# 第 1 章　引言

## 1.1　研究背景

网络自诞生以来，随着数十年的发展，已经成为了最重要的信息化基础设施之一。如图 1.1所示，第 46 次《中国互联网络发展状况统计报告》[1] 指出，截至 2020 年 6 月, 我国的移动网民用户规模已经累计达到 9.40 亿, 互联网的网络普及率已经高达 67.0%, 互联网的主要应用领域覆盖范围已经达到包括即时移动通信、搜索结果引擎、网络即时新闻、线下线上教育、购物以及出行等各个方面, 可以说如今互联网已经和每一个人的工作日常生活息息相关密不可分。

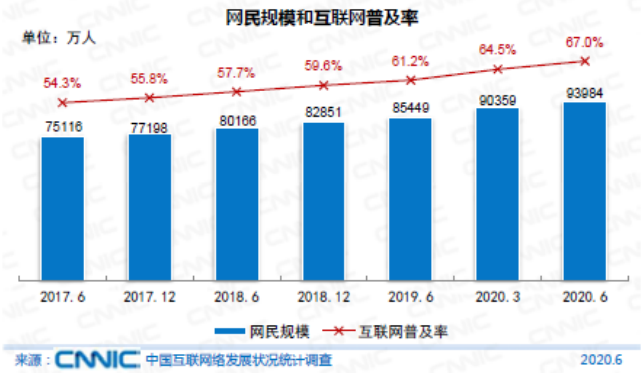


图 1.1　网民规模和互联网普及率

随着网络重要性的提升、用户规模的膨胀, 管理网络的难度也越来越大。网络系统就是一个繁琐而又复杂的网络, 它的设计、运营以及维护都必须依靠专业的网络运维技术人员。早期的网络运维管理工作绝大多数都是由运维工作人员手工操作来完成, 此后人们逐渐发现一些重复性的工作可以用自动化脚本来实现, 于是诞生了自动化运维。自动化的运维系统可以认为是一种建立在专家经验和人为制订规则的系统之上。但是随着移动互联网的规模迅速增长和巨大扩展, 以及其服务种类和方式的日益多样化, 由人们自主设定的基础运维规则当中的方法与技术，渐渐难以应付企业的大规模运维需求。在此情况下，智能运维应运而生。它与自动化运维仰仗专门人员的知识、人为制定规则有别，智能运维的着重点在于使用新颖的机器学习算法，先从庞大的运维管理数据中反复吸收经验进行学习，进而有条不紊的逐步提取规则。

异常检测是智能运维的关键环节，具有至关重要的意义。从网络故障管理的角度来说，做好异常检测可以提前预测故障的发生；从性能管理的角度来说，可以发现性能不佳的区域，避免因误配置、架构不合理导致性能下降；从安全管理的角

度来说，在网络攻击的前期阶段，及时发现并预警后续攻击，进而做出防御措施。因此，在复杂的网络环境中甄别出有效和异常流量尤为重要，在重大事故发生前，根据各项流量特征的变化，提前预测出即将发生的事故，提高应急响应速度，防患于未然。

本文以校园网为例，进行网络流量异常检测算法和系统的研究。清华大学校园网是全球规模最大，架构最复杂，流量场景最多变的校园网之一，具有以下几个特点：

1. 用户规模大，流量峰值高。每天有 10 万台设备活跃在线，同时在线设备最多为 7 万台，峰值流量约为 30Gbps/s，如图 1.2所示。
2. 用户应用类型多，如图 1.3所示，清华大学校园网中的用户类型远比一般的企业网复杂，在网络环境中几百种应用同时使用，这给数据分析带来了很多困难。
3. 异常流量是常态。扫描流量、攻击流量占比多。

