doi :10.3969/j.issn.1671-1122.2020.11.007

基于 WGAN 的移动恶意对抗样本生成研究

李红娇，陈红艳

（上海电力大学计算机科学与技术学院，上海 201306）

摘　要：近年来，利用机器学习算法进行移动终端恶意软件的检测已成为研究热点，而恶意软件制作者为了使恶意软件能够逃避检测，采用各种方法来制作恶意对抗样本。文章提出一种基于 Wasserstein GAN（WGAN）的算法 MalWGAN 来生成移动终端恶意对抗样本，使其能够绕过基于机器学习算法的黑盒模型检测器来逃避检测。与现有基于静态梯度方法生成的对抗样本不同，MalWGAN 模型结合了 API 调用和静态特征来生成对抗样本。由于对抗样本是由黑盒模型检测器的反馈动态生成的，因此逃避黑盒模型检测器检测的概率更高。

关键词：对抗样本；WGAN；检测逃避

中图分类号：TP309　**文献标志码：**A　**文章编号：**1671-1122（2020）11-0051-08

中文引用格式：李红娇，陈红艳 . 基于 WGAN 的移动恶意对抗样本生成研究 [J]. 信息网络安全，2020，20（11）：51-58.

英文引用格式：LI Hongjiao, CHEN Hongyan. Research on Mobile Malicious Adversarial Sample Generation Based on WGAN[J]. Netinfo Security, 2020, 20(11): 51-58.

Research on Mobile Malicious Adversarial Sample Generation Based on WGAN

LI Hongjiao, CHEN Hongyan

(*School of Computer Science and Technology*, *Shanghai Electric Power University*, *Shanghai* 201306, *China*)

Abstract: In recent years, using machine learning algorithm to detect mobile terminal malware has become a research hotspot. In order to make the malware evade detection, malware producers use various methods to make malicious adversarial samples. This paper proposes an algorithm MalWGAN based on Wasserstein GAN (WGAN) to generate mobile terminal malicious adversarial samples, which can bypass the black box model detector based on machine learning algorithms to evade detection. Different from the existing adversarial samples generated by static gradient methods, the MalWGAN model combines API calls and static features to generate adversarial samples. Since adversarial samples are dynamically generated by the feedback of the black box model detector, the probability of escaping from the detection of the black box model detector is higher.

Key words: adversarial sample; WGAN; evasion of detection

收稿日期：2020-9-21

基金项目：国家自然科学基金 [61403247,61702321]；上海市信息安全综合管理技术研究重点实验室开放课题 [AGK2015005]；上海市科委地方能力建设项目 [15110500700]

作者简介：李红娇（1974—），女，河南，副教授，博士，主要研究方向为信息系统安全、隐私保护以及入侵检测等；陈红艳（1995—），女，安徽，硕士研究生，主要研究方向为信息安全。

通信作者：陈红艳 15000388434@163.com

0 引言

移动终端的操作系统自发布以来应用数量就呈现出爆炸性增长，由于Android操作系统的开源性以及应用市场的复杂性，移动恶意软件的数量也在急剧增加。根据McAfee的报告[1]，到2017年第3季度，移动恶意软件样本数量就已增加到2200万个。因此对用于检测和分析移动恶意软件的自动化工具的需求也在增加，利用机器学习算法来进行移动终端恶意软件的检测成为研究热点。例如，通过提取Android应用软件所请求的权限和敏感API作为特征，使用深度置信网络（DBN）进行Android恶意软件的检测[2]。

基于机器学习的检测器准确度高且能够检测出恶意软件的变体，但有研究发现，基于机器学习的恶意软件检测器容易受到对抗样本的攻击[3]。对抗样本是指对合理的输入添加经过精心设计的扰动后得到的样本，这些扰动会迫使机器学习模型输出错误的预测。现阶段对抗样本攻击主要分为白盒攻击和黑盒攻击，白盒攻击中攻击者知道模型所使用的算法以及算法所使用的参数，黑盒攻击中攻击者无法得知模型所使用的算法和参数。

基于梯度的攻击一般都是白盒攻击。GOODFELLOW[4]等人提出一种快速梯度符号方法（FGSM）来生成对抗性图像，在输入图像梯度方向上增加扰动导致模型输出一个高可信度的错误答案。KURAKIN[5]等人改进了FGSM算法，原FGSM算法是在一定阈值限制下沿着梯度方向移动一步或添加一次扰动，而改进算法则通过多步更小的移动，利用更加精细的扰动使得构造结果更加精准。PAPERNOT[6]等人提出基于Jacobian矩阵生成对抗样本的方法，该方法生成的对抗样本的扰动方向是目标类别标记的预测值的梯度方向。文献[7]研究了在Android恶意软件领域针对神经网络进行对抗性制作的可行性，采用了攻击神经网络正向导数的算法来生成对抗样本，该算法最初由PAPERNOT[6]等人提出。文献[7]在Android恶意软件数据集DREBIN上使用对抗样本来欺骗不同的基于神经网络的恶意软件分类器，实现了高达80%的错误分类率。

