密级：　　保密期限：

北京郵電大學

硕士学位论文

题目：基于生成对抗网络的入侵检测技术研究

学　　号：2020110955

姓　　名：刘宇轩

专　　业：网络空间安全

导　　师：李丽香

学　　院：网络空间安全学院

2023 年 6 月 3 日

Secret Level: **Confidentiality Period:**

BEIJING UNIVERSITY OF POSTS AND TELECOMMUNICATIONS

Thesis for Master Degree

Title: **Research on intrusion detection technology based on generative adversarial network**

Student ID: 2020110955

Candidate: Liu Yuxuan

Subject: Cyberspace Security

Supervisor: Li Lixiang

Institute: School of Cyberspace Security

June 3, 2023

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：_____ 日期：_____

关于论文使用授权的说明

本人完全了解并同意北京邮电大学有关保留、使用学位论文的规定，即：北京邮电大学拥有以下关于学位论文的无偿使用权，具体包括：学校有权保留并向国家有关部门或机构送交学位论文，有权允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，有权允许采用影印、缩印或其它复制手段保存、汇编学位论文，将学位论文的全部或部分内容编入有关数据库进行检索。（保密的学位论文在解密后遵守此规定）

本人签名：_____ 日期：_____

导师签名：_____ 日期：_____

基于生成对抗网络的入侵检测技术研究

摘 要

关键词：

RESEARCH ON INTRUSION DETECTION TECHNOLOGY BASE ON GENERATIVE ADVERSARIAL NETWORK

ABSTRACT

KEY WORDS:

目录

第一章 绪论

1.1 研究背景和意义

随着科学技术的告诉发展，互联网早已进入了以从旧时王谢堂前燕飞入寻常百姓家。随着互联网技术日新月异，我们的生活方式也发生了极大的改观。现如今网络技术已经浸入了我们的日常生活中，我们的所作所为也与网络密不可分。但是，不论是对于我们普通的互联网用户还是国家的保密机关，网络空间绝非是臻善臻美的乌托邦，而是充斥着阴谋与奸诈的虚拟世界。习近平总书记强调没有网络安全就没有国家安全，并将网络安全提升到国家安全的高度。正如总书记所说，我国的网络安全面临日益严峻的挑战。

1.1.1 互联网的飞速发展

我国互联网技术近年来迅猛发展。根据中国互联网络信息中心的统计，截至 2022 年 6 月，我国网民的总数为 10.51 亿，同时网络普及率为 74.4%。此外，我国的手机网民规模为 10.47 亿，相较于 2021 年 12 月增长 1785 万，网民使用手机上网的比例为 99.6%，与 2021 年 12 月基本持平。在工业互联网领域，工业互联网已经在 45 个国民经济大类中得到应用，产业规模已达到万亿元[1]。

互联网在产业丰富度上超越其他的产业。传统的基础应用类应用，如即使通信、搜索引擎早已融入了人们的生活。在新的赛道上，网络直播等网络娱乐型应用和在线医疗等公共服务类应用早已成为新的风口，也极大的改变了人类的生活。新兴的元宇宙概念成为了流量的宠儿，正在一次又一次的刷新人们的视线。

总之，近年来互联网的发展使人的生活方式发生了很大的改观，我们不自觉的将个人生活和互联网的虚拟空间融合在一起，将个人的意志投射到网络世界中。互联网已不再是一种普通的工具，而是人们生活的一部分。

1.1.2 严峻网络安全形式的挑战

正如总书记所强调的那样，现今的网络安全形势不容乐观。根据国家互联网应急中心的数据在 2021 年 7 月发布的《2020 年中国互联网网络安全报告》[2]：2020 年，我国境内受到恶意进攻的 IP 地址约为 5,541 万个，约占我国 IP 地址总数的 14.2%；同年，我国境内感染计算机恶意程序的主机约 533.82 万台，位于境外的约 5.2 万台计算机恶意程序控制服务器控制了我国境内约 531 万台主机；通过自主捕获和厂商交换新增获得移动互联网恶意程

序数量约为 302.8 万个。根据瑞星公司在 2022 年 1 月发布的《2021 年中国网络安全报告》[3]，2021 年瑞星安全系统截获的病毒数量为 1.19 亿个，病毒感染次数 2.59 亿次，其中截获手机病毒样本 275.6 万个；在全球范围内共截获恶意网址（URL）总量 6,279 万个，其中挂马类网站 4,366 万个，钓鱼类网站 1,913 万个。

随着信息技术发展，网络攻击手段也在突飞猛进。传统的被动防御手段如防火墙、虚拟网、用户身份加密技术已逐渐无法抵御现如今层出不穷的攻击手段。在这种境遇下，更加可靠的防御方法就应运而生了。入侵检测是主动防御（Proactive Defense） 技术的一种，其目的在于作为传统被动防御手段的补充，帮助网络快速发现攻击。 入侵检测相较于之前的防火墙等被动防御方式，可以更快更有效地发现攻击行为。

入侵检测常用的数据是网络流量数据。首先需要通过系统相应的功能模板收集网络流量并对网络流量数据包进行处理获取结构化的数据信息。然后通过分析网络流量数据的特征来判断网络流量是正常流量还是攻击流量。并根据分析结果进行后续的处理步骤。另一种常用的数据是日志，通过日志数据进行入侵检测。日志数据包含丰富的信息，如时间戳和日志键。且日志存在隐含逻辑，类似于自然语言，系统日志是由遵循一组严格的逻辑和控制流的程序产生的。 但正因为日志蕴含了丰富的信息，针对日志数据的处理成为众多研究的重点。

1.2 国内外研究现状

本节将从以下三个方面介绍国内外研究现状，主要包括入侵检测、生成对抗网络和生成对抗网络用于异常检测三个方面的相关研究。

1.2.1 入侵检测

入侵检测系统（Intrusion Detection Systems， IDS）是一种安全工具，该工具用于加强信息和通信系统的安全。对入侵检测的研究最早来自 James，他于 1980 年最早提出了入侵检测的概念，该研究立足于构建一个威胁监控系统。 1987 年， Denning 和 Neumann 提出了有史以来第一个入侵检测系统模型 IDES（入侵检测专家系统），首次完整地阐释了入侵检测系统的结构。随着入侵复杂性的提高，入侵数量的增多，涉及范围扩大，传统的入侵检测系统模型逐渐难以适应不断恶化的环境。因此，国际互联网工程任务组成立 IDWG 专门小组进行入侵检测系统模型的设计。在经过一系列研究后 Denning 等人提出了一种通用入侵检测模型，称为 CIDF（通用入侵检测

系统框架， Common Intrusion Detection Framework）。该系统分为事件产生器、事件分析器、响应单元、事件数据库几个部分。CIDF 将入侵检测系统分析的数据统一称为事件，事件可以是系统日志也可以是网络中的数据包。CIDF 模型具有很强的拓展性，目前获得业界广泛的赞同[4]。通用的入侵检测系统如图 1 所示。CIDF 在 2000 年被集成入 IETF，并将缩写调整为 IDWG（入侵检测工作组，Intrusion Detection Working Group）。在考虑的四个功能模块后，定义了一个通用的 IDS 架构[5]。一个通用的入侵检测系统的示意图见图 1-1。

针对入侵检测的算法可以简单分为基于数据挖掘和机器学习的入侵检测算法和基于深度学习的入侵检测算法。数据挖掘和机器学习算法自上世纪其被广泛应用于各个领域。将经典数据挖掘和机器学习算法应用于入侵检测的例子有：关联规则[14]、支持向量机（SVM）[8]、k 近邻（KNN）[9]、随机森林（RF）[10]、人工神经网络（ANN）[11]等算法。深度学习作为机器学习的一个分支，在近年来被证明在各个邻域有非常好的效果。有学者将该深度学习的算法应用于入侵检测。A. Javaid 等人提出了一种基于深度神经网络的算法用于基于流的入侵检测，实验结果表明该算法可用于软件定义网络的入侵检测[12]。T. A. Tang 等人基于 NSL-KDD 数据集提出了一个基于深度学习的自学习方法并证明了该方法的有效性[13]。

从信息源角度，IDS 可以分为基于主机的 IDS 和基于网络的 IDS。基于主机的 IDS 会对进程和系统调用等操作系统事件进行分析。而基于网络的入侵检测分析网络相关事件如：流量、IP 地址、服务端口、协议使用情况等信息[6]。

从检测方法的角度，可以将入侵检测分为基于误用检测（或称为基于特征）的入侵检测和基于异常检测的入侵检测。误用检测需要预先指定已知攻击的攻击库，并根据攻击库分析信息中定义的特征和模式。误用检测常见的算法有专家系统和模式匹配。这种方式对已知攻击有较高的检出效率。异常检测方法的思想在于保护系统的“正常”行为，在观测值和异常值之间的差距达到阈值后产生异常报警。异常检测的优势在于可以检测到之前不可知的入侵行为[5][6]。本文中所研究的入侵检测即基于异常检测的入侵检测方法。

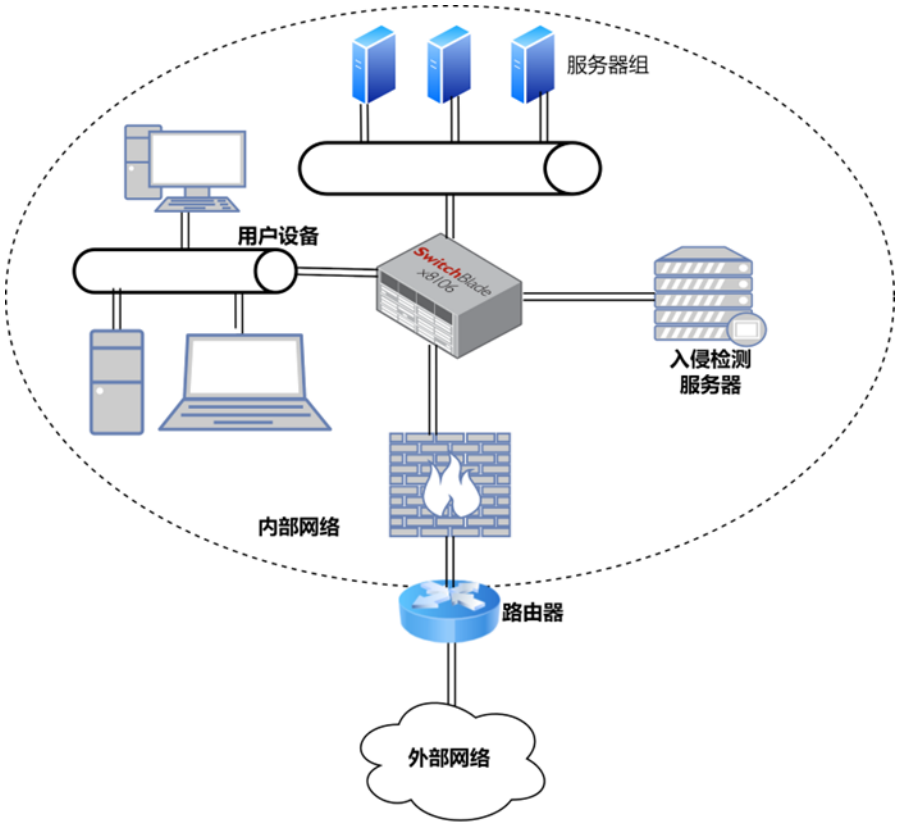

图 1-1 入侵检测系统示例图

1.2.2 异常检测

异常检测（Anomaly detection）指识别和多数数据显著不同的数据的任务[19]。异常检测可应用在很大场合，例如：工业产品缺陷检测、基础设施故障、医疗等邻域。异常（Anomaly），离群值（outlier）和奇异值（novelty）等术语描述内容相似，在本文中不作特别的区分，关于这三个术语的区分见[15]。对于异常概念，可以定义为不符合特征空间样本正常分布的样本。

大多数异常检测使用标签作为样本是正常还是异常的依据。根据标签的可用性，可用将异常检测分为三种[15][16]：

- 监督异常检测

以监督学习的方式对正常行为和异常行为进行建模。在训练时需要将数据标记为正常和异常两类，在训练完成后通过训练的模型进行异常的识别。

- 半监督异常检测

这种技术假设训练数据已经标记了正常类的实例，而不需要标记异常类。这类技术比监督技术被广泛使用。例如[20]中提出的算法，利用半监督算法对在线社交网络进行异常检测。

- 无监督异常检测

在无监督异常检测中，数据集通常只标记一个标签。训练模型自己从数据集中检测异常类别。其的工作原理是聚类机制，根据样本的特征找到相似组的样本集群。但是，这种假设有先天的缺陷，因为许多异常样本也会形成具有相似模式的集群。无监督学习不能有效地识别异常，并且经常存在较高的假阳性率[21]。

异常检测的算法可用大致分为三类：基于统计的算法，基于机器学习的算法，基于深度学习的算法。

- 基于统计的算法

基于统计的算法基于统计指标如均值和标准差，数据的分布和构建行为轮廓的概率，并用统计检验来判断当前样本是否正常[22]。基于统计的算法下面又可以分为有参数和无参数两种。有参数假设正常数据由参数和数据实例的异常度得分构成。参数化技术还可以分为基于回归模型、基于高斯模型和混合模型。无参数方式使用普通数据实例生成模型，不假设模型为推断模型。异常得分由给定数据与模型的偏差来确定[16]。

- 基于机器学习的算法

机器学习算法的优势在于可以根据经验提高从正常行为中分辨出异常行为的能力。基于机器学习的算法可以分为分类，近邻和聚类。分类是将数据根据特征分到设定的某几个类中，例如神经网络[23]、贝叶斯网络[24]。近邻算法利用距离或密度的函数衡量样本和最近邻之间的距离，异常分数就是通过距离来计算。例如将最近邻算法用于交通数据[25]。基于聚类的算法使用无监督学习算法在训练数据中实现同类的聚集，并以此来识别样本点。异常可能解决稀疏的簇或是无法被分为任何簇。聚类可用于手机信息的异常检测[26]。

- 基于深度学习的算法

深度学习在检测异常的时候有比之前的算法更好的性能[15]。现如今已有很大研究将深度学习算法应用于异常检测，例如 Lee 等人利用局部内在维数(LID)表征对抗样本的维数，实现异常检测[17]、Perera 等人[18]设计了一种深度单类分类器，实现了学习多标记数据的特征，用于异常检测。本文主要讨论的将生成对抗网络应用于异常检测见 1.2.4 一节。

1.2.3 生成对抗网络

生成对抗网络（Generative Adversarial Network，简称 GAN）最早由 Goodfellow 等人提出，利用生成器模型和判别器模型实现半监督学习模式[27]。 GAN 模型与传统的神经网络模型。

不同之处在于：在模型训练中引入了对抗因素，生成器和判别器彼此的

优化方向不同互成竞争关系。相较于传统的生成算法， GAN 模型可以生成更高质量的样本。

在 GAN 模型提出后，一些改进模型相继被提出。 f-散度是衡量两个分布间相似度的一种方法， f-GAN 通过任意给定的函数作为 f-散度拓展了 GAN 模型的目标函数[3]。Arjovsky 等人提出的 Wasserstein GAN（WGAN）算法提出了用于衡量 $p_{data}(x)$和$p_\theta(x)$ 之间距离有效的方法[28]。 在该算法中通过生成函数g_θ将输入的z转化为p_θ的方式进行学习，而不是直接学习概率分布$p_{data}(x)$自身。 对 WGAN 改进的观点指出在训练 WGAN 的时候对关键部分的权重裁剪会使判别器出现病态行为。 因此，建议用一个梯度标准作为惩罚项来代替权重裁剪[29]。LS-GAN 和 WGAN 一样使用了 Lipschitz 约束，但是二者用法不同[30]。LS-GAN 通过学习损失函数L_θ而不是关键因子使得一个真实样本的损失会比生成样本小。另一个相关改进是 RWGAN，通过结合 Bregman 散度和 Wasserstein 距离获得一个宽松的 Wasserstein 距离[31]。

在对 GAN 模型结构的改进方面， 在 DCGAN 算法中， 通过将卷积神经网络（CNN）引入 GAN，使得 GAN 的训练更为稳定[32]。在 DCGAN 算法中， 提出了一种架构指导方案，其中生成器是用一个转置的 CNN 模型，判别器是用输出维数为 1 的 CNN 模型。 目前提出的各种体系结构采用了多个生成器和判别器。例如， 在 MAD-GAN 算法中， 通过使用多个生成器学习真实数据分布的模式来解决模式崩溃问题[33]。 设计 StackedGAN 模型的目的是通过堆叠多个生成器判别器对来学习分层表示[34]。 在 GoGAN 算法中，使用多个 WGAN 对来改进 WGAN[35]。 尤其是 LR-GAN 组合了一个判别器和两个生成器[36]。在 LR-GAN 算法中，以递归的方式生成图像，用背景生成器生成图像的背景，前景生成器生成一个对象，将该对象贴在生成的背景上[36]。

时间序列是按时间顺序索引的一系列数据点。最常见的情况是在连续的等间隔时间点上记录的序列。众多学者在时间序列异常检测方面进行了大量研究。在基于统计的论文中，常见的是这几种模型，即自回归综合移动平均（ARIMA）、累积和统计（CUSUM）、指数加权移动平均（EWMA）等。随着深度学习理论的高速发展，在过去的几年里，大量基于深度学习的无监督异常检测方法被提出。许多研究使用神经网络学习时间序列数据中的隐含关系，据此建立预测模型，通过预测值在每个时间点与实际值的偏差来检测异常。例如，基于 LSTM 预测模型的异常检测方法以正常时间序列数据为模型，通过比较预测值与真值的残差来识别异常。Malhotra 等人使用在非异常数据上训练的堆叠 LSTM 网络作为多个时间步长的预测器，在时间序列

中进行异常检测。其他预测模型包括多层感知器（MLP）预测器和支持向量回归[37, 38]。

日志是非结构化数据，且无统一的输出格式[39, 40]。对日志信息进行处理首先需要对日志进解析。一种较为有效的方法是根据源码获取日志模板。但是通常无法获取源码且该方法缺乏泛用性。常用的日志解析的方法主要有基于启发式的方法，聚类和最长公共子序列等方法。基于聚类思想的日志解析是根据日志表现出的特征，将日志划分到不同的簇中，簇内部数据相似度高，簇间的相似度低。然后根据不同的簇提取日志模板。常见的方式有 SLCT（Simple Logfile Clustering Tool）方法。这种方法会根据日志中出现的高频率词进行聚类，然后再根据高频率词提取日志模板[41]。LogMine 是基于层次聚类的日志解析方法。使用正则表达式检测指定的类型并将类型替换为指定标识符，再进行聚类[42]。另一种常见的方法是基于最长公共子序列的方法。例如，Spell。Spell 假设日志源码相同的情况下打印的日志将拥有最长的公共子序列。这种方法可以实现在线日志系统的日志解析[43]。还有基于启发式的日志解析方法，例如，AEL，使用基于相似度量的检测方法[44]。

1.2.4 生成对抗网络用于异常检测

使用 GAN 进行异常检测的最找尝试来自 Schlegl 等人提出的 AnoGAN 算法[27]。该算法在训练阶段，只使用正常样本来学习潜在空间的无监督流形分布。在推理过程中，定义了一个损失函数，通过多次反向传播迭代找到流形空间中最接近样本分布的向量(Z)。生成器使用前向传播输出重建图像，并将其与原始图像进行比较以识别异常区域。此外，判别器的输出可以作为异常值来判断样本是否异常。值得注意的是 AnoGAN 算法基于假设而不是理论。CIAFL[28]提供了对抗性特征学习和推理面临的任务的定义:产生一种数据表示，在消除噪声信号的同时保持数据的有意义变化。这种定义完全符合异常检测的要求。随着理论的进展，学者们根据对抗特征学习和推理的相关理论改进基于 GAN 的异常检测方法。例如 Zenati 等人[29]基于 BiGAN 提出了 Efficient-GAN，并建立了基于 GAN 的异常检测基线模型，该模型在 AnoGAN 的基础上增加了一个从图像空间到隐空间的编码器，大大提高了网络的推理速度。在 ALICE[31]将对抗学习和条件熵正式统一为循环一致性之后，ALAD[30]利用其理论，使用三个判别器进一步限制了数据在潜在空间中的分布，从而提高了异常检测的性能。在自注意力机制[32]作为一种新的理论被提出后，该研究在深度学习领域大放异彩。自注意力机制提高了模型训练的稳定性并降低了模式崩溃的几率。基于 GAN 的异常检测方法立即开始利用 GAN 的最新成果进行改进。例如，ADWGAN[33]和 f-AnoGAN[34]

通过 WGAN 学习训练数据可变性的平滑表示，提高了异常检测的能力。

随着对无监督学习的重视，研究人员意识到表示学习的能力直接影响异常检测的性能。此外，相关研究[35]表明，与自监督学习相关的数据增强任务可以提高 GAN 的表示学习能力。因此，相关的自监督学习被引入该领域。GEOM[36]是一项通过几何变换预测进行高效表示学习的开创性工作，提高了异常检测的性能。自监督方法侧重于判别器方向，其对生成器效果不明显。

在时间序列方面，因为时间序列的特殊性，因此用于检测图像和其他数据类型的算法不能直接应用于时间序列这种连续数据。时间序列类型的数据有很大针对的算法，例如 RNN，LSTM 或是 Transfomer。很多学者根据时间序列数据的特性，提出了很多有针对性的算法[19]。最为简洁的算法是 Bashar MA 等人提出的 TAnoGAN 算法[37]。该算法将异常检测的过程分为训练阶段和异常检测阶段。TAnoGAN 的训练阶段与 AnoGAN 模型相同。将真实序列除以滑动窗口作为真实样本输入，由生成器从隐空间采样潜变量产生假样本。生成器和鉴别器都由 LSTM 层组成，以处理时间序列。Li Y 等人提出了 DCT-GAN 算法[38]，该算法基于 TAnoGAN。算法使用卷积神经网络和 Transformer 构建生成器和判别器，利用卷积神经网络提取信息，用 Transformer 处理时间序列。相较于原生 GAN，DCT-GAN 使用多个生成器避免模式崩溃。Geiger A 等人提出的 TadGAN 算法[39]在生成器和鉴别器中也使用了 LSTM，该算法使用了多种计算重构误差的方式。Maru C 等人提出了 MARU-GAN[40]，该算法的生成器是一个 Seq2Seq 的解码器，判别器由 RNN 和全连接神经网络构成。解码器可以使算法可以处理更大窗口的数据，有利于识别出集体异常。

1.2.5 日志异常检测

日志异常检测可以分为基于监督学习的异常检测和基于无监督学习的异常检测[59]。监督学习通过标记好的数据进行训练，标记数据会表示出数据是正常还是异常。常见的方法有逻辑回归，支持向量机和决策树。与监督方式不同，无监督方式使用没有标签的数据[60]。因为在实际环境中数据大多都是无标签的，因此，无监督学习在生产环境中的适应性更好。无监督学习常见的方法有日志聚类、PCA、关联规则挖掘。

对于日志数据的建模角度可以将日志异常检测分为基于文本特征建模的日志异常检测、基于序列建模的日志异常检测。基于文本特征建模的日志异常检测着眼于日志文本中出现的特定关键字，如“kill”，“fail”等和语义相关的关键字。基于文本特征建模会通过统计的手段用机器学习方法计算出某条日志数据正常的概率。Farshchi 等人采用基于回归的分析技术来

查找操作活动日志与操作活动对资源影响的相关性[63]。陈明等人将决策树引入异常检测，使用决策树诊断网络中的异常[61]。李琦等人提出了LogCluster 聚类方法来识别系统中的异常，这个算法包括将日志向量化和聚类的初始化阶段。和进行训练构建知识库的阶段[62]。另一种是基于序列建模的日志异常检测，将日志视为序列数据。主要分为两个方向：基于统计模型的方向和基于循环神经网络模型。循环神经网络在自然语言处理和图片语音识别方面取得了很大的成果。针对序列数据的特性，学者们将长短时记忆网络（Long Short-Term Memory，简称 LSTM），自动编码器，GAN 用于异常检测。Min Du 等人提出的 DeepLog 日志异常检测方法，利用 LSTM 神经网络的生成能力，根据正常数据训练网络，输出当前时刻日志键的出现概率，并和当前时刻实际的日志键进行比对来判断结果是否正常[64]。段晓宇等人提出基于 LSTM 的编码-解码器框架 GAN-EDC。整个框架由基于 LSTM 的自动编码器的生成器和基于 CNN 的判别器构成。GAN-EDC 通过基于 LSTM 的自动编码器生成日志模板，再通过 CNN 将生成的日志模板和实际的日志模板进行对比，判断是否异常[39]。李丹等人针对多变量序列异常检测问题提出基于 GAN 的时间序列异常检测方法 MAD-GAN[37]。MAD-GAN 具有处理多元时间序列的能力。MadGAN 还在 GAN 的生成器和鉴别器中使用 LSTM 来处理序列数据。在异常检测阶段 MadGAN 使用判别损失和重构损失来判断样本是否是异常。

