sci, 2014, /abs/1409.0473 [2014-09-01]. https://arxiv.org/abs/1409.0473.

[13] 李荣. 利用卷积神经网络的显著性区域预测方法 [J]. 重庆邮电大学学报: 自然科学版, 2019, 31: 37.

[14] Severyn A, Moschitti A. Learning to rank short text pairs with convolutional deep neural networks [C]//Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval. Santiago: ACM, 2015.

[15] Ester M, Kriegel H P, Sander J, *et al*. A density-based algorithm for discovering clusters in large spatial databases with noise [C]//Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining. Portland: AAAI Press, 1996.

[16] 徐健锐, 詹永照. 基于 Spark 的改进 K-means 快速聚类算法 [J]. 江苏大学学报: 自然科学版, 2018, 39: 73.

[17] 姜建华, 吴迪, 郝德浩, 等. 基于 CDbw 和人工蜂群优化的密度峰值聚类算法 [J]. 吉林大学学报: 理学版, 2018, 56: 1469.

[18] Rand W M. Objective criteria for the evaluation of clustering methods [J]. J Am Stati Assoc, 1971, 66: 846.

引用本文格式:

中 文: 刘凯, 方勇, 张磊, 等. 基于图卷积网络的恶意代码聚类 [J]. 四川大学学报: 自然科学版, 2019, 56: 654.

英 文: Liu K, Fang Y, Zhang L, *et al*. Malware clustering based on graph convolutional networks [J]. J Sichuan Univ: Nat Sci Ed, 2019, 56: 654.