

基于模型相似度的模型恶意代码夹带检测方法

汪德刚, 孙奕, 周传鑫, 高琦, 杨帆

(信息工程大学, 河南 郑州 450001)

摘要: 联邦学习主要通过源数据不出本地而仅交互模型参数的方式保护参与共享用户数据的隐私安全, 然而其仍然面临众多安全挑战, 目前研究者主要针对如何增强模型隐私保护和检测恶意模型攻击等问题展开较为广泛的研究, 然而利用联邦学习过程中频繁交互的模型数据进行恶意代码夹带导致风险扩散的问题鲜有研究。针对联邦学习训练过程中通过模型传递恶意代码导致风险扩散的问题, 提出一种基于模型相似度的模型恶意代码夹带检测方法。通过分析联邦学习本地模型与全局模型的迭代过程, 提出计算模型距离的方法, 并使用模型距离量化模型之间的相似度, 最终根据各客户端模型之间的相似度对携带恶意代码的模型进行检测。实验结果表明, 提出的检测方法具有较好的性能指标, 当训练集为独立同分布时, 在 178 MB 大小的模型中嵌入 0.375 MB 恶意代码, 检测方法的真正率为 82.9%, 误报率为 1.8%; 嵌入 0.75 MB 恶意代码时, 检测方法的真正率为 96.6%, 误报率为 0.38%。当训练集为非独立同分布时, 检测方法的准确率随恶意代码嵌入率以及联邦学习训练轮数的增加而增加。在对恶意代码进行加密的情况下, 提出的检测方法仍然能够达到 90% 以上的准确率。在多攻击者的场景中, 攻击者数量已知与未知时的检测方法准确率均能保持在 90% 左右。

关键词: 联邦学习; 模型; 模型相似度; 恶意代码; 检测

中图分类号: TP309.2

文献标志码: A

DOI: 10.11959/j.issn.2096-109x.2023056

Malicious code within model detection method based on model similarity

WANG Degang, SUN Yi, ZHOU Chuanxin, GAO Qi, YANG Fan

Information Engineering University, Zhengzhou 450001, China

Abstract: The privacy of user data in federated learning is mainly protected by exchanging model parameters instead of source data. However, federated learning still encounters many security challenges. Extensive research has been conducted to enhance model privacy and detect malicious model attacks. Nevertheless, the issue of risk-spreading through malicious code propagation during the frequent exchange of model data in the federated learning process has received limited attention. To address this issue, a method for detecting malicious code within

收稿日期: 2022-03-29; 修回日期: 2022-05-30

通信作者: 孙奕, 11112072@bjtu.edu.cn

引用格式: 汪德刚 孙奕, 周传鑫, 等. 基于模型相似度的模型恶意代码夹带检测方法[J]. 网络与信息安全学报, 2023, 9(4): 90-103.

Citation Format: WANG D G, SUN Y, ZHOU C X, et al. Malicious code within model detection method based on model similarity[J]. Chinese Journal of Network and Information Security, 2023, 9(4): 90-103.

models, based on model similarity, was proposed. By analyzing the iterative process of local and global models in federated learning, a model distance calculation method was introduced to quantify the similarity between models. Subsequently, the presence of a model carrying malicious code is detected based on the similarity between client models. Experimental results demonstrate the effectiveness of the proposed detection method. For a 178MB model containing 0.375MB embedded malicious code in a training set that is independent and identically distributed, the detection method achieves a true rate of 82.9% and a false positive rate of 1.8%. With 0.75MB of malicious code embedded in the model, the detection method achieves a true rate of 96.6% and a false positive rate of 0.38%. In the case of a non-independent and non-identically distributed training set, the accuracy of the detection method improves as the rate of malicious code embedding and the number of federated learning training rounds increase. Even when the malicious code is encrypted, the accuracy of the proposed detection method still achieves over 90%. In a multi-attacker scenario, the detection method maintains an accuracy of approximately 90% regardless of whether the number of attackers is known or unknown.

Keywords: federated learning, model, model similarity, malicious code, detection

0 引言

近年来, 隐私泄露事件频发, 隐私问题比以往更受关注。作为一种具有隐私保护特性的分布式机器学习训练框架, 联邦学习^[1]迅速成为研究热点。联邦学习主要通过源数据不出本地而仅交互模型参数的方式保护参与共享用户数据的隐私安全, 实现数据安全共享与协同分析。然而, 由于模型反演^[2-3]、成员推理^[4]、模型提取^[5-6]以及模型投毒^[7-12]等攻击的存在, 共享模型仍然存在模型隐私泄露和恶意模型攻击等问题, 模型隐私泄露是指攻击者利用本地模型或全局模型恢复源数据或提取关于源数据的一些隐私信息, 如银行卡号^[13]等。恶意模型攻击的目的是影响最终训练得到的全局模型质量, 如在全局模型中植入后门^[8]。目前研究者主要针对如何增强模型隐私保护和检测恶意模型攻击等问题展开较为广泛的研究, 然而利用联邦学习过程中频繁交互的模型数据进行恶意代码夹带导致风险扩散的问题鲜有研究。现有研究^[14-16]表明, 攻击者可以在模型参数中嵌入恶意代码以规避安全检测, 这种方法可以在不影响模型精度的同时携带大型恶意软件, 达到恶意代码扩散的目的, 对整个联邦学习系统造成严重威胁。因此, 如何在联邦学习系统中有效、准确地进行模型恶意代码夹带检测是亟待解决的问题。

本文提出一种基于模型相似度的模型恶意代码夹带检测方法, 使用模型距离来量化客户端模

型之间的相似度, 进而对携带恶意代码的模型进行检测, 本文的贡献点如下。

1) 首次分析了联邦学习中利用模型数据进行恶意代码传播导致风险扩散的问题并提出相应的检测方法。

2) 通过分析联邦学习本地模型与全局模型的迭代过程, 提出计算模型距离的方法并使用模型距离来量化模型之间的相似度。

3) 提出基于模型相似度的恶意代码夹带检测方法, 并通过实验证明提出的检测方法具有较好的性能指标, 如在 178 MB 的大小模型中嵌入 0.375 MB 恶意代码时, 检测方法的真正率为 82.9%, 误报率为 1.8%; 嵌入 0.75 MB 恶意代码时, 检测方法的真正率为 96.6%, 误报率为 0.38%。

1 相关研究

1.1 联邦学习

联邦学习最早由 Google 于 2016 年提出^[1], 其本质是一种分布式机器学习训练框架, 主要包括客户端 (也称参与方) 和中心服务器 (也称参数聚合器, 以下简称中心)。客户端在中心的协调下共同训练模型, 其中客户端使用本地数据训练得到本地模型, 中心负责对本地图模型进行加权聚合, 生成全局模型。经过多轮迭代后得到一个与集中式训练精度相似的模型, 有效降低了集中式训练通过聚合源数据带来的数据隐私泄露风险。

联邦学习工作原理如图1所示, 假设有 k 个客户端, 各自拥有隐私数据集 $D_i (i \in k)$, 在中心的协调下进行机器学习训练, 其中第 r 轮训练过程如下。① 各客户端从中心下载第 $r-1$ 轮训练得到的全局模型 θ^r , 然后使用数据集 D_i 对 θ^r 继续训练得到本地模型 θ_i^{r+1} 。② 各方将 θ_i^{r+1} 上传给中心。③ 中心收到所有本地模型后进行加权平均, 得到全局模型 θ^{r+1} 。重复以上3个步骤直到全局模型收敛或者模型精度达到指定的阈值即可停止训练。在第一轮训练时, 客户端可以从中心下载预训练模型 θ^1 , 否则各个客户端直接使用本地数据进行训练。