1.3 论文主要研究内容

本论文拟借助自动编码器、生成对抗网络， 对网络入侵检测进行数学抽象和建模。获得网络流量和日志，对不同匹配规则的优劣进行数值分析。设计所使用的神经网络算法的优化方法，网络结构用控制变量法进行分析。通过在数据集上比较正确率获得更有效的算法。通过设计实验对本课题提出算法的效果进行验证，并与相同条件下其它的入侵检测算法进行对比。本课题采取由基础理论向具体问题逐步推进的论述思路，计划按照以下的技术路线来完成研究，达到预期的研究效果。

1.3.1 基于互补生成对抗网络的入侵检测算法

在实际的网络环境中，绝大多数的网络流量都是正常流量，只有少数流量是异常流量。实际可用的有标签的数据集数量较少。因此使用已知的数据进行训练有所不足。因此本文计划使用无监督学习的方式，在训练阶段使用正常数据进行训练。

首先本文使用互补对抗生成对抗网络生成离群点的方式生成样本。生成的样

本位于正常样本的互补空间且尽可能接近样本分布。生成的异常样本可以作为异常样本辅助训练。本文使用的生成对抗网络架构，该架构使用生成器生成数据，在用判别器判别生成器生成的样本和真实样本，通过判别交替迭代生成器和判别器。本文的模型使用生成器生成异常样本，在用判别器判定样本是来自于训练集中的正常样本还是生成的异常样本。判别器的判别结果可以反过来优化生成器的生成模型，让生成器可以生成更多高质量的样本。

在判别器方面本文提出了一个堆叠非对称自动编码器的判别器算法。自动编码器是一种解压缩模型，该模型通过编码器将输入压缩到较小维度的隐藏状态再通过解码器将数据解压缩会原始维度。正常数据和异常数据经过自动编码器压缩解压缩后重构的结果不同。自动编码器模型可以基于深度学习的思想，将多个自动编码器堆叠起来，构成堆叠自动编码器来获得性能更优的模型。本文就是依据这个原理将自动编码器作为判别器。本文使用的非对称自动编码器相较于对称自动编码器省略了一半的参数，提高了运算的性能，更适用于现实中流量检测对实时性的高要求。因为直接使用自动编码器性能较差，因此本文使用支持向量机对非对称自动编码器的输出结果进行处理，输出正常还是异常的结果。

1.3.2 基于 LSTM/GRU 的生成对抗网络日志异常检测算法

在现代大型分布式网络系统中日志是查找定位问题的重要信息来源。但是日志数据结构复杂且语义晦涩，且大部分的日志系统都会产生海量的数据，此外日志数据还具有时序特性，在进行异常检测的时候需要针对该特性涉及算法。针对日志的异常检测算法面对上述挑战。本文提出了一种基于日志数据进行异常检测算法。

日志数据是非结构化数据，无法直接用于异常检测。因此需要先对日志数据进行解析。日志解析的主要内容将文本形式的日志数据解析成日志事件序列。日志事件可以从日志模板角度进行建模，即将单条日志序列解释为一个日志模板。本文提出了一种基于前缀树聚类的日志解析算法。该算法首先需要对日志数据进行预处理，删除日志中的一些停用字符。然后从构建日志前缀树，再根据前缀树进行聚类。因为日志前缀树中部分节点可能不是日志词，因此需要将部分子树进行合并。此外，日志前缀树不适合过大，因为从日志数据中提取出过多日志模板会对算法性能参数不利影响。因此该算法会根据日志中词语出现的频率对日志前缀树进行剪枝。在整个日志解析流程处理完后我们就获得了日志词典或日志模板列表。

然后本文提出了一种基于 LSTM/GRU 和生成对抗网络的日志异常检测算法。在生成器方面，本文使用了自动编码器架构。生成器使用自动编码器在重构方式上的能力。首先，将日志序列数据从解码器输入获得隐藏状态。然

后，将隐藏状态和随机生成的噪声输入解码器中，获得重构之后的日志序列。生成器的评价指标是数据构成的残差损失。判别器也使用 LSMT/GRU 模型，将日志数据输入后获得判别结构。用生成对抗网络的一般训练方式对生成器和判别器进行训练。在异常检测阶段综合生成器的重构分数和判别器的判别分数为异常分数对数据进行异常检测，如果异常分数超出阈值，则判定为异常，反之亦然。

1.4 论文组织结构安排

本文共有五个章节，各个章节的研究内容如下：

第一章为论文绪论部分。首先介绍了现今网络发展的形式和网络安全面对的考验，由此引出入侵检测的必要性；其次从整体上介绍了入侵检测和异常检测，生成对抗网络和生成对抗网络应用于入侵检测，日志解析等领域的研究进展；最后介绍本论文的研究内容和组织结构。

第二章为论文基础知识部分。首先介绍了生成对抗网络的基本概念和基本结构；其次是自动编码器的基础知识；然后介绍了日志解析方法和针对时间序列的网络模型；最后介绍了入侵检测和日志异常检测常用的数据集并给出的评价异常检测的常用指标。

第三章研究基于互补生成对抗网络的流量入侵检测技术，基于非对称自动编码器的判别算法，并阐述基于互补对抗网络的流量入侵检测算法的实现。

第四章研究基于 LSTM/GRU 的生成对抗网络日志入侵检测技术，阐述适用于时间序列的前缀树日志解析算法，基于 LSTM/GRU 的自动编码生成器和基于 LSTM/GRU 的判别器，以及算法的整体架构。

第五章为阶段性总结与未来展望部分。首先，对目前已经完成的阶段性研究工作进行总结；随后，对下一步拟开展的研究方向和相关工作进行展望。

第二章 相关基础理论

本章通过对人工智能，神经网络的综合运用， 对生成对抗网络算法在入侵检测上的应用展开研究。在已有研究成果的基础上， 通过对已有算法的研究与创新，结合生成对抗网络模型和自动编码器，提出一种应用于入侵检测的一种新的入侵检测算法。 并为智能化网络空间安全贡献自己的力量。 本课题重点研究以下三个方面的内容。

2.1 生成对抗网络

生成对抗网络（Generative Adversarial Network, GAN），由 Goodfellow 等人在 2014 年首次提出[26]。该模型是深度生成模型的一种，其首要功能是通过神经网络拟合分布能力，在不进行假设的情况下模拟特定的分布，这种模型称为隐式密度模型。

GAN 包含两个模型：生成模型（Generative Mode）和判别模型（Discriminative Model）。生成模型用来生成“神似”真实样本的伪造样本来骗过判别器，而判别器的任务是识别给定的样本是真实的还是伪造的。因为 GAN 拥有生成原本不存在样本的特性，所以该模型常用于原始样本数量较少的场景中。本课题所研究的针对未知威胁的入侵检测所需判别的样本是先前还不存在的样本种类，因此，GAN 模型非常适合用来训练。在训练过程中，只输入正常样本进行训练，通过生成器生成的伪造样本与正常样本一同输入判别器中进行训练。在测试过程中将正常样本和异常样本对判别器进行测试，获得判别器实际的分类效果。

原始的 GAN 的训练优化目标为：

$$\min_{G}\max_{D}V(D,G)=E_{x\sim p_{data}(x)}[\log D(x)]+E_{z\sim p_z(z)}[\log(1-D(G(z)))]$$

对判别器而言优化函数是：

$$arg\max_{\phi}\left(E_{x\sim p(x)}[\log D(x,\phi)]+E_{z\sim p(z)}[\log(1-D(G(z,\theta),\phi))]\right)$$

用于训练判别器 D 使得最大概率地判断对训练样本的类别（最大化$\log D(x)$和$\log(1-D(G(z)))$）

对生成器而言优化函数为：

$$arg\min_{\theta}\left(E_{z\sim p(z)}[\log(1-D(G(z,\theta),\phi))]\right)$$

训练网络G最小化$\log(1-D(G(z)))$，即最大化D的损失。在训练过程中固定一方，更新另一个网络的参数，交替迭代，使对方错误最大化，最终获得良好的

模型。理论上 GAN 经过足够次数的迭代训练，最终使生成器学到了样本数据的分布规律，模拟出数据的分布。在这种情况下判别器将无法识别出样本是真实样本还是由生成器生成的样本，也就是不论样本是真实还是生成，判别器的输入概率为 0.5，达到纳什均衡。

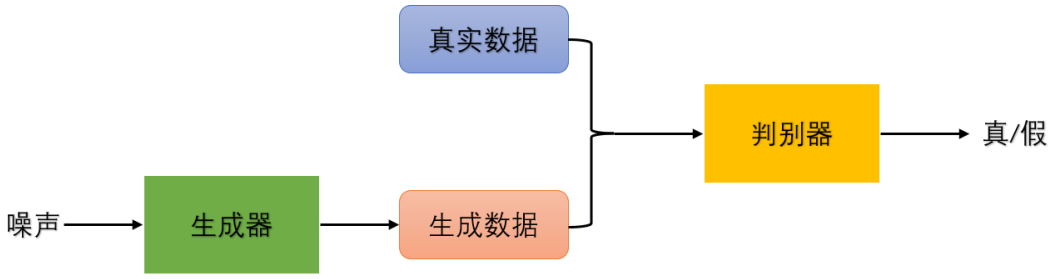

图 2-1 生成对抗网络的网络结构

2.2 深度自编码器

目前在深度学习研究中使用的一种比较流行的技术是自动编码器，我们提出的算法就使用了这种技术。自动编码器是一种基于无监督神经网络的特征提取算法，它通过使重构输出尽可能接近实际输出来学习最佳参数。自动编码器的目标是能够提供比主成分分析（PCA）更强大和非线性的泛化能力[65]。

模型的训练方式是通过应用反向传播并将目标值设置为等于输入。换句话说，它试图学习恒等函数的近似值。自动编码器通常有一个输入层、输出层（与输入层具有相同的尺寸）和一个隐藏层。这个隐藏层通常比输入层的维数小（称为欠完整或稀疏自动编码器）。自动编码器的示例如图 2 所示。

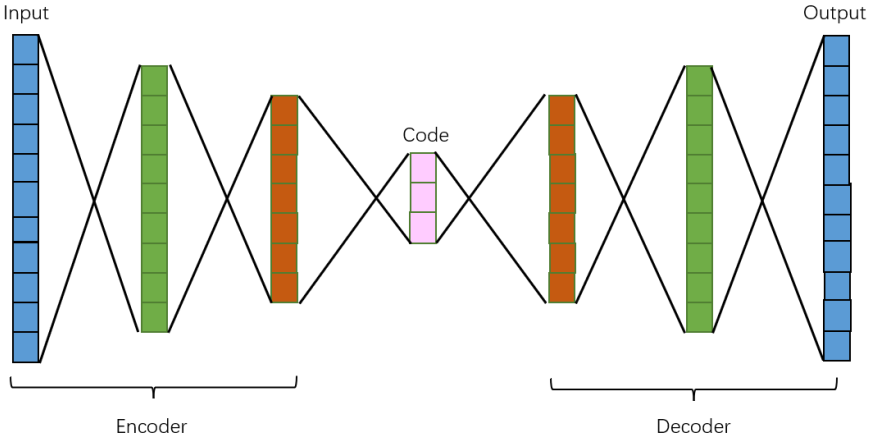

图 2-2 自动编码器的网络结构

大多数研究人员使用自动编码器实现非线性转换，通过在网络上施加其他约束来发现有趣的数据结构，并将结果与 PCA（线性转换）的结果进行比较。这些方法基于编码器-解码器范式。首先将输入转换为典型的低维空间(编码器)，然

后拓展重构初始数据(解码器)。一旦一层被训练，它的代码就会被提供给下一层，以便更好地在输入中建模高度非线性的依赖性。该范式侧重于降低输入数据的维数。为了实现这一点，在深层自动编码器结构的中心有一个特殊的层——编码层。此编码层用于在堆叠自动编码器中作为压缩特征向量来进行分类和组合[65]。

隐藏层用于创建高维数据的低维版本（称为编码）。通过降低维数，自动编码器被迫捕获数据分布的最显著特征。在理想的情况下，自动编码器生成的数据特征将比原始数据本身提供更好的数据点表示。

自动编码器的目的是尝试和学习如下的函数：

$$h_{W,b}(x) \approx x$$

通过调整h的参数W（权重）和b（偏移），将计算结果尽可能接近输入数据x。

简而言之，尝试通过学习找到一个函数恒等式，其中x'尽可能接近于x。学习过程被描述为重构误差最小化函数，具体为。

$$L(x,d(f(x)))$$

其中，L是一个损失函数，作为x和$d(f(x))$间距离的惩罚项，d是解码函数，f是一个编码函数。

2.3 基于时间序列的日志解析算法

机器数据通常由系统或程序产生。程序开发者为监控程序执行过程，会让程序在执行完一步或几步操作后输出相应的执行结果，该结果通常会记录系统或程序在什么时间、哪个主机、哪个程序上执行了什么操作及出现了什么问题等信息，这些信息被称为机器数据，即日志（Log）。

从内容角度日志是某些操作及其结果的集合，它们按照时间排序。可以将日志视为常量和变量部分组成的纯文本，由不同代码打印出来的日志会存在区别。日志中可能包含了日志来源地址、日志产生的时间戳、请求方式、状态码、消息长度、系统描述等信息。因为日志是非结构化数据，无特定格式的日志结构化表示，对日志数据处理首先需要对日志数据进行非结构化。日志数据存在一些特点，首先是日志中出现词的频率分布不均匀，大部分单词都是不频繁的，同时只有一小部分单词会频繁出现，频繁出现的词通常包含了比较重要的信息。这部分小频率出现的单词有很强关联性，通常是由固定的字符串组成。比如“print log invalid xxxx, err=***”。其中“print”和“err”都是比较重要的词且存在关联性，用于打印程序运行中出现的错误。该语句在程序运行的流程中会反复出现，使固定字符串成为频繁词集合。并且这些单词存在一定的逻辑关系和语义关系，拥有领域知识的

专家可以解读出日志的含义。同时和频繁词一同出现但是不断变化的部分可以称为变量部分，包括时间戳和日志位置。其中时间戳反映了日志的时序信息。

针对日志的特点，找到合适的日志解析方法。解析日志首先需要提取日志模型模板，即对抽取日志数据中相对不变的部分，抽取的部分称为日志模板或者称为日志事件。日志模板中不变的部分称为日志键，可变的部分称为参数。例如，日志项“Took 10 seconds to build instance.”的日志键是“Took * seconds to build instance”，参数在日志模板中被表示为*。除了日志键，还需要提取每个日志的时间戳。在入侵检测发现中，时间戳是一个较为重要的信息，例如，拒绝服务攻击会操作程序执行的耗时提升，反映在时间戳上就是间隔较大上。

在提取出日志模板之后，需要对日志特征进行建模，用来提取日志中有价值的信息，这些信息可以用来进行入侵检测。这一步需要将上面提取出的日志模型进行向量化表示。为了提取需要特征需要适用滑动窗口，将日志序列划分为有限序列，并将划分出来的有限序列作为输入传入下一阶段。

综合以上，本论文在基于时间序列的日志解析方面的研究重点在于：（1）日志模板的抽取，将非结构化的日志数据结构化为日志模板；（2）日志特征建模，将日志模板中的日志键，参数和时间戳向量化。

2.4 针对时间序列的算法

传统的机器学习算法 ML 无法有效的分析出时间序列数据的内在逻辑 [64]，因此循环神经网络(recurrent neural network，简称 RNN)被研究者提出来用来处理时间序列问题。和传统的神经网络算法不同，RNN 在隐藏层中可以具有基于其在与上下文信息重复过程的短期记忆。但是，由于梯度消失和爆炸的问题，RNN 无法提供长期记忆[65]。

早先的生成模型算法通过调整高斯分布的参数拟合未知的分布 。通过将深度学习引入生成算法中，可以通过多层神经网络的建模能力获得一个复杂分布。生成模型可用于生成样本，通过给定概率密度函数为$p(z,\theta)$的分布，生成一些服从这个分布的样本。具体的过程分为两步：

（1） 根据隐变量的先验分布$p(z,\theta)$进行采样，得到样本z，通常假设先验分布符合正态分布。

（2） 根据条件分布$p(x|z,\theta)$进行采样，得到x。

但是在时间序列数据中生成的样本不仅和当前时刻的输入有关，还和上一时刻的输出相关。一种合理的思路是用一个特殊的延时单元记录上一时刻的信息并对当前的输入产生影响，有了延时单元，神经网络就具有了记忆能力。在实际中，通常会将延时单元放在输入和输出层间作为一层隐藏层。大多数研究人员处理长

序列的方法是把它们分成小序列。他们使用一个称为“滑动窗口”的窗口从序列的开始滑动到结束，被窗口拦截的部分被用作训练或测试样本。“滑动窗口”的大小通常称为“窗口长度”，“滑动窗口”的步长(一次移动多少帧)通常称为“窗口步长”。然后，当“滑动窗口”移动时，我们可以得到一组小的时间序列，用于训练或测试。异常标签由对应标签中的时间戳标签确定。

针对时间序列的一种常用的生成模型是长短期记忆(long short-term memory, LSTM)网络。LSTM 主要由输入门、输出门和遗忘门组成。LSTM 最大的优点是解决了 RNN 无法处理的长距离依赖问题。LSTM 通过门机制维护单元状态，可以同时解决短时和长时记忆依赖问题，从而避免梯度消失和爆炸问题[66]。

图 1 描述了一个存储单元中具有三个门的基本 LSTM 单元。输入门的功能是跟踪存储单元中的最新信息;输出门的功能是控制最新信息在网络其余部分的传播。第三个门(忘记门)功能是根据前一个单元格的状态确定是否删除信息。下面的公式解释了如何实现和更新 LSTM 单元状态并计算 LSTM 输出：

$$I_t = \sigma(W_{xi}X_t + W_{hi}H_{t-1} + B_i)$$

$$F_t = \sigma(W_{xf} + W_{hf}H_{t-1} + B_f)$$

$$\overline{C}_t = \sigma(W_{xc}X_t + W_{hc}H_{t-1} + B_c)$$

$$C_t = F_t * C_{t-1} + I_t * \overline{C}_t$$

$$O_t = \sigma(W_{xo}X_t + W_{ho}H_{t-1} + B_o)$$

$$H_t = o_t \tanh(C_t)$$

$$Y_t = \sigma(W_{hy}H_t + B_y)$$

$$\sigma(x) = \frac{1}{1+\exp^{-x}}$$

其中X_t=输入向量；Y_t=输出向量；I_t=输入们的输出；F_t =遗忘门的输出；O_t=是输出门的输出C_t=记忆块的结束状态；$\overline{C}_t$=中间值；σ是 sigmoid 函数；

W_{xf}, W_{xi}, W_{xc}，并且W_{x0}是输入权重向量；W_{hf}, W_{hi}, W_{hc}和W_{h0}是循环权重矩阵；W_{hy}是输出权重矩阵；B_f，B_i，B_c，B_0和B_y是相对的偏移向量[68]。LSTM 的详细内容见[67]。

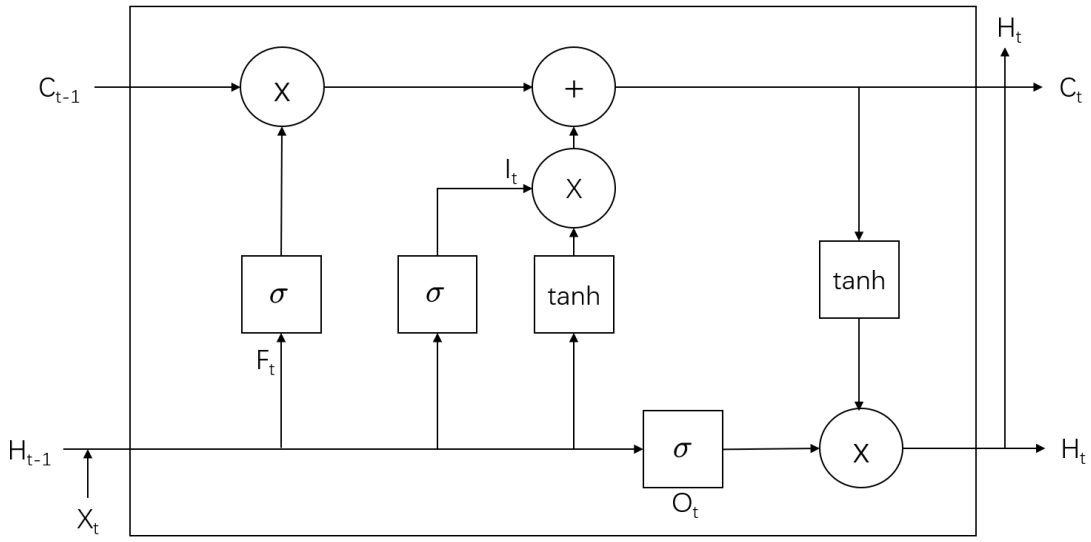

图 2-3 LSTM 的网络单元结构

在 LSTM 的基础上门控单元神经网络（GRU）省去了一个门，只留下重置门（reset gate）和更新门（update gate）。其中重置门用于控制对前一时序信息的忽略程度，更新门用于控制前一时序的信息传入的程度。GRU 还设置两个状态，隐藏态h和候选态$\tilde{h}$，隐藏态h的取决于候选态$\tilde{h}$和输入向量x[68]。下面的公式解释了如何具体计算h：

$$r_j = \sigma\left((W_r x)_j + (U_r h_{t-1})_j\right)$$
$$z_j = \sigma\left((W_z x)_j + (U_z h_{t-1})_j\right)$$
$$h_j^t = z_j h_j^{t-1} + (1 - z_j)\tilde{h}_j^t$$
$$\tilde{h}_j^t = \varnothing\left((Wx)_j + (U(r \odot h_{t-1}))j\right)$$

其中r_j表示重置门向量的第j个元素的计算结果，σ表示 logistic sigmoid 函数，x和h_{t-1}表示输入和上一时刻的隐藏状态，W和U表示学习到的权重矩阵。同理z_j表示更新门的计算结果。h_j^t表示 t 时刻的隐藏态，$\tilde{h}_j^t$表示 t 时刻的候选态。当重置门接近 0 时，隐藏态被强制忽略前一个隐藏态，仅使用当前重置状态。更新门控制从前一个隐藏状态转移到当前隐藏状态的信息数量，类似于 LSTM 中的记忆单元。

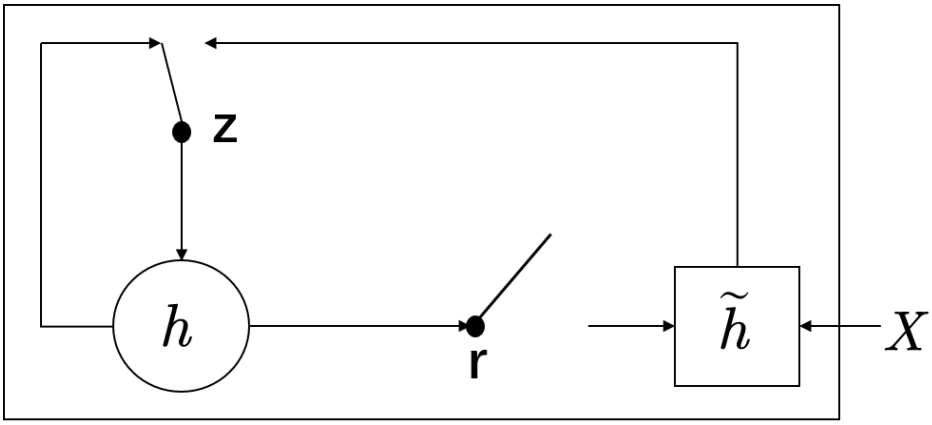

图 2-4 GRU 网络单元结构

综上所述本论文的研究重点在于构建合适的生成式神经网，生成日志数据。

2.5 入侵检测常用数据集

2.5.1 网络流量数据集

研究人员在找的合适的数据集来测试评估自己的工作成果方面付出了很多努力。构成一个合适的数据集的难度很大。因为在实际的网络环境中将某条数据流视为异常或是非异常需要有足够的知识才能判断，而这就需要耗费非常多的人力进行数据标注。而满足这种条件的数据集需要：(i)包含多种攻击；(ii)符合实际情况；（iii）公开可用[70]。目前常用的数据集有如下几个，更多的数据集见[70]。

（1） KDD99 数据集[71]

KDD99（KDD Cup 1999）数据集的开发是为了训练和验证与传统信息技术系统相关的入侵检测系统。该数据集基于 DARPA98 入侵检测系统评估数据集，由哥伦比亚大学的 Sal Stolfo 教授和来自北卡罗莱纳州立大学的 Wenke Lee 用数据挖掘的方法对 DARPA98 数据集进行了数据分析和预处理，创建了一个新的数据集，该数据集用于 1999 年举行的 KDD CUP 竞赛中，成为著名的 KDD99 数据集[71]。KDD99 数据集是评估入侵检测模型使用最广泛的数据集之一。

KDD99 数据集由 500 万条数据组成，还包括一个由 10%数据构成的训练子集和测试子集。KDD99 数据集将模拟的攻击分为 4 大类:(1) 未授权的本地超级用户特权访问(unauthorized access to local superuser privileges by a local unpivileged user，U2R)，(2) 未授权的本地超级用户特权访问(Unauthorized access from a remote machine to a local machine，R2L)，(3)端口扫描(Probing attack)和(4)拒绝服务攻击(denial-of-service，DoS)。每条数据有 42 项，包括 41 个特征和 1 个标识，41 个特征，可分为以下四类:(1)基本特征，(2)流量特征和(3)内容特征（4）

