

兰州理工大学

硕士论文

兰州理工大学图书馆

学校代号 10731

分 类 号 TP391

学 号 212081203018

密 级 公 开



兰州理工大学  
LANZHOU UNIVERSITY OF TECHNOLOGY

## 硕士学位论文

# 基于深度学习的恶意域名检测 方法研究

学位申请人姓名	吴喜川
培 养 单 位	计算机与通信学院
导师姓名及职称	马栋林 副教授
学 科 专 业	计算机应用技术
研 究 方 向	模式识别与人工智能
论文提交日期	2024 年 4 月 10 日

学校代号：10731

学 号：212081203018

密 级：公开

兰州理工大学硕士学位论文

基于深度学习的恶意域名检测  
方法研究

学位申请人姓名：	吴喜川
导师姓名及职称：	马栋林 副教授
培 养 单 位：	计算机与通信学院
专 业 名 称：	计算机应用技术
论文提交日期：	2024 年 4 月 10 日
论文答辩日期：	2024 年 5 月 20 日
答辩委员会主席：	火久元

Research on malicious domain name detection method based on  
deep learning

by

WU Xichuan

B.E. (Chongqing Jiaotong University) 2021

A thesis submitted in partial satisfaction of the

Requirements for the degree of

Master of Engineering

in

Computer Application Technology

in the

School of Computer and Communications

of

Lanzhou University of Technology

Supervisor

Associate Professor MA Donglin

May, 2024

# 目 录

第 1 章 绪论.....	1
1.1 研究背景与意义.....	1
1.2 国内外研究现状.....	2
1.3 研究内容.....	5
1.4 论文组织结构.....	6
第 2 章 相关理论与技术 .....	7
2.1 域名相关知识.....	7
2.1.1 域名.....	7
2.1.2 域名解析过程.....	7
2.1.3 恶意域名.....	8
2.1.4 恶意域名生成算法.....	8
2.2 向量化方法.....	9
2.2.1 One-hot 编码.....	9
2.2.2 Word2Vec 词向量编码.....	9
2.3 深度学习基础.....	10
2.3.1 卷积神经网络.....	10
2.3.2 长短期记忆网络.....	15
2.3.3 门控循环网络.....	16
2.3.4 自编码器网络.....	17
2.3.5 变分自编码器网络.....	18
2.3.6 损失函数.....	21
2.3.7 性能评价指标.....	22
2.4 本章小结.....	23
第 3 章 基于 CNN-BiGRU-AFF 的恶意域名检测 .....	24
3.1 引言.....	24
3.2 检测流程.....	24
3.2.1 域名数据预处理.....	24
3.2.2 关联特征提取.....	25
3.2.3 局部特征提取.....	26
3.2.4 全局特征提取.....	27
3.2.5 特征融合.....	27
3.2.6 域名分类.....	28
3.2.7 损失函数.....	28
3.3 实验设计与结果分析.....	29

3.3.1 数据集.....	29
3.3.2 实验环境及评价指标.....	29
3.3.3 对比实验.....	30
3.3.4 消融实验.....	31
3.4 本章小结.....	33
第 4 章 基于自适应深度变分自编码器的恶意域名检测 .....	34
4.1 引言.....	34
4.2 检测流程设计.....	34
4.2.1 域名数据预处理.....	35
4.2.2 深度变分自编码器训练.....	36
4.2.3 自适应阈值分类器构建.....	39
4.2.4 恶意域名识别.....	41
4.3 实验与分析.....	41
4.3.1 数据集.....	41
4.3.2 实验环境及评价指标.....	42
4.3.3 对比实验.....	43
4.4 本章小结.....	45
总结与展望.....	46
参考文献.....	48

## 摘要

通过域名系统（Domain Name System, DNS）实现 IP 地址与域名的映射，从而方便了用户对网络资源的访问。然而，DNS 的设计并未充分集成安全防护措施，不能直接阻止网络攻击者利用恶意域名传播恶意软件、盗取敏感数据等非法行为。这些行为严重侵害了用户的数据安全和隐私，因此对恶意域名进行精确的识别，并采取恰当的防御措施，对保障网络用户的安全具有重要的作用。

现有的恶意域名检测模型已取得了较好的效果，但仍存在一些不足。其一，现有模型提取的域名特征单一，特征利用率低，从而影响检测精度；其二，现有模型对新出现或新变种的恶意域名泛化性差。针对上述问题，本文进行了探索和研究，主要的内容包括：

（1）提出了基于 CNN-BiGRU-AFF 的恶意域名检测模型。首先注意力机制对向量化后的原始域名信息进行过滤，增强关键特征的向量表示，除去冗余信息；其次卷积神经网络（Convolutional Neural Network, CNN）与双向门控循环单元（Bidirectional Gated Recurrent Unit, BiGRU）并行，分别提取域名的局部特征和全局特征；然后通过注意力特征融合（Attentional Feature Fusion, AFF）模块对域名的两种特征进行有效融合；最后通过全连接层和 softmax 函数获得域名分类的概率分布。为了检验模型性能，构建了两种数据集进行测试，结果表明，CNN-BiGRU-AFF 模型能多维度提取域名特征，从而有效提高了域名特征的利用率。

（2）有监督的深度学习模型难以快速捕获新或变异的恶意域名，因为恶意域名通常在发起攻击后才被加入黑名单，从而导致模型泛化性差。因此提出了基于自适应深度变分自编码器的恶意域名检测模型 Ada-DVAE（Adaptive Deep Variational Auto-encoder）。首先构建并训练深度变分自编码器，并进行参数微调，以获得精确的域名重建概率；然后利用域名重建概率构建自适应阈值分类器；最后使用分类器将恶意分数大于阈值的域名识别为恶意域名。通过组合多种数据集对性能进行验证，实验表明，本模型能有效缓解现有模型对新出现或新变种的恶意域名泛化能力差的问题。

**关键词：**恶意域名检测；卷积神经网络；深度变分自编码器；注意力机制

## Abstract

The Domain Name System (DNS) facilitates the mapping of IP addresses to domain names, thereby enabling convenient access to network resources for users. However, the design of DNS does not fully integrate security measures, thus it cannot directly prevent malicious actors from using malicious domain names to distribute malware, steal sensitive data, and engage in other illegal activities. These actions severely compromise user data security and privacy. Therefore, accurate identification of malicious domain names and implementation of appropriate defense measures play a crucial role in safeguarding the security of network users.

Existing models for malicious domain name detection have achieved good results but still have some shortcomings. First, the domain name features extracted by existing models are singular and have a low utilization rate, affecting detection accuracy; second, existing models have poor generalization ability for newly emerged or variant malicious domain names. In response to these issues, this paper explores and studies the following main content:

(1) A malicious domain name detection model based on CNN-BiGRU-AFF is proposed. First, the attention mechanism filters the vectorized original domain name information, enhancing the vector representation of key features and removing redundant information. Second, the Convolutional Neural Network (CNN) and Bidirectional Gated Recurrent Unit (BiGRU) are used in parallel to extract local and global features of domain names, respectively. Then, the Attentional Feature Fusion (AFF) module effectively integrates the two types of features of the domain name. Finally, the domain name classification probability distribution is obtained through a fully connected layer and softmax function. To test the model's performance, two types of datasets were constructed for testing, showing that the CNN-BiGRU-AFF model can extract domain name features in multiple dimensions, thereby effectively improving the utilization rate of domain name features.

(2) Supervised deep learning models struggle to quickly identify new or mutated malicious domain names since these domains are often only added to blacklists after launching attacks, resulting in poor model generalization. Thus,

the Adaptive Deep Variational Autoencoder (Ada-DVAE) model for malicious domain name detection is proposed. Initially, a deep variational autoencoder is constructed and trained, followed by parameter fine-tuning to obtain precise domain name reconstruction probabilities. Subsequently, an adaptive threshold classifier is developed using the domain name reconstruction probabilities. Finally, the classifier is used to identify domain names with malicious scores above the threshold as malicious. Performance validation through the combination of multiple datasets indicates that this model effectively mitigates the issue of existing models' poor generalization capabilities for newly emerged or mutated malicious domain names.

**Key words:** Malicious Domain Name Detection; Convolutional Neural Network; Deep Variational Auto-encoder; Attention Mechanism

## 附图索引

图 1.1 国内外研究现状 .....	2
图 1.2 论文章节结构 .....	6
图 2.1 域名解析过程 .....	7
图 2.2 CBOW 模型结构 .....	10
图 2.3 Skip-gram 模型结构 .....	10
图 2.4 卷积操作示意图 .....	12
图 2.5 Sigmoid 函数 .....	12
图 2.6 Softmax 函数 .....	13
图 2.7 Tanh 函数 .....	14
图 2.8 池化过程 .....	14
图 2.9 全连接层结构 .....	15
图 2.10 LSTM 结构 .....	15
图 2.11 GRU 结构 .....	16
图 2.12 自编码器网络结构 .....	18
图 2.13 变分自编码器网络结构 .....	19
图 3.1 基于 CNN-BiGRU-AFF 的恶意域名检测流程 .....	24
图 3.2 数据集域名长度统计 .....	25
图 3.3 关联信息提取过程 .....	26
图 3.4 域名局部特征提取 .....	26
图 3.5 域名全局特征提取 .....	27
图 3.6 特征融合模块 .....	28
图 3.7 串联模型和并联模型 AUC .....	33
图 4.1 基于自适应深度变分自编码器的恶意域名检测流程 .....	35
图 4.2 域名向量化过程 .....	36
图 4.3 域名长度分布 .....	37
图 4.4 数字字符比率分布 .....	37
图 4.5 分隔符内最大字符串长度分布 .....	38
图 4.6 信息熵分布 .....	38
图 4.7 参数微调 .....	39
图 4.8 训练集对准确率和 F1 值的影响 .....	44
图 4.9 样本数量少的恶意域名家族检测结果比较 .....	44

## 附表索引

表 2.1 变分自编码器构建算法 .....	20
表 2.2 混淆矩阵 .....	22
表 3.1 DN1 数据集划分 .....	29
表 3.2 实验环境配置 .....	29
表 3.3 实验超参数设置 .....	30
表 3.4 基准模型 .....	30
表 3.5 DN1 数据集对比实验结果 .....	31
表 3.6 DN2 数据集对比实验结果 .....	31
表 3.7 消融实验结果 .....	32
表 4.1 域名特征 .....	36
表 4.2 数据集描述 .....	41
表 4.3 数据集划分 .....	42
表 4.4 实验环境配置 .....	42
表 4.5 实验超参数设置 .....	42
表 4.6 对比实验结果 .....	43

# 第1章 绪论

## 1.1 研究背景与意义

在互联网中，每个设备都分配了具有唯一性的 IP 地址作为标识符。目前主要的 IP 地址形式包括 IPv4 和 IPv6 两种，然而 IP 地址缺乏直观的含义，不易直接记忆。因此采用域名代替 IP 地址，以使用户更便捷地访问相同的网络资源<sup>[1]</sup>。域名在互联网中的应用不仅限于合法用途，还可能被用作执行网络诈骗、传播恶意软件、盗取敏感数据等非法活动，这对网络和个人信息安全构成了严重的威胁。在 2023 年，根据国家应急响应中心发布的数据，国内面临的网络安全问题中，与恶意域名相关的问题凸显。从数量分布来看，我国境内植入并传播木马或僵尸网络排名前五的恶意域名家族主要是：AgentTesla、Mirai、RedLineStealer、Formbook、Amadey。这五个家族的木马或者僵尸网络感染主机的数量约占有被感染主机总数的 78.7%。从地区分布来看，受感染主机主要分布在广东省（占感染数量 14.1%）、河南省（占 12.3%）、山东省（9.2%）等省份。恶意域名对网络安全造成巨大威胁，准确地检测恶意域名已经成为一个重要的研究课题<sup>[2]</sup>。

恶意域名检测是指识别并标记那些用于网络钓鱼、分布恶意软件、进行欺诈活动或其他恶意行为的域名<sup>[3]</sup>。互联网作为一个多模态信息载体，涵盖丰富的信息源，使得恶意域名检测成为一个极具挑战性的任务。早期的恶意域名检测工作主要依赖于安全专家的手动分析和规则设定，这种方法不仅耗时耗力，而且难以适应网络环境的快速变化<sup>[4]</sup>。随着互联网中域名数量的剧增，仅依靠人工的方式已难以应对。近年来，随着人工智能技术的发展，结合深度学习方法进行恶意域名检测已成为重要的研究方向。该方法能够有效克服人工设计特征的局限性，提高检测的效率<sup>[5]</sup>。

目前，深度学习技术在恶意域名检测已取得较好的表现。研究者通过利用卷积神经网络<sup>[6]</sup>、长短期记忆网络<sup>[7]</sup>和 Transformer 网络<sup>[8]</sup>等深度学习模型，显著提升了对恶意域名的识别效率，从而证明了深度学习技术在恶意域名检测中的有效性。

恶意域名检测任务的目标是通过对域名的综合分析，准确地得出域名的恶意程度评估，使得安全团队能够基于这些评估及时做出决策，从而保障网络环境的安全。随着恶意域名生成算法的不断演化，单一维度的分析已难以满足现代网络安全的需求。基于深度学习的恶意域名检测模型，能够从多维度学习域名特征，从而实现对恶意域名进行有效识别。精确地识别并拦截恶意域名，可以提升网络

安全防护的整体效能，对维护网络安全具有重大的现实意义。

综上所述，恶意域名检测是网络安全领域中重要的研究内容，在保护网络环境、预防网络攻击等多个方面发挥着重要的作用。恶意域名检测是一个充满挑战的任务，现有的研究成果仍未完全满足网络安全需求。基于深度学习技术在前期恶意域名检测中的优异表现，本文利用 CNN、变自编码器等深度神经网络，进行恶意域名检测任务的研究，提升了检测的精度和缓解了现有模型对新出现、新变种的恶意域名泛化性差的问题。这项工作具有一定研究意义。

## 1.2 国内外研究现状

随着恶意域名变体的增加、复杂性的提高，恶意域名检测领域的研究工作也持续更新。目前，恶意域名检测的研究主要有三种核心技术：基于域名黑名单的检测技术，利用预先定义的恶意域名列表进行匹配识别；基于机器学习方法的恶意域名检测，通过分析域名的特征来自动识别恶意性；采用深度学习技术的恶意域名识别，通过构建复杂的模型来自动提取域名特征，从而提高检测精度<sup>[9,10]</sup>，如图 1.1 所示。

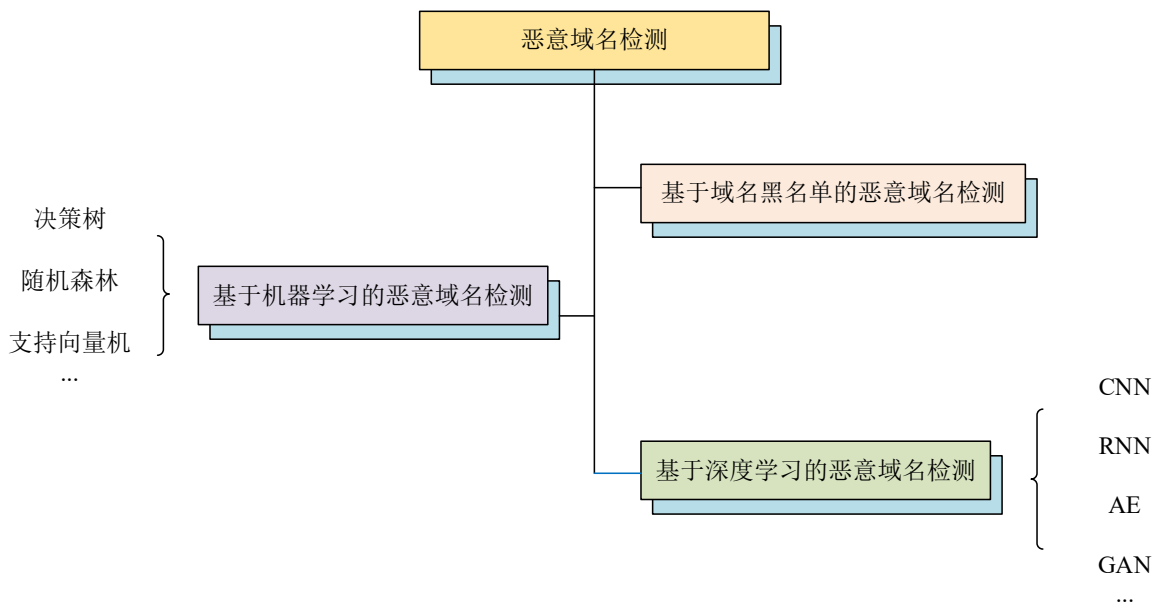


图 1.1 恶意域名检测技术

### （1）基于域名黑名单的恶意域名检测

基于域名黑名单的恶意域名检测方法通过分析历史的恶意域名信息特征，如 WHOIS 信息、域名的 IP 信息、域名已生存时间和域名 TTL（Time To Live）值等参数，建立域名黑名单用于识别恶意域名。这种方法通常需要人工定义特征规则，并通过分类或者聚类算法对待测域名进行判定。**WHOIS 信息**：如果具有模糊或不完整的历史记录，以及经常更换所有者或注册信息可能是恶意域名。在恶意域名的识别过程中，域名与 IP 地址之间的关联信息，特别是 IP 地址的历史记

录稳定性,具有重要的研究价值。具体而言,如果一个域名关联的 IP 地址历史记录短暂或出现频繁更换的现象,这可能暗示该域名具有恶意性质。此外,域名系统记录的生存时间的变化也是判定域名属性的一个关键因素。如不稳定或频繁变化的 TTL 值,尤其是在短时间内出现大幅度调整的情况,可能表明其为恶意域名。Saiyan 等<sup>[11]</sup>则通过匹配域名样本与黑名单中的字符,快速识别出潜在的恶意域名。赵宏等<sup>[12]</sup>提出了一种基于词特征分析的恶意域名识别技术。通过评估待测域名与已列入黑名单的域名之间的编辑距离,该方法能够快速识别出恶意域名,从而大幅度提高了检测流程的即时性。经过实验验证,该方法在准确率和推理时间上均表现出显著的优势。继之,Chen 等<sup>[13]</sup>使用有意义的分词描述 DGA 的组成,引入标准差以衡量词分布特征,并为分词构建额外的 11 维统计特征作为补充,然后,通过结合 3-gram 和 1-gram 序列特征,提高了对 DGA 恶意域名的检测性能。此外,Zhao 等<sup>[14]</sup>设计了一个两阶段的恶意域名检测框架,初步利用黑名单筛选域名,随后通过 N-Gram 模型对域名的信誉值进行评估,有效地区分了恶意域名与合法域名。

虽然通过对恶意域名与其历史特征数据库的匹配,以及对域名的多维特征分析,如长度、结构以及字符出现频率等,已经在恶意域名检测中取得了一定程度的成效,但是这些方法普遍存在对历史数据的过分依赖问题。这种依赖限制了模型在识别新出现或新变种的恶意域名时的准确性,无法及时过滤掉域名生成算法(Domain Generation Algorithm, DGA)家族恶意域名,从而对网络安全造成重大损失。因此,研究更加高效的恶意域名检测方法,对于保障网络环境的安全至关重要。

## (2) 基于机器学习的恶意域名检测

在恶意域名检测的研究领域中,机器学习方法被广泛应用于自动识别恶意域名<sup>[15,16]</sup>。常用于恶意域名检测的机器学习算法,如决策树、随机森林和支持向量机等,实验表明这些算法在恶意域名识别中表现优异<sup>[17,18]</sup>。例如,Cucchiarelli 等<sup>[19]</sup>针对算法生成的恶意域名,通过分析 2-gram 和 3-gram 特征的相似度,使用 Kullback-Leibner 散度和 Jaccard 系数作为判断依据,提升了特征提取的效率。针对随机性高的恶意域名,Alshdadi 等<sup>[20]</sup>通过加入网页特征,增强域名的特征表示,然后利用支持向量机将域名进行分类,提高了恶意域名的检测精度。Cheng 等<sup>[21]</sup>提出了一种基于 AdaBoost 的轻量级恶意域名检测方法,该方案通过分析异常 WHOIS 记录以主动识别恶意域名,增加了域名注册过程的安全性。该方法在处理大型数据集时显示了其有效性。马栋林等<sup>[22]</sup>则通过改进 Relief 算法对域名全局特征进行权重计算,选取权重最高的前 20 个特征构建分类器,简化了分类模型的计算复杂度,提高了恶意域名检测效率。王伟等<sup>[23]</sup>通过从网络流量中提取的 DGA 恶意域名,结合域名实体与自然语言特征,利用机