根据数据特征和数据样本在客户端中的分布情况, 联邦学习一般可以分为横向联邦学习

(HFL, horizontally federated learning)、纵向联邦学习 (VFL, vertically federated learning) 和联邦迁移学习 (FTL, federated transfer learning) [17]。Lyu 等[18]根据参与方的数量、参与模型聚合的频率和技术能力进一步将 HFL 分为面向企业的 HFL (H2B) 和面向消费者的 HFL (H2C)。在 H2B 模式下, 通常只有少数参与者, 每个参与者可以频繁地被选择进行参数聚合, 参与者往往具有较高的技术能力和算力。在 H2C 模式下, 可能有数千甚至数百万的参与者, 在每一轮的训练中, 通常聚合器只会选择其中一个子集进行参数聚合, 因此每个参与者被频繁地选择进行参数聚合的概率很低, 它们通常具有有限的算力。

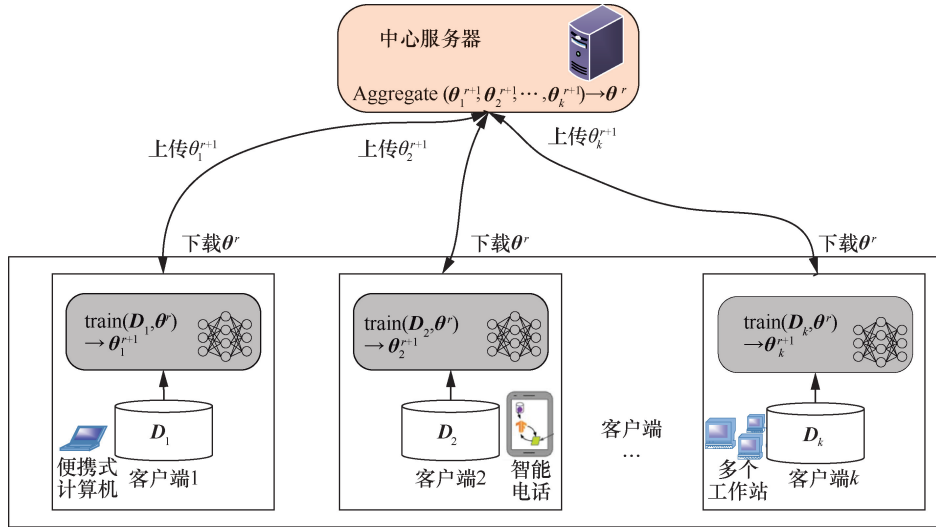


图1 联邦学习工作原理

Figure 1 Workflow of federated learning

1.2 恶意代码嵌入与检测

在僵尸网络、勒索软件、APT 等高级恶意软件活动中, 一些攻击者会将恶意代码附加到看似无害的载体 (如图片等) 文件中而不破坏载体的结构。虽然普通用户通常看不到这些秘密信息, 但它们很容易被检测软件发现。更加隐蔽的方法是使用隐写术[19-20]将要传递的恶意代码嵌入载体中, 然而, 一般的隐写载体由于信道容量小, 不适合传递大型恶意软件。现有研究[14-16]表明, 攻击者可以在机器学习模型参数中嵌入恶意代码以规避安全检测, 这种方法可以在不影响模型精度的同时携带大型恶意软

件。为了叙述方便以及不与联邦学习中的恶意模型混淆, 下文将嵌入恶意代码的机器学习模型统称为恶意代码模型。恶意代码和恶意软件术语交替使用, 都是指能够在目标主机执行的恶意程序。

在神经网络模型中, 每个神经元都有若干权重参数和1个偏置参数, 如图2所示。模型隐藏层 h_1 的第一个神经元的输出可以由式(1)计算。

$$o_1 = f(\sum x_i \cdot w_{1i} + b_1) \quad (1)$$

其中, f 代表激活函数, x_i 代表上层输入, w_{1i} 代

表各输入对本神经元的权重值， b_i 为本神经元的偏置值。大多数机器学习编程框架使用 IEEE 754^[21]单精度浮点数作为神经网络参数的默认存储类型。如图3所示，IEEE 754单精度浮点数可

以表示为 $S \times 1.M \times 2^E$ 。其中，符号位(S)决定该浮点数的正负；指数位(E)是2的幂，它的作用是对浮点数加权；尾数位(M)是二进制小数，代表浮点数的有效小数。

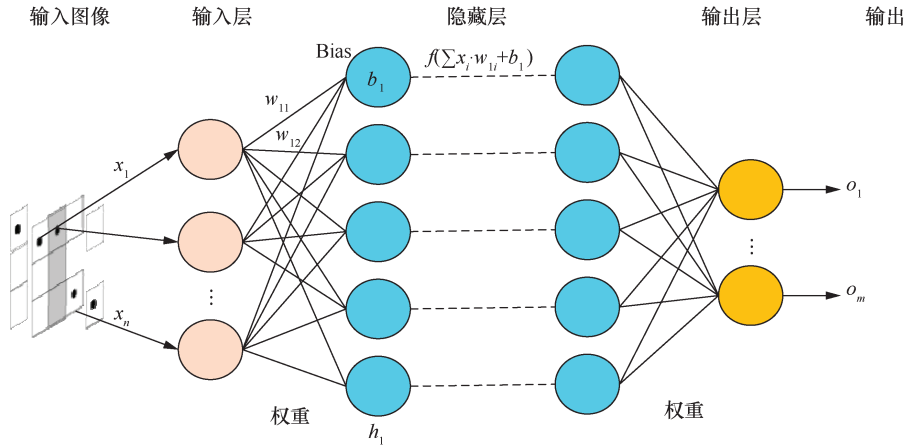


图2 多层神经网络模型
Figure 2 Multi-layer neural network

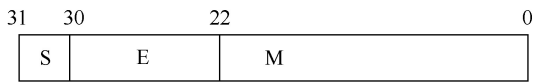


图3 IEEE 754单精度浮点数表示方法
Figure 3 IEEE 754 single precision floating point number representation

Liu等^[14]提出的StegoNet通过先将恶意软件嵌入深度神经网络(DNN, deep neural network)模型中,然后通过DNN模型供应链进行传播。StegoNet使用4种嵌入恶意软件的方法:最低有效位(LSB)替换、弹性训练、值映射和符号映射。借鉴图像隐写中的LSB算法^[20],StegoNet通过替换模型参数最低有效位将恶意软件嵌入DNN模型中。对于大型DNN模型,该方法可以在不降低模型性能的前提下嵌入大型恶意软件。但对于较小的模型,随着恶意软件字节数的增加,模型性能会急剧下降。在弹性训练中,攻击者首先挑选一些模型参数进行恶意软件嵌入,然后将这些参数固定,再次训练该模型,使其精度恢复至未嵌入恶意软件时的状态。与LSB替换方法相比,该方法可以在模型中嵌入更大的恶意软件。值映射和符号映射方法将恶意软件字节映射到模型参数中,但是这种映射方法需要一个置换映射表来恢复恶意软件,这会限制嵌入恶意软件的大小。

Wang等^[15]通过分析浮点数的格式,发现浮

点数的绝对值是由指数部分决定的,通过调整指数部分可以使浮点数绝对值固定在一定的区间内。例如,将浮点数的第3~6 bit设为1,最后24 bit设为任意值(即0x3c000000~0x3cffffff),则浮点数的绝对值在0.0078~0.0313。因此,Wang等^[15]提出固定神经元参数的首字节为0x3c或0xbc等值,在其后的3 byte嵌入恶意软件,以此将参数变化限制在一定区间内,然后通过再训练的方式保证模型的精度。他们在不同的模型和恶意软件样本上验证了快速替换的可行性,如在178 MB的AlexNet模型中嵌入36.9 MB的恶意代码,而模型的精度损失不到1%。

Liu等^[14]和Wang等^[15]通过实验发现常规的恶意软件检测引擎无法对恶意代码模型进行检测。Liu等^[14]通过实验指出,由于DNN的复杂性,使用传统的隐写分析方法^[22]无法对StegoNet进行有效检测。Wang等^[15]提出可以将神经元参数,先由浮点数转换为十六进制数表示,再与现有的恶意软件特征数据库进行对比,但是这种基于特征数据库的检测方法很容易通过加密等方式进行规避。他们还指出,可以通过模型压缩、裁剪和调优等方式主动破坏模型中可能存在的恶意软件,同时保持模型的精度不受影响。