统计特征。特征是从 TCP/IP 连接中提取的。流量特征分为两组(即“相同主机”特征和“相同服务”特征)。内容特性涉及数据部分的可疑行为。

KDD99 数据集的特征具体为：

1） TCP 连接的基本特征：基本特征包含了一些连接的基本特征，如协议类型、传输数据大小等信息。

2） TCP 连接的内容信息：对于某些攻击无法从数据头中识别出现，需要结合负载中的信息进行判断，因此抽取了数据中部分信息用于检测攻击。如是否成功登录、尝试登录失败的次数、访问敏感文件的次数。

3） 基于时间的网络流量特征：部分网络攻击和时间有强关联性，考虑数据的时间特性可以发掘出连接与连接之间的关系。具体分为“主机连接”，只记录在前 2 秒与当前连接有相同目标主机的连接，例如相同连接数；“相似服务”，只记录前 2 秒内与当前连接有相同服务的连接，如具体的连接数量。

4） 基于主机的网络流量特征：当前连接之前 100 个连接记录中与当前连接具有相同目标主机的统计信息，用于针对基于时间的网络流量特征无法发现 2 秒外的数据关联问题。例如统计前 100 个连接中，与当前连接具有相同目标主机相同服务的连接数。

（2）CICDS2017 数据集[73]

该数据集来自通信安全机构(CSE)与加拿大网络安全研究所(CIC)合作项目，该项目对从 1998 年来提出的 11 个数据集进行评估，认为大多数数据集（比如经典的 KDD99，NSLKDD 等）已经过时且不可靠。针对之前数据集存在的问题，该研究模拟真实的网络环境构造了新的入侵检测数据集 CICDS2017 数据集。

CICIDS2017 数据集由 Sharafaldin 等人提出[72]。该数据集收集了包含从人工构造的被攻击网络中收集的从 2017 年 7 月 3 日(周一)到 2017 年 7 月 7 日(周五)共 5 天的数据,其中星期一是正常流量，在周二、周三、周四、周五进行网络攻击。该数据集实包含暴力 SSH、DoS、心血漏洞、Web 攻击、渗透、僵尸网络和 DDoS 以及暴力 FTP 等网络攻击。然后使用 CICFlowMeter 工具[131]从生成的网络流量中提取出 80 个网络流特征。此外，CICIDS2017 数据集基于 FTP、HTTPS 等协议提取了 25 个用户的抽象行为。

在评估阶段，研究人员从数据集中取 80 个特征，并使用随机森林回归算法确定用于检测每个攻击族的最佳短特征集。然后根据前人提出的评估框架[74]，用 11 个指标对数据集与 1998 年至 2016 年的其他公开数据集进行质量比较进行评估，最终证明了 CICDS2017 数据集要优于之前提出的数据集。

表 2-1 CICDS2017 数据集信息统计

Days	Labels
Monday	Benign
Tuesday	BForce,SFTP and SSH
Wednes.	DoS and Hearbleed Attacks Slowloris,Slowhttptest Hulk and GoldenEye
Thurs.	Web and Infiltration Attacks Web BFORCE, XSS and Sql Inject Infiltraion Dropbox Download And Cool disk
Friday	DDoS LOIT, Botnet ARES, PortScans (sS,sT,sF,sX,sN,sP,sV,sU, sO,sA,sR,sL, and B

（3）其他数据集

1）DARPA 1998 数据集

在 Sharafaldin 等人的工作中于 1998 年 2 月首次公布 75。该数据集基于网络流量和审计日志。训练数据包含 7 周的基于网络的攻击，而测试数据包含 2 周的基于网络的攻击。该数据集不代表真实世界的网络流量。

2） NSL-KDD 数据集

该数据集由 Tavallaee 等人[76]提出，旨在用于解决 KDD99 数据集的一些存在的问题。NSLKDD 数据集与原数据集相比有以下改进:(1)去除了冗余记录;(2)去除了重复记录;(3)选择的记录数按记录数的百分比组织;值得注意的是，许多关于入侵检测的论文在性能评估中同时使用了这两个数据集(即 KDD Cup 1999 数据集和 NSL-KDD 数据集)，它们发现通常 NSL-KDD 数据集上的结果会更好。

3） UNSW-NB15 dataset

该数据集由 4 个工具创建，分别是 IXIA PerfectStorm 工具、Tcpdump 工具、Argus 工具和 Bro-IDS 工具。这些工具可用于创建某些类型的攻击，包括拒绝服务攻击、漏洞利用、通用攻击、侦察、Shellcode 和蠕虫攻击。UNSW-NB15 数据集包含大约 200 万个向量和 54 万 044 个特征。此外，Moustafa 等人[77]发布了该数据集的分区，其中包含训练集(175,341 向量)和测试集(82,332 向量)。

4） DEFCON dataset

该数据集有两个版本，包括 DEFCON- 8(2000)和 DEFCON-10(2002)。DEFCON-8 数据集中的攻击包括端口扫描和缓冲区溢出。DEFCON-10 数据集中

的攻击包括探测性攻击和非探测性攻击(例如，坏包、端口扫描、端口扫描等)。Nehinbe Ojo Joshua[78]使用这两个版本对网络入侵进行重新分类。

5） CAIDA 数据集

该数据集由应用互联网数据分析中心提出，包含不同的数据集，包括 CAIDA DDOS、CAIDA Internet traces 2016 和 RSDoS 攻击元数据(2018-09)。具体来说，CAIDA 的 DDOS 攻击包括一个小时的 DDOS 攻击流量分割 5 分钟的 pcap 文件，这些文件是来自 CAIDA 的 Equinix-Chicago 的被动流量跟踪。rdos 攻击元数据(2018-09)包括从 UCSD 网络望远镜收集的反向散射数据包推断出的随机欺骗拒绝服务攻击[79]。

6） CDX 数据集

该数据集由应用互联网数据分析中心提出，包含不同的数据集，包括 CAIDA DDOS、CAIDA Internet traces 2016 和 RSDoS 攻击元数据(2018-09)。具体来说，CAIDA 的 DDOS 攻击包括一个小时的 DDOS 攻击流量分割 5 分钟的 pcap 文件，这些文件是来自 CAIDA 的 Equinix-Chicago 的被动流量跟踪。rdos 攻击元数据(2018-09)包括从 UCSD 网络望远镜收集的反向散射数据包推断出的随机欺骗拒绝服务攻击[80]。

7） CIC DoS 数据集

该数据集包含使用不同工具的 4 种类型的应用层 DoS 攻击。Jazi 等人[110]提出了 CIC 拒绝服务攻击数据集，其中应用层拒绝服务攻击通常出现在大容量(如使用 HULK (HTTP Load King)产生的高容量 HTTP 攻击)或低容量变化(如低容量 DoS 攻击)中。大量的 HTTP 攻击包括 Goldeneye (DoS improved GET)、ddossim (DDoS GET)和 hulk (DoS GET)。低容量 HTTP 攻击包括慢发送正文(Slowhttptest)、慢发送头(Slowhttptest)、和慢读(Slowhttptest)[81]。

8） CSE-CIC-IDS2018 数据集

该数据集由通信安全机构(CSE)和加拿大网络安全研究所(CIC)提出。数据集 CSE-CIC-IDS2018 包含 7 种不同的攻击场景，包括心血漏洞、暴力破解、DoS、DDoS、Web 攻击、僵尸网络和渗透。与 CICDS2017 数据集类似，使用 CICFlowMeter 工具从生成的网络流量中提取 80 个网络流特征[69]。

2.5.2 日志异常数据集

日志是由源代码中记录日志的相关语句 (例如 printf()、println())打印出来的非结构化文本。日志可能来自分布式系统、超级计算机、操作系统、移动系统、服务器应用程序和独立软件，不同来源的日志有各自代表的数据集[82]。

（1）HDFS 数据集

HDFS 来自于 Hadoop 系统的分布式文件系统（Hadoop Distributed File

System），公开使用的数据集是从亚马逊 EC2 集群收集到的日志数据，因此该数据集属于分布式系统日志。该系统使用了 203 个节点，记录了 38.7 个小时，共产生了 24,396,061 条数据，日志数据集大小约为 2000MB。该数据集的异常由 Hadoop 领域专家标注。该日志数据集的每个工作流都分配了唯一 ID，该 ID 对应了一个时序窗口，每个有唯一 ID 的时序窗口都包含一定时间内的日志序列，序列中包含了程序的各种行为，例如写入、复制、分配、删除。

（2）其他数据集

可用于日志异常检测的数据集比较少，绝大多数的数据集是不公开的。比较常用的有 BGL 数据集，来自超级计算机的公开的日志数据集，收集自加利福尼亚州利弗莫尔的劳伦斯利弗莫尔国家实验室的 BlueGene/L 超级计算机系统。该数据的标签信息可用于报警检测和预测研究[83]。还有 OpenStack 数据集，来自云操作系统 OpenStack。该数据集是在 CloudLab 上生成的，该系统是一个用于云计算研究的灵活、科学的基础设施。同时提供了正常日志和故障注入后的异常案例，使数据更适合于异常检测研究。

2.6 入侵检测的性能评价指标

在入侵检测进行分类的过程中对所有的样本进行分类的结果有 4 种情况：真阳性（True Positive, TP），真阴性（True Negative，TN），假阳性（False Positive，FP），假阴性（False Negative，FN），描述见表 2-2。

表 2-2 分类结果表

样本分类结果	分类结果描述
真阳性	异常样本被检测为异常样本
真阴性	正常样本被检测为异常样本
假阳性	正常样本被检测为异常样本
假阴性	异常样本被检测为正常样本

这 4 个分类结果应用到常用的评价指标中，在入侵检测领域常见的指标是准确率（precision），召回率（recall）和 F1 分数（F1-score）：

（1） 精确率指的是在识别为异常样本的所有样本为，实际为异常样本所占的比例，该数值越大越好。

$$precision = \frac{TP}{TP + FP}$$

（2） 召回率指的是在实际为异常样本的样本，被识别为异常样本所占的比例，该数值越大越好。

$$recall = \frac{TP}{TP + FN}$$

（3） F1 分数是将精确率和召回率相结合的评价指标，该值越大越好。

$$F1-score=2*\frac{precision*recall}{precision+recall}$$

（4） 混淆矩阵用来表示模型预测值与真实值之间的分布情况，本论文的算法是二分类，混淆矩阵是一个 2*2 的矩阵，如表 2-3 所示。

表 2-3 混淆矩阵

分类	预测为正常	预测为异常
实际为正常	TN	FP
实际为异常	FN	TP

2.7 本章小结

本章主要介绍了本论文所涉及的各项基础知识，包括生成对抗网络、深度自动编码器、日志解析、时间序列网络模型。总结在网络安全领域入侵检测和日志异常检测常用数据集和本论文计划使用的算法性能衡量指标。

第三章 基于互补对抗网络的入侵检测算法

第四章 本章提出了一种基于互补对抗生成网络和非对称堆叠自动编码器的网络模型，该网络可用于实现网络流量入侵检测的功能。在本章中，我们首先介绍互补生成算法的概念，然后介绍非对称堆叠自动编码器在网络模型中的应用，随后给出网络的整体架构和实现入侵检测的具体过程。然后对提出的算法性能从各个角度进行评估，并和其他算法进行对比。最后对本章进行小结。

3.1 引言

现在主流的入侵检测算法有基于规则的入侵检测算法，基于机器学习的入侵检测算法和深度学习入侵检测算法。早期的入侵检测系统使用基于规则的入侵检测算法，将入侵检测视为一个专家系统。专家根据自己的经验知识设定一系列的规则，例如，产生式规则，状态转移图。基于规则的入侵检测算法具有误报少，准确率高的优点。但是只能发现已知攻击，无法识别之前未出现的攻击，且规则库的构建成本较高。基于机器学习的入侵检测算法使用机器学习将机器学习的技术引入了入侵检测领域。比较常用的有支持向量机和决策树。支持向量机是一种线性分类器，将数据实例映射到特征空间，使得每类样本距离超平面的平均距离最大。在测试时根据样本落在超平面的哪一侧进行分类。决策树基于信息熵实现分类，决策树的每个非叶节点都代表了某个特征的分类，根据特征属性值的不同进入下个节点，直至根节点得到最终分类。其中选择的每个节点都选择信息熵最大的特征，最后的决策过程构成了树状结构。近年来随着计算机算力的提升，深度学习被广泛运用到各个领域，其中比较重要的一种就是生成对抗网络。因为生成对抗网络的生成样本能力，该算法被广泛运用于原始样本或带标签样本较少不便于训练的场合。在入侵检测中，因为绝大多是数据都是正常数据，涉及网络攻击的只是很少一部分数据，而且获取带标签的数据非常困难，实际所能收集到的网络攻击数量相对有限且由人工对网络流量打标签的成本很高，因此在现实中很难获取数量足够的有标签的数据。此外，现如今网络环境丰富，入侵检测的场景已不仅是常见的家用或是工业网络，在其他场景例如自动驾驶邻域也面对着网络攻击的威胁。在针对现如今复杂的场景，将生成对抗网络引入入侵检测系统得到了很多学者的论证。

近年来出现了很多使用生成对抗网络进行异常检测的算法。一种经典的思路是用模型拟合出正常样本的分布，然后计算待测样本和分布的偏差值，认为偏差大的样本为异常样本。该方法的困难在于最初模型的假设，模型的参数的估

计需要足够的经验。另一种场景的思路是近邻算法，该算法不需对数据集进行任何假设。算法计算的是样本点的稀缺度，例如计算离待测样本最近的k个样本（KNN）或是计算局部可达密度（LOF）[83,84,85]。另一种方式是人工生成异常点，然后训练分类器识别正常样本和异常样本间的边界。这种方式不需要精确的分布假设，且可以通过生成足够的异常样本点可以有效的提升判别的准确率。但是随着维数的增加，生成样本所包含的信息量逐渐降低[84]。一种改进方法是用正常数据来生成异常样本。由于数据结构越来越复杂，这种方法并不能保证始终具有良好的性能。因此本论文使用基于 GAN 的方法生成样本辅助训练。GAN 的方式生成样本本质是通过神经网络的拟合能力模拟具有某种分布的模型。GAN 的架构可以通过最小最大博弈过程获取真实数据的深度表示。前人针对 GAN 的异常检测算法通常将 GAN 当作特征提取器和重构器，但是本论文着重研究 GAN 在生成异常样本方面的能力。本论文使用互补对抗生成网络生成和正常样本接近的异常样本，来辅助判别器的训练。

在判别器方面，本论文将自动编码器引入架构中。在入侵检测场景已经有了不少自动编码器的应用，例如[64,10]。通常的生成对抗网络判别器使用常规的神经网络。但是本文注意到了自动编码器在入侵检测领域的应用，部分使用自动编码器的算法取得了较好的结果，因此本文将自动编码器算法应用到判别器上。自动编码器的原理是通过生成器和解码器对数据进行压缩和解压缩，其本质上是通过神经网络对输入数据进行重构。因此，使用正常数据训练自动编码器模型，将类似于网络攻击等异常数据输入自动编码器后，自动编码器重构的数据相较于原始数据会有较大的重构误差，基于此特性自动编码器实现了异常检测。在深度学习兴起后，学者们将深度学习的思想引入了自动编码器，将多个自动编码器嵌套在一起提升性能。相较于浅层自动编码器，深度自动编码器在误差上会更具稳定性。此外，为了提高自动编码器的运算速度，避免过多层数的神经网络造成计算开销过大，一个思路上去掉自动编码器的解码器部分，只是用编码器部分，这样相当于减少了一半的运算量。只使用自动编码器用于异常检测的性能可能会达不到预期，因此可以将传统的机器学习算法，类似随机森林或是支持向量机和自动编码器相结合，提升性能。

综上所述，为了实现更高效的入侵检测算法，我们结合的互补对抗生成网络和非对称堆叠自动编码器，用互补对抗生成器生成异常数据并用非对称堆叠自动编码器作为判别器识别正常或异常数据。具体而言，该算法使用了生成对抗网络结构，在训练使用正常数据进行训练，在测试阶段使用正常数据和异常数据进行测试，可实现异常数据的识别。算法使用互补对抗生成器生成位于正常样本附近的离群点，作为训练时的异常数据，实现异常数据的生成。使用非对称堆叠

自动编码器实现样本的判别，该算法相较于常规的自动编码器算法运算速度更快。在实际环境中进入入侵检测时，使用判别器检测网络攻击即可，这样保证了在真实世界网络数据流量较大时可用实现对流量进行实时的入侵检测，最大程度保证系统的安全。在数据预处理方面我们考虑使用自动编码器对数据进行预处理，降低数据本身的噪声，提出数据中隐藏的关键信息。

3.2 互补对抗生成网络

3.2.1 算法原理

随着对抗学习的发展，如何生成有效的对抗样本成为了一个热门的话题。最直接的方式就是基于模型的方式生成，即假定样本的分布符合某种模型分布，例如高斯混合模型，在通过最大似然估计的方式计算该模型的参数从而获得合理的生成模型[84]。生成对抗网络的思路是使用博弈论模型，用生成器生成拟真的样本，在用判别器进行判别，两个子模型彼此竞争，依次迭代，最好获得可以生成“逼真”样本的生成器。

首先从数学角度对异常检测进行描述。设定样本数据集为$X=[x_1,x_2,...,x_n]\in\mathbb{R}^{d\times n}$其标签$Y=[y_1,y_2,...,y_n]$未知，其中$x_i\in\mathbb{R}^d$代表数据点。标签$y_i=1$代表正常数据$y_i=0$表示异常数据。我们的目标是找到一个划分边界将异常数据从正常数据中划分出来。为了描述这个边界，设定一个评分函数$\xi(x)\in(0,1)$。优化边界过程可以等同于最小化损失函数$\mathcal{L}_\xi$，该函数试图将正常数据打分为 1，异常数据打分为 0。设c_0和c_1分别为异常值和正常值的误分类代价，损失函数$\mathcal{L}_\xi$可定义为：

$$\mathcal{L}_\zeta=-\frac{1}{n}\sum_{i=1}^{n}(c_n y_i\log(\xi(x_i))+c_0(1-y_i)\log(1-\xi(x_i)))$$

因此问题转化为找的一个最优的函数可以最小化$\mathcal{L}_\xi$。但是问题在于缺少y_i的先验信息。为了解决这一问题，考虑了异常值的固有属性，假设异常数据的密度低于正常数据，这与前人的许多研究是一致的。基于此，引入一个分布位于$\mathbb{R}^d$标准参考分布μ，并用相对密度$\rho(x)$来定义x的集中度。当相对密度$\rho(x)$小于合理分布阈值τ，这个样本将被判断为一个离群点，反之亦然。然而，ρ(x)的计算通常需要大量的计算资源。因此，我们尝试用分类来代替密度水平检测过程。我们从参考分布中生成 n 个数据点作为潜在的异常值，然后引入分类器$C(x)\in(0,1)$将他们从原始数据集X中区分出来。损失函数$\mathcal{L}_\xi$可以被定义为:

$$\mathcal{L}_C=-\frac{1}{2n}\sum_{i=1}^{2n}(y_i\log(C(x_i))+(1-y_i)\log(1-C(x_i)))$$

其中y_i被标注为 1 或是 0，当x_i分别是从X或是μ中提取出来。为了最小化$\mathcal{L}_C$，分类器应该对原始数据输出更高的相对密度$\rho(x)$（代表μ）值；反之亦然。因此，对于任何正数(例如，τ)，我们可以通过最小化$\mathcal{L}_C$使$C(x)$服从以下条件：

$$C(x|\rho(x) \geq \tau) \to 1$$
$$C(x|\rho(x) < \tau) \to 0$$

任取x，满足：

$$C(x|\rho(x) \geq \tau) \geq C(x|\rho(x) < \tau)$$

这表示我们最小化损失函数$\mathcal{L}_C$，优化分数函数$\zeta(x)$可以被$C(x)$取代，这样不依赖任何对正常数据的假设且消耗更少计算资源。在生成对抗网络中使用判别器D作为分类器，因此上述公式变为：

$$\mathcal{L}_D = -\frac{1}{2n}\sum_{i=1}^{2n}(y_i \log(D(x_i)) + (1-y_i)\log(1-D(x_i)))$$

将生成器G代入，G用于模拟上面提出的参考分布μ用来生成异常样本。即可得到生成对抗网络判别器的目标函数：

$$D^* = arg\max_D E_{x\sim p_{data}(x)} \log(D(x)) + E_{z\sim p(z)} \log(1-D(G(x)))$$

本论文使用的生成器基于互补对抗生成网络思想，通过生成器学习可以获得正常样本的互补分布，来辅助判别器的训练[87]。生成对抗网络基于博弈论的思想，通过生成器和判别器进行对抗训练，获取能够生成逼真样本的生成器。因为将生成器生成的样本设定为异常样本，因此生成的异常样本和正常样本需要有一定的区分度，以保证异常样本包含有足够的信息。因此提出了专门用来生成异常样本的互补 GAN 生成器。互补 GAN 的生成器G是一个前馈神经网络，其输出层与样本x的维数相同。形式上，定义生成的样本为$\tilde{x} = G(z)$。常规 GAN 可以学习到与正常样本分布p_{data}相似的样本分布，但是互补 GAN 的生成器G可以学习到与正常样本的互补分布p^*接近的分布p_G，这正是生成异常样本所追求的目标。可以将互补分布p^*定义为：

$$p^* = \begin{cases} \frac{1}{\varepsilon}\frac{1}{p_{data(\tilde{x})}} & if\ p_{data}(\tilde{x}) > \tau\ and\ \tilde{x} \in B_x \\ C & if\ p_{data}(\tilde{x}) \ll \tau\ and\ \tilde{x} \in B_x \end{cases}$$

其中τ是用于判断生成样本是否位于高密度区域的阈值；ε是正则化参数；C是较小的常量；B_x是样本空间，需要保证所有生成的$\tilde{x}$都位于样本空间中。

为使生成的样本符合p^*分布，也就是使生成分布p_G接近互补分布p^*。使用f散度表示两个分布之间的差距，f散度是用来计算两个分布间差距的通用函数，f散度表达式为：

$$D_f(P||Q)=\int q(x)f\left(\frac{p(x)}{q(x)}\right)dx$$

使用f散度描述p_G和p^*之间的差距为：

$$D_f(p_G||p^*)=\int p^*(\tilde{x})f\left(\frac{p_G(\tilde{x})}{p^*(\tilde{x})}\right)d\tilde{x}$$

其中f是表示$D_f(p_G||p^*)$的超参数，作为一个函数f满足 1）该函数是一个凹函数；2）$f(1)=0$。

另$f(x)=x\log x$即用KL散度计算两个分布之间的差距，则生成器G需要进行训练使p_G和p^*间的KL散度最小，表达式为：

$$\begin{aligned}\mathcal{L}_{KL(p_G||p^*)}&=-\mathcal{H}(p_G)-\mathbb{E}_{\tilde{x}\sim p_G}\log p^*(\tilde{x})\\&=-\mathcal{H}(p_G)+\mathbb{E}_{\tilde{x}\sim p_G}\log p_{data}(\tilde{x})I[p_{data}(\tilde{x})>\tau]\\&+\mathbb{E}_{\tilde{x}\sim p_G}(I[p_{data}(\tilde{x})>\tau]\log\varepsilon-I[p_{data}(\tilde{x})\leqslant\tau]\log C)\end{aligned}$$

其中，$\mathcal{H}(\cdot)$是熵，且$I[\cdot]$是指示函数。上式最后一部分可省略，因为ε和C为常数，在求导的时候为 0。

同时互补生成器 G 采用特征匹配损失，确保生成样本被限制在用户表示空间B_x中，表达式为：

$$\mathcal{L}_{fm}=\|\mathbb{E}_{\tilde{x}\sim p_G}f(\tilde{x})-\mathbb{E}_{x\sim p_{data}}f(x)\|^2$$

其中$f(\cdot)$表示作为x特征表示的判别器的中间层输出。

因此，定义生成器完整的目标函数为：

$$\begin{aligned}\min_G\ &-\mathcal{H}(p_G)+\mathbb{E}_{\tilde{x}\sim p_G}\log p_{data}(\tilde{x})I[p_{data}(\tilde{x})>\tau]\\&+\|\mathbb{E}_{\tilde{x}\sim p_G}f(\tilde{x})-\mathbb{E}_{x\sim p_{data}}f(x)\|^2\end{aligned}$$