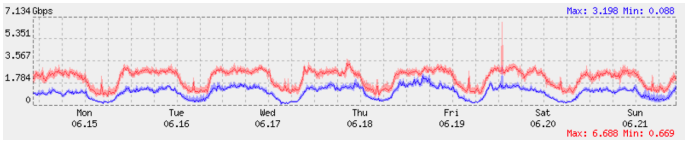


图 1.2　用户流量变化图

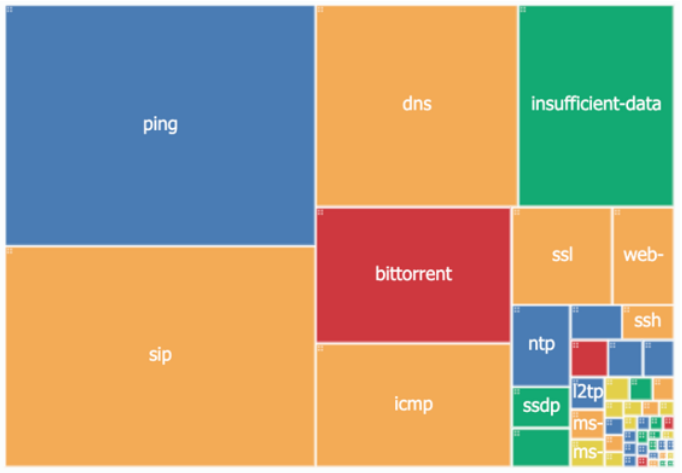


图 1.3　用户应用类型图

## 1.2　论文的研究内容

围绕 1.1 节中提到的清华大学校园网流量异常检测的特点和挑战，本文分别展开了以下三方面的研究：

1. 本文分析对比了流量异常检测领域的两个经典开源数据集 UNSW-NB15 和

CICIDS2017，引入了清华大学校园网的真实数据集，并且分别在这几个数据集上对现有的异常检测算法进行了实验对比。

2. 在现有算法的基础上，本文将特征关系引入到神经网络的训练中，提出了一种基于特征之间关系图的循环神经网络算法 (FG-RNN，Feature Graph-Recurrent Neural Network)。在复杂的网络流量环境下，流量特征种类繁多，且特征之间的相关性会随着流量变化而变化。原有的异常检测算法通常直接利用提取的特征进行训练，本文提出的 FG-RNN 算法有效利用了特征之间的相关性信息，将其加入到神经网络的训练过程中。
3. 本文对基于特征之间关系图的循环神经网络算法进行了流式改造，使之能够满足当前实时异常检测的需求。为了应对海量的校园网流量数据和高速数据流，本文设计了一个基于 spark streaming 的实时异常检测系统。该系统分为输入模块、模型模块、检测模块三部分，输入模块负责将流量数据进行窗口划分、特征选择、特征抽取、合并计算，得到的特征矩阵与预训练得到的关系矩阵一同送到模型中进行训练，模型中的参数每 2 小时更新一次，最后由检测模块对当前流量进行判别，并给出异常流量的类别。

## 1.3　论文内容和组织结构

第一章为引言，主要介绍本文的研究背景以及本文的研究内容。

第二章是关于网络流量异常检测的相关工作综述。首先对网络流量异常的定义和分类进行了介绍；接下来着重介绍了基于不同原理的异常检测算法；然后在开源数据集上对部分经典算法进行了复现，对比了其优缺点；最后，指出了现有的异常检测算法在当前大规模流量环境下存在的主要问题。

第三章至第五章是本文的主要研究内容。第三章分析对比了开源数据集与校园网真实数据集，并且在多个数据集上对现有的算法进行了实验对比。第四章提出了基于特征关系图的循环神经网络算法，并进行了算法性能评估。第五章设计并实现了一个基于 spark streaming 的流式处理系统。

最后，第六章对全文进行了总结，并且提出了未来展望。

# 第 2 章　相关工作综述

## 2.1　引言

本章对网络流量异常检测领域的相关工作进行了综述。首先介绍了网络流量异常的定义和分类，异常检测系统的通用框架如图 2.1所示。

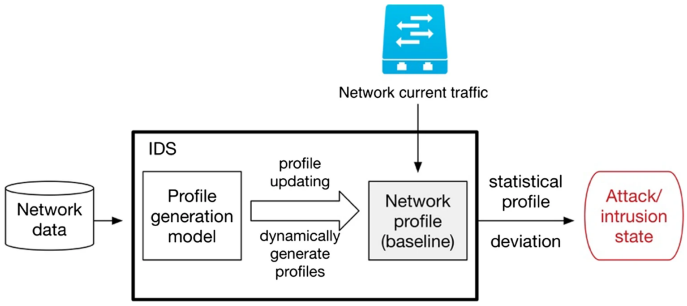


图 2.1　异常检测的通用框架

## 2.2　网络流量异常的定义和分类

Hawkins(1980)[2] 为“异常”下了一个定义: 他指出某些数据由于在数据集中表现出差异，因此值得怀疑这些数据不是一般的随机偏差，而是来自完全不同的机制，我们称这些数据为“异常”。例如在道路交通领域，某条道路的车流量突然增多，甚至多至堵塞，又或者突然减少，此时车流量数据就是一个异常。因此网络行为的异常就是指那些与正常的、标准的，我们所预期的行为相异的表现。为了检测网络异常，网络所有者必须有一个预期或正常行为的概念，我们称其为“基线”。要检测网络行为的异常，就需要持续监控网络中的意外趋势或事件，那些可能改变网络流量特征或者监控指标的恶意行为。

限于篇幅，本文只关注引起网络流量特征变化的恶意行为，而对于系统权限提升，缓冲区溢出等黑客攻击手段暂时不做研究。

网络流量异常具体有哪些类别，学术界没有统一的意见。本文中关注的网络流量异常按照产生意图分为恶意和非恶意两类，其中恶意行为主要有拒绝服务攻击、网络扫描、BGP 劫持、网络蠕虫、僵尸网络等；非恶意行为主要有物理故障、突发事件等。接下来我们对这些异常分别进行介绍:

- 拒绝服务攻击（Denial of Service，DoS）: 攻击者往往通过构造大量请求访问目标主机，使其无法处理正常的流量请求，导致正常用户被拒绝服务。
- 网络扫描: 攻击者在发动网络攻击之前，普遍会先进行网络扫描，以最低代

价寻找到可攻击的目标。因为黑客需要确定某个子网范围内哪些主机是活跃的，活跃主机上正在运行哪些存在漏洞的服务。因此，网络扫描通常是网络攻击的前奏。

- BGP 劫持：BGP 劫持是指通过一个自治系统错误宣称 IP 为其所有，使得使用边界网关协议维护的互联网路由表的路由器错误地将用户发送的数据传送给非数据传送目的地。BGP 劫持很像是攻击者修改了道路路口的路标，使得经过的车辆驶向错误的出口。
- 网络蠕虫：网络蠕虫通过网络和电子邮件进行复制和传播，它会扫描和攻击网络上存在系统漏洞的节点主机，通过局域网或者互联网从一个节点传播到另外一个节点。例如“熊猫烧香”及其变种就是蠕虫病毒。
- 僵尸网络：僵尸网络指由感染了恶意软件的计算机组成的网络，这些计算机集群由攻击者控制。
- 物理故障：物理故障是指路由器故障，链路破坏，线缆损坏，断电等不可预测的突发事件。
- 突发事件：突发事件是指引起网络瞬时拥塞的事件，有可能是管理员配置错误，也有可能是正常的网络操作，如某网站访问量激增。