在联邦学习中,中心只是将各个本地模型进

行加权聚合得到全局模型后下发给各个客户端。训练一旦开始,机器学习模型结构、超参数等信息就已确定,无法再修改。因此,主动破坏模型中可能存在的恶意软件这种防御方法在联邦学习中并不适用。本文提出一种基于模型相似度的模型恶意代码夹带检测方法,利用诚实模型之间相似度大于诚实模型与恶意代码模型之间相似度的特点来检测恶意代码模型。

2 威胁模型

文献[14-15]假设攻击者通过模型供应链进行

恶意代码传播,如图4所示,攻击者首先自行训练或下载一个训练好的模型,然后将恶意软件嵌入其中,根据嵌入代码后模型的精度来判断是否需要再微调模型以提高模型的性能。随后攻击者将嵌入恶意代码的模型发布到互联网上,控制受害主机下载该模型,提取恶意软件并执行。

本文的威胁模型建立在面向企业的横向联邦学习场景下,如图5所示,在该联邦学习系统中有 k 个客户端和1个中心服务器,威胁模型定义如下。

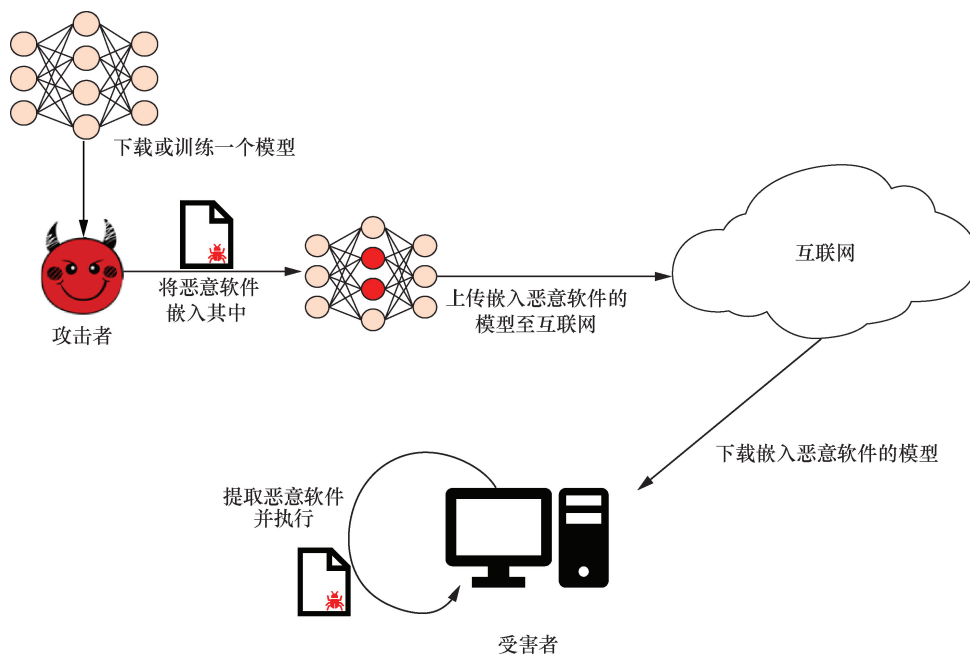


图4 通过模型供应链进行恶意代码传播
Figure 4 Malicious code spreading through the model supply chain

受害者:受害者为联邦学习中心。假设中心在一个安全的环境中部署模型聚合服务,该安全环境使用本文提出的检测方法对收到的客户端模型进行分析,检测通过后还会对客户端模型进行性能测试。如果客户端模型没有通过检测或模型性能过低,则不会发送给中心。

攻击者:假设攻击者为参与联邦学习训练的一个或多个客户端。攻击者拥有训练模型的数据,并且会遵守联邦学习协议正常执行模型训练,只是在本地模型训练结束后在模型中嵌入恶意软件。

攻击者的目标:攻击者的目标是在中心服务器上执行恶意软件,因为中心会对客户端模型的

性能进行测试,所以攻击者需要保证嵌入恶意代码的模型性能不会过低而导致模型被拒绝。

诚实模型:诚实模型是指客户端遵循联邦学习协议训练得到未嵌入任何数据的本地模型。

3 基于模型相似度的检测方法

3.1 模型相似性分析

在联邦学习每轮训练开始时,中心会将上一轮训练得到的全局模型 θ^r 发送给参与训练的各个客户端。因为各客户端都是在模型 θ^r 的基础上使用其私有数据继续训练,并且各客户端都有一个共同的目标,那就是减小模型损失(提高模型准确率),

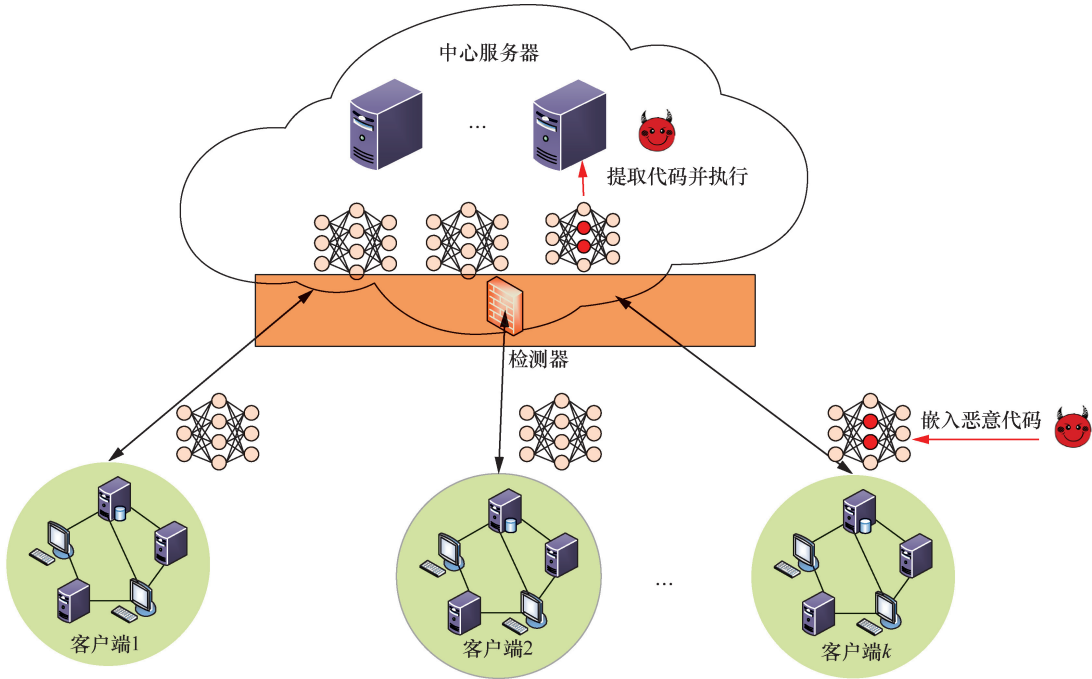


图5 联邦学习中利用模型进行恶意代码传递
Figure 5 Malicious code spreading using models in federated learning

所以各客户端得到的本地模型 θ_i^r 彼此相似。如式(2)所示, 客户端模型参数 x_i^r 可以看作全局模型参数 x^r 加上一个很小的扰动量 ε ($\varepsilon \in \mathbb{R}$) 得到, 因此各诚实模型的参数值在一个很小的区间内波动。

$$x_i^r = x^r + \varepsilon \quad (x^r \in \theta^r) \quad (2)$$

快速替换方法^[15]在模型参数的后 3 byte 随意嵌入恶意代码, 并且不像诚实模型那样以提高模型准确率为目标改变模型参数的大小, 势必会降低恶意代码模型与诚实模型之间的相似度。如图6所示, 将 2 个诚实模型 (model_0、model_1) 和 1 个恶意代码模型 (malware model) 对应参数值相减再取绝对值 (计算模型参数之间的距离), 可以发现多数情况下诚实模型与恶意代码模型的距离曲线在诚实模型之间的距离曲线之上。

