软件学报 ISSN 1000-9825, CODEN RUXUEW
[doi: 10.13328/j.cnki.jos.007651] [CSTR: 32375.14.jos.007651]
©中国科学院软件研究所版权所有.
E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于数据驱动蒸馏与性能激励的入侵检测迭代模型*

田 润[1,2], 张海霞[1], 连一峰[1], 张立武[1]

[1](中国科学院 软件研究所 可信计算与信息保障实验室, 北京 100190)

[2](中国科学院大学, 北京 100049)

通信作者: 张海霞, E-mail: haixia@iscas.ac.cn

摘 要: 入侵检测系统对先验知识的依赖及其输入数据中丰富的文本语义使其较为适合在大语言模型中应用, 但大语言模型的算力开销与数据隐私约束使其难以直接落地, 与此同时小模型虽便于部署但性能受限. 鉴于此, 提出一种基于数据驱动蒸馏与性能激励的入侵检测迭代模型 RADOM, 通过大小模型之间的数据交互实现知识蒸馏, 并以小模型的错误预测作为迭代反馈改进大语言模型的数据生成质量. 同时, 引入特征维度优化机制, 借助大语言模型完成特征筛选与迭代更新, 以进一步提升小模型的分类能力. 在公开数据集与自采数据集上的实验结果表明, RADOM 能够有效提升小模型对攻击行为的检测与分类性能, 小模型的准确率由 67.30% 提高到 96.19%, 验证了该方法的有效性.

关键词: 入侵检测; 大语言模型; 数据生成

中图法分类号: TP309

中文引用格式: 田润, 张海霞, 连一峰, 张立武. 基于数据驱动蒸馏与性能激励的入侵检测迭代模型. 软件学报. http://www.jos.org.cn/1000-9825/7651.htm

英文引用格式: Tian R, Zhang HX, Lian YF, Zhang LW. Iterative Model for Intrusion Detection via Data-driven Distillation and Performance Motivation. Ruan Jian Xue Bao/Journal of Software (in Chinese). http://www.jos.org.cn/1000-9825/7651.htm

Iterative Model for Intrusion Detection via Data-driven Distillation and Performance Motivation

TIAN Run[1,2], ZHANG Hai-Xia[1], LIAN Yi-Feng[1], ZHANG Li-Wu[1]

[1](Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)

[2](University of Chinese Academy of Sciences, Beijing 100049, China)

Abstract: Intrusion detection systems rely on prior knowledge, and their input data contain rich semantic information, making them well suited for large language models (LLMs). However, the computational overhead of LLMs and data privacy constraints make their direct deployment difficult. Meanwhile, although small models are convenient for deployment, their performance is limited. To address this issue, this study proposes an iterative model for intrusion detection based on data-driven distillation and performance motivation, termed as RADOM. RADOM performs knowledge distillation through data exchange between LLMs and small models, and improves the quality of LLM-generated data by using incorrect predictions from the small models as iterative feedback. At the same time, this study introduces a feature dimension optimization mechanism, in which LLMs are used to perform feature selection and iterative updating, thereby further improving the classification capability of small models. Experimental results on public datasets and self-collected datasets show that RADOM effectively improves the detection and classification performance of small models for attack behaviors, and the accuracy of small models increases from 67.30% to 96.19%, verifying the effectiveness of the proposed method.

Key words: intrusion detection; large language model (LLM); data generation

入侵检测系统 (intrusion detection system, IDS) 是现代网络系统的重要组成部分[1–3], 其旨在通过对网络传输行为进行监控, 对其中可疑活动发出警报或采取主动措施. 入侵检测系统的概念最早可追溯至 20 世纪 80 年代, 起

* 基金项目: 国家重点研发计划 (2023YFB3107203)
收稿时间: 2025-08-12; 修改时间: 2026-01-03, 2026-02-04; 采用时间: 2026-02-11; jos 在线出版时间: 2026-08-12

初通常由研究人员和学者基于规则或模式匹配方法进行开发[4]. 随着人工智能与机器学习等算法的不断发展, 近年来的入侵检测系统大量采用了神经网络[5]、聚类分析[6]等方法, 较大幅度地改善了其对攻击行为的识别准确率. 此类方法通常需要将网络流量转换为定长向量表示, 而用于该转换的特征维度集往往会对模型性能产生显著影响[7]. 现在大语言模型在自然语言处理等各个领域表现十分突出, 而入侵检测任务不仅需要十分丰富的先验知识, 其输入还通常含有大量文本信息, 因此较为适合大语言模型的深入应用.

通常而言, 入侵检测系统的各类攻击识别算法部署于防火墙[8]、流量探针[9]等专用防护设备, 或以守护进程的形式运行于主机端[10,11], 但无论部署位置如何, 可支配的算力与内存资源均较为有限[12]; 与此同时, 当代互联网流量规模愈发庞大[13], 对入侵检测的吞吐量与响应时延提出了严格要求. 另一方面, 网络流量中含有大量敏感信息[14], 数据隐私与合规约束也是不容忽视的问题. 上述因素共同导致以大语言模型为代表的高算力需求模型目前难以直接应用于入侵检测的在线场景. 相较之下, 基于机器学习与小规模神经网络的小模型具有更高的部署灵活性与可扩展性[15], 可按需增量引入新数据进行训练, 同时不会引入过高的训练与推理成本, 更易适配真实环境[3]. 然而, 小模型在表达能力与知识储量方面较大语言模型仍存在差距, 因此在资源与隐私约束下如何有效融合二者优势, 是一个值得深入研究的关键问题.

本文提出一种基于数据驱动蒸馏与性能激励的入侵检测迭代模型 RADOM, 其核心思想是利用大语言模型的生成能力, 通过数据生成实现从大语言模型向小模型的知识蒸馏; 有别于传统单向蒸馏, 小模型同时通过性能激励的方式为大语言模型提供数据方向引导与新知识反馈, 从而形成面向目标任务的闭环迭代. 在此基础上, 为进一步提升小模型对目标攻击行为的识别能力, 本文进一步借助大语言模型对现有网络流量特征维度集进行优化与迭代更新. 基于 CSE-CIC-IDS2018 数据集[7]与 SQLi-Labs 平台[16]采集数据集的实验结果表明, 该方法能够有效支撑小模型的引导式训练.

本文的主要贡献如下.

(1) 提出一种基于数据驱动蒸馏与性能激励的入侵检测迭代模型, 利用大语言模型生成的网络流量数据对小模型进行训练, 使其可以有效识别实际环境中采集到的网络攻击行为. 该模型包含大语言模型与小模型之间的沟通迭代机制, 通过双向交互实现数据生成的自适应.

(2) 为增强对以 SQL 注入为代表的 Web 攻击行为的检测能力, 在模型输入中引入攻击载荷嵌入向量, 并提出一种基于性能激励的特征维度迭代优化算法, 在大语言模型辅助下对 CSE-CIC-IDS2018 数据集的流量特征维度进行精简与迭代, 进而获得更具判别性的特征表达.

(3) 提出一种基于检索增强[17]的网络流量数据生成算法, 用于构建小模型的初始训练集. 在此基础上提出一种基于数据驱动蒸馏的模型训练与迭代算法, 将小模型在真实环境中的预测错误将反馈给大语言模型, 后者据此迭代生成新的针对性数据集以补充小模型训练数据, 从而通过迭代提升性能.

(4) 在 CSE-CIC-IDS2018 数据集与基于 SQLi-Labs 采集的数据集上开展多轮测试, 结果显示, 即便使用最简单的全连接网络, 本方法亦可使小模型的准确率由 67.30% 提高到 96.19%, 验证了本方法的有效性.

本文第 1 节介绍相关研究工作. 第 2 节介绍 RADOM 模型的理论分析与主要算法. 第 3 节基于 CSE-CIC-IDS2018 数据集与基于 SQLi-Labs 采集的数据集对本文提出的方法进行实验评估. 第 4 节对本文进行总结.

1 相关工作

1.1 基于机器学习的入侵检测技术

从算法范式上看, 入侵检测技术可大致分为两类: 基于签名的检测与基于异常的检测[18]. 前者通常依赖模式匹配技术识别攻击, 当观测到的入侵特征与签名库中的既有签名匹配时, 即判定为攻击, 其代表性方法包括 Meiners 等人[19]提出的签名状态机与 Lin 等人[20]提出的语义条件. 然而, 随着零日攻击 (即利用已发现但尚无有效防御措施的漏洞实施的攻击) 比例上升[21], 此类依赖先验知识的方法在面对未知威胁场景时的有效性受限, 且签名库的维护成本通常较高. 后者即基于异常的检测, 其核心假设是攻击行为与正常使用行为在统计分布上存在可

区分差异, 进而通过对正常通信或目标异常进行建模来完成识别. 典型方法包括 Annachhatre 等人[22]提出的隐藏马尔可夫模型分析、Liu 等人[23]提出的邻近算法及 Atefinia 等人[5]提出的模块化深层神经网络模型等. 此外, Fu 等人[24]提出的 Whisper 通过提取频率特征与限制特征规模实现了入侵检测系统的高准确率和高吞吐量. Zhou 等人[25]提出的 TrafficFormer 则借鉴了自然语言处理领域常用的预训练技术, 通过引入多种预训练任务抽取流量数据间的关联信息, 提升流量数据的表示能力. 与基于签名的方法相比, 基于异常的检测对未知威胁具备更强的发现潜力, 但在正常行为建模不足或环境分布发生漂移时容易产生较高误报, 需要在特征表示、阈值设定与自适应机制方面精心设计.