## 2.3　网络流量异常检测算法

本节将介绍在异常检测领域主流的一些算法，根据所依赖的技术原理的不同，将这些算法分为了基于分类、基于统计、基于聚类、基于信息论的异常检测算法。

### 2.3.1　基于分类的异常检测算法

基于分类的方法依赖于建立知识库的正常流量活动特征，并将偏离基线特征的活动视为异常活动。这种方法的优势在于它们能够检测到完全新颖的攻击，假设这些攻击表现出大量的偏离正常基线的情况。需要注意的是，由于知识库中未包含的正常流量被认为是攻击，因此会产生无意中的误报。为了避免这种情况，异常检测技术需要进行训练，以建立正常的活动基线，这个建立基线的过程通常非常耗时，而且还取决于是否有完全正常的流量数据集。在实践中，获得无攻击的流量实例是非常罕见且昂贵的。此外，在如今信息更新变换快速的动态网络环境中，保持正常基线的更新是非常困难的。在现有的大量基于分类的网络异常检测技术中，我们主要讨论以下三种技术，支持向量机（Support Vector Machine，SVM），贝叶斯（Bayes）以及神经网络（Neural Network，NN）。

#### 2.3.1.1　SVM

Eskin et al.[3] 引入无监督 SVM 的概念来检测异常事件。常规的 SVM 的原理是推导出一个超平面，使得正类样本和负类样本之间的分离余量最大化，将特征空间中的两类数据进行分离。标准的 SVM 算法是一种监督学习方法，需要标记数据来创建分类规则。而该算法经过改进 SVM，试图将整个训练数据集从原点分离出来，找到一个以最大余量将数据实例与原点分离的超平面，

Hu et al.[4] 提出了一种忽略噪声数据的异常检测方法，该方法使用 Robust SVM(RSVM) 来开发。标准的 SVM 有一个主要假设，即所有的训练样本数据都是独立且相同分布的 (i.i.d)。但是在实际场景中，训练数据往往包含噪声，这就会导致标准 SVM 会学习出一个高度非线性的决策边界，从而导致通用性较差。有鉴于此，RSVM 以类中心的形式加入了平均化技术，使得决策面更加平滑。此外，RSVM 另一个优点是能大大降低支持向量的数量，从而减少运行时间，提高效率。

Taeshik Shon 等人[5] 提出了一种基于增强支持向量机 (Enhanced Support Vector Machines) 的异常检测算法。增强支持向量机是该文作者提出的一种新型的支持向量机，该向量机同时具备了传统支持向量机的高性能和单类支持向量机检测新异常的能力。在预处理阶段，检测算法使用数据包过滤器滤除畸形数据包。随后，检测算法使用遗传算法对数据包头的信息进行特征选择。接下来，检测算法利用 SOM 网络对正常流量进行聚类，并用这些聚类来训练增强支持向量机，从而得到最终的异常检测器。

#### 2.3.1.2　贝叶斯

朴素贝叶斯是一个简单的概率分类器，通常用于网络入侵检测问题。它将先验信息与样本信息结合起来，并以统计推论的方式进行实施，从而利用概率显示出各种形式的不确定性。朴素贝叶斯假设了所有输入属性在条件上都是彼此独立的。

Kruegel 等人[6] 假设异常检测系统包含许多模型，用于分析一个事件的不同特征。他们指出了在这种系统下异常检测技术造成高误报率的两个主要原因：一是异常检测系统通过将多个概率模型的输出进行汇总，而每个模型往往只给出一个事件的常态/异常的得分或概率，从而导致高误报率；二是异常检测系统无法处理那些不正常但合法的行为，如 CPU 利用率、内存使用率突然增高等。基于贝叶斯网络的概念，[6] 提出了一种解决上述问题的方法。对于一个输入事件的有序流 ($\boldsymbol{S} = e_1, e_2, e_3...$)，异常检测系统决策每个事件是正常还是异常。该决策基于 k 个模型 ($\boldsymbol{M} = m_1, m_2, ...m_k$) 的输出 ($o_i | i = 1, 2, ...k$) 和可能的附加信息 ($\boldsymbol{I}$)。应用贝叶

斯网络来识别异常事件，引入根节点，根节点代表一个具有两种状态的变量。一个子节点用于捕捉模型的输出，子节点与根节点相连，预计当输入异常或正常时，输出事件会有所不同。

#### 2.3.1.3　神经网络

神经网络已经被应用于各个应用领域，如图像和语音处理，但其对计算量的要求很高。神经网络对数据进行分类的优势也可被用于网络异常检测。在网络异常检测领域，神经网络通常会和其他技术进行结合，如统计方法。

Hawkins 等人[7] 提出了一个多层的前馈神经网络，该神经网络可以用来进行异常值的检测。这个多层前馈的神经网络 (multi-layer feed-forward neural networks) 也就是 Replicator Neural Networks。具体而言，它的运行方式是设置了三个隐藏层，夹在输入层和输出层的中间，目标是通过训练，使输出层能以最小的误差重现出输入的数据模式。其原理在于输入层与输出层节点的数量比隐藏层节点的数量要多，因此隐藏层能够具有压缩数据和恢复数据的作用。Replicator Neural Networks 的示意图如图 2.2所示。

