软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software [doi: 10.13328/j.cnki.jos.007003]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

MCL4DGA: 基于多视角对比学习的 DGA 域名检测方法*

王继虎[1], 刘子雁[2], 倪金超[2], 孔凡玉[1], 史玉良[1,3]

[1](山东大学 软件学院, 山东 济南 250101)

[2](国网山东省电力公司信息通信公司, 山东 济南 250021)

[3](山大地纬软件股份有限公司, 山东 济南 250200)

通信作者: 史玉良, E-mail: shiyuliang@sdu.edu.cn

摘　要: 在网络安全领域, 由域名生成算法 (domain generation algorithm, DGA) 产生的虚假域名被称为 DGA 域名. 与正常域名类似的是, DGA 域名通常是字母或数字的随机组合, 这使得 DGA 域名具有较强的伪装性. 网络黑客利用 DGA 域名的伪装性实施网络攻击, 以达到绕过安全检测的目的. 如何有效地对 DGA 域名进行检测, 进而维护信息系统安全, 成为当前的研究热点. 传统的统计机器学习检测方法需要人工构建域名字符特征集合. 然而, 人工或者半自动化方式构建的域名特征存在质量参差不齐的情况, 进而影响检测的准确性. 鉴于深度神经网络强大的特征自动化抽取和表示能力, 提出一种基于多视角对比学习的 DGA 域名检测方法 (MCL4DGA). 与现有方法不同的是, 所提方法结合了注意力神经网络、卷积神经网络和循环神经网络, 能够有效地捕获域名字符序列中的全局、局部和双向多视角特征依赖关系. 除此之外, 通过多视角表示向量之间的对比学习而产生的自监督信号, 能够增强模型的学习能力, 进而提高检测的准确性. 通过在真实数据集上与当前 DGA 域名检测方法实验对比验证了所提方法的有效性.

关键词: 网络安全; DGA 域名检测; 深度神经网络; 对比学习

中图法分类号: TP393

中文引用格式: 王继虎, 刘子雁, 倪金超, 孔凡玉, 史玉良. MCL4DGA: 基于多视角对比学习的DGA域名检测方法. 软件学报. http://www.jos.org.cn/1000-9825/7003.htm

英文引用格式: Wang JH, Liu ZY, Ni JC, Kong FY, Shi YL. MCL4DGA: DGA Domain Detection Method Based on Multi-view Contrastive Learning. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7003.htm

MCL4DGA: DGA Domain Detection Method Based on Multi-view Contrastive Learning

WANG Ji-Hu[1], LIU Zi-Yan[2], NI Jin-Chao[2], KONG Fan-Yu[1], SHI Yu-Liang[1,3]

[1](School of Software, Shandong University, Jinan 250101, China)

[2](Information and Telecommunication Company of State Grid Shandong Electric Power Company, Jinan 250021, China)

[3](Dareway Software Co. Ltd., Jinan 250200, China)

Abstract: In the field of cyber security, the mendacious domains generated by the domain generation algorithm (DGA) are called DGA domains. Similar to real domains, they are usually a random combination of characters or numbers, which makes DGA domains highly camouflaged. Hackers take advantage of the disguised nature of DGA domains to carry out cyber attacks, so as to bypass security detection. How to effectively detect DGA domains has become a research hotspot. Traditional statistical machine learning detection methods require the manual construction of domain feature sets. However, the quality of domain features constructed manually or semi-automatically varies, which affects the accuracy of detection. In view of the powerful automatic feature extraction and representation capability of deep neural networks, a DGA domain detection method based on multi-view contrastive learning (MCL4DGA) is proposed. Different from existing methods, it incorporates attentional neural networks, convolutional neural networks, and recurrent neural networks

* 基金项目: 山东省重点研发计划 (重大科技创新工程)(2021CXGC010103)
收稿时间: 2022-03-28; 修改时间: 2023-02-04, 2023-06-06; 采用时间: 2023-07-17; jos 在线出版时间: 2023-11-29

to effectively capture global, local, and bidirectional multi-view feature dependencies of domain sequences. Besides, the self-supervision signals derived by contrastive learning can enhance the expressiveness between multi-view feature learning encoders and thus improve the accuracy of detection. The effectiveness of the proposed method is verified by experimental comparison with current methods on a real dataset.

Key words: cyber security; domain generation algorithm (DGA) domain detection; deep neural network (DNN); contrastive learning (CL)

进入信息时代以来, 互联网为我们的生活提供了极大的便利. 然而, 互联网的普及为网络攻击创造了先天条件. 网络黑客通过大肆传播恶意程序, 实现窃取个人隐私、破坏系统程序的目的. 研究表明[1], 以恶意程序、勒索软件等为代表的网络攻击行为给各个国家的国防安全和群众的个人隐私造成了极大的威胁. 在恶意程序攻击过程中, 传统攻击方式利用硬编码的域名或 IP 地址直接与命令和控制 (command and control, C&C) 服务器连接, 接受来自 C&C 服务器的攻击操作指令. 然而, 这种硬编码的域名容易被网络安全员通过某种手段捕获, 进而切断恶意程序与 C&C 服务器之间的通信[2]. 近年来, 为了绕过网络安全管理员有针对性的防御, 网络攻击者利用域名生成算法 (domain generation algorithm, DGA)[3], 通过随机组合字符或者数字周期性地产生一系列虚假域名, 实现与 C&C 服务器的动态连接. 这种方式被称为基于 DGA 域名的攻击方法, 其工作流程如图 1 所示, 主要包括 5 步: (1) 由域名生成算法生成一系列 DGA 域名; (2) 攻击者提前进行域名注册并将其指向 C&C 服务器的 IP 地址; (3) 宿主机上的恶意程序产生相同的 DGA 域名列表; (4) 恶意程序按照生成的 DGA 域名列表依次请求域名系统 (domain name system, DNS) 服务器进行域名解析并返回解析得到的 C&C 服务器 IP 地址; (5) 根据返回的 IP 地址, 恶意程序与 C&C 服务器建立通信连接, 接受来自控制中心的蓄意破坏指令. 基于 DGA 域名的攻击方法以其隐蔽性能够轻易地逃避检测, 对企业信息安全和个人数据隐私造成极大威胁.

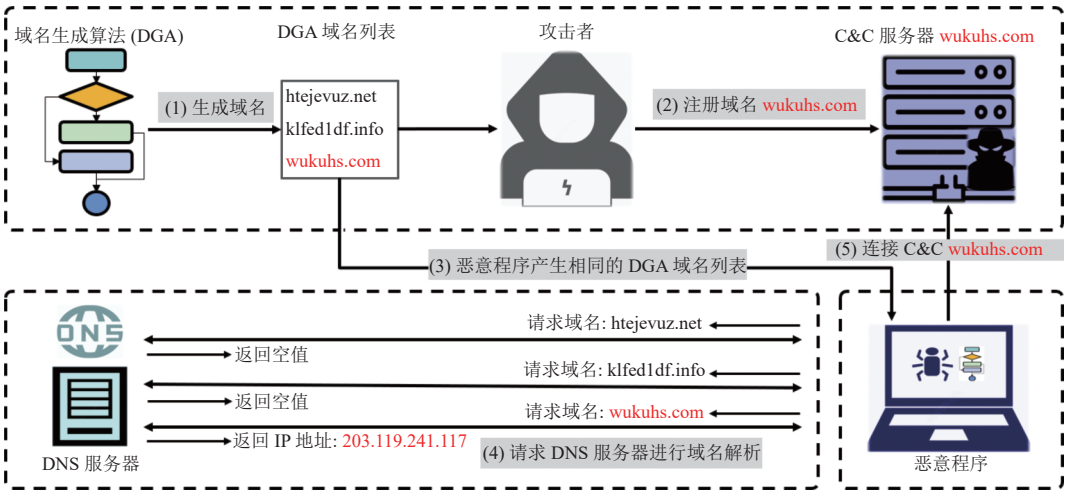


图 1 基于 DGA 域名的攻击方法示意图

DGA 域名检测是指利用相关方法和技术, 将虚假的 DGA 域名与正常域名区别开来, 进而阻断恶意程序与 C&C 服务器之间的通信, 避免恶意程序对宿主机器的破坏. 在众多实践中, 黑名单是一种直接的 DGA 域名检测方法, 该方法通过构建攻击域名黑名单的方式, 实现对 DGA 域名的被动拦截. 然而, 黑名单无法有效覆盖不断增加的海量虚假域名. 为了解决该问题, 研究人员提出了一系列基于特征工程的统计机器学习方法. 与黑名单检测方法不同的是, 基于特征工程的方法通过构建的域名字符特征集合, 训练机器学习检测模型, 实现对 DGA 域名的主动检测. 常用的域名字符特征包括 N 元模型 (N-Gram)[4]、语义学[5]、发音规则[6]、字符分布[7]和信息熵[8]等特征. 然而, 以手动或半自动化方式构建的域名字符特征集合存在耗时以及质量参差不齐等问题, 并且攻击者能够通过对特征集进行逆向工程, 升级域名生成算法, 轻易绕过检测, 进而影响检测的效率与准确性.

随着深度神经网络的发展, 基于深度学习的 DGA 域名检测方法逐渐代替基于特征工程的机器学习方法, 成

为 DGA 域名检测的主流解决方案. 相比于机器学习方法, 深度学习模型直接将域名字符序列作为输入, 以其强大的特征自动化抽取和表示能力, 构建 DGA 域名检测模型. 深度学习检测方法能够有效避免以手动或半自动化方式抽取的特征的质量对检测结果的扰动. 常用的深度学习检测方法包括循环神经网络、卷积神经网络等. 例如 Woodbridge 等人[9]首次提出了将长短期记忆网络 (long short-term memory networks, LSTM)[10]应用到 DGA 域名检测任务上. 随后, Yu 等人[11]利用一维卷积神经网络 (one-dimensional convolutional neural networks, 1DConv)[12]捕获域名字符序列中的字符依赖关系, 实现 DGA 域名检测. 最近, 研究人员提出将不同类型的神经网络组合在一起, 提高检测准确性, 例如 Highnam 等人[13]提出的结合 LSTM 和 1DConv 的 Bilbo 模型以及 Qiao 等人[14]提出的组合 LSTM 和注意力机制的 DGA 检测模型.

尽管基于深度学习的 DGA 检测方法取得了优异的检测效果. 然而, 随着虚假域名生成算法的不断升级, DGA 域名在形式上与正常域名越来越相似, 具有较强的欺骗性, 这为 DGA 域名检测任务带来了新的挑战. 第 3.4 节的实验结果表明, 现有双视角 DGA 检测模型 (如 LSTM+1DConv 和 LSTM+Attention), 无法捕获足够的域名特征信息, 进而导致模型准确性较差. 同时, 当缺乏足够的训练样本时, 监督学习信号较弱, 导致 DGA 检测模型训练不充分. 目前, 以对比学习 (contrastive learning, CL)[15,16]为代表的自监督学习在计算机视觉、自然语言处理以及推荐系统任务上取得了优异的表现, 它通过最小化不同增广视角的数据, 提高模型一致性信息表达能力, 进而实现自监督信号的提取. 如何有效地将对比学习应用到 DGA 域名检测任务上, 成为当前亟待解决的问题. 为此, 本文提出了一种基于多视角对比学习的 DGA 域名检测方法 (MCL4DGA). 该方法是一种端到端的深度学习模型, 利用不同的深度学习算法捕获不同视角下的域名字符序列特征, 随后通过不同视角下域名表示向量之间的对比学习, 产生额外的自监督信号, 进而提升模型的 DGA 域名检测准确性. 与现有检测方法不同的是, 本文利用不同神经网络算法实现域名字符序列中的全局、局部和双向特征捕获, 并将对比学习引入到 DGA 域名检测上, 为检测模型的训练拟合提供充足的监督信号. 在真实数据集上的仿真结果验证了本文方法的有效性.

本文第 1 节介绍 DGA 域名检测的相关方法和研究现状. 第 2 节介绍本文提出的基于多视角对比学习的 DGA 域名检测方法. 第 3 节通过对比实验验证所提方法的有效性. 最后总结全文.

1 DGA 域名检测相关工作

DGA 域名检测方法大致可分为两类: (1) 基于特征工程的机器学习检测方法; (2) 基于深度学习的检测方法. 本节首先对域名生成算法进行介绍, 随后对两类域名检测方法进行调研和分析, 最后对相关工作进行总结.