通常攻击者无法访问所要攻击的神经网络的体系结构和权重，所以黑盒攻击成为主流的对抗样本攻击。黑盒攻击一般通过估计目标函数的梯度和利用对抗样本的可转移属性这两种方法来完成攻击，实际的黑盒攻击主要关注对抗样本的可转移性。文献[8]通过训练替代模型来攻击黑盒目标模型，利用FGSM方法和Jacobian矩阵生成对抗样本，验证了攻击者在不了解真实模型的情况下生成的对抗样本针对DNN分类器的黑盒攻击是切实可行的。实验证明DNN分类器错误分类了84.24%的对抗样本。文献[9]增强了文献[8]中引入的算法，通过Jacobian矩阵扩充数据时不再使用固定步长作为参数，而是通过使步长周期性地在正负值之间交替来提高算法精度。同时，通过采用库采样来减少对分类器的查询数量以降低计算复杂度。与文献[9]只在小规模数据集上研究对抗样本的可转移性不同，文献[10]针对大规模数据样本集研究对抗样本的可转移性以及黑盒攻击下的特性，在没有替代模型的情况下使用基于集合的对抗样本生成方法来进行黑盒攻击。

现有的针对黑盒攻击的对抗样本生成方法都存在一个相同问题：黑盒攻击需要大量访问目标模型来获取输出结果，从而间接获取模型结构、训练参数等模型信息。为了解决这一问题，HU[11]等人利用生成式对抗网络中“生成”和“对抗”[12]的思想，提出MalGAN模型来生成恶意软件的对抗样本，生成的样本能够绕过基于机器学习算法的黑盒模型检测器。HU[11]等人所提模型的优点是没有直接攻击黑盒模型，而是使用替代检测器学习模型的梯度信息，将黑盒模型攻击问题转换成替代检测器的样本分类问题。唐川[13]等人利用深度卷积生成式对抗网络（DCGAN）来生成Android的恶意软件对抗样本，这些对抗样本可绕过基于卷积神经网络（CNN）的检测系统的检测，并且能实际运行而不影响原始恶意软件的恶意功能。文献[11]和文献[12]两种方法的局限性是使用的特征向量仅仅是API，无法保留原始恶意软件的特征。此外，训练通常不稳定，这是生成式对抗网

络（GAN）在训练上的一个通病。

与传统的GAN相比，Wasserstein GAN（WGAN）[14]大大降低了训练过程中出现崩溃的概率，保证了恶意对抗样本生成的多样性。本文利用WGAN生成移动终端恶意软件的对抗样本，这些对抗样本可以绕过基于机器学习算法的黑盒模型检测器。

本文的主要贡献有：

1）与现有的仅使用移动终端的API调用特征来生成对抗样本而无法保留原始恶意软件的特征的方法不同，本文方法结合了移动终端敏感API的调用信息和移动终端的静态特征信息，保留了原始恶意软件的特征，可以更为全面地表征移动终端应用。

2）提出一个名为MalWGAN的模型，首次将WGAN算法应用在移动终端恶意软件检测领域。该模型基于WGAN算法生成恶意软件对抗样本，这些对抗样本由基于机器学习算法的黑盒模型检测器的反馈动态生成，能够有效绕过黑盒模型检测器。MalWGAN模型的优势在于解决了训练不稳定问题，能够将生成的对抗样本的真正率降低到接近0。

1 WGAN 介绍

受博弈论的启发，GOODFELLOW[12]等人在2014年提出了生成式对抗网络（GAN）。GAN由生成器和判别器组成，生成器根据噪声来生成以假乱真的对抗样本，判别器用来区分接收的输入是真样本还是生成器生成的假样本。GAN本质上是生成器和判别器之间的动态博弈游戏，其基本结构如图1所示。