1.2 大语言模型赋能的入侵检测技术

在基于大语言模型的入侵检测研究方面, Ali 等人[26]提出的 HuntGPT 能够生成入侵检测的辅助信息, 但该工作中大语言模型并未实际参与入侵检测模型的训练过程, 且缺少对其知识的持续补充与校验. 另一些研究, 如 Li 等人[27]提出的 IDS Agent, 由大语言模型直接参与检测决策, 但也因此带来了较高的时间与算力开销, 难以满足入侵检测对时效性的要求. Cui 等人[28]的 TrafficLLM 提出了一种面向网络流量的大语言模型适配框架, 通过流量领域专用分词、双阶段指令-流量微调解耦等机制, 提升大语言模型在流量监测分类领域的泛化能力. 总体而言, 现有做法在实时性与可落地性方面仍存在不足, 有必要在保证隐私与资源约束的前提下探索更为轻量且可迭代增强的路径.

在实际应用领域中, 大语言模型不仅可以作为模型本身直接参与模型预测, 还可以利用其优秀的内容生成能力生成辅助数据集, 从而支持其他模型的训练. Ye 等人[29]提出的 ZeroGen 是运用大语言模型进行数据生成的代表性研究, 可在零样本条件下生成训练数据以支持小模型训练. 然而, 该方法仍停留在知识的单向传递, 缺乏基于反馈的迭代机制以持续提升生成数据的相关性与质量. 在此基础上, Wang 等人[30]提出的 S3 框架实现了大语言模型与小模型的迭代交互, 但研究范围仍主要局限于自然语言处理任务, 只能生成自然语言数据, 同时所用大语言模型未连接外部知识库或进行面向任务的微调, 难以将小模型从真实环境数据中获得的增量知识长期固化. 在网络安全方向, 也出现了基于大语言模型的数据生成探索, 如 Daneshvar 等人[31]提出的 VulScribeR 将检索增强生成策略用于脆弱性代码生成. 总体来看, 现有方法要么缺乏闭环反馈以保障数据生成的持续优化, 要么受限于任务形态与知识更新能力, 距离满足入侵检测场景的时效性与可迁移性要求仍存在差距.

此外, 大语言模型的知识储备也使其适合作为知识蒸馏中的教师模型. 知识蒸馏的研究可追溯至 Buciluǎ等人[32]的工作, 他们首次提出模型压缩的思想, 旨在将大型模型或模型集中的知识转移到较小模型的训练过程中, 同时尽量避免显著的精度下降. 随后, Hinton 等人[33]对这一思想予以系统化, 概括为知识蒸馏范式. 经典的知识蒸馏框架由教师模型、学生模型与蒸馏算法这 3 部分构成[34], 其中常见做法是由教师模型对目标数据集进行标注, 生成软标签或伪标签, 再由学生模型在蒸馏损失与常规监督损失的联合约束下完成学习. 目前在基于大语言模型的入侵检测蒸馏模型方面的代表性研究包括 Yang 等人[35]提出的于大语言模型知识蒸馏的异常检测训练方法等.

然而, 该类方法也面临若干局限: 第一, 仍依赖外部提供的目标数据集, 在数据受限或隐私约束场景中适用性不足; 第二, 学生模型主要学习教师模型在目标数据上的判别边界, 难以获得教师在目标分布之外的潜在知识; 第三, 对教师模型能力依赖较高, 当教师模型存在偏差或与应用域发生分布漂移时, 学生模型难以获得稳定收益. 为缓解上述问题, 后续研究提出了多种扩展方法, 包括无数据蒸馏[36]、终身学习[37]等, 以期在数据、计算与知识迁移之间取得更稳健的折中与统一.

1.3 入侵检测经典数据集

由于网络流量数据具有较强的隐私属性, 公开的入侵检测系统数据集对研究工作至关重要. 当前在入侵检测研究中常用的数据集包括 KDD 99[38]、NSL-KDD[39]、CTU-13[40]、DARPA[41]、UNSW-NB15[42]、CIC-IDS2017[7]、CSE-CIC-IDS2018[7]等. 其中, CSE-CIC-IDS2018 数据集规模较大, 包含超过 1 600 万条流量实例、6 类攻击、14 种攻击方法, 并提出 79 项统计特征维度, 因而应用较为广泛, 其数据分布参见表 1. 然而, 这些数据集通常缺乏时效性, 单纯基于这些数据集训练出的入侵检测模型无法有效应对快速变化的新型攻击行为, 同时本文后续实验也

说明这些数据集可能还存在数据不平衡、攻击同质化等问题.

表 1　完整的 CSE-CIC-IDS2018 数据集每日流量实例数量

子数据集	正常流量	攻击流量
2018/02/14 星期三-暴力破解	667 626	380 949
2018/02/15 星期四-DoS攻击	996 077	52 498
2018/02/16 星期五-DoS攻击	446 772	601 802
2018/02/20 星期二-DDoS攻击	7 372 557	576 191
2018/02/21 星期三-DDoS攻击	360 833	687 742
2018/02/22 星期四-Web攻击	1 048 213	362
2018/02/23 星期五-Web攻击	1 048 009	566
2018/02/28 星期三-渗透攻击	544 200	688 871
2018/03/01 星期四-渗透攻击	238 037	93 063
2018/03/02 星期五-Bot攻击	762 384	286 191
总计	13 484 708	2 748 235

2　基于数据驱动蒸馏与性能激励的入侵检测迭代模型

正如第 1 节相关工作中所述, 现有入侵检测技术要么基于已知攻击进行学习, 在先验未覆盖攻击出现时缺乏针对模型输入与模型本身的自动调整能力; 要么时效性与可迁移性不满足实际需求. 因此, 本文将这些局限性总结为以下 3 个研究问题.

RQ1: 如何基于少样本数据学习得到具有相对良好性能的入侵检测模型?

RQ2: 如何增强入侵检测模型对先验未覆盖攻击行为的自适应性?

RQ3: 如何在入侵检测模型性能与效率之间取得更优的均衡?

为解决所提出的研究问题, 本文提出基于数据驱动蒸馏与性能激励的入侵检测迭代模型 RADOM, 其核心思想借鉴强化学习[43]思路, 引入性能激励[44]机制, 将小模型的实际表现作为奖励指标, 引导大语言模型在特征维度筛选与训练数据生成上实现自我迭代. 图 1 展示了 RADOM 的主要技术架构, 主要分为以下 3 部分.

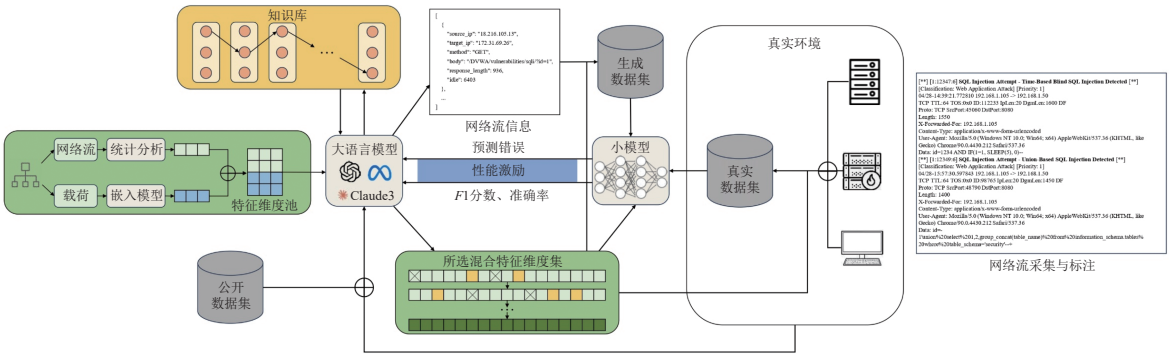

图 1　RADOM 技术架构图

(1) 基于性能激励的特征维度迭代优化算法: 为提升后续训练模型对网络攻击行为的检测能力, 该算法利用大语言模型从语义与数据样例角度筛选出能产生正向贡献的特征维度, 并将每轮测试的 $F1$ 分数与准确率作为结果反馈给大语言模型进行迭代, 从而确定最合适的特征维度集.

(2) 基于检索增强的网络流量数据生成算法: 由于网络流量数据并非大语言模型擅长处理的自然语言数据, 该算法融合检索增强思想, 通过预设模板与网络攻击行为相关信息引导大语言模型生成适配的网络流量训练数据.