机器学习算法训练模型，大幅度提升了模型检测效率<sup>[24]</sup>。基于机器学习的方法在恶意域名的检测中表现出了高效的性能，检测效果明显优于基于黑名单的恶意域名检测方法，但其需要人工设计特征，随着恶意攻击技术的持续进化，特别是新变种恶意域名的出现，现有基于机器学习的模型会表现出性能退化<sup>[25,26]</sup>。

### （3）基于深度学习的恶意域名检测

通过建立复杂的深度神经网络模型，并利用大规模域名数据集进行自学习域名特征，缓解了传统模型需要手动提取特征的情，同时提升了检测精度<sup>[27,28]</sup>。例如引入深度神经网络<sup>[29]</sup>（Deep Neural Network, DNN）和生成对抗网络<sup>[30]</sup>（Generative Adversarial Network, GAN），增强了模型对域名的特征学习，从而模型能有效识别出更为复杂的恶意域名类型。

Highnam 等<sup>[31]</sup>提出了一种 Bilbo 的混合型神经网络模型，可以深入分析域名特征，同时评估这些域名由域名生成算法生成的概率。此模型特别针对不同类型的 DGA 恶意域名家族分类任务，表现了其较强的泛化能力。经过综合评价，包括 F1 得分和准确率在内的多项性能指标上，Bilbo 模型均实现了优异的表现，证明了其在恶意域名检测的有效性和可靠性。Qiao 等<sup>[32]</sup>基于 LSTM 具有处理字符串上下文特征的能力，提出了 LSTM 网络对 DGA 生成的恶意域名进行分类的方法，该方法提高了恶意域名检测的准确率。Yan 等<sup>[33]</sup>提出了一种基于 BiLSTM 的 DGA 恶意域名检测方法。通过双向学习域名信息，提取更加详细的域名上下文特征，从而获得更详细的域名特征学习表示，提高了对 DGA 恶意域名的检测精度。Kumar 等<sup>[34]</sup>提出使用深度神经网络的增强型 DGA 恶意域名识别模型，该模型融合了经典的手工统计特征和深度学习模型自动化提取的特征。在 DGA 恶意域名识别任务中，该模型通过实验验证，表现了其相对于传统方法如支持向量机和随机森林更优越的性能。进一步，Huang<sup>[35]</sup>成功构建了一种基于 TextCNN 的恶意域名识别模型，该模型优化了多种现有技术的优点，使得在恶意域名的检测过程中获得了更加准确的检测结果。

此外，Yadav 等<sup>[36]</sup>探索了一种基于 TextCNN 的轻量级模型，该模型在设计上注重减少训练参数的数量，从而有效降低了内存使用量，同时保持了高效的检测能力。Namgung 等<sup>[37]</sup>通过结合长短期记忆网络和卷积神经网络构建了一种 DGA 恶意域名检测模型，展现了深度学习在恶意域名检测方面相较于传统机器学习方法的明显优势。张清等<sup>[38]</sup>提出了 CNN-BiLSTM 模型，利用双层 LSTM 网络提取域名字符串在字节层的全局语义特征，使用 CNN 提取域名局部特征实验结果也验证了所设计模型的优越性。以上两种模型都是串联结构，没有真正实现多维度提取域名特征，依然提取比较单一的域名特征。王志强<sup>[39]</sup>等提出一种动态卷积算法，通过在模型词嵌入阶段采用字符嵌入和词嵌入的方法，实现了对域名中不常见的词语和新生成的域名单词有效表示，提高了后续对域名特征的利用

率,从而提高了模型对恶意域名的识别能力。邱颖豫等<sup>[40]</sup>为了解决训练样本不足导致的过拟合问题,提出了一种基于群体卷积神经网络的方法来检测恶意域名。该方法通过加入类间平衡系数来减轻过拟合现象,从而增强了模型的泛化能力,实现了恶意域名的检测任务。然后,Luo 等<sup>[41]</sup>基于图卷积网络和注意力机制提出了 AGCN-Domain 模型,首先将域名信息转换到二值图中,然后采用注意力机制提取关键的图特征,从而获得有效的域名特征表示,最后利用分类器对图片进行分类,实验结果验证了该模型在域名分类任务中的高效性能。

生成式模型也成功用于恶意域名检测。如周康等<sup>[42]</sup>提出了一种结合自编码器网络与长短期记忆神经网络的新型恶意域名识别技术。赵宏等<sup>[43]</sup>利用深度自编码网络提取域名字符统计特征,然后采用决策树算法将域名分类。均获得了良好的恶意域名检测效果。为了解决训练集数据不平衡的问题,Wang 等<sup>[44]</sup>,利用生成式神经网络,将恶意域名数量少的家族进行扩充,使模型能更好的学习该类家族的特征。在此基础上,Zhang 等<sup>[45]</sup>通过对域名生成算法的研究,结合生成式神经网络,模拟构造新出现和新变种的恶意域名,用来训练模型,以增强模型的泛化性。

### 1.3 研究内容

本文提出了基于 CNN-BiGRU-AFF 的恶意域名检测和基于自适应深度变分自编码器的恶意域名检测,主要研究内容如下:

#### (1) 基于 CNN-BiGRU-AFF 的恶意域名检测

针对现有恶意域名检测方法特征提取单一,特征利用不充分,导致模型检测性能不强的问题。提出了基于 CNN-BiGRU-AFF 的恶意域名检测模型。首先使用 Word2Vec 技术将域名向量化,并利用注意力机制对向量化后的原始域名信息进行过滤,除去冗余信息;其次采用 CNN 提取局部特征,BiGRU 提取域名全局特征;然后通过 AFF 模块对域名的两种特征进行有效融合,构成域名的有效特征表示,最后利用全连接层和 softmax 函数获得域名的分类结果。并采用焦点损失函数对模型进行优化。构建了两类不同的数据集进行实验,结果表明,本章模型具有较强的鲁棒性,有效了提高恶意域名的特征利用率。

#### (2) 基于自适应变分自编码器的恶意域名检测

现有监督的学习方法不能及时的检测出新出现或新变种的恶意域名。因为只有发生攻击后才能加入黑名单列表中,模型没有提前学习到其特征,从而导致模型泛化能力不强。为此提出了一种基于自适应深度变分自编码器的恶意域名检测方法。首先分析合法域名与恶意域名的特征存在明显差异,表明了本模型具有可行性。然后构建并训练深度变分自编码器,并进行参数微调,以增强对域名特征的学习。最后构建自适应阈值分类器,以待测域名的恶意得分作为分类判据,实

现域名的精确分类。通过结合不同数据集进行性能验证，实验结果证明，该模型有效缓解了现有模型对新出现或新变种恶意域名泛化性差的问题。

## 1.4 论文组织结构

本文在分析当前研究局限性的基础上，提出了两种恶意域名识别模型，并成功的实现。全文共分为四个主要部分，其组织结构如图 1.2 所示。

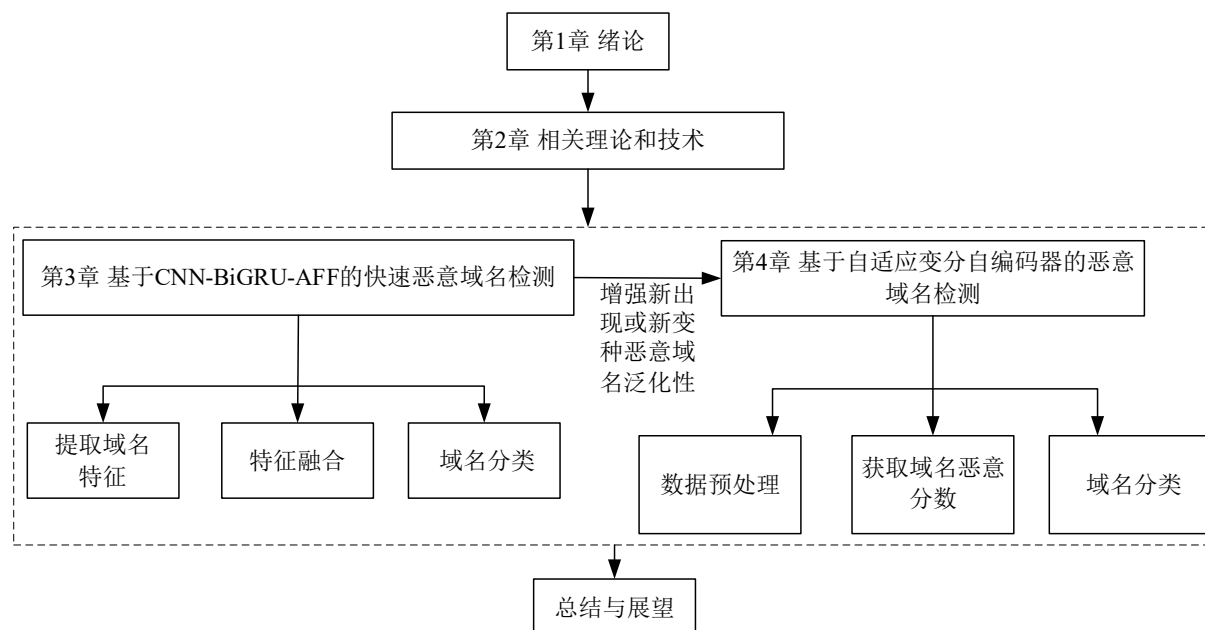


图 1.2 论文章节

第 1 章 绪论。本章深入阐述了研究主题的背景和重要性，概览了国内外在该领域的研究进展，并详细说明了本文的研究目标、内容以及结构安排。

第 2 章 相关理论和技术。本章主要介绍了与域名相关的基本知识、域名向量表示技术、基本的神经网络模型概念、常见的损失函数以及评估恶意域名识别算法性能的关键指标等内容。

第 3 章 基于 CNN-BiGRU-AFF 的恶意域名检测。介绍了 CNN-BiGRU-AFF 用于恶意域名检测的流程，然后对每个步骤进行了详细阐述，最后构建了两种不同的数据集对模型进行验证，实验表明该模型在恶意域名检测中表现优异。

第 4 章 基于自适应变分自编码器的恶意域名检测。针对现有模型对新生成或新变种恶意域名检测模型泛化能力不强的问题，提出了基于自适应变分自编码器的恶意域名检测。详细介绍了构建模型的每个步骤。最后通过大量的对比实验，结果表明，本模型能有效检测新出现和新变种的恶意域名。

总结与展望：针对所涉研究主题和相关成果进行了综述和总结，同时提出了当前的研究可以进一步改进的方向，并对未来的研究工作进行了展望。

## 第2章 相关理论与技术

### 2.1 域名相关知识

#### 2.1.1 域名

域名是设备在互联网中的标识<sup>[46]</sup>。在互联网中用于标记和定位网络资源，为用户提供了便捷的访问方式，同时也是互联网通信和信息交流的基础之一。域名通过域名系统转换为 IP 地址。

#### 2.1.2 域名解析过程

域名系统是互联网上用于将域名解析为对应 IP 地址的一种机制，将易于记忆的域名映射到数字形式的 IP 地址，从而使用户能够通过域名访问特定的网络资源。DNS 在互联网资源定位过程中起着关键的作用，是互联网中不可或缺的基础设施之一<sup>[47]</sup>。

域名系统主要包含本地域名解析器、域名服务器、域名空间和资源记录，其中域名解析器将用户输入的域名解析为对应的 IP 地址，并将解析出的 IP 地址信息返回给用户。如图 2.1 所示，描述 DNS 系统进行域名解析的过程。

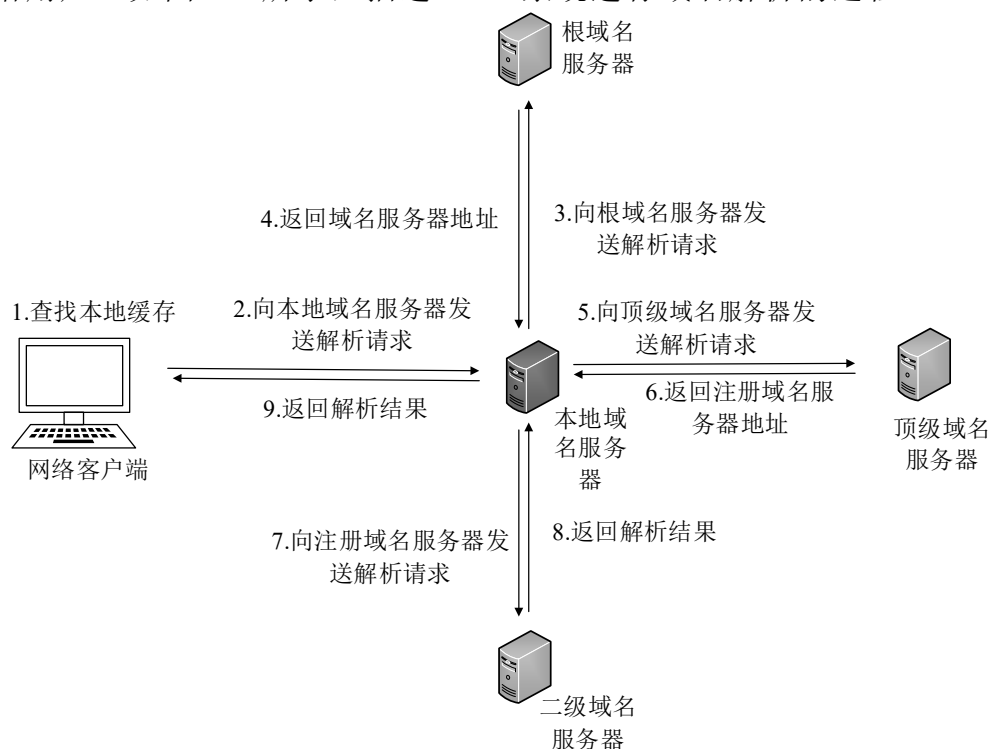


图 2.1 域名解析过程

首先查找本地缓存，如果有对应的 IP 地址，则网络客户端直接访问对应的互联网资源；否则，客户端便向本地 DNS 服务器发起域名查询请求。如果本地

域名服务器的缓存中有该 IP 地址，便直接将该地址提供给客户端；否则，本地域名服务器会向根域名服务器发出解析请求。根服务器响应中包含了顶级域名服务器的 IP 地址列表。随后，本地服务器请求顶级域名服务器进行解析，顶级服务器回复含有二级域名服务器 IP 地址的列表。本地服务器接着向二级域名服务器发起请求，二级服务器最终返回目标域名对应的 IP 地址。本地域名服务器将该 IP 地址转发给客户端，客户端将地址存入缓存，便于后来使用。

### 2.1.3 恶意域名

这些域名被攻击者创建或控制，用于欺骗用户、传播病毒、窃取用户信息，破坏网络安全<sup>[48]</sup>。恶意域名有许多种类，其中一些常见的包括：

（1）钓鱼域名：通过模仿合法网站的域名规则生成恶意域名，攻击者将这些恶意域名用于创建虚假登录页面或欺诈性网站，欺骗用户输入敏感信息，如家庭住址、账号和密码等，从而达到窃取用户敏感信息的目的。

（2）恶意软件域名：攻击者将恶意软件插入伪装后的恶意域名，用户访问这些域名会自动下载并执行恶意代码，从而导致用户端遭受损失。

（3）命令与控制（command and control, C&C）服务器域名：攻击者利用此类恶意域名用于远程控制受感染的计算机或设备，执行各种操作，如发送垃圾邮件、发起分布式拒绝服务攻击等。

恶意域名的危害包括：窃取用户个人信息，欺骗用户造成金融损失，诱骗用户浏览不良网站对用户心理健康造成负面影响，传播恶意软件、病毒或间谍软件，组建僵尸网络等。恶意域名对网络安全造成巨大威胁，因此及时识别恶意域名并阻止其恶意行为至关重要。

### 2.1.4 恶意域名生成算法

恶意域名生成算法（Domain Generation Algorithm, DGA）是一种网络攻击者用来为有恶意行为的命令和控制服务器生成新域名的技术<sup>[49]</sup>。域名生成算法能快速生成数千个用于恶意行为的域名，生成的域名可以作为攻击者与恶意服务器通信的集合点。然而算法每天生成的数以万计的 DGA 域名，其中大多数尚未注册，攻击者使用未注册的域名来隐藏已注册的域名，从而逃避使用签名和 IP 信誉的检查引擎的检测。恶意域名生成算法的工作流程可以概括为：算法在客户端和攻击者之间均执行，从攻击者和恶意服务器统一的种子开始，将生成域名以及将注册的域名作为恶意软件的通信通道。如果该域名被安全系统识别并阻止，攻击者可以快速切换到另一个域名以保持与恶意服务器的联系，继续破坏网络空间安全，如传播病毒间谍软件和入侵用户系统等。DGA 的威胁包括：在短时间内能生成大量随机域名，使得用户难检测和阻止它们。不断为恶意软件提供的新的恶意域名用于快速切换被检测出的恶意域名，使得恶意软件快速且有效地躲避

网络安全监测机制。因此，快速、有效地检测并阻止 DGA 生成的恶意域名具有十分重要的意义。

域名生成算法作为一种动态生成恶意域名的算法，有许多种类型，常见的几种 DGA 分类包括：

（1）确定型域名生成算法：利用确定性的算法生成域名，生成的域名由特定的种子值和算法组合产生，在相同的种子值下会生成相同的域名序列。

（2）伪随机型域名生成算法：通过伪随机数生成器生成域名。通常基于时间戳或其他环境变量作为种子值，以及预定义的算法生成域名。由于伪随机数生成器的使用，生成的域名是随机的，但在相同的种子值和算法下会生成相同的域名序列。

（3）加密型域名生成算法：使用加密技术生成域名。生成的域名是通过对种子值进行加密或哈希，然后将结果转换为域名形式。由于加密技术的使用，生成的域名具有高度的随机性和唯一性，使其更难以被检测和分析。

（4）混合型域名生成算法：结合了多种生成域名的技术生成恶意域名。如结合确定性算法和伪随机数生成器，或者确定性算法和加密技术，以产生具有一定随机性和唯一性的域名序列。混合 DGA 可以增加检测和对抗的复杂性。

## 2.2 向量化方法

### 2.2.1 One-hot 编码

One-hot 编码是用一个  $N$  维的向量来表示一个词。其中  $N$  是词典的大小，词典中单词索引对应的维度为 1，其他维度都设置为 0。

例如：

“朋友”表示为 [1 0 0 0 0 0 0 0 ...]

“家人”表示为 [0 0 0 0 1 0 0 0 ...]