综上所述，互补生成器的目标函数旨在让生成分布p_G接近互补样本p^*，即$p_G=p^*$，并使得生成样本来自于与正常样本不同的区域。

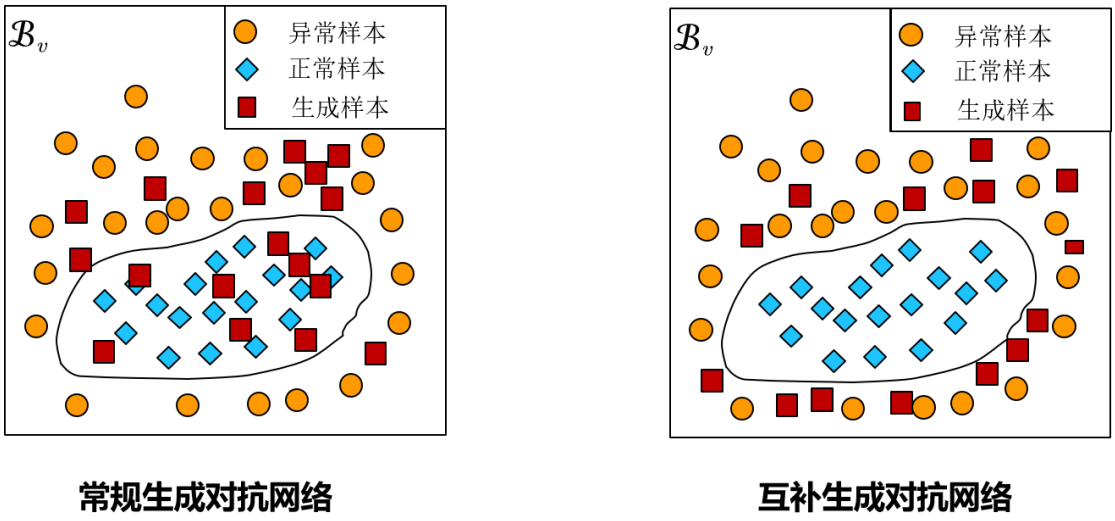

图 3-1 常规 GAN 和互补 GAN 生成数据的对比图

在生成的早期阶段，生成器可能无法围绕正常样本生成足够的异常样本。但是经过足够次数的迭代，生成器 G 逐渐学会了生成机制，并生成了越来越多发生在真实数据内部或接近真实数据的潜在异常值。这是一个主动学习的过程。

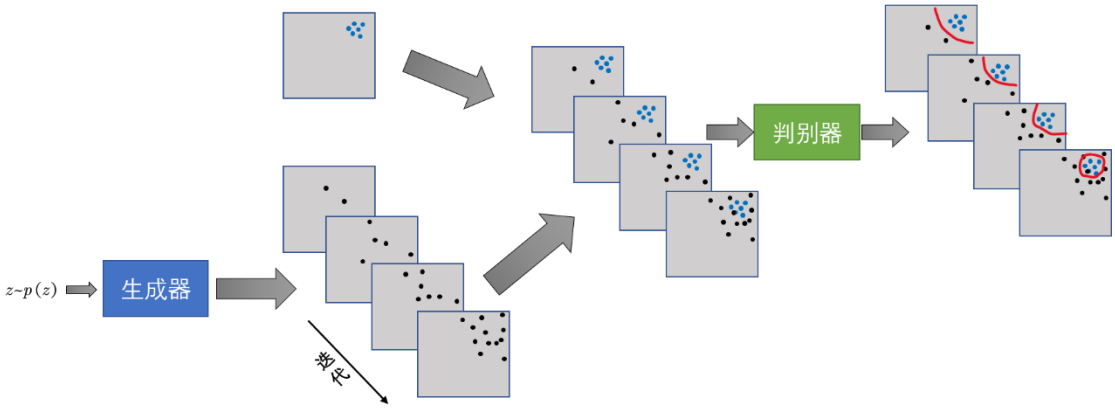

图 3-2 互补生成器的训练过程

$\mathcal{H}$项不好进行直接计算，因此，采用一个 pull-away 项（PT项）来代替熵[88]。该项用来衡量数据的正交程度，增加了样本的多样性，优化该项等同于优化熵。将PT项定义为：

$$L_{PT}=\frac{1}{N(N-1)}\sum_{i}^{N}\sum_{j\neq i}^{N}\left(\frac{f(\tilde{v}_i)^T f(\tilde{v}_j)}{\|f(\tilde{v}_i)\|\|f(\tilde{v}_j)\|}\right)^2$$

其中N表示批量训练时批量的大小。将生成器的整体优化函数更新为：

$$\min_G \; L_{PT}+E_{\tilde{v}\sim p_G}\log p_{data}(\tilde{v})\mathbb{I}[p_{data}(\tilde{v})>\epsilon]$$
$$+\left\|E_{\tilde{v}\sim p_G}f(\tilde{v})-E_{v\sim p_{data}}f(v)\right\|^2$$

在训练完成后可以直接使用判别器对样本进行分类，最后的输出层和样本的维数相同，实现了样本的生成。

常规 GAN 和互补 GAN 的架构相似，但是互补 GAN 有更强的异常检测能力。常规 GAN 的生成器用于生成和正常样本相同的分布数据。因此，在生成器训练到一定程度后生成的异常样本与正常样本位于相同的区域(如图 a)。常规 GAN 的判别器判断样本是正常和异常的概率都接近 0.5，使常规 GAN 的判别器无法正常识别出正常或是异常数据。但是使用互补 GAN 网络的生成器可以生成正常样本的互补样本。由于生成的样本与异常样本具有相同的分布(如图 2b)，因此互补 GAN 的判别器可以异常样本。

3.2.2 生成器网络结构

本论文使用生成器使用的网络架构如图所示，图中的网络结构为 255×128×64×128×样本维度，最后输出的向量维度和数据集样本的维度相同。在层和层之间使用 ReLU 作为激活函数，在输出层使用 tanh 作为激活函数。传统的使用 Simgoid 作为激活函数并使用随机初始化权重向量会造成生成的样本点集中到样本空间中心，也就是真实数据所在的区域，这和生成异常数据辅助训练的目标不一致[84]。因此本文使用 ReLU 作为激活函数并使用随机正交初始化权重，提高生成高质量异常点的概率。

生成器的网络架构可以进行改变，图中的网络架构针对的是 KDD99 数据集。在对一些更复杂数据集或是现实中的网络流量时，可根据流量特征灵活调整网络架构。因为绝大多数的流量数据的维数不高，因此使用全连接网络即可反映出数据的特性，并在判别器的引导下生成异常样本辅助训练。对于学习率和迭代次数没有已知的好的估算方法，只能通过实验逐步找的最优的参数。

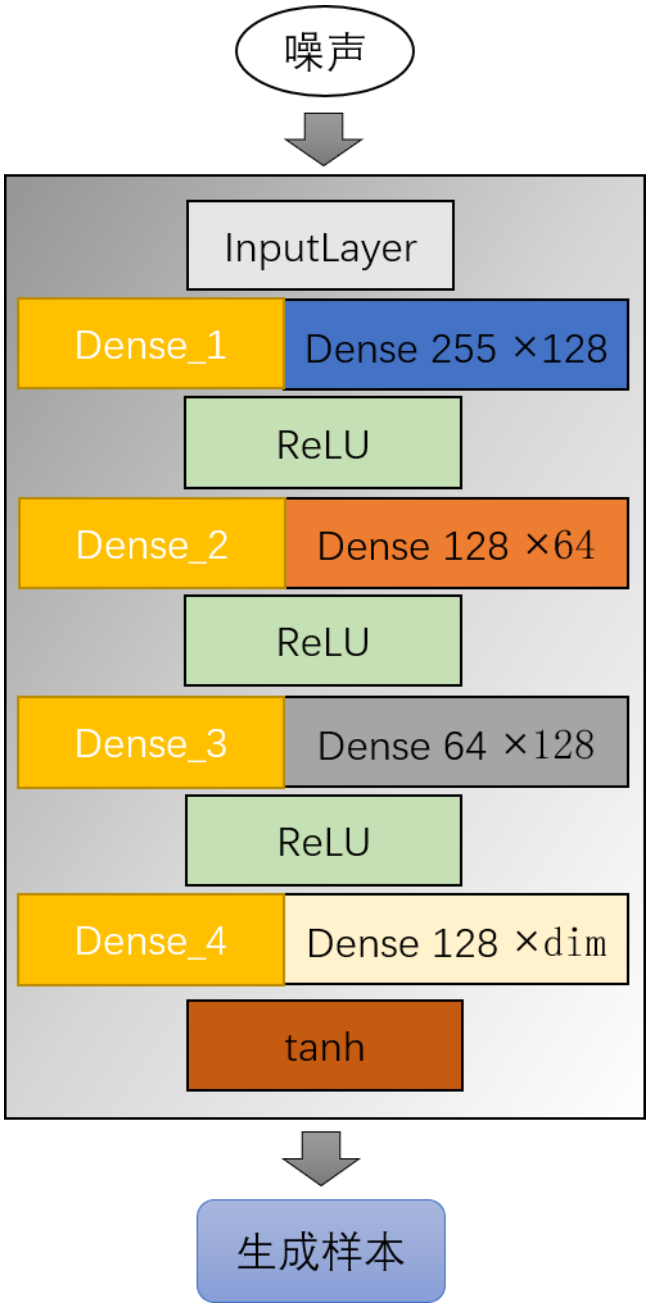

图 3-3 生成器网络结构

3.3 堆叠非对称自动编码器

3.3.1 堆叠非对称自动编码器的原理

本论文中，生成对抗网络的判别器使用了堆叠非对称自动编码器结构，该结构可以在不影响正确率的情况下减少计算量。

自动编码器是一种自监督学习模型，常用于数据压缩[64,89]。在神经网络兴起之后，学者们使用生成对抗网络实现自动编码器[90]。自动编码器是一种特殊的神经网络模型，常用于提取特征。自动编码器模型通常由三层组成：（1）输

入层；（2）输出层；（3）隐藏层。和输入层和输出层相比，隐藏层的神经元数量较少。该模型的运行过程是先通过编码器将输入转化成维数更少的形态，再使用解码器将数据还原重构原始的输入，该过程成为编码解码-过程。原始的自动编码器强调数据压缩，即通过隐藏层的输出获取原始数据的压缩数据，功能类似于 PCA。本论文使用自动编码器的特征提取能力进行正常样本和异常样本的判别[64]。

自动编码器包含两个过程：

（1）从输入层经过编码器获得编码数据的过程：

$$h = f_\theta(x) = \sigma(W_1 x + b_1)$$

（2）从隐藏层经过输出层获得输出数据的过程：

$$\hat{x} = g_\theta(h) = \sigma(W_2 h + b_2)$$

其中数据重构误差的损失函数为：

$$J(W,b) = \frac{1}{m}\sum_{r=1}^{m}\frac{1}{2}\|\hat{x}^{(r)} - x^{(r)}\|^2$$

通过损失函数可以通过梯度对模型进行优化。

而堆叠式自编码器是由多个自动编码器嵌套的神经网络，其中每个自动编码器包含了一定层数的神经网络用于编码解码，这种模型构建的思路来源于深度学习[91]。这种深度网络的训练是通过依次训练每一层自动编码器实现的。深度的增加将减少训练所需的数据量，并降低计算成本，提升准确率。每个隐藏层的输出用作下一层的输入。堆叠自动编码器的第一层可以作为原始输入进行学习。第二层从第一层学习复杂的特征类别，更高层学习更高级的功能。图 2 所示为堆叠声发射的一个实例。堆叠式自编码器的结构如图 1 所示。确定的函数f_1表示神经网络运算，用来将输入映射到隐空间h_1，然后映射函数g_1从隐变量h_1重构原始输入。第一个自动编码器得到的h_1用于训练第二个自动编码器，使映射函数f_2尝试将h_1转换为名为h_2的隐藏表示。另一个映射函数g_2可以从h_2中重构出原来的h_1。每个自编码器的重构损失函数如下:

$$L_1\left(X,\ \tilde{X}\right) = \sum_{j=1}^{M}\left(x_j - g_1(f_1(\tilde{x}_j))\right)^2$$

$$L_2\left(H_1,\ \tilde{H}_1\right) = \sum_{j=1}^{M}\left(h_1 j - g_2\left(f_2\left(\tilde{h}_1 j\right)\right)\right)^2$$

其中L_1和L_2分别是第一个和第二个自动编码器的损失函数，M是样本数。$X = x_1, x_2, ..., x_M$是原始的输入，$\tilde{X} = \tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_M$是重构的输入。$H_1 = h_{11}, h_{12}, ..., h_{1M}$是第一个编码器的隐藏变量，$\tilde{H}_1 = \tilde{h}_{11}, \tilde{h}_{12}, ..., \tilde{h}_{1M}$是第一个编码器的重构隐变量。

$H_2 = h_{21}, h_{22}, ..., h_{2M}$是第二个编码器构建的隐变量。

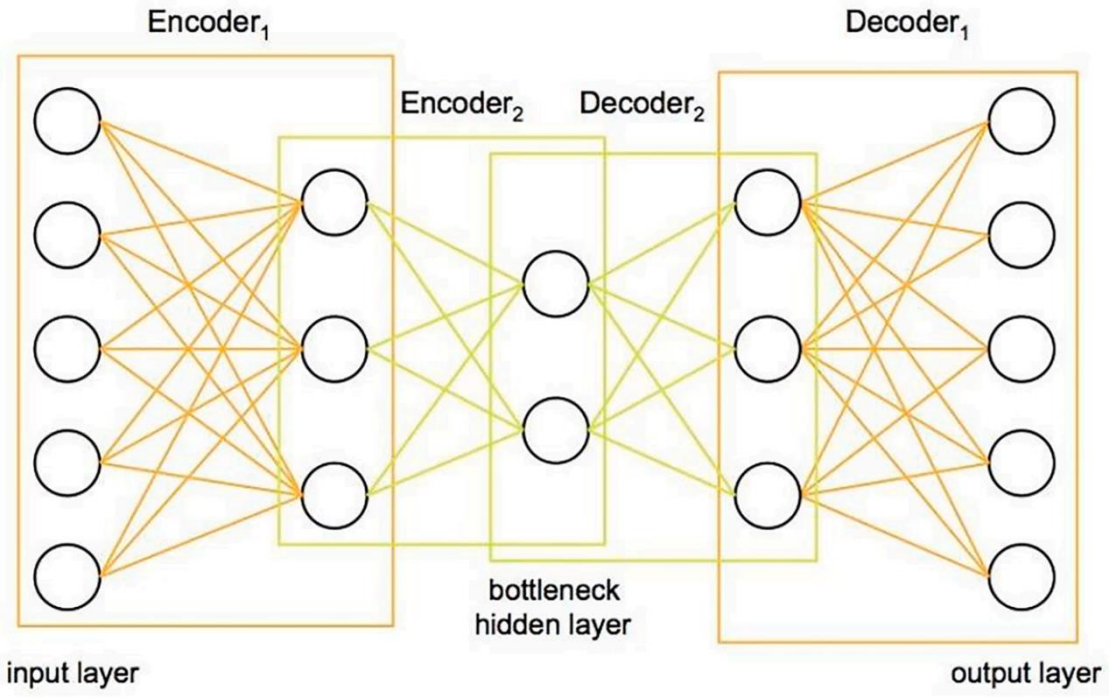

图 3-4 堆叠非对称自动编码器结构

本论文使用的堆叠非对称自动编码器，将两个非对称自动编码器堆叠起来，创建了一个深度学习的层次结构。和常规自动编码器不同，非对称自动编码器没有解码器而是直接计算出隐藏表示[6,9]。相较于对称自动编码器，本文使用的非对称自动编码器在网络结构设计合理的情况下可以减少计算开销，并且对正确率几乎无影响。非对称自动编码器可以对输入的高维数据进行无监督特征提取结构，其训练方式和常规自动编码器相似。

非对称自动编码器接收输入并逐步将其映射到隐藏层，隐藏层的激活函数为：

$$h_i = \sigma(W_i \cdot h_{i-1} + b_i);\ i = \overline{1,n},$$

其中，$h_0 = x$，σ是一个激活函数，n是隐藏层的数量。本研究使用自动编码器，不包含解码器，输出结果的计算函数为：

$$y = \sigma(W_{n+1} \cdot h_n + b_{n+1})$$

使用最小平方误差作为损失函数：

$$E(\theta) = \sum_{i=1}^{m} (x^{(i)} - y^{(i)})^2$$

对于损失函数而言，正常样本的重构损失应该尽可能小，而异常样本的重构损失应该尽可能大。

这个模型可以根据通过学习不同特征间的复杂关系，实现对样本的判别。其中每个非对称编码器使用 14×28×28 的网络结构（以识别 KDD99 数据集数据为例），使用 ReLU 作为激活函数。堆叠非对称自动编码器网络结构如图 3 所示。

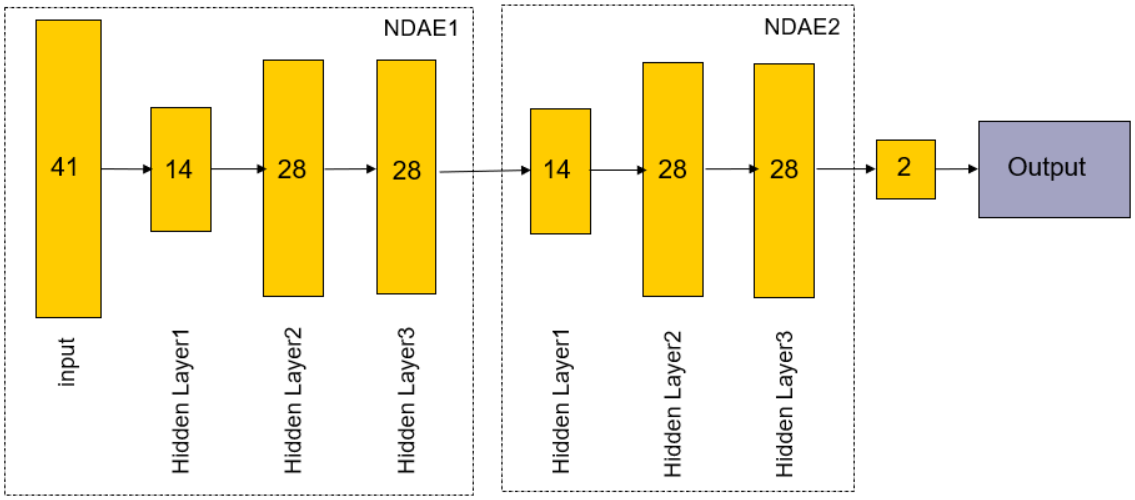

图 3-5 堆叠非对称自动编码器网络结构

自动编码器的优势在于可以更好的提取样本中的特征，但是纯粹的分类能力较弱，直接使用 softmax 函数的分类效果不如其他的机器学习算法。因此，本论文将自动编码器和浅层分类器相结合，共同实现分类。本论文选用 SVM 作为分类器，SVM 在二分类方面有更好的效果[92]。因此在计算出隐藏表示后，将结果输入 SVM 进行分类，实现异常检测。SVM 基于最大间隔原理实现分类，使用最大间隔决策函数。基于核函数 SVM 可以实现非线性分类。软支持向量机（SSVM）和径向基函数(RBF)核函数常常被一同应用到网络入侵检测系统中[93]，因此本论文采用了基于 RBF 核函数的软支持向量机作为浅层分类器。在训练阶段使用堆叠非对称自动编码器计算得到的隐藏表示和正常与异常的标签训练 SVM，获得分类器。在测试阶段使用分类器进行分类。

3.3.2 判别器的训练

原始的生成对抗网络 JS 散度作为目标函数进行迭代，但是 JS 散度存在缺陷：在两个分布不重叠时，JS 散度将为 0，这就造成了生成对抗网络在接近最优时，生成器的梯度将为 0，也就是出现梯度消失问题，无法进行优化。此外原始 GAN 还存在模式崩溃问题，无法生成足够多样性的样本。原始 GAN 中虽然提出了一些解决办法，但是效果较为有限，无法从根本上解决问题。

因此本文在训练时使用 Wasserstein GAN（又称 WGAN）进行训练，该算法用来解决传统 GAN 遇到的梯度消失、训练时梯度不稳定以及模式崩溃等问题。WGAN 使用 Wasserstein 距离衡量分布间的差距。Wasserstein 也称 EM 距离，其原理是将某分布上的数据点全部“推动到”另一个分布上的最短距离总和。设有两

个分布P和Q，x_p表示P中不同位置的数据量，x_q表示 Q 中不同位置的数据量，设γ为数据移动代表的矩阵，那么将数据从 P 移动到 Q 的总代价为：

$$B(\gamma)=\sum_{x_p x_q}^{n}\gamma(x_p,x_q)\|x_p-x_q\|$$

其中$\gamma(x_p,x_q)$表示移动行为，即需要将分布 P 的多少数据移动到分布 Q，而$\|x_p-x_q\|$表示移动的距离。该式将移动的数据量和移动的距离相乘，计算出移动的总代价。这些代价中的最小值表示为：

$$W(P,Q)=\min_{\gamma\in\Pi}B(\gamma)$$

其中γ表示所有可能的移动行为，通过穷举所有可能的总代价，最小的就是分布 P 和 Q 之间的 Wasserstein 距离。

该式的常用表示是：

$$W(P,Q)=inf_{\gamma\in\Pi(P,Q)}E_{(x,y)\sim\gamma}[\|x-y\|]$$

其中$\Pi(P,Q)$表示两个分布 P 和 Q 组合的所有可能的联合分布的集合。γ表示某种可能的联合分布，可以从γ中采样样本点x和y，这两个样本点位于不同分布。$E_{(x,y)\sim\gamma}[\|x-y\|]$表示表示$x$和$y$之间的样本距离的期望值。$inf$表示下界，用来表示所有期望值的最小值。

Wasserstein 距离的相较于 KL 散度或 JS 散度的优势在于：在两个数据分布之间没有重叠或重叠的部分可以被忽略的情况下，仍然可以用该距离衡量两个数据分布之间的差异程度，极大程度上解决了梯度消失和模式崩溃等问题。

将 Wasserstein 用于 GAN，只需将判别器的目标函数更改为：

$$V(G,D)=\max_{D\in 1-Lipschitz}\{E_{x\sim p_{data}}[D(x)]-E_{x\sim p_G}[D(x)]\}$$

该式来经 Wasserstein 距离推导而来。其中$1-lipschitz$函数用于约束输入的变化率。$lipschitz$函数定义为：

$$\|f(x_1)-f(x_2)\|\leqslant K\|x_1-x_2\|$$

该函数要求输出的变化要小于K倍的输入变化，这个限制条件要求输入变化不能导致输出变化太大，否则就会违背这个不等式，也就是要求输出的变化率不能过高，从而保证了函数足够平滑。当倍数 K=1 时，就称此时的$lipschitz$函数为$1-lipschitz$。

基于这个理论进行训练，在更新判别器的时候使用 WGAN 的目标函数，在生成器生成的异常样本的辅助下进行优化。

3.4 网络整体架构和算法的训练

互补生成对抗网络的网络流量入侵检测模型结构如图 3-6 所示。算法架构是较为典型的生成对抗网络。算法的描述如下：

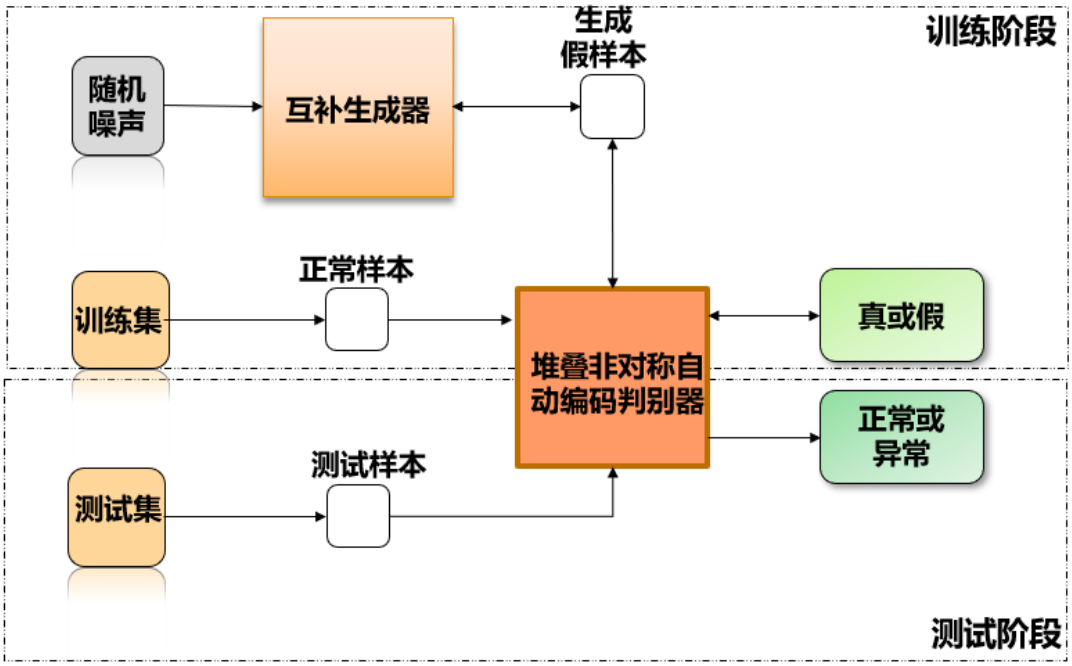

图 3-6 互补对抗生成网络的整体结构

算法 互补生成对抗网络流量入侵检测算法

输入：训练数据集 S，随机噪声 z

输出：最后的分类结果，样本是否正常

训练阶段：

1） 数据预处理阶段：将训练用的数据集进行清洗，填充缺失值，使用独热码对离散特征进行编码。将所用数据进行归一化，保证数据的范围是[0,1]。使用训练好的自动编码器对数据进行降噪处理。