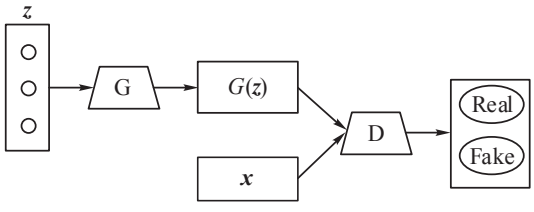

图 1 GAN 基本结构

图1中，z是输入的一个随机噪声向量；G是生成器，根据输入生成一个假样本$G(z)$；D是判别器，是一个二元分类器，输出为真实样本（Real）的概率和为假样本（Fake）的概率。GAN的核心原理可用值函数V（D，G）表示为

$$\min_{G}\max_{D}V(D,G)=E_{x\sim P_{data}(x)}(\log(D(x)))+E_{z\sim P_z(z)}(\log(1-D(G(z)))) \quad (1)$$

其中，E表示求分布的期望，$x\sim P_{data}(x)$表示真实样本的分布，$z\sim P_z(z)$为服从均匀分布或正态分布的随机噪声，$D(x)$表示D判断x是真实样本的概率，$D(G(z))$表示D判断$G(z)$是真实样本的概率。

ARJOVSKY[15]等人用数学推导证明了GAN中存在两个主要问题：一是D越训练到接近最优，G的损失函数$E_{z\sim P_z(z)}(\log(1-D(G(z))))$越近似为真实分布与生成分布之间的JS散度的最小化。当这两个分布之间不存在重叠或者重叠非常小时，JS散度就是一个常数，导致生成器的梯度近似为0，出现梯度消失的问题，从而使得G无法反馈足够的梯度以继续进行优化。二是如果D训练得比较弱，G会出现梯度不稳定、多样性不足的问题，从而导致模式崩溃。

为解决这些问题，ARJOVSKY[14]等人通过用EM距离（又称Wasserstein距离）代替JS散度，采用EM距离来确定生成器与真实数据分布间的差值，提出了WGAN。EM距离相对JS散度具有优越的平滑特性，理论上可以解决梯度消失问题。文献[14]中通过一系列数学公式推导出如公式（2）所示的近似EM距离公式。

$$W=E_{x\sim P_r}(f_w(x))-E_{x\sim P_g}(f_w(x)) \quad (2)$$

其中，W表示EM距离，P_r,P_g分别表示真实样本分布和由生成器生成的样本分布。含参数w的判别器网络f_w在限制w不超过某个范围的条件下使得W尽可能取最大值，此时W就会近似接近真实分布与生成分布之间的EM距离。

WGAN中生成器的损失函数和判别器的损失函数分别为

$$L(D)=-E_{x\sim P_r}(D(x))+E_{x\sim P_g}(D(x)) \quad (3)$$

$$L(G)=-E_{x\sim P_g}(D(x)) \quad (4)$$

WGAN实际上只修改了GAN的以下4个方面：

1）判别器的最后一层神经网络去掉了函数sigmoid()；

2）生成器和判别器的损失函数不取log()；

3）每次更新判别器的参数之后将参数的绝对值限制在不超过一个固定常数c，将权重限制到一定的范围内；

4）不使用基于动量的优化算法（包括momentum和Adam），推荐使用RMSProp、SGD作为优化器。

WGAN基本解决了GAN中的模式崩溃问题，确保了生成样本的多样性。此外，WGAN不需要小心平衡生成器与判别器在训练过程中的步长，也不需要精心设计网络结构，最简单的全连接层网络就可以使用。在训练过程中，可以用EM距离来指示训练进程，EM距离数值越小，代表WGAN训练得越好。

2 MalWGAN 设计与实现

本文提出的MalWGAN模型主要由生成器和替代检测器组成，输入随机噪声生成恶意对抗样本绕过黑盒模型检测器。黑盒模型检测器作为MalWGAN模型的外部系统，采用基于机器学习的恶意软件检测算法。

2.1 总体架构

图2给出了MalWGAN模型的总体架构。

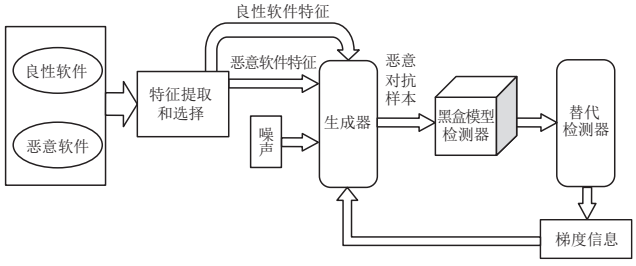

图 2 MalWGAN 模型总体架构

模型大致流程为：

1）采用静态分析方法提取应用软件数据集的特征；

2）将良性软件特征、恶意软件特征和生成的恶意对抗样本输入黑盒模型检测器，得到由黑盒模型检测器打上标签的样本；

3）将由步骤2）得到的样本输入替代检测器，学习黑盒模型检测器的特征；

4）将梯度信息反馈给生成器，实现生成器和替代检测器协同攻击黑盒模型检测器。

本文假设恶意软件的制作者唯一知道的是黑盒模型检测器中恶意软件检测器所使用的功能，并不知道使用了哪种机器学习算法，也不知道训练模型的参数。

2.2 应用软件特征提取和选择

应用软件的特征主要分为静态特征和动态特征。动态分析是指在严格控制的环境（如沙盒、虚拟机、物理隔绝的主机等）中执行软件的安装、运行等操作，缺点是资源和时间消耗很大。静态分析是指通过逆向手段抽取程序特征，分析其中的指令序列等。鉴于动态分析消耗过大，本文采用静态分析的方法提取应用软件特征并构造特征集。