One-hot 编码向量的维度等于词典中单词的总量数，单词总量数庞大时存在维数灾难。此外，One-hot 编码没有捕获不同单词之间的关系，没有包含有关上下文的信息。

### 2.2.2 Word2Vec 词向量编码

Word2Vec 是一种常用的词嵌入方法。将每个单词映射到一个固定长度的向量，每一个向量包含了单词在语义上的特征，根据单词含义的相似程度，设定在向量空间中单词之间的距离。Word2Vec 词嵌入方法包含两种模型，即连续词袋模型（Continuous Bag-Of-Words, CBOW）和 Skip-Gram 模型<sup>[50]</sup>。

CBOW 模型：通过目标单词前后位置单词来预测中间的目标单词，如图 2.2 所示。在这个模型中，将上下文单词传递到一个嵌入层，该层使用随机初始化的

权重来将单词映射到连续的向量空间中。然后，利用一个  $\lambda$  层，对这些词嵌入进行平均操作，以得到整个上下文的表示。最后，将这个平均表示通过一个全连接层，用于预测目标单词。

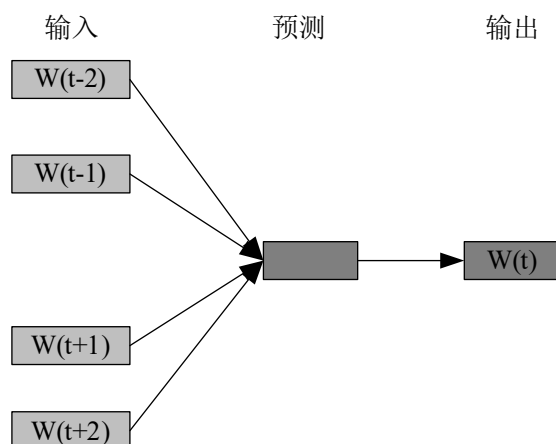


图 2.2 CBOW 模型结构

**Skip-gram 模型：**通过中间的目标单词预测目标单词前后位置单词，如图 2.3 所示。在这个模型中，目标单词被输入到一个嵌入层，将得到的词嵌入向量。利用获取的嵌入向量作为输入，通过一个全连接层预测上下文单词。

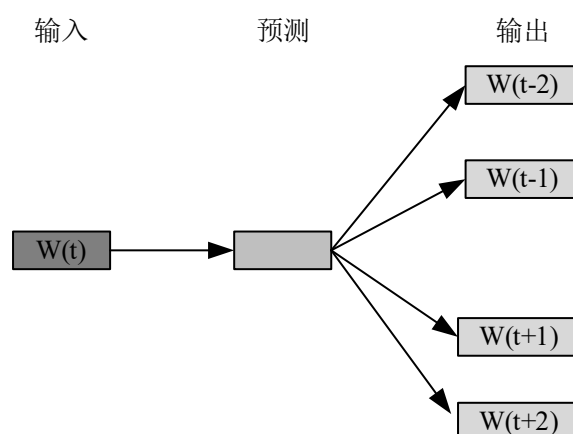


图 2.3 Skip-gram 模型结构

## 2.3 深度学习基础

### 2.3.1 卷积神经网络

卷积神经网络在深度学习领域广泛应用于文本处理和图像处理任务的一种模型，其主要组成部分为卷积层、池化层、全连接层<sup>[51]</sup>。在卷积层利用一个可学习的滤波器（卷积核）在输入数据上执行卷积操作，用以捕捉数据的局部特征。为了增强模型的表达力，卷积层后常跟随非线性激活函数，例如 ReLU 函数。进一步地，池化层被应用于减小特征图的维度，以此降低计算量和参数数量，同时提升模型对输入变化调节能力。经历多个卷积和池化层处理后，为了实现针对不同任务（如分类或回归）的有效执行，深度学习模型通常会在其结构中引入全连

接层和输出层。这些网络的训练依赖于反向传播技术，用于优化参数以降低损失值，其中随机梯度下降（Stochastic Gradient Descent, SGD）及其衍生算法被广泛应用，例如 Adam 和 RMSprop<sup>[52]</sup>。CNN 的一大优势是其能够自动从数据中学习特征，避免了手动特征设计的需要，并且因其结构特性，具备一定程度的平移不变性和局部不变性，使其适用于处理文本等含有丰富局部结构信息的数据。特别是在处理短文本分类问题时，CNN 表现出了其效率和有效性<sup>[53]</sup>。以下是该网络处理文本分类的一般步骤。

数据预处理：在处理文本数据前，必须进行预处理，包括清洗、分词、提取词干等步骤，随后将文本转换为向量形式。这一转换可以通过词袋、TF-IDF 或预训练词向量（如 Word2Vec）实现，选择哪种方法取决于特定的任务需求。

卷积神经网络模型构建：对于文本分类，设计的卷积神经网络模型通常包括卷积层、池化层和全连接层。卷积层用于捕获文本特征中的不同长度包含的信息，而池化层则用于降低特征维度，而全连接层则承担着分类任务。

训练模型：模型训练过程将文本特征向量和标签数据输入至网络中，并利用随机梯度下降算法或其他优化技术对网络参数进行调节。同时，需要选择适当的损失函数，例如交叉熵损失函数或聚焦损失函数，以提高模型分类的准确率。

模型评估：模型评价阶段，利用测试集检验模型表现，采用准确率、精确度、召回率、F1 分数等指标。如果模型的表现未能满足标准要求，可通过调整模型结构或超参数来进行改进，例如卷积核尺寸、池化方式、学习率等，或增加训练样本数量来提升模型性能。

总体而言，文本分类任务所涉及的卷积神经网络应用包含多个阶段，包括数据预处理、向量化、构建模型、训练模型和评估模型等不同步骤。但是在特定应用场景下，如恶意域名分类，需要选择合适的模型结构，并进行参数调整，以达到最优的分类效果。

### （1）卷积层

卷积层是卷积神经网络的关键组成部分，利用卷积操作从输入数据中提取特征。如图 2.4 所示，卷积操作通过卷积核在输入数据上的滑动实现。卷积核通常为一个二维张量，常呈正方形或矩形，内含权重参数。在卷积过程中，卷积核与当前滑动位置上的输入数据对应元素相乘并求和，得到一个值作为特征的输出。通过对卷积核权重进行调整，可以从输入数据中提取出多样化的特征。这一过程使得卷积层能够捕捉数据的局部结构和形式，为后续的分类或回归任务提供有力的特征表示。

在卷积层中，涉及若干关键的超参数设置，包括卷积核的大小、步长及填充方式等。卷积核的大小由其宽度和高度定义；步长代表卷积核移动时的间隔距离；填充操作是通过向输入数据的外部添加一定数量的数据来实现的，其目的在

于始终保持输出数据的尺寸与输入数据相同或者保持一定的比例关系。卷积过程中，数据大小计算方法见式(2.1)。

$$N = \frac{W - F + 2P}{S + 1} \quad (2.1)$$

假设输入数据的大小为  $W$ ，卷积核的尺寸为  $F$ ，边缘填充量为  $P$ ，步长大小为  $S$ 。

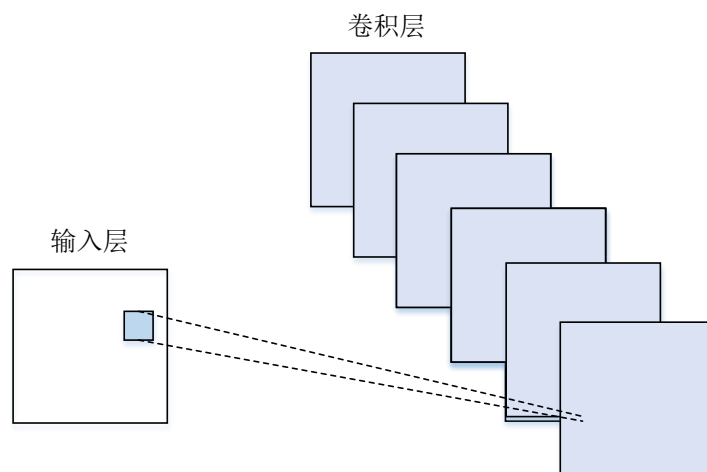


图 2.4 卷积操作示意图

## (2) 激活函数

**Sigmoid 激活函数：**Sigmoid 是一种常见的激活函数，可以通过将实数输入映射到 0 和 1 之间的范围来实现。该函数的具体表达式见式(2.1)。

$$\text{sigmoid}(x) = \frac{1}{(1 + e^{-x})} \quad (2.1)$$

其中，输入  $x$  为实数，Sigmoid 函数具有将实数映射到 0 和 1 之间的特性，因此其输出可解释为概率估计，并广泛应用于二分类问题的输出层，作为激活函数。Sigmoid 函数的主要优势在于它是可微的，能够将输出严格限制在一定范围内。然而，Sigmoid 函数存在明显的缺陷是梯度消失或梯度爆炸，即当输入值过大或过小时，Sigmoid 函数的梯度会接近于零，如图 2.5 所示。因此，在当前研究中普遍采用 ReLU 函数以及其衍生变体作为替代方法。

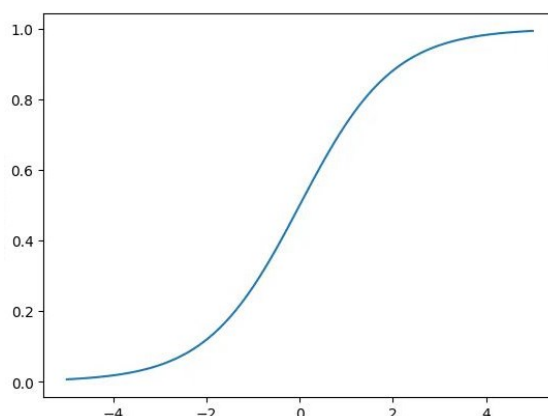


图 2.5 Sigmoid 函数

**Softmax 激活函数：**是一种关键的激活函数，在数值转换成概率分布方面具有重要作用。其保证了输出的每个数值落在 0 到 1 的范围内，并且所有元素的总和等于 1。这种特性使得 Softmax 在处理多分类问题时具有广泛的适用性，例如文本分类，每个概率值代表着相应类别的可能性。

具体地说，给定一个包含  $N$  个实数的向量，该向量代表了深度学习模型在最后一层的原始输出。Softmax 函数将这个  $N$  维的实数向量转化为一个具有相同维度的输出向量，其中输出向量的每个元素都通过特定的计算方式获得，从而描述了该元素所对应类别的概率估计，其每个元素的计算公式如(2.3)所示：

$$\text{softmax}(x) = \frac{e^{x_j}}{\sum_{i=1}^N e^{x_i}} \quad (2.3)$$

其中， $e^x$  表示指数函数。Softmax 函数具有独特的特性，其功能是将输入值转化为一个概率分布，确保每个输出的概率值位于 0 到 1 之间，并且所有输出概率值的总和为 1。通过应用 Softmax，网络的输出可以解释为概率分布，便于在多类别问题中识别最可能的类别。Softmax 函数可视化表示如图 2.6 所示。

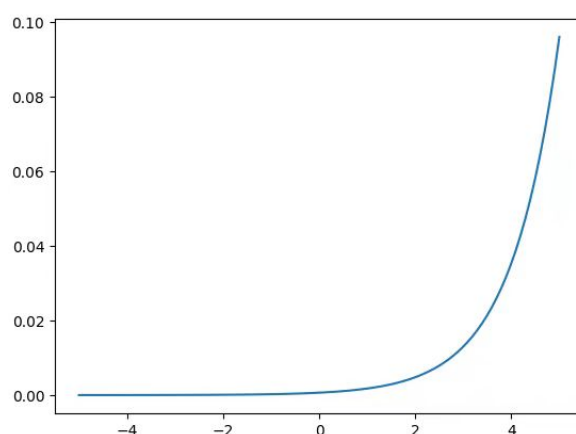


图 2.6 Softmax 函数

**tanh 激活函数：**值域为  $(-1,1)$ ，可由 Sigmoid 函数推导出，其函数图像如图 2.7 所示。与 Sigmoid 激活函数相比，tanh 函数的输出以 0 为中心，这一特性使得其在初期训练阶段相对于 Sigmoid 函数有更快的收敛速度，因为输出值的中心化减少了每一层输出的偏差，有助于加速梯度下降的过程。然而，尽管 tanh 函数在某些方面优于 Sigmoid 函数，仍然面临着梯度消失的问题，在神经网络的深层中，由于乘积累积效应，梯度可能迅速减小到接近于 0 的程度，导致网络难以学习和更新权重，进而影响模型的训练效率和性能。尽管存在局限，但由于其对称性，在许多神经网络模型中仍然是一个非常重要的工具，尤其是在需要处理负数输入时。其数学表达式见式(2.4)。

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 2\sigma(2x) - 1 \quad (2.4)$$

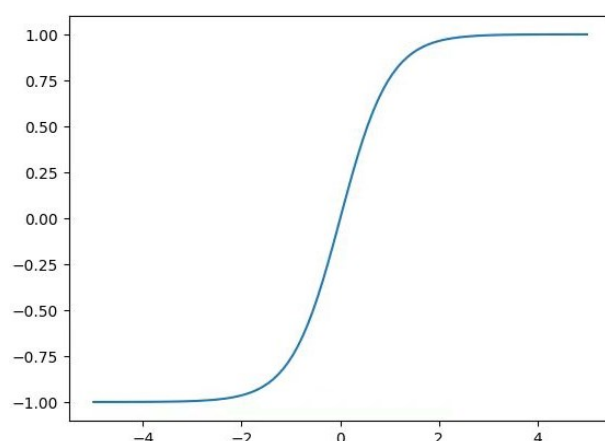


图 2.7 Tanh 函数

### (3) 池化层

池化层，也称作下采样层，在卷积神经网络中起着重要作用，其目标是减小特征图的维度。位于卷积层之后的这一层通过简化数据处理，既减轻了计算负担，又保持了关键特征，有利于提升模型对新数据的适应能力并减少过拟合风险。通过对小片区域内数据进行汇总，池化层能够概括区域特征。它主要通过两种方式进行：最大池化和平均池化，这有助于提取和保留重要信息。如图 2.8 所示。在尽管此过程降低了数据的空间维度，但特征图的深度保持不变。

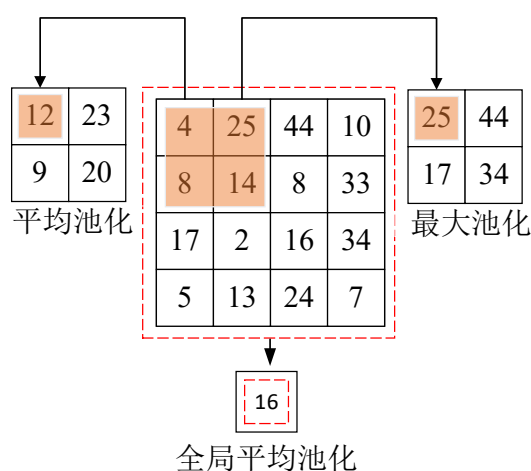


图 2.8 池化过程

当输入尺寸过大时，减少可训练参数数量是一种良好的实践。这样做可以在保留重要信息的同时，降低特征维度。为了实现这一目标，可以在连续的卷积层之间周期性地引入池化层。池化操作，也被称为子采样或降采样，用于减小每个特征图的空间尺寸，同时保留了最重要的信息。空间池化操作具有多种类型，包括最大池化、平均池化等。分别通过提取局部区域内的最大值和平均值来实现特征的压缩。最大池化通过保留最显著的特征，有助于模型学习关键信息；而平均池化通过平滑处理，有助于抑制噪声，这两种池化方法的选择和应用根据特定任务需求做出具体的选择。

#### (4) 全连接层

全连接层，也被称为密集层，是一种基础的神经网络构建块，在此层中的每个节点都与前一层的全部节点连接，该结构使得数据经过网络传输时可以经过加权和激活函数的处理，以生成最终的输出结果。结构如图 2.9 所示。此层通常用于数据分类或回归任务，并能够有效地减少特征维度，从而优化网络的性能。

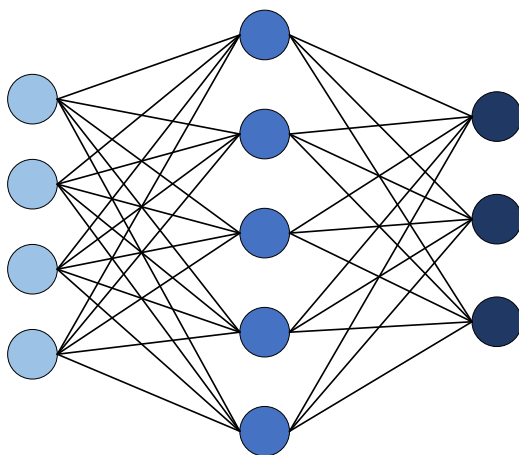


图 2.9 全连接层结构

全连接层通常位于网络的尾部，跟随在卷积层和池化层之后。主要用于处理经过展平的一维数据，通过非线性激活函数的处理后，利用 Softmax 或 Sigmoid 等函数输出最终的类别概率，实现对输入数据的分类。

全连接层的优势在于其能够构建起输入特征间复杂的非线性映射，增强模型对不同数据特性的适应性。然而，这层面临的问题包括参数众多、易于过拟合和高计算成本。在面对特定任务，如文本分类时，通常会采用 dropout 和正则化等策略来缓解过拟合问题，以提升模型的泛化能力。

#### 2.3.2 长短期记忆网络

长短期记忆网络是一种常用的提取时序特征的架构，它使用一系列乘法门来控制信息如何流入和流出网络的内部状态<sup>[54]</sup>。LSTM 结构如图 2.10 所示。

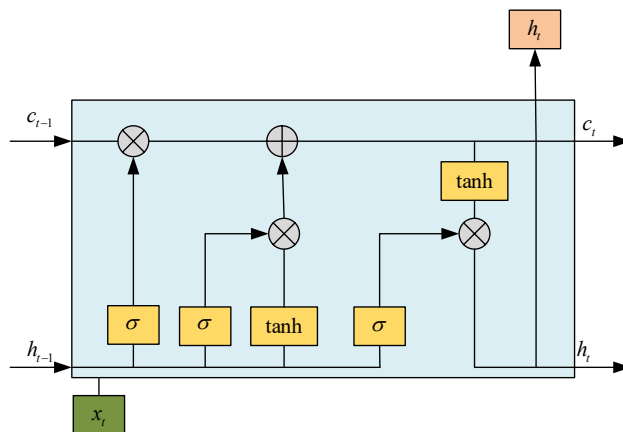


图 2.10 LSTM 结构

LSTM 隐藏状态接收输入  $x_t$  和前一个的隐藏状态  $h_{t-1}$ 。见式(2.5)。

$$\hat{h}_t = W_{hx}x_t + W_{hh}h_{t-1} \quad (2.5)$$

LSTM 网络有 3 个门单元-输入门 $i$ 、输出门 $o$ 和遗忘门 $f$ ，它们既有循环连接，也有前馈连接。见式(2.6)-(2.8)。

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1}) \quad (2.6)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1}) \quad (2.7)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1}) \quad (2.8)$$

其中， $\sigma$ 为 logistic sigmoid 函数。输入门控制每个隐藏单元的输入有多少被写入内部状态向量  $c_t$ ，而遗忘门决定有多少先前的内部状态  $c_{t-1}$  被保留。这种写入和遗忘门的组合允许网络控制应该在每个时间步长存储和覆盖哪些信息。内部状态见式(2.9)。

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(\hat{h}_t) \quad (2.9)$$

输出门控制每个单元的激活被保留多少。允许 LSTM 单元保留与当前输出不相关但以后可能相关的信息。隐藏状态的最终输出见式(2.10)。

$$h_t = \tanh(c_t) \odot o_t \quad (2.10)$$

### 2.3.3 门控循环网络

为应对传统循环神经网络（Recurrent Neural Network, RNN）面临的长距离依赖及在反向传播过程中梯度消失或爆炸的问题，门控循环单元（Gated Recurrent Unit, GRU）作为一种循环神经网络变体，在 2014 年被提出。GRU 与 LSTM 在多种建模场景下展现出了相似的性能<sup>[55,56]</sup>。随后根据文献[57]基于公开的语料库数据集所进行的 GRU 与 LSTM 的比较研究，GRU 在处理长文本内容和较小规模数据集时表现出更优性能，并且研究表明 GRU 相对于 LSTM 拥有更佳的性能成本比。GRU 通过借鉴 LSTM 的设计思想，并在内部结构上进行优化，如取消单元状态  $c_t$  和简化门控机制，只利用隐藏状态  $h_t$  在各单元间传递信息，因此具备更少的参数数量和更高的训练效率。GRU 单元结构如图 2.11 所示。