2）将随机噪声输入互补生成器，互补生成器生成伪造的样本$\tilde{X}$。将伪造的样本标注为异常和训练数据X一同输入堆叠非对称自动编码器中。堆叠非自动编码器经过计算后获得隐藏表示并经过 SVM 分类器得出样本是正常还是异常。

3） 根据样本的分类结果计算判别器的损失L_D，根据损失和判别器的目标函数$V(G,D)$进行更新，迭代生成新的自动编码器权重和 SVM。

4）同样根据生成器的损失L_G对生成器进行更新，根据优化函数对互补生成器的权重进行更新，让生成器生成更接近真实样本的伪造样本。

5） 交替进行 3）和 4），直至达到设定的训练轮数或是正确率趋于稳

定。

测试阶段：

用包含正常样本和异常样本的测试集进行测试，计算判别器的准确率。

3.5 实验过程和结果分析

如 2.5 节介绍针对网络流量的入侵检测有多个公开数据集，其中被广泛使用的是 KDD99 数据集和 CICDS-2017 数据集。关于这两个数据集见前文的介绍。本文在使用数据集之前首先对数据进行清洗并归一化，去除不需要的特征并填充缺失值。对部分字符串类型的特征（比如使用协议，服务类型）转化为数值表示。在使用之前将数据集划分为训练集和测试集。

本论文使用的实验指标见 2.6 节。

3.5.1 实验环境

本章实验使用 windows10 操作系统，Intel(R)Core(TM)i7-10750H CPU@2.90GHz 处理器，内存 32GB，NIVDIA GeForce RTX 2060。编程工具为 TensorFlow2.7.2，python3.6。

3.5.2 算法性能

为了便于模型性能的评估，需要找到模型训练最优的学习率（lr）。本文利用控制变量法，在其他参数恒定的情况下控制学习率，设置学习率区间为[0.1, 0.0001]，生成等步长的 10 个不同的学习率进行测试，训练轮数为 200 轮。不同学习率的检测效果如图 3-7 所示：

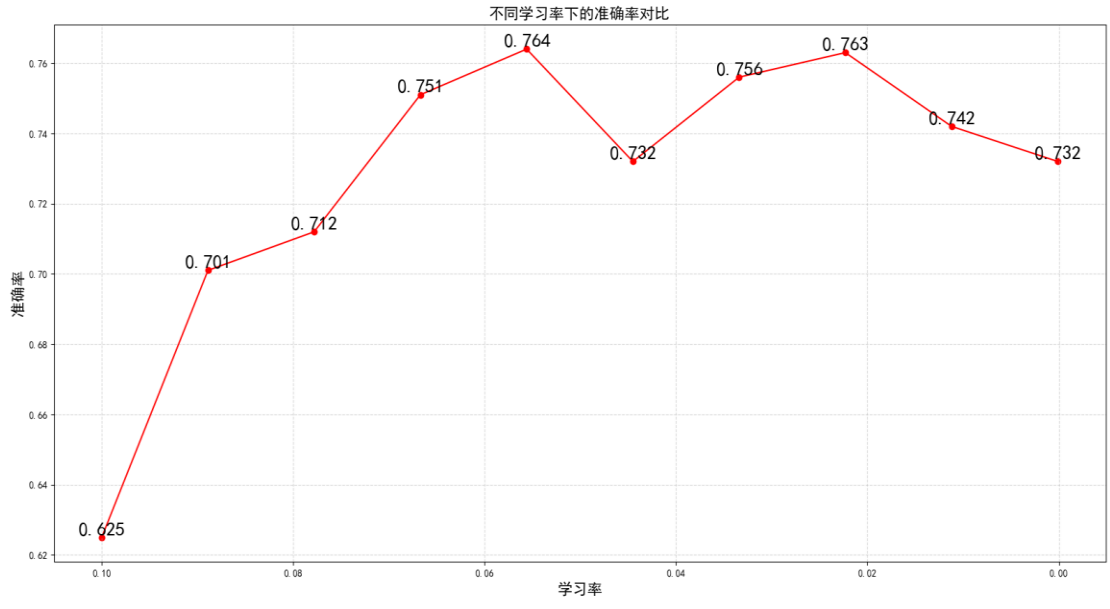

图 3-7 不同学习率的准确率率对比

从图中可知，学习率不同时，模型识别异常能力有所差异，根据图中不同学习率在测试集上的表现，学习率介于[0.05, 0.001]之间时准确率最高。

找到模型训练最佳轮数（epoch）。固定模型结构权重，学习率等条件，在数据集中抽取 5000 个样本（其中包含 2500 个正常样本和 2500 个异常样本）作为测试集。在此基础上对模型进行训练，从 150 轮开始直至 500 轮结束，记录每次的准确率。不同训练轮数的检测效果如图 3-8 所示：

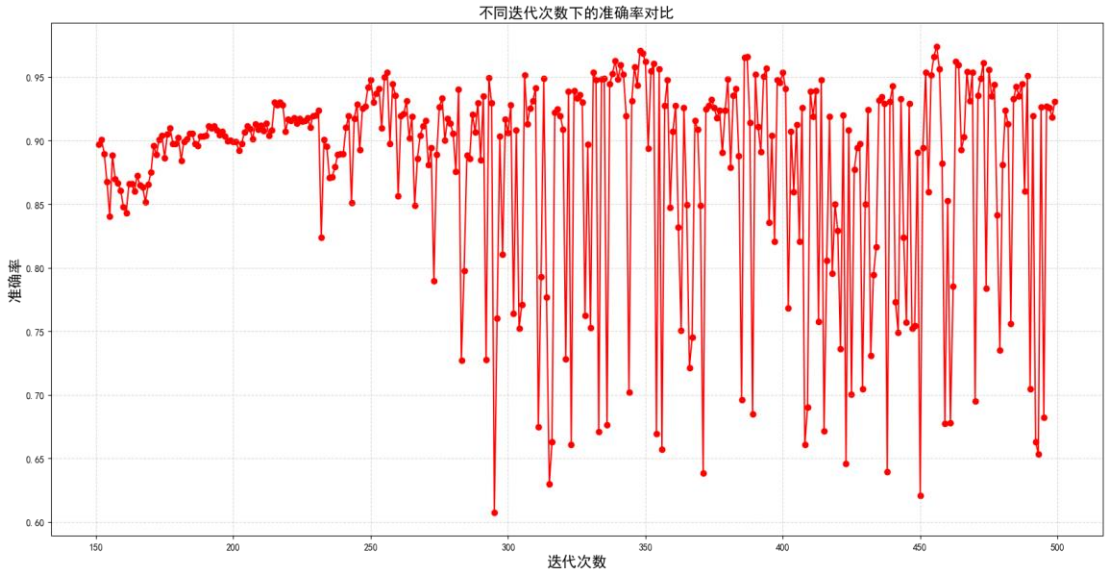

图 3-8 不同学习率的准确率率对比

从图可知，在迭代轮数达到 200 时算法性能已达到最优，因此为了减少训练时间本论文设置迭代轮数为 200。找到模型训练最佳批次数（batch）。固定模型结构权重，学习率等条件，在数据集中抽取 5000 个样本（其中包含 2500 个正常样本和 2500 个异常样本）作为测试集。在此基础上对模型进行训练，设置批量区间为[50, 256]，生成等步长的 30 个不同的批次进行测试，每个批次训练 300 轮。记录每次训练后得出的正确率。不同的训练批次的检测效果如图所示。

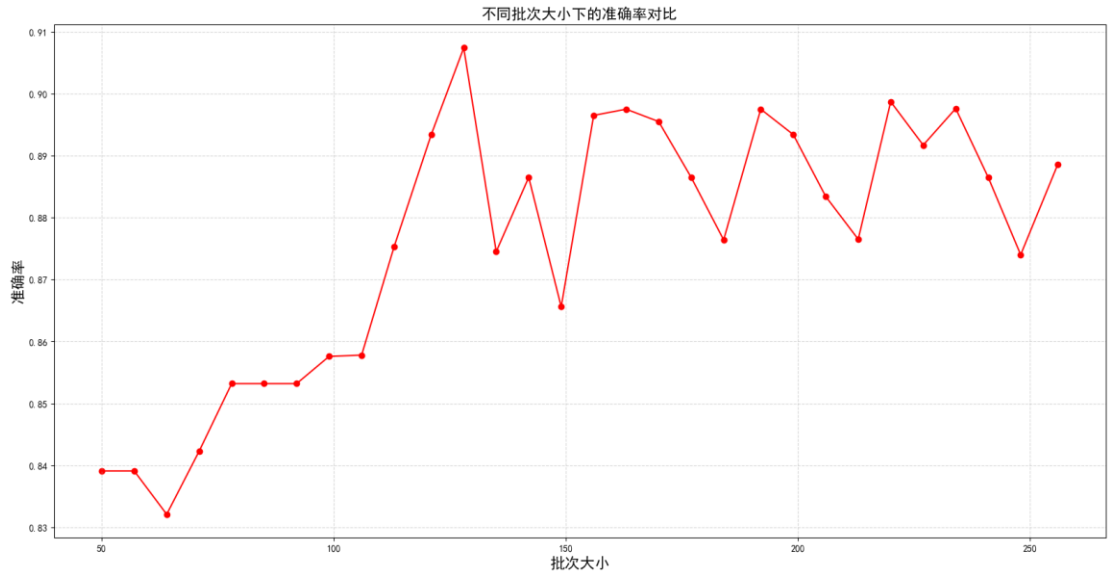

图 3-9 不同训练批次下检测效果的对比图

从图中可以看出，在训练批次小于 100 的时候性能较差，当训练批次大于 100 直到 256 为止性能较为稳定，因此本论文设置训练批次大小为 128.

从测试集中正常样本和异常样本比例的角度测试模型的性能。在实际应用中，需要在测试前计算出污染率 c%，计算出所有样本的异常得分后，取分数最高的 c%，将其相关的样本标记为异常入侵。污染率的定义为异常样本的经验比例：

$$\frac{anomalous}{normal + anomalous}$$

考虑到不同的网络入侵发生频率，本文测试了不同的污染率对模型的影响，因此本文选择 20%，10%，5%和 1%逐步增大的比例作为污染率，在模型上进行测试并计算准确率，召回率和 F1 值。测试结果的性能变化如图所示。

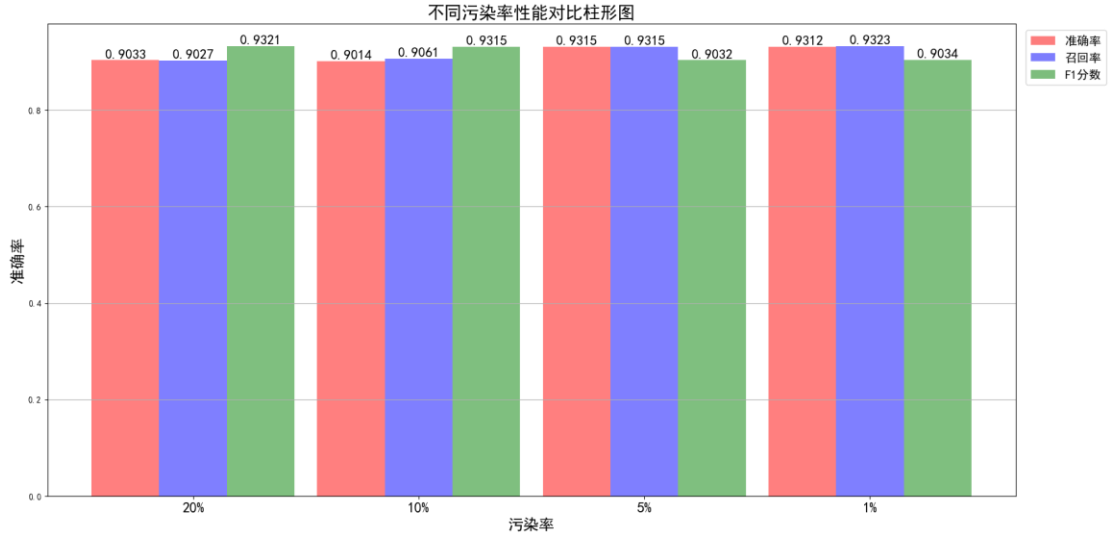

图 3-10 不同污染率下的性能对比图

从图中可以看出：当污染率的数值较小的时候性能较好，表明当测试数据中异常样本比例越小正确率越高，但是对性能的整体影响不大。

本文提出的算法在测试集上二分类测试结果的混淆矩阵如图所示：

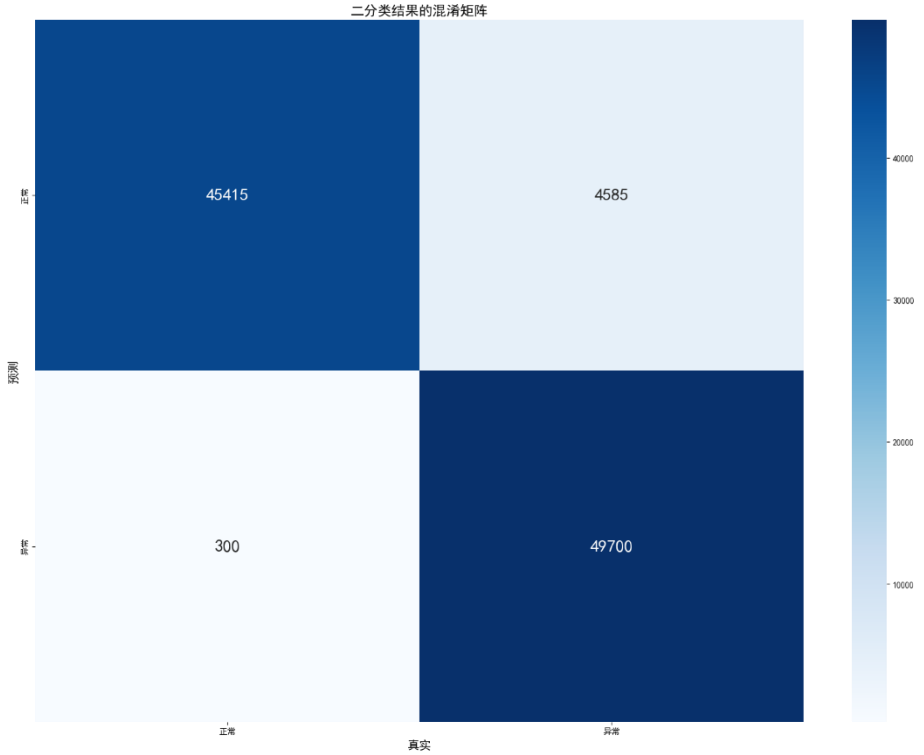

图 3-11 二分类结果的混淆矩阵

我们将本文提出的模型和孤立森林、OC-SVM、其他基于 GAN 的算法进行对比。我们测试了选择算法的准确率、召回率和 F1 分数，并将本论文的模型的结果和其他算法的运行结果进行对比，取得的结果如表 3-1 和图 3-12 所示：

表 3-1 不同算法的性能对比

模型	准确率	召回率	F1 分数
Isolation Forest	0.4415	0.3260	0.3750
OC-SVM	0.7457	0.8523	0.7954
DSEBM-r	0.8521	0.6472	0.7328
DSEBM-e	0.8619	0.6446	0.7328
AnoGAN	0.8786	0.8297	0.8865
BiGAN	0.8698	0.9523	0.9058
Efficient-GAN	0.9324	0.9473	0.9398
My Model	**0.9315**	**0.9316**	**0.9315**

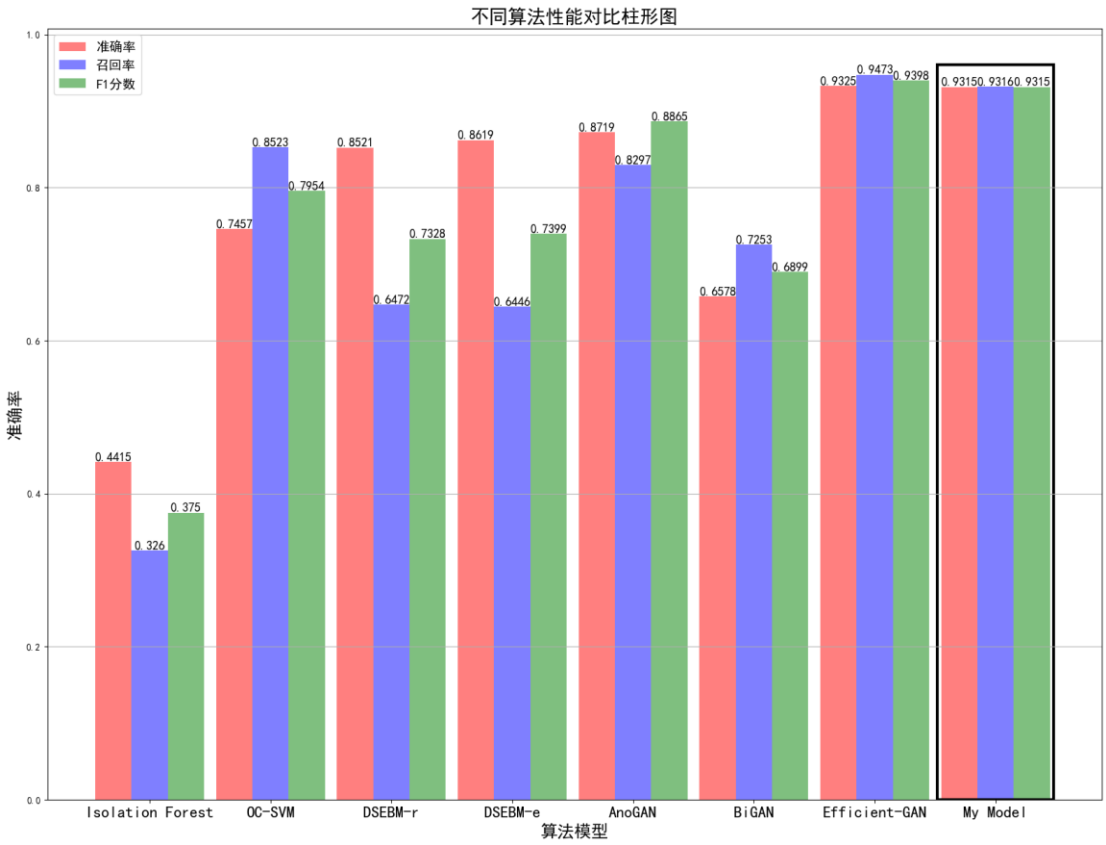

图 3-12 不同算法的性能对比柱形图

从图中可得，本文提出的算法在性能上仅次于 Efficient-GAN，很大程度上优于其他之前提出的算法。从准确率角度本文提出的算法较优。

另一个较为重要的指标是运行速度。在测试上本文使用的算法只使用判别器进行异常分类，在理论上计算量上要小于需判别器和生成器计算异常分数的算法，例如 AnoGAN。因此本文下相同的机器上将本文提出的算法和 AnoGAN 和 BiGAN 比较。在本次测试中随机选择 100000 个样本（50000 个正常样本，50000 个异常样本）作为测试集，共计算 10 次记录每次测试消耗的时间，取消耗时间的平均值作为比较的标准。

表 3-2 不同算法的时间性能对比

模型	时间（ms）	基准时间
AnoGAN	132443	4.42
BiGAN	35986	1.20
Our Model	**29986**	**1**

经过比对可以得出本论文提出的算法在运算速度上好于 BiGAN，远好于 AnoGAN。

3.5 本章小结

基于生成对抗网络架构，本章设计了一个基于互补生成器和非对称堆叠自动编码器判别器的生成对抗网络。该算法在训练时可以只使用正常样本进行训练，在测试阶段可以识别出异常样本。该算法利用了互补分布的思想，训练生成器生成样本分布的互补分布辅助训练，以此提升了运算的性能。并使用非对称编码器来提升运算的性能。该算法相较于传统算法极大的提升了识别准确率并在很大程度上提高了运算速度。该算法适用于没有或缺少已知的异常流量的网络数据并适用于实时入侵检测。

第四章 基于 LSTM/GRU 的生成对抗网络日志日志异常检测

本章提出了基于堆叠 LSTM/GRU 的日志异常检测模型。主要包括日志解析和日志异常检测两个部分内容。在日志解析方面，本文使用了一种基于前缀树的日志解析算法。该算法计算过程简单，但准确性较高。此外本文提出了一种基于生成对抗网络的日志异常检测算法。该算法可以根据时间戳读取日志序列，并判断当前读取日志数据是否正常。并介绍了该算法的具体训练过程。然后对本文提出的算法的性能进行评估并和其他同类算法的性能进行对比。最后进行了本章小结。

4.1 引言

系统日志的自动生成对于大规模系统是必不可少的。它们可以记录系统的状态和关键事件的细节，方便管理员发现和解决系统的 bug、故障和错误。因此，日志到达的密度和日志的描述会影响知识量对于提高运行系统性能的重要性。在防御网络系统发生安全问题时，日志分析是有效的途径。部分攻击无法从流量角度识别或进行拦截，此时日志分析就成了必不可少的手段。对日志日志进行异常检测有很多挑战：1）日志作为非结构化数据，描述过于口语化和模糊不清，无法反映系统的状态；2）日志数据是时间序列，传统的生成对抗网络结构无法很好的处理这种数据；3）在日志数据中正常实例远多于异常实例，这种情况下日志异常检测系统面对着快速响应和精确诊断的巨大挑战。日志异常检测的过程如图 4-1 所示。

日志模板的抽取即日志解析是实现日志异常检测的前置操作。最原始的日志数据是以行为单位的日志文档，日志解析的目标是将日志由行转为结构化事件序列，将日志结构化处理便于对日志进行分析。这个过程可以称为日志解析（log parse）。日志解析存在很多困难。首先是日志模板的缺点困难，日志解析最核心的步骤就是确定日志模板，但是只有从代码层面入手才能确切的识别出日志的打印格式，这通常是不现实的。因此只能从统计角度分析日志的打印格式。通常单条的日志数据可以分成常量部分和变量部分，其中常量部分由固定的文本构成，代表了日志对应的日志模板，在日志系统中固定不变。第二个问题就是日志解析对性能要求高。在实际场景中，日志数据规模非常大，通常会达到 TB 级别，日志解析的效率过低时会影响异常检测的性能。传统日志解析非常依赖人工定义的正则表达式，也就是由人工识别日志模式，但是在现今超大规模的日志系统中这种方式非常低效。现代系统中的日志打印语句频繁地更新，使该问题更加复杂。

此外，开发人员必须定期检查各个系统组件更新的日志打印语句，以维护正则表达式。因此，现在学者们开始研究更为高效的日志解析算法，例如基于聚类的日志解析算法、基于启发式的日志解析算法。基于此，本文提出了基于前缀树模板的层次聚类日志解析算法。

在日志的异常检测方面，日志数据结构复杂，数据量极大，包含了各种异构的本结构化数据，有人工对日志数据判断需要有领域知识的工作人员来完成。因此在实际的场景中具有正常或异常标签的训练样本数量有限。针对现实的问题，本文提出一种无监督的日志检测检测算法，在训练阶段使用正常数据进行训练，在测试阶段用混合了正常数据和异常数据的测试集进行测试。该算法使用生成对抗网络结构，用生成对抗网络的竞争学习的模式，训练生成器用于生成尽可能接近正常样本的伪造样本，训练判别器尽可能识别出生成样本还是正常样本。生成器使用堆叠 LSTM/GRU 构成自动编码器对输入的数据进行重构，要求对正常数据的重构误差尽可能低，自动编码器可以通过数据重构实现异常检测。判别器则使用堆叠 LSTM/GRU，将时序日志数据输入进行异常检测，判断当前的数据是否符合其所在时间窗口内日志数据的分布规律，最后用 Softmax 层输出样本是正常还是异常的概率。在进行异常判别是需要综合生成器的重构误差和判别器正常异常的概率，计算最后的异常分数，并与阈值比较得出最后的结论。该算法可以减少因为有标签的数据量不足而造成无法训练或无法识别异常的问题。

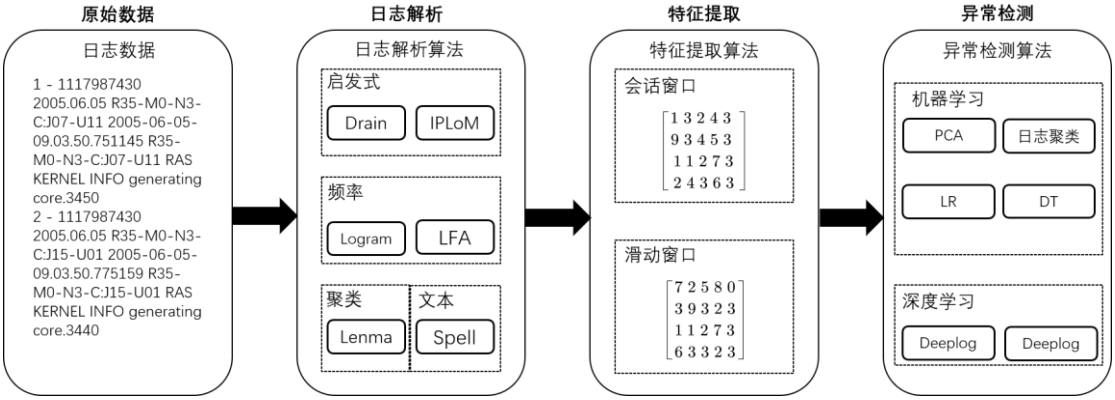

图 4-1 日志解析异常检测的示意图

4.2 日志解析

4.2.1 预处理

日志进行分析前首先需要进行日志预处理，即将日志数据处理成便于算法使用的状态。这一过程通常称为日志的实体识别，目标是识别出日志中的关键信息。

首先需进行特定领域知识的预处理。学者发现利用领域知识进行简单的预处理可以提高解析精度。预处理需要领域专家提供相应的正则表达式[94]。这个过

程主要用于处理一些专有名词或是特定属性限制。主要有两个方式，一个方式是删除日志中特定的字段，例如通过正则表达式删除 HDFS 日志数据中的 block_id 部分。另一种方式是确定日志的所属的类别，也就是将选定的日志绑定到同一模板，这一过程提供了由专家手动指定模板的能力。