本文以Android应用软件为例，对其特征进行提取和选择。如图3所示，本文从Android应用软件的APK文件中提取的特征包括权限、意图、组件和API调用接口等文件。权限用来在文件配置时声明应用程序必须拥有哪些权限才能访问API的受保护部分并与其他应用程序交互。意图反映了不同组件之间的关联关系。Android平台提供了一个框架API，由一组核心的软件包和类组成，应用程序可使用这些软件包和类与基础Android操作系统进行交互。因为大多数应用程序使用大量的API调用，因此大部分研究者都选择API调用作为Android特征的表征。

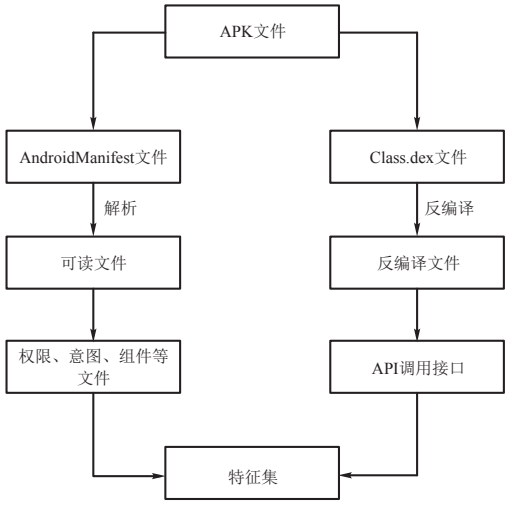

图 3 Android 应用软件的特征提取

本文选择了使用频率较高的357个Android特征[16]，为每个应用构造了一个357维的特征向量$\boldsymbol{m}$={1,0,0,0,

1,…}。特征表示是二进制的，当应用软件包含某特征时，特征向量中对应该特征的位置取1，否则取0。

2.3 MalWGAN 生成器

生成器将恶意软件特征向量$\boldsymbol{m}$和噪声向量$\boldsymbol{z}$作为输入，生成恶意对抗样本。其中，$\boldsymbol{m}$是M维的二元向量，即生成器网络的输出有M个神经元；$\boldsymbol{z}$是服从范围[–1,1]中均匀分布的随机向量，它的作用是允许生成器从单个恶意软件特征向量生成不同的对抗样本。生成器网络的最后一层使用的激活函数是tanh()，其他层的激活函数使用的是ReLU()的变体PReLU()。PReLU()是非饱和激活函数，使用PReLU()的优势在于PReLU()能解决梯度消失问题，且能加快收敛速度。

2.4 MalWGAN 替代检测器

替代检测器的训练数据来源包括良性Android应用软件的特征集合和由生成器生成的恶意对抗样本。因为制作恶意软件的攻击者对黑盒模型检测器的详细结构一无所知，因此使用替代检测器来学习黑盒模型检测器的特征，并提供梯度信息来训练生成器。黑盒模型检测器首先检测训练数据，并预测输出的程序是良性的还是恶意的，替代检测器使用预测标签来学习黑盒模型检测器的特征。

替代检测器是以特征向量集h为输入的多层前馈神经网络，其权重为θ_d。替代检测器的作用是检测程序是良性的还是恶意的，并实现和生成器协同攻击黑盒模型检测器。替代检测器使用$D_{\theta_d}(h)$来表示对恶意软件的预测概率。

2.5 训练 MalWGAN

WGAN是传统生成式对抗网络的升级版，它彻底解决了GAN训练不稳定的问题，还能使生成样本多样化。MalWGAN中替代检测器的损失函数如公式（5）所示。

$$L_{\mathrm{D}} = E_{h\in B_{\mathrm{Benign}}}(1-D_{\theta_d}(h)) - E_{h\in B_{\mathrm{Malware}}}(D_{\theta_d}(h)) \quad (5)$$