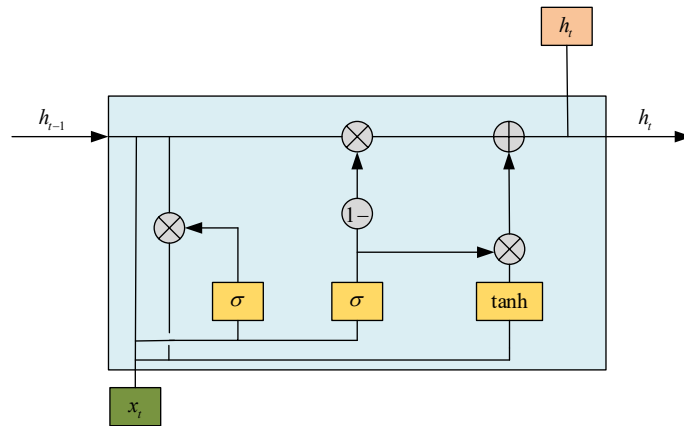


图 2.11 GRU 结构

GRU 只有两个门控单元，重置门和更新门。通过更新门和重置门，GRU 能够在不同时间步骤之间有效地传递信息，从而保持长期依赖的信息。

(1) 更新门：决定保留和更新多少之前的状态见式(2.11)。

$$z_t^f = \sigma(W_{xz}^f x_t + W_{hz}^f h_{t-1}^f + b_z^f) \quad (2.11)$$

其中,  $z_t^f$  表示 GRU 中的更新门的方程,  $W_{xz}^f$  和  $W_{hz}^f$  表示与其相关联的权重矩阵, 其分别表示将当前时间步长的输入连接到更新门的权重和将先前时间步长的隐藏状态连接到更新栅极的权重。  $b_z^f$  是与更新门相关联的偏置项。

(2) 重置门：控制先前状态的哪些信息被遗忘或重置见式(2.12)。

$$r_t^f = \sigma(W_{xr}^f x_t + W_{hr}^f h_{t-1}^f + b_r^f) \quad (2.12)$$

其中,  $r_t^f$  是 GRU 的重置门,  $\sigma$  是 sigmoid 激活函数。  $W_{hr}^f$  是将前向 GRU 单元的隐藏状态  $h_{t-1}^f$  连接到其重置门  $r_t^f$  的权重矩阵。它是从上一个步长的隐藏状态向正向重置门的转换。  $W_{xr}^f$  是与当前时间步长  $x_t$  中到复位门的输入相关联的权重矩阵。  $b_r^f$  是与该门相关的偏置项。

(3) 候选隐藏状态：表示可能包含在当前状态中的新信息见式(2.13)。

$$\tilde{h}_t^f = \tanh(W_{xh}^f x_t + r_t^f \odot (W_{hh}^f h_{t-1}^f) + b_h^f) \quad (2.13)$$

其中,  $\tilde{h}_t^f$  候选隐藏状态,  $\odot$  表示元素乘法,  $\tanh$  是双曲正切激活函数。权重  $W_{xh}^f$  和  $W_{hh}^f$  表示将输入连接到候选激活以及将前一时间步长的隐藏状态连接到候选激活的权重矩阵,  $b_h^f$  表示候选隐藏状态偏置。

(4) 隐藏状态：将先前与候选状态结合, 然后由更新和重置门控制。公式见式(2.14)。

$$h_t^f = (1 - z_t^f) \odot h_{t-1}^f + z_t^f \odot \tilde{h}_t^f \quad (2.14)$$

通过这些门控机制, LSTM 和 GRU 均改善了传统 RNN 在处理长序列依赖问题, 使 LSTM 和 GRU 非常适合进行域名上下文信息的处理与分析任务。

### 2.3.4 自编码器网络

自编码器 (Auto-encoder, AE) 这一概念最早由 Rumelhart 于 1986 年提出。到了 2006 年, 在《Science》杂志上发表了一篇文章, 介绍了一种利用自编码器进行降维的方法<sup>[58]</sup>。该研究通过采用快速贪心算法逐层训练深度前馈网络, 优化了网络参数, 有效避免了传统方法易于落入局部最优的问题, 从而引入了深度自编码器的概念<sup>[59]</sup>。在此基础上进一步探索, 引入稀疏性约束, 旨在通过尽可能少的神经元捕获数据的重要特征, 由此发展出稀疏自编码器。此外, 从增强算法鲁棒性的角度着手, 通过对输入数据添加噪声干扰作为预处理步骤, 以提升模型的重构能力, 形成了去噪自编码器。提出了收缩自编码器, 增加了对特征维度处理的约束。紧随其后, 研究者提出了堆叠卷积自编码器, 用于分次地提取特征。在自编码器的持续优化和发展过程中, 已在自然语言处理、图像分类、人脸识别等多个领域被广泛应用, 并取得了显著成就<sup>[60,61]</sup>。

自编码器作为一种基于深度学习的无监督特征学习方法，可以简要看作是一个三层的神经网络结构：输入层，隐藏层和输出层，如图 2.12 所示。 $X$  表示输入， $\hat{X}$  表示输出，自编码器网络的目的是将输入数据经过网络之后近似输出，则 AE 的目标函数见式(2.15)。优化目标函数，以最小化重建损失。AE 的隐藏层神经元少于输入层和输出层神经元，因此可以在隐藏层提取低维特征。

$$L(X, \hat{X}) = \|X - \hat{X}\|^2 \quad (2.15)$$

设计自编码算法的思路包括以下步骤：

(1) 利用无监督学习技术对未标注数据的特性进行学习。首先，输入向量通过编码器被转化为一个隐藏的潜在表示，然后这个表示通过解码器重新构建为输出向量。通过反向传播技术调节编解码阶段的参数，以减少重建损失，从而使得重建的输出与原始输入相似程度高。

(2) 由编码器生成的特征被送入后续网络层，并在各层中进行训练。这个通过训练层获得的编码过程实际上是对输入数据的一种再现，随后的每一层都会重复这一过程。通过反向传播技术调整网络参数，一旦训练完成，自编码器便能有效地提取特征，这些通过自编码得到的特征能够尽可能完整地代表原始输入数据。

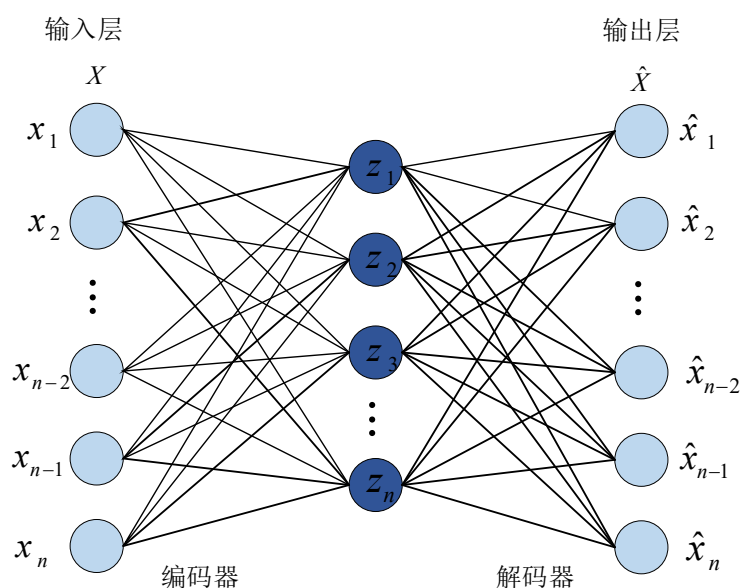


图 2.12 自编码器网络结构

### 2.3.5 变分自编码器网络

变分自编码器 (Variational Auto-encoder, VAE) 是一种有向概率图模型，其后验由神经网络逼近，形成一种类似自编码器的结构<sup>[62]</sup>。如图 2.13 所示，将  $z$  限制为满足一定的高斯分布，然后从高斯分布  $p(z)$  中得到  $z$ ，最后从分布  $p_{\theta}(x|z)$  中生成数据。任意分布  $p(x)$  的数据可以由满足高斯分布的隐变量  $z$  通过神经网络生成。在 VAE 的概率图模型中，生成模型  $p_{\theta}(x|z)$  相当于解码器，识别模型  $q_{\phi}(z|x)$  相当于编码器。

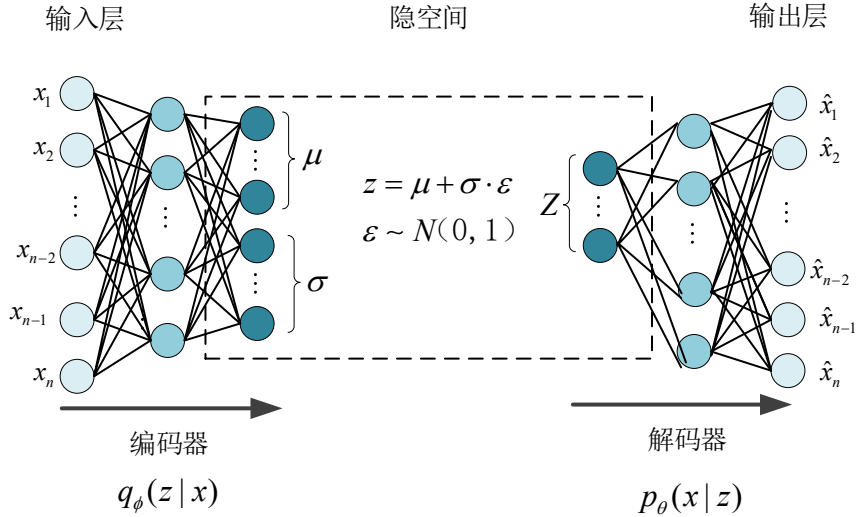


图 2.13 变分自编码器网络结构

为了使生成的数据  $\hat{x}$  与原始数据  $x$  最相似。分布  $p_\theta(x)$  应最大化。然后，使用最大似然法对似然函数进行最大化， $p_\theta(x)$  的表达见式(2.16)。

$$p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz \quad (2.16)$$

其中， $p_\theta(x|z)$  表示隐变量  $z$  对原始数据  $x$  的重建， $p_\theta(z)$  表示隐变量  $z$  的先验分布。 $p_\theta(z|x)$  表示从原始  $p_\theta(z|x)$  数据中得到隐变量  $z$  的过程。

由于  $p_\theta(z|x)$  相对难以计算，因此变分自编码器用服从高斯分布的近似后验  $q_\phi(z|x)$  代替真实后验  $p_\theta(z|x)$ 。为了度量两个分布之间的相似性，通常使用 Kullback-Leibler 散度(KL 散度)来度量两个随机分布之间的距离，当两个随机分布相同时，其 KL 散度为零。KL 散度公式见式(2.17)。

$$\begin{aligned} D_{KL}(q_\phi(z|x) \| p_\theta(z|x)) \\ &= E_{q_\phi(z|x)} \log \frac{q_\phi(z|x)}{p_\theta(z|x)} \\ &= E_{q_\phi(z|x)} [\log q_\phi(z|x) - \log p_\theta(z|x)] \\ &= E_{q_\phi(z|x)} [\log q_\phi(z|x) - \log p_\theta(x|z) - \log p_\theta(z)] + \log p_\theta(x) \end{aligned} \quad (2.17)$$

则

$$\begin{aligned} \log p_\theta(x) - D_{KL}(q_\phi(z|x) \| p_\theta(z|x)) \\ &= E_{q_\phi(z|x)} [-(\log q_\phi(z|x) + \log p_\theta(z)) + \log p_\theta(x|z)] \end{aligned} \quad (2.18)$$

由于散度  $D_{KL}(q_\phi(z|x) \| p_\theta(z|x)) \geq 0$ ，见式(2.19)。

$$L(\theta, \phi; x) = E_{q_\phi(z|x)} [-(\log q_\phi(z|x) + \log p_\theta(z)) + \log p_\theta(x|z)] \quad (2.19)$$

通过(2.18)和(2.19)得到：

$$\log p_\theta(x) \geq L(\theta, \phi; x) \quad (2.20)$$

然后最大化对数似然函数  $\log p_\theta(x)$ ，使后验分布  $q_\phi(z|x)$  接近真实后验分布  $p_\theta(z|x)$ ，即  $D_{KL}(q_\phi(z|x) \| p_\theta(z|x))$  趋近于 0， $L(\theta, \phi; x)$  是  $\log p_\theta(x)$  的变分下界。为了优化  $\log p_\theta(x)$  和  $D_{KL}(q_\phi(z|x) \| p_\theta(z|x))$ ，就需要优化变分下界，及  $L(\theta, \phi; x)$  就作

为 VAE 的损失函数见式(2.21)。

$$\begin{aligned} L(\theta, \varphi; x) &= E_{q_{\varphi}(z|x)}[-(\log q_{\varphi}(z|x) + \log p_{\theta}(z)) + \log p_{\theta}(x|z)] \\ &= -D_{KL}(q_{\varphi}(z|x) \| p_{\theta}(z)) + E_{q_{\varphi}(z|x)}[\log p_{\theta}(x|z)] \end{aligned} \quad (2.21)$$

其中， $E_{q_{\varphi}(z|x)}[\log p_{\theta}(x|z)]$  可以看作变分自编码器的重建误差。则优化损失函数  $L(\theta, \varphi; x)$  的过程就是优化正则化项和重建误差两部分。

#### (1) 正则化项

$p_{\theta}(z)$  服从  $N(0; I)$  的高斯分布， $q_{\varphi}(z|x)$  服从  $N(\mu, \sigma^2)$  的高斯分布，则正则项可以表示为见式(2.22)。

$$loss\_kl = \frac{1}{2} \sum_{j=1}^n (1 + \log(\sigma_j^2) - (\mu_j^i)^2 - (\sigma_j^i)^2) \quad (2.22)$$

其中， $j$  表示  $z$  的维度。

#### (2) 重建误差项

重建误差通常通过计算原始输入数据与重建数据之间的差异来度量。可以采用均方误差表示见式(2.23)。

$$loss\_rec = MAE(x, \hat{x}) = \frac{1}{n} \sum_{i=1}^n |x_i - \hat{x}_i| \quad (2.23)$$

变分自编码器通过编码、采样、解码生成原始输入，使用最小化重构误差及 KL 散度的方式进行优化。变分自编码器构建流程见算法如表 2.1 所示。

表 2.1 变分自编码器构建算法

#### 算法 2.1：变分自编码器算法

**输入：** 正样本训练集  $\mathbf{X}_{train} = \{\mathbf{x}_{train}^{(1)}, \mathbf{x}_{train}^{(2)}, \dots, \mathbf{x}_{train}^{(N)}\}$ , 最大执行次数  $Epoch$ .

**输出：** 编码器  $f_{\phi}$ , 解码器  $g_{\theta}$ .

**过程：**

1 初始化参数  $\phi, \theta$ ;

2 **repeat**

3   **for**  $i=1$  to  $N$  **do**

4     从标准正态分布  $N(0,1)$  总体抽取  $K$  个样本

5     计算  $z^{(i,l)} = \mu^i + \delta^i \cdot \varepsilon^i$ ,  $i=1, \dots, N, l=1, \dots, L$ ;

6   **end for**

7     计算  $L(x) = \sum_{i=1}^N -D_{KL}(q_{\phi}(z|x^{(i)}) \| p_{\theta}(z))$   
 $\quad + \frac{1}{K} \sum_{k=1}^K \log p_{\theta}(x^{(i)} | z^{(i,k)})$ ;

8    $\phi, \theta \leftarrow$  使用随机梯度下降更新参数;

9 **until** 参数  $\phi, \theta$  收敛;

变分自编码器不仅展现出可扩展性，还具备灵活性。其编码器和解码器部分能够采纳多样化的神经网络架构，比如卷积神经网络、多层感知机以及循环神经网络等。这一特性让变分自编码器能够适配多种数据类型，包括文本、时间序列数据等。进一步地，变分自编码器能通过加入多种的限制条件和目标函数进行功能上的扩充，如融入生成模型的特性，使其满足更广泛的任务需求和应用场景。

### 2.3.6 损失函数

在深度学习的体系结构中，损失函数发挥着重要的作用，主要负责量化模型预测与真实标签之间的差异。优化算法，如梯度下降法，利用这种量化指标来指导模型参数的调整，以减小预测输出与实际观测之间的偏差。设计损失函数时，既要考虑预测值与真实值的偏离程度。同时，确保模型能够全面适应数据特性也是重要的。一些广泛使用的损失函数有交叉熵（Cross Entropy Loss）、焦点损失（Focal Loss）和标签平滑（Label Smoothing）等。

以均方误差为例，假设模型的预测结果为  $y_i$ ，实际标签为  $t_i$ ，则均方误差如公式(2.24)所示。

$$L(y, t) = \frac{1}{n} \sum_{i=1}^n (y_i - t_i)^2 \quad (2.24)$$

其中， $n$  表示样本数量， $y_i$  和  $t_i$  分别表示第  $i$  个样本的预测结果和实际标签。通过对损失函数进行计算，可衡量模型预测结果与实际标签之间的差异程度。在模型训练过程中，通过优化算法调整参数，目的是缩小预测与真实结果之间的偏差。这一过程依赖于损失函数的指示，以实现模型性能的持续优化。

损失函数起着重要的作用，不仅衡量模型预测与实际值之间的偏差，而且还指导优化过程中的参数调整。针对不同的场景和模型，选取合适的损失函数对于提高模型的精确度和稳健性至关重要。

#### （1）交叉熵损失函数

交叉熵损失函数在处理分类问题时具有关键性作用，尤其在监督学习领域中应用广泛。其主要功能在于精准衡量模型预测结果与真实标签之间的差异。在涉及  $n$  个类别的分类任务中，模型会输出一个概率分布  $P = (P_1, P_2, P_3, \dots, P_n)$ ，而真实标签则经过独热编码转化为向量形式。特别地，对于第  $i$  个类别，其真实标签以特定方式表示  $Y = (y_1, y_2, y_3, \dots, y_n)$ ，且不会与其他类别的标签重叠。交叉熵损失函数通过采用特定的计算公式见式(2.25)。有效地量化模型预测与真实情况之间的偏差，进而指导模型的优化过程。

$$loss = - \sum_{i=1}^n y_i \log p_i \quad (2.25)$$

其中， $\log$  为自然对数， $p_i$  为模型输出的第  $i$  个类别的概率。自然对数  $\log$  主要用来量化模型对特定分类的预测概率。交叉熵损失函数基于一个原则：如果模型预测的概率分布与真实标签的 One-hot 表示一致，损失将为零；反之，任何偏差都会使得损失增加，这种机制不仅准确地量化了模型预测的准确性，而且也直观地反映了分类任务中预测误差的大小。这一特性使交叉熵成为优化算法，如梯度下降、反向传播在优化模型时的重要工具。

#### （2）聚焦损失函数

Focal Loss 损失函数被广泛应用于克服数据集类别不平衡问题，尤其是针对

数据集中存在的样本类别数量不均的情况。通常情况下，传统损失函数容易导致模型偏向数量较多的类别，而忽视数量较少的类别。为应对这一问题，Focal Loss 引入了调节因子和缩减项，对传统损失函数进行优化，从而解决了类别不平衡的挑战。调节因子的作用在于调整不同类别在损失计算中的权重，特别针对那些数量众多的类别进行权重降低，以减少它们对模型训练的过度影响。而缩减项的主要目的在于降低模型对易分类样本的偏向，从而使得模型更加专注于难以识别的样本。这一策略有效地调整了模型对各类别样本的重视程度，进而提高了模型在识别少数类别上的性能。Focal Loss 是应对类别不平衡问题的损失函数。实际中，数据集常存在类别样本数量不均的问题，传统交叉熵损失易使模型偏向多数类别。Focal Loss 引入平衡因子和衰减因子，优化性能。通过引入平衡因子来减少对多数类别的权重，同时通过衰减因子降低对易分类样本的重视，这样的调整有助于提升模型对少数类别的识别能力，增强了模型的整体表现能力。

设模型预测样本正确分类的概率值用  $p_i$  表示，而  $t$  作为调整因子， $\gamma$  作为衰减系数，共同影响着损失的计算，Focal Loss 的表达式如公式(2.26)。特别地，当  $\gamma$  设置为 0 时，Focal Loss 将简化为标准的交叉熵损失，从而展示了其具备灵活性与通用性的特质。Focal Loss 作为解决类别不均衡问题的一种有效方法，在多个领域，如文本分类和目标检测中展现了其卓越性能，并因其优异的应用效果而被广泛采用。

$$FL(p_i) = -\alpha_i(1 - p_i)^\gamma \log(p_i) \quad (2.26)$$

### 2.3.7 性能评价指标

本文提出的恶意域名检测模型，主要用于判断输入的域名是否为合法或恶意。为了准确评估该模型的性能，采用混淆矩阵对测试数据集中的预测结果与实际标签之间的一致性进行了明确呈现<sup>[63]</sup>。