(3) 基于数据驱动蒸馏的模型训练与迭代算法: 该算法使用大语言模型生成的训练数据集与挑选出的特征维

度集对小模型进行训练, 并将小模型投入实际环境进行检测. 小模型在实际环境中产生的错误预测将反馈给大语言模型, 大语言模型据此迭代生成新的针对性数据集, 补充至小模型训练集, 以迭代提升效果.

2.1 模型优化理论分析

本节将给出 RADOM 的数学表示与理论分析, 阐明该方法如何基于性能激励实现小模型性能提升. 在此将小模型于给定入侵检测任务上的表现表示为公式 (1):

$$P = f_\phi(F, D, D_{\text{target}}) \tag{1}$$

其中, ϕ 代表模型结构, D 代表使用的训练数据集, F 代表模型输入的特征维度集, D_{target} 代表模型面对的测试数据集或真实环境. 在不改变模型结构的前提下, 模型训练的目标可以被总结为公式 (2):

$$\arg\max_{F,D} f_\phi(F, D, D_{\text{target}}) \tag{2}$$

本方法要求大语言模型基于小模型在 D_{target} 上的表现, 不断迭代修正 F 和 D. 一般地, 大语言模型在每轮迭代中的工作可表示为公式 (3)、(4):

$$F_{t+1} = LLM(prompt_F, S_t^F, S_t^{(F1,acc)}, f_\phi(F_t, D_t, D_{\text{target}})) \tag{3}$$

$$D_{t+1} = LLM(prompt_D, S_t^D, S_t^\varepsilon, f_\phi(F_t, D_t, D_{\text{target}})) \tag{4}$$

其中, S_t^F、S_t^D、$S_t^{(F1,acc)}$、S_t^ε 分别代表分别表示从初始状态到第 t 次迭代为止的特征维度集合序列、数据集序列、$F1$ 值与准确率序列以及预测误差序列. 理想情况下, 同时优化 F 和 D 将得到全局最优解. 然而, 这在实践中难以实现. 一方面, F 和 D 的迭代所需性能度量不同, 基于两类度量 ($(F1, acc)$ 与 ε) 同时迭代会超出大语言模型的处理能力. 另一方面, 仅就 F 寻优, 其解空间为组合数 $\binom{T}{X}$, 其中 T 为初始统计特征维度池的规模, X 为最终特征维度集的规模; 若在未固定 X 的情况下同时调整 F 和 D, 解空间将进一步增大. 考虑到计算效率约束, 使用大语言模型穷举探索以同时为 F 和 D 找到全局最优在实践中不可行. 为简化起见, 本文采用坐标下降法: 先固定训练数据集 D 并优化特征空间 F, 对应于“特征维度优化”阶段 (第 2.2 节).

$$F^* = \arg\max_F f_\phi(F, D_0, D_{\text{target}}) \tag{5}$$

随后, 在给定已获得的最优特征维度 F^* 的条件下迭代更新数据集 D, 以最小化性能差距, 对应于“模型训练与迭代”阶段 (第 2.4 节).

$$D^* = \arg\max_D f_\phi(F^*, D, D_{\text{target}}) \tag{6}$$

最终得到的特征维度集 F^* 与数据集 D^* 将作为下游模型训练的最优配置. 为保证迭代能够稳定收敛, 本文的所有迭代过程中均采用检查点机制, 即只保留带来性能提升的改进, 当迭代后性能下降时, 下一轮迭代将从最后一个性能提升节点开始而非当前节点. 后续实验结果证明了坐标下降法的有效性.

2.2 基于性能激励的特征维度迭代优化算法 (PM-FeDO)

网络流量信息通常以 pcap (packet capture, 指由 Wireshark 或 tcpdump 等工具捕获的网络流量数据所采用的通用标准格式) 文件格式存储. 为将其转换为适合模型输入的向量表示, 需要一组恰当的特征维度作为数据嵌入标准. 算法 1 展示了本文提出的基于性能激励的特征维度迭代优化算法——PM-FeDO 算法.

算法 1. 基于性能激励的特征维度迭代优化算法 (PM-FeDO 算法).

输入: 原始参考数据集 D_{origin}, 初始统计特征维度集 F, 文本嵌入模型 E, 嵌入特征维度数 d, LLM, 迭代轮数上限 T;

输出: 选择特征维度集 F^*.

1. 从 D_{origin} 中抽取请求体, 并借助 E 编码成嵌入向量集 δ
2. 使用 PCA 算法将 δ 降维至 d 维, 得到和对应的维度集 F_δ

3. 从 D_{origin} 中抽取一个正负样本均衡的样本集 S
4. $F_0 \leftarrow LLM(prompt, F, S)$
5. 将 F_0 和 F_δ 拼接得到初始特征维度集 X_0
6. **for** t = 1 to T **do**
7. 使用 X_{t-1} 训练和评估小模型, 获得指标 $(F1, acc)$
8. $F_t \leftarrow LLM(prompt, F_{t-1}, F1, acc)$
9. 将 F_t 和 F_δ 拼接得到初始特征维度集 X_t
10. **if** $(F1, acc)$ 达到要求 **then**
11. **break**
12. **end if**
13. **end for**
14. **return** $F^* \leftarrow F_t$

网络流量所包含的信息可分为两类: 流量载荷与流量统计. 因此, 本次选取的特征维度集也将同时涵盖这两部分. 对于流量载荷, 本方法选取合适的文本嵌入模型将网络请求体转换为嵌入向量, 并采用 PCA 方法[45]将维度降至所需规模. 对于流量统计, 本方法参考 CSE-CIC-IDS2018 数据集给出的特征维度集合, 该集合提供了较为全面的流量统计特征, 但并非所有维度都适用于分析 Web 攻击行为. 因此, 本方法使用大语言模型对该特征集合进行精炼, 主要从两个方面展开: 特征维度的语义信息与数据集的判别能力.

PM-FeDO 算法将预设提示词 (模板见图 2, 其中中括号内的部分为用户填写内容)、特征维度的语义信息以及相应的样例数据一并输入大语言模型, 并要求模型筛选出满足要求的特征维度集合. 随后, 将精炼后的流量统计特征与降维后的载荷嵌入向量进行拼接, 构建初始的网络流量特征维度池. 为最大化特征维度集的判别力, PM-FeDO 算法还引入了迭代机制. 对于每一个得到的网络流量特征维度集, 使用固定数据集进行训练与预测; 将得到的 $F1$ 值与准确率反馈给大语言模型, 由其决定在特征维度集中增删维度. 该循环反复执行, 直至达到预期性能或触及交互迭代次数上限; 最终得到的特征维度集将作为后续数据处理与模型训练的标准.

I need to optimize the input feature dimension set for my IDS (Intrusion Detection System) model. I currently have an initial pool of feature dimensions. To further improve detection performance, I also plan to incorporate several PCA-reduced embedding dimensions.

You will be given:
·The index, name, and semantic meaning of each feature in the initial feature pool;
·A small reference dataset for analysis.

Your task is to select a subset of feature dimensions that are most suitable for identifying {ATTACK TYPE}, based on semantic relevance and inspection of the reference dataset. The selected feature set will then be concatenated with the {EMBEDDING DIMENSIONS} PCA-reduced embedding dimensions to form the complete feature set for the model.

I will evaluate the selected feature set and return performance metrics including F1-score and accuracy. After each feedback round, you must decide whether to add or remove any feature dimensions to improve performance.

Your output should be an array containing the indices of the selected features.

Reference Information:
Attack description: {INPUT THE DESCRIPTION HERE}
Initial feature pool: {FILE}
Sample dataset: {FILE}

图 2 特征维度优化提示词模板

2.3 基于检索增强的网络流量数据生成算法 (RA-Flow)

初始训练数据集的生成面临两个主要挑战. 第一, 网络流量数据通常以结构化或半结构化的形式存储, 而非大语言模型所擅长处理的自然语言格式. 第二, 网络安全领域相关的背景知识在已有模型中的覆盖可能较为有限, 从而导致生成数据的泛化能力不足.

为解决第 1 个问题, 本文提出了一种基于自然语言交互的数据生成方法. 通过预定义提示, 引导模型输出包含网络流量关键信息的 JSON 结构体, 内容包括 IP 地址、端口、请求方法、载荷、模拟响应内容、响应长度及空闲时间等. 随后, 利用网络流量模拟器将该结构转换为 pcap 文件, 并进行数据清洗以剔除重复流量.