在联邦学习中, 无论各参与方的数据集为独立同分布 (IID, independent identically distributed) 还是非独立同分布 (non-IID), 随着训练的持续进行, 参数聚合操作以及各客户端模型的损失逐渐减小导致客户端模型改变会越来越小, 所以各客户端模型之间的相似度会越来越大, 本文检测方法检测性能也会持续增加。对于非独立同分布的数据, 只是训练初期各客户端模型之间的相似度

差异较大, 导致检测算法的性能较低。

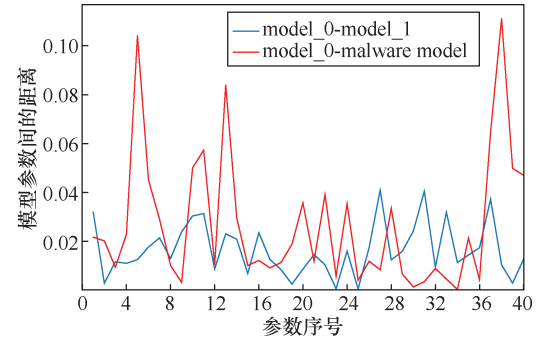


图6 模型参数间的距离
Figure 6 Model parameter distance

3.2 模型距离

由上面的分析可知, 可以使用参数之间的距离来量化模型之间的相似度。从图6可以发现, 恶意代码模型与诚实模型之间参数距离小于诚实模型之间距离, 这是因为各客户端的数据不同, 部分参数在训练时的变化方向和幅度会有所不同。为了克服这个问题, 给出模型距离的计算方法: 给定两个模型 θ_x 和 θ_y , 其模型距离为各参数间的距离之和, 如式(3)所示。

$$D_{\theta_x, \theta_y} = \sum_{i=1}^n |x_i - y_i| \quad (3)$$

其中, x_i 和 y_i 为模型 θ_x 和 θ_y 相同位置的神经元参数值, $|x_i - y_i|$ 代表 x_i 和 y_i 的距离, n 为需要计算距离的参数总量。模型距离是参数间距离的统计量, 能够有效平衡诚实模型间少量参数间距离大于诚实模型与恶意代码模型参数间距离的情况。

3.3 检测工作原理

基于上面的工作, 本文提出基于模型相似度的模型恶意代码夹带检测方法, 使用模型距离来量化模型之间的相似度, 与其他模型相似度最低的模型将被判定为恶意代码模型。检测原理如下。

1) 中心收集全部客户端模型。对每个客户端模型 θ_i^r , 计算 θ_i^r 与剩余其他模型 θ_j^r 的模型距离 $D_{i,j}$, 得到一个距离列表 $\{D_{i,j}\}_i^{k-1}$, 共可得到 k 个距离列表。

2) $\{D_{i,j}\}_i^{k-1}$ 中最大的 $D_{i,j}$ 代表模型 θ_j^r 与模型 θ_i^r 的相似度最低, 更可能为恶意代码模型, 因此对模型 θ_j^r 进行投票。对所有距离列表进行上述操作, 统计所有投票, 得票数大于或等于阈值 t 的客户端模型被判定为恶意代码模型。

3) 对于多个攻击者的情况, 首先执行上述步骤, 假设客户端模型 θ_z^r 被判定为恶意代码模型, 则将 θ_z^r 以外的客户端模型对应的距离列表 $\{D_{i,j}\}_i^{k-1}$ 中的 $D_{i,z}$ 去除, 得到 $k-1$ 个新的距离列表, 然后执行步骤 2), 得到下一个恶意代码模型。重复执行步骤 3) 即可对多个攻击者进行检测。

各客户端模型之间的距离如图 7 所示, 可以发现模型 0 与其他模型之间的相似度都是最低的, 所以模型 0 被判定为恶意代码模型。

4 实验

4.1 实验准备

4.1.1 模型选择与恶意软件嵌入

本文选择两个卷积神经网络模型 AlexNet 和 CNNMnist 作为载体进行恶意软件嵌入, CNNMnist 和 AlexNet 分别在 MNIST 和 Fashion-MNIST 数据集上训练得到。CNNMnist 模型由 2 个卷积层和 2 个全连接层组成, AlexNet 由 5 个卷积层和 3 个全连接层组成, 其参数分布及大小如表 1 所示。相对于卷积层, 全连接层神经元数量更多、嵌入容量更大, 且修改离输出层更近的参数层对模型的精度影响较大, 所以只在模型的全连接层 FC1 嵌入恶意代码, 这与 Wang 等^[15]的设置相同。CNNMnist 全连接层 FC1 有 50 个神经元, 每个神经元有 320 个权重参数, 所以 CNNMnist 模型的全连接层 FC1 可嵌入 $50 \times 320 \times 3 / 1024 = 46.875$ kB 大小的恶意软件。同理, AlexNet 的全连接层 FC1 最多可以嵌入 75 MB 大小的恶意软件。本文从 InQuest^[23]恶意软件样本中选取 10 个样本, 如表 2 所示 (md5 值只显示前 4 byte), 覆盖 Windows 上主流的恶意软件类型, 包含 EXE、DLL、PDF、DOC 和 PowerShell Script (表 2 中用 PSS 表示), 这些恶意软件样本均可被

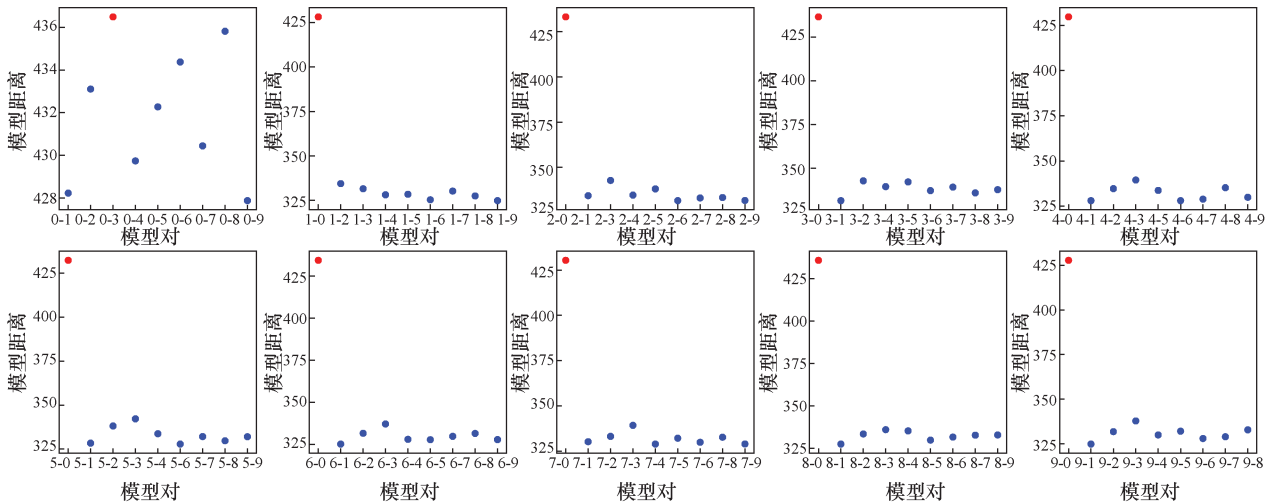


图 7 各客户端模型之间的距离
Figure 7 Distance between client models

在线病毒查杀工具 Virus Total 检测到。

本文主要考虑攻击者通过模型传递大型恶意软件带来的风险问题，因此使用 Wang 等^[15]提出的快速替换方法在模型中嵌入恶意软件，该方法能够在几乎不影响模型精度的情况下嵌入大体积的恶意软件。如图 8 所示，恶意软件嵌入流程可以分为参数提取、代码嵌入和参数替换 3 个步骤。嵌入率（ER，embedding rate）是指嵌入的恶意软件字节数占模型容量的百分比。在实验中，考虑不同嵌入率对检测结果的影响。在恶意代码实际嵌入过程中，当恶意软件已经完全嵌入但还未达到指定嵌入率时，选择重复嵌入该恶意软件，而当恶意软件还未完全嵌入但已经达到指定嵌入率时停止嵌入。