其中，B_{Benign}是被黑盒模型检测器检测为良性软件的集合，B_{Malware}是被黑盒模型检测器检测为恶意软件的集合。

生成器的损失函数如公式（6）所示。

$$L_{\mathrm{G}} = E_{\boldsymbol{m}\in S_{\mathrm{Malware}},z}(D_{\theta_d}(G_{\theta_g}(\boldsymbol{m},\boldsymbol{z}))) \quad (6)$$

其中，S_{Malware}是实际恶意软件数据集，$\boldsymbol{z}$是服从[–1,1]范围内均匀分布的噪声向量。L_{G}最小化会降低恶意软件被预测为恶意的概率，进而推动替代检测器将恶意软件识别为良性软件。生成器的训练会进一步愚弄黑盒模型检测器。

本文使用PReLU()作为激活函数，如公式（7）所示。

$$\mathrm{PReLU}(y_i)=\begin{cases} y_i & y_i>0 \\ a_i y_i & y_i\leqslant 0 \end{cases} \quad (7)$$

其中，y_i是输入数据，a_i是y_i小于或等于0时y_i的系数。

算法1给出了MalWGAN训练的全过程。

算法1 MalWGAN训练过程

While 算法未收敛时 **do**

采样生成小批量的恶意样本 *Mal*;

生成器输入 *Mal* 生成对抗样本 *Mal'*;

采样生成小批量的良性样本 *B*;

利用黑盒模型检测器对 *Mal'* 和 *B* 打上标签;

沿梯度方向$\nabla_{\theta_d}L_{\mathrm{D}}$下降来更新替代检测器的权重$\theta_d$;

沿梯度方向$\nabla_{\theta_g}L_{\mathrm{G}}$下降来更新替代检测器的权重$\theta_g$;

End While

MalWGAN生成的恶意对抗样本的概率分布是由生成器的权值决定的。训练集和测试集中的样本遵循相同或相似的概率分布才能使算法1有效。生成器能够更改对抗样本的概率分布，使其与黑盒模型检测器训练集的概率分布相差较大，从而生成器就可以引导黑盒模型检测器将恶意软件误分类为良性软件。

3 实验

3.1 实验环境

使用程序语言Python 3.6.0、Tensorflow 2.0.0和Keras 2.0.0创建MalWGAN，使用Scikitlearn 0.20.0创建恶意软件检测器。

为了验证MalWGAN生成的对抗样本的可转移性，本文在黑盒模型检测器中使用了几种不同的机器学习算法。近年来深度学习的兴起使得很多研究者将深度学习技术用于恶意软件的检测，但是目前投入实际应用的恶意软件检测算法大都是基于机器学习的，如DREBIN[17]、MAMADROID[18]等。本文在黑盒模型检测器中使用的

分类器主要有AdaBoost算法（AB）、决策树（DT）、随机森林（RF）、支持向量机（SVM）、逻辑回归（LR）、多层感知机（MLP）以及基于这些分类器的组合算法（VOTE）。

MalWGAN模型采用RMSprop作为优化器，用来更新和计算影响模型训练和模型网络的参数，使模型的损失函数达到最小化。RMSprop优化器的学习率设置为0.00005。RMSprop优化器的原理和动量梯度下降算法类似，它限制了垂直方向上的振荡，使其自身可在水平方向上采取更大的摆动幅度，从而可以更快地收敛。实验中生成器的层大小设置为357-256-128-64-357，替代检测器的层大小设置为357-700-1。

3.2 数据集处理

实验使用的Android恶意软件数据集从VirusShare[19]下载，良性软件数据集从Google Play抓取。使用敏感API调用和静态特征结合的Android软件特征357个，为每个应用程序构建了357维的二进制特征向量。

实验采用两种方式来分割数据集以进行训练。第一种分割方式是将80%的数据集作为训练集，剩下的20%作为测试集，MalWGAN和黑盒模型检测器使用相同的训练集。MalWGAN进一步挑选出25%的训练数据作为验证集，使用剩余的训练数据来训练神经网络。但是注意MalWGAN的验证集不能用于黑盒模型检测器，黑盒模型检测器需要使用不同的随机数来挑选验证集。

第二种分割方式是选择40%的数据集作为MalWGAN的训练集，另选择40%的数据集作为黑盒模型检测器的训练集，使用剩余20%的数据集作为测试集。

在现实世界里，攻击者收集的训练数据和恶意软件检测者所收集的训练数据是不同的。但是，如果他们都从公共来源收集数据，训练数据将存在相互重叠的情况。在这种情况下，MalWGAN的实际性能将介于上述两种数据分割方式的性能之间。