针对第 2 个问题, 本文引入检索增强生成机制, 以替代依赖 fine-tuning[46]与提示词工程[47]的传统方案, 从而在不增加训练开销的前提下提升数据生成质量. 算法 2 展示了本文提出的基于检索增强的网络流量数据生成算法: RA-Flow 算法, 其使用的提示词模板如图 3 所示. RA-Flow 算法将一次完整攻击划分为多个阶段, 并在知识库中针对每一阶段存储相关内容, 包括攻击技术、阶段描述、可能使用的示例载荷片段及阶段之间的继承关系等. 在数据生成过程中, 检索模块从知识库中提取覆盖各阶段的攻击链, 并通过预设提示输入模型, 进而生成高质量的训练样本.

```
Now I want you to expand my network traffic logs to help train my IDS (Intrusion
Detection System). The attack to be simulated is {ATTACK TYPE}, which
consists of {PHASE NUM} attack phases.

For each phase, I will provide:
·The name of the phase
·A description of the phase
·Sample payloads used during this phase

Your goal is to output a complete simulated attack chain that includes all phases,
and must take place within a single TCP conversation. Between each request,
introduce a random time interval (in seconds) to simulate the attacker's
preparation time. You should explicitly show the idle time in seconds between
requests.

The output format should follow this JSON structure:
[
  {
    "method": "GET",
    "body": "http://192.168.1.207:8080/DVWA/?id=1",
    "response_length": 939,
    "idle": 3
  },
  {
    "method": "POST",
    "body": "http://192.168.1.207:8080/DVWA/",
    "data": {
      "username": "admin",
      "password": "12345",
      "submit": "Submit"
    },
    "response_length": 939,
    "idle": 4
  },
  ...
]
Your task is to generate {TARGET NUM} full sequences of such data (i.e.,
{TARGET NUM} complete attack chains). The example above is for reference
only.

You are encouraged to:
·Vary the payloads  based on the attack type provided.
·Randomize the response_length (server response size).
·Vary the number of actions per attack chain — for example, one chain might
include 1 action, another might include 7. Avoid using the same number across all
chains.

Your final answer should be a single large JSON array containing all generated
records, and it should be returned via file download.

Reference Information:
Attack description: {INPUT THE DESCRIPTION HERE}
Phase 1 - Name: {PHASE NAME}
Phase 1 - Description: {PHASE DESCRIPTION}
Phase 1 - Sample Payloads: {SAMPLE PAYLOAD}
Phase 2 - Name: {PHASE NAME}
Phase 2 - Description: {PHASE DESCRIPTION}
Phase 2 - Sample Payloads: {SAMPLE PAYLOAD}
... (continue for all phases)
```

图 3　初始训练数据集生成提示词模板

算法 2. 基于检索增强的网络流量数据生成算法 (RA-Flow 算法).

输入: 攻击阶段集 $P=\{P_1,P_2,\ldots,P_n\}$, 知识库 K, LLM;

输出: 初始训练数据集 D_0.

1. 在知识库 K 中检索从 P_1 到 P_n 所有可能的攻击链路, 得到攻击链路集 C
2. $D_0 \leftarrow \varnothing$
3. **for** each $c \in C$ **do**
4. $\quad D_0 \leftarrow D_0 \cup LLM(prompt, c)$
5. **end for**
6. **return** D_0

2.4 基于数据驱动蒸馏的模型训练与迭代算法 (D3-MoTI)

算法 3 给出了基于数据驱动蒸馏的模型训练与迭代算法——D3-MoTI 算法的具体流程. 在获得初始训练数据集的基础上, 首先对选定的小模型进行训练, 并将训练后的模型部署于真实环境中进行测试. 随后, 收集模型在实际环境中产生的预测误差, 并将其作为反馈信息输入至模型, 引导其基于预测错误的网络行为生成更多相似数据, 以供后续训练使用. 后文图 4 给出了 D3-MoTI 算法使用的提示词模板.

算法 3. 基于数据驱动蒸馏的模型训练与迭代算法 (D3-MoTI 算法).

输入: 初始训练数据集 D_0, 小模型 M, 知识库 K, LLM, 真实数据集 D_r;
输出: 小模型 M.

1. $D \leftarrow D_0$
2. **for** t = 1 to T **do**
3. $\quad$ 在 D 上训练 M
4. $\quad$ 在 D_r 上测试 M, 并收集预测错误集 Err
5. $\quad$ **for** each $e \in Err$ **do**
6. $\quad\quad$ 补充数据集 $A_e \leftarrow LLM(prompt, e)$
7. $\quad\quad$ **if** e 为漏报 **then**
8. $\quad\quad\quad$ 在 K 中检索 e 所属的方法 m
9. $\quad\quad\quad$ **if** m 不存在 **then**
10. $\quad\quad\quad\quad$ 从 e 中抽取新的知识并添加到 K 中
11. $\quad\quad\quad$ **end if**
12. $\quad\quad$ **end if**
13. $\quad$ **end for**
14. $\quad D \leftarrow D \cup \{\bigcup_e A_e\}$
15. **end for**
16. **return** M

当预测误差属于漏报 (false negative) 时, 模型会被指示通过检索模块查询知识库中是否存在对应的攻击链. 若未检索到相关攻击路径, 则将该新的攻击技术补充至知识库, 以增强后续数据生成的完整性和适应性. 每轮迭代中生成的新数据将与原始训练数据合并, 构成小模型用于下一轮训练的数据集. 通过这一迭代机制, 模型的检测性能得以持续优化.

3 实验评估

为验证有效性, 本节选择以 SQL 注入攻击检测为评估对象. 通常情况下, Web 攻击的响应策略与攻击行为类型密切相关, 若在检测的同时能够识别攻击类型, 便可快速采取针对性响应以降低危害. 因此, 现代入侵检测技术不仅关注是否存在攻击行为, 也关注对攻击类型的识别. 基于此研究设定, 本次实验要求小模型还要对使

用的攻击方式进行分类. 实验在多种训练集与测试集组合下开展, 以评估本方法在不同条件下的表现. 本次实验采用 ChatGPT-4o 作为大语言模型, 同时为了体现对小模型选择的通用性并凸显迭代带来的性能差异, 本次实验选用一个 4 层全连接神经网络作为小模型 (每个隐藏层包含 50 个隐含单元, 使用 Adam 优化器, 学习率为 5E–4).

```
I have trained my IDS (Intrusion Detection System) using the data you generated,
and deployed the model in a real-world environment for testing. However, some
network traffic instances have been misclassified.

Your task is as follows:
·For each misclassified instance I provide, generate {REFLECT NUM}
corresponding supplemental training samples to help correct the model.
·If the instance is a false negative, check whether the attack method is already
present in the knowledge base.
·If it is not found in the knowledge base, you must analyze its technical
characteristics and return a new entry using the format I provide.

Supplemental training samples should be returned in the same format as the
original misclassified instance. If you discover a new attack method or technique,
return it using the following JSON structure:
{
  "phase": "phase name",
  "method": "method name",
  "request_method": "GET/POST/...",
  "description": "A brief explanation of the attack's behavior and how it differs
from known methods.",
  "sample_payloads": ["..."]
}

If you need to query the knowledge base, return the following JSON object:
{
  "type": "phase/method/technique"
}

The knowledge base will return a response structured as:
[
  {
    "name": "XXX",
    "description": "XXX"
  },
  ...
]

Your final output should include all generated supplemental training samples, and
any newly discovered attack knowledge (if applicable). It should be returned as a
downloadable file.
```

图 4　模型训练与迭代提示词模板

本次实验从 CSE-CIC-IDS2018 数据集中选取包含 Web 攻击行为的两个子集 (2018/02/22 星期四-Web 攻击与 2018/02/23 星期五-Web 攻击) 作为基线数据集. 然而, CSE-CIC-IDS2018 在 Web 攻击场景中存在明显的数据不平衡与攻击同质化问题. Zuec 等人[48]以及 Gopalan 等人[49]的工作指出, 上述两天内记录的 Web 攻击共计 928 条, 仅占全部攻击样本的 0.034%, 其在所选子集的流量中也仅占 0.044%. 其中, SQL 注入攻击最为严重, 正样本仅 87 条, 负正样本不平衡比高达 153911:1, 极大制约了模型的训练与预测性能. 为缓解上述问题并验证本方法在实际部署层面的可行性, 本次实验基于 SQLi-Labs 构建测试环境以采集 SQL 注入攻击流量作为正样本, 同时抓取正常通信流量作为负样本. 出于安全原因, 全部实验均在仿真环境中进行.

本次实验在一台搭载 Intel Xeon Silver 4210 处理器、64 GB 内存与 NVIDIA RTX A6000 显卡的服务器上进行, 操作系统版本为 CentOS 7.9.2009, Python 版本为 3.10.13.

3.1　特征维度优化

本次实验的第 1 个目标是通过特征维度优化选择合适的特征维度集. 按照 PM-FeDO 算法, 对于流量的载荷部分, 本次实验采用 e5-mistral-7b[50,51]作为文本嵌入模型, 该模型可将任意长度的文本输入映射为 4096 维嵌入向量. 出于训练效率考虑, 本次实验使用 PCA 将其输出结果降至最具判别力的 30 维.

对于流量统计特征, 本次实验采用 CSE-CIC-IDS2018 数据集提供的 77 个统计维度 (因协议与时间戳维度难以量化而予以排除) 作为候选特征池, 输入大语言模型的样例数据集由 100 条随机选取的正常流量与按原始类别分布比例抽样的 100 条攻击流量组成. 本方法将 77 个统计特征维度的定义及样例数据嵌入预设提示词中, 要求大

语言模型从中选择 30 个统计特征维度作为初始特征维度集的一部分.