混淆矩阵用于基准模型在测试数据集上的预测结果与实际标签情况。其中，真正例（True Positive, TP）指模型成功预测出恶意域名的数量，假正例（False Positive, FP）指模型错误将合法域名预测为恶意的数量，假反例（False Negative, FN）指模型错误将恶意域名预测为合法的数量，而真反例（True Negative, TN）指模型成功预测出合法域名的数量，如表 2.2 所示。

表 2.2 混淆矩阵

真实情况	预测情况	
	正例	反例
正例	TP	FN
反例	FP	TN

评估分类模型性能时，需综合考虑多方面的指标，主要包括以下几个方面：

准确率（Accuracy）：反映了模型预测正确的比例，如公式(2.27)所示。准

准确率越高，通常意味着模型的分类效果越好。

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.27)$$

精确率 (Precision): 是指模型在预测为正例的样本中, 实际为正例的比例, 该指标能够反映出模型对正例的识别能力。如公式(2.28)所示。精确率越高, 模型在预测正例时的可靠性越强。

$$Precision = \frac{TP}{TP + FP} \quad (2.28)$$

召回率 (Recall): 用于衡量模型对实际正例的正确预测比例。如公式(2.29)所示。召回率越高, 模型在识别实际正例时效果越好。

$$Recall = TPR = \frac{TP}{TP + FN} \quad (2.29)$$

假正例 (False Positive Rate, FPR): 是表示在实际为负例的样本中, 被错误预测为正例的比例。如公式(2.30)所示。FPR 越低, 说明模型在区分反例时的表现越出色, 特别是在恶意域名检测等应用中, 降低 FPR 至关重要, 因为误判正常样本为异常可能带来严重后果。

$$FPR = \frac{FP}{FP + TN} \quad (2.30)$$

F1 分数 (F1-Score): 是精确率和召回率的调和平均数, 它综合考虑了两者的影响, 特别是在处理不平衡数据集时, F1 分数能够提供一个更全面的性能评估。如公式(2.31)所示。

$$F1-score = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (2.31)$$

## 2.4 本章小结

本章节主要介绍了域名的相关理论知识以及利用深度学习进行恶意域名检测相关的部分基础理论和技术。第一节介绍了域名基础知识。第二节介绍了域名向量化表示的向量化方法。第三节介绍了深度学习基础知识。其中包括卷积神经网络、门控循环单元和变分自编码器网络等, 然后再介绍了几种损失函数以及本文采用的模型性能评价指标。

## 第3章 基于 CNN-BiGRU-AFF 的恶意域名检测

### 3.1 引言

现有的恶意域名检测模型，通常只提取局部或全局维度特征，使得模型对域名字符串的有限信息利用不充分，为此本章提出了一种基于 CNN-BiGRU-AFF 的恶意域名检测方法。首先，将域名向量化表示，并用注意力机制对域名关键信息进行标注。其次利用 CNN 提取域名的局部特征，BiGRU 获取域名全局特征。最后利用交叉注意力机制对 CNN 和 BiGRU 获取的特征向量进行融合作为全连接层的输入。最后，通过全连接层和 softmax 函数对待测域名进行分类。实验结果表明，与现有恶意域名检测技术相比，CNN-BiGRU-AFF 模型在检测准确性、精确度、召回率、F1 分数方面都表现出显著优势。

### 3.2 检测流程

用于恶意域名检测提出的 CNN-BiGRU-AFF 模型主要包括以下模块：域名数据预处理，注意力机制，CNN，BiGRU，AFF 和分类器等模块。首先，将待测域名样本通过域名数据预处理模块进行向量化表示并用注意力机制对域名元素进行权重分配；其次利用 CNN 和 BiGRU 模块并行获取域名特征；然后通过注意力特征融合模块对获取的域名特征进行融合；最后通过全连接层和 softmax 函数获取域名类别的最大概率分布。如图 3.1 所示。

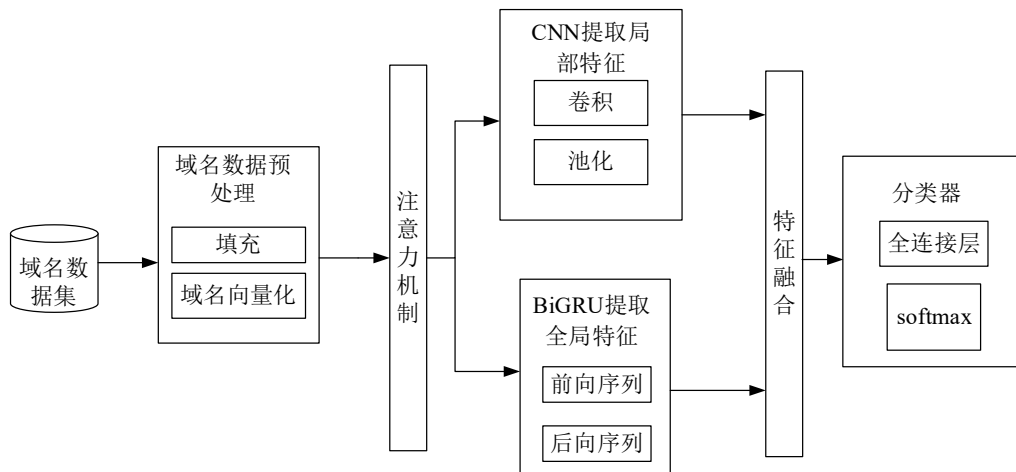


图 3.1 基于 CNN-BiGRU-AFF 的恶意域名检测流程

#### 3.2.1 域名数据预处理

通过统计，组成域名数据集的字符包括 26 个英文字母“a-z”和“A-Z”，10 个阿拉伯数字“0-9”，1 个连字符“-”，1 个分隔符“.”以及设置的“PAD”和

“UNK”，其中英文字母不区分大小写，共有  $N=40$  个不同的字符。利用 Word2Vec 技术对以上字符进行向量化，将每个字符转化为  $D=30$  维的向量，并存入向量表  $C_1 \in R^{N \times D}$  中。

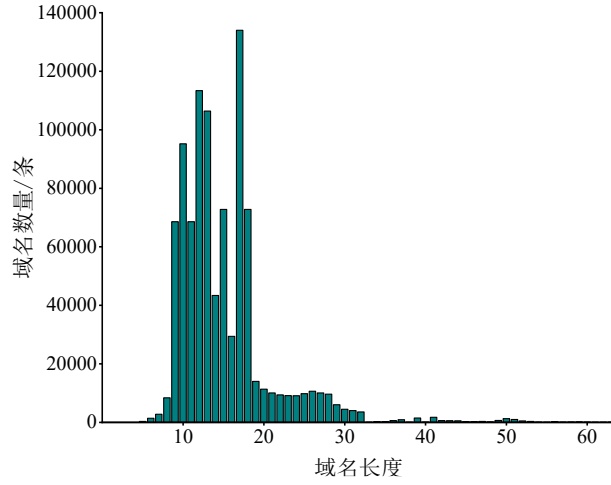


图 3.2 数据集域名长度统计

域名数据集中域名字符串最大长为 63 个字符，如图 3.2 所示。将输入的字符串向量长度设置为  $Len=63$ ，长度小于 63 的域名字符串用零向量填充，然后序列化域名字符串中所有字符，获取域名字符串的向量表示。然后作为注意力机制模块输入。

### 3.2.2 关联特征提取

通过域名数据预处理后原始域名转化为向量表示，其中包含域名冗余信息。注意力层的任务是通过为域名字符分配权重，从大量的输入域名信息中选择最关键的信息。例如恶意域名会包含特定的字符组合，用来模仿知名网站的拼写，这些特征有助于识别恶意域名。在注意力机制中，通过计算域名中每个字符的相关系数来确定字符向量的权重。因此，域名中更重要的字符会受到更大的权重，从而增强关键特征的利用率，降低冗余信息对模型的影响。具体计算流程如图 3.3 所示。

输入的域名字符串  $X=[x_1, \dots, x_n] \in R^{Len \times D}$ ，设  $Q$  为查询向量， $K$  为键向量， $V$  为值向量。通过公式(3.1)-(3.3)将每个域名字符串样本映射到  $Q$ 、 $K$ 、 $V$  三种向量空间中。

$$Q = W_q X \in R^{D_q \times D} \quad (3.1)$$

$$K = W_k X \in R^{D_k \times D} \quad (3.2)$$

$$V = W_v X \in R^{D_v \times D} \quad (3.3)$$

其中， $W_q \in R^{D_q \times Len}$ ， $W_k \in R^{D_k \times Len}$  以及  $W_v \in R^{D_v \times Len}$  表示可训练参数矩阵。因此，通过注意力机制对域名信息过滤的输出  $H$  的表达式，见式(3.4)。

$$H = attention((K, V), Q) = V \text{softmax}\left(\frac{K^T Q}{\sqrt{D_k}}\right) \quad (3.4)$$

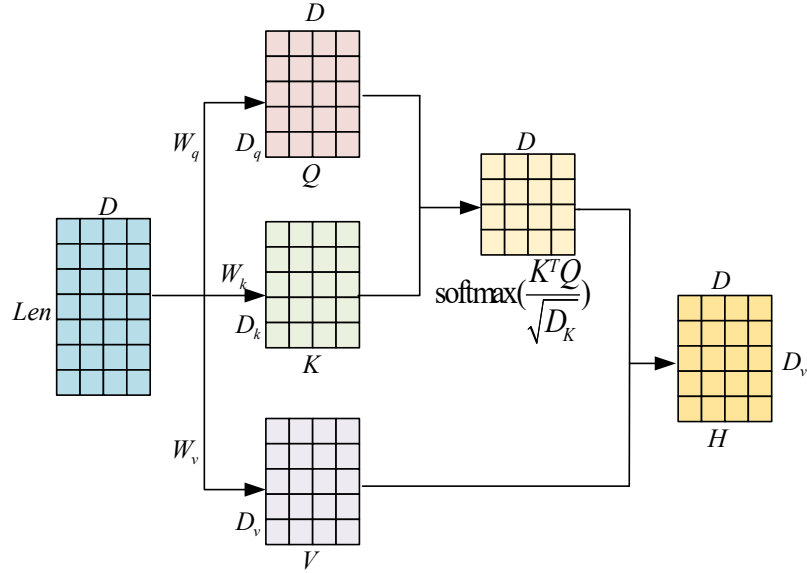


图 3.3 关联信息提取过程

### 3.2.3 局部特征提取

利用 CNN 模块对待测域名局部特征进行提取。在本章中 CNN 模块主要包括卷积层、最大池化层以及激活函数。在卷积层使用一维卷积进行卷积操作，获取待测域名局部特征信息，其中内核大小设置为  $\{2, 3, 4, 5\}$ ，各类滤波器个数为 128，见式(3.5)。然后将得到的特征图输入最大池化层池化并拼接池化值，得到待测域名局部特征表示。具体流程如图 3.4 所示。

$$h_i = \text{ReLU}(\sum_{j=1}^m w_j x_{i+j-1} + b) \quad (3.5)$$

其中， $h_i$  表示第  $i$  个位置的输出， $\text{ReLU}(\bullet)$  是激活函数， $w_i$  表示第  $j$  个滤波器的权重。 $x_{i+j-1}$  是位置  $i+j-1$  处的值的表示， $b$  是偏差。

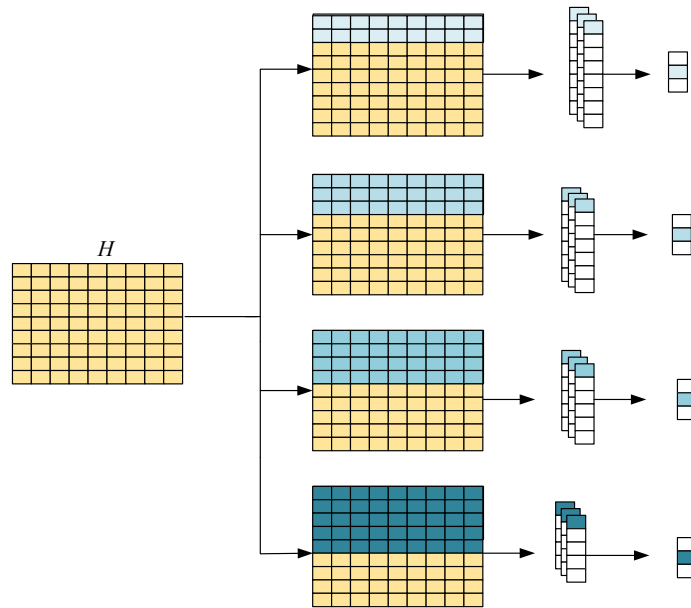


图 3.4 域名局部特征提取

### 3.2.4 全局特征提取

通过 BiGRU 模块提取域名全局特征。BiGRU 是一个双向 GRU，有两个方向，即正向 GRU 和反向 GRU，这种双向结构使得 BiGRU 能够捕捉到待测域名字符串序列中的长距离依赖关系<sup>[64]</sup>。如图 3.5 所示。

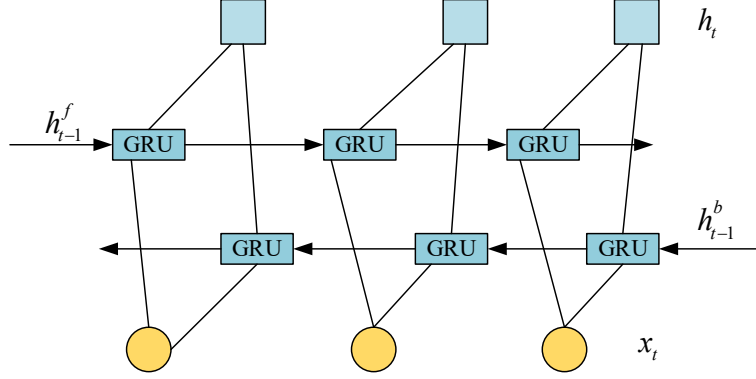


图 3.5 域名全局特征提取

将  $H$  输入 BiGRU 中，使用 BiGRU 计算具有权重值的特征向量，共分为三个阶段，其计算公式见式(3.6)-(3.8)。

$$h_t^f = GRU(x_t, h_{t-1}^f) \quad (3.6)$$

$$h_t^b = GRU(x_t, h_{t-1}^b) \quad (3.7)$$

$$h_t = w_t h_t^f + v_t h_t^b + b_t \quad (3.8)$$

其中， $x_t$  表示输入的待测域名向量， $h_t^f$ 、 $h_{t-1}^f$  分别是前向传播和上一步长中的隐含域名特征信息， $h_t^b$ 、 $h_{t-1}^b$  分别是后向传播和上一步长中的隐含域名特征信息， $h_t$  表示 BiGRU 的输出域名特征信息， $w_t$  和  $v_t$  分别表示  $h_t^f$ 、 $h_t^b$  的权重， $b_t$  表示偏置。

### 3.2.5 特征融合

特征融合模块将 CNN 和 BiGRU 提取到的特征作为输入，将融合后的特征作为输出，如图 3.6 所示。由于特征提取模块提取后的局部特征和全局特征进行融合时，往往会伴随特征饱和的问题。因此采用自适应特征融合，可以优化特征表达并提升模型性能。该模块能够动态地识别并筛选出域名关键的特征，同时抑制那些不相关或冗余的特征，从而有效降低模型的计算复杂度并提升处理效率。自适应融合特征的具体计算流程见式(3.9)。

$$Z = M(f_{CNN} \uplus f_{BiGRU}) \otimes f_{CNN} + (1 - M(f_{CNN} \uplus f_{BiGRU})) \otimes f_{BiGRU} \quad (3.9)$$

其中， $Z$  是融合后的特征， $f_{CNN}$  和  $f_{BiGRU}$  分别表示局部和全局特征， $\uplus$  表示初始特征集成，选择逐元素求和作为初始积分。特征融合模块框图如图 3.6 所示，其中虚线表示  $1 - M(f_{CNN} \uplus f_{BiGRU})$ 。

融合权重  $M(f_{CNN} \uplus f_{BiGRU})$  由 0 到 1 之间的实数组成，则  $1 - M(f_{CNN} \uplus f_{BiGRU})$  也是由 0 到 1 之间的实数组成，这使得网络能够在  $f_{CNN}$  和  $f_{BiGRU}$  之间进行加权平均。通过这种方法，模型更好地利用双层次的特征信息，提高检测性能。

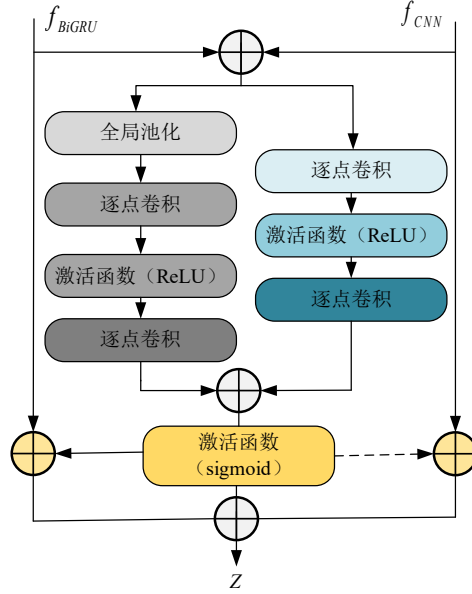


图 3.6 特征融合模块

### 3.2.6 域名分类

从特征融合模块获得域名特征向量后，全连接层将特征向量作为输入，将其从特征空间转换到样本标签空间。为了增强网络的表示能力，引入了一个隐藏层，并随后与 Sigmoid 激活层进行全连接。在全连接层中，每个神经元的输入与前一层神经元的输出均保持完全连接，建立了输入与输出之间的线性关系。通过这种连接方式，获得特征映射的结果，从而进一步提取出对域名分类的关键特征。最后，待测域名类别是通过 softmax 函数来确定。见式(3.10)。

$$y_i = \frac{e^{(w_j z_i + c)}}{\sum_{i=1, j=1}^2 e^{(w_j z_i + c)}} \quad (3.10)$$

其中， $y_i$  表示预测的输出概率，其值在 0 到 1 之间， $z_i$  为输入的待测域名特征向量， $w_j$  和  $c$  分别表示全连接层权重和偏置。

### 3.2.7 损失函数

DGA 产生的恶意域名涵盖多种家族，各家族之间的检测难易程度存在差异，导致恶意域名检测过程存在不平衡问题。为了减轻简单样本对损失计算的影响，使模型更加关注难分类的样本。本文引入了 Focal Loss 作为优化目标，增强模型对于各类样本分类能力的整体性能。损失函数见式(3.11)。

$$FL = \begin{cases} -\alpha(1-y')^\beta \log y', y=1 \\ -(1-\alpha)y' \log(1-y'), y=0 \end{cases} \quad (3.11)$$

其中， $y_i$  为待测域名真实标签， $y'_i$  为模型预测值， $\alpha$  用于调控域名样本比例， $\beta$  用于关注难分类的域名样本。

### 3.3 实验设计与结果分析

#### 3.3.1 数据集

为评估 CNN-BiGRU-AFF 恶意域名检测模型对恶意域名的识别能力，构建了两种数据集。第一种数据集(DN1)：合法域名数据集采用 Cisco 网站开源的前 1000000 网站列表中随机抽取 400000 条；恶意域名数据集采用 360 安全实验室发布的恶意域名列表中 40 个家族，共 396791 条恶意域名。为了平衡数据，对于数量多的家族，采取随机抽取 2 万条恶意域名的方法。通过交叉验证法将数据集按照 8:1:1 划分为训练集、验证集、测试集。在 AKAMAI 公司执行 DNS 查询处理过程中的分析表明，所有解析过的域名中，约有 20%为恶意域名。为了验证模型在现实环境下的性能，构建第二种数据集（DN2）：来自文献[65]的数据集，并按照 4:1 划分合法域名和恶意域名，其中包括 160000 条合法域名，40000 条恶意域名。设定合法域名标签为 0，恶意域名标签为 1。数据集划分如表 3.1 所示。