表 1 载体模型参数分布及大小

Table 1 Distribution and size of carrier parameters

模型名称	数据集	层类型	参数数量	参数总大小
CNNMnist	MNIST	Conv-1	260	85 kB
		Conv-2	5 020	
		FC1	16 050	
		FC2	510	
AlexNet	Fashion-MNIST	Conv-1	11 712	178 MB
		Conv-2	614 656	
		Conv-3	885 120	
		Conv-4	1 327 488	
		Conv-5	884 992	
		FC1	26 218 496	
		FC2	16 781 312	
		FC3	40 970	

4.1.2 检测评价指标

判断一个模型是否为恶意代码模型是一个二分类问题，可根据模型的真实类别与检测算法预测类别的组合划分为真正例（true positive）、假正

例（false positive）、真反例（true negative）、假反例（false negative）4 种情形，令 TP、FP、TN、FN 分别表示其对应的样本数量，如表 3 所示，显然有 $TP+FP+TN+FN=\text{样本总数}$ 。对于恶意软件检测，常用的度量指标包括准确率、精确率、召回率/真正率、误报率和漏报率^[24]，计算方法如表 4 所示。

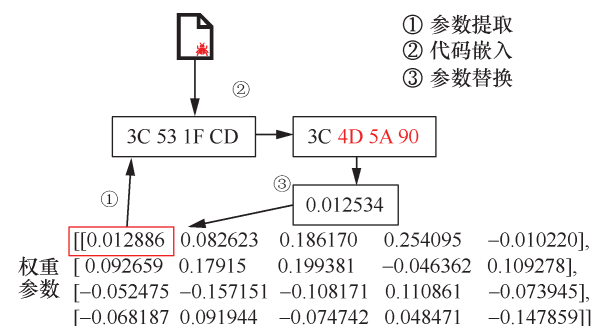


图 8 恶意软件嵌入流程
Figure 8 Malware embedding process

表 3 分类结果混淆矩阵

Table 3 Classification result confusion matrix

真实类别	预测结果	
	恶意模型	诚实模型
恶意模型	TP（真正例）	FN（假反例）
诚实模型	FP（假正例）	TN（真反例）

4.2 实验步骤与结果讨论

面向企业的联邦学习中客户端多为企业，其数量较少，本文威胁模型建立在面向企业的联邦学习场景下，因此实验中设置 10 个客户端。考虑各客户端数据集为独立同分布和非独立同分布的情况：数据集为独立同分布时模型收敛速度很快，对 CNNMnist 和 AlexNet 模型，模拟 10 轮联邦学习训练；客户端数据集为非独立同分布时，对于 CNNMnist 模型模拟 300 轮训练。在每轮联邦学习训练中，实验步骤如下。① 从表 2 所述 10 个样本中选择 1 个恶意软件。② 选择一个客户端模

表 2 恶意软件样本信息

Table 2 Malware Sample Information

序号	md5	大小	类型	序号	md5	大小	类型
1	2a95b033	14 kB	PSS	6	5b26880f	79 kB	EXE
2	602bc1de	150 kB	PDF	7	fe8938f0	3.4 MB	EXE
3	16ba8f5d	213 kB	DOC	8	9b748436	368 kB	EXE
4	b855b1b7	456 kB	EXE	9	455ca106	6.0 MB	DLL
5	08eb8f2e	3.6 MB	EXE	10	43a74748	105 kB	DLL

表4 检测器性能指标
Table 4 Detector performance metrics

指标	描述	计算公式	指标	描述	计算公式
准确率	被正确预测的正负样本 占所有样本的比例	$ACC = \frac{TP+TN}{TP+TN+FP+FN}$	误报率	负样本被预测为正样本占 总负样本的比例	$FPR = \frac{FP}{TN+FP}$
精确率	正样本被预测为正样本 占所有被预测为正样本 的比例	$Precision = \frac{TP}{TP+FP}$	漏报率	正样本被预测为负样本占 总正样本的比例	$FNR = \frac{FN}{TP+FN}$
召回率/真正率	正样本被预测为正样本 占总正样本的比例	$Recall = TPR = \frac{TP}{TP+FN}$			

型, 将步骤①选择的恶意软件嵌入其中, 即每次 10 个客户端模型中只有 1 个是恶意代码模型。③ 使用本文提出的检测方法对所有客户端模型进行检测。以上实验步骤重复进行直到遍历所有恶意软件样本和客户端模型, 统计所有 TP、FP、TN 和 FN, 然后计算各项指标。如未特殊声明, 下面实验结果均为 10 轮联邦学习训练检测结果的平均值。

4.2.1 恶意代码嵌入对模型精度的影响

Wang 等^[15]提出的快速替换方法能够在几乎不影响模型精度的情况下嵌入大体积的恶意软件。嵌入率越高, 对模型的影响越大, 所以本文只展示在嵌入率为 100% 时恶意代码嵌入对模型精度影响的实验结果。通过计算恶意代码嵌入前后模型测试准确率的差值来评估恶意代码嵌入对模型精度的影响程度, 差值越大, 代表恶意代码嵌入操作对模型精度的影响越大。通过实验发现, CNNMnist 和 AlexNet 模型在数据集为独立同分布和非独立同分布的实验结果相似, 因此图 9 只展示 CNNMnist 模型在数据集为独立同分布时的实验结果, 可以发现在恶意软件嵌入前后模型的测试准确率变化不大, 其中最大值为 0.008 2, 最小值为 -0.001 7, 负值代表在嵌入恶意软件后, 模型的测试准确率上升。

4.2.2 阈值的选取

检测方法的有效性依赖于阈值的选取, 阈值过大会导致漏报率增加, 较难实现有效检测, 阈值过小会导致误报率增加。本节通过实验为检测方法选择合适的阈值, 将测试取自 $(k/2, k-1)$ 区间的所有阈值 t 。恶意软件检测方法的主要目的是获得较高的真正率以及较低的误报率^[24], 此外漏报率也是衡量恶意软件检测方法的重要指标, 因此

根据这三项指标来选取合适的阈值。实验结果如图 10 所示, CNNMnist 模型在数据集为独立同分布与非独立同分布时 (恶意代码嵌入率为 10%), 阈值越大, 检测方法的真正率和误报率越低, 漏报率越高。阈值为 5 时检测方法的真正率和漏报率达到最佳, 虽然此时的误报率最高, 但是仅有 0.8%, 因此在后续实验中如未特殊声明, 检测方法的阈值均设为 5。

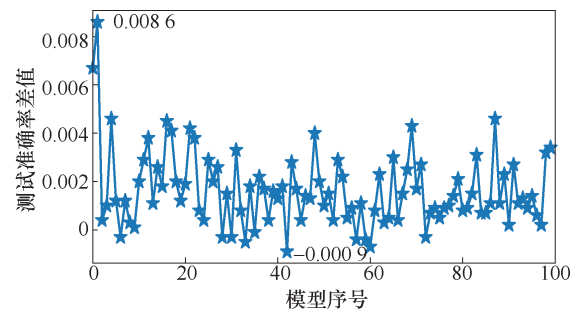


图9 模型在恶意代码嵌入前后的测试准确率差值
(嵌入率为 100%)

Figure 9 The model test accuracy difference before and after malicious code embedding (the embedding rate is 100%)