首先评估初始特征维度集的判别效能. 本次实验在 CSE-CIC-IDS2018 数据集上比较了 3 种特征配置的表现: 原始的 77 维特征维度集、组合的 77+30 维特征维度集 (77 维统计特征+30 维 PCA 特征) 以及 30+30 维的优化特征维度集. 鉴于数据集中存在严重的类别失衡, 本次实验采用均衡采样策略, 确保每次迭代中正负样本数量相等; 最终结果在 5 次训练与测试后取平均. 实验结果见表 2.

表 2　训练集为 CSE-CIC-IDS2018 时, 3 种特征维度集在不同测试集上的表现

测试集	指标	特征维度集		
		77	77+30	30+30
CSE-CIC-IDS2018	*F*1值	0.11	1.00	1.00
	准确率 (%)	77.98	100.00	100.00
混合数据集	*F*1值	0.03	0.32	0.51
	准确率 (%)	21.06	35.26	52.70

通过比较 77 维与 77+30 维特征维度集的实验结果, 可观察到加入 30 维载荷嵌入向量后, 模型在原始 CSE-CIC-IDS2018 数据集上可实现完美分类. 这表明载荷嵌入对 Web 攻击行为检测具有正向贡献, 也侧面反映了 CSE-CIC-IDS2018 数据集中攻击模式的同质性. 然而, 鉴于 30+30 维特征维度集同样实现了完美分类, 上述结果不足以有效区分不同特征维度集的优劣. 考虑到 SQLi-Labs 环境涵盖更为多样的 SQL 注入攻击类型, 仅基于 CSE-CIC-IDS2018 训练的模型可能较难具备良好的泛化能力, 因此, 本次实验通过提高任务难度来检验泛化能力: 要求小模型仅在 CSE-CIC-IDS2018 数据集上训练, 而在由 CSE-CIC-IDS2018 与 SQLi-Labs 流量组成的混合数据集上进行测试. 为确保跨数据集的特征定义一致, 本方法在 SQLi-Labs 数据集以及后续由大语言模型生成的数据集上均使用 CICFlowMeter-V4.0[52]进行特征提取. 表 2 中的实验结果证明了上述假设的正确性.

在迭代阶段, 每次实验得到的 *F*1 值与准确率会反馈给大语言模型, 由其决定下一轮需要增加或删除的特征维度. 正如第 2.1 节所述, 穷举解空间不可行, 因此本方法将交互迭代上限设为 10 轮. 为验证该假设, 本方法共进行了 8 轮特征维度优化实验, 并对每一迭代步的 8 次结果取平均. 实验结果见图 5.

在图 5 中, “Default”表示使用完整的 77+30 维特征维度集得到的结果, 作为对照. 可以看到, 所有迭代的 *F*1 值与准确率均高于完整特征维度集 (列“Default”) 与初始 30+30 维特征维度集 (列“0”). 这表明, 相较完整特征维度集, 更紧凑且精炼的特征集合在 Web 攻击行为检测上反而能表现更好. 可以推测, 这一改进源于原始 77 个统计维度中存在冗余甚至有害的特征. 本次实验进一步采用 SHAP[53]分析各迭代得到的特征集合. 如图 6 所示, 所选特征的平均 SHAP 值随迭代呈上升趋势. 最终, 选择第 1 轮第 6 次迭代得到的 37 维流量统计特征 (参见表 3) 与 30 维载荷嵌入特征组合, 作为最终特征维度集.

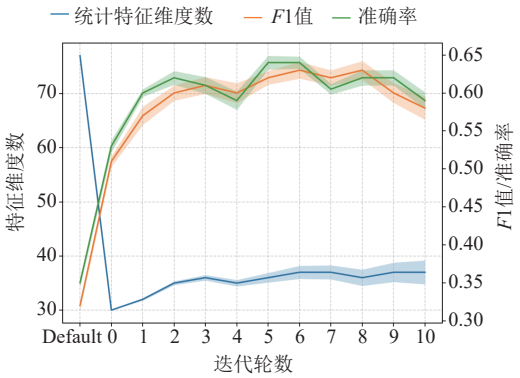

图 5　8 次特征维度优化的平均结果

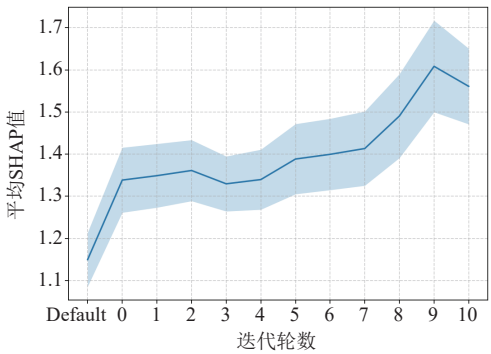

图 6　8 次特征维度优化的平均 SHAP 值变化趋势

表 3　最终选择的 37 维统计特征维度列表

序号	特征维度名称	特征维度含义
1	Dst Port	目标端口
2	Flow Duration	流持续时间 (以ms为单位)
3	Tot Fwd Pkts	前向总包数
4	Tot Bwd Pkts	后向总包数
5	TotLen Fwd Pkts	前向包总长度
6	TotLen Bwd Pkts	后向包总长度
7	Fwd Pkt Len Max	前向最大包长
8	Fwd Pkt Len Mean	前向平均包长
9	Fwd Pkt Len Std	前向包长标准差
10	Bwd Pkt Len Max	后向最大包长
11	Bwd Pkt Len Mean	后向平均包长
12	Bwd Pkt Len Std	后向包长标准差
13	Flow Byts/s	每秒传输字节数
14	Flow Pkts/s	每秒传输包数
15	Flow IAT Mean	相邻流之间平均时长 (以ms为单位)
16	Flow IAT Std	相邻流之间时长标准差
17	Fwd IAT Mean	前向相邻包之间平均时长 (以ms为单位)
18	Fwd IAT Std	前向相邻包之间时长标准差
19	Bwd IAT Mean	后向相邻包之间平均时长 (以ms为单位)
20	Bwd IAT Std	后向相邻包之间时长标准差
21	Fwd Header Len	前向包头总长度
22	Bwd Header Len	后向包头总长度
23	Down/Up Ratio	下载/上传比率
24	Init Fwd Win Byts	前向初始窗口字节数
25	Init Bwd Win Byts	后向初始窗口字节数
26	Fwd Act Data Pkts	前向流中至少包含1 Byte TCP载荷的包数
27	Active Mean	在进入空闲状态之前流活跃的平均时长 (以ms为单位)
28	Active Std	在进入空闲状态之前流活跃的时长标准差
29	Pkt Len Max	最大包长
30	Pkt Len Std	包长标准差
31	Bwd Pkt Len Min	后向最小包长
32	Bwd Seg Size Avg	观察到的后向数据包平均包长
33	Fwd Seg Size Avg	观察到的前向数据包平均包长
34	Subflow Fwd Byts	前向子流平均字节数
35	Subflow Fwd Pkts	前向子流平均包数
36	Subflow Bwd Byts	后向子流平均字节数
37	Subflow Bwd Pkts	后向子流平均包数

3.2 数据生成与迭代

本节将评估 RADOM 的核心部分: 基于数据驱动蒸馏实现小模型的训练与迭代. 第 1 步是生成初始训练数据集, 在该步骤中, SQL 注入攻击被划分为探测阶段与执行阶段. 表 4 汇总了攻击知识库中存储的攻击方法信息. 依据 RA-Flow 算法遍历所有可能的攻击链, 共生成 1 484 条 SQL 注入流量样本; 同时, 从 CSE-CIC-IDS2018 数据集中随机选择等量正常流量作为良性样本.

随后使用由大语言模型生成的初始训练数据集训练小模型, 并预测代表实际环境的 SQLi-Labs 数据集, 其中 SQLi-Labs 数据集中的攻击样本已按执行阶段方法完成标注. 为检验 RADOM 对先验未覆盖攻击的知识拓展能

力, 本次实验保留了一些未纳入知识库的技术. 表 5 给出了初始训练数据集 (与 CSE-CIC-IDS2018 的正常流量混合后) 与 SQLi-Labs 数据集中不同数据类型的占比.