1.1 域名生成算法

作为黑客实施网络攻击的重要工具, 域名生成算法为其生成大量的虚假域名, 用于与 C&C 服务器建立通信连接, 实现网络攻击命令的发布与接受. 通常情况下, DGA 生成虚假域名需要借助随机种子, 随机种子可以是数值常数、时间戳、字符串等. 网络黑客与恶意程序共享相同的随机种子以便生成相同的 DGA 域名列表. 根据所使用的随机种子类型和生成算法不同, DGA 可分为多种类型[17]. 例如, Cryptolocker[18]是一种基于算数运算的域名生成算法. 它首先生成一连串的数值, 这些数值可能存在可用于域名映射的 ASCII 码, 或者指定一个或多个硬编码数组中的偏移量, 即可生成域名字符串. Dyre[19]是一种基于哈希算法的域名生成算法. 该类 DGA 使用哈希值的 16 进制表示法来产生虚假域名. 常用的虚假域名散列函数包括 MD5 和 SHA256. suppobox[20]是一种基于单词列表的域名生成算法. 该类算法通常采用单词随机组合策略生成虚假域名. 相比于其他类型的虚假域名, 基于单词列表随机组合的域名具有更强的欺骗性. 不同于以上分类标准, DGA 也可按照使用用途进行划分[21]. 例如, banjori[22]用于勒索网上银行用户的隐私信息. 而 ramnit 和 tinba 等[23]常用于窃取金融机构的隐私信息.

1.2 基于特征工程的机器学习检测方法

对于基于特征工程的机器学习检测方法来说, 构建一个高质量的域名字符特征集合是正确检测 DGA 域名的关键. 为此, Bilge 等人[24]从 DNS 数据中解析出时间、DNS 流量、生存时间、域名等相关特征, 然后基于这些特征构建了 J48 决策树模型, 用于 DGA 域名检测. Luo 等人[25]通过分析 10 万个正常域名的词法模式和发音规则, 抽

取出了字符分布模板、字符结构模板和发音特性等相关特征, 随后基于抽取的特征, 利用随机森林构建 DGA 域名检测分类器. Alenazi 等人[26]从 DGA 域名序列中抽取了 15 种词汇及其统计学特征, 并基于高斯朴素贝叶斯、决策树、随机森林等算法构建 DGA 域名检测模型. 实验结果表明, 相较于高斯朴素贝叶斯, 决策树和随机森林等算法取得了较好的检测结果. Yadav 等人[27]重点分析了 DGA 虚假域名与正常域名在字符分布上的差异性, 进而提出了多个评价准则, 包括 KL (Kullback-Leibler) 距离准则、JI (Jaccard index) 距离准则和编辑距离准则. 基于这些域名评价准则, 作者构建了 L1 正则化线性回归分类器用于检测 DGA 域名. 除此之外, Upadhyay 等人[28]额外构建了域名的互联网检索相关特征用于进一步提高检测的准确性. 具体地, 作者把待检测域名作为搜索关键字输入到搜索引擎 (例如 Google) 中, 然后统计在前 50 条互联网检索结果中输入域名的出现次数, 一般情况下正常域名在检索列表中的出现频率高于 DGA 虚假域名. Zhu 等人[29]提出了一种反馈支持向量机算法 (feedback support vector machine, F-SVM) 用于实现准确性更高的虚假域名检测. F-SVM 是一种改进的基于自反馈学习的 SVM 变体, 它借鉴了半监督学习和在线学习的思想, 用于解决训练样本不充分问题. da Silva 等人[30]利用梯度提升树算法 (extreme gradient boosting, XGBoost) 构建 DGA 域名检测模型. 相较于决策树、随机森林, XGBoost 是一种特殊的梯度提升决策树 (gradient boosting decison tree, GBDT), 其在对优化目标求解的时候使用了二阶导数的信息, 因此会使优化目标的梯度更加精确, 训练速度更快. 为了进一步提高 DGA 域名检测的准确性, 除了抽取域名的统计学特征之外, 研究人员还尝试构建了额外的辅助特征, 例如 WHOIS (who is) 特征[31]和 NXDomain (non-existent domain) 特征[32]等.

1.3 基于深度学习的检测方法

近年来, 深度学习在众多领域取得了突破性的进展. 因此, 一些研究者尝试将深度学习引入到 DGA 域名检测框架当中. 例如, Woodbridge 等人[9]首次利用 LSTM 神经网络算法实现 DGA 域名检测. 具体地, 作者首先对输入域名进行字符级别的分词获得域名字符序列, 然后利用字符嵌入神经网络将域名字符序列转换成向量序列. 随后, 作者将域名字符向量序列输入到 LSTM 算法中, 得到待检测域名的表示向量. 基于域名表示向量, 通过多层感知机获取待检测域名属于不同域名类别的概率分布, 最终实现 DGA 域名检测. 实验结果表明, 使用 LSTM 神经网络进行 DGA 域名检测是可行的和有效的. 为进一步提高模型性能, Qiao 等人[14]提出在 LSTM 算法的输出端增加注意力机制神经网络模块以区分域名字符序列中各个字符的重要性. 实验表明该方法优于纯 LSTM 检测模型. 除了以 LSTM 为代表的循环神经网络算法, 一维卷积神经网络算法 1DConv 以其强大的局部特征捕获能力, 在 DGA 域名检测任务中得到了广泛应用. 例如, Yu 等人[11]提出了基于 1DConv 神经网络的域名检测方法. 实验结果表明, 该方法在特定数据集上性能优于 LSTM 算法. DGA 域名按照生成算法的不同可分为不同的种类, 如 nymaim、ramnit、suppobox 等. 相比于采用字符随机组合策略生成的 DGA 域名 (例如采用 nymaim 生成的域名“mvkzsiggmp.org”和“frjvrkal.com”), 随机组合单词生成的域名 (例如 suppobox 算法生成的域名“nightchoose.net”和“decideperiod.net”) 更具有欺骗性, 因而更加难以检测和识别. 为了解决该问题, Highnam 等人[13]提出了一种 LSTM 与 1DConv 相结合的组合模型, 用于更准确地检测采用单词随机组合策略生成的 DGA 域名. 最近, 随着以对比学习为代表的自监督框架的提出, 越来越多的研究者尝试将其引入到序列表示学习任务中, 例如序列推荐、时序分析、文本表示等. 对比学习的引入能够更加充分地利用序列不同视角间的自监督信号, 提高模型准确性. 例如, Hu 等人[33]将对比学习引入到 DGA 域名检测模型中以提高检测准确性. 具体地, 他们首先利用基于孪生神经网络的 BiLSTM 获取输入域名序列中的嵌入特征向量, 同时将对比学习中的同类实例距离最小化和异类实例距离最大化的思想引入到特征抽取器中, 以提高特征抽取的准确性; 随后将抽取的域名特征向量输入到 SVM、随机森林、K-近邻分类器中, 得到对应的概率分布; 最后采用投票的方式确定最终的域名类别.

1.4 总　结

为了更加清晰的呈现各类方法的特点, 我们对相关工作进行了总结. 如表 1 所示, 尽管基于特征工程的 DGA 域名检测方法具有训练速度快、可解释性好等优点, 然而特征抽取质量以及严格的数据管控措施限制了该类方法的进一步发展. 相较于基于特征工程的检测方法, 基于深度学习的方法具有准确性高、训练效率高等优点, 然而数

据样本的不足往往会导致监督信号的缺乏, 进而导致深度学习模型无法得到充分的训练. 因此, 本文在现有双视角检测方法以及对比学习的研究基础上, 将两者结合起来, 提出了一种基于多视角对比学习的 DGA 域名检测方法 (MCL4DGA). 该方法通过多个信息捕获视角之间的对比学习, 生成额外的自监督信号, 以提高模型的表示学习能力, 进而提高域名检测的准确性.

表 1　DGA 域名检测方法总结

方法分类	主要算法	优点	缺点
基于特征工程的机器学习检测方法	高斯朴素贝叶斯、决策树、随机森林、线性回归、SVM、XGBoost等	(1) 机器学习检测模型训练速度快 (2) 利用各种特征工程技术能够挖掘域名序列中的有效信息, 并且模型检测结果具备一定的可解释性	(1) 特征抽取往往需要专业领域知识 (2) 特征质量对检测结果的扰动较大 (3) 严格的数据管控措施使部分特征无法获取到
基于深度学习的检测方法	LSTM神经网络、注意力机制、1DConv、LSTM+1DConv等	(1) 神经网络模型理论上可以拟合任意复杂映射函数, 因此深度学习检测方法准确性高 (2) 不需要额外构建域名特征集合, 提高训练效率	(1) 训练神经网络模型需要足够的训练样本使模型达到收敛状态 (2) 神经网络是一种黑箱模型, 其预测结果缺乏一定的可解释性

2　基于多视角对比学习的 DGA 域名检测方法 (MCL4DGA)

如图 2 所示, 本节将对基于多视角对比学习的 DGA 域名检测方法进行详细介绍.

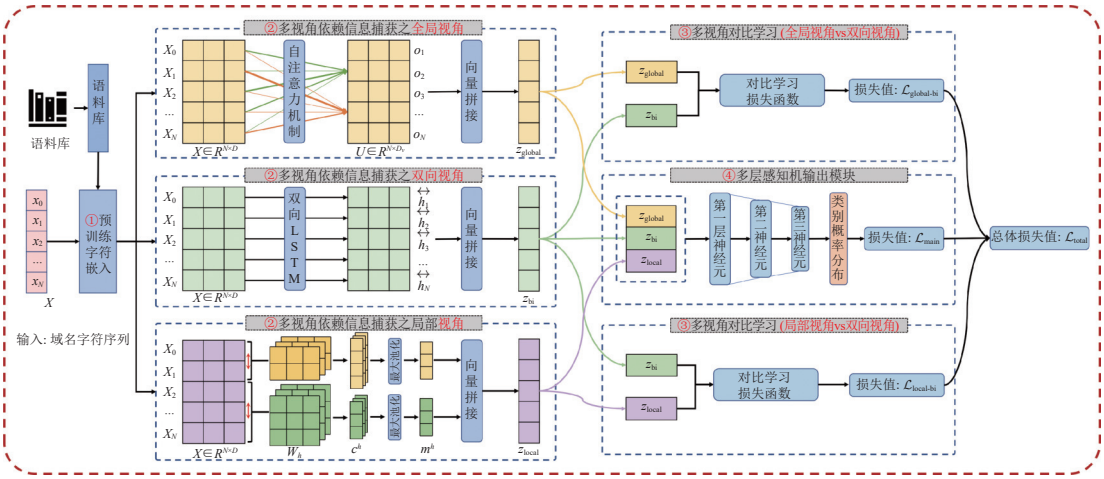


图 2　基于多视角对比学习的 DGA 域名检测方法

2.1　预训练字符嵌入

预训练词嵌入是一种将单词映射成密集表示向量的技术, 它通过在大型语料库上进行预训练, 将语义和词法先验知识固化到字符嵌入向量当中. 相对于随机初始化神经网络参数, 利用预训练的词嵌入向量作为模型的嵌入层可以提高模型的收敛速度和准确率. 常用的预训练单词嵌入方法包括 Word2Vec[34]、GloVe[35]、BERT[36]等. 然而, 单词预训练模型存在 OOV (out-of-vocabulary) 问题, 即预训练语料库无法覆盖所有的单词.

为了解决以上问题, 研究人员提出了预训练字符嵌入方法. 该方法以组成单词的字符为最小单位进行字符预训练. 该方法能够覆盖已知的所有字符, 因而不存在 OOV 问题. 例如, Boukkouri 等人[37]提出了一种基于 BERT 的预训练字符嵌入语言模型-CharacterBERT. 该模型首先将 1DConv 作用于单词的字符序列上, 以获取单词的表示向量, 然后将单词表示向量输入到 BERT 模型中进行预训练. 为了更好地将语义和词法先验知识嵌入到字符向量中, 该模型在 Wikipedia 和 OpenWebText 两种大型语料库上进行了预训练. 为了引入这种先验知识, 进而提高

DGA 域名检测的准确性, 将通过 CharacterBERT 预训练的字符嵌入向量迁移到本文提出的 DGA 检测模型中. 在模型训练阶段, 预训练字符嵌入向量根据本文定义的损失函数进行微调. 如图 2 所示, 假设输入的待检测域名字符序列为 $X=[x_1,x_2,\ldots,x_n,\ldots,x_N]$, 其中 x_n 表示域名中的第 n 个字符, N 表示字符序列长度. 通过查询 CharacterBERT 中的预训练字符嵌入表, 可得到预训练字符嵌入序列 $X=[x_1,x_2,\ldots,x_n,\ldots,x_N]\in\mathbb{R}^{N\times D}$, 其中 x_n 表示字符 x_n 的预训练嵌入向量, D 表示嵌入向量维度.