4.2.3 方法有效性验证

CNNMnist 模型在数据集为独立同分布与非独立同分布情况下, 各项指标在不同嵌入率下的实验结果如图 11 所示, 在两种情况下模型的检测性能随着嵌入率增加都逐渐增加。表 5 为 AlexNet 模型在恶意代码嵌入率为 0.5%、1% 和 5% 时的实验结果, 即使很低的嵌入率, 检测方法也能得到很好的性能指标, 这是因为 AlexNet 模型的嵌入容量很大 (75 MB), 即使 1% 的嵌入率, 也会在模型中嵌入 0.75 MB 大小的恶意软件, 这会严重降低恶意代码模型与诚实模型之间的相似度, 本文检测方法能很好地捕捉到这一点。

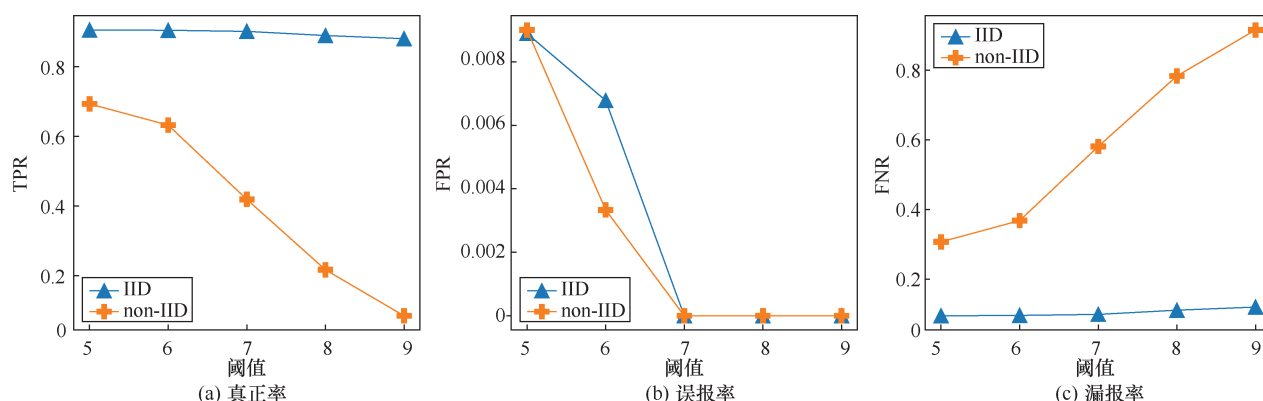


图 10 嵌入率为 10% 时, 不同阈值对应的真正率、误报率和漏报率

Figure 10 The true rate, false alarm rate and false negative rate corresponding to different thresholds when the embedding rate is 10%

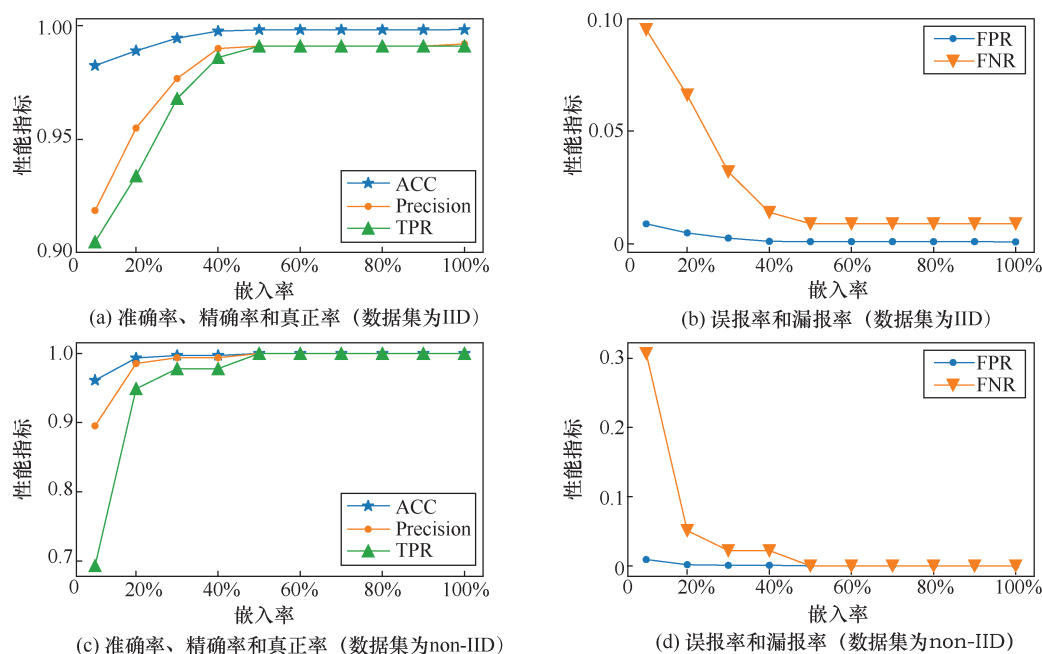


图 11 不同嵌入率下的各项指标

Figure 11 The metrics under different embedding rates

表 5 AlexNet 模型在数据集为 IID 时的各项指标
Table 5 Indicators of AlexNet model when the dataset is IID

指标	嵌入率		
	0.5%	1%	5%
准确率	96.63%	99.32%	100%
精确率	83.32%	96.6%	100%
真正率	82.9%	96.6%	100%
误报率	1.8%	0.38%	0
漏报率	17.1%	3.4%	0

4.2.4 模型聚合轮数对检测方法的影响

无论客户端数据集为独立同分布还是非独立

同分布, 各客户端模型之间的相似度会随着模型聚合轮数的增加而升高, 因此检测方法的性能会持续增加。如图 12 和图 13 所示, CNNMnist 模型在数据集为独立同分布和非独立同分布时, 检测方法真正率、误报率和漏报率在不同聚合轮数 (也称通信轮数) 下的实验结果能够证明这一点。同时发现, 当数据集为非独立同分布且恶意代码嵌入率为 10% 时, 检测方法的各项指标随通信轮数波动, 但总趋势是在逐渐改善的, 而当嵌入率增加到 20% 时, 检测方法能够保持在较高的性能。

4.2.5 检测方法通用性验证

在高级恶意软件活动中, 攻击者会通过加密

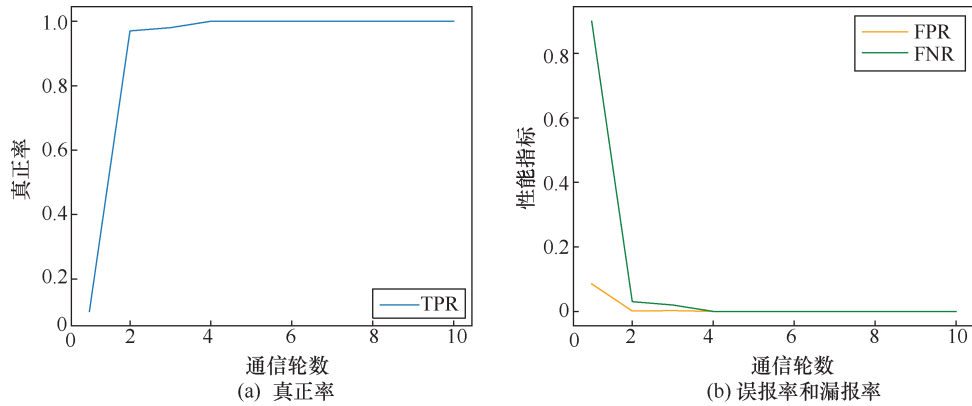


图 12 数据集为 iid, 嵌入率为 10% 时, 不同通信轮数对应的各项指标

Figure 12 The metrics of different communication rounds when the data set is iid and the embedding rate is 10%