表 4　SQL 注入攻击知识库中的攻击方法

探测阶段	执行阶段
尝试单/双引号	联合注入
尝试获取数据库列数	报错注入
尝试获取报错信息	时间盲注
尝试获取数据库元信息	表单注入 布尔盲注

表 5　初始训练数据集与 SQLi-Labs 数据集中不同攻击类型的数据统计

数据类型	初始训练数据集	SQLi-Labs数据集
正常流量	1 484	3 150
联合注入	297	120
报错注入	277	90
时间盲注	320	120
表单注入	280	180
布尔盲注	310	120
总计	2 968	3 780

在迭代阶段, 本次实验设定对每个预测错误由大语言模型生成 10 条相似的攻击流量样本. 图 7 中展示了大语言模型生成数据、正常数据与 CSE-CIC-IDS2018 中 SQL 注入样本的 t-SNE 可视化结果. 可以看到, 大语言模型生成的数据能够有效覆盖 CSE-CIC-IDS2018 中 SQL 注入样本的分布, 同时与正常数据保持清晰可分. 初始训练数据集与迭代过程的实验结果见图 8 与表 6 (所报告结果为 5 次独立实验的平均值). 同时, 本次实验采用随机森林算法、Bamber 等人[54]提出的 CNN-LSTM 杂交模型和 IDS Agent 作为 baseline, 其中将随机森林算法与 CNN-LSTM 杂交模型的训练集设为由大语言模型生成的初始训练数据集, 即与第 0 次迭代的实验配置一致, IDS Agent 选取相同的 ChatGPT-4o 大语言模型.

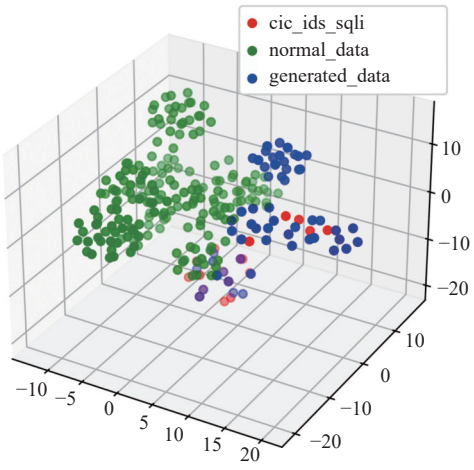

图 7　SQL 注入数据的 t-SNE 可视化结果

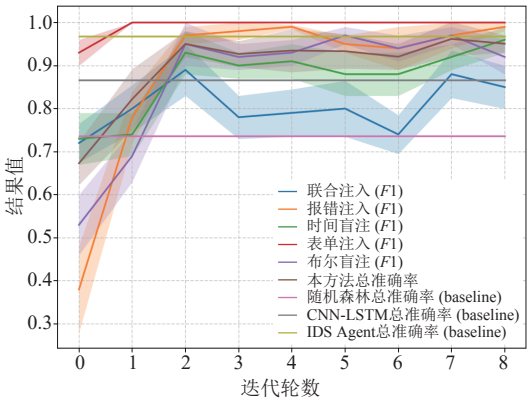

图 8　模型训练与迭代结果

表 6　迭代轮数不同时的模型训练结果

类别	0	1	2	3	4	5	6	7	8
联合注入 (*F*1)	0.72	0.80	**0.89**	0.78	0.79	0.80	0.74	0.88	0.85
报错注入 (*F*1)	0.38	0.78	0.97	0.98	**0.99**	0.95	0.94	0.97	**0.99**
时间盲注 (*F*1)	0.73	0.74	0.93	0.90	0.91	0.88	0.88	0.92	**0.96**
表单注入 (*F*1)	0.93	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
布尔盲注 (*F*1)	0.53	0.69	0.95	0.92	0.93	0.97	0.94	**0.97**	0.92
准确率 (%)	67.30	82.22	95.03	92.69	93.49	93.33	92.06	**96.19**	95.07

可以看到使用 RADOM 进行迭代训练的小模型在性能上取得显著提升. 在仅以结构简单的 4 层全连接网络作为小模型的情况下, SQL 注入检测与分类的准确率在两轮迭代内由 67.30% 提升至 95.03%, 在 7 轮迭代后达到 96.19%. 与 baseline 的对比也可以看出, 在第 0 次迭代中 4 层全连接网络的表现差于随机森林算法, 在第 2 轮迭代前表现差于 CNN-LSTM 杂交模型, 但在后续迭代过程中实现了反超, 这也证明了 RADOM 对小模型性能的显著提升. 可以发现 $F1$ 值与准确率并不严格单调, 在第 2 轮迭代后出现轻微波动, 可能源于过拟合, 这与全连接网络的容量与泛化能力有限有关. 另外, 小模型在后续迭代过程中实现了对表单注入攻击的完美分类, 本文推测其原因在于该类注入具有独特的数据特征, 即该类请求主要使用 POST 方法, 导致其载荷与其他攻击类型存在显著差异, 从而增强了嵌入式载荷特征维度的判别力并提升了分类准确率.

在迭代过程中, 本方法能够有效识别最初未收录于知识库的攻击技术, 并通过知识库更新将其纳入. 图 9 给出了一轮迭代中大语言模型响应中涉及新知识的片段. 经对知识库查询后, 大语言模型成功识别出反馈中包含的未收录的二次注入攻击, 并按规范生成包含相关信息的结构化输出.

大语言模型响应中涉及新知识的部分

```
** New Knowledge Detected **
{
  "phase": "Execution Phase",                                  // 来源阶段
  "method": "Form-based Injection",                            // 使用方法
  "request_method": "POST",                                    // 请求方式
  "description": "Unlike the sample payloads in the knowledge base, this attack features a
separation between the injection point and the execution point. The malicious payload was
not executed at the time of injection; instead, the attack was triggered when the system
later retrieved the payload from the database. This suggests that a second-order SQL
injection technique was likely employed.",                   // 攻击概况描述
  "sample_payloads": [                                         // 样例载荷
    {
      "username": "test'#",
      "password": "1234",
      "re_password": "1234",
      "submit": "Register"
    },
    {
      "username": "test'#",
      "password": "123456",
      "re_password": "123456",
      "submit": "Reset"
    }
  ]
}
```

图 9　在迭代中大语言模型关于新知识抽取的响应

3.3　性能分析

观察图 8 可以注意到, 在 8 次迭代过程中, IDS Agent 表现始终优于 RADOM, 但 RADOM 旨在在入侵检测模型的性能与效率之间取得更优的平衡, 并通过部署小模型的方式解决大模型直接参与流量监测带来的算力与时间开销问题.

在 IDS Agent 方法中, 大语言模型被直接部署于检测流程中, 对每一条网络流量样本执行一次完整推理, 并输出对应的检测或分类结果. 该方案的主要问题在于其在线推理开销较高, 难以满足入侵检测系统对实时性与吞吐量的要求. 在本次实验中, IDS Agent 在处理一条网络流量时, 平均消耗 1 263 个 token, 单次推理平均耗时 11.32 s. 这意味着当网络流量规模较大或检测节点资源受限时, 该方法将难以稳定运行; 此外该方法需要持续占用较高算力资源, 难以部署于防火墙、流量探针等边缘设备. 与上述方法不同, RADOM 将大语言模型的使用限制在离线阶段, 主要用于训练数据生成、特征维度优化及知识库更新等任务, 而将在线检测阶段完全交由小模型完成. 本次实验中 RADOM 生成一条训练数据并更新知识库平均消耗 162 个 token, 耗时 1.08 s. 在完成小模型的迭代训练后, 实际部署阶段仅需将训练完成的小模型投入运行, 本次实验中一个 4 层全连接神经网络占用显存 346 MB, 处理一条流量平均耗时 0.143 ms, 满足入侵检测系统实时性需求.

3.4　消融实验

为证明 RADOM 各关键组成部分的有效性, 本文进行了 3 项消融实验, 分别是去除特征维度优化机制、去除

初始训练数据集和去除预测错误反馈机制. 实验设置基本沿用前述的迭代生成实验, 区别在于去除特征维度优化机制实验将由 77 维统计特征与 30 维载荷嵌入特征组成的完整特征维度集作为小模型的输入; 去除初始训练数据集则不生成初始训练数据集, 让随机化的神经网络直接进行预测与迭代, 同时为了避免大量预测错误对后续迭代以及数据规模带来的影响, 本次实验设定每一轮次迭代最多反馈 600 条数据; 去除预测错误反馈机制实验中大语言模型不会根据小模型的预测错误生成针对性补充数据, 而是基于初始知识库生成固定数量数据 (本次实验设定为每一轮次迭代反馈 600 条数据), 实验结果如表 7 所示.

表 7　迭代轮数不同时的消融实验准确率结果 (%)

模型	0	1	2	3	4	5	6	7	8
去除特征维度优化机制	59.52	64.79	63.17	71.84	72.38	71.26	75.08	72.06	71.74
去除初始训练数据集	15.89	27.19	34.10	49.23	58.06	62.17	69.75	74.94	78.20
去除预测错误反馈机制	67.30	68.00	68.31	68.25	68.40	68.87	68.68	68.84	68.92
完整模型	67.30	82.22	95.03	92.69	93.49	93.33	92.06	96.19	95.07

可以观察到, 去除 3 个关键组成部分均带来了性能下降. 其中在不进行特征维度优化的情况下, 由于性能激励驱动的迭代效应, 整体性能同样呈上升趋势. 然而, 相较于采用优化后特征维度集的情形, 其提升速度以及多轮迭代后的性能均明显偏低; 去除初始训练数据集后, 初期迭代过程中由于训练数据集规模较小, 准确率较低, 但随着迭代次数的增加, 模型性能提升显著, 证明了 RADOM 对先验未覆盖攻击的自适应优化能力; 去除预测错误反馈机制后, 模型性能提升较为有限, 将该组实验与去除初始训练数据集组进行对照, 说明 RADOM 迭代过程中的性能提升主要来源于预测错误反馈机制.

