基于深度迁移学习的智能合约漏洞检测

一、说明

随着区块链技术的快速发展，智能合约作为去中心化应用的重要组成部分，越来越受到关注。然而，智能合约的安全性问题也日益突出，其漏洞可能导致巨大的经济损失。本技术报告提出一种基于深度迁移学习的方法，旨在有效且高效地检测以太坊智能合约中的多种漏洞。

本技术方法使用一个通用的特征提取器来学习智能合约的通用字节码语义，并分别学习每个漏洞类型的特征来训练可迁移学习的多标签分类器，使其可同时检测出智能合约的多个漏洞。与以往的检测方法相比，本方法可以通过迁移学习轻松扩展到数据有限的新漏洞类型。当新的漏洞类型出现时，本方法向训练好的特征提取器添加一个新的分支，并用有限的数据对其进行训练，由此将模型修改和重新训练开销降至最低。与现有的非机器学习工具相比，本方法可以应用于任意复杂度的智能合约，并确保100%的智能合约覆盖率。此外，本技术方法使用单个统一的技术框架实现了对多种漏洞类型的并发检测，大幅度缩短检测时间。

本技术报告针对现有检测方法的不足，提出了一种基于深度迁移学习的检测方法，该方法能通过有限数据进行迁移学习新的漏洞类型。与现有的智能合约漏洞检测工具相比，本技术方法具有以下优势：

* 操作比源代码更易访问的合约字节码
* 通过有效的迁移学习，用有限的数据检测新的漏洞类型
* 能够在一次扫描中快速并发检测多个漏洞类型

二、实验

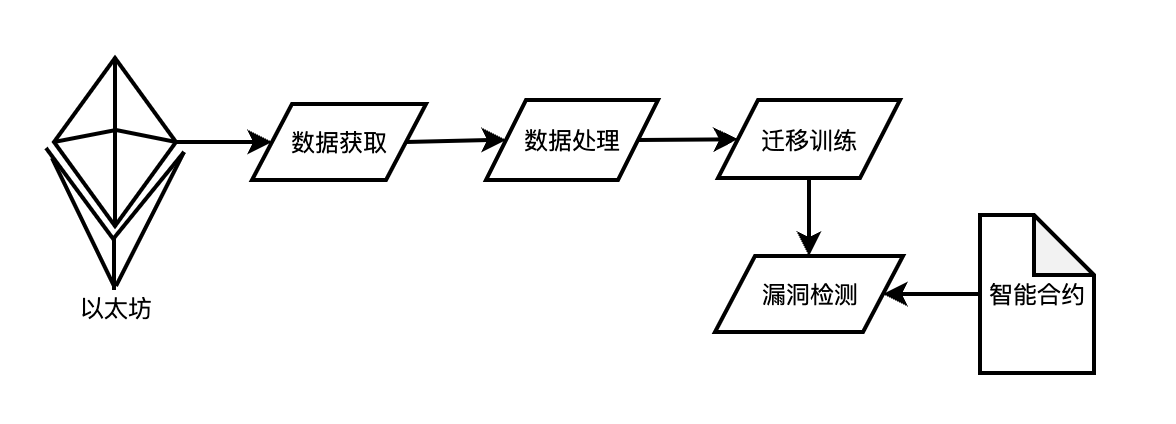
本方法的技术框架如图1所示。通过谷歌的bigquery查询到以太坊的公开数据集，获取智能合约的字节码数据，对字节码数据进行数据处理，将处理后的数据使用深度迁移学习技术训练模型，最后使用训练完的模型检测智能合约的漏洞类型。

图1

本方法的技术核心在于数据处理和深度学习架构，主要包括以下几个部分：

1. 数据处理

考虑到智能合约可能为空或自毁的情况，本方法过滤掉获取的空字节码0x。字节码由十六进制数字组成，表示特定的操作序列和参数，考虑到之后模型训练的性能问题，本方法将字节码长度限制在17.5k以内。本技术首先将收集的原始字节码转换为由唯一分隔符划分的操作序列，并从字节码中删除输入参数以减小输入大小。此外，将具有相同功能的操作合并为一个通用操作——命令PUSH1-PUSH32（由字节0x60-0x7f表示）被PUSH操作（由0x60表示）取代。如果已获取的字节码中的一些十六进制数字与以太坊黄皮书中定义的任何操作都不对应，则这些字节被视为无效操作，并用值XX代替。此操作合并步骤可能会将字节码映射到相同的预处理字节码。本方法还对经过数据预处理后的字节码数据集进行了重复数据消除，最后剩余15893个字节码。由于操作码足以捕获语义，因此本方法只对操作码进行建模，而忽略操作数。