值得说明的是, 真实情况下待检测域名的长度往往是不一致的, 这种情况在序列表示学习相关任务 (如自然语言处理、时间序列分析等任务) 中是普遍存在的问题. 为了使模型能够批量处理长度不一致的输入序列, 目前常用的方法是对长短不一的序列进行填补 (padding) 操作. 具体过程为: (1) 首先对域名字符串进行分词操作 (tokenization), 即利用字符索引表将域名转化为字符索引序列; (2) 统计所有域名字符索引序列的长度, 得到序列的最大长度; (3) 对所有序列进行填补操作, 即把长度小于最大长度的字符索引序列用索引“0”进行填充, 使其长度统一为最大长度. 其中索引“0”表示占位符, 不参与模型训练, 即当序列输入到模型时, 字符索引为 0 的部分被视为无效信息.

2.2 多视角依赖信息捕获

2.2.1 全局视角依赖信息

通常情况下, 待检测域名中不同字符对检测结果具有不同的影响力. 如何自适应地捕获域名序列中各个字符对检测结果的相对重要程度, 是提高域名检测准确性的关键. 自注意力机制通过计算输入序列中各个元素的注意力权重来确定各元素对域名表示向量的贡献程度. 这种注意力权重是全局性的, 作用于整个输入序列. 因此, 针对域名字符序列, 本文把这种各个字符之间的相对重要性称为全局视角下的字符依赖信息. 利用自注意力机制捕获域名字符序列中的全局视角依赖信息过程如下.

如图 2 所示, 给定待检测域名的预训练嵌入向量序列 $X=[x_1,x_2,\ldots,x_n,\ldots,x_N]$, 采用自注意力机制中“查询 (query)-键 (key)-值 (value)”的形式将输入矩阵 $X\in\mathbb{R}^{N\times D}$ 映射到不同嵌入空间:

$$Q=XW_q \tag{1}$$

$$K=XW_k \tag{2}$$

$$V=XW_v \tag{3}$$

其中, $W_q\in\mathbb{R}^{D\times D_q}$, $W_k\in\mathbb{R}^{D\times D_k}$, $W_v\in\mathbb{R}^{D\times D_v}$ 为神经网络中的可训练参数矩阵; $D_q=D_k=D_v$ 表示神经元个数; $Q=[q_1,q_2,\ldots,q_n,\ldots,q_N]$, $K=[k_1,k_2,\ldots,k_n,\ldots,k_N]$, $V=[v_1,v_2,\ldots,v_n,\ldots,v_N]$ 分别为查询矩阵、键矩阵和值矩阵. 域名字符序列中各元素之间的注意力权重通过查询矩阵 Q 与键矩阵 K 之间的缩放点击产生, 即:

$$A=Softmax\left(\frac{QK^{\mathrm{T}}}{\sqrt{D_k}}\right) \tag{4}$$

$$O=AV \tag{5}$$

其中, $\sqrt{D_k}$ 表示缩放系数, 用于避免向量相乘导致元素值过大; $Softmax(\cdot)$ 表示 $Softmax$ 函数, 用于归一化注意力权重值; $A=[a_1,a_2,\ldots,a_n,\ldots,a_N]\in\mathbb{R}^{N\times N}$ 表示注意力权重矩阵. 将注意力权重矩阵 A 与值矩阵 V 相乘得到输出矩阵 O. 输出矩阵 O 可表示为向量序列的形式, 即 $O=[o_1,o_2,\ldots,o_n,\ldots,o_N]$. 其中, 第 n 个字符所对应的输出向量 o_n 可通过以下公式计算得到, 公式 (6) 和公式 (7) 分别是公式 (4) 和公式 (5) 的详细表示:

$$a_n^i=Softmax\left(q_nk_i^{\mathrm{T}}/\sqrt{D_k}\right)=\frac{\exp\left(q_nk_i^{\mathrm{T}}/\sqrt{D_k}\right)}{\sum_{j=1}^{N}\exp\left(q_nk_j^{\mathrm{T}}/\sqrt{D_k}\right)} \tag{6}$$

$$o_n=\sum_{i=1}^{N}a_n^iv_i \tag{7}$$

其中, $\exp(\cdot)$ 表示以自然常数 e 为底的指数函数; $a_n^i\in a_n$ 为注意力权重值, 表示第 i 个字符表示向量 x_i 相对于输出向量 o_n 的贡献度. 最终的域名表示向量可通过拼接所有的输出向量计算得到, 即:

$$z_{\text{global}} = o_1 \oplus o_2 \oplus \ldots \oplus o_N \tag{8}$$

其中, $\oplus$表示向量拼接操作, z_{global}表示由自注意力机制计算得到的最终的域名表示向量.

2.2.2 局部视角依赖信息

卷积神经网络 (convolutional neural networks, CNN) 常用于从图像中提取局部像素特征, 作为 CNN 的一种变体, 一维卷积神经网络 1DConv 以序列数据为输入, 利用不同尺寸的卷积核提取不同范围内的序列局部视角依赖信息. 借助一维卷积神经网络这一特性, 本文将 1DConv 应用于域名字符序列以提取局部视角下的字符间依赖信息. 除此之外, 通过改变卷积核的尺寸, 能够实现多尺度域名字符序列局部视角依赖信息提取.

如图 2 所示, 给定待检测域名的预训练嵌入向量序列$X = [x_1, x_2, \ldots, x_n, \ldots, x_N]$, 对其进行以下卷积操作:

$$c_i^h = ELU(W_h \cdot X_{[i:i+h-1]} + b) \tag{9}$$

其中, $X_{[i:i+h-1]} = [X_i, X_{i+1}, \ldots, X_{i+h-1}] \in \mathbb{R}^{h \times D}$表示输入向量序列$X$的第$i$个滑动窗口内的子序列切片; $W_h \in \mathbb{R}^{h \times D}$表示卷积核, 其中$h$表示卷积核尺寸, 它用来控制局部视角依赖信息的感知范围; b表示可训练参数; 本文选用$ELU(\cdot)$作为激活函数, 因为当输入为负数时, 其收敛于某一常数的平滑性可以缓解梯度消失和爆炸问题; c_i^h表示第i个滑动窗口对应的输出特征值. 沿着序列方向逐步滑动卷积核, 可得到以下一维特征映射:

$$c^h = \left[c_1^h, c_2^h, \ldots, c_{N-h+1}^h\right] \in \mathbb{R}^{N-h+1} \tag{10}$$

将特征映射向量c^h输入到最大池化层, 以提取特征映射向量中的最大值作为在卷积核尺寸为h时的显著特征:

$$m^h = MaxPooling\left(c^h\right) \tag{11}$$

其中, $MaxPooling(\cdot)$表示最大池化操作, 即取输入向量中的最大值; m^h表示在卷积核尺寸为h时获取到的序列卷积显著特征. 如图 2 所示, 当堆积多个卷积核并重复以上步骤时, 得到显著特征向量$m^h = \left[m_1^h, m_2^h, \ldots, m_T^h\right]$, 其中$T$表示卷积核的个数. 为了提取待检测域名字符序列中的多尺度局部视角依赖信息, 本文使用不同尺寸的卷积核进行卷积操作. 表 2 展示了本文方法所选取的卷积核尺寸、卷积核个数以及对应输出的显著特征向量.

表 2 卷积核尺寸、卷积核个数以及对应输出的显著特征向量

卷积核尺寸	卷积核个数	显著特征向量
2	256	$m^2 \in \mathbb{R}^{256}$
3	256	$m^3 \in \mathbb{R}^{256}$
4	256	$m^4 \in \mathbb{R}^{256}$
5	256	$m^5 \in \mathbb{R}^{256}$

基于表 2 所示的在不同卷积核尺寸下得到的显著特征向量, 域名表示向量可通过向量拼接的形式得到:

$$z_{\text{local}} = m^2 \oplus m^3 \oplus m^4 \oplus m^5 \tag{12}$$

其中, z_{local}表示由一维卷积神经网络 1DConv 计算得到的域名表示向量, 该域名表示向量捕获了域名字符序列中的多尺度字符间局部依赖信息. 至此, 本文已获取域名全局视角下的表示向量z_{global}和局部视角下的表示向量z_{local}.

2.2.3 双向视角依赖信息

LSTM 是循环神经网络 (recurrent neural network, RNN) 的一种变体, 其中的门控机制解决了普通 RNN 算法在面对长序列时存在的梯度消失和爆炸问题. 双向 LSTM 包含前向 LSTM 和后向 LSTM, 其中前向 LSTM 用于捕获序列的正向依赖信息, 而后向 LSTM 用于捕获序列的反向依赖信息. 得益于双向 LSTM 具备的双向信息处理能力, 将其应用到 DGA 域名检测上, 以捕获隐藏在域名字符序列中的双向视角下的字符间依赖信息.

如图 2 所示, 给定第t个字符向量x_t和隐藏状态h_{t-1}, LSTM 通过以下公式计算当前位置的隐藏状态h_t:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{13}$$

$$f_t = \sigma\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{14}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{15}$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1}) \tag{16}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{17}$$

$$h_t = o_t \odot \tanh(c_t) \tag{18}$$

其中, i_t , f_t , o_t 分别为输入、遗忘和输出门, 用来控制 LSTM 单元中的信息流动; $W_{\{i,f,o\}} \in \mathbb{R}^{D_h \times D}$, $U_{\{i,f,o\}} \in \mathbb{R}^{D_h \times D}$, $b_{\{i,f,o\}} \in \mathbb{R}^{D_h}$ 表示神经网络中的可学习参数, 其中 D_h 为隐藏层神经元个数; c_t 表示单元状态, $\tilde{c}_t$ 为单元候选状态; $\odot$ 表示向量之间元素对应相乘; σ 表示激活函数; h_t 表示输出的当前位置隐藏状态.

如图 2 所示, 假设前向 LSTM 的输出隐藏状态表示为 $\overrightarrow{h}_t$, 后向 LSTM 的输出隐藏状态表示为 $\overleftarrow{h}_t$, 则双向 LSTM 的最终输出隐藏状态 $\overleftrightarrow{h}_t$ 表示为:

$$\overleftrightarrow{h}_t = \overrightarrow{h}_t \oplus \overleftarrow{h}_t \tag{19}$$

给定待检测域名的预训练嵌入向量序列 $X = [x_1, x_2, \ldots, x_n, \ldots, x_N]$, 则经过双向 LSTM 得到的输出向量序列表示为 $H = \left[\overleftrightarrow{h}_1, \overleftrightarrow{h}_2, \ldots, \overleftrightarrow{h}_N\right] \in \mathbb{R}^{N \times 2D_h}$, 将输出向量序列拼接, 得到最终的域名表示向量:

$$z_{bi} = \overleftrightarrow{h}_1 \oplus \overleftrightarrow{h}_2 \oplus \ldots \oplus \overleftrightarrow{h}_N \tag{20}$$

其中, z_{bi} 表示由双向 LSTM 计算得到的域名表示向量, 它捕获了域名字符序列中双向视角下的字符间依赖信息.

至此, 本文已经得到多视角下的域名表示向量, 即 (1) 捕获全局视角下字符间依赖信息的域名表示向量 z_{global} ; (2) 捕获局部视角下字符间依赖信息的域名表示向量 z_{local} ; (3) 捕获双向视角下字符间依赖信息的域名表示向量 z_{bi} . 随后将利用计算得到的多视角域名表示向量, 进行多视角对比学习, 以获取额外的自监督信号.

2.3 多视角对比学习

研究表明, 对比学习以其强大的自监督信号提取能力, 能够有效提升深度学习模型性能. 针对序列数据挖掘任务, 常见的对比学习范式是通过在序列数据上进行数据增广操作, 以获取不同视角下的增广序列, 然后利用共享神经网络编码模型处理增广序列, 进而得到不同视角下的序列表示向量, 最后基于得到的不同视角序列表示向量和对比学习损失函数进行自监督信号提取. 例如 CL4Rec 模型[38]对原始输入序列进行遮蔽、重排等序列增广操作以获取不同视角的表示向量. 然而, 直接改变原始序列内部结构这种对比学习范式可能无意间破坏原始序列中的内在固有模式. 为了解决该问题, 在 DGA 域名检测任务中, 本文提出利用不同的神经网络算法实现多视角序列表示向量的生成, 这种方式对原始序列是无损的, 即不会破坏序列中元素之间的固有顺序.