然后是通用的预处理过程：用通配符替换掉日志中的 IP、时间戳等参数并删除特殊符号例如标点符号和停顿词。这一步骤用于去除日志解析时不需要的参数，留下需要的参数。这一过程通常使用正则表达式来完成。这一处理可以去除和日志解析无关的数据，减少冗余的数据，提高效率。

表 4-1 符号在 5 个日志数据集上的计数结果

符号	HDFS	OpenStack	Spark	Hadoop	Zookeeper
.	400450	1005200	205750	728900	228850
:	250500	615000	333050	528200	501450
/	177800	352750	223850	30600	69950
-	67350	1514500	8900	394100	407250
_	163700	1813500	68850	178850	
$	52850			650	71700
*	32950			100	14250
(	50	15800	37900	16050	6850
)	50	15800	37900	16500	6850
[		127500	100	118100	108150
]		127500	100	118100	108100
“		101700			
,		23700	88200	122100	134900

最后是统计日志中的词频，并将列表按从词频从高到低降序排列。这一过程是为了后续处理做准备。

4.2.2 前缀树层次聚类

该过程称为日志的模式识别。本文以层次聚类的方式建立日志模板树，对日志集合构建前缀树。

将日志数据集词表中包含的词分为模板词和参数词。参数词没有语义信息，通常以不同的形式出现。模板词显示日志条目的事件模板，并且对于每次事件发生都保持相同。每个日志条目包括三部分：时间戳、日志类型和日志事件。时间戳记录事件发生的时间，日志类型通常表示事件的严重程度（如 INFO 或 WARN），日志事件记录系统执行情况。首先解析原始日志以查找日志模板。其次，提取每个日志模板的日志键。选择日志类型、日志子类型和日志事件的第一个单词作为

日志键，每个日志键都对应一个日志模板。最后构建一个字典 C，它包含所有的日志模板词，一个日志字典的示例如表 4-2。将日志实体序列作为日志数据集，每个日志实体都由多个单词组成，包括日志键和参数词。出现频率很高的词将被视为日志键[39]。构建日志模板前缀树的形式如图 4-2 所示。

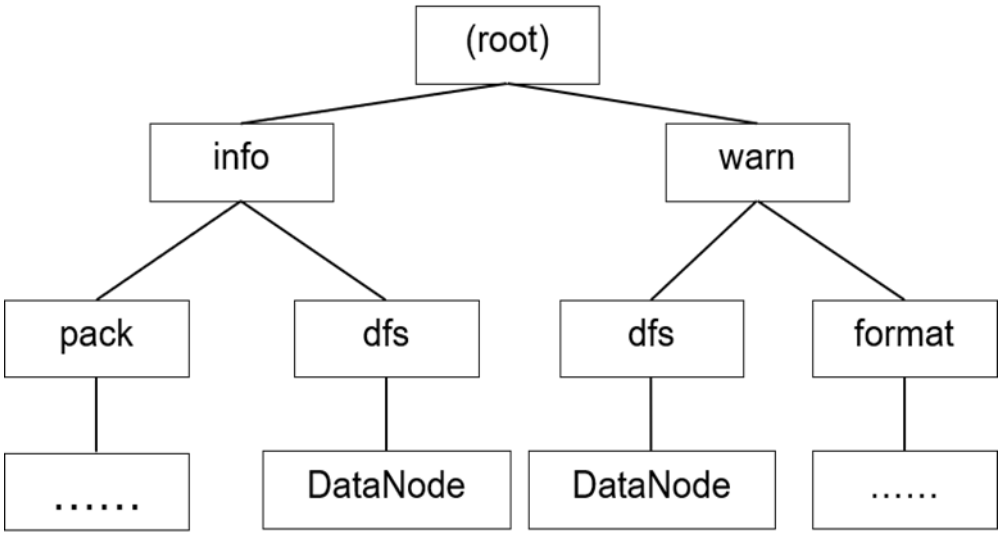

图 4-2 日志模板树示例图

在日志模板提取中，计算日志事件中单词在数据集中出现的次数（即词频），并推导出一个单词列表 F，单词的词频按降序排列。每个单词都有一个固定的序列号（ID）。当新的日志条目到达时。我们创建树的根，它被标记为 log 类型，如 INFO 或是 WARN。然后根据列表 F 中单词的 ID 构造树的第一个路径。当处理下一个日志条目时，如果当前日志条目与处理过的日志条目具有共同的前缀单词；则与现有路径共享一个公共前缀;否则，将创建一个新的分支作为节点的子树。也就是根据前缀对日志数据进行层次聚类，因为不同的日志模板的前缀通常不同，利用前缀可以将相关的日志消息准确的划分为一起。每个子树构成的日志消息基本相似，但仍然可能存在不同 log print 语句输出的日志消息位于不同子树的情况。例如，在一个日志文件中，一种特定的日志类型只会记录两个事件，一个服务成功，另一种是服务失败。这意味着这些日志由两种 print 语句输出。那么两个 print 语句之间的区别可能是其中一个单词在某个位置上是不同的，例如一个是' success '，另一个是' fail '。这两个日志数据不会，那么前一个聚类步骤将把日志分类到不同子树，这是不合理的[94]。这个问题需要经过子树合并的过程得到解决。

构建完成后进行子树合并。日志数据分为常量（不变部分）和变量（可变部分），常量是打印语句的固定信息，也就是代码中的模板关键字，而变量会根据日志打印时的参数进行改变。日志数据中的公共部分就是我们所希望得到的日志模板，我们需要忽略不同的部分。基于前缀树的层次聚类同时将常量和变量用于

聚类，因此在聚类完成之后需要对子树进行合并。这一部分需要日志中所有位置单词的统计信息，并根据单词频率设置分离阈值。当某个前缀树节点的 token 值的频率超过了阈值，则将该位置的 token 视为变量，否则是常量也就是日志模板词。当某个位置的 token 被视为变量时，我们需要对日志进行更改，并将该 token 更改为<*>通配符，表示在提取日志模板词时忽略该 token。然后再针对更改过的日志数据对前缀树节点进行节点合并，将相同前缀子节点合并到同一子树上。当某个位置的 token 被认为是常量时，前缀树不变。这将多次继续子树合并操作，算法首先将公共 token 序列保存在子树内，然后用上述操作遍历处理其他 token。子树合并的过程如图 4-3 所示。

然后是剪除出现频率较少的子树。对于每个前缀树节点，都应该包含较少数量的字节的。每个字节的参数类型可以是多样的。对于日志模板树，若一个节点的子树较多，则该节点的子节点可能是参数词，需要舍弃掉。基于此，设定当一个节点的日志词词频不在最高的 k 个词中且该节点有超过 l 个子树点，则删除该节点和该节点的所有子树（其中 k 和 l 是自己设置的阈值）。此时从根节点到叶子节点的每条路径就是一个日志键。这棵前缀树就构成了日志词典，其中词典中的每个日志键都有唯一 ID 作为唯一索引，前缀树如图 5 所示。然后对每个日志项使用 LCS 算法匹配日志模板并提取其中参数构成日志向量。在提取日志向量的过程中会使用滑动窗口，一次性读取窗口大小的日志序列数据作为日志时间序列输入模型中。

表 4-2 日志模板词典示例

模板 ID	日志模板
1	Connection closed by <*>
2	Connection reset by <*>
3	Invalid <*> from <*>
4	PacketResponder <*> for block <*>

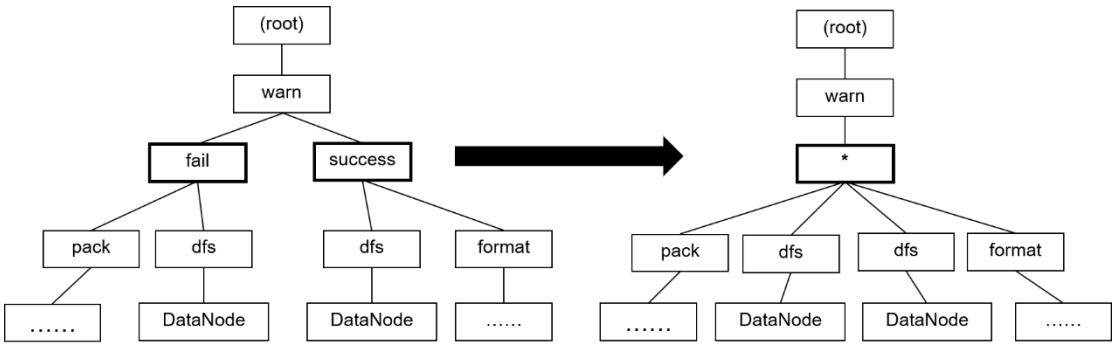

图 4-3 日志子树合并示意图

4.2.3 基于 Spell 的在线日志解析

之前的操作基于的是离群日志数据的日志解析，但是部分程序需要在线的监控和处理。因此本文引入了 Spell 算法用于日志解析，用来在离线生成的日志模板的基础上在线添加日志模板。Spell 基于 LCS 算法，并使用 LCS 算法构建了日志流解析器[43]。

在本文中，每个日志模板和参数作为一个 Spell 的 LCSObject 数据结构，其中日志模板作为 LCSseq，LCSObject 的总和即为 LCSMap。在 Spell 中 LCSObject 还保存了 LCSseq 相应日志条目的 logID。Spell 的基础流程是当一个新的日志条目e_i到达时，我们首先将它与 LCSMap 中已有的 LCSObject 中的所有 LCSseq 进行比较，然后根据结果，将行 ID 插入到已有的 LCSObject 的 logID 列表中，或者计算一个新的 LCSObject 并将其插入到 LCSMap 中。

在实时更新的日志流中，当新的日志条目出现时，首先将该日志条目进行预处理解析成为一个 token 序列。然后将该 token 序列和之前得到的日志模板集（LCSMap）中所有的 LCSObject 的 LCSseq 进行对比。以查看日志序列是否匹配现有的 LCSseq 中的一个，如果是，则认为该日志的日志模板为匹配到的日志模板，否则我们需要为 LCSMap 创建一个新的 LCSObject，也就是增加一个日志模板。

Spell 算法使用 LCS 算法计算匹配度。设日志条目e经过处理后得到 token 序列s。搜索 LCSMap，假设其中第 i 个 LCSObect 的 LCSseq 是q_i，计算$LCS(q_i,s)$的值l_i。搜索过程中保留最大的l_i值和对应的 LCSObject 的索引。搜索完成后，如果$l_i=\max(l_i's)$大于阈值τ(默认情况下$\tau=|s|/2$，其中$|s|$表示序列s的长度，即日志条目s中的 token 数量)，则认为 LCSseq 和日志序列s有相同的日志模板。也就是说，q_j和s的 LCS 是 LCSMap 中所有 LCSObject 中 LCS 的最大值，且 LCS 的长度(q_j,s)至少为s长度的一半;因此，除非e中参数词的总长度超过其大小的一半，否则$LCS(q_j,s)$是一个判断第 j 个 LCSObject 中的日志条目是否与e属于相同的日志模板(即是$LCS(q_j,s)$)的指标。如果存在多个具有相同最大值的 LCSObject，则选择索引值最小的哪一个，因为它与新日志序列集合相似性值较高。

用 Spell 算法添加的日志模板可能存在将日志键元素被错判断为参数，或者参数数量过多时参数被错判断为日志键元素的情况。例如 Spell 算法可能会将不同日志类型的日志划分到同一日志模板中，例如日志实体为如下的情况时：

```
info Error: Console-Busy Port already in use
info Error: Console-Busy Port already in use
warning Error: Console-Busy Port already in use
```

Spell 算法会将日志模板提取为：

* Error: Console-Busy Port already in use

但是明显这是两种日志模板。因此需要对 Spell 算法加入的日志模板进行合并和拆分处理。

合并过程首先将可能具有相同消息类型的当前 LCSObject 聚类在一起，然后在每个位置计算不同标记的数量这些 LCSObject 中的 LCSseq。在将 LCSMap 划分集群的过程中，如果每个集群满足一定条件，就认为个集群内的所有 LCSObject 具有相同的消息类型。因此，我们将它们合并为单个 LCSObject。

分割过程是在 LCSObject 中引入了另一个字段 Params，这个参数是一个键值对集合，这个键值对记录从历史记录出现的所有参数词位置的所有参数值，以及每个参数词出现的频次。拆分过程的基本思想是利用上述的键值对，对所有的 LCSObject 遍历其中包含的 LCSObject.Params 字段中键值对应的参数列表，并统计每个参数位置上唯一 token 的数量。如果该位置的唯一 token 数量小于拆分阈值，则认为该位置对日志模板拆分有影响。在这种情况下，将该日志模板拆分出几个新的 LCSObject，用每个新 LCSObject 的 LCSseq 中同一位置的唯一 token 替换该日志模板中的参数。

本文提出的日志解析算法流程如下所示：

算法：日志模板提取算法
输入：日志数据集、停用词列表、专业领域知识正则表达式 **输出：日志模板列表**
1） 日志预处理 **对每个日志项** **1) 用专业领域正则表达式处理日志数据** **2) 删除日志数据中的停用词** **3) 用通配符替换掉日志中的参数词** **2） 日志模板提取** **构建一个前缀树用来提取模板** **1) 用一个空节点作为根节点，用日志等级作为下层节点** **2) 对所有日志数据根据前缀进行聚类，构成前缀树** **3) 对前缀树进行子树合并，去掉参数词节点** **4) 对前缀树进行剪枝，删除频率较低的节点** **根据前缀树提取日志模板** **3） 用 Spell 在线添加模板** **预处理日志序列**

判断日志序列是否属于某个模板

1) 根据 LCS 算法计算$LCS(q_j,s)$

2) 找到$l_i=\max(l_i's)$

3) 有相同l_i的情况下选择索引最小的

4) 根据l_i的值判断是否要提取当前的日志作为日志模板

根据上一步的结果添加日志模板

4.3 日志异常检测

在日志异常检测中，首先构建日志序列，可以将日志序列理解为正在发生的一系列事件，也就是原始日志序列中隐含的日志模板序列。对输入的日志序列，以滑动窗口的形式构建子序列，并以子序列为单位进行异常检测。

4.3.1 基于堆叠 LSTM 自动编码生成器

在涉及到时间序列数据时，LSTM 算法可以很好的处理数据的序列特征。GRU 的结构和 LSTM 非常相似，但是少了一个门控单元，极大的降低了运算的成本和时间。在基于 GAN 模型的算法中，生成器需要输入符合概率分布为$p(z)$的输入z。早先提出的 GAN 架构使用全连接神经网络和卷积神经网络生成数据的方式无法体现出时间的时序特性，因此本文通过使用 LSTM 模型来接近这个问题。LSTM 和 GRU 是一类基于门控的循环神经网络，核心思想是通过门控单元控制记忆信息的积累速度。传统的 LSTM 拥有三个门：遗忘门，输入门和输出门。GRU 在 LSTM 的基础上将输入门和遗忘门合并，同时省略了记忆单元，直接用历史状态h_{t-1}线性变化到当前状态h_t。在传统的单层 LSTM/GRU 网络之外，基于深度学习的理念可以将多层网络堆叠起来，构成堆叠循环神经网络（Stacked Recurrent Neural Network，SRNN），将第$l-1$层的输出作为l层的输入。双层 LSTM/GRU 的网络结构如图 4-4 所示。

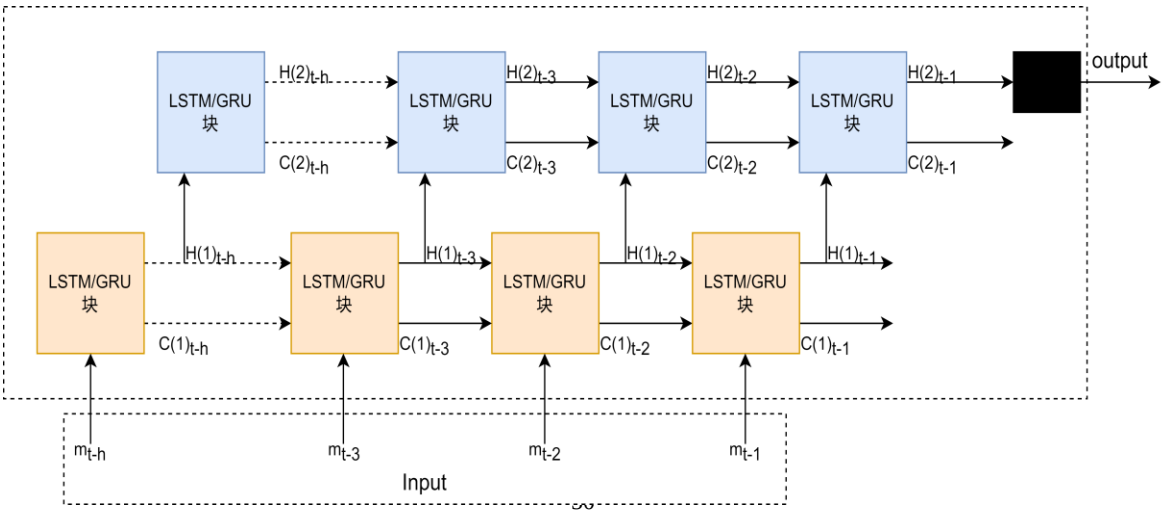

图 4-4 双层 LSTM/GRU

在本论文中，选择基于 LSTM 的编码器-解码器方案来构建生成模型 G。整体的网络使用了基于堆叠 LSTM/GRU 的自动编码器结构。在将 h 和输入样本输入相同结构的解码器中获取重构的样本。编码器用两层的 LSTM/GRU 结构组成，进行运算获取隐藏状态h。

基于 LSTM/GRU 的生成器一种思路是基于自动编码器的思路，即利用自动编码器的重构能力，将整个生成器分成编码器和解码器两部分。常见的思路是使用自动编码器（Autoencoder，AE）作为 GAN 网络的生成器。自动编码器拥有数据重构能力，可以通过对输入数据编码得到隐藏表示，在经过解码器重构得到重构数据的方式学习数据分布[39]。设定日志集合$S=\{s_1,s_2,...,s_n\}$，其中n是日志模板的总数，s_i是一个日志模板。每个日志模板s_i都由多个词组成，$s_i=\{w_1,w_2,...,w_n\}$。 其中每个w_i表示日志模板s_i的第i个词，其中部分词是日志键。设s_i的日志键为$K=\{k_1, k_2, k_3...,k_L\}$，长度为$L$，$h_E^{(i)}$是编码器在$t_i$时刻的隐藏状态。将序列数据$K$输入编码器中，并和先前的隐藏状态$h_E^{(i-1)}$进行运算获取当前状态的隐藏状态$h_E^{(h)}$。将隐藏状态作为解码器的输入和解码器的隐藏状态运算得到生成的$\hat{k}$。在当前的场景下K是先前的日志序列，$\hat{k}$是根据先前的日志序列重构的当前时刻日志键。通过计算$\hat{k}$和当前时刻实际的日志键k的重构误差来进行更新。基于 LSTM/GRU 的自动编码生成器如图 4-5 所示：

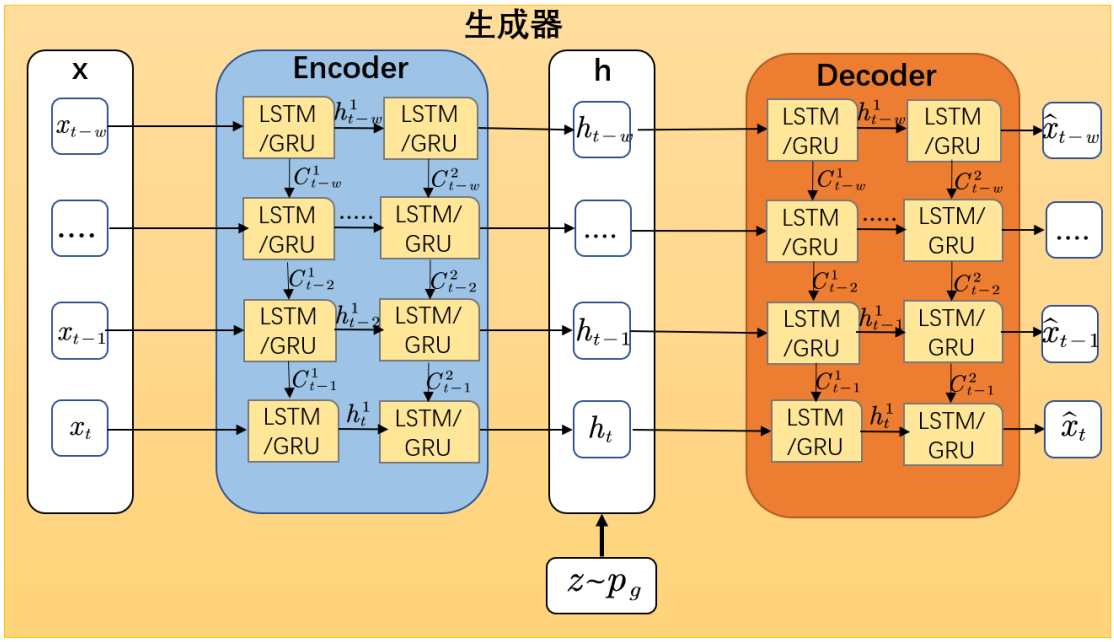

图 4-5 基于 LSTM/GRU 的自动编码生成器

而解码器将编码器的获得的隐藏状态h_E作为初始化状态。为了编码在训练是出现过拟合的问题，将随机生成的噪声z_h和h_E一同输入解码器中。以这种方式处理时间窗口的内的时间的序列。可以得到每次的隐藏状态，并将隐藏状态输

入到 Softmax 函数中，得到一个概率分布，该概率分布就是表示重构的日志键来自日志词典中每个日志键的概率。我们选择具有最高概率的单词作为输出。单个日志项输出可以表示为：

$$\hat{x}_i = g(x, h_D^{(i)}, h_E^{(L)})$$

其中$h_E^{(i)}$是解码器 LSTM 在t时刻隐藏状态，x_{i-1}是前一时刻的输入并作为t时刻的输出。g是一个非线性的多层感知器，它表示$\hat{x}_i$属于日志键词典中每个日志键的概率。整体重构的日志序列可以表示为：

$$\hat{X} = G(X, h_D, h_G)$$

该生成器的核心在于将日志键序列k输入编码器获得隐藏状态$h_E^{(h)}$。该隐藏状态可作为日志键序列k的潜在表示z。该过程等同于将真实数据序列$x \in X$映射到潜在空间$z \in Z$。然后通过解码器将隐藏状态还原到$\hat{k}$的过程相当于原始生成对抗网络中生成器生成数据的部分，即将潜在分布映射到真实数据空间$Z \to X$的过程，也就是$G(z)$~X。x和$G(z)$的形似程度取决于x的分布和我们用于训练生成器G的分布p_g的紧密程度。因此编码器的映射G^{-1}:$X \to Z$，解码器G:$Z \to X$[38]。

设计面向日志的生成对抗网络入侵检测模型。在我们的研究中，选择基于 LSTM 的编码器-解码器框架来构建生成模型 G。首先，将日志关键字输入到编码器阶段的 LSTM 中，得到固定维数的隐藏状态。经过编码器处理后，将隐藏状态输入到解码器相位的 LSTM 中。最后，从生成器获得日志模板（即，我们训练生成器生成日志模板的实例）。对于编码器：LSTM 编码器学习输入的定长向量表示。给一个日志集合，每个日志集合都由若干个日志模板组成。每个日志模板含有若干个日志词，部分日志词输入日志键。将日志键输入 LSTM 编码器中。记忆单元是 LSTM 的核心部分，由参数和门控单元系统组成。它被用来判断信息是否有用。每个门控单元系统包括三个门，即输入门、遗忘门和输出门。LSTM 使用一个单独的内存单元来记住长期依赖关系，它可以根据当前输入进行更新。在每个时间步，LSTM 用当前的输入和先前的记忆单元状态作为输入计算当前单元的状态。最终，我们得到了隐藏状态 (包含若干个时间步)。将编码器获取的隐藏状态作为解码器初始的状态，将编码器的输入获取解码器每个时刻的隐藏状态。将隐藏状态输入 Softmax 函数，获取输出是各个模板词的概率。基于 LSTM 的判别器和常规的 LSTM 相同，最终输出结果是一个向量表示样本是正常还是异常的概率。在最终的入侵检测阶段通过判别-重构分数进行异常检测。判别分数来源于判别器将异常样本和正常样本区分的能力。而重构分数是计算生成器生成的样本和实际测试样本的相似度，也就是用二者之间的残差来识别测试数