本技术使用人工标记的方法对预处理数据集进行漏洞标记以保证准确性，一共标记出8个漏洞类型，如表1所示。

|  |  |  |
| --- | --- | --- |
| 漏洞类型 | **SWC-ID** | 描述 |
| ARTHM | SWC-101 | 整数溢出和下溢 |
| DOS | SWC-113, SWC-128 | 拒绝服务攻击 |
| LE | - | 被锁住的以太币 |
| RENT | SWC-107 | 重入攻击 |
| TimeM | SWC-116 | 时间操纵（使用区块值作为时间代理） |
| TimeO | SWC-114 | 时间戳排序（交易顺序依赖） |
| Tx-Origin | SWC-115 | 通过 tx.origin 进行授权 |
| UE | SWC-104 | 未处理的异常 (未检查的调用返回值) |

表1

1. 深度学习架构

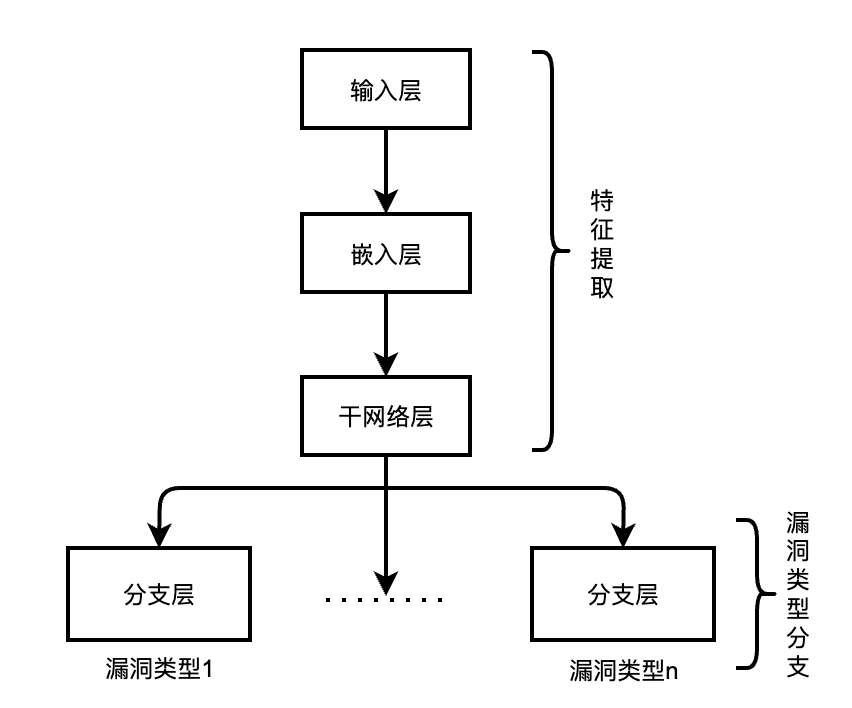
本技术方法采用可扩展DNN模型进行训练，具体模型结构如图2所示。

图2

* 特征提取：可扩展 DNN 模型的第一个组件是所有底层分支共享的公共特征提取器(即“干网络层”)。特征提取器是一堆层，其目的是学习输入数据中的基本特征，这些特征在不同的属性之间是通用的和有用的。基于智能合约，对特征提取器进行训练，从合约的字节码中学习语义和句法信息。为此，本方法在特征提取器中加入了几个关键层：
  + 嵌入层：智能合约的字节码是长的十六进制数，而 DNN 通常使用小数来实现高精度。嵌入可以解决这种差异，因为它将单词嵌入存储在数值空间中，并使用索引检索它们。嵌入层提供了两个关键的好处: 一是通过线性映射压缩输入，从而降低特征维数；二是在嵌入空间中学习字节码。这有助于表示探索，并在彼此相邻的地方收集相似的字节码。本方法利用嵌入层的优势来捕获输入字节码中的语义。
  + GRU/LSTM层：图2中的主干层和分支层可以包括用于处理顺序输入的GRU/LSTM层。门控递归单元（GRU）和长短期记忆（LSTM）是递归神经网络中的两个典型层，有助于使用“门控”机制克服短期记忆约束和消失梯度问题。更具体地说，这两种类型的层都有内部门，它们沿着时间序列调节信息流，并决定哪些数据应该保留/忘记。在本方法的DNN设计中，主要使用GRU层。
* 漏洞分支：本方法的多输出DNN体系结构的第二个组成部分是多个漏洞分支的集合。每个分支都是经过训练以学习相应漏洞类型的隐藏表示层。虽然不同分支之间没有直接依赖关系，但它们共享相同的特征提取器，即每个分支的输入是相同的。因为分支输入（也是特征提取器的输出）应捕获合约字节码中的语义，这是对不同漏洞类型有用的通用信息。每个漏洞分支的最后一层是具有一个神经元的密集层。该神经元的sigmoid评估给出了输入合约具有特定漏洞的概率。因此，本方法通过为其诊断提供置信度得分，而不是关于漏洞存在的二元决策，从而产生具有更好解释性的检测结果。
* 迁移学习：当识别出新的漏洞时，本方法相应地构建新的训练数据集，并通过添加新的漏洞分支（即层堆栈）来更新聚合DNN检测器。在迁移学习过程中，公共特征提取器和现有漏洞分支的参数是固定的，只有新添加的分支中的参数才会使用新的漏洞数据集进行更新。冻结特征提取器和收敛分支确保更新的DNN分类器保持对旧漏洞的检测精度，同时训练新分支使更新的模型能够学习新攻击。