多视角对比学习过程如图 2 所示, 给定生成的多视角域名字符序列表示向量, 以双向视角作为对比基准, 分别构建“全局视角 vs. 双向视角”和“局部视角 vs. 双向视角”两种对比学习损失优化目标, 以产生更丰富多样的自监督信号. 本文选用 InfoNCE[39]作为对比学习损失函数, 把同一域名在不同视角下的表示向量作为损失函数的正例对, 把不同域名的表示向量作为负例对, 具体形式如下所示:

$$\mathcal{L}_{\text{global-bi}} = -\sum_{i=1}^{B} \log \frac{\exp\left(sim\left(W_g z^i_{\text{global}}, W_b z^i_{bi}\right)/\tau\right)}{\exp\left(sim\left(W_g z^i_{\text{global}}, W_b z^i_{bi}\right)/\tau\right) + \sum_{i \neq j} \exp\left(sim\left(W_g z^i_{\text{global}}, W_b z^j_{bi}\right)/\tau\right)} \tag{21}$$

$$\mathcal{L}_{\text{local-bi}} = -\sum_{i=1}^{B} \log \frac{\exp\left(sim\left(W_l z^i_{\text{local}}, W_b z^i_{bi}\right)/\tau\right)}{\exp\left(sim\left(W_l z^i_{\text{local}}, W_b z^i_{bi}\right)/\tau\right) + \sum_{i \neq j} \exp\left(sim\left(W_l z^i_{\text{local}}, W_b z^j_{bi}\right)/\tau\right)} \tag{22}$$

其中, $W_{\{g,l,b\}}$ 表示可学习的参数矩阵, 用于将对应向量嵌入到统一的对比学习向量空间; $sim(\cdot,\cdot)$ 表示余弦相似度函数. 余弦相似度是一种常用的空间向量相似性度量指标, 它具有计算效率高以及准确率高等优点, 常用于计算给定向量之间的空间距离. τ 表示温度系数; B 表示模型训练的批大小; $\mathcal{L}_{\text{global-bi}}$, $\mathcal{L}_{\text{local-bi}}$ 分别表示全局视角和双向视角之间以及局部视角和双向视角之间域名表示向量的对比损失. 如公式 (21) 和公式 (22) 所示, 对比学习损失函数的作用是最大化正例对之间的相似性同时最小化负例对之间的相似性, 使不同的域名表示向量在表示空间中呈现出聚集性分布. 因此, 对比学习作为一种辅助优化学习任务能够提供额外的自监督信号, 辅助模型训练. 多视角对比学习的核心思想在于, 通过最大化来自不同序列编码模型的域名表示向量之间的相似性, 去寻找同一域名经过

不同类型深度学习模型编码之后的共性特征信息.

温度系数 τ 是对比学习损失函数中的一个超参数, 用于控制对比学习效果. 通常情况下, 温度系数越小, 对比学习效果越强, 即正例对样本之间的距离越小, 而负例对样本之间的距离越大. 然而, 温度系数也不能太小, 容易使模型陷入过拟合状态. 本文在现有研究基础上, 借鉴文献 [40] 中的参数设置, 将温度系数 τ 设置为 0.08.

2.4 多层感知机输出模块

基于第 2.2 节多视角依赖信息捕获中计算得到的多视角下域名表示向量, 采用常用的多层感知机神经网络模型 (multi-layer perception, MLP) 预测待检测域名的具体类别, 计算过程可表示成如下形式:

$$z_0 = z_{\text{global}} \oplus z_{\text{local}} \oplus z_{\text{bi}} \tag{23}$$

$$z_1 = ELU(W_1 z_0 + b_1) \tag{24}$$

$$z_2 = ELU(W_2 z_1 + b_2) \tag{25}$$

$$z_3 = Softmax(W_3 z_2 + b_3) \tag{26}$$

如图 2 所示, 多层感知机输出模块由 4 层神经元组成: 输入层、第 1 层、第 2 层、输出层. 其中输入层的输入向量由多视角域名表示向量拼接而成, 即 $z_0 = z_{\text{global}} \oplus z_{\text{local}} \oplus z_{\text{bi}}$; $W_{l\in\{1,2,3\}} \in \mathbb{R}^{D_l \times D_{l-1}}$ 和 $b_{l\in\{1,2,3\}} \in \mathbb{R}^{D_l \times 1}$ 为多层感知机神经网络中的可训练参数, 其中 $D_{l\in\{1,2,3\}}$ 表示各层神经元个数; 输出层选用 $Softmax(\cdot)$ 作为激活函数, 其输出为归一化的域名类别概率分布向量 $z_3 \in \mathbb{R}^{D_3 \times 1}$, 其中 D_3 等于域名类别个数.

给定预测的域名类别概率分布向量 $y_{\text{pred}} = z_3$ 和真实的域名类别独热编码向量 $y_{\text{true}} \in \mathbb{R}^{D_3 \times 1}$, 本文采用交叉熵损失函数 (categorical cross-entropy loss) 计算 DGA 域名检测任务的损失值, 损失值定义如下:

$$\mathcal{L}_{\text{main}} = \sum_{i=1}^{D_3} \left(y_{\text{true}}^i \cdot \log\left(y_{\text{pred}}^i \right) \right) \tag{27}$$

2.5 模型训练

本文所提出的 DGA 域名检测方法是一种端到端的神经网络模型, 模型的损失值包含 3 部分: 主任务损失 $\mathcal{L}_{\text{main}}$、多视角对比损失 ($\mathcal{L}_{\text{global-bi}}$ 和 $\mathcal{L}_{\text{local-bi}}$). 采用联合优化策略优化该模型, 其总体损失定义如下:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{main}} + \alpha \mathcal{L}_{\text{global-bi}} + \beta \mathcal{L}_{\text{local-bi}} \tag{28}$$

其中, α 和 β 表示对比损失系数, 用来控制对比学习在模型优化过程中所占的比重; $\mathcal{L}_{\text{total}}$ 表示模型的联合损失函数. 本文采用 Adam[41]优化器优化神经网络模型 MCL4DGA 中的可训练参数.

为了防止模型出现过拟合情况, 采用 Dropout 策略[42]随机丢弃神经网络中的神经元. 同时, 在神经网络全连接层之后进行层归一化操作 (layer normalization)[43], 以加速模型收敛速度和防止模型出现梯度消失和爆炸.

3 实验分析

3.1 实验数据

本文采用公开的域名检测数据集 (https://github.com/Juhong-Namgung/Malicious-URL-and-DGA-Domain-Detection-using-Deep-Learning) 进行实验, 该数据集包含两部分域名: 正常域名和由域名生成算法产生的虚假域名 (DGA 域名). 其中正常域名来源于开源网站 Alexa (https://www.alexa.com/topsites), 该网站提供了一个常用域名的列表. 通常情况下, 该列表中的域名访问量较高, 域名具有特定的语义信息. 因此, 将在 2020 年 5 月 4 日收集的常用域名列表中的所有 603 387 个域名作为正常域名 (Alexa 域名). 虚假的 DGA 域名来源于 Bambenek Consulting OSINT (https://osint.bambenekconsulting.com/feeds/), 该网站维护了由 50 多种不同的域名生成算法 (如 banjori、tinba、post、ramnit 等) 生成的虚假域名数据. 按照数量从多到少, 选取前 20 种不同类别的共计 832 276 个虚假域名作为 DGA 域名. 表 3 给出了实验数据的详细统计情况. 该数据集共包含 1 435 663 个域名, 按照 8:1:1 的比例将该数据集随机划分为训练集、验证集和测试集. 依照惯例, 本实验在训练集上进行模型训练, 在验证集上进行超参数调优, 在测试集上进行模型有效性验证.

表 3 实验数据统计

二分类	多分类	个数	示例
正常域名	Alexa	603 387	baidu.com、google.com
DGA域名	banjori	439 223	kuxeepictom.com、bzsamen.com
	tinba	66 789	xqftlmhggyjb.com、ywhcremfwcdc.com
	post	66 000	lmhw166rbtr91chw2an7er7a2.org
	ramnit	64 605	serttrckasufp.com、iqumlymtijotbx.com
	qakbot	40 000	erutzutfxcctcckqylmfh.com、abgwsejrx.info
	necurs	32 768	slfviphasvxrnogwokf.pw、fdbiopidydvvxxyarh.tw
	murofet	28 520	ghmvmwioxxxlkhtx.biz、ovrrzqqmfrwqmskt.com
	bebloh	15 021	w4wqk2buhd.com、ogaqug4utalt12l.net
	simda	14 755	masilom.info、wedevaq.info
	ranbyus	13 200	jsvqnsptwbxpgb.me、nkgypstvpncauy.in
	pykspa	10 024	aaykpwiugkeq.biz、oazisgeoya.info
	dyre	7 998	oef1c6464ad288ee2f96d79bc38ea227ce.tk
	kraken	7 878	xzqmavvj.dyndns.org、jndrmrb.mooo.com
	Cryptolocker	6 000	ptyjwsefmtslk.org、kesgkclvsmyw.ru
	nymaim	6 000	rsduicauo.net、drznubhga.com
	locky	4 014	hxjxqnw.xyz、rlyxlwtfhxheusdq.su
	vawtrak	3 150	uvkpehcf.top、rwtipmxoxsv.com
	shifu	2 331	ourmptk.info、pmrlvtw.info
	ramdo	2 000	kucocasmcoqoisau.org、eiwygswseaukekme.org
	p2p	2 000	sfuzdcykbemhixluohfutzpqoqk.com

3.2 基准模型

本实验选择如下方法作为基准模型与本文方法进行对比, 其中 GNB[26], RF[24], GBDT[30]为基于特征工程的机器学习方法, LSTM[9], L+A[14], Bi+1D[13]为基于深度学习的检测方法, DGAD-SN[33]为混合检测方法.

● GNB: 高斯朴素贝叶斯 (Gaussian naive Bayes, GNB) 方法是贝叶斯方法的一种变体, 它假设所有特征值都服从高斯分布. 该算法常用于分类任务上.

● RF: 随机森林 (random forest, RF) 是一种常用于 DGA 域名检测的集成学习算法, 它通过并行地组合多个弱决策树分类器, 利用投票的方法, 使整体模型具有较高的准确度和泛化性能.

● GBDT: 梯度提升决策树 (gradient-boosted decision trees, GBDT) 也是一种常用的集成学习算法, 与随机森林不同的是, 它采用串行策略组合多个弱决策树分类器, 下游分类器学习上游分类器的累计误差.

● LSTM: 基于长短期记忆网络 (long short-term memory network, LSTM) 的 DGA 域名检测方法是一种常用的深度学习基准模型, 它较早地将 LSTM 应用到 DGA 域名检测任务上, 并取得了较好的检测结果.

● L+A: 该方法通过作用于 LSTM 隐层输出序列上的注意力机制 (attention), 区分域名中各个字符对检测结果的重要性, 以此来提高域名检测性能. 将该方法记为 L+A.

● Bi+1D: 该方法是一种最新提出的基于深度学习的 DGA 域名检测模型, 它通过堆叠多层双向 LSTM 和 1DConv 实现域名字符序列中的多尺度信息捕获. 将该方法记为 Bi+1D.

● DGAD-SN: 该方法是机器学习和深度学习的混合检测方法. 它通过基于孪生神经网络的 BiLSTM 输出特征之间的对比学习, 自动提取高质量域名特征. 随后构建基于机器学习的域名分类器.

对于基于特征工程的机器学习方法 (GNB, RF, GBDT), 本文构建了一个常用的 DGA 域名检测特征集, 包括字符特征 (2-gram 和 3-gram), 元音比例特征 (为了获得良好的可读性, 正常域名往往比 DGA 域名具有更少的元音), 域名长度特征 (如表 3 所示, DGA 域名普遍具有较长的长度).

3.3 实验设置

本文所提出的模型 MCL4DGA 的超参数设置如表 4 所示. 通常情况下, 对于深度学习模型的超参数选择, 目前还没有一种确定的、高效的参数寻优方式, 大多数依靠开发者经验或网格搜索法 (grid search) 进行参数选取. 网格搜索法是指定参数值的一种穷举搜索方法, 该方法将各个参数可能的取值进行排列组合, 然后将各参数组合用于模型训练, 最终选取实验结果最好的一组参数组合作为模型超参数取值. 在本文实验设置中, 我们结合两种参数设置方法确定模型的超参数. 对于超参数 B、lr、$dropout$、D_1、D_2, 我们参考以往模型搭建经验, 将超参数设置为经验值. 对于超参数 D、D_h、$D_{\{q,k,v\}}$、α、β, 我们采用网格搜索的方式确定参数数值. 值得注意的是, D_3 表示多层感知机输出层神经元个数, 由于输出层经过 *Softmax* 函数之后的输出值表示各类别概率分布, 因此该参数与具体分类任务相关, 即二分类任务将该参数设置为 2; 多分类任务设置为 21.