据中的异常。综合两个分数对样本进行异常检测。针对实时环境对时间性能的高度敏感，可以将使用的 LSTM 替换为 GRU，GRU 在计算性能大幅提高的情况下对准确率影响不大。

该网络的判别器D也使用堆叠 LSTM，该网络的生成器不需要过于复杂的网络结构，本文使用的判别器是两层的堆叠 LSTM，如上图所示。判别器D函数用于判断数据是否是真实数据，该模型的损失函数用于最大化$D(x)$并最小化$D(G(S))$，其中S是日志数据。在生成器经过足够的训练后，判别器无法区分原始样本和生成器重构样本。

4.3.2 竞争学习

该网络架构的训练思路和常规的生成对抗网络相同，生成器追求能生成欺骗判别器的样本，而判别器追求能识别正常样本还是判别器生成的异常样本。生成对抗网络的目标函数定义为：

$$\min_{G}\max_{D}V(D,G)=E_{x\sim p_{data}(x)}(\log D(x))+E_{z\sim p_{z}(z)}(\log(1-D(G(S))))$$

本文使用随机梯度下降法(Stochastic gradient descent, SGD)来训练网络模型。

为了衡量生成器和判别器的损失，本文定义两个损失函数，残差损失L_R和判别损失L_D。残差损失L_R用来衡量真实样本序列x和生成的重构样本$G(x)$的之间的差异度，也可以解释为重构损失[38]。将该损失定义为：

$$L_R(\theta_G)=\sum|x-G(x,z)|$$

判别损失用来表示拥有丰富中间特征的判别器的估计的判别器损失，即判别器的中间层$f(\bullet)$的输出用于表示输入序列的统计量。判别器损失定义为：

$$L_D(\theta_D)=\sum|f(x)-f(G(x,z))|$$

整体的损失函数定义为残差损失和判别损失的加权和：

$$L(\theta_D,\theta_G)=(1-\gamma)\cdot L_R(\theta_G)+\gamma\cdot L_D(\theta_D)$$

其中L_R用来衡量生成的日志序列和真实日志之间的相似度。L_D表示生成的日志序列和真实的日志序列的分类误差。L_R和L_D分别用来通过反向传播更新θ_G和θ_D参数。

4.3.3 排列模型

因为在实际场景中存在不确定的网络延迟和日志的高并发写入，因此无法保证日志实际上按序到达。我们使用排列事件建模来提高检测的效率。系统日志的时间序列包含了足够的信息，这些信息对于检测异常非常重要。然而，LSTM 对模式中日志的时间顺序极其敏感，换句话说，LSTM 倾向于将观测样本的新排列

视为未观测样本。为了克服 LSTM 的这一局限性，本文使用了一种排列事件建模策略[95]。该策略将最频繁的模式的后续事件作为排列模式基线的后续事件，而该模式的后续事件这些模式和后续事件和最频繁模式的后续事件之间的差异构成。这种方式需要假设给定的日志序列经过排列模型处理后仍然会得到一个正常的日志序列。由于目前大多数大规模系统中正常样本与异常样本之间存在不平衡现象，该假设在大多数情况下是成立的。

另一方面，异常在数据集中的占比非常少，因此发现那些没有被观察到的异常比发现那些已经被观察到的异常更加重要。不过，在经过排列方式扩充后日志序列包括了很多信息来判别之前未被发现的异常。由于神经网络推理能力不足，模型很容易遭受对抗样本攻击。实验结果显示，该模型表现很大程度上取决于训练数据集质量。所以，在基于置换事件建立模型之后，产生更有价值样本，并给予模型有价值知识可以提升发现没有被观察到异常和鉴别没有被观察到正常事件表现。

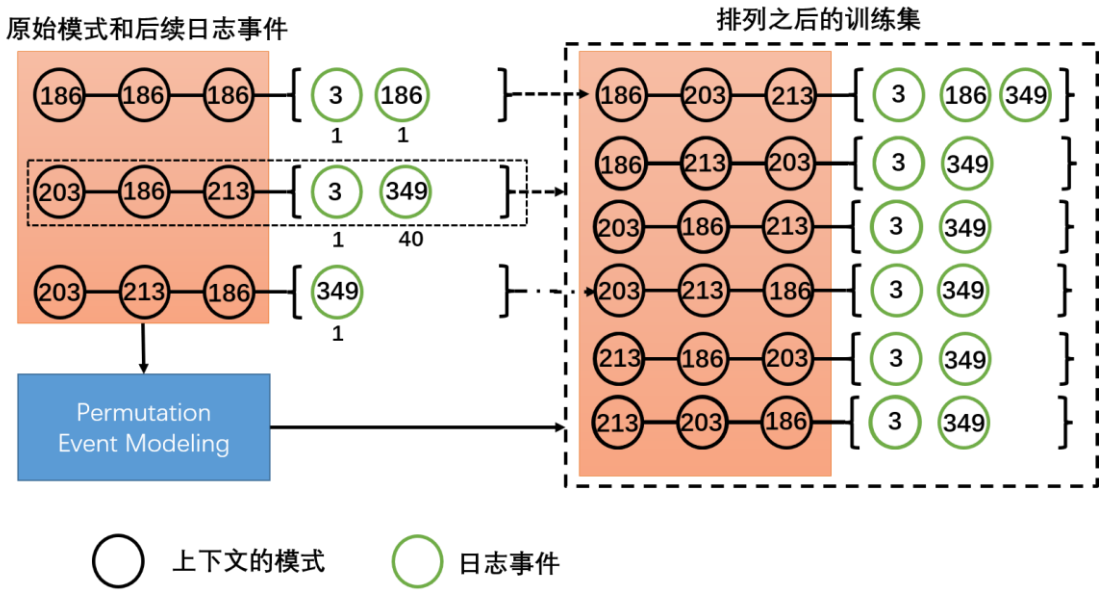

图 4-6 排列事件建模的机制

4.3.4 异常检测

异常检测需要综合判别器和生成器的误差。在日志时间序列数据的异常检测阶段，整个网络模型将测试集中的每个时间窗口内的序列x判别为正常(即源自一般数据分布)或异常(即偏离一般数据分布)观测值。

在对抗训练过程中，生成器G学习真实数据分布空间的数据分布p_{data}和针对数据分布重构模式，即每一个生成的小日志序列$G(x,z)$都是基于X的一般数据分布生成的。在模型每次更新迭代时，损失函数L评估了生成日志序列和真实日志序列x的相异度。可以直接从L推导出表示给定x进行重构对一般数据分布(即

真实数据分布)拟合的异常分数$A(x)$:

$$A(x) = (1-\gamma)\cdot R(x) + \gamma \cdot D(x)$$

其中$A(x)$是设定的异常分数，x是输入的日志实体。在映射过程到潜在空间的最终更新迭代时，残差分数$R(x)$和判别分数$D(x)$分别由残差损失$L_R(\theta_G)$和判别损失$L_D(\theta_D)$定义。参数γ是由经验决定的一个系数。当异常分数$A(x)$较大时表明该日志序列被判别为异常序列，而异常分数较小时表明通过竞争学习一个日志序列通过生成器G学到了真实数据x的一般分布。我们选择阈值ρ，如果$A(x)$小于ρ，则输入日志正常，否则判定为异常。本文的日志异常检测总体框架如图 4-7 所示。

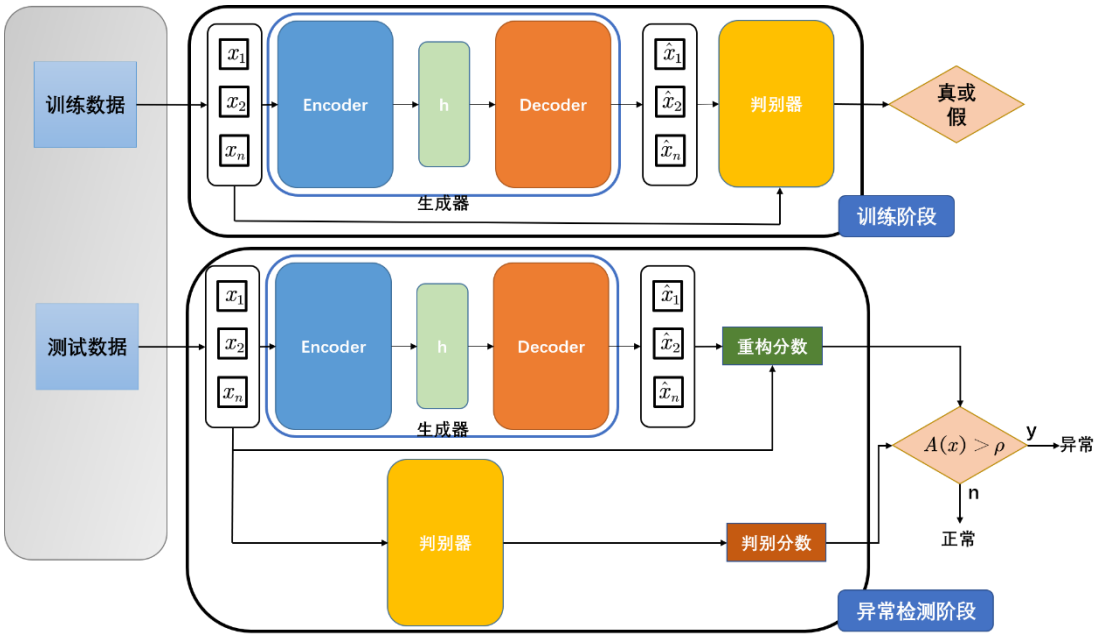

图 4-7 日志异常检测总体架构

4.4 使用过程和结果分析

本节旨在通过设置相应实验，测试本文提出算法的性能，并和其他基线算法进行对比以分析本文提出算法的优缺点。

4.4.1 实验环境

本章实验使用的实验环境同 3.5.1 节。

4.4.1 实验设置

可用于实验的公开的大规模系统数据较为少见，因为这些数据涉及到系统中的某些敏感信息，自己收集的规模的数据集无法反映系统的真实情况。本论文使用 HDFS 数据集，该数据集是一个大规模的分布式系统日志数据集，关于数据集的描述见 2.5 节。本论文使用的实验指标见 2.6 节，包括准确率、召回率、F1 分

数。

本文首先测试了不同的网络结构，以找到性能最后的网络结构便于进行测试。本论文在测试中使用了生成对抗网络架构，其中判别器使用的网络架构是两层的 LSTM 模型，隐藏层使用 80 个神经单元。生成器使用堆叠 LSMT 自动编码器架构，编码器和解码器的隐藏层使用 25 个神经单元。学习率和阈值等参数的设定由更详细的实验得出。

4.4.3 算法性能

本文首先通过实验找到算法合适的学习率（lr）。本文使用控制变量法，在其他参数不变的情况下改变学习率，设置学习率的区间为[0.1, 0.0001]，生成等步长的 10 个不同的学习率进行测试。在本实验中使用数据集 10%的数据进行作为训练数据，另外 10%的数据作为测试数据，不同学习率的检测效果如图所示：

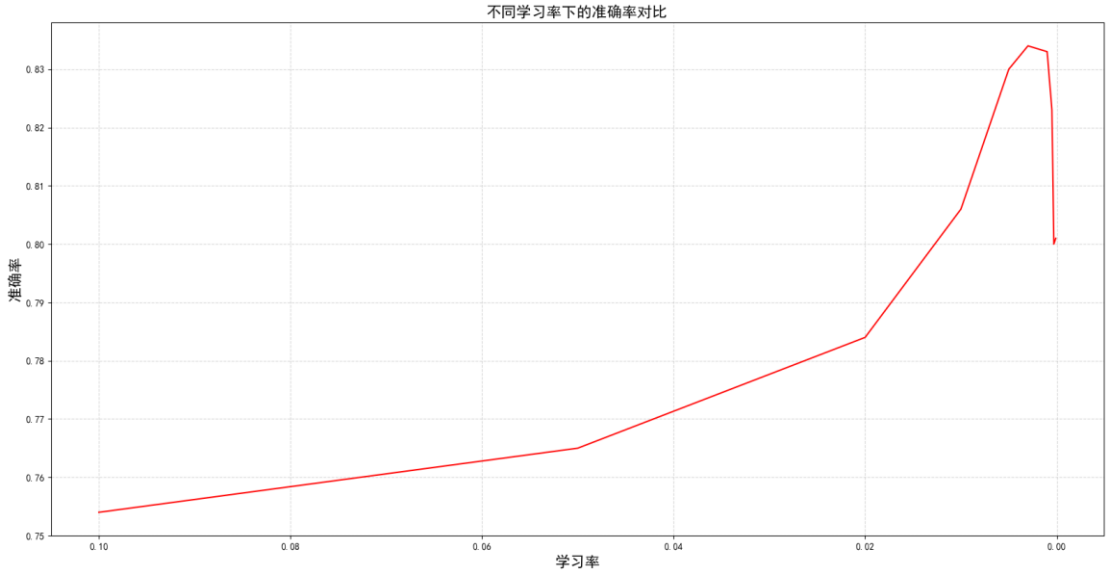

图 4-8 不同学习率对比

如果所示，本论文使用的算法受到学习率设定值的影响，随学习率从 0.1 的降低准确率先上升后下降。从图中可得出在学习率达到 0.001 时候准确率最高。因此本文在接下来的实验中使用 0.001 作为学习率。

另一个较为重要的参数是滑动窗口的大小。该参数表示每次进行训练或测试时一个日志时间序列的大小，每次滑动一定距离时滑动窗口内都会包含一定量的数据。本文测试了滑动窗口为 2 到 5 的情况下的正确率。本次实验使用 10%的数据作为训练集，另外 10%的数据作为测试集，不同窗口大小的检测效果如图 4-9 所示：

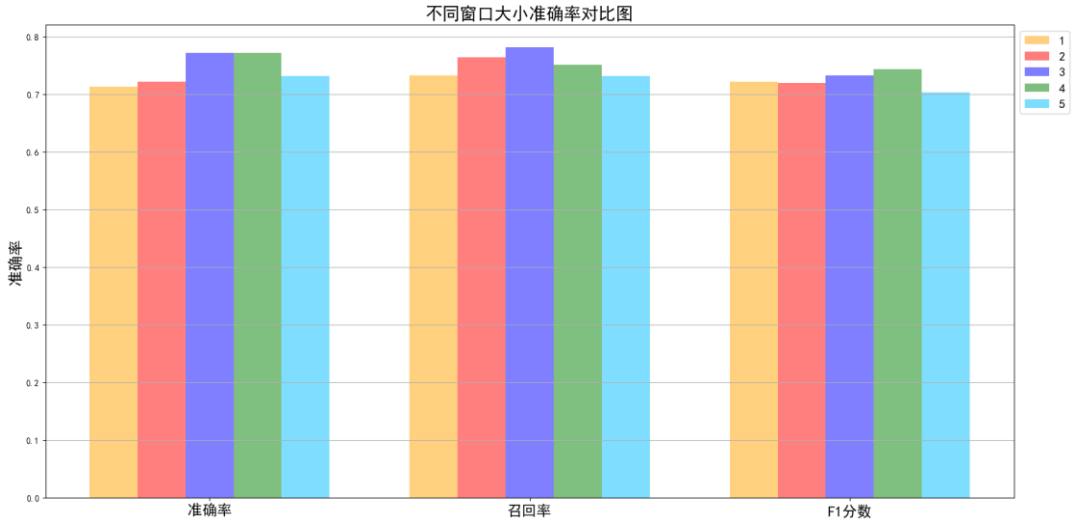

图 4-9 不同窗口大小对比

从图中可以看出不同的窗口大小对准确率、召回率和 F1 分数有影响，可以从图中得出当窗口大小为 1 的时候算法的性能较低，当窗口大小逐渐增大为 2，3，4 时性能逐渐上升，在窗口大小增大为 5 时性能下降。从准确率角度而言窗口大小为 3 和 4 的时候性能最后，从召回率角度当窗口大小为 3 的时候性能最好，当窗口大小为 4 时 F1 分数最高。

接下来测试阈值（ρ）对正确率的影响。阈值用来控制算法计算的异常分数被判定的为异常的区间。本文测试不同的阈值对实验正确率的影响。本文设置阈值从 0.2 到 0.6，比较不同阈值下正确率。本次实验使用 10%的数据作为训练集，另外 10%的数据作为测试集，不同阈值的检测效果如图 4-10 所示：

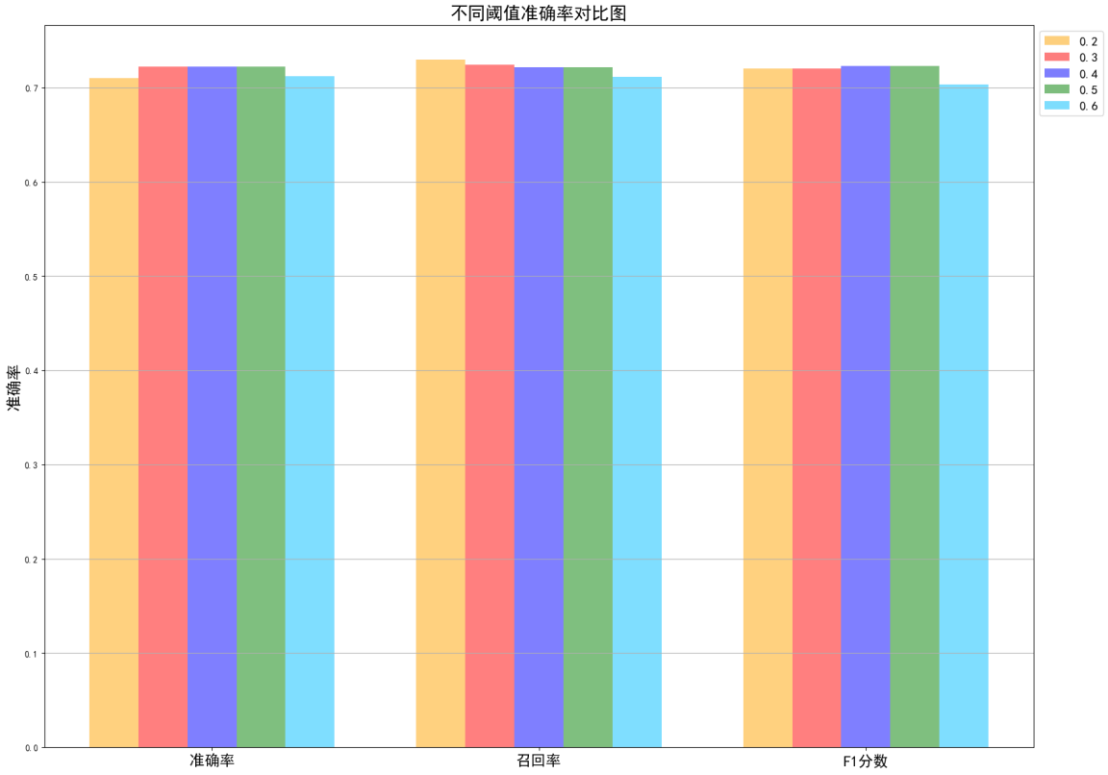

图 4-10 不同阈值对比

如图所示，阈值对正确率的影响影响较小，但是当正确率达到 0.6 的时候的性能略微弱于其他的阈值。从本实验可得，阈值的最佳选择在 0.2 到 0.5 左右，但是不能大于或等于 0.6，否则性能会下降。本文在接下来的实验中使用 0.4 作为阈值，超过该阈值则认为日志样本为异常。

首先将本文提出的算法和传统机器学习算法例如聚类、支持向量机（SVM）、决策树等和 PCA 等算法进行对比，比较准确率、召回率和 F1 分数。本文中的算法和其他算法的性能对比如图 4-11 所示：

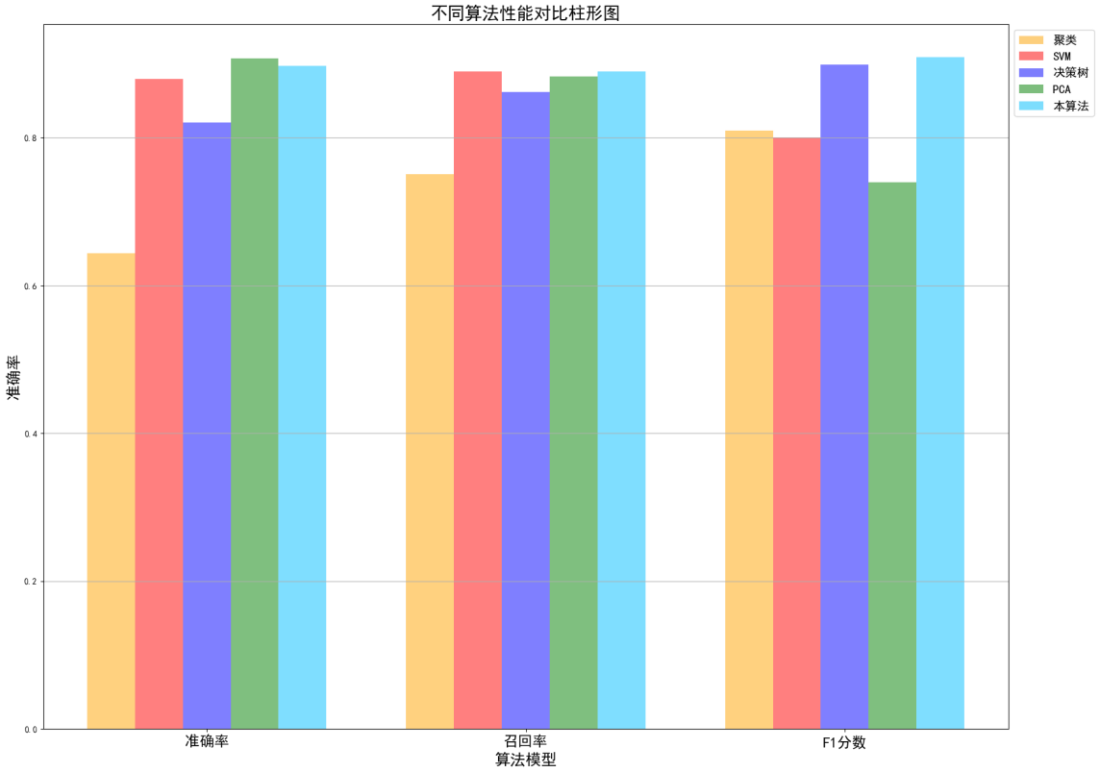

图 4-11 本论文算法和机器学习算法对比

从图中可以看出相较于其他机器学习算法，本论文性能较优。从准确率上本论文的性能仅次于 PCA 算法。从召回率上看，本论文的算法和 SVM 性能相当，超过了其他算法。在 F1 分数上，本论文的算法性能优于其他算法。

将本文提出的算法与其他使用生成对抗网络的算法如 DeepLog，Mad-GAN，TAnoGAN，LogGAN 算法性能进行对比，对比正确率。在本次实验中，使用 30%的数据集作为训练集，剩余数据作为测试集。本论文和其他算法对比如图 4-12：

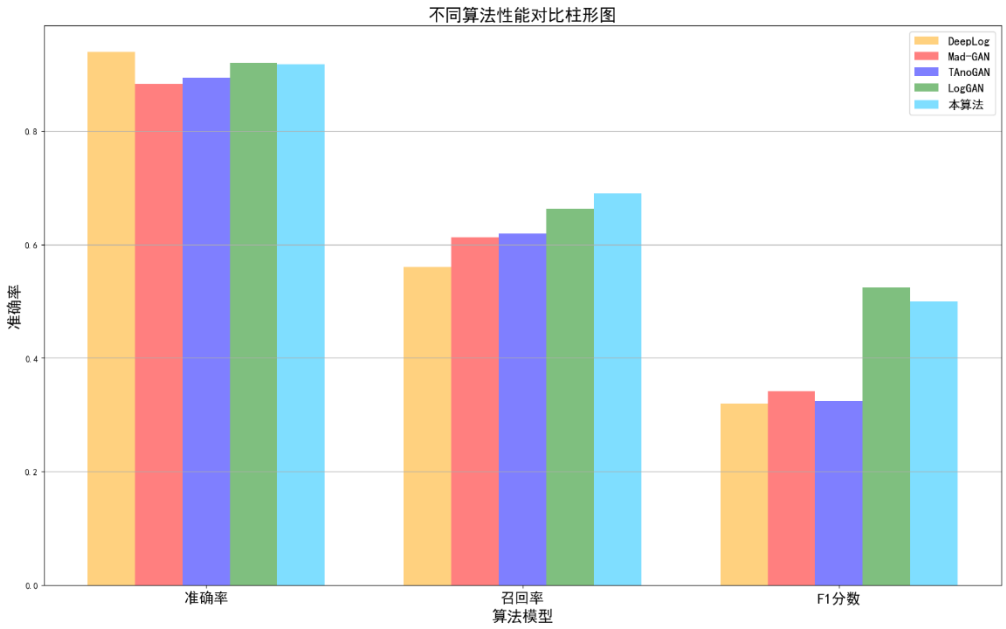

图 4-12 本论文算法和机器学习算法对比

从图中看出，本论文使用的算法的整体性能强于 Mad-GAN 和 TAnoGAN，但是弱于 DeepLog 和 LogGAN。在准确率上本论文的算法性能低于 DeepLog 和 LogGAN。本论文的算法在召回率上优于其他算法。