表 3.1 DN1 数据集划分

类别	合法域名	恶意域名	总数量
训练集	320000	317433	637433
验证集	40000	39679	79679
测试集	40000	39679	79679
测试集	160000	40000	200000
总量	560000	836791	1396791

#### 3.3.2 实验环境及评价指标

本章的实验环境如表 3.2 所示。

表 3.2 实验环境配置

实验环境	参数
操作系统	Windows 10
CPU 处理器	i7-1165G7 512G 4.8GHZ
GPU	NVIDIA GeForce RTX 3060 6GB
存储空间	16GB
加速库	CUDA11.0
编程语言	Python3.6.7

在进行实验之前，必须对模型的超参数进行初始化。超参数无法通过分析域名数据直接获得，因此需要手动配置来调整，以提升模型的整体表现。目前，设置超参数主要依赖于领域专家的经验 and 通过实验累积的知识。如表 3.3 所示。

表 3.3 实验超参数设置

超参数名称	参数值
优化器	Adam
Learning Rate	0.001
填充值	0
激活函数	ReLU、sigmoid、softmax
训练迭代次数	20
损失函数参数 $\alpha$	0.5
损失函数参数 $\beta$	2

实验采用准确率、召回率、精确率、F1 值以及 AUC 作为评价指标，全面分析和优化模型的性能。将评价指标作为对模型进行优化的依据，实现对检测精度的提升，从而保证模型具备更高的性能，确保对恶意域名进行准确识别。

### 3.3.3 对比实验

为了更全面地验证本章提出的 CN-BiGRU-AFF 模型的有效性，在数据集上与其他 5 种恶意域名检测模型进行对比，如表 3.4 所示。

表 3.4 基准模型

模型结构	参考文献
BiLSTM	[33]
TextCNN	[35]
LW-TextCNN	[36]
CNN-LSTM	[37]
CNN-BiLSTM	[38]

#### (1) DN1 数据集的对比实验

由表 3.5 可以看出，本章提出的基于 CNN-BiGRU-AFF 的恶意域名检测模型，在准确性、召回率、精确率及 F1 分数表现优异。BiLSTM 作为一种双向长短期记忆网络，虽然精度低，但其结构简单。CNN-BiGRU-AFF 与 TextCNN 比较，优势不明显，因为 TextCNN 也使用并行结构，对域名特征提取较全面，但 CNN 对文本全局特征提取能力较低。LW-TextCNN，是 TextCNN 的轻量级版本，在检测精度上虽低于 TextCNN，但其参数少，模型小的特点，与 CNN-BiGRU-AFF 模型在多个性能指标上表现差异也比较小。CNN-LSTM 采用单向 LSTM 对上下文特征提取不充分，使得该模型表现低于 CNN-BiLSTM。虽然 BiLSTM 与 BiGRU 有相似结构和功能，但本章模型与 CNN-BiLSTM 比较，具有明显的性能优势。因为本模型增加了注意力模块，对提取关键特征有更高的效率。实验表明，本章模型综合性能表现优异。

表 3.5 DN1 数据集对比实验结果

模型检测	准确率	召回率	精确率	F1 值
BiLSTM	0.9551	0.9421	0.9623	0.9520
TextCNN	0.9780	0.9739	0.9803	0.9772
LW-TextCNN	0.9751	0.9720	0.9779	0.9727
CNN-LSTM	0.9542	0.9491	0.9619	0.9554
CNN-BiLSTM	0.9771	0.9711	0.9725	0.9717
CNN-BiGRU-AFF	<b>0.9852</b>	<b>0.9827</b>	<b>0.9889</b>	<b>0.9857</b>

## (2) DN2 数据集对比实验

为了验证提出的 CNN-BiGRU-AFF 检测模型在真实环境下的性能，用 DN2 数据集作测试集。采用准确率、召回率、精确率、F1 值，作为评价指标。在以上指标中均表现出良好的检测效果，如表 3.6 所示。对比实验结果表明本章提出的检测模型，在综合评价指标上超越了其他比较模型。能够更有效地提取域名特征，表现出更高的检测性能。

表 3.6 DN2 数据集对比实验结果

模型检测	准确率	召回率	精确率	F1 值
BiLSTM	0.9241	0.9221	0.9323	0.9271
TextCNN	0.9399	0.9369	0.9373	0.9371
LW-TextCNN	0.9451	0.9501	0.9522	0.9511
CNN-LSTM	0.9323	0.9341	0.9403	0.9371
CNN-BiLSTM	0.9459	0.9461	0.9455	0.9457
CNN-BiGRU-AFF	<b>0.9547</b>	<b>0.9507</b>	<b>0.9579</b>	<b>0.9542</b>

### 3.3.4 消融实验

本模型采用 CNN 与 BiGRU 并行结构的策略，为了验证本研究提出的模型中不同网络结构，如串行、全连接融合特征等对域名分类效果的影响，在 DN1 数据集构建的测试集上进行消融实验，并采用了 TPR@FPR 这项指标进行性能评估。具体的测试结果与网络性能比较如表 3.7 所示。

其中五种模块的结构描述如下：

(1) BiGRU-Attention：在本模型中用于域名字符串上下文序列进行特征学习，获取域名的全局特征。

(2) CNN-Attention：在本模型中用于提取域名字符串局部关键进行特征学习。获取 BiGRU 未获取到的局部关键特征。

(3) CNN-BiGRU：采用并行方式进行特征提取，但是采用全连接的方式对两种域名特征进行融合。

(4) Series-CNN-BiGRU-AFF:采用与本章模型相同的模块,但采用串联方式构建的检测模型。

(5) CNN-BiGRU-AFF:本章提出的模型,采用并联方式构建的检测模型,并且采用 AFF 对特征进行动态融合。

表 3.7 消融实验结果

	TPR@FPR Level				AUC
	0.0001	0.001	0.01	0.1	
Global:					
BiGRU-Attention	0.6233	0.8912	0.9390	0.9542	0.9712
Local:					
CNN-Attention	0.6165	0.8401	0.9289	0.9416	0.9656
FC:					
CNN-BiGRU	0.7525	0.9328	0.9457	0.9875	0.9937
Whole:					
Series-CNN-BiGRU-AFF	0.7181	0.9351	0.9694	0.9968	0.9955
CNN-BiGRU-AFF	0.7395	0.9429	0.9870	0.9979	0.9979

特征组合方面,在全局特征中加入局部特征和在局部特征中加入全局特征都可以提高 AUC,验证了模型对域名特征的提取能力会影响检测效果。Series - CNN-BiGRU-AFF 和 CNN-BiGRU-AFF 利用了两种特征显著优于 CNN-AFF 和 BiGRU-AFF;网络结构方面,串联结构各项指标低于并联结构,因为两个模块分别用于提取两种不同的特征表示,在模型推理时 BiGRU 处理 CNN 过滤之后的特征信息,限制了 BiGRU 对域名全局特征的分析处理。在所有指标中本章提出的模型都有更大的优势。

CNN-BiGRU-AFF 能够有效捕捉域名多种类型的信息,而且使用 AFF 模块更加关注关键的特征,实现特征间的有效融合,获得更全面的域名特征向量表示。与 CNN-BiGRU 相比,网络的特征融合能力能影响网络的检测结果,引入 AFF 模块能提升模型的检测精度。

本章采用并联的 CNN 和 BiGRU,然后再使用 AFF 模块进行特征融合。为了更加直观分析本模型在结构上的优势,将本章并联式 CNN-BiGRU-AFF 和串联式 CNN-BiGRU-AFF 的 AUC 进行可视化,如图 3.7 所示。并联结构的允许模型以并行方式独立处理并整合来自 CNN 和 BiGRU 的特征信息,从而实现了对恶意域名隐藏特征的学习。这种结构不仅优化了信息流的传递效率,还增强了模型对复杂域名识别的能力,使得检测精度得到显著提升。此外,可视化分析结果清楚地揭示了并联式结构在处理恶意域名检测任务时,相比于传统串联式结构,能够提供更高的性能。

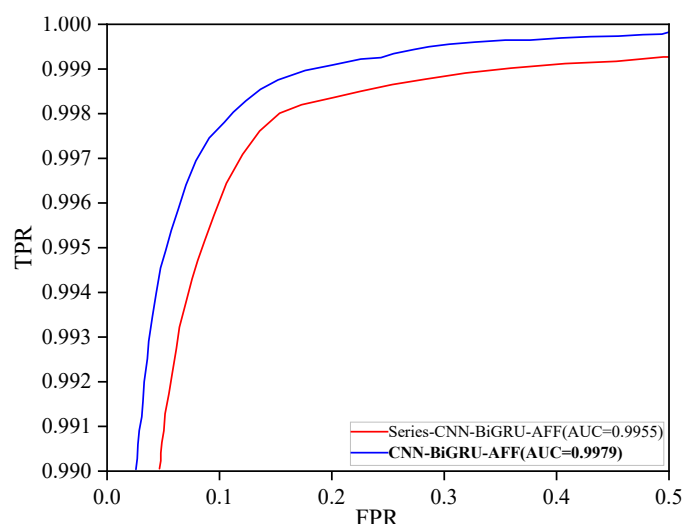


图 3.7 串联模型和并联模型 AUC

### 3.4 本章小结

本章针对目前恶意域名检测模型对提取特征单一，特征利用不充分，提出了一种并行 CNN-BiGRU-AFF 模型来检测恶意域名。通过提取域名全局和局部特征，然后采用动态融合的方式，将两种特征有效融合，提高了域名特征的利用率，最后通过分类网络输出域名的类别最大概率分布。通过构建两种不同的数据集对模型进行验证，并与现有模型比较，本模型各类指标有明显提升。但在模拟真实情况下的数据集 DN2 上，模型检测精度有明显下降。这说明本模型泛化能力不强。后续将设计对数据集依赖性低的恶意域名检测模型。

## 第4章 基于自适应深度变分自编码器的恶意域名检测

### 4.1 引言

互联网及移动和边缘计算技术的发展带来了日益增加的网络安全挑战，其中恶意域名的出现频率和多样性也随之上升。这些域名常用于执行网络攻击如钓鱼和恶意软件传播，并不断演化以规避传统安全监测。为应对这些挑战，通过分析恶意域名生成算法的规律，利用如生成数据集的方法来增强模型训练，从而提升对新出现或新变种恶意域名的识别能力。尽管如此，但已知域名样本的数量远超过未知恶意域名样本，这种不平衡问题仍然限制了恶意域名检测技术的发展。这导致模型在训练过程中过分适应已知域名的特征，而忽略新产生的恶意域名，进而可能在实际应用中引发较高的误报率。因此，尽管深度学习在提升恶意域名检测准确性方面取得了一定成果，但上述问题仍阻碍恶意域名检测模型性能提升。

在第三章中，探索了一种从多维度提取域名特征的复合结构模型，用于提升模型对域名特征的提取能力和检测精度。该模型通过 CNN 提取域名中的局部特征，结合 BiGRU 捕获序列中的长期依赖关系，而 AFF 则有助于对两种特征的有效融合，整体上显著增强了恶意域名检测的鲁棒性。但该模型对新出现或新变种的恶意域名检测泛化性差。

发生上述问题的根本原因是域名数据不平衡。因为恶意域名在发生攻击之后才能加到恶意域名黑名单中，基于有监督的恶意域名检测深度学习模型难以及时捕获新出现或新变种的恶意域名的特征，导致模型对该类恶意域名的泛化性差，为此提出了基于自适应深度变分自编码器的恶意域名检测方法。首先通过分析合法域名和恶意域名存在的特征差异，初步评估本章模型的可行性。其次将待测域名进行预处理，利用深度变分自编码器网络对待测域名进行重建；最后通过变分自编码器潜空间采样，获得域名特征重建概率计算恶意得分，结合神经网络分类器能够自适应设置阈值的优点将域名分类为合法域名与恶性域名。在大量数据集上进行验证，结果表明，能有效缓解恶意域名检测模型对新出现或新变种恶意域名泛化性差的问题。

### 4.2 检测流程设计

基于自适应深度变分自编码器的恶意域名检测方法，主要包含训练和检测两阶段。训练阶段使用的数据为合法域名数据。首先，对域名进行数据清洗并进行正则化处理，便于后续模型更有效地学习合法域名的特征分布；然后通过深度变分自编码网络重建每个域名特征；最后将合法域名样本特征的正态分布作为输

出，构建自适应深度变分自动编码器模型，并通过计算模型输出值和未处理数据之间的差异优化网络参数和权值。在检测阶段，利用训练后的模型计算待测域名的恶意得分，并根据恶意得分识别合法域名与恶意域名。本章恶意域名检测方法流程如图 4.1 所示。

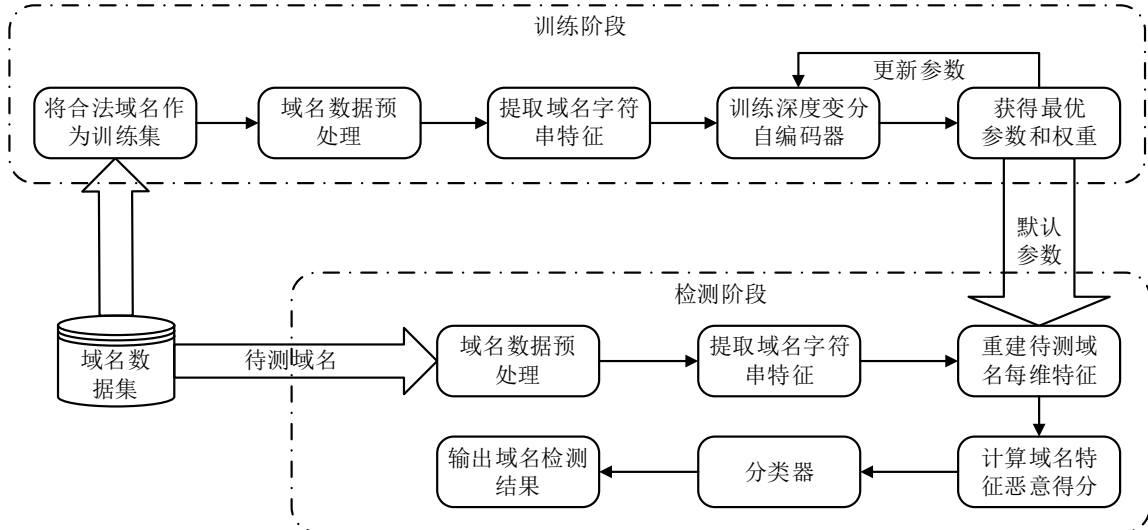


图 4.1 基于自适应深度变分自编码器的恶意域名检测流程

#### 4.2.1 域名数据预处理

域名由标点符号、特殊字符以及各级域名（顶级域名、二级域名等）组成。顶级域名因其数量有限、长度短、且普遍知名度较高，很少直接被用于构造恶意域名。恶意域名与合法域名之间的区别通常更明显地体现在二级域名、三级域名及其他更深层次的域名结构上。因此，将域名顶级域名、顶级域名后的“.”等项去除，然后将域名字符向量化。

字符向量化是将字符转换为深度学习算法可以理解的数值形式。One-hot 编码是一种常见的向量化方法，为每个唯一的字符分配一个唯一的索引，并使用 0 和 1 的向量表示字符出现的位置，但这种方法获得的高维稀疏矩阵使得模型处理效率低下，同时丢失了字符间的顺序信息。相比之下，Word2Vec 通过训练将字符映射到固定大小的密集向量空间中，这些向量捕捉了字符间的关系，为字符间的相似性提供了量化度量。特别是在恶意域名检测领域，Word2Vec 能够从域名中提取有意义的特征，使得模型不仅能够识别已知的恶意模式，还能泛化到新出现的恶意域名上，提高了恶意域名检测的准确性和效率。

以域名“Baidu”为例，域名量化过程如图 4.2 所示。由于模型输入为定长，设  $Len$  作为模型输入长度，将  $Len$  设置为 64。域名长度小于 64 时，采用零向量补全法；当域名长度大于 64 时，将域名多余部分裁剪。

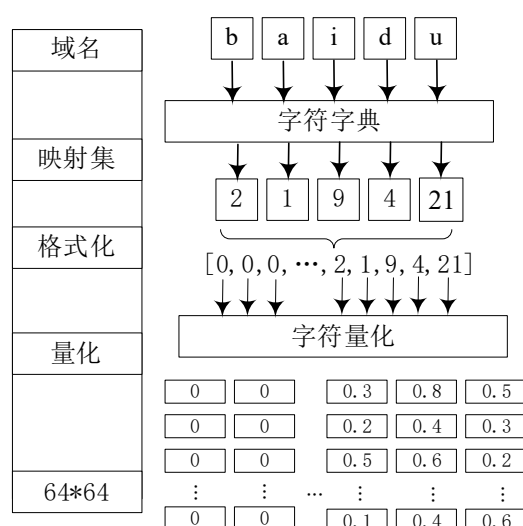


图 4.2 域名向量化过程

## 4.2.2 深度变分自编码器训练

### (1) 域名特征分析

为了验证域名的每个特征之间是否存在差异，从而证明本模型的可行性。选取如表 4.1 所示的域名特征类型对恶意域名与合法域名的特征进行分析。为了更好地理解合法域名和恶意域名之间的差异，还将域名特征的分布可视化分析。

表 4.1 域名特征

域名特征	特征含义
域名长度	域名字符串中含有字符个数
数字字符比率	域名字符串中数字出现的总数除以域名长度
分隔符“.”频率	分隔符“.”出现的次数
连续字符串长度	域名中连续字符串序列的最大长度
字符串字符频率	单个字符串字符出现的次数的矩阵
特殊字符比率	特殊字符出现总次数除以域名长度
元音字符频率	元音字符 a、e、i、o、u 出现的总次数
连续相同字符长度	字符串中连续的相同字符序列的长度
连续辅音的最大长度	字符串中具有相同的辅音字符序列的最大长度
分隔符内最大字符串长度	两个相邻分隔符之间字符串中字符最大长度
信息熵	域名中字符分布的随机性或不确定性程度
2-gram	域名中相邻的两个字符序列

**域名长度：**为了便于用户记忆和输入，提供更好的用户体验，通常合法域名采用比恶意域名短的字符串表示，由一个或两个英文单词或中文拼音组成，例如“google.com”、“baidu.com”等。合法域名的长度通常在 20 个字符以内很少会超过 20 个字符，大多数情况下在 5-20 个字符之间，如图 4.3 所示。因此，域名长度特性对于识别合法域名具有重要意义。

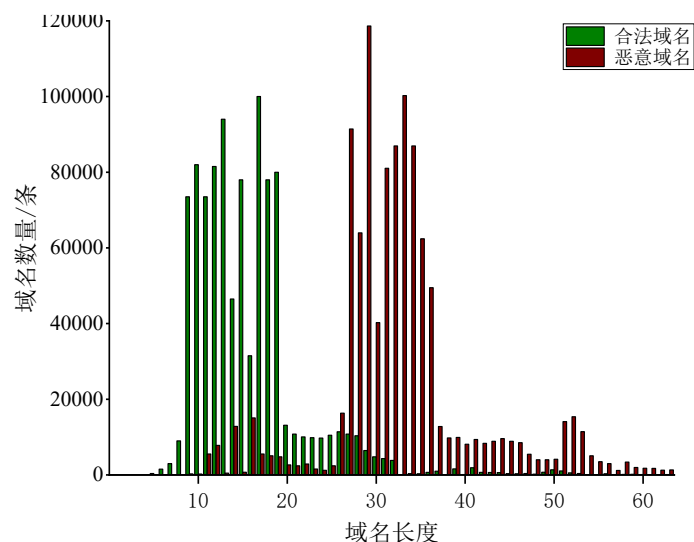


图 4.3 域名长度分布

**数字字符比率：**域名可以由字母、数字和连字符随机组合，但为了满足域名注册目的，如反映网站业务、品牌保护等，域名字符组合按照人们用于习惯进行设计。合法域名则通常较少使用数字字符，而部分恶意域名将大量数字字符包含在域名中以混淆检测模型，如图 4.4 所示。因此，数字字符比率特征能够作为检测模型识别恶意域名的重要判别依据之一。