表 4　模型超参数设置

超参数符号	数值	含义	选取方式
B	64	模型训练样本批大小	经验值
lr	0.000 1	模型训练学习率	经验值
$dropout$	0.5	Dropout策略神经元丢弃率	经验值
D	16	域名字符的嵌入维度	网格搜索法, 搜索范围: {8, 16, 32, 64}
D_h	128	BiLSTM中的隐层神经元个数	网格搜索法, 搜索范围: {32, 64, 128, 256, 512}
$D_{\{q,k,v\}}$	64	自注意力机制嵌入层神经元个数	网格搜索法, 搜索范围: {16, 32, 64, 128, 256}
D_1	1 024	多层感知机第1层神经元个数	经验值
D_2	256	多层感知机第2层神经元个数	经验值
D_3	2或21	多层感知机第3层神经元个数	确定值, 二分类任务为2; 多分类任务为21
α	0.1	对比损失系数	网格搜索法, 搜索范围: {0.01, 0.05, 0.1, 0.5}
β	0.1	对比损失系数	网格搜索法, 搜索范围: {0.01, 0.05, 0.1, 0.5}

本实验采用精确度 (*Precision*) , 召回率 (*Recall*), *F*1 值评价模型性能, 评价指标定义如下:

$$Precision = \frac{\sum TP}{\sum TP + \sum FP},$$

$$Recall = \frac{\sum TP}{\sum TP + \sum FN},$$

$$F1 = \frac{Precision \times Recall}{Precision + Recall},$$

其中, TP 表示预测为正, 实际也为正的真阳性 (true positive, *TP*); FP 表示预测为正, 实际为负的假阳性 (false positive, *FP*); FN 表示预测为负, 实际为正的假阴性 (false negative, *FN*); $Precision$ 表示模型的查准率; $Recall$ 表示模型的查全率. $F1$ 值表示精准度与召回率之间的调和平均值, 用于衡量模型的综合性能. 此外, 本文还给出了 3 种指标的宏观平均值 (macro average, Macro Avg.) 和加权平均值 (weighted average, Weighted Avg.). 宏观平均值是指算术平均值, 不考虑类别不平衡性问题; 加权平均值是指各类别指标的加权平均. 考虑了类别不平衡因素.

3.4 实验结果与分析

为全面准确地评估模型性能, 我们在两种 DGA 域名检测任务上进行实验: (1) 二分类任务: 二分类是指将待检测域名分类为正常域名或虚假 DGA 域名; (2) 多分类任务: 多分类是指将待检测域名分类为如表 3 所示的具体 21 个类别. 针对以上两种检测任务, 分别得到如表 5 和表 6 所示的二分类实验结果以及如表 7、表 8 和图 3 所示的多分类实验结果. 基于实验结果, 得到以下分析结论.

表 5 二分类任务下各检测方法准确率 (*Precision*)

二分类	GNB	RF	GBDT	LSTM	L+A	Bi+1D	DGAD-SN	MCL4DGA
正常域名	0.8809	0.9636	0.9592	0.9821	0.9846	0.9860	0.9879	**0.9918**
DGA域名	0.9575	0.9909	0.9868	0.9931	0.9950	0.9955	0.9964	**0.9971**
Macro Avg.	0.9192	0.9773	0.9730	0.9876	0.9898	0.9907	0.9922	**0.9945**
Weighted Avg.	0.9254	0.9794	0.9752	0.9885	0.9906	0.9915	0.9928	**0.9949**

表 6 二分类任务下各检测方法 *F*1 值 (*F*1)

二分类	GNB	RF	GBDT	LSTM	L+A	Bi+1D	DGAD-SN	MCL4DGA
正常域名	0.9116	0.9755	0.9705	0.9863	0.9889	0.9899	0.9921	**0.9941**
DGA域名	0.9319	0.9819	0.9782	0.9900	0.9919	0.9926	0.9943	**0.9961**
Macro Avg.	0.9217	0.9787	0.9743	0.9882	0.9904	0.9913	0.9932	**0.9951**
Weighted Avg.	0.9233	0.9792	0.9750	0.9885	0.9906	0.9915	0.9934	**0.9953**

表 7 多分类任务下各检测方法准确率 (*Precision*)

多分类	GNB	RF	GBDT	LSTM	L+A	Bi+1D	DGAD-SN	MCL4DGA
Alexa	0.9619	0.9360	0.9558	0.9829	0.9833	0.9875	**0.9908**	0.9890
banjori	0.9954	1.0000	0.9995	0.9994	0.9995	0.9998	0.9999	**1.0000**
tinba	0.2687	0.8872	0.8859	0.9202	0.9220	0.8942	0.9255	**0.9259**
post	0.8407	0.9964	0.9995	1.0000	0.9998	0.9998	0.9998	**1.0000**
ramnit	0.3493	0.7622	0.7923	0.8009	0.8418	0.8397	0.8453	**0.8587**
qakbot	0.5214	0.7315	0.7378	0.7275	0.7699	0.7628	0.7819	**0.8147**
necurs	0.2727	0.8427	0.8554	0.9136	0.9559	0.9387	0.9562	**0.9673**
murofet	0.1819	0.7410	0.7575	0.7823	0.8540	0.8182	0.8448	**0.8623**
bebloh	0.2401	0.9669	0.9573	0.9881	**0.9904**	0.9882	0.9647	0.9861
simda	0.2401	0.9161	0.9202	0.9750	0.9712	0.9680	0.9899	**0.9907**
ranbyus	0.0453	**0.8609**	0.7777	0.8556	0.8326	0.8422	0.8478	0.8521
pykspa	0.2966	0.9620	0.9602	0.9230	0.9377	0.9663	0.9764	**0.9839**
dyre	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**1.0000**
kraken	0.5163	0.9239	0.9194	0.9632	0.9498	**0.9849**	0.9404	0.9354
Cryptolocker	0.0716	**0.9342**	0.6828	0.7179	0.4596	0.5544	0.5069	0.5823
nymaim	0.0526	0.5833	0.5078	0.4917	0.5000	0.6635	0.4845	**0.7053**
locky	0.0399	**1.0000**	0.7582	0.8235	0.9828	0.7771	0.8881	0.7778
vawtrak	0.0162	**1.0000**	0.8681	0.8649	0.6294	0.9602	0.9291	0.9057
shifu	0.0169	0.8906	0.8327	0.8688	0.8830	0.8522	0.8897	**0.9000**
ramdo	**1.0000**	0.9758	0.9856	0.9848	0.9951	0.9903	0.9951	0.9951
p2p	0.0126	0.0000	0.1064	0.4021	0.4500	0.4599	0.4140	**0.4643**
Macro Avg.	0.3305	0.8529	0.8219	0.8564	0.8567	0.8680	0.8653	**0.8808**
Weighted Avg.	0.8122	0.9342	0.9414	0.9586	0.9634	0.9638	0.9676	**0.9702**

● 针对二分类任务, MCL4DGA 全面优于对比模型. 其中, 准确率的宏平均值和加权平均值分别为 0.9945, 0.9949, *F*1 值的宏平均值和加权平均值分别为 0.9951, 0.9953. 针对多分类任务, MCL4DGA 在大多数情况下优于对比模型. 其中, 多分类任务的准确率宏平均值和加权平均值分别为 0.8808, 0.9702, *F*1 值的宏平均值和加权平均值分别为 0.8575, 0.9697.

● 图 3 展示了多分类任务的召回率实验结果. 其中, 纵坐标代表真实类别, 横坐标代表预测类别, 对角线表示模型的召回率. 直观上看, 对角线颜色越深且非对角线区域颜色越浅, 代表模型性能越好. 从图 3 可以看出, 相较于

基准模型, 本文提出的 MCL4DGA 检测结果中对角线颜色较深并且非对角线区域颜色较浅, 这表明 MCL4DGA 将真实类别错分为其他类别的概率较小, 即模型召回率较高.

● 如图 3 所示, 相当一部分 nymaim 和 p2p 域名被误分类为 qakbot. 这是因为这 3 类 DGA 域名由随机组合的字符生成, 并且字符频率分布几乎相同[44], 导致域名检测模型易发生误判现象.

表 8　多分类任务下各检测方法 $F1$ 值 ($F1$)

多分类	GNB	RF	GBDT	LSTM	L+A	Bi+1D	DGAD-SN	MCL4DGA
Alexa	0.5551	0.9644	0.9723	0.9859	0.9893	0.9899	0.9928	**0.9936**
banjori	0.9719	1.0000	0.9997	0.9997	0.9998	0.9999	1.0000	**1.0000**
tinba	0.3548	0.9122	0.9205	0.9521	0.9559	0.9434	0.9595	**0.9600**
post	0.7721	0.9982	0.9998	0.9998	0.9997	0.9996	0.9998	**1.0000**
ramnit	0.0787	0.7690	0.7937	0.8497	0.8707	0.8654	0.8794	**0.8834**
qakbot	0.0353	0.7106	0.7166	0.7370	0.7682	0.7601	0.7756	**0.7924**
necurs	0.0343	0.7055	0.7502	0.8678	0.8717	0.8793	0.8800	**0.8886**
murofet	0.0944	0.6844	0.7567	0.8115	0.8270	0.8305	**0.8374**	0.8337
bebloh	0.3201	0.9024	0.9223	0.9401	**0.9451**	0.9442	0.9415	0.9446
simda	0.2681	0.9164	0.9367	0.9766	0.9760	0.9740	0.9830	**0.9867**
ranbyus	0.0286	0.6427	0.7496	0.8523	0.8567	0.8610	**0.8658**	0.8654
pykspa	0.3737	0.9404	0.9533	0.9424	0.9510	0.9763	0.9829	**0.9881**
dyre	0.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	**1.0000**
kraken	0.3552	0.8309	0.8513	0.8733	0.8861	0.8806	0.8909	**0.8993**
Cryptolocker	0.0911	0.2008	0.3109	0.3884	0.4697	0.4727	**0.4864**	0.4566
nymaim	0.0148	0.0892	0.1836	0.1686	0.2204	0.2018	0.3872	**0.6351**
locky	0.0696	0.1372	0.2695	0.4022	0.4246	0.4221	0.4421	**0.4493**
vawtrak	0.0318	0.1027	0.3901	0.1823	0.4421	0.7495	0.8792	**0.9114**
shifu	0.0332	0.7600	0.8311	0.9074	0.9222	0.9035	0.9276	**0.9368**
ramdo	**1.0000**	0.9806	0.9927	0.9652	0.9976	0.9951	0.9976	0.9976
p2p	0.0249	0.0000	0.0932	0.2806	0.4724	0.4674	0.3846	**0.5849**
Macro Avg.	0.2623	0.6784	0.7330	0.7659	0.8041	0.8308	0.8334	**0.8575**
Weighted Avg.	0.6029	0.9296	0.9408	0.9569	0.9627	0.9627	0.9672	**0.9697**

● 深度学习检测方法普遍优于机器学习方法, 这意味着深度神经网络以其强大的拟合能力, 能够更加有效地捕获域名字符序列间的依赖关系, 提高检测准确性. 除此之外, 深度学习检测模型不需要构建额外的特征集合, 避免人为因素干扰实验结果, 具有较强的鲁棒性.

● 分析基于机器学习算法的检测模型可知, 相比于 GNB, 集成模型 RF 和 GBDT 取得了较好的检测结果, 这表明集成学习中通过堆叠多个单一分类器以增强算法性能这种策略是有效的.

● 分析基于深度学习算法的检测模型可知, 由于缺少注意力机制提供的字符权重信息和一维卷积神经网络捕获的局部感知信息, LSTM 相比于 L+A 和 Bi+1D 取得了较差的实验结果. 这表明, 堆叠多视角特征捕获算法有利于提高 DGA 域名检测的准确性. 相较于其他基准模型, DGAD-SN 表现优异, 这是因为其引入的对比学习能够辅助模型捕获高质量域名表示向量.

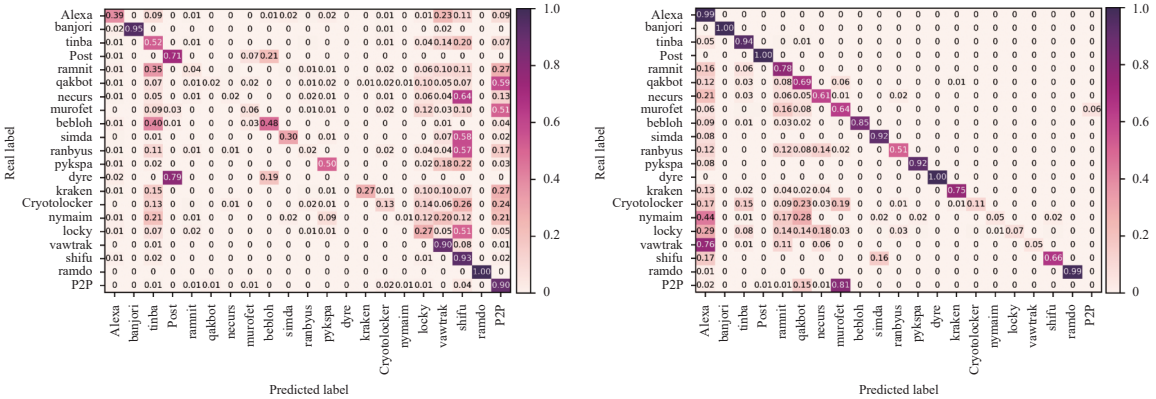
● 总的来说, MCL4DGA 表现最佳, 这得益于两方面: (1) 相较于 L+A 和 Bi+1D, MCL4DGA 额外引入了第 3 种特征捕获算法, 有助于学习域名字符序列中丰富的特征依赖信息; (2) 通过引入对比学习, 辅助捕获域名字符序列多视角特征之间的潜在一致性, 生成额外的自监督信号, 进而提升模型性能.