3.3 实验结果

模型的评价指标采用真正率（*TPR*，又叫召回率），其计算方式如公式（8）所示。

$$TPR=\frac{TP}{TP+FN} \tag{8}$$

其中，*TP* 表示实际标签为正、预测也为正的样本数量；*FN* 表示实际标签为负、预测却为正的样本数量。

1）第一种分割方式

首先分析MalWGAN和黑盒模型检测器使用相同训练集的情况，这时*TPR*表示在恶意软件检测中恶意软件的检测率。对抗性攻击之前，*TPR*越高，代表模型的检测能力越好。经过对抗性攻击之后，*TPR*的减少能够反映出恶意软件样本成功绕过了检测算法。表1给出了当MalWGAN和黑盒模型训练器使用相同的训练集时，原始样本和对抗样本的真正率。

表 1 使用相同训练集时原始样本与对抗样本的真正率

分类器	训练集		测试集	
	原始样本	对抗样本	原始样本	对抗样本
RF	95.7068	0.0000	95.7547	0.0000
LR	95.5853	0.0000	96.4401	0.0000
DT	99.7975	0.0000	1.0000	0.0000
SVM	95.0992	0.0000	94.3366	0.0000
AB	94.4920	0.0000	94.1747	0.0000
MLP	96.4763	0.0016	95.9547	0.0065
VOTE	96.0308	0.0000	94.9838	0.0000

从表1能够看出，RF、LR、DT、SVM、AB、MLP以及VOTE对训练数据都能进行较好的分类。除了MLP在训练集和测试集上的对抗样本的*TPR*在0.1%~0.7%的范围内、原始样本的*TPR*均大于95外，在当前数据集上各分类器的对抗样本的*TPR*基本都为0。也就是说，对于这些后端分类器，黑盒模型检测器几乎检测不到生成器生成的任何恶意对抗样本，即MalWGAN模型已经成功学习如何绕过这些基于机器学习的恶意软件检测算法。

在MalWGAN训练过程中，*TPR*在训练集和验证集上的收敛曲线如图4所示，其中*Epoch*为训练轮数。这里使用的黑盒模型检测器是MLP，MLP实际上是神经网络，替代检测器能够高精度和它拟合。

2）第二种分割方式

再来分析MalWGAN和黑盒模型检测器使用不同训

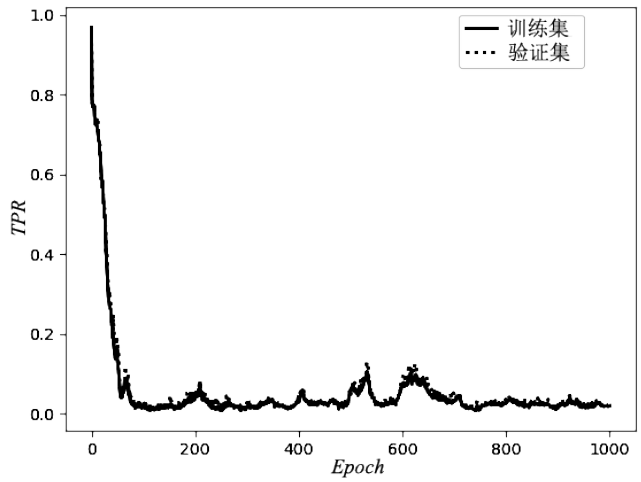

图 4 第一种分割方式下训练集和验证集上的 *TPR* 收敛曲线

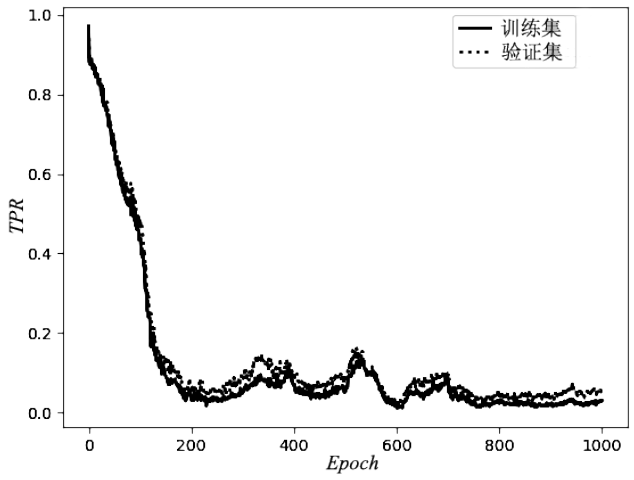

图 5 第二种分割方式下训练集和验证集的 *TPR* 收敛曲线

练集的情况。表2给出了当MalWGAN和黑盒模型检测器使用不同训练集时，原始样本和对抗样本的真正率。

表 2 使用不同训练集时原始样本与对抗样本的真正率

分类器	训练集		测试集	
	原始样本	对抗样本	原始样本	对抗样本
RF	95.4656	0.0008	96.2783	0.0032
LR	95.3846	0.0729	94.8220	0.0566
DT	99.8381	0.0000	99.8382	0.0000
SVM	94.7368	0.0000	94.3366	0.0000
AB	93.7652	0.0033	94.3666	0.0065
MLP	96.7611	0.0000	96.7638	0.0016
VOTE	96.2753	0.0000	95.1456	0.0000