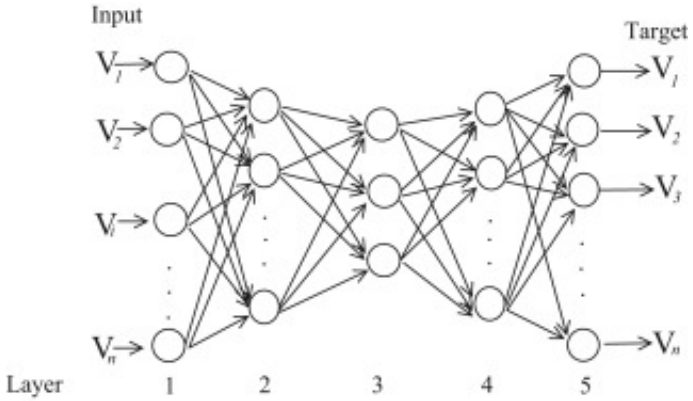


图 2.2　Replicator Neural Networks 示意图

Zhang[8] 提出了一种神经网络与统计模型相结合的分层入侵检测系统。将神经网络分类器的输出表示为一个连续变量 ($t$)，其中-1 表示有绝对把握的入侵，1 表示没有攻击。此外，自组织地图 (SOM) 被用于网络异常检测。Ramadas 等 (2003) 提出，利用 SOM，可以对网络流量进行实时分类。SOM 依赖于这样一个假设，即网络攻击可以由不同的神经元组来描述，这些神经元组与其他神经元组相比，在输出神经元图上覆盖更大的区域。Poojitha 等人[9] 开发了一个由反向传播算法训练的前馈神经网络，利用给定的数据集与计算机网络在正常和异常行为期间的相关信息来检测异常。

### 2.3.2　基于统计的异常检测算法

从前检测异常是使用假设-检验的方法，也就是要利用统计与概率模型。第一步是预测数据的分布，并从中找到预先认定的 “异常”，在此操作下总是使用极值分析或假设检验。也就是说，当前存在基础的一维数据时，首先拟定数据是遵守正态分布的，若有超出均值达到某个范围外的点，就认定是异常点。此操作推广到高维后，假设各个维度相互独立。这类方法的好处是速度一般比较快，但因总要进行” 假设”，所以效果不一定很好。因此统计技术的主要挑战是减少硬阈值引起的误报[10]。例如，可以使用统计信号处理程序来提高检测率，同时减少误报，如 Lakhina 等人在主成分分析工作中所做的工作[11-13]。

Nong 等人[14] 基于卡方检验的距离测量方法，将卡方检验的理论应用于异常检测。根据此技术，首先需要建立一个正常事件的基线，然后将偏离统计值的点视为异常。基于 chi-square 检验统计量的距离测量方法为：

$$\chi^2 = \sum_{i=1}^{n} \frac{(X_i - E_i)^2}{E_i} \tag{2-1}$$

其中 $X_i$ 为第 $i$ 个变量的观测值，$E_i$ 为第 $i$ 个变量的期望值，$n$ 为变量的数量。当一个变量的观测值接近预期时，$\chi^2$ 的值就会很低。根据 $3\sigma$ 定律，当观测值的 $\chi^2$ 大于 $\bar{X^2} + 3S_X^2$ 时，该值被视为异常。

#### 2.3.2.1　小波分析

小波变换的基本原理涵盖在傅里叶变换的过程当中，起初三角函数基是无限长的，通过小波变换将其变成长度有限且会衰减的小波基。强大的基函数小波，在时间和频率上是局部的，允许被表示的序列和它们的系数之间有密切的联系。如此一来频率得以获取，同时还可以定位到时间。

Callegari 等人[15] 提出了一种利用小波与基线相结合的实时异常检测方法。它是通过提取 NetFlow 轨迹，并将其转化为 ASCII 数据文件，进行路由器级别的分析。经过格式化后，通过哈希函数将不同的流量汇总到基线中。然后，将时间序列数据进行小波变换，若发现不连续点则视为异常点。

另一项使用小波的研究是由 Hamdi 等人[16] 产生的。它依赖于通过区分危险和非威胁性的异常来识别与攻击相关的异常。这项任务是在周期观测概念的基础上完成的，小波理论被用来分解一维信号，以分析其特殊频率和时间定位。

#### 2.3.2.2　主成分分析

主成分分析 (Principal component analysis，PCA) 方法的通常作用是降维，也可以用于网络流量异常检测领域。该方法的主要原理是将特征空间为 $n$ 的原数据集映射到新的更小的特征空间 $k$ 中，新特征即为主成分 (PC)，其中 $k << n$，这些 PC 是一组正交向量，它们构成了一个 $k$ 维子空间。同理，异常点的判别方法，是将存在于初始空间内的那些数据，映射到主成分空间，接着再把投影拉回到原始的空间。假如进行投影和重构的数据只包含第一主成分，那么大多数的数据在重构前后误差值不大；但是对于异常点而言，重构之后的误差，相较此前的误差值依然很大。

Lakhina 等人[11] 利用 PCA 将流量测量结果有效地分离为正常和异常子空间，解决了网络流量的异常诊断问题。该方法是基于将一组网络流量测量所占的高维空间分离成不相干的子空间，分别对应正常和异常的网络条件。PCA 的结果是 $k$ 个子空间，对应正常的网络流量行为，而剩余的 $m$ 个子空间 ($m = n - k$) 则由异常和噪声组成。然后，将每一个新的流量测量值映射到这两个子空间上，这样就可以通过设置不同的阈值将这些测量值划分为正常或异常。

Pascoal 等人[17] 提出的异常检测方法采用了鲁棒 PCA 检测器与鲁棒特征选择算法合并，以获得对不同网络背景和环境的适应性。该鲁棒 PCA 与经典的 PCA 方法相反，它对异常值不敏感，并且无需使用可靠标记的数据集进行训练。