3.5 低性能模型对比实验

本次实验所选用的 ChatGPT-4o 大语言模型对训练阶段提出了较高的资源要求, 实际部署环境可能无法支撑, 因此在模型训练与迭代阶段本次实验选择了开源模型 OpenAI-oss-120B[55]作为对比, 从而展示大语言模型性能下降对 RADOM 性能与迭代速度的影响, 实验结果如表 8 所示.

可以观察到随着使用的大语言模型性能下降, RADOM 的提升速度和多轮迭代后的性能均会下降, 在初始分类效果差的种类上更加明显 (如报错注入和联合注入). 对于使用低性能模型的 RADOM 性能如何提升的问题, 大模型量化[56]和多智能体协作[57]可能是可行方向.

表 8　迭代轮数不同时, 开源模型 OpenAI-oss-120B 的模型训练结果

类别	0	1	2	3	4	5	6	7	8
联合注入 ($F1$)	0.72	0.69	0.73	0.70	0.75	0.74	0.71	0.75	0.77
报错注入 ($F1$)	0.38	0.55	0.57	0.54	0.55	0.60	0.56	0.59	0.59
时间盲注 ($F1$)	0.73	0.78	0.85	0.79	0.79	0.80	0.81	0.78	0.81
表单注入 ($F1$)	0.93	0.99	1.00	1.00	0.99	1.00	1.00	1.00	1.00
布尔盲注 ($F1$)	0.53	0.67	0.74	0.78	0.76	0.81	0.80	0.79	0.82
准确率 (%)	67.30	72.61	79.78	76.21	76.84	78.54	76.19	78.15	79.80

4 总　结

本文提出一种基于数据驱动蒸馏与性能激励的入侵检测迭代模型 RADOM, 该模型通过大语言模型与小模型之间的结构化数据交互实现知识蒸馏, 并基于小模型预测错误的反馈迭代提升数据生成质量. 此外, 该模型包含特征维度优化机制, 借助大语言模型迭代精炼特征空间, 进一步提升小模型的分类性能.

本文以 ChatGPT-4o 作为大语言模型, 在 CSE-CIC-IDS2018 数据集与基于 SQLi-Labs 采集的自建数据集上进行了全面实验. 结果表明, 该模型显著增强了小模型的分类效能; 同时开展的消融实验进一步验证了特征维度优化机制、初始训练数据集与预测错误反馈机制的有效性. 本文仍存在以下几个需要进一步讨论的问题, 可作为未来

研究方向.

(1) RADOM 本身原理不与攻击类型绑定, 因此理论上可以应用于其他网络安全攻击检测, 将 RADOM 横向迁移至对其他种类攻击的检测工作中很有意义.

(2) 理论上 RADOM 在 F 和 D 上寻找全局最优解将带来进一步的性能提升, 但由于前文所述限制, 本文仍采用坐标下降法寻求局部最优. 如能实现 F 和 D 的联合全局优化将是对 RADOM 的一项重大改进;

(3) RADOM 对所选大语言模型的性能提出了较高要求, 这一点在实际应用过程中可能存在限制, 如何在使用更低性能的大语言模型的前提下保证模型最终性能是一个值得研究的问题.

References

[1] Lunt TF. A survey of intrusion detection techniques. Computers & Security, 1993, 12(4): 405–418. [doi: 10.1016/0167-4048(93)90029-5]

[2] Mandala S, Ngadi A, Abdullah AH. A survey on manet intrusion detection. Int'l Journal of Computer Science and Security, 2007, 2(1): 1–11.

[3] Liao HJ, Richard Lin CH, Lin YC, Tung KY. Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, 2013, 36(1): 16–24. [doi: 10.1016/j.jnca.2012.09.004]

[4] Denning DE. An intrusion-detection model. IEEE Trans. on Software Engineering, 1987, SE-13(2): 222–232. [doi: 10.1109/TSE.1987.232894]

[5] Atefinia R, Ahmadi M. Network intrusion detection using multi-architectural modular deep neural network. The Journal of Supercomputing, 2021, 77(4): 3571–3593. [doi: 10.1007/s11227-020-03410-y]

[6] Wang SS, Yan KQ, Wang SC, Liu CW. An integrated intrusion detection system for cluster-based wireless sensor networks. Expert Systems with Applications, 2011, 38(12): 15234–15243. [doi: 10.1016/j.eswa.2011.05.076]

[7] Sharafaldin I, Lashkari AH, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proc. of the 4th Int'l Conf. on Information Systems Security and Privacy (ICISSP 2018). Funchal: SciTePress, 2018. 108–116. [doi: 10.5220/0006639801080116]

[8] Cavusoglu H, Raghunathan S, Cavusoglu H. Configuration of and interaction between information security technologies: The case of firewalls and intrusion detection systems. Information Systems Research, 2009, 20(2): 198–217. [doi: 10.1287/isre.1080.0180]

[9] Gregorczyk M, Żórawski P, Nowakowski P, Cabaj K, Mazurczyk W. Sniffing detection based on network traffic probing and machine learning. IEEE Access, 2020, 8: 149255–149269. [doi: 10.1109/ACCESS.2020.3016076]

[10] Liu M, Xue Z, Xu XH, Zhong CM, Chen JJ. Host-based intrusion detection system with system calls: Review and future trends. ACM Computing Surveys (CSUR), 2019, 51(5): 98. [doi: 10.1145/3214304]

[11] Martins I, Resende JS, Sousa PR, Silva S, Antunes L, Gama J. Host-based IDS: A review and open issues of an anomaly detection system in IoT. Future Generation Computer Systems, 2022, 133: 95–113. [doi: 10.1016/j.future.2022.03.001]

[12] Lira OG, Marroquin A, To MA. Harnessing the advanced capabilities of LLM for adaptive intrusion detection systems. In: Proc. of the 38th Int'l Conf. on Advanced Information Networking and Applications. Cham: Springer, 2024. 453–464. [doi: 10.1007/978-3-031-57942-4_44]

[13] Hall M, Wiley K. Capacity verification for high speed network intrusion detection systems. In: Proc. of the 5th Int'l Symp. on Recent Advances in Intrusion Detection. Zurich: Springer, 2002. 239–251. [doi: 10.1007/3-540-36084-0_13]

[14] Yao YF, Duan JH, Xu KD, Cai YF, Sun ZB, Zhang Y. A survey on large language model (LLM) security and privacy: The good, the bad, and the ugly. High-confidence Computing, 2024, 4(2): 100211. [doi: 10.1016/j.hcc.2024.100211]

[15] Chen LH, Varoquaux G. What is the role of small models in the LLM era: A survey. arXiv:2409.06857, 2024.

[16] Audi-1. Sqli-labs. 2026. https://github.com/Audi-1/sqli-labs

[17] Lewis P, Perez E, Piktus A, Petroni F, Karpukhin V, Goyal N, Küttler H, Lewis M, Yih WT, Rocktäschel T, Riedel S. Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Proc. of the 34th Int'l Conf. on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2020. 793.

[18] Khraisat A, Gondal I, Vamplew P, Kamruzzaman J. Survey of intrusion detection systems: Techniques, datasets and challenges. Cybersecurity, 2019, 2(1): 20. [doi: 10.1186/s42400-019-0038-7]

[19] Meiners CR, Patel J, Norige E, Torng EK, Liu AX. Fast regular expression matching using small TCAMs for network intrusion detection and prevention systems. In: Proc. of the 19th USENIX Security Symp. Washington: USENIX Association, 2010. 285–300.

[20] Lin PC, Lin YD, Lai YC. A hybrid algorithm of backward hashing and automaton tracking for virus scanning. IEEE Trans. on Computers, 2011, 60(4): 594–601. [doi: 10.1109/TC.2010.95]

[21] Symantec Corporation. Internet security threat report, volume 22. 2017. https://docs.broadcom.com/doc/istr-22-2017-en

[22] Annachhatre C, Austin TH, Stamp M. Hidden Markov models for malware classification. Journal of Computer Virology and Hacking Techniques, 2015, 11(2): 59–73. [doi: 10.1007/s11416-014-0215-x]

[23] Liu XX, Zhu PD, Zhang Y, Chen K. A collaborative intrusion detection mechanism against false data injection attack in advanced metering infrastructure. IEEE Trans. on Smart Grid, 2015, 6(5): 2435–2443. [doi: 10.1109/TSG.2015.2418280]

[24] Fu CP, Li Q, Shen M, Zhang X, Liu Y, Xu K. Realtime robust malicious traffic detection via frequency domain analysis. In: Proc. of the 2021 ACM SIGSAC Conf. on Computer and Communications Security. ACM, 2021. 3431–3446. [doi: 10.1145/3460120.3484585]

[25] Zhou GM, Guo XW, Liu ZT, Li T, Li Q, Xu K. Trafficformer: An efficient pre-trained model for traffic data. In: Proc. of the 2025 IEEE Symp. on Security and Privacy (SP). San Francisco: IEEE, 2025. 1844–1860. [doi: 10.1109/SP61157.2025.00102]

[26] Ali T, Kostakos P. HuntGPT: Integrating machine learning-based anomaly detection and explainable AI with large language models (LLMs). arXiv:2309.16021, 2023.