由表2可知，在训练集和测试集上，DT、SVM和VOTE的对抗样本的*TPR*都为0，RF、LR和AB的对抗样本的*TPR*比第一种分割方式要高，但是增加的数字非常小，黑盒模型检测器仍然无法检测出大多数恶意对抗样本。由此可以得出结论，即使在不同的训练集上，MalWGAN仍然可以成功欺骗黑盒模型检测器。

此种分割方式下，*TPR*在训练集和验证集上的收敛曲线如图5所示，其中*Epoch*为训练轮数。使用的黑盒模型检测器仍然是MLP。

3.4 与 HU[11] 等人的算法对比

现有的对抗样本生成算法主要是针对图像的。图像与恶意软件的最大区别在于，图像的特征是连续的，而恶意软件的特征是二进制的。

HU[11]等人将使用机器学习的恶意软件检测器的检测结果集成到基于GAN的神经网络中，以生成恶意对抗样本逃避检测。但是其算法中使用的特征向量仅仅是API，无法保留原始恶意软件的特征。将HU等人提出的算法（用MalGAN表示）和本文提出的算法（用MalWGAN表示）的*TPR*进行对比。首先对比第一种分割方式下的*TPR*，如图6所示。

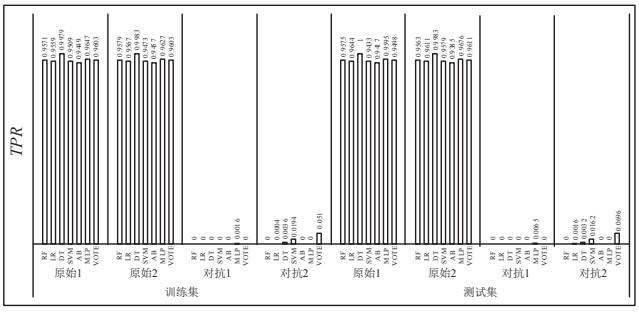

图 6 第一种分割方式下 MalWGAN 与 MalGAN 的 *TPR* 对比

图6中，原始1、对抗1表示MalWGAN中的原始样本和对抗样本，原始2、对抗2表示MalGAN中的原始样本和对抗样本。由图6可以看出，MalWGAN的对抗样本的*TPR*几乎均为0，比MalGAN的对抗样本的*TPR*更低，意味着MalWGAN绕过这些基于机器学习的恶意软件检测算法的概率更高，更加难以被检测到。

第二种分割方式下的*TPR*的对比情况如图7所示。图7中，原始1、对抗1表示MalWGAN中的原始样本和对抗样本，原始2、对抗2表示MalGAN中的原始样本和对抗样本。由图7可以看出，当黑盒模型检测器

算法为DT、SVM和VOTE时，MalWGAN的对抗样本的*TPR*比MalGAN的对抗样本的*TPR*要低。当黑盒模型检测器使用其他几种机器学习算法时，MalWGAN的对抗样本的*TPR*比MalGAN的对抗样本的*TPR*略高。

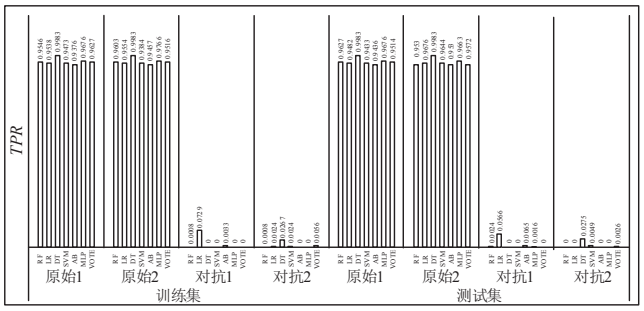

图 7 第二种分割方式下 MalWGAN 与 MalGAN 的 *TPR* 对比

4 结束语

本文提出一种名为MalWGAN的模型，首次将WGAN算法应用在Android恶意软件检测领域。实验结果表明，MalWGAN模型生成的恶意对抗样本可以有效绕过基于机器学习的黑盒模型检测器，且算法训练更稳定，绕过黑盒模型检测器的概率更高。（责编 马珂）

参考文献：

[1] McAfee. McAfee Mobile Threat Report Q1 2018[EB/OL]. https://www.mcafee.com/enterprise/en-us/assets/reports/rp-mobile-threat-report-2018.pdf, 2020-3-20.