Fernandes 等人[18] 提出了一种基于统计程序主成分分析的异常检测算法。该算法首先生成一个 “使用流量分析的网络段数字签名 (DSNSF)” 的网络基线，然后将该基线与真实网络流量进行比较以识别异常事件。该系统分析了几天内的历史网络流量数据，从中找出最重要的流量时间间隔，同时对数据集进行了缩减，使新的缩减集能够有效地描述正常的网络行为。然后，以 DSNSF 作为阈值，通过得到的 PCA 参数，限制一个区间，偏离阈值的被认为是异常的。该系统共使用七个流量特征，三个 IP 流量特征 (bits/s，packets/s，flows/s) 被用来生成 DSNSF，四个流量属性（源 IP 地址、目的 IP 地址、源 TCP/UDP 端口和目的 TCP/UDP 端口）被用来生成一个包含有关异常流量间隔的报告。这种方法的缺点是只使用流量属性进行异常检测，只考虑检测基于流量的攻击。

#### 2.3.2.3　协方差矩阵

协方差矩阵是二阶统计，已经被证明是一种强大的异常检测方法。该领域的一个有趣的方向是寻找哪些变量最能标记网络异常，用以提高检测性能。

Yeung 等人[19] 采用协方差矩阵分析来检测洪泛攻击。该方法将网络流量建模

为协方差矩阵样本，以利用时序样本中包含的统计量达到检测泛洪攻击的目的，直接利用协方差矩阵的变化和相关特征的差异来揭示正常流量与各种类型的洪泛攻击之间的变化。

Miao Xie 等人[20] 研究了一种基于段的方式处理数据的技术。因为在无线传感器网络 (WSNs) 中，人们观察到大多数异常事件都会持续相当长的时间。由于现有的异常检测技术通常是以基于点的方式单独处理每个观测值，它们无法可靠和有效地报告在单个传感器节点中出现的这种长期异常。因此该方法采用斯皮尔曼等级相关系数（Spearman's rank correlation coefficient 或 Spearman's $\rho$）和差分压缩概念近似的样本协方差矩阵，以大幅降低计算和通信成本。

### 2.3.3　基于信息论的异常检测算法

信息论是一门以信息量化和冗余分析为核心的数学学科，其前身是 1948 年 Claude E. Shannon 在寻求信号处理和通信操作的数据压缩、传输和存储时提出的设想[21]。然而，它的应用扩展到许多其他领域，如电信、决策支持系统、模式识别等。常用的信息理论测量方法有香农熵、广义熵、条件熵、相对熵、信息增益和信息成本等。信息论应用于异常检测的途径主要是依靠计算流量特征的相互信息或熵值来识别异常分布。

#### 2.3.3.1　熵

熵 (entropy) 在实验中可以当作是不确定性的度量，因为它是根据收到的每条消息中所包含的信息，平均后得出的量值，所以信息来源越随机，熵的值就越大。异常检测领域中，熵可以有效地将流量特征描述为分布，例如源/目的端口或 IP 地址，因为有某些类型的异常会对严重影响这些分布。通过这种方式，可以检测到例如由目的端口熵的变化表示的端口扫描攻击。

Behal 等人[22] 指出，由于 DDoS 攻击和突发事件会引起网络流量模式的大幅改变，而基于信息理论的熵或散度可以快速捕捉网络流量行为中的这种差异。因此，他们提出了一种利用流量之间的熵差进行异常检测的算法。通过采用了一组泛化的 $\phi$-熵和 $\phi$-散度，检测合法流量和攻击流量之间的信息距离。经过实验，该算法对于突发事件和 DDoS 的检测精度较高，而在其他数据集上表现一般。

David 等人[23] 提出了一种通过快速熵和基于流量的分析来增强对 DDoS 攻击的检测的方法。作者将观察到的流量汇总成一个单一的流量，并考虑到每个连接在一定时间间隔内的流量数，而不是取每个连接的数据包数。第二步基本上是计算每个连接的流量计数的快速熵。最后，根据快速熵和流量计数的均值和标准差生成一个自适应阈值。阈值随流量模式状况不断更新，提高了检测精度，同时快

速熵的使用减少了计算处理时间。

#### 2.3.3.2　KL 散度

KL 散度，又称相对熵，通常用于测量一个随机变量 $X$ 的真实概率分布 $P$ 与任意概率分布 $Q$($P$ 的近似) 之间的差异。设 $p(x)$、$q(x)$ 是离散随机变量 $X$ 中取值的两个概率分布，则 $p$ 对 $q$ 的相对熵是：

$$D_{KL}(p||q) = \sum_{x} p(x) \log \frac{p(x)}{q(x)} = E_{p(x)} \log \frac{p(x)}{q(x)} \tag{2-2}$$

在机器学习中，$P$ 往往用来表示样本的真实分布，$Q$ 用来表示模型所预测的分布。

Xie 等人[24] 利用 KL 散度着重检测了无线传感器网络 (WSN) 中的一种特殊类型的异常，这种异常会同时出现在邻近节点的集合中，并持续相当长的时间。基于节点的技术在此场景下效果和效率都不尽如人意。作者提出了基于分布式段的递归核密度估计，通过计算两个时间段的概率密度函数的差异，来判断是否发生了异常。同时也为了以较低的通信成本实现分布式估计，作者采用 KL 散度作为度量方法。利用真实世界的数据集对算法进行评估，结果表明，该算法可以以更低的通信成本实现很好的性能。

Li 等人[25] 以检测无线传感器网络中的异常数据值为目标，提出了一种基于差分 KL 散度的异常检测方案。该方案首先将整个传感器网络划分为若干个簇，每个簇中的传感器在物理上相互接近，并且具有相似的感知值。然后，在每个簇内使用 KL 散度，以通过统计测量两个数据集之间的差异来检测异常值。他们的工作取得了良好的检测率和较低的误报率，同时比其他文献中的类似研究消耗更少的 CPU、内存等资源。