三、训练结果

本方法建立了一个可扩展DNN模型，它包含八个主要模型训练分支，然后扩展了五个新的分支用于迁移学习。多输出DNN神经网络，优势在于它能够利用输入数据中的共享特征，同时处理多个漏洞类型。模型可以通过学习不同漏洞类型之间的相似性和差异性来提高整体的检测性能，并能够同时检测多个漏洞类型。在这个阶段，本方法通过监督式学习来了解标记的字节码数据集中的漏洞。

使用表2中所示的超参数来训练上述实例化的 DNN。这些超参数是通过网格搜索找到的。在数据传递到模型的输入层之前，需要对字节码序列进行向量化。这是由标记器实现的，它将十六进制数据转换为数值向量。在标记化后，一个超参数的最大序列长度被应用到输入向量，本方法将其设定为5018。

BN (Batch Normalization) 层用于提高网络的训练速度和稳定性，通常被插入到网络的隐藏层或卷积层中，该层主要目的是通过对每个小批量数据进行归一化处理，使得网络的输入在训练过程中具有相似的分布，这有助于缓解由于网络层之间输入数据分布变化较大而导致的训练困难问题。

Dropout层是一种在深度学习神经网络中常用的正则化技术，旨在减少过拟合问题，它被应用于隐藏层，其作用是在训练过程中随机地将一部分神经元的输出置为零（断开连接），从而减少神经网络的复杂性。

Dense层（全连接层）中，所有输入神经元与输出神经元之间都有连接，因此每个输入神经元都与每个输出神经元相连。这意味着Dense层中的每个神经元都与前一层中的所有神经元相连接，形成了一个完全连接的图结构。这种连接方式使得Dense层能够捕捉到输入数据中的复杂关系和模式。Dense层的主要作用是将输入数据进行线性变换，并应用激活函数以引入非线性性，该层可以与其他类型的层（如卷积层、池化层等）交替使用，构建出复杂的神经网络结构。最后的输出层通常也是一个Dense层，用于进行分类、回归或其他任务的预测。

|  |  |  |
| --- | --- | --- |
| 层（类型） | 输出形状 | 参数 |
| input\_3 (InputLayer) | [(None,5018)] | 0 |
| embedding\_2 (Embedding) | (None, 5018, 60) | 4620 |
| gru\_2 (GRU) | (None, 32) | 9024 |
| batch\_normalization\_6 (BatchNormalization) | (None, 32) | 128 |
| dropout\_6 (Dropout) | (None, 32) | 0 |
| dense\_8 (Dense) | (None, 16) | 528 |
| dense\_10 (Dense) | (None, 16) | 528 |
| batch\_normalization\_7 (BatchNormalization) | (None, 16) | 64 |
| batch\_normalization\_8 (BatchNormalization) | (None, 16) | 64 |
| dropout\_7 (Dropout) | (None, 16) | 0 |
| dropout\_8 (Dropout) | (None, 16) | 0 |
| dense\_9 (Dense) | (None, 1) | 17 |
| dense\_11 (Dense) | (None, 1) | 17 |

表2

本方法对数据处理后的12714个字节码进行模型训练，并使用3179个测试集数据进行检测，模型检测速率为4/sec。具体的模型训练指标如表3所示。

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 评估指标 | 漏洞类型 | | | | | | | | |
| 准确率 | ARTHM | DOS | LE | TimeM | TimeO | UE | RENT | Tx-Origin | Safe |
| 0.3976 | 0.9201 | 0.9025 | 0.8610 | 0.9236 | 0.8912 | 0.6861 | 0.9984 | 0.3108 |

表3

三、结论

本技术报告中提供的方法能够为智能合约的安全检测提供一种新的思路，与非机器学习技术相比，本技术是一个统一的自动化框架，完全覆盖了具有任意复杂性的智能合约。此外，本技术通过一次运行提供了对各种漏洞的并发检测，因此与应用多个非机器学习工具相比，其部署开销要小得多。本技术方法的可扩展DNN设计具有高度模块化、可扩展性和高效性。