[2] OUYANG Li, LU Tianliang. Android Malware Detection Based on Deep Belief Network[J]. Information Technology and Network Security, 2019，5(5): 22-27.
欧阳立、卢天亮 . 基于深度置信网络的 Android 恶意软件检测 [J]. 信息技术与网络安全，2019，5(5)：22-27.

[3] CARLINI N, WAGNER D. Towards Evaluating the Robustness of Neural Networks[C]//IEEE. 2017 IEEE Symposium on Security and Privacy (SP), May 22-26, 2017, San Jose, CA, USA. NJ: IEEE, 2017: 39-57.

[4] GOODFELLOW I, SHLENS J, SZEGEDY C. Explaining and Harnessing Adversarial Examples[EB/OL]. https://www.researchgate.net/publication/269935591_Explaining_and_Harnessing_Adversarial_Examples, 2020-3-20.

[5] KURAKIN A, GOODFELLOW I, BENGIO S. Adversarial Examples in the Physical World[EB/OL]. https://www.researchgate.net/publication/305186613_Adversarial_examples_in_the_physical_world, 2020-3-20.

[6] PAPERNOT N, MCDANIEL P, JHA S, et al. The Limitations of Deep Learning in Adversarial Settings[C]//IEEE. 2016 IEEE European Symposium on Security and Privacy (EuroS&P), March 21-24, 2016, Saarbrucken, Germany. NJ: IEEE, 2016: 372-387.

[7] KATHRIN G, NICOLAS P, PRAVEEN M, et al. Adversarial Perturbations against Deep Neural Networks for Malware Classification[EB/OL]. https://arxiv.org/abs/1606.04435, 2016-6-16.

[8] PAPERNOT N, MCDANIEL P, GOODFELLOW I, et al. Practical Black-box Attacks against Deep Learning Systems Using Adversarial Examples[EB/OL]. https://arxiv.org/abs/1602.02697v2, 2016-2-19.

[9] PAPERNOT N, MCDANIEL P, GOODFELLOW I. Transferability in Machine Learning: from Phenomena to Black-box Attacks Using Adversarial Samples[EB/OL]. https://arxiv.org/abs/1605.07277, 2016-5-24.

[10] LIU Yanpei, CHEN Xinyun, LIU Chang, et al. Delving into Transferable Adversarial Examples and Black-box Attacks[EB/OL]. https://arxiv.org/abs/1611.02770, 2017-2-7.

[11] HU Weiwei, TAN Ying. Generating Adversarial Malware Examples for Black-Box Attacks Based on GAN[EB/OL]. https://arxiv.org/abs/1702.05983, 2017-2-20.

[12] GOODFELLOW I, POUGETABADIE J, MIRZA M, et al. Generative Adversarial Networks[EB/OL]. https://arxiv.org/abs/1406.2661, 2014-1-10.

[13]TANG Chuan, ZHANG Yi, YANG Yuexiang, et al. DroidGAN: an Android Adversarial Sample Generation Framework Based on DCGAN[J]. Journal on Communications, 2018, 18(S1): 64-69.
唐川，张义，杨岳湘，等 . DroidGAN：基于 DCGAN 的 Android 对抗样本生成框架 [J]. 通信学报，2018，18(S1)：64-69.

[14] ARJOVSKY M, CHINTALA S, BOTTOU L. Wasserstein GAN[EB/OL]. https://arxiv.org/abs/1701.07875, 2017-12-6.

[15] ARJOVSKY M, BOTTOU L. Towards Principled Methods for Training generative Adversarial Networks[EB/OL]. https://arxiv.org/abs/1701.04862, 2017-1-17.

[16] CHEN Sen, XUE Minhui, FAN Lingling, et al. Automated Poisoning Attacks and Defenses in Malware Detection Systems: An Adversarial Machine Learning Approach[J]. Computers & Security, 2018, 73 (3): 326-344.

[17] DEMONTIS A, MELIS M, BIGGIO B, et al. Yes, Machine Learning Can Be More Secure! A Case Study on Android Malware Detection[J]. IEEE Transactions on Dependable and Secure Computing, 2019, 16(4): 711-724.

[18] ONWUZURIKE L, MARICONTI E, ANDRIOTIS P, et al. MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Models[EB/OL]. https://arxiv.org/abs/1711.07477, 2019-3-2.

[19] Corvus Forensics. VirusShare[EB/OL]. https://virusshare.com/, 2020-3-20.