3.5 消融实验

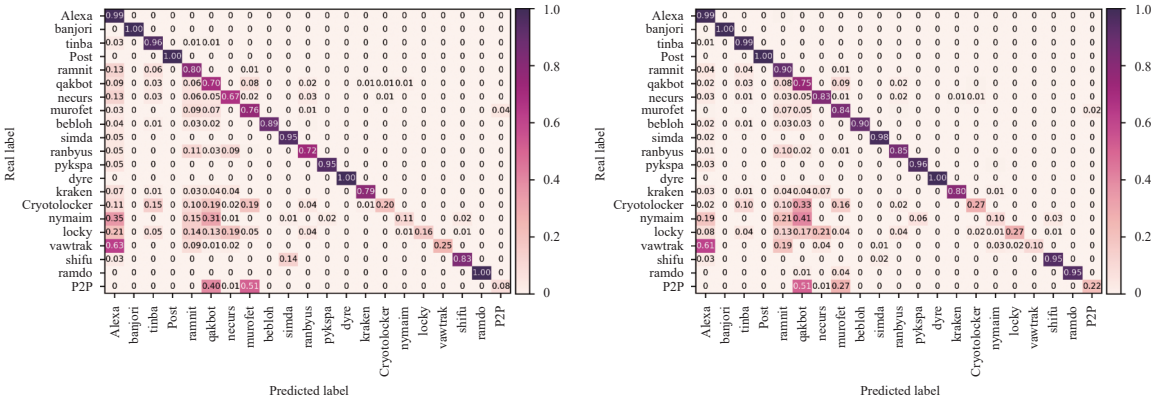
3.5.1 单一视角消融实验

为了验证本文所提出的多视角域名字符序列特征的有效性, 本节分别对自注意力机制提供的全局视角 (global

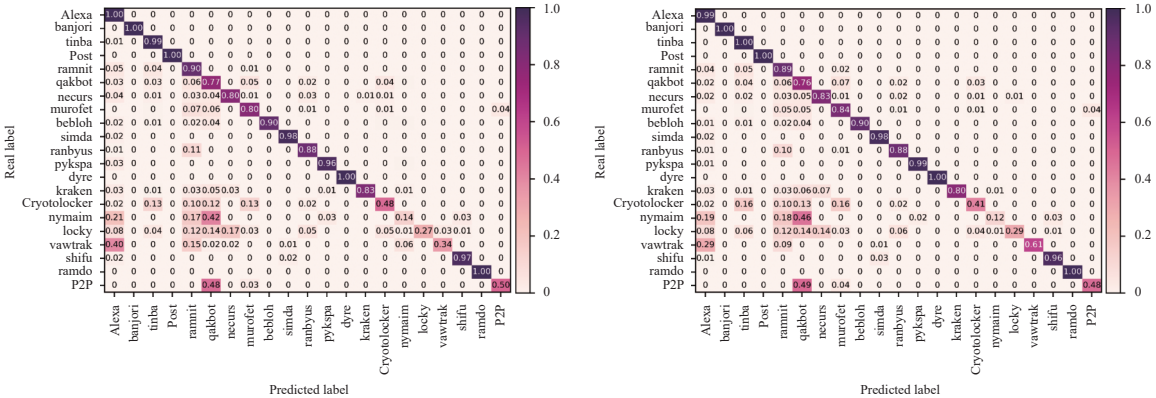
view)、双向长短期记忆网络提供的双向视角 (bi-directional view)、一维卷积神经网络提供的局部视角 (local view) 进行消融实验. 我们通过修改公式 (23) 中的变量, 生成以下变体模型: (1) MCL4DGA w/o global view: 将公式中的z_{global}去掉, 使全局视角下的字符序列特征失效; (2) MCL4DGA w/o bi-directional view: 将公式中的z_{bi}去掉, 使双向视角下的字符序列特征失效; (3) MCL4DGA w/o local view: 将公式中的z_{local}去掉, 使局部视角下的字符序列特征失效. 保持其他参数不变, 比较完整 MCL4DGA 与以上变体模型之间的性能.



(a) GNB 召回率 (b) RF 召回率

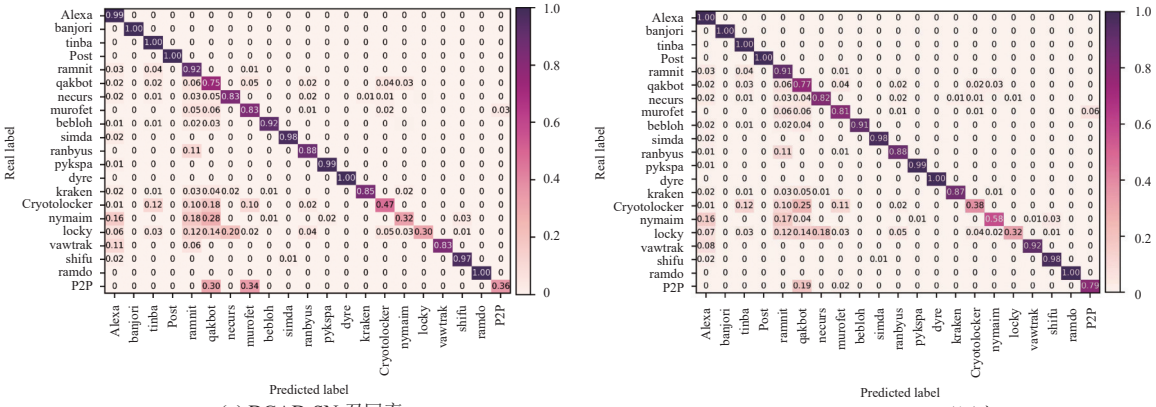


(c) GBDT 召回率 (d) LSTM 召回率



(e) L+A 召回率 (f) Bi+1D 召回率

图 3 多分类任务下各检测方法召回率 (*Recall*)



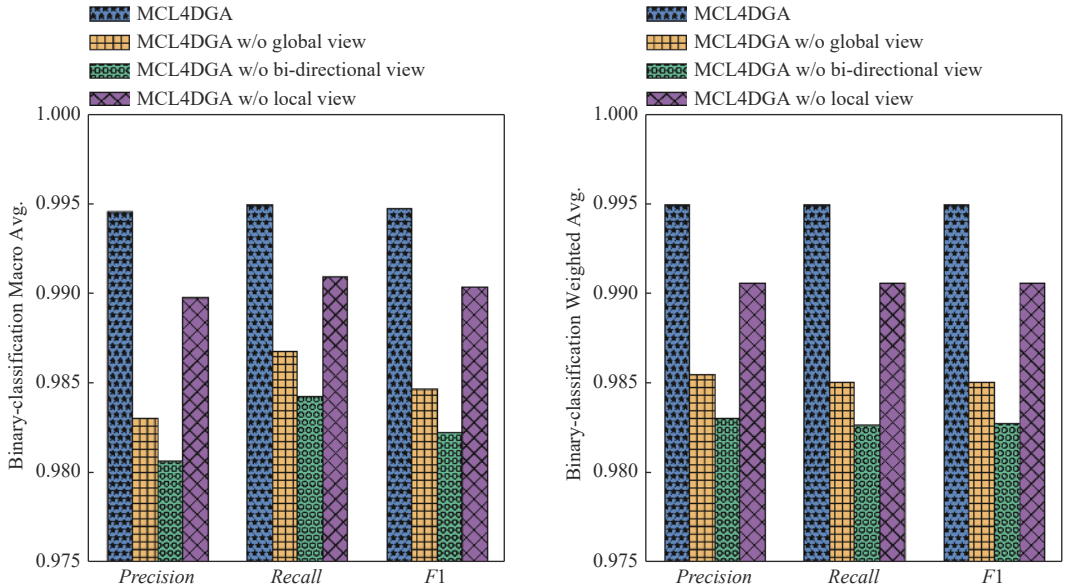
(g) DGAD-SN 召回率 (h) MCL4DGA 召回率

图 3 多分类任务下各检测方法召回率 (*Recall*)(续)

从图 4 中的实验结果可以看出, 所有变体模型均比完整模型 MCL4DGA 表现差, 这表明多视角域名字符序列特征对准确识别 DGA 域名具有积极贡献. 具体的, 相比于变体模型 MCL4DGA w/o global view 和 MCL4DGA w/o local view, 去掉双向视角字符序列特征的变体模型 MCL4DGA w/o bi-directional view 取得了最差的检测结果, 这表明双向 LSTM 能够有效捕获域名字符序列中的双向字符依赖信息, 这对准确地检测 DGA 域名是不可或缺的. 分析变体模型 MCL4DGA w/o global view 和 MCL4DGA w/o local view 的消融实验结果可知, 相比于一维卷积神经网络提供的局部序列特征, 自注意力机制所捕获的字符序列全局依赖信息能够提供更高的收益.

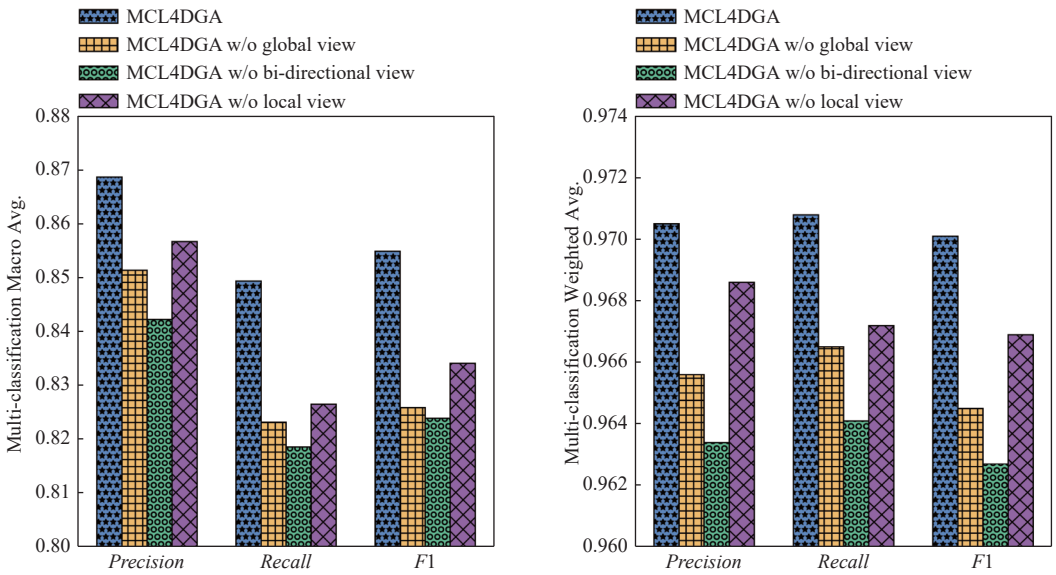
3.5.2 对比学习消融实验

为了验证本文所提出的多视角对比学习的有效性, 本节对多视角对比学习模块进行消融实验. 具体地, 我们通过控制公式 (28) 中的对比学习损失系数 α 和 β, 生成以下变体模型: (1) MCL4DGA w/o MCL: 将系数 α 和 β 设置为 0, 使多视角对比学习模块失效; (2) MCL4DGA w/o global vs. bi: 将系数 α 设置为 0, 使“全局视角 vs. 双向视角”对比学习模块失效; (3) MCL4DGA w/o local vs. bi: 将系数 β 设置为 0, 使“局部视角 vs. 双向视角”对比学习模块失效. 保持其他参数不变, 比较完整 MCL4DGA 与以上变体模型之间的性能.