[27] Li YJ, Xiang Z, Bastian ND, Song D, Li B. IDS-agent: An LLM agent for explainable intrusion detection in IoT networks. In: Proc. of the 38th Conf. on Neural Information Processing Systems. 2024.

[28] Cui TY, Lin XJ, Li SJ, Chen M, Yin QL, Li Q, Xu K. TrafficLLM: Enhancing large language models for network traffic analysis with generic traffic representation. arXiv:2504.04222, 2025.

[29] Ye JC, Gao JH, Li QT, Xu H, Feng JT, Wu ZY, Yu T, Kong LP. Zerogen: Efficient zero-shot learning via dataset generation. In: Proc. of the 2022 Conf. on Empirical Methods in Natural Language Processing. Abu Dhabi: ACL, 2022. 11653–11669. [doi: 10.18653/v1/2022.emnlp-main.801]

[30] Wang RD, Zhou WCS, Sachan M. Let's synthesize step by step: Iterative dataset synthesis with large language models by extrapolating errors from small models. In: Findings of the Association for Computational Linguistics: EMNLP 2023. Singapore: ACL, 2023. 11817–11831. [doi: 10.18653/v1/2023.findings-emnlp.791]

[31] Daneshvar SS, Nong Y, Yang X, Wang SW, Cai HP. VulScribeR: Exploring RAG-based vulnerability augmentation with LLMs. ACM Trans. on Software Engineering and Methodology, 2025. [doi: 10.1145/3760775]

[32] Buciluǎ C, Caruana R, Niculescu-Mizil A. Model compression. In: Proc. of the 12th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining. Philadelphia: ACM, 2006. 535–541. [doi: 10.1145/1150402.1150464]

[33] Hinton GE, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv:1503.02531, 2015.

[34] Gou JP, Yu BS, Maybank SJ, Tao DC. Knowledge distillation: A survey. Int'l Journal of Computer Vision, 2021, 129(6): 1789–1819. [doi: 10.1007/s11263-021-01453-z]

[35] Yang Y, Tian B, Yu F, He YH. An anomaly detection model training method based on LLM knowledge distillation. In: Proc. of the 2024 Int'l Conf. on Networking and Network Applications (NaNA). Yinchuan: IEEE, 2024. 472–477. [doi: 10.1109/NaNA63151.2024.00084.]

[36] Chen HT, Wang YH, Xu C, Yang ZH, Liu CJ, Shi BX, Xu CJ, Xu C, Tian Q. Data-free learning of student networks. In: Proc. of the 2019 IEEE/CVF Int'l Conf. on Computer Vision. Seoul: IEEE, 2019. 3513–3521. [doi: 10.1109/ICCV.2019.00361.]

[37] Jang Y, Lee H, Hwang SJ, Shin J. Learning what and where to transfer. In: Proc. of the 36th Int'l Conf. on Machine Learning. Long Beach: PMLR, 2019. 3030–3039.

[38] Stolfo S, Fan W, Lee W, Prodromidis A, Chan P. KDD Cup 1999 Data. 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[39] Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD Cup 99 data set. In: Proc. of the 2009 IEEE Symp. on Computational Intelligence for Security and Defense Applications. Ottawa: IEEE, 2009. 1–6. [doi: 10.1109/CISDA.2009.5356528]

[40] García S, Grill M, Stiborek J, Zunino A. An empirical comparison of botnet detection methods. Computers & Security, 2014, 45: 100–123. [doi: 10.1016/j.cose.2014.05.011]

[41] Lippmann R, Fried DJ, Graf I, Haines JW, Kendall KR, McClung D, Weber DJ, Webster SE, Wyschogrod D, Cunningham RK, Zissman MA. Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. In: Proc. of the 2000 DARPA Information Survivability Conf. and Exposition. Hilton Head: IEEE, 2000. 12–26. [doi: 10.1109/DISCEX.2000.821506]

[42] Moustafa N, Slay J. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: Proc. of the 2005 Military Communications and Information Systems Conf. (MilCIS). Canberra: IEEE, 2015. 1–6. [doi: 10.1109/MilCIS.2015.7348942]

[43] Sutton RS, Barto AG. Reinforcement Learning: An Introduction. Cambridge: MIT Press, 1998.

[44] Judge TA, Ilies R. Relationship of personality to performance motivation: A meta-analytic review. Journal of Applied Psychology, 2002, 87(4): 797–807. [doi: 10.1037/0021-9010.87.4.797]

[45] Maćkiewicz A, Ratajczak W. Principal components analysis (PCA). Computers & Geosciences, 1993, 19(3): 303–342. [doi: 10.1016/0098-3004(93)90090-R]

[46] Ouyang L, Wu J, Jiang X, Almeida D, Wainwright CL, Mishkin P, Zhang C, Agarwal S, Slama K, Ray A, Schulman J, Sutskever I.

Training language models to follow instructions with human feedback. In: Proc. of the 36th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2022. 2011.

[47] Shinn N, Cassano F, Gopinath A, Narasimhan K, Yao S. Reflexion: Language agents with verbal reinforcement learning. In: Proc. of the 37th Int'l Conf. on Neural Information Processing Systems. New Orleans: Curran Associates Inc., 2023. 377.

[48] Zuech R, Hancock J, Khoshgoftaar TM. Detecting SQL injection web attacks using ensemble learners and data sampling. In: Proc. of 2021 IEEE Int'l Conf. on Cyber Security and Resilience (CSR). Rhodes: IEEE, 2021. 27–34. [doi: 10.1109/CSR51186.2021.9527990]

[49] Gopalan SS, Ravikumar D, Linekar D, Raza A, Hasib M. Balancing approaches towards ML for IDS: A survey for the CSE-CIC IDS dataset. In: Proc. of the 2021 Int'l Conf. on Communications, Signal Processing, and their Applications (ICCSPA). Sharjah: IEEE, 2021. 1–6. [doi: 10.1109/ICCSPA49915.2021.9385742]

[50] Wang L, Yang N, Huang XL, Jiao BX, Yang LJ, Jiang DX, Majumder R, Wei FR. Text embeddings by weakly-supervised contrastive pre-training. arXiv:2212.03533, 2022.

[51] Wang L, Yang N, Huang XL, Yang LJ, Majumder R, Wei FR. Improving text embeddings with large language models. In: Proc. of the 62nd Annual Meeting of the Association for Computational Linguistics (Vol. 1: Long Papers). Bangkok: ACL, 2024. 11897–11916. [doi: 10.18653/v1/2024.acl-long.642]

[52] Ahlashkari. CICFlowMeter. 2026. https://github.com/ahlashkari/CICFlowMeter

[53] Lundberg SM, Erion G, Chen H, DeGrave A, Prutkin JM, Nair B, Katz R, Himmelfarb J, Bansal N, Lee SI. From local explanations to global understanding with explainable AI for trees. Nature Machine Intelligence, 2020, 2(1): 56–67. [doi: 10.1038/s42256-019-0138-9]

[54] Bamber SS, Katkuri AVR, Sharma S, Angurala M. A hybrid CNN-LSTM approach for intelligent cyber intrusion detection system. Computers & Security, 2025, 148: 104146. [doi: 10.1016/j.cose.2024.104146]

[55] Agarwal S, Ahmad L, Ai J, *et al*. gpt-oss-120b & gpt-oss-20b model card. arXiv:2508.10925, 2025.

[56] Zhao YL, Lin CY, Zhu K, Ye ZH, Chen LQ, Zheng SZ, Ceze L, Krishnamurthy A, Chen TQ, Kasikci B. Atom: Low-bit quantization for efficient and accurate LLM serving. In: Proc. of the 7th Annual Conf. on Machine Learning and Systems. Santa Clara, 2024. 196–209.

[57] He JD, Treude C, Lo D. LLM-based multi-agent systems for software engineering: Literature review, vision, and the road ahead. ACM Trans. on Software Engineering and Methodology, 2025, 34(5): 1–30. [doi: 10.1145/3712003]

作者简介

田润, 博士生, 主要研究领域为网络空间安全, 网络行为智能认知.

张海霞, 博士, 高级工程师, 主要研究领域为网络空间安全, 网络行为智能认知.

连一峰, 博士, 正高级工程师, 博士生导师, 主要研究领域为网络空间安全, 网络行为智能认知.

张立武, 博士, 正高级工程师, 博士生导师, 主要研究领域为网络空间安全, 网络信任体系.