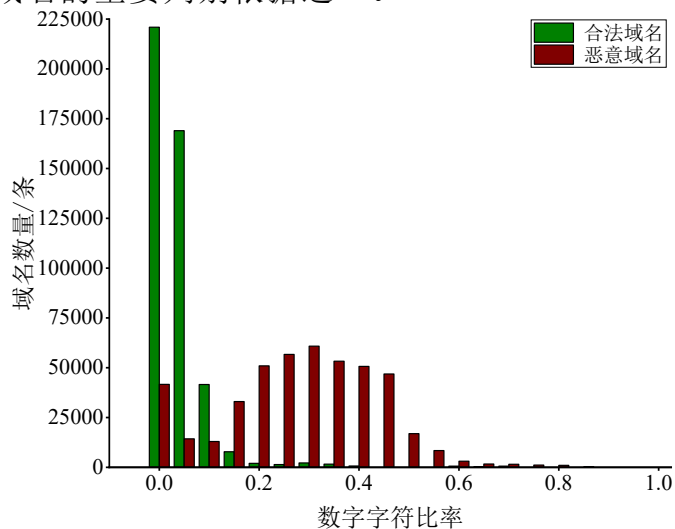


图 4.4 数字字符比率分布

**分隔符内最大字符串长度：**在设计域名时，为了提高用户的记忆效率，合法域名通常会采用较短的字符串或单词，这些字符串不仅简短，而且含义明确、易于理解。相反，恶意域名为了规避检测、增加识别的难度，会在相邻两个分隔符内使用大量连续的字母或数字，字符串组合复杂且没有实际意义。因此，分隔符内最大字符串长度成为了识别恶意域名的有效特征之一。域名中两个分隔符之间的最大字符串长度如果明显长于常见的合法域名，则此域名是恶意域名的可能性很大。通过将这一特征与其他域名特征进行融合，可以显著提高模型的性能。

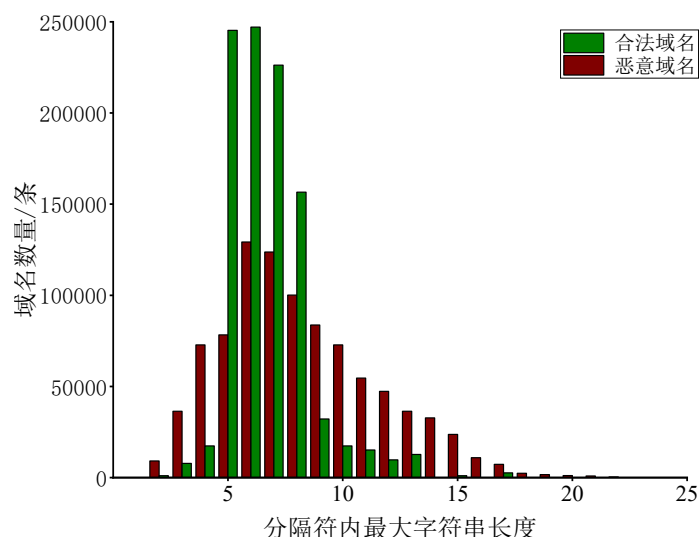


图 4.5 分隔符内最大字符串长度分布

**域名信息熵：**域名信息熵是指对于一个域名字符串的熵值，用来衡量域名字符串中字符的分布情况，以及字符的随机性或不确定性，其中可以用香浓熵公式表示不确定度，见式(4.1)高熵值表示域名字符串中的字符分布更加随机或不均匀。反之低信息熵表示域名字符串中的字符分布规律是合法域名的可能性大。

$$H(X) = -\sum_{i=1}^n p(x_i) \times \log p(x_i) \quad (4.1)$$

其中， $X$ 表示域名， $x_i$ 表示域名中的某个字符， $p(x_i)$ 表示该字符在域名字符串中出现的频率。域名信息熵分布如图 4.6 所示。

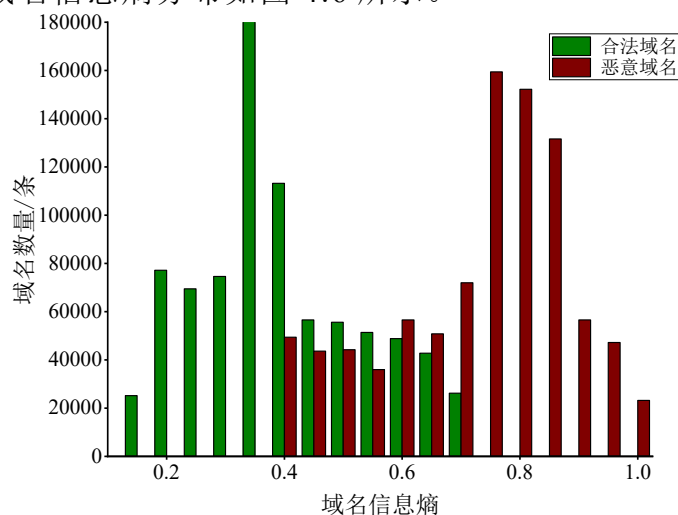


图 4.6 信息熵分布

## (2) 预训练

为了解决深度变分自编码器网络在训练过程中遇到的“梯度消失”问题，采用分层递进的贪心训练策略，针对每一层网络进行优化。具体方法是，从第一层变分自编码网络开始，通过最小化原始输入数据与目标数据之间的重建损失来训练网络。完成后，利用第一层变分自编码网络的输出作为下一层变分自编码网络的输入，继而逐步减少重建损失。这一过程持续进行，直至所有层的变分自编码网

络训练完成。

Step1: 输入  $H_1$  单层自编码器, 完成训练后移除  $H_1$  解码器, 保留编码过程。

Step2: 使用  $H_1$  输出作为  $H_2$  输入, 完成训练后移除  $H_2$  解码器。

Step3:  $H_2$  输出转为  $H_3$  输入, 训练后移除  $H_3$  解码器。

Step4: 重复上述步骤, 直至所有的网络层训练完成。

### (3) 参数微调

参数微调依赖于有监督学习方法, 减少输入数据与目标输出之间的重建损失。通过反向传播算法, 将各网络层的参数进行微调。训练流程如图 4.7 所示。

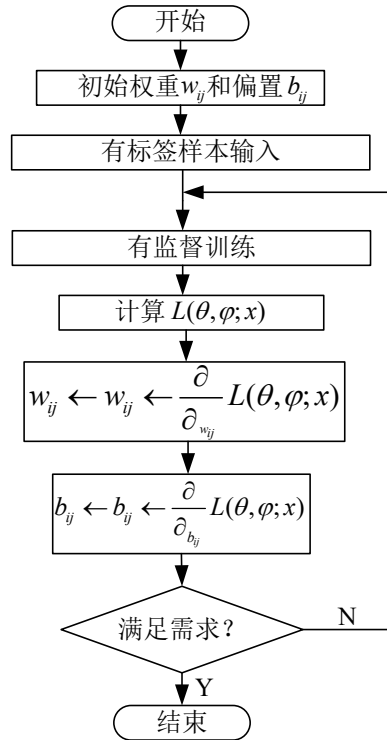


图 4.7 参数微调

## 4.2.3 自适应阈值分类器构建

### (1) 重建概率

重建概率表示给定隐变量的情况下, 模型能够生成与原始输入样本的概率, 是域名进行分类的重要依据。重建概率与损失函数密切相关, 见式(4.2)。

$$\begin{aligned}
 L(\theta, \varphi; x) &= E_{q_\varphi(z|x)} [-(\log q_\varphi(z|x) + \log p_\theta(z)) + \log p_\theta(x|z)] \\
 &= -D_{KL}(q_\varphi(z|x) \| p_\theta(z)) + E_{q_\varphi(z|x)} [\log p_\theta(x|z)]
 \end{aligned} \quad (4.2)$$

其中重建概率是  $E_{q_\varphi(z|x)} [\log p_\theta(x|z)]$  的蒙特卡洛估计, 不能直接计算, 需要通过采样过程求解。采样过程是在分布  $z_{i,l} \sim q_\varphi(z|x_i)$  中对隐变量  $z_{i,l}$ ,  $l=1,2,3,\dots$ , 进行  $K$  次采样求平均。经过此过程可以利用真实样本与期望样本之间的距离求解重建概率。然而为了能够准确计算这个过程, 可以采用类似自编码器重建误差的计算值代替重建概率, 以衡量原始域名与重建后的域名的差异, 本章采用均方误差 MSE 作为距离度量。见式(4.3)。

$$MSE = \frac{1}{n} \sum_{i=1}^n (x_i - \hat{x}_i)^2 \quad (4.3)$$

其中,  $n$  表示特征点,  $x_i$  和  $\hat{x}_i$  分别表示实际值和预测值。

## (2) 恶意分数

恶意分数是用来测量深度变分自编码器模型在训练阶段获取的合法域名特征分布与重建域名特征分布之间的偏离距离。待测域名样本和其重建后所有特征的平均值之差作为恶意得分, 恶意得分的定义见式(4.4)。

$$s(x) = \left( x - \frac{1}{n} \sum_{i=1}^n \hat{x}_i \right) \left( x - \frac{1}{n} \sum_{i=1}^n \hat{x}_i \right)^T \quad (4.4)$$

由式(4.4)可得恶意域名检测得分向量  $S = \{s(x_1), s(x_2), \dots, s(x_n)\}$ 。对于合法域名, 训练完成的模型对合法域名重建能力强, 则其恶意得分低; 对于恶意域名, 模型在训练阶段没有学习相关特征, 则其恶意得分高。

## (3) 分类器

当待测域名样本的恶意分数大于阈值时, 判定其为恶意域名; 否则, 为合法域名。因此选择合适的阈值有重要意义。选择基于核密度估计 (Kernel Density Estimation, KDE) 方法进行阈值设置。将合法域名的恶意得分输入分类器, 使用 KDE 估计合法域名恶意得分的概率密度, 然后利用累积分布函数 (Cumulative Distribution Function, CDF) 获取合法域名恶意得分的概率分布, 最后结合显著性水平来设置阈值。

由构成训练集的合法域名样本恶意得分向量  $S$ , 可以利用 KDE 获得概率密度函数见式(4.5)。

$$p(s) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{s-s_i}{h}\right) \quad (4.5)$$

其中,  $s_i$  是训练集中第  $i$  个域名样本的恶意得分;  $K(\bullet)$  是核函数;  $h$  是带宽。本章采用径向基核函数 (Radial Basis Functions, RBF), 见式(4.6)。

$$K_{RBF}(s; h) \propto \exp\left(-\frac{s^2}{2h^2}\right) \quad (4.6)$$

其中,  $h$  表示带宽, 其计算见式(4.7)。

$$h = \left[ \frac{n(d+2)}{4} \right]^{-\frac{1}{(d+4)}} \quad (4.7)$$

由式 (4.5) 可得 CDF 公式。见式(4.8)。

$$F(s) = \int_{-\infty}^s p(s) ds \quad (4.8)$$

其中,  $\int_{-\infty}^s$  表示域名恶意得分的概率累计。然后结合显著性水平  $\alpha \in [0, 1]$ , 设置阈值为  $s_\alpha$ , 其满足  $F(s_\alpha) = 1 - \alpha$ 。当待测域名样本的恶意得分  $s \geq s_\alpha$  时, 其有  $(1 - \alpha) \cdot 100\%$  的概率是恶意域名。

#### 4.2.4 恶意域名识别

在恶意域名识别过程中，按照待测域名恶意得分与阈值比较给出是否为恶意域名的判断。

**Step1:** 域名数据集进行预处理，包括词域名向量化和数据集划分。其中训练集只包含合法域名，测试集包含合法和恶意域名。

**Step2:** 训练模型，输入训练集  $X = \{x^{(i)}\}_{i=1}^n$  到深度变分自编码器网络中进行重建，获取合法域名的恶意得分向量  $S \in R_n \times d$ 。

**Step3:** 阈值设定，利用 Step2 中得到的恶意得分向量，结合 CDF 和显著水平  $\alpha$  自动设置阈值  $s_\alpha$ 。

**Step4:** 恶意域名识别，将待测域名样本输入具有最优参数的深度变分自编码网络模型，获得该样本的恶意得分  $S_{new}$ ，如果  $s_{new} \geq s_\alpha$  则判为恶意域名，否则为合法域名。

### 4.3 实验与分析

#### 4.3.1 数据集

为了验证本章模型的性能，从 Alexa、Cisco、360Netlab 以及 DGArchive 搜集整理了 1836164 条合法域名和恶意域名用于构建本实验数据集。其中包括来自 Alexa 统计的 500000 条合法域名、Cisco 网站中的 500000 条、360Netlab 开源的恶意域名数据集中 413652 条以及 DGArchive 网站中的 411512 条。共计 1824652 条域名。为保证样本数据的多样性，对于数量充足的恶意域名家族只采用前 30000 条。如表 4.2 所示。

表 4.2 数据集描述

类型	来源	数目
合法域名	Alexa	500000
	Cisco	500000
恶意域名	DGArchive	411512
	360Netlab	424652

为了验证该模型不依赖于某个特定的训练集，使用由不同数据集组成的两种训练集来训练模型。此外，为了确保 Ada-DVAE 和基准模型都能学习到最优参数和权重，根据模型训练需求 Ada-DVAE 模型仅使用合法域名进行训练，基准模型使用相同的合法域名和额外的恶意域名进行训练。其次验证数据集规模对模型性能的影响，评估我们的模型在训练集较小的情况下是否能保持其检测性能。最后，验证模型对数量少的恶意域名家族的检测精度。下面描述了训练集组合。

(1) **Dx:** 由 Cisco 数据集部分样本构成作为 Ada-DVAE 模型训练集；

- (2) Dm: 由 Alexa 数据集部分样构成作为 Ada-DVAE 模型训练集;
- (3) DxDn: 由 Cisco 和 360Netlab 数据集构成作为基准模型的训练集;
- (4) DmDh: 由 Alexa 和 DGArchive 数据集构成作为基准模型的训练集。

在现实情况中, 恶意域名检测模型面临的是不平衡的检测数据, 即合法域名的数量远高于恶意域名的数量。为了模拟现实情况, 按照合法域名样本和恶意域名样本 4:1 的比例来划分测试集。并且训练集和测试集域名数量之比约为 4:1。由于本模型的训练集与基准模型数据集构建有明显差异, 因此根据模型训练需求进行数据集划分, 如表 4.3 所示。

表 4.3 数据集划分

数据集与模型	训练集		测试集	
	合法域名	恶意域名	合法域名	恶意域名
DxAda-DVAE	400000	—	100000	25580
DmAda-DVAE	400000	—	100000	25580
DxDn 基准模型	400000	399675	100000	25580
DmDh 基准模型	400000	399786	100000	25580

### 4.3.2 实验环境及评价指标

本章实验环境配置如表 4.4 所示。

表 4.4 实验环境配置

实验环境	参数
操作系统	Windows 10
CPU 处理器	i7-1165G7 512G 4.8GHZ
GPU	NVIDIA GeForce RTX 3060 6GB
存储空间	16GB
加速库	CUDA11.0
编程语言	Python3.6.7

超参数设置如表 4.5 所示。

表 4.5 实验超参数设置

超参数名称	参数值
网络结构	64-30-20-10-5
Learning Rate	0.001
激活函数	ReLU
迭代次数	90
采样次数	5
微调次数	40

### 4.3.3 对比实验

本章提出并实现了基于自适应深变分自编码器的恶意域名检测模型，为了验证本模型的性能优劣，分别与文献[42]中的 AN-LSTM 模型和第三章中的 CNN-BiGRU-AFF 模型做比较。对比实验，采用准确率、召回率、精确率、F1 值作为评价指标。

如表 4.6 所示，虽然本章模型仅使用合法域名进行训练，但在每个评估指标中都优于其他的检测模型。基准模型显示了一个缺点，即与精度相比，模型伴随着较低的召回率。较低的召回率意味着存在假阴性错误，模型将恶意域名识别为合法域名。这种假阴性错误至关重要，因为该模型无法区分具有与合法域名相似形式的恶意域名。另一方面，本章模型同时达到了较高的准确率和召回率，这意味着本模型减少了假阳性误差。原因是模型通过良好的训练，对合法域名能够更加准确的重建其表示，而对恶意域名缺乏相关的特征学习导致其重建效果差，因此分类器可以很容易地从合法域名中识别出恶意域名。此外，无论训练集的类型如何，该模型都表现了优异的检测性能。该实验表明本章提出的模型能有效检测恶意域名。

表 4.6 对比实验结果

模型	准确率	召回率	精确率	F1 值
DxDnAN-LSTM	0.9603	0.9601	0.9689	0.9644
DmDhAN-LSTM	0.9501	0.9491	0.9515	0.9502
DxDnCNN-BiGRU-AFF	0.9728	0.9737	0.9757	0.9746
DmDhCNN-BiGRU-AFF	0.9622	0.9613	0.9606	0.9609
<b>DxAda-DVAE</b>	<b>0.9825</b>	<b>0.9812</b>	<b>0.9813</b>	<b>0.9801</b>
<b>DmAda-DVAE</b>	<b>0.9829</b>	<b>0.9813</b>	<b>0.9827</b>	<b>0.9812</b>

通过对模型在受限数据集上的检测性能进行验证，表明了除非使用极少量的训练数据，否则本模型能够展现出稳定的性能，显著降低了恶意样本数据采集的难度。为了评估在有限的训练数据环境下的模型。通过调整训练数据集的规模，对 Ada-DVAE 模型与基准模型的检测性能进行了对比，实验结果，如图 4.8 所示，证明了本章提出的模型对于小规模训练数据依然表现出优异的性能。相比之下，基准模型的检测性能随着训练数据的减少而明显下降。此外，在训练数据量达到 50000 条后，Ada-DVAE 模型的准确率和 F1 分数展现出一致性。进一步分析发现，即使在训练数据规模变化小时，准确率和 F1 分数依然保持在 0.96 左右，表明即便在受限的数据环境下，Ada-DVAE 也能有效识别恶意域名。在由 Dx 和 Dm 组成的不同类型的训练集上，Ada-DVAE 模型都保持显著的性能优势。实验表明，尽管在训练数据受限的情况下，模型仍能有效检测恶意域名，证明了其对数据集依赖较小。

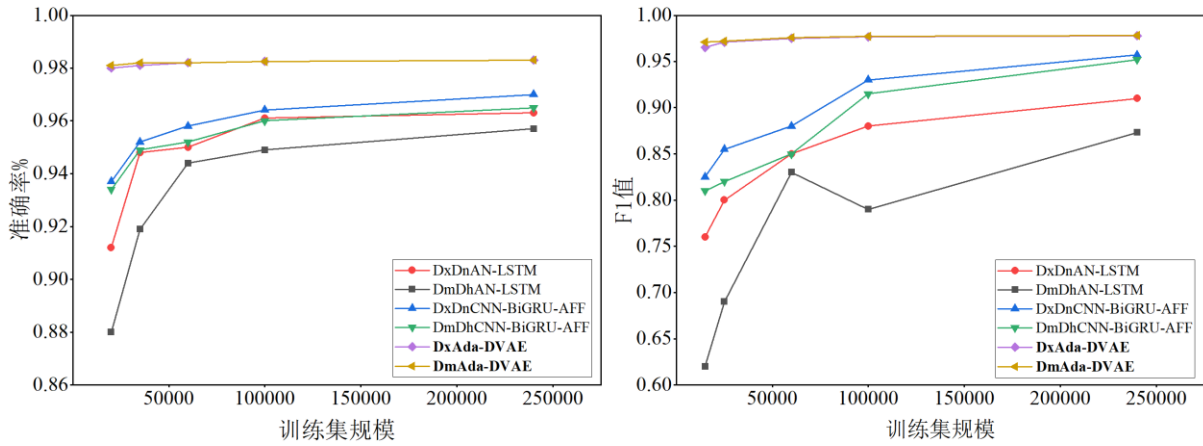


图 4.8 训练集对准确率和 F1 值的影响

如图 4.9 所示，为更加全面的评估模型的检测性能，利用 8 个样本数量少的恶意域名家族构建了域名黑名单，其中 **suppobox**，**chinad** 属于难检测家族。本章模型分别和上述基准模型做对比。