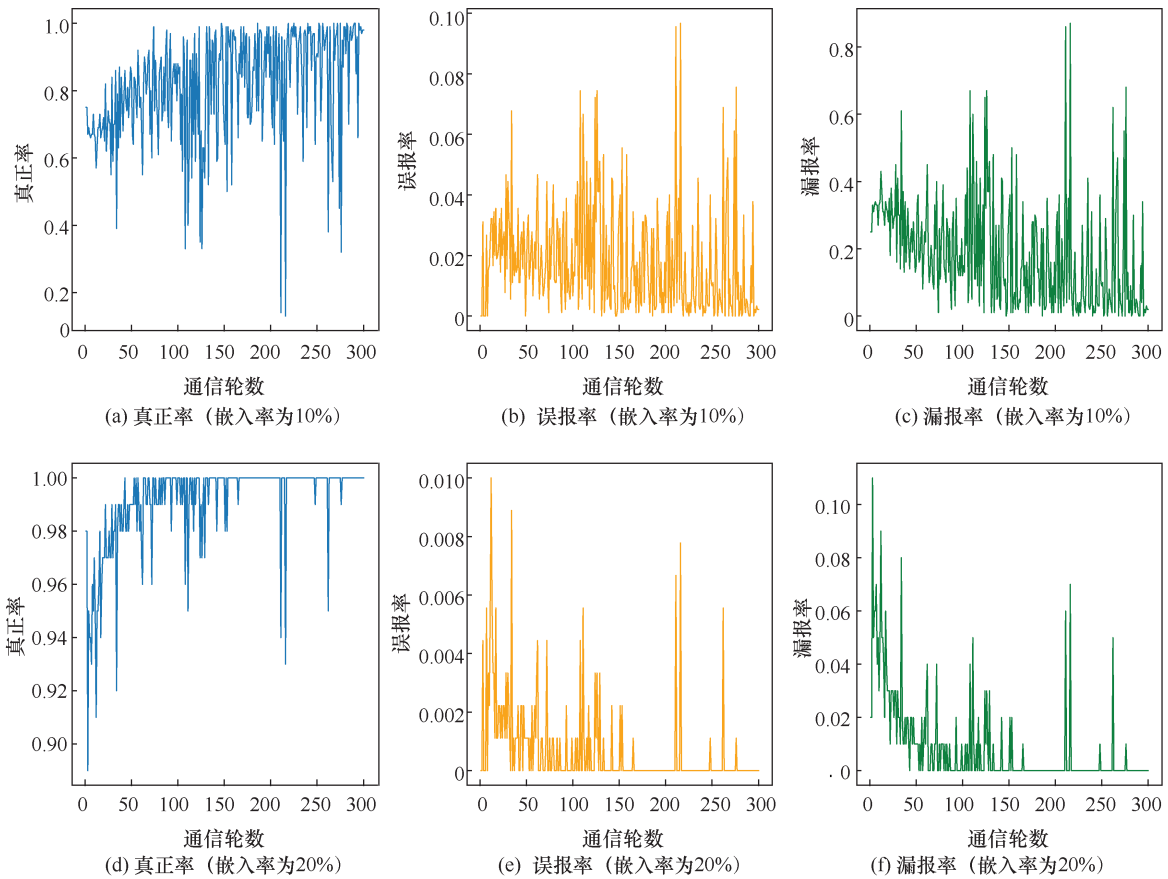


图 13 数据集为 non-IID, 不同通信轮数的各项指标

Figure 13 The metrics of different communication rounds when the data set is non-IID

的方式绕过基于特征码的恶意软件检测。因此在实验中, 将恶意代码与随机密钥进行异或后再嵌入模型中, 以验证恶意代码被加密时检测方法的有效性。除了恶意软件在嵌入前被加密之外, 其余实验设置与 4.2.3 节相同。图 14 的实验结果表明, 即使恶意代码被加密, 本文检测方法仍有较高的检测性能。

4.2.6 多攻击者的检测

这里将实验分为攻击者数量已知和未知两种情况。在已知攻击者数量时 (设为 N_A), 将阈值设为 $(k - N_A) / 2$, 即至少有一半的诚实模型与某个模型的相似度都最低时, 将其判定为恶意代码模型。CNNMnist 模型在数据集为独立同分布情

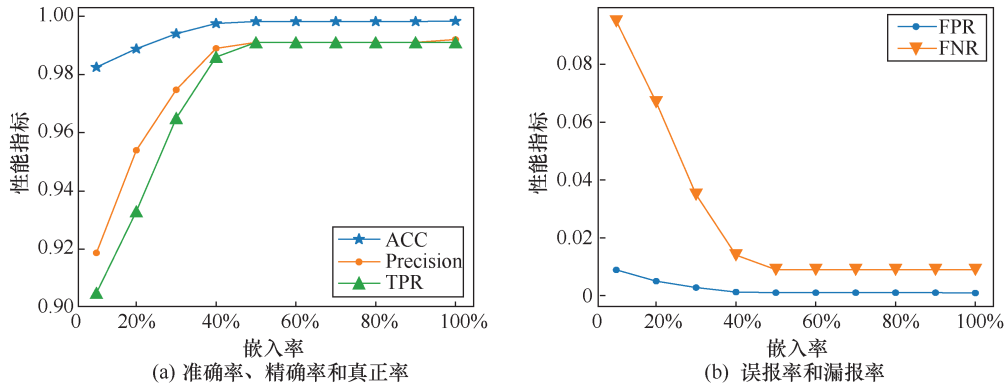


图 14 恶意软件被加密时不同嵌入率下的各项指标

Figure 14 The metrics under different embedding rates when malware is encrypted

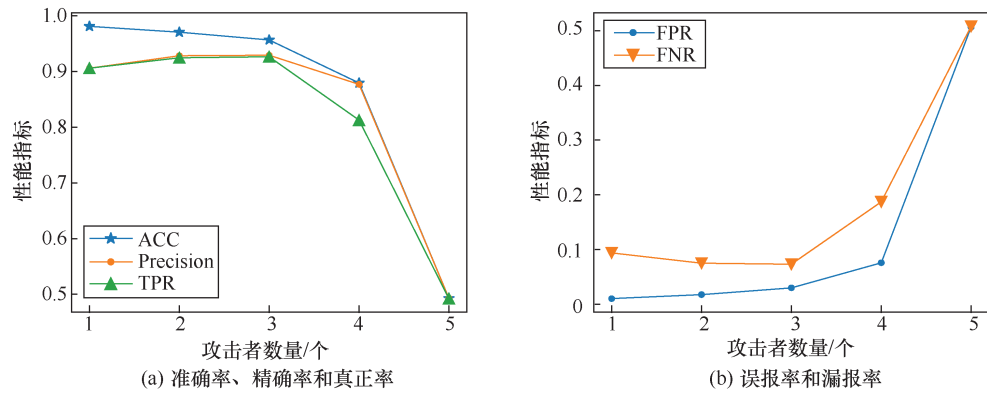


图 15 不同攻击者数量下的各项指标

Figure 15 The metrics under different number of attackers

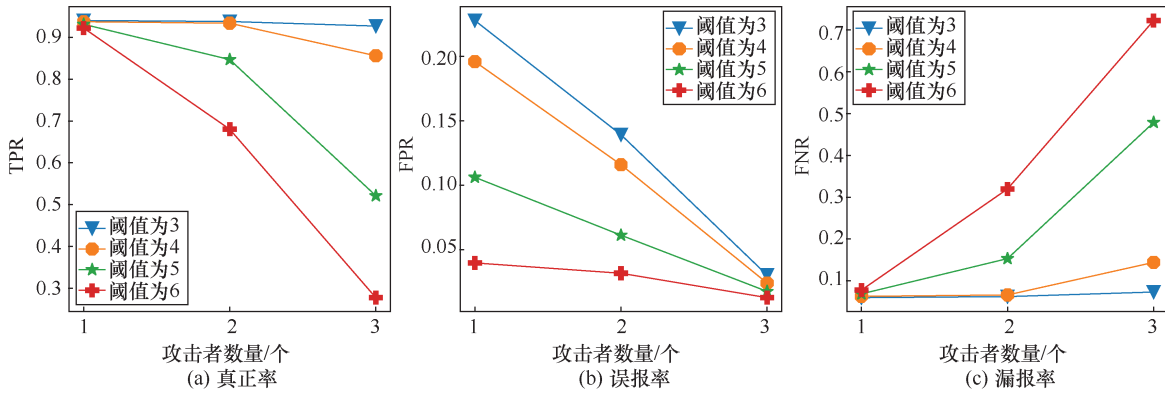


图 16 不同攻击者数量在不同阈值下的各项指标

Figure 16 The metrics under different number of attackers in different thresholds