### 2.3.4　基于聚类的异常检测算法

Rajasegar 等人[26] 发明了一种聚类算法，是根据分布式超球面集群得出的，主要作用在于检查测试无线传感器网络中有无异常。该算法利用聚类对每个节点的流量数据进行建模，通过使用 k 个最近邻（KNN）集群的平均集群间距来识别异常集群，就可以将数据向量分类为正常或异常。该算法的特点是在分布式系统下进行，传感器节点上报集群聚类的信息，在与其他节点通信之前，由中间节点先行合并，从而使得通信开销最小化。

K-means 是一种经典的聚类技术，能够将数据分为不同的类别，但是它存在局部收敛性和对聚类中心点选择的敏感性等缺点。因此，许多研究者尝试将 k-means 与其他技术相结合，以克服这些缺点。Karami 等人[27] 设计了一种基于粒子群优化 (particle swarm optimization，PSO) 和 k-means 与局部优化混合的模糊异常检测

系统，以确定最优的簇数。它分为两个阶段：训练阶段旨在通过将 PSO 的全局搜索的边界处理方法与 k-means 的快速收敛相结合，既要找到近似的最优解，也要尽量使局部最优解的困境不要发生。在检测阶段，由于对于任何数据 (正常或攻击)，都有可能与某些集群处于近距离，因此通过引入模糊方法，可以有效降低误报率。

Carvalho 等人[28] 开发了一种主动式网络监控系统，可以检测异常事件，减少决策中的人工干预和错误概率。他们提出一种创建网络基线轮廓 DSNSF（使用流量分析的网段数字签名）的方法。该方法通过修改蚁群优化算法，使用聚类方法描述正常的网络使用情况，该方法称为 ACODS。ACODS 在大量高维输入数据中，通过无监督学习机制优化提取行为模式，对网络流量发现进行表征。然后为了检测异常行为，他们首先计算每个时间区间内真实流量与正常曲线的相似度；然后计算序列之间的距离，并提供基于距离的测量方法。作者所提出的告警系统采用七种流量属性 (Bits，Bytes，Flows，Origin IP，Destination IP，Origin Port，Destination Port) 工作，利用熵来计算 IP 地址和端口特征的相关信息。当检测到异常时，ACODS 会提供一份包含 IP 流量信息的完整报告，说明每个属性对检测到的异常时间间隔的影响。ACODS 具有平方复杂度，导致解的收敛要经过多次迭代，作者试图通过使用局部搜索和信息素更新来缓解。

Dromard 等人[29] 提出了一种基于网格增量聚类算法和离散时间滑动窗口的无监督异常检测器。网格增量聚类比常规的聚类算法更有效率，因为后者只更新之前的特征空间分区，而不是每当增加或删除很少的点时，就对整个空间进行重新分区。增量网格聚类的使用有助于降低系统复杂度，从而使其在实时检测方面更加可行。最后系统合并这些更新的分区，用以识别最不相似的异常值。

Syarif 等人[30] 研究了各种聚类算法在应用于异常检测时的性能。他们使用了五种不同的方法，即 k-means、改进的 k-means、k-medoids、期望最大化（EM）聚类和基于距离的异常检测算法。表2.1展示了这些算法在 NSL-KDD 数据集下的性能表现。

表 2.1　聚类算法在 NSL-KDD 数据集的性能对比

| Algorithm | Accuracy(%) | False positive(%) |
|---|---|---|
| k-means | 57.81 | 22.95 |
| Improved k-means | 65.4 | 21.52 |
| k-medoids | 76.71 | 21.83 |
| EM clustering | 78.06 | 20.74 |
| Distance-based anomaly detection | 80.15 | 21.14 |

## 2.4　现有异常检测算法存在的问题

近些年出现了数以千计的异常检测算法，它们的检测原理，适用范围以及所使用的流量特征各不相同。通过本文的实验对比我们可以看出，在应对大规模、应用类型多、异常流量是常态且多种异常相互叠加的场景下，现有异常检测算法大多存在以下缺陷：

1. 面对海量数据规模时，无法或很难做到实时性；
2. 应用类型多导致的流量特征复杂，因此很难达到较高的检测率和较低的误报率；
3. 大多数异常检测算法仅能报告是否发生异常，难以对确定异常种类和定位异常来源给出指导性意见。

由于上述缺陷，当前大多数异常检测算法仍处于概念验证或模拟实现的阶段，距离实际的部署和应用还有不小的距离。

# 第 3 章　数据集分析与对比

## 3.1　开源数据集

流量异常检测领域的开源数据集要么过于久远，无法反映当前的网络环境，要么模拟访问环境过于简单，可信度很低。因此流量异常检测领域的高质量的开源数据集很少。本文采用目前应用最为广泛的两个数据集，UNSW-NB15 和 CICDIS2017，尽管这两个数据集也有很多不足。澳大利亚网络安全中心（ACCS）的网络范围实验室在 KDD99 数据集的基础上，生成了 UNSW-NB15 数据集[31] 的原始网络流量数据。该数据集主要利用 IXIA PerfectStorm 工具生成正常活动的流量和人为构造的多种攻击流量。相比于陈旧的 KDD99 数据集[32]，其更能代表真实的网络流量。

CICIDS2017 数据集是由加拿大网络安全研究所（Canadian Institute for Cybersecurity）提供。与 UNWS-NB15 类似，CICIDS2017 也是在仿真环境中模拟正常流量和攻击流量生成的。