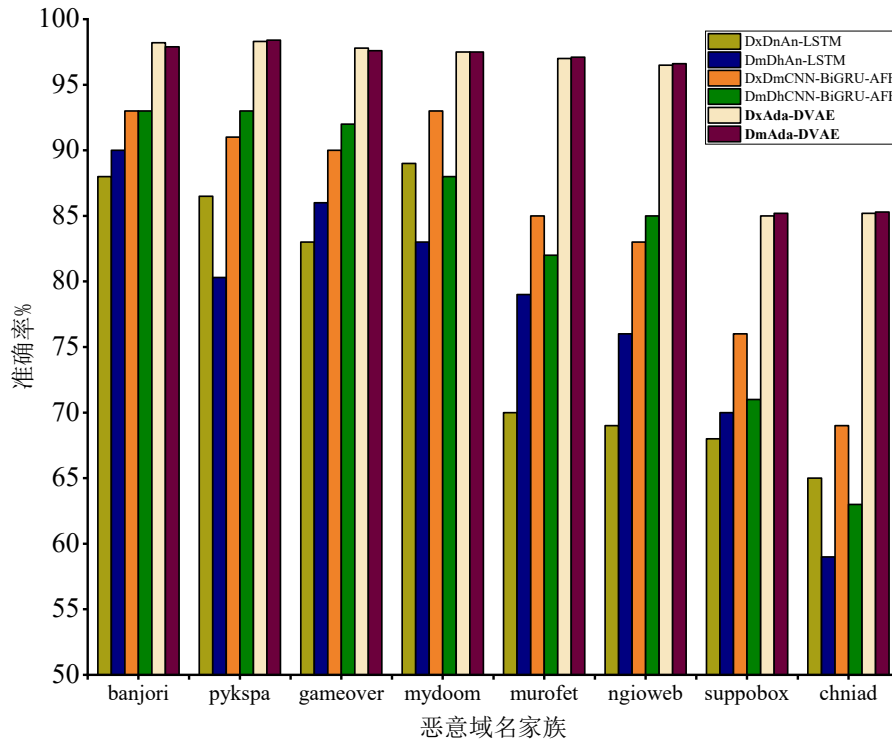


图 4.9 样本数量少的恶意域名家族检测结果比较

本章模型在前六个 DGA 家族中，检测的准确率保持在 97%以上，并且在 **suppobox**、**ngioweb** 这两个检测难度大的 DGA 家族中的准确率明显高于其他模型，达到 85%以上，进一步验证了基于自适应变分自编码器检测恶意域名方法的有效性。

由实验结果分析可知，四个基准模型利用在训练阶段学习到的恶意域名特征能够有效检测普通的 DGA 家族，但是对于负样本少、难以识别的 DGA 家族检测准确率较低。Ada-DVAE 模型采用基于域名重建概率，侧重挖掘合法域名隐藏特征，增强模型对合法域名的辨识度，可提高模型的泛化能力。

#### 4.4 本章小结

提出了一种基于自适应深度变分自编码器的恶意域名检测模型，通过深度变分自编码器网络对域名进行重建，利用重建过程的获得的重建概率计算恶意得分，并输入分类器，在分类阶段通过设计的自适应阈值分类器识别恶意域名。为了能够全面验证本模型的有效性，构建了不同数据集，并且进行了多个对比实验。实验结果表明，本模型减少了对数据集的依赖，从而缓解了现有恶意域名检测模型对新出现或新变种的恶意域名检测性能差的问题。

## 总结与展望

本文聚焦于在数字时代的快速发展中，随着互联网的普及而面临的网络安全问题。其中恶意域名检测是主要的研究领域，该领域的研究致力于识别那些可能导致网络攻击、恶意软件扩散及其他安全威胁的域名，为网络安全防御提供科学依据和技术支持。在恶意域名检测的初期阶段，研究主要集中在应用传统机器学习算法，依赖于手动提取的特征和精心设计的分类器来区分恶意域名。然而，随着网络攻击手段的不断演进和复杂化，这些传统方法的局限性逐渐显现。为了应对这一挑战，通过引入先进的深度学习架构，能够使模型自动从域名数据中学习特征，从而显著提高了恶意域名检测的准确性和模型的整体鲁棒性。此外，将深度学习技术与传统机器学习算法结合的研究也显示出优异的性能，为提高恶意域名检测效率和准确率提供了一种重要的思路。随着深度学习技术在恶意域名检测中的广泛应用和验证，其在未来网络安全领域的作用预计将进一步增强。深度学习不仅能够提升恶意域名的准确率，还能够加速安全防御机制的响应速度，为网络安全的保护提供强有力的技术支持。因此，未来的研究将继续探索深度学习技术在恶意域名检测方面的应用，不断优化模型，以适应网络威胁的不断变化，保障网络环境的安全稳定。

在深度学习技术应用于恶意域名识别的研究领域中，虽已取得一定程度的成果，但诸多研究难题依然存在，需深入探讨与解决。本文提出的 CNN-BiGRU-AFF 可以更加全面的提取到域名中关键特征信息，在恶意域名检测中取得了不错的效果。但是该网络的检测精度依赖于数据集，对新出现和新变种的恶意域名检测效率不高。然后针对上述问题又提出了基于自适应深度变分自编码的恶意域名检测模型，实现了合法域名训练模型，缓解了恶意域名检测模型对新出现和新变种的恶意域名泛化性不强的问题。

(1) 为解决现有恶意域名检测模型在特征提取方面的不足，提出了一种基于 CNN-BiGRU-AFF 机制的检测模型。该模型通过增强对关键域名特征信息的处理和分析域名数据，显著提升恶意域名的准确率。首先使用注意力机制对域名数据进行筛选，有效去除冗余信息，随后利用 CNN 和 BiGRU 技术并行提取域名的局部与全局特征。这一步骤确保了从不同维度全面捕捉域名特征。紧接着，AFF 模块的引入使得两种特征得到有效融合，强化了模型对于关键特征的利用率，为分类决策提供了准确的信息基础。模型通过全连接层及 softmax 函数计算出域名的分类概率分布，并通过焦点损失函数进行优化，以减少误分类。为全面评估该模型性能，本研究构建了两种类型的数据集进行实验。实验结果表明了

CNN-BiGRU-AFF 模型在提高恶意域名特征提取能力方面具有显著优势，验证了本章模型的有效性。

(2) 在第三章提出的模型面临数据不平衡问题，导致其在新出现和新变种的恶意域名泛化能力上表现不足。针对上述问题，提出了一种基于自适应深度变分自编码器的恶意域名检测模型。通过变分自编码器较强的无监督学习能力，提高提取域名特征的效率。在实验之初，从域名统计特征的角度分析了合法域名与恶意域名存在巨大的差异。从理论证明了变分自编码器能够利用合法域名进行训练。然后为了提升更加全面的特征表示，进一步训练深度自编码器，采用动态设置阈值的方式设置分类阈值。为了全面评估所提模型的性能，本研究结合了多种数据集进行了实验。实验结果显示，虽然 Adv-DVAE 仅采用合法域名进行实验，但该模型对新出现或新变种的恶意域名具有较高的检测精度。实验表明，通过引入自适应机制和深度变分自编码的恶意域检测方法，可以有效地缓解恶意域名检测模型对新出现或新变种的恶意域名泛化性差的问题。

本文构建的模型在数据集上展现出了优异的分类性能，然而，仍存在一些局限性。未来的工作将围绕以下几个关键方面进行进一步的完善和深入研究。

(1) 本文提出的模型只能进行二分类，很难识别恶意域名具体类别。需要进一步的研究，对恶意域名进行多家族分类。

(2) 随着恶意域名生成技术的不断演化，其生成策略和变体也在持续更新，对新兴变种域名的发展动态进行持续的监测与深入分析。根据新出现的恶意域名变体对模型进行相应的改进。

(3) 将恶意域名检测模型嵌入实际的网络安全系统或应用中，进行实时的恶意域名识别和响应。此外，考虑与其他网络安全组件如入侵检测系统、端点保护平台等进行集成，形成多层次、多方位的网络安全防御机制。同时，开展广泛的实际应用测试，评估模型在真实环境中的性能和效果，从而有助于进一步调优和改进模型。

## 参考文献

- [1] Lu G, Jia X, Zhang Y, et al. A domain name management system based on account-based consortium blockchain[J]. Peer-to-Peer Networking and Applications, 2023, 16(2): 1211-1226.
- [2] Singh M, Kaur S. Issues and challenges in DNS based botnet detection: A survey[J]. Computers & Security, 2019, 86: 28 -52.
- [3] Wagan A, Li Q, Zaland Z, et al. A unified learning approach for malicious domain name detection[J]. Axioms, 2023, 12(5): 458.
- [4] Mvula P K, Branco P, Jourdan G V, et al. COVID-19 malicious domain names classification[J]. Expert Systems with Applications, 2022, 204: 117553.
- [5] Srinivasan S, Vinayakumar R, Arunachalam A, et al. DURLD: Malicious URL detection using deep learning-based character level representations[J]. Malware analysis using artificial intelligence and deep learning, 2021: 535-554.
- [6] Wang Z, Ren X, Li S, et al. A malicious URL detection model based on convolutional neural network[J]. Security and Communication Networks, 2021, 2021: 1-12.
- [7] Selvi J, Rodríguez R J, Soria-Olivas E. Toward optimal LSTM neural networks for detecting algorithmically generated domain names[J]. IEEE Access, 2021, 9: 126446-126456.
- [8] 杨成, 芦天亮, 闫尚义等. 基于 N-gram 和 Transformer 的 DGA 恶意域名检测[J]. 中国人民公安大学学报(自然科学版), 2022, 28(03): 100-108.
- [9] Kaloudi N, Li J. The ai-based cyber threat landscape: A survey[J]. ACM Computing Surveys, 2020, 53(1): 1-34.
- [10] 王媛媛, 吴春江, 刘启和, et al. 恶意域名检测研究与应用综述[J]. 计算机应用与软件, 2019, 36(09): 310-316.
- [11] Saiyod S, Chanthakoummane Y, Benjamas N, et al. Improving intrusion detection on snort rules for botnet detection[J]. Software Networking, 2018, 2018(1): 191-212.
- [12] 赵宏, 常兆斌, 王乐. 基于词法特征的恶意域名快速检测算法[J]. 计算机应用, 2019, 39(01): 227-231.

- [13] Chen S, Lang B, Chen Y, et al. Detection of Algorithmically Generated Malicious Domain Names with Feature Fusion of Meaningful Word Segmentation and N-Gram Sequences[J]. Applied Sciences, 2023, 13(7): 4406.
- [14] Zhao H, Chang Z, Wang W, et al. Malicious Domain Names Detection Algorithm Based on Lexical Analysis and Feature Quantification [J]. IEEE Access, 2019, 7(9): 128990-128999.
- [15] Abad S, Gholamy H, Aslani M. Classification of malicious URLs using machine learning[J]. Sensors, 2023, 23(18): 7760.
- [16] Abu Al-Haija Q, Al-Fayoumi M. An intelligent identification and classification system for malicious uniform resource locators (URLs)[J]. Neural Computing and Applications, 2023, 35(23): 16995-17011.
- [17] Almashhadani A O, Kaiiali M, Carlin D, et al. MaldomDetector: A System for Detecting Algorithmically Generated Domain Names with Machine Learning [J]. Computers & Security, 2020, 93: 101787-101793.
- [18] Aljabri M, Altamimi H S, Albelali S A, et al. Detecting malicious URLs using machine learning techniques: review and research directions[J]. IEEE Access, 2022, 10: 121395-121417.
- [19] Cucchiarelli A, Morbidoni C, Spalazzi L, et al. Algorithmically generated malicious domain names detection based on n-grams features[J]. Expert Systems with Applications, 2021, 170: 114551-114567.
- [20] Alshdadi A A, Alghamdi A S, Daud A, et al. Blog backlinks malicious domain name detection via supervised learning[J]. International Journal on Semantic Web and Information Systems (IJSWIS), 2021, 17(3): 1-17.
- [21] Cheng Y, Chai T, Zhang Z, et al. Detecting malicious domain names with abnormal whois records using feature-based rules[J]. The Computer Journal, 2022, 65(9): 2262-2275.
- [22] 马栋林, 张澍寰, 赵宏. 改进 Relief-C5.0 的恶意域名检测算法[J]. 计算机工程与应用, 2022, 58(11): 100-106.
- [23] 王伟, 罗鹏宇. 基于机器学习建模的 DGA 恶意域名检测[J]. 通信技术, 2022, 55 (06): 753-761.
- [24] Palaniappan G, Sangeetha S, Rajendran B, et al. Malicious domain detection using machine learning on domain name features, host-based features and web-based features[J]. Procedia Computer Science, 2020, 171: 654-661.

- [25] Hoang X D, Vu X H. An improved model for detecting DGA botnets using random forest algorithm[J]. Information Security Journal: A Global Perspective, 2022, 31(4): 441-450.
- [26] Patgiri R, Biswas A, Nayak S. deepBF: Malicious URL detection using learned bloom filter and evolutionary deep learning[J]. Computer Communications, 2023, 200: 30-41.
- [27] Umer M, Sadiq S, Karamti H, et al. Deep learning-based intrusion detection methods in cyber-physical systems: Challenges and future trends[J]. Electronics, 2022, 11(20): 3326.
- [28] Ahmed A A, Jabbar W A, Sadiq A S, et al. Deep learning-based classification model for botnet attack detection[J]. Journal of Ambient Intelligence and Humanized Computing, 2022, 13(7): 3457-3466.
- [29] Johnson C, Khadka B, Basnet R B, et al. Towards Detecting and Classifying Malicious URLs Using Deep Learning[J]. J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl, 2020, 11(4): 31-48.
- [30] Iliyasu A S, Deng H. N-GAN: a novel anomaly-based network intrusion detection with generative adversarial networks[J]. International Journal of Information Technology, 2022, 14(7): 3365-3375.
- [31] Highnam K, Puzio D, Luo S, et al. Real-time detection of dictionary dga network traffic using deep learning[J]. SN Computer Science, 2021, 2(2): 110.
- [32] Qiao Y, Zhang B, Zhang W, et al. DGA domain name classification method based on long short-term memory with attention mechanism[J]. Applied sciences, 2019, 9(20): 4205.
- [33] Yan L, Yang Y, Li Y, et al. Classification of Malicious DGA Domain Name Families Based on BiGRU and Attention Mechanisms[C]//2023 IEEE 11th Joint International Information Technology and Artificial Intelligence Conference (ITAIC). IEEE, 2023, 11: 1147-1150.
- [34] Kumar A D, Thodupunoori H, Vinayakumar R, et al. Enhanced domain generating algorithm detection based on deep neural networks[J]. Deep learning applications for cyber security, 2019: 151-173.
- [35] Huang X, Li H, Liu J, et al. A Malicious Domain Detection Model Based on Improved Deep Learning[J]. Computational Intelligence and Neuroscience, 2022, 2022: 26-42.
- [36] Yadav R. Light-weighted CNN for text classification[J]. arXiv preprint

- arXiv: 2004. 07922, 2020.
- [37] Namgung J, Son S, Moon Y S. Efficient deep learning models for DGA domain detection[J]. Security and Communication Networks, 2021, 2021: 1-15
- [38] 张清, 张文川, 冉兴程. 基于 CNN-BiLSTM 和注意力机制的恶意域名检测[J]. 中国电子科学研究院学报, 2022, 17(09): 848-855.
- [39] 王志强, 李舒豪, 池亚平等. 基于深度学习的恶意 DGA 域名检测[J]. 计算机工程与设计, 2021, 42(3): 601-606.
- [40] 邱颖豫, 许强. 基于群卷积神经网络的恶意域名检验方法[J]. 太赫兹科学与电子信息学报, 2022, 20(11): 1190-1197.
- [41] Luo X, Li Y, Cheng H, et al. AGCN-Domain: Detecting Malicious Domains with Graph Convolutional Network and Attention Mechanism[J]. Mathematics, 2024, 12(5): 640.
- [42] 周康, 万良, 丁红卫. 基于 AN 和 LSTM 的恶意域名检测[J]. 计算机工程与应用, 2020, 56(04): 92-98.
- [43] 赵宏, 常兆斌, 王伟杰. 基于深度自编码和决策树的恶意域名检测[J]. 微电子学与计算机, 2020, 37(05): 13-17.
- [44] Wang Z, Guo Y. Neural networks based domain name generation[J]. Journal of Information Security and Applications, 2021, 61: 102948.
- [45] Zhang K, Huang B, Wu Y, et al. A WGAN-Based Method for Generating Malicious Domain Training Data[C]//International Conference on Artificial Intelligence and Security. Cham: Springer International Publishing, 2022: 257-270.
- [46] 杨永红. 从域名领土看网络空间主权的边界[J]. 学术界, 2023, (10): 37-59.
- [47] 戈晶晶. 邢志杰: 抓住机遇打造下一代 DNS[J]. 中国信息界, 2022, (05): 80-83.
- [48] Aslam N, Khan I U, Mirza S, et al. Interpretable machine learning models for malicious domains detection using explainable artificial intelligence [J]. Sustainability, 2022, 14(12): 7375.
- [49] Manasrah A M, Khmour T, Freehat R. DGA-based botnets detection using DNS traffic mining[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(5): 2045-2061.
- [50] 吴晨. 一种基于 Word2Vec 的 M-TextRank 文本摘要模型[J]. 信息技术与信息化, 2023, (05): 125-128.
- [51] Alzubaidi L, Zhang J, Humaidi A J, et al. Review of deep learning:

- concepts, CNN architectures, challenges, applications, future. Journal of directions[J]. big Data, 2021, 8: 1-74.
- [52] Barak B, Edelman B, Goel S, et al. Hidden progress in deep learning: Sgd learns parities near the computational limit[J]. Advances in Neural Information Processing Systems, 2022, 35: 21750-21764.
- [53] Zhai Z L, Zhang X, Fang F, et al. Text classification of Chinese news based on multi-scale CNN and LSTM hybrid model[J]. Multimedia Tools and Applications, 2023, 82(14): 20975-20988.
- [54] Yu Y, Si X, Hu C, et al. A review of recurrent neural networks: LSTM cells and network architectures[J]. Neural computation, 2019, 31(7): 1235-1270.
- [55] Mahjoub S, Chrifi-Alaoui L, Marhic B, et al. Predicting energy consumption using LSTM, multi-layer GRU and drop-GRU neural networks[J]. Sensors, 2022, 22(11): 4062.
- [56] Zulqarnain M, Ghazali R, Hassim Y M M, et al. A comparative review on deep learning models for text classification[J]. Indones. J. Electr. Eng. Comput. Sci, 2020, 19(1): 325-335.
- [57] Gambhir M, Gupta V. Deep learning-based extractive text summarization with word-level attention mechanism[J]. Multimedia Tools and Applications, 2022, 81(15): 20829-20852.
- [58] Knop S, Tabor J, Podolak I, et al. Cramer-wold auto-encoder[J]. Journal of Machine Learning Research, 2020, 21(164): 1-28.
- [59] Sewak M, Sahay S K, Rathore H. An overview of deep learning architecture of deep neural networks and Auto-encoders[J]. Journal of Computational and Theoretical Nanoscience, 2020, 17(1): 182-188.
- [60] Michelucci U. An introduction to Auto-encoders[J]. arXiv preprint arXiv: 2201. 03898, 2022.
- [61] Li P, Pei Y, Li J. A comprehensive survey on design and application of Auto-encoder in deep learning[J]. Applied Soft Computing, 2023, 138: 110176.
- [62] Vahdat A, Kautz J. NVAE: A deep hierarchical variational Auto encoder[J]. Advances in neural information processing systems, 2020, 33: 19667-19679.
- [63] Valero-Carreras D, Alcaraz J, Landete M. Comparing two SVM models through different metrics based on the confusion matrix[J]. Computers &

Operations Research, 2023, 152: 106131.

- [64] Qin Y, Shi Y, Hao X, et al. Microblog text emotion classification algorithm based on TCN-BiGRU and dual attention[J]. Information, 2023, 14(2): 90.
- [65] Hajaj C, Hason N, Dvir A. Less is more: Robust and novel features for malicious domain detection[J]. Electronics, 2022, 11(6): 969.