(a) 二分类宏平均值 (b) 二分类加权平均值

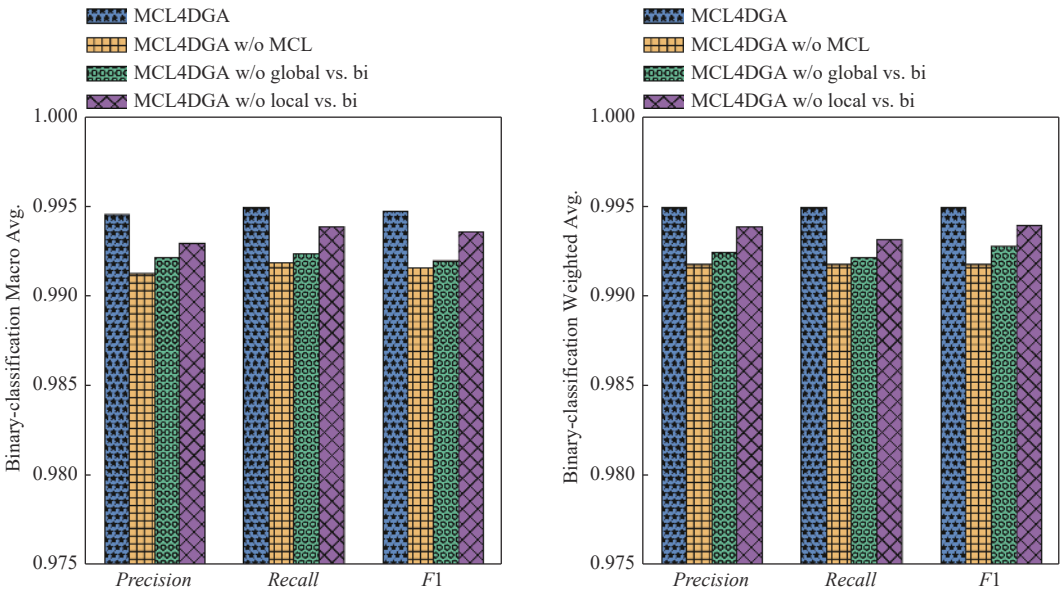
图 4 单一视角消融实验结果



(c) 多分类宏平均值　　(d) 多分类加权平均值

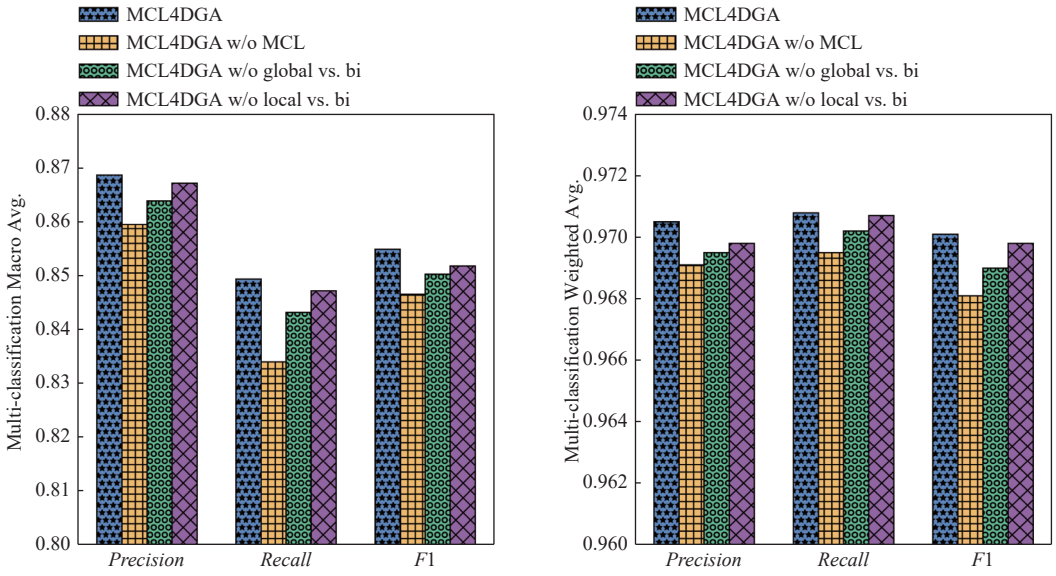
图 4　单一视角消融实验结果 (续)

从图 5 中的实验结果可以看出, 与完整模型 MCL4DGA 相比, 其变体模型 MCL4DGA w/o MCL、MCL4DGA w/o global vs. bi 和 MCL4DGA w/o local vs. bi 取得了较差的检测结果. 这表明本文提出的多视角对比学习能够通过引入对比学习优化损失, 捕获域名字符序列多视角特征之间的潜在一致性, 生成额外的自监督信号, 进而提升模型性能, 提高 DGA 域名检测的准确性. 具体地, 在变体模型中, MCL4DGA w/o MCL 表现最差, MCL4DGA w/o global vs. bi 次之, MCL4DGA w/o local vs. bi 表现最好, 这说明相较于单一视角的对比学习, 多视角域名字符序列特征间的对比学习在提高 DGA 域名检测准确率方面更有优势.



(a) 二分类宏平均值　　(b) 二分类加权平均值

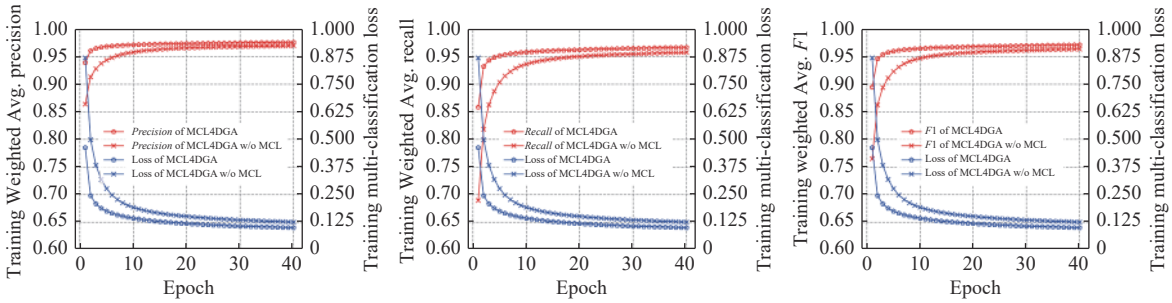
图 5　对比学习消融实验结果



(c) 多分类宏平均值　　(d) 多分类加权平均值

图 5　对比学习消融实验结果 (续)

为了进一步说明多视角对比学习在增强监督信号方面的优势, 将完整模型 MCL4DGA 和其变体模型 MCL4DGA w/o MCL 的训练过程可视化为图 6 所示形式. 通常情况下, 模型训练过程中的损失值大小及其收敛速度能够侧面反映监督信号的强弱, 即模型训练损失值越小、收敛速度越快, 则反向传播的监督信号越强; 反之, 训练损失值越大、收敛速度越慢, 则反向传播的监督信号越弱. 观察图 6 中的模型训练评价指标与损失值变化曲线可知, 相比于缺少多视角对比学习模块的变体模型 MCL4DGA w/o MCL, 完整模型 MCL4DGA 的训练损失值较小并且收敛速度较快, 相应地各评价指标均优于变体模型. 这表明引入多视角对比学习能够为 DGA 检测模型提供额外的监督信息, 进而增强模型性能.



(a) 训练加权平均精确度　(b) 训练加权平均召回率　(c) 训练加权平均 *F*1 值

图 6　多分类任务下模型训练评价指标与损失值变化曲线

3.6　参数影响分析

本节对 MCL4DGA 模型中的重要超参数进行参数影响分析, 以探究超参数对模型性能的影响.

3.6.1　自注意力机制嵌入维度

本实验分别将自注意力机制中的嵌入维度设置为 16, 32, 64, 128, 256. 实验结果如表 9 所示, 从实验结果可以看出嵌入维度的增加能够提升模型性能, 这是因为维度较高的向量能够存储足够的信息. 尽管本文已经采用 Dropout 策略以及层归一化操作防止模型出现过拟合现象, 然而过大的嵌入维度依然会导致模型陷入过拟合, 限制

模型发挥出最优性能. 由于算法特性和数据规模的限制, 深度神经网络模型出现过拟合现象是一种普遍存在的问题. 为了进一步缓解过拟合问题, 提高模型性能, 计划在未来研究工作中引入知识蒸馏技术. 该技术能够将存储在已训练好的检测模型中的归纳推理知识压缩到小模型当中, 减小模型参数规模, 避免出现过拟合现象.

表 9 自注意力机制嵌入维度对 MCL4DGA 性能的影响

分类	$D_{\{q,k,v\}}$	16	32	64	128	256
二分类	*F*1宏平均值	0.9931	0.9847	**0.9951**	0.9946	0.9935
	*F*1加权平均值	0.9939	0.9851	**0.9953**	0.9947	0.9936
多分类	*F*1宏平均值	0.8287	0.8304	**0.8575**	0.8505	0.7014
	*F*1加权平均值	0.9651	0.9694	**0.9697**	0.9662	0.9455

3.6.2 BiLSTM 隐层维度

本实验分别将 BiLSTM 嵌入维度设置为 32, 64, 128, 256, 512. 表 10 中的实验结果表明, 在不同 BiLSTM 嵌入维度下, 模型表现出不同的 DGA 域名检测性能, 即模型性能对 BiLSTM 嵌入维度具有敏感性. 选择合适的嵌入维度对充分发挥模型性能具有重要意义, 综合表 10 中的实验结果, 最佳 BiLSTM 嵌入维度为 128.

表 10 BiLSTM 嵌入维度对 MCL4DGA 性能的影响

分类	D_h	32	64	128	256	512
二分类	*F*1宏平均值	0.9931	0.9939	**0.9951**	0.9942	0.9941
	*F*1加权平均值	0.9932	0.9940	**0.9953**	0.9944	0.9942
多分类	*F*1宏平均值	0.8133	0.8216	**0.8575**	0.8461	0.8356
	*F*1加权平均值	0.9665	0.9670	0.9697	**0.9708**	0.9698

3.7 复杂度分析

本节将对所提出的模型 MCL4DGA 进行复杂度分析, 以验证模型在真实应用场景下的可行性. 复杂度分析具体包含两个方面: (1) 空间复杂度: 是指算法在运行过程中临时占用存储空间大小的量度, 本文选用模型参数作为空间复杂度的度量指标. 通常情况下, 模型参数越多, 其空间复杂度越高; (2) 时间复杂度: 是指执行当前算法所消耗的时间, 本文选用模型训练时间作为时间复杂度的度量指标. 模型训练时间越长, 其时间复杂度越高.

复杂度分析的实验环境配置信息如下: 操作系统 (operating system) 为 Ubuntu 18.04.3 LTS; 随机存储器 (random access memory) 为 128 GB DDR4 @ 3200 MHz; 中央处理器 (central processing unit) 为 Intel (R) Core (TM) i9-9980XE CPU @ 3.00 GHz; 图形处理器 (graphic processing unit) 为 NVIDIA TITAN RTX; 主要环境库为 Python 3.7.5、Keras 2.4.3、TensorFlow 2.2.0、NumPy 1.18.5、Scikit-learn 0.22.

由于本文所提出的方法 MCL4DGA 为基于深度神经网络的模型, 因此选择同为深度学习检测方法的 LSTM, L+A, Bi+1D 作为对比, 进行模型复杂度评估. 实验结果如表 11 所示.

表 11 模型复杂度分析

模型	空间复杂度 (模型参数)	时间复杂度 (训练时间)
LSTM	680085	2 h 49 min
L+A	3340610	3 h 7 min
Bi+1D	1802133	4 h 28 min
MCL4DGA	4906709	2 h 13 min

从表 11 的实验结果可以看出, 得益于多视角对比学习所提供的额外自监督信号, 模型 MCL4DGA 相比其他方法具有训练时间短, 收敛速度快的特点, 这表明模型的时间复杂度较低, 在实际应用场景下模型加载更快. 然而,

引入多视角特征不可避免的会导致模型参数量较大, 空间复杂度变高. 为了解决该问题, 我们将在未来研究工作中持续优化该模型, 利用知识蒸馏、模型压缩等技术获取参数量较少的轻量级模型, 降低模型空间复杂度.

4 结 语

为了提高 DGA 域名检测的准确性, 本文提出了一种基于多视角对比学习的 DGA 域名检测方法 (MCL4DGA). 该模型主要由 4 部分组成: (1) 预训练字符嵌入: 通过引入字符预训练, 将字符的语义和词法先验知识嵌入到字符表示向量中; (2) 多视角依赖信息捕获: 在现有研究基础上, 本文提出捕获域名字符序列中的全局视角、局部视角和双向视角信息, 以获取表达能力更强的域名表示向量; (3) 多视角对比学习: 通过引入多视角对比学习, 捕获域名字符序列多视角表示向量之间的潜在一致性, 生成额外的自监督信号, 提高 DGA 域名检测的准确性; (4) 多层感知机输出模块: 通过堆叠多层神经元, 形成多层感知机, 经过 *Softmax* 函数之后的输出值表示各类别概率分布, 用于 DGA 域名的分类及检测. 本文在真实域名数据集上进行了实验, 实验表明本文提出的模型能够有效地提高 DGA 域名检测的准确性.

在未来研究工作中, 我们计划有针对性地利用知识蒸馏技术对 DGA 域名检测模型进行压缩. 压缩后的模型参数量大大降低, 可以避免出现严重的过拟合现象. 同时, 轻量级模型还具备部署方便以及执行速度快等优点.