UNSW-NB15 的实验环境划分了 3 个子网，采用了 45 个独立的 ip 地址，持续约 30 个小时，采集了共 100GB 的原始网络流量数据。CICIDS2017 的实验环境划分了 2 个子网，分为受害者子网和攻击者子网，受害者子网共有 12 台主机，攻击者子网共 4 台主机，累计测量时间持续约 5 天，共采集了 51.1GB 的 pcap 流量数据。由这两者的实验环境可以看出，开源数据集在规模上远远无法和真实流量环境对比。

以 UNSW-NB15 为例，图3.1展示了该数据集的生成过程。首先，通过使用 Tcupdump 工具监测访问环境中的流量情况，生成 pcap 文件；然后使用 Argus①和 Bro-IDS②工具从报文信息中直接提取基于数据包和数据流的特征，并根据五元组（源 ip 地址，源端口号，目的 ip 地址，目的端口号，协议类型）信息进行匹配，此时基本特征、内容特征和时间特征已生成；接下来为了能够更有效地识别攻击，还需要额外生成一些统计特征。最终将这些特征保存成 csv 文件。

这些数据集生成流程和特征提取方法给后续我们分析清华大学校园网的流量数据提供了参考。

① https://qosient.com/argus/index.shtml
② https://www.bro.org/index.html

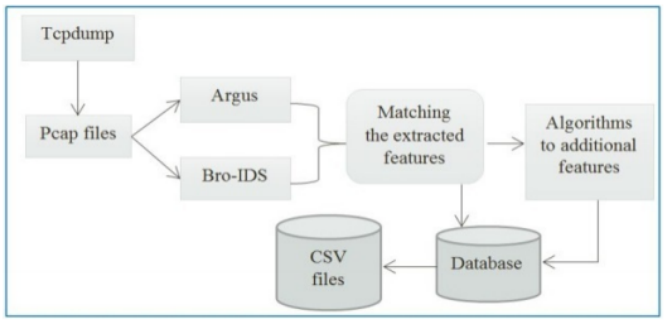


图 3.1　UNSW-NB15 数据集生成过程

## 3.2　校园网流量数据集

清华大学无线网规模巨大，具有超过 6 万名日活跃用户，13 万个可用 ip 地址，是全球最大的校园无线网之一。通过出口网关的服务器我们可以得到海量的真实流量数据，但是这与可进行分析和训练的数据集之间还有巨大的鸿沟。因此本节参考图3.1中的数据集生成流程，结合 UNSW-NB15 和 CICIDS2017 两个数据集中特征的特点，生成了校园网的特征数据集。校园网数据集制作流程如图3.2所示。首先将 pcap 流量数据切分成多个会话（session，双向流中所有的数据包），依次从每个会话中提取报文信息以及统计信息构成特征；然后以五元组作为流的标识符，把从 SOC 平台中得到的告警信息与特征信息合并起来，就生成了最终的数据集。

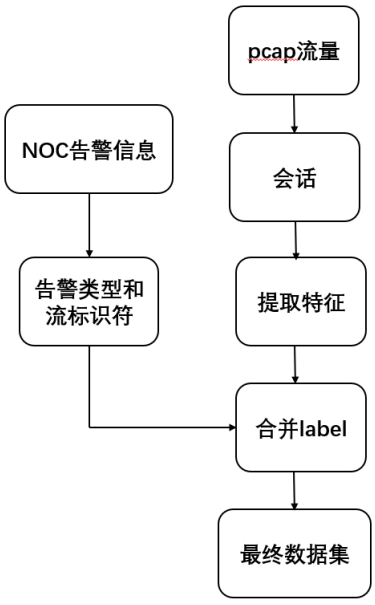


图 3.2　流量数据集制作流程

本节共采集了两类数据集，分别是出口网关处的全流量数据、安全管理平台（Security Operations Center，SOC）的威胁告警日志。接下来分别介绍这四类数据。

### 3.2.1　全流量日志数据

校园网的原始数据为抓包（Packet Capture，pcap）流量，如下图 3.3所示，其中包含每个数据包的详细信息，如五元组、报文头部信息、报文内容等，本文对所有用户隐私信息（如 IP 地址和 MAC 地址，报文中的明文信息等）都进行了匿名化处理，保证不会泄露用户的任何隐私信息。