况下, 各项指标在不同攻击者数量下的实验结果如图 15 所示, 可以看到, 随着攻击者数量的增加, 检测方法的性能呈下降趋势, 然而在真实的面向企业的横向联邦学习场景中, 各参与方相互独立, 攻击者能够控制多个参与方的概率较低。

在攻击者数量未知的情况下, 假设攻击者最多能够控制 3 个参与方。如图 16 所示, 与 4.2.2 节

中实验结果相同, 阈值越低, 检测方法的真正率越高, 漏报率越低, 在阈值为 3 时, 实际攻击者数量为 1~3 时的真正率和漏报率都是最佳的, 虽然此时误报率最高, 但是在联邦学习中, 检测方法即使发生误报将诚实模型排除在参数聚合之外, 对最终得到的全局模型精度影响不是很大, 如 CNNMnist 模型在数据集独立同分布情况下, 即使每轮只有 5 个客户端参与模型聚合, 10 轮联

邦学习获得的全局模型测试准确率依然可以达到98.74%，因此在实际场景中可以将检测方法的阈值设置得低一些。

5 结束语

本文提出了一种基于模型相似度的模型恶意代码夹带检测方法，通过大量的实验证明，该方法能够有效检测夹带恶意代码的机器学习模型。实验结果表明，提出的检测方法拥有高真正率、低误报率和漏报率。不足之处在于该检测方法依赖于多个彼此相似的客户端模型，只适用于在联邦学习的中心服务器端进行检测。此外，在联邦学习各参与方数据集为非独立同分布且恶意代码嵌入率较低时，检测方法的性能指标较低，且攻击者数量较多时检测方法的有效性有待提高。

参考文献:

- [1] MCMAHAN B, MOORE E, RAMAGE D, et al. Communication-efficient learning of deep networks from decentralized data[C]//Artificial Intelligence and Statistics (AISTATS). 2017: 1273-1282.
- [2] FREDRIKSON M, JHA S, RISTENPART T. Model inversion attacks that exploit confidence information and basic countermeasures[C]//Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS). 2015: 1322-1333.
- [3] HITAJ B, ATENIESE G, PEREZ-CRUZ F. Deep models under the GAN: information leakage from collaborative deep learning[C]//Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS). 2017: 603-618.
- [4] SHOKRI R, STRONATI M, SONG C, et al. Membership inference attacks against machine learning models[C]//2017 IEEE Symposium on Security and Privacy (S&P). 2017: 3-18.
- [5] TRAMÈR F, ZHANG F, JUELS A, et al. Stealing machine learning models via prediction {APIs}[C]//25th USENIX Security Symposium (USENIX Security 16). 2016: 601-618.
- [6] GU T, DOLAN-GAVITT B, GARG S. Badnets: identifying vulnerabilities in the machine learning model supply chain[J]. arXiv Preprint arXiv:1708.06733, 2017.
- [7] JI Y, ZHANG X, WANG T. Backdoor attacks against learning systems[C]//2017 IEEE Conference on Communications and Network Security. 2017: 1-9.
- [8] BAGDASARYAN E, VEIT A, HUA Y, et al. How to backdoor federated learning[C]//International Conference on Artificial Intelligence and Statistics (AISTATS 2020). 2020: 2938-2948.
- [9] JI Y, ZHANG X, JI S, et al. Model-reuse attacks on deep learning systems[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS 2018). 2018: 349-363.
- [10] YAO Y, LI H, ZHENG H, et al. Regula sub-rosa: latent backdoor attacks on deep neural networks[J]. arXiv Preprint arXiv: 1905.10447, 2019.
- [11] YAO Y, LI H, ZHENG H, et al. Latent backdoor attacks on deep neural networks[C]//Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019). 2019: 2041-2055.
- [12] BHAGOJI A N, CHAKRABORTY S, MITTAL P, et al. Analyzing federated learning through an adversarial lens[C]//International Conference on Machine Learning (ICML). 2019: 634-643.
- [13] CARLINI N, LIU C, ERLINGSSON Ú, et al. The secret sharer: evaluating and testing unintended memorization in neural networks[C]//28th USENIX Security Symposium (USENIX Security 19). 2019: 267-284.
- [14] LIU T, LIU Z, LIU Q, et al. StegoNet: turn deep neural network into a stegomalware[C]//Annual Computer Security Applications Conference (ACSAC'20). 2020: 928-938.
- [15] WANG Z, LIU C, CUI X. EvilModel: hiding malware inside of neural network models[C]//2021 IEEE Symposium on Computers and Communications (ISCC). 2021: 1-7.
- [16] WANG D G, SUN Y, ZHOU C X. A covert communication method based on gradient model[C]//2021 IEEE 6th International Conference on Signal and Image Processing (ICSIP). 2021: 926-930.
- [17] 周传鑫, 孙奕, 汪德刚, 等. 联邦学习研究综述[J]. 网络与信息安全学报, 2021, 7(5): 77-92.
- [17] ZHOU C X, SUN Y, WANG D G, et al. Survey of federated learning research[J]. Chinese Journal of Network and Information Security, 2021, 7(5): 77-92.
- [18] LYU L, YU H, ZHAO J, et al. Threats to federated learning[M]//Federated Learning. Berlin: Springer, 2020: 3-16.
- [19] PROVOS N, HONEYMAN P. Hide and seek: an introduction to steganography[J]. IEEE Security & Privacy, 2003, 1(3): 32-44.
- [20] CHAN C K, CHENG L M. Hiding data in images by simple LSB substitution[J]. Pattern Recognition, 2004, 37(3): 469-474.
- [21] KAHAN W. IEEE standard 754 for binary floating-point arithmetic[J]. Lecture Notes on the Status of IEEE, 1996: 11.
- [22] FRIDRICH J, GOLJAN M, DU R. Reliable detection of LSB steganography in color and grayscale images[C]//Proceedings of the 2001 Workshop on Multimedia and Security: New Challenges (MM&Sec). 2001: 27-30.

[23] InQuest malware-samples[EB]. 2022.

[24] 范铭, 刘烜, 刘均, 等. 安卓恶意软件检测方法综述[J]. 中国科学: 信息科学, 2020, 50(8): 1148-1177.

FAN M, LIU T, LIU J, et al. Android malware detection: a survey[J]. Scientia Sinica Informationis, 2020, 50(8): 1148-1177.

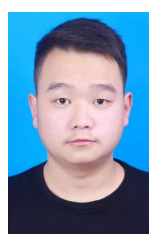


周传鑫(1997-), 男, 安徽蚌埠人, 信息工程大学硕士生, 主要研究方向为数据安全交换、机器学习隐私保护。

[作者简介]



汪德刚(1996-), 男, 陕西安康人, 信息工程大学硕士生, 主要研究方向为数据安全交换、恶意检测。



高琦(1997-), 男, 湖北襄阳人, 信息工程大学硕士生, 主要研究方向为机器学习隐私保护。



孙奕(1979-), 女, 河南郑州人, 博士, 信息工程大学副教授, 主要研究方向为网络与信息安全、数据安全交换。



杨帆(1996-), 女, 天津人, 信息工程大学硕士生, 主要研究方向为隐私计算、可验证计算。