References:

[1] Hao ZC, Wang ZSH. Analysis of the global cyberspace security posture in 2021. Information Security and Communications Privacy, 2022(1): 2–10 (in Chinese with English abstract). [doi: 10.3969/j.issn.1009-8054.2022.01.001]

[2] Liu SL, Qi ZH. Malicious domain detection based on diversified characteristics. Journal of Nanjing University of Posts and Telecommunications (Natural Science Edition), 2021, 41(6): 95–100 (in Chinese with English abstract). [doi: 10.14132/j.cnki.1673-5439.2021.06.013]

[3] Sood AK, Zeadally S. A taxonomy of domain-generation algorithms. IEEE Security and Privacy, 2016, 14(4): 46–53. [doi: 10.1109/MSP.2016.76]

[4] Tong V, Nguyen G. A method for detecting DGA botnet based on semantic and cluster analysis. In: Proc. of the 7th Symp. on Information & Communication Technology. Ho Chi Minh City: ACM, 2016. 272–277. [doi: 10.1145/3011077.3011112]

[5] Han CY, Zhang YZ. CODDULM: An approach for detecting C&C domains of DGA on passive DNS traffic. In: Proc. of the 6th Int'l Conf. on Computer Science and Network Technology (ICCSNT). Dalian: IEEE, 2017. 385–388. [doi: 10.1109/ICCSNT.2017.8343724]

[6] Chen Y, Yan S, Pang TY, Chen R. Detection of DGA domains based on support vector machine. In: Proc. of the 3rd Int'l Conf. on Security of Smart Cities, Industrial Control System and Communications. Shanghai: IEEE, 2018. 1–4. [doi: 10.1109/SSIC.2018.8556788]

[7] Wang Z, Jia ZT, Zhang B. A detection scheme for DGA domain names based on SVM. In: Proc. of the 2018 Int'l Conf. on Mathematics, Modelling, Simulation and Algorithms (MMSA 2018). Chengdu: Atlantis Press, 2018. 257–263. [doi: 10.2991/mmsa-18.2018.58]

[8] Antonakakis M, Perdisci R, Nadji Y, Vasiloglou N, Abu-Nimeh S, Lee WK, Dagon D. From throw-away traffic to bots: Detecting the rise of DGA-based malware. In: Proc. of the 21st USENIX Conf. on Security Symp. Bellevue: USENIX Association, 2012. 491–506.

[9] Woodbridge J, Anderson HS, Ahuja A, Grant D. Predicting domain generation algorithms with long short-term memory networks. arXiv: 1611.00791, 2016.

[10] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735–1780. [doi: 10.1162/neco.1997.9.8.1735]

[11] Yu B, Gray DL, Pan J, De Cock M, Nascimento ACA. Inline DGA detection with deep networks. In: Proc. of the 2017 IEEE Int'l Conf. on Data Mining Workshops. New Orleans: IEEE, 2017. 683–692. [doi: 10.1109/ICDMW.2017.96]

[12] Lecun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, Jackel LD. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1989, 1(4): 541–551. [doi: 10.1162/neco.1989.1.4.541]

[13] Highnam K, Puzio D, Luo S, Jennings NR. Real-time detection of dictionary DGA network traffic using deep learning. SN Computer Science, 2021, 2(2): 110. [doi: 10.1007/s42979-021-00507-w]

[14] Qiao YC, Zhang B, Zhang WZ, Sangaiah AK, Wu HL. DGA domain name classification method based on long short-term memory with attention mechanism. Applied Sciences, 2019, 9(20): 4205. [doi: 10.3390/app9204205]

[15] Doersch C, Gupta A, Efros AA. Unsupervised visual representation learning by context prediction. In: Proc. of the 2015 IEEE Int'l Conf. on Computer Vision. Santiago: IEEE, 2015. 1422–1430. [doi: 10.1109/ICCV.2015.167]

[16] Gidaris S, Singh P, Komodakis N. Unsupervised representation learning by predicting image rotations. arXiv:1803.07728, 2018.

[17] Plohmann D, Yakdan K, Klatt M, Bader J, Gerhards-Padilla E. A comprehensive measurement study of domain generating malware. In: Proc. of the 25th USENIX Conf. on Security Symp. Austin: USENIX Association, 2016. 263–278.

[18] Liao K, Zhao ZM, Doupé A, Ahn GJ. Behind closed doors: Measurement and analysis of Cryptolocker ransoms in Bitcoin. In: Proc. of the 2016 APWG Symp. on Electronic Crime Research (eCrime). Toronto: IEEE, 2016. 1–13. [doi: 10.1109/ECRIME.2016.7487938]

[19] Kuhn J, Mueller L, Kessem L. The dyre wolf: Attacks on corporate banking accounts. 2015. https://portal.sec.ibm.com/mss/html/en_US/support_resources/pdf/Dyre_Wolf_MSS_Threat_Report.pdf

[20] Mac H, Tran D, Tong V, Nguyen LG, Tran HA. DGA botnet detection using supervised learning methods. In: Proc. of the 8th Int'l Symp. on Information and Communication Technology. Nha Trang City: ACM, 2017. 211–218. [doi: 10.1145/3155133.3155166]

[21] Namgung J, Son S, Moon YS. Efficient deep learning models for DGA domain detection. Security and Communication Networks, 2021, 2021: 8887881. [doi: 10.1155/2021/8887881]

[22] Sivaguru R, Choudhary C, Yu B, Tymchenko V, Nascimento A, de Cock M. An evaluation of DGA classifiers. In: Proc. of the 2018 IEEE Int'l Conf. on Big Data (Big Data). Seattle: IEEE, 2018. 5058–5067. [doi: 10.1109/BigData.2018.8621875]

[23] Stiborek J, Pevný T, Rehák M. Probabilistic analysis of dynamic malware traces. Computers & Security, 2018, 74: 221–239. [doi: 10.1016/j.cose.2018.01.012]

[24] Bilge L, Sen S, Balzarotti D, Kirda E, Kruegel C. EXPOSURE: A passive DNS analysis service to detect and report malicious domains. ACM Trans. on Information & System Security, 2014, 16(4): 1–28. [doi: 10.1145/2584679]

[25] Luo X, Wang LM, Xu Z, Yang J, Sun M, Wang J. DGASensor: Fast detection for DGA-based malwares. In: Proc. of the 5th Int'l Conf. on Communications and Broadband Networking. Bali: ACM, 2017. 47–53. [doi: 10.1145/3057109.3057112]

[26] Alenazi A, Traore I, Ganame K, Woungang I. Holistic model for HTTP botnet detection based on DNS traffic analysis. In: Proc. of the 1st Int'l Conf. on Intelligent, Secure, and Dependable Systems in Distributed and Cloud Environments. Vancouver: Springer, 2017. 1–18. [doi: 10.1007/978-3-319-69155-8_1]

[27] Yadav S, Reddy AKK, Reddy ALN, Ranjan S. Detecting algorithmically generated malicious domain names. In: Proc. of the 10th ACM SIGCOMM Conf. on Internet Measurement. Melbourne: ACM, 2010. 48–61. [doi: 10.1145/1879141.1879148]

[28] Upadhyay S, Ghorbani A. Feature extraction approach to unearth domain generating algorithms (DGAs). In: Proc. of the 2020 IEEE Int'l Conf. on Dependable, Autonomic and Secure Computing, Int'l Conf. on Pervasive Intelligence and Computing, Int'l Conf. on Cloud and Big Data Computing, Int'l Conf. on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech). Calgary: IEEE, 2020. 399–405. [doi: 10.1109/DASC-PICom-CBDCom-CyberSciTech49142.2020.00077]

[29] Zhu JC, Zou FT. Detecting malicious domains using modified SVM model. In: Proc. of the 21st IEEE Int'l Conf. on High Performance Computing and Communications; the 17th IEEE Int'l Conf. on Smart City; the 5th IEEE Int'l Conf. on Data Science and Systems (HPCC/SmartCity/DSS). Zhangjiajie: IEEE, 2019. 492–499. [doi: 10.1109/HPCC/SmartCity/DSS.2019.00079]

[30] da Silva LM, Silveira MR, Cansian AM, Kobayashi HK. Multiclass classification of malicious domains using passive DNS with XGBoost: (Work in progress). In: Proc. of the 19th IEEE Int'l Symp. on Network Computing and Applications (NCA). Cambridge: IEEE, 2020. 1–3. [doi: 10.1109/NCA51143.2020.9306705]

[31] Curtin RR, Gardner AB, Grzonkowski S, Kleymenov A, Mosquera A. Detecting DGA domains with recurrent neural networks and side information. In: Proc. of the 14th Int'l Conf. on Availability, Reliability and Security. Canterbury: ACM, 2019. 20. [doi: 10.1145/3339252.3339258]

[32] Tong MK, Sun XQ, Yang JH, Zhang H, Zhu S, Liu XR, Liu H. D3N: DGA detection with deep-learning through NXDomain. In: Proc. of the 12th Int'l Conf. on Knowledge Science, Engineering and Management. Athens: Springer, 2019. 464–471. [doi: 10.1007/978-3-030-29551-6_41]

[33] Hu XY, Li M, Cheng G, Li RD, Wu H, Gong J. Towards accurate DGA detection based on siamese network with insufficient training samples. In: Proc. of the 2022 ICC IEEE Int'l Conf. on Communications. Seoul: IEEE, 2022. 2670–2675. [doi: 10.1109/ICC45855.2022.9838409]

[34] Tomas M, Ilya S, Kai C, Greg C, Jeffrey D. Distributed representations of words and phrases and their compositionality. In: Proc. of the 26th Int'l Conf. on Neural Information Processing Systems. Lake Tahoe: Curran Associates Inc., 2013. 3111–3119.

[35] Pennington J, Socher R, Manning C. GloVe: Global vectors for word representation. In: Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing. Doha: Association for Computational Linguistics, 2014. 1532–1543. [doi: 10.3115/v1/D14-1162]

[36] Devlin J, Chang MW, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Vol. 1 (Long and Short Papers). Minneapolis: Association for Computational Linguistics, 2019. 4171–4186. [doi: 10.18653/v1/N19-1423]

[37] Boukkouri HE, Ferret O, Lavergne T, Noji H, Zweigenbaum P, Tsujii J. CharacterBERT: Reconciling ELMo and BERT for word-level

open-vocabulary representations from characters. In: Proc. of the 28th Int'l Conf. on Computational Linguistics. Barcelona: Int'l Committee on Computational Linguistics, 2020. 6903–6915. [doi: 10.18653/v1/2020.coling-main.609]

[38] Xie X, Sun F, Liu ZY, Wu SW, Gao JY, Zhang JD, Ding BL, Cui B. Contrastive learning for sequential recommendation. In: Proc. of the 38th IEEE Int'l Conf. on Data Engineering (ICDE). Kuala Lumpur: IEEE, 2022. 1259–1273. [doi: 10.1109/ICDE53745.2022.00099]

[39] van den Oord A, Li YZ, Vinyals O. Representation learning with contrastive predictive coding. arXiv:1807.03748, 2019.

[40] Diba A, Sharma V, Safdari R, Lotfi D, Sarfraz MS, Stiefelhagen R, van Gool L. Vi^{2}CLR: Video and image for visual contrastive learning of representation. In: Proc. of the 2021 IEEE/CVF Int'l Conf. on Computer Vision. Montreal: IEEE, 2021. 1482–1492. [doi: 10.1109/ICCV48922.2021.00153]

[41] Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv:1412.6980, 2017.

[42] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 2014, 15(1): 1929–1958.

[43] Ioffe S, Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proc. of the 32nd Int'l Conf. on Machine Learning. Lille: JMLR.org, 2015. 448–456.

[44] Spooren J, Preuveneers D, Desmet L, Janssen P, Joosen W. On the use of DGAs in malware: An everlasting competition of detection and evasion. ACM SIGAPP Applied Computing Review, 2019, 19(2): 31–43. [doi: 10.1145/3357385.3357388]

附中文参考文献:

[1] 郝志超, 王旨思虹. 2021年全球网络空间安全态势分析. 信息安全与通信保密, 2022(1): 2–10. [doi: 10.3969/j.issn.1009-8054.2022.01.001]

[2] 刘善玲, 祁正华. 基于特征多样化的恶意域名检测. 南京邮电大学学报(自然科学版), 2021, 41(6): 95–100. [doi: 10.14132/j.cnki.1673-5439.2021.06.013]

王继虎(1992－), 男, 博士生, 主要研究领域为网络安全, 深度学习, 数据挖掘.

孔凡玉(1978－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为数据安全与隐私计算, 信息安全.

刘子雁(1990－), 男, 硕士, 主要研究领域为数据挖掘, 软件工程, 网络安全.

史玉良(1978－), 男, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为大数据, 人工智能, 信息安全.

倪金超(1990－), 男, 硕士, 主要研究领域为软件工程, 软件测试, 网络安全.