将本文提出的算法中的 LSTM 网络换成 GRU 网络，测试更改网络对算法的性能有什么影响。本文使用已经被处理好的日志数据分别输入原始模型和将 LSTM 替换为 GRU 的模型。对测试数据集进行 10 次重复的异常检测，取所有测试消耗时间的平均值。消耗时间对比如下表 4-3 所示：

表 4-3 LSTM 和 GRU 的性能对比

模型	总时间	平均时间	时间成本	准确率
LSTM	10024	1.114	1.00	0.9100
GRU	8077	1.013	0.90	0.9098

从表中可以看到本论文使用的算法使用 LSTM 和 GRU 作为神经网络模型，准确率相差 0.0002 时，基于 GRU 的模型相较于基于 LSTM 的时间性能提高了 10%。

4.6 本章小结

本章提出了一种基于生成对抗网络架构的日志异常检测算法。算法针对实际

场景中有标签的样本数量非常少且绝大多数的样本都是正常样本的条件，在训练时使用正常日志样本进行训练，在测试阶段使用正常或异常样本进行测试，可识别出异常样本。该算法首先对系统收集到的日志进行解析，使用的解析的算法效率较高，可以更高效的实现日志解析。然后进入异常检测阶段，本文提出生成对抗网络框架计算出日志数据的异常分数，以此为依据判断日志样本是否正常，如果日志异常分数超出阈值则认定数据为异常样本。本论文提出的算法相较于传统的日志算法在准确率方面性能较高。

第五章 总结与展望

5.1 总结

在现今互联网技术飞速发展的背景下，网络安全面对异常严峻挑战。因此，保护网络系统的安全是当今信息技术领域的研究重点之一。本文立足于现今复杂的网络环境，提出了基于互补生成对抗网络的入侵检测算法和基于 LSTM 的日志异常检测算法。本文的具体工作概况如下：

（1）首先对互联网发展和网络安全的严峻形式做了总结，揭示了入侵检测系统的重大意义。然后介绍入侵检测、异常检测、生成对抗网络，日志异常检测等邻域的发展情况。然后介绍了生成对抗网络、自动编码器、日志解析、时序模型等基础知识，并梳理了领域常用的各种数据集和常用的评价指标。

（2）提出基于互补生成对抗网络流量入侵检测算法。传统基于有监督学习的方式需要大量带标签的数据，但是在实际场景中很难获得对应的数据。而且使用带标签的数据只能学习到已知的入侵网络攻击类型，如果遇到之前从未出现的新型网络攻击传统的算法就无能为力了。将互补生成对抗网络引入到入侵检测领域可以很好地解决传统算法的问题。互补对抗生成网络可以生成位于样本空间以外互补空间的样本，用这类样本进行训练可以模拟网络攻击。以这种方式训练可以实现能够可以识别网络攻击的判别器，且具有较高的正确率。此外，本文使用的非对称堆叠自动编码器可以有效的识别数据中的异常样本。非对称自动编码器算法在入侵检测问题上有足够优秀的准确率，且模型省去了解码器部分，在计算性能要优于传统的自动编码器。

（3）提出基于 LSTM 的生成对抗网络的日志入侵检测算法。针对网络日志中隐藏可能包含网络攻击的隐藏信息，本论坛提出一种针对日志的异常检测算法。针对日志数据一类明显的时序数据，可以使用 LSTM 算法提取时序信息。其中生成器使用基于 LSTM 的自动编码器，根据该样本前的样本重构该样本，而判别器使用 LSTM 判断当前样本是正常样本还是异常样本。在实际异常检测阶段，将样本同时输入生成器和判别器，综合重构误差和判别器判别结果计算一个异常分数，如果异常分数达到阈值则将该样本判断为异常。在训练阶段只需使用正常数据进行训练，在异常检测阶段可以识别出异常数据。对于在实际场景中时间要求较高的情况，同时训练一个由 GRU 构成的神经网络，因为 GRU 相对于 LSTM 结构简单计算更加快速，使用 GRU 构成的神经网络更有利于在实际场景中进行部署。该算法适用于序列数据的入侵检测，在训练阶段只使用正常数据训练，测试阶段可以识别出网络攻击。

5.2 展望

在未来的工作中，还可以从模型、数据集、系统等角度对本论文对本论文提出的算法进行改进。

（1） 本文提出算法在CICDS-2017数据集这种维数较多的数据集上的性能较弱，无法有效识别样本中的正常流量或是网络攻击。基于此，本文提出的算法还需要改进。本文提出的算法使用的全连接神经网络结构不足以有效的建模更为复杂的数据类型。因此还需要构建更为复杂的模型。因此，需要尝试将更复杂的网络应用到算法中的可能性，近年来一些较为知名的算法如Transformer在其他领域获得了较为显著的成就，可以尝试将相关算法应用到本文提出的算法中。

（2） 本文提出的日志异常检测算法在HDFS数据集上训练，该数据集包含了一部分网络攻击造成的异常数据，但是部分异常是机器故障等原因造成的。因此本文提出的算法无法区分识别出的异常是源于网络攻击还是机器故障、程序运行抖动等以为因素造成的异常。另一方面，日志提供的信息有限，这也会造成一些入侵行为不会出现在日志内，系统也就无法识别出该网络攻击。该问题的改进方式在于根据实际的入侵检测场景进行具体分析，从业务角度分析被判定为异常的数据并进行相应的处理。例如，某在日志中显示IP频繁登录，则该行为就可以被判定为洪泛攻击。通过对这类规则进行挖掘，形成一个攻击规则库，找出异常日志中涉及入侵检测的相关日志。

（3） 本文提出了两种算法基于互补生成对抗网络的是基于流量的入侵检测，而基于LSTM/GRU的生成对抗网络是基于主机的入侵检测。这两种算法目前是彼此独立的，相互之间没有联系。在未来的研究中考虑将两种算法结合，构造完整的入侵检测系统。该系统分为在线入侵检测和离线入侵检测两部分。在线部分使用互补生成对抗网络算法，用于实时的处理进行系统中的网络流量数据，发现其中的网络攻击。而基于LSTM/GRU的生成对抗网络算法在于离线的扫描系统生成的日志。两种算法实现有机结合，相互协同发掘入侵行为。

参考文献

[1] 中国互联网中心．第 50 次中国互联网络发展状况统计报告［R］．http://cnnic.cn/n4/2022/0916/c38-10594.html

[2] 中国互联网应急中心．2020 年中国互联网网络安全报告[R]. https://www.cert.org.cn/publish/main/46/2021/20210721130944504525772/20210721130944504525772_.html

[3] 瑞星. 2021 年中国网络安全报告[R]. 北京：北京瑞星网安技术股份有限公司，2022.

[4] 李明昭. 基于机器学习的入侵检测技术研究[D]. 吉林大学, 2020.

[5] Garcia-Teodoro P , Diaz-Verdejo J , Macia-Fernandez G , et al. Anomaly-based network intrusion detection: Techniques, systems and challenges[J]. Computers & Security, 2009, 28(1-2):18-28.

[6] 侯海霞. 基于深度学习的入侵检测方法和模型[D].北京邮电大学,2021.

[7] 杨焱青. 基于深度生成模型的网络攻击检测技术研究[D].北京邮电大学,2021.

[8] Kuang F, Xu W, Zhang S. A novel hybrid KPCA and SVM with GA model for intrusion detection[J]. Applied Soft Computing, 2014, 18: 178-184.

[9] Li W, Yi P, Wu Y, et al. A new intrusion detection system based on KNN classification algorithm in wireless sensor network[J]. Journal of Electrical and Computer Engineering, 2014, 2014.

[10] Farnaaz N, Jabbar M A. Random forest modeling for network intrusion detection system[J]. Procedia Computer Science, 2016, 89: 213-217.

[11] Ingre B, Yadav A. Performance analysis of NSL-KDD dataset using ANN[C]//2015 international conference on signal processing and communication engineering systems. IEEE, 2015: 92-96.

[12] Javaid A, Niyaz Q, Sun W, et al. A deep learning approach for network intrusion detection system[C]//Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS). 2016: 21-26.

[13] Tang T A, Mhamdi L, McLernon D, et al. Deep learning approach for network intrusion detection in software defined networking[C]//2016 international conference on wireless networks and mobile communications (WINCOM). IEEE, 2016: 258-263.

[14] Azeez N A, Ayemobola T J, Misra S, et al. Network intrusion detection with a hashing based apriori algorithm using Hadoop MapReduce[J]. Computers,

2019, 8(4): 86.

[15] Xia X, Pan X, Li N, et al. GAN-based anomaly detection: a review[J]. Neurocomputing, 2022.

[16] Muruti G, Rahim F A, bin Ibrahim Z A. A survey on anomalies detection techniques and measurement methods[C]//2018 IEEE conference on application, information and network security (AINS). IEEE, 2018: 81-86.

[17] Lee K, Lee K, Lee H, et al. A simple unified framework for detecting out-of-distribution samples and adversarial attacks[J]. Advances in neural information processing systems, 2018, 31.

[18] Perera P, Patel V M. Learning deep features for one-class classification[J]. IEEE Transactions on Image Processing, 2019, 28(11): 5450-5463.

[19] Li H, Li Y. Anomaly detection methods based on GAN: a survey[J]. Applied Intelligence, 2022: 1-23.

[20] Hassanzadeh R, Nayak R. A semi-supervised graph-based algorithm for detecting outliers in online-social-networks[C]//Proceedings of the 28th annual ACM symposium on applied Computing. 2013: 577-582.

[21] Zhao Z, Mohan C, Mehrotra K. Adaptive sampling and learning for unsupervised outlier detection[C]//The Twenty-Ninth International Flairs Conference. 2016.

[22] Chae Y. Representing Stastical Network-Based Anomaly Detection by Using Trust[M]. University of Rhode Island, 2017.

[23] Moghaddam A H, Moghaddam M H, Esfandyari M. Stock market index prediction using artificial neural network[J]. Journal of Economics, Finance and Administrative Science, 2016, 21(41): 89-93.

[24] Heckerman D. Bayesian networks for data mining[J]. Data mining and knowledge discovery, 1997, 1: 79-119.

[25] Dang T T, Ngan H Y T, Liu W. Distance-based k-nearest neighbors outlier detection method in large-scale traffic data[C]//2015 IEEE International Conference on Digital Signal Processing (DSP). IEEE, 2015: 507-510.

[26] El Attar A, Khatoun R, Lemercier M. Clustering-based anomaly detection for smartphone applications[C]//2014 IEEE network operations and management symposium (NOMS). IEEE, 2014: 1-4.

[27] Goodfellow I, Pouget-Abadie J, Mirza M, et al. Generative adversarial nets [J]. Advances in Neural Information Processing Systems, 2014: 2672-2680.

[28] Arjovsky M, Chintala S, Bottou L. Wasserstein generative adversarial networks [C]//International conference on machine learning. PMLR, 2017: 214-223.

[29] Gulrajani I, Ahmed F, Arjovsky M, et al. Improved training of wasserstein GANs [C]//Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017: 5769-5779.

[30] Qi G J. Loss-sensitive generative adversarial networks on lipschitz densities [J]. International Journal of Computer Vision, 2020, 128(5): 1118-1140.

[31] Guo X, Hong J, Lin T, et al. Relaxed Wasserstein with applications to GANs [C]//ICASSP 2021-2021 IEEE International Conference on coustics, Speech and Signal Processing (ICASSP). IEEE, 2021: 3325-3329.

[32] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks [J]. Advances in Neural Information Processing Systems, 2012, 25: 1097-1105.

[33] Ghosh A, Kulharia V, Namboodiri V P, et al. Multi-agent diverse generative adversarial networks [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 8513-8521.

[34] Huang X, Li Y, Poursaeed O, et al. Stacked generative adversarial networks [C]//CVPR. 2017, 2: 3.

[35] Borji A. Pros and cons of gan evaluation measures [J]. Computer Vision and Image Understanding, 2019, 179: 41-65.

[36] Bousmalis K, Silberman N, Dohan D, et al. Unsupervised pixel-level domain adaptation with generative adversarial networks [C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 3722-3731.

[37] Li D, Chen D, Jin B, et al. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks[C]//International conference on artificial neural networks. Springer, Cham, 2019: 703-716.

[38] Bashar M A, Nayak R. TAnoGAN: Time series anomaly detection with generative adversarial networks[C]//2020 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2020: 1778-1785.

[39] Duan X, Ying S, Yuan W, et al. A generative adversarial networks for log anomaly detection[J]. Computer Systems Science and Engineering, 2021, 37(1): 135-148.

[40] Du M, Li F, Zheng G, et al. Deeplog: Anomaly detection and diagnosis from system logs through deep learning[C]//Proceedings of the 2017 ACM SIGSAC conference on computer and communications security. 2017: 1285-1298.

[41] Risto Vaarandi, Mauno Pihelgas. 2015. A data clustering algorithm for mining patterns from event logs[C]. The 11th International Conference on

Network and Service Management (CNSM), Barcelona, Spain, 2015: 1-7.

[42] Hamooni H, Debnath B, Xu J, et al. Logmine: Fast pattern recognition for log analytics[C]//Proceedings of the 25th ACM International on Conference on Information and Knowledge Management. 2016: 1573-1582.

[43] Min Du and Feifei Li. 2011. Spell: Online Streaming Parsing of Large Unstructured System Logs[J]. In IEEE Transactions on Knowledge and Data Engineering, 2011, 31(11): 2213-2227.

[44] Jiang Z M, Hassan A E, Flora P, et al. Abstracting execution logs to execution events for enterprise applications (short paper)[C]//2008 The Eighth International Conference on Quality Software. IEEE, 2008: 181-186.

[45] Schlegl T, Seeböck P, Waldstein S M, et al. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery[C]//Information Processing in Medical Imaging: 25th International Conference, IPMI 2017, Boone, NC, USA, June 25-30, 2017, Proceedings. Cham: Springer International Publishing, 2017: 146-157.

[46] Xie Q, Dai Z, Du Y, et al. Controllable invariance through adversarial feature learning[J]. Advances in neural information processing systems, 2017, 30.

[47] Zenati H, Foo C S, Lecouat B, et al. Efficient gan-based anomaly detection[J]. arXiv preprint arXiv:1802.06222, 2018.

[48] Zenati H, Romain M, Foo C S, et al. Adversarially learned anomaly detection[C]//2018 IEEE International conference on data mining (ICDM). IEEE, 2018: 727-736.

[49] Li C, Liu H, Chen C, et al. Alice: Towards understanding adversarial learning for joint distribution matching[J]. Advances in neural information processing systems, 2017, 30.

[50] Zhang H, Goodfellow I, Metaxas D, et al. Self-attention generative adversarial networks[C]//International conference on machine learning. PMLR, 2019: 7354-7363.

[51] Haloui I, Gupta J S, Feuillard V. Anomaly detection with Wasserstein GAN[J]. arXiv preprint arXiv:1812.02463, 2018.

[52] Schlegl T, Seeböck P, Waldstein S M, et al. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks[J]. Medical image analysis, 2019, 54: 30-44.

[53] Hendrycks D, Mazeika M, Kadavath S, et al. Using self-supervised learning can improve model robustness and uncertainty[J]. Advances in neural information processing systems, 2019, 32.

[54] Golan I, El-Yaniv R. Deep anomaly detection using geometric transformations[J]. Advances in neural information processing systems, 2018, 31.

[55] Bashar M A, Nayak R. TAnoGAN: Time series anomaly detection with generative adversarial networks[C]//2020 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2020: 1778-1785.

[56] Li Y, Peng X, Zhang J, et al. DCT-GAN: dilated convolutional transformer-based gan for time series anomaly detection[J]. IEEE Transactions on Knowledge and Data Engineering, 2021.

[57] Geiger A, Liu D, Alnegheimish S, et al. Tadgan: Time series anomaly detection using generative adversarial networks[C]//2020 IEEE International Conference on Big Data (Big Data). IEEE, 2020: 33-43.

[58] Maru C, Kobayashi I. Collective anomaly detection for multivariate data using generative adversarial networks[C]//2020 International Conference on Computational Science and Computational Intelligence (CSCI). IEEE, 2020: 598-604.

[59] He S, Zhu J, He P, et al. Experience report: System log analysis for anomaly detection[C]//2016 IEEE 27th international symposium on software reliability engineering (ISSRE). IEEE, 2016: 207-218.

[60] 杨瑞朋. 日志异常检测与诊断关键技术研究[D]. 战略支援部队信息工程大学, 2020.

[61] Chen M, Zheng A X, Lloyd J, et al. Failure diagnosis using decision trees[C]//International Conference on Autonomic Computing, 2004. Proceedings. IEEE, 2004: 36-43.

[62] Lin Q, Zhang H, Lou J G, et al. Log clustering based problem identification for online service systems[C]//2016 IEEE/ACM 38th International Conference on Software Engineering Companion (ICSE-C). IEEE, 2016: 102-111.

[63] Farshchi M, Schneider J G, Weber I, et al. Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis[C]//The 26th International Symposium on Software Reliability Engineering. IEEE, 2015: 24-34.

[64] 王易东,刘培顺,王彬. 基于深度学习的系统日志异常检测研究[J]. 网络与信息安全学报, 2019, 5(5): 105-118.

[65] Shone N, Ngoc T N, Phai V D, et al. A deep learning approach to network intrusion detection[J]. IEEE transactions on emerging topics in computational intelligence, 2018, 2(1): 41-50.

[66] Wen L, Gao L, Li X. A new deep transfer learning based on sparse auto-encoder for fault diagnosis[J]. IEEE Transactions on systems, man, and cybernetics: systems, 2017, 49(1): 136-144.

[67] Graves A, Graves A. Long short-term memory[J]. Supervised sequence labelling with recurrent neural networks, 2012: 37-45.

[68] Ghimire S, Deo R C, Wang H, et al. Stacked LSTM sequence-to-sequence autoencoder with feature selection for daily solar radiation prediction: a review and new modeling results[J]. Energies, 2022, 15(3): 1061.

[69] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation[J]. arXiv preprint arXiv:1406.1078, 2014.

[70] Ferrag M A, Maglaras L, Moschoyiannis S, et al. Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study[J]. Journal of Information Security and Applications, 2020, 50: 102419.

[71] KDD99 数据集[EB/OL]. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, 2022-12-15.

[72] Sharafaldin I, Lashkari A H, Ghorbani A A. Toward generating a new intrusion detection dataset and intrusion traffic characterization[J]. ICISSp, 2018, 1: 108-116.

[73] CICDS2017 数据集[EB/OL]. https://www.unb.ca/cic/datasets/ids-2017.html, 2022-12-15

[74] Gharib A, Sharafaldin I, Lashkari A H, et al. An evaluation framework for intrusion detection dataset[C]//2016 International Conference on Information Science and Security (ICISS). IEEE, 2016: 1-6.

[75] Sharafaldin I, Habibi Lashkari A, Ghorbani A A. A detailed analysis of the cicids2017 data set[C]//Information Systems Security and Privacy: 4th International Conference, ICISSP 2018, Funchal-Madeira, Portugal, January 22-24, 2018, Revised Selected Papers 4. Springer International Publishing, 2019: 172-188.

[76] Tavallaee M, Bagheri E, Lu W, et al. A detailed analysis of the KDD CUP 99 data set[C]//2009 IEEE symposium on computational intelligence for security and defense applications. Ieee, 2009: 1-6.

[77] Moustafa N, Slay J, Creech G. Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks[J]. IEEE Transactions on Big Data, 2017, 5(4): 481-494.

[78] Nehinbe J O. A simple method for improving intrusion detections in corporate networks[C]//Information Security and Digital Forensics: First International

Conference, ISDF 2009, London, United Kingdom, September 7-9, 2009, Revised Selected Papers 1. Springer Berlin Heidelberg, 2010: 111-122.

[79] CAIDA Dataset[EB/OL]. https://www.caida.org/data/overview/

[80] Homoliak I, Barabas M, Chmelar P, et al. ASNM: Advanced security network metrics for attack vector description[C]//Proceedings of the International Conference on Security and Management (SAM). The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2013: 1.

[81] Jazi H H, Gonzalez H, Stakhanova N, et al. Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling[J]. Computer Networks, 2017, 121: 25-36.

[82] He S, Zhu J, He P, et al. Loghub: a large collection of system log datasets towards automated log analytics[J]. arXiv preprint arXiv:2008.06448, 2020.

[83] Liang Y, Zhang Y, Sivasubramaniam A, et al. Filtering failure logs for a bluegene/l prototype[C]//2005 International Conference on Dependable Systems and Networks (DSN'05). IEEE, 2005: 476-485.

[84] Liu Y, Li Z, Zhou C, et al. Generative adversarial active learning for unsupervised outlier detection[J]. IEEE Transactions on Knowledge and Data Engineering, 2019, 32(8): 1517-1528.

[85] 王震. 基于距离的离群点检测算法分析与研究[D].重庆大学,2011.

[86] 赵新想. 基于密度的局部离群点检测算法的研究与改进[D].华中师范大学,2014.

[87] Zheng P, Yuan S, Wu X, et al. One-class adversarial nets for fraud detection[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33(01): 1286-1293.

[88] Dai Z, Yang Z, Yang F, et al. Good semi-supervised learning that requires a bad gan[J]. Advances in neural information processing systems, 2017, 30.

[89] Rumelhart D E, Hinton G E, Williams R J. Learning internal representations by error propagation[R]. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[90] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks[J]. science, 2006, 313(5786): 504-507.

[91] Imran M, Haider N, Shoaib M, et al. An intelligent and efficient network intrusion detection system using deep learning[J]. Computers and Electrical Engineering, 2022, 99: 107764.

[92] Wang W, Du X, Shan D, et al. Cloud intrusion detection method based on stacked contractive auto-encoder and support vector machine[J]. IEEE transactions on

cloud computing, 2020, 10(3): 1634-1646.

[93] Liu C, Yang J, Wu J. Web intrusion detection system combined with feature analysis and SVM optimization[J]. EURASIP Journal on Wireless Communications and Networking, 2020, 2020: 1-9.

[94] Zhang L, Xie X, Xie K, et al. An efficient log parsing algorithm based on heuristic rules[C]//Advanced Parallel Processing Technologies: 13th International Symposium, APPT 2019, Tianjin, China, August 15–16, 2019, Proceedings 13. Springer International Publishing, 2019: 123-134.

[95] Xia B, Bai Y, Yin J, et al. Loggan: a log-level generative adversarial network for anomaly detection using permutation event modeling[J]. Information Systems Frontiers, 2021, 23: 285-298.

致 谢

我很荣幸能在北京邮电大学校园的秋天遇见李丽香老师。李老师作为硕士生导师，工作勤奋、认真负责、一丝不苟地指导我和其他硕士生，在课题组讨论中总能提出有效的建议，并能引导我们在该领域深入探索和学习。李老师不仅是我们的导师，也是我们的朋友，在科研之外，在生活上、精神上也都给予了我们无限的支持，近三年来的相处让我收获很多，在李老师身上看到了很多教书育人的美德和与人相处的艺术。另外，彭海朋教授也给了我很多帮助，我敬佩彭教授对科研的热情，敬佩彭教授在实验室管理中严谨细致，敬佩彭教授在学术上的卓越成就，他不仅在白天指导我的论文，还会在深夜发消息给我提出有建设性的意见和方向。

在研究生阶段，我有幸与来自五湖四海的实验室同学们并肩作战，共同经历了三年的风雨历程。他们是我的好伙伴，我们一起探讨课题、技术、文献，一起充实我们的知识库。他们是我的好师长，他们带我熟悉科研和生活的方方面面，倾听我的困惑，分享他们的经验。他们是我的好师弟妹，他们让我体会到了传道授业的乐趣，我们一起学习新知识，一起外出旅游，一起放松心情。

我很荣幸能够跟着课题组参与了很多学术活动，看到了许多科研人员的努力和成果，让我感受到了技术的快速进步，学术会议、讲座等各种形式的活动让我更接近技术，无论是线下还是线上，都能拓宽我的视野，激发我的创意，对我的学术研究有很大的帮助。作为一名硕士研究生，了解和掌握专业领域的前沿知识是必不可少的工作，通过参加几次学术活动，我对专业领域有了一个大概的认识，确定了自己喜欢的研究方向，并且结交了很多学术好友。

我很幸运有一群支持我的家人，在此感谢父母的理解和安慰，感谢所有朋友和实验室同学的陪伴和鼓励。虽然你们不太明白我的专业知识，但是你们给我的支持非常珍贵，在我遇到困难和挫折时，总能及时向你们倾诉。我的每一份成就都离不开你们。

有太多太多幸运充满了我的研究生生活，在这里无法一一道谢，请相信它们都在我的心里温暖着我每一个清晨和黄昏。同时也希望未来的日子依然光芒万丈、精彩纷呈！参与项目和攻读学位期间获得的研究成果目录

一、发表论文情况

[1] 王慧, 王励成, 柏雪,等. 区块链隐私保护和扩容关键技术研究[J]. 西安电子科技大学学报, 2020, 47(5):12.

[2] 王慧, 王励成, 刘清华. 面向银行 KYC 业务的区块链存储与共享方案[EB/OL]. 北京：中国科技论文在线 [2022-03-25].