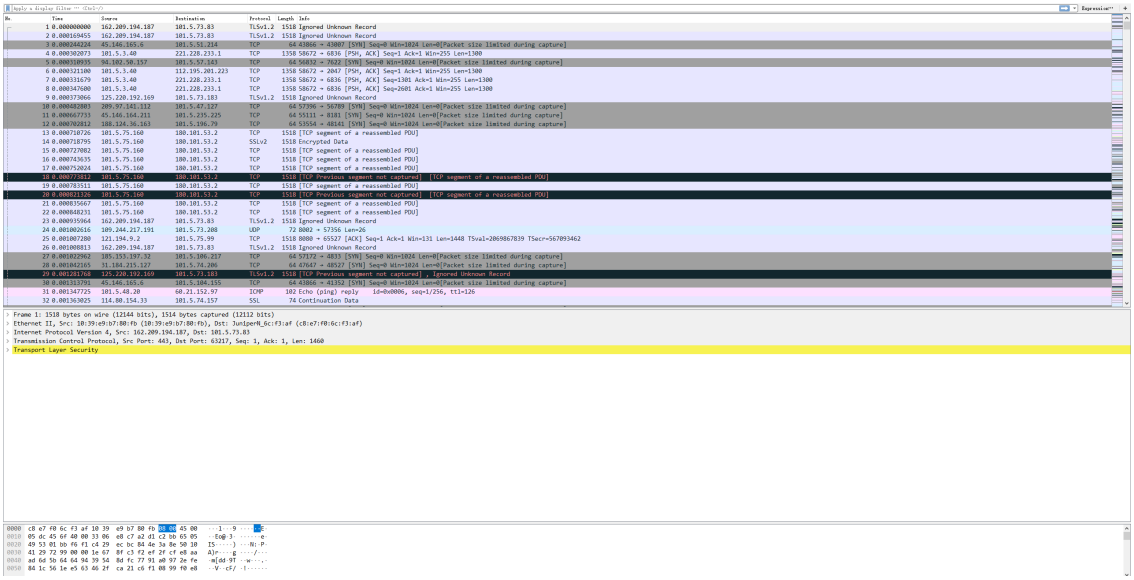

图 3.3　流量数据示意图

### 3.2.2　威胁告警日志

我们通过全流量日志得到了特征数据集，但是由于缺乏标注，该数据集无法进行训练以及验证。因此为了得到可进行训练的有效数据集，我们还需要对特征数据集添加标注。标注数据是根据现有的 SOC 平台得到，需要进行数据清洗、数据预处理等步骤。图3.4为 NOC 平台的数据样例。

## 3.3　现有算法在不同数据集的评估

本节分别在以上三个数据集上进行实验，对比了现有的经典算法，以下是这些算法的名称、特点及实验中的参数。

待补充结果对比图

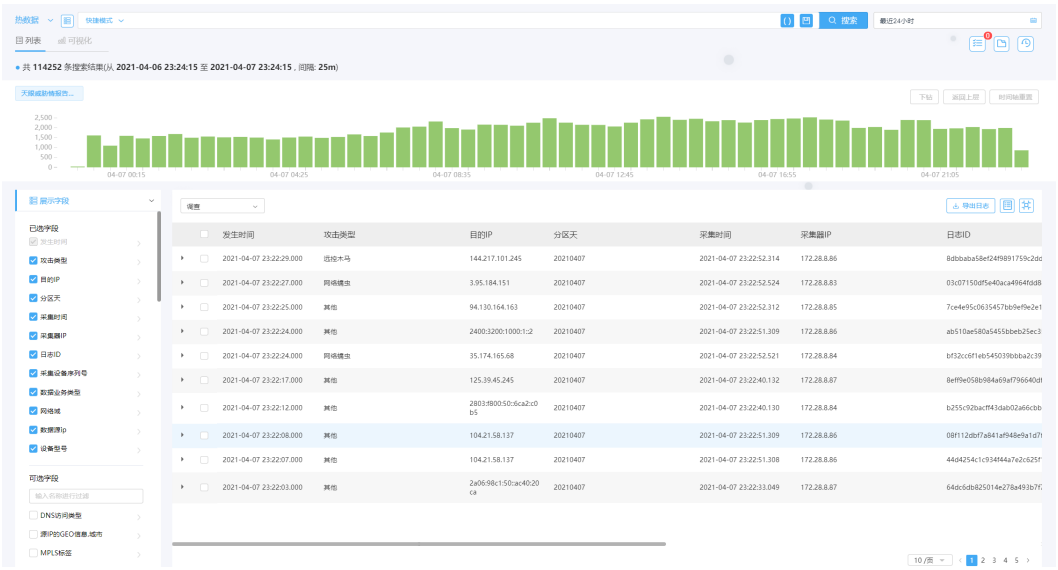

图 3.4　NOC 平台数据样例

表 3.1　部分评估结果，待填充具体数值

| 数据集 | 任务 | LR | NB | DT | CNN | CNN-LSTM | GRU | DCRNN-A | DCRNN-B |
|---|---|---|---|---|---|---|---|---|---|
| UNSW-NB15 | 二分类 | 0.848 | 79.33 | 78.52 | 80.51 | 62.74 | 68.91 | **82.69** | 81.66 |
| | 多分类 | 76.73 | 77.96 | 76.82 | 77.98 | 47.11 | 56.30 | **81.05** | 79.94 |
| NSL-KDD | 二分类 | 68.26 | 67.11 | 65.54 | 70.18 | 65.04 | 58.49 | 70.26 | **70.88** |
| | 多分类 | 67.01 | 67.37 | 66.41 | 68.31 | 56.16 | 53.18 | 68.47 | **70.24** |
| CAMPUS | 二分类 | 79.98 | 77.95 | 79.51 | 75.62 | 79.80 | 75.25 | **80.69** | 79.10 |
| | 多分类 | 76.33 | 77.01 | 77.54 | 73.01 | 76.59 | 59.28 | 78.12 | **78.89** |

# 第 4 章　基于特征关系图的 RNN 及其网络流量异常检测算法

## 4.1　引言

从第三章的内容可以知道，现有的异常检测算法能够在公开数据集取得较为不错的表现，但是在真实数据集——清华大学校园网流量数据上效果均无法达到要求。这是因为相比于人工构造的公开数据集，清华大学校园网数据具有规模大、应用类型种类繁多、异常流量占比多、难以学到基线等特点。现有的算法无论是传统机器学习的 LR、NB、DT，还是拟合能力更强大的深度学习模型 CNN、LSTM、GRU，都有精确率低、误报率高的问题。本章将先从特征分析着手，引入特征之间的关系矩阵，将其加入到神经网络的训练中，从而获得更好的检测效果。

## 4.2　特征分析

流量特征可以视为反映当前网络流量的一组观察值，从多个角度描述当前的流量场景。好的特征能够真实地反映出流本身的状态信息。例如对于一条单个 TCP 流，这条流可能是一个正常的端到端的链接，也可能是分布式拒绝服务攻击的一部分，在被攻击者的视角内，它的入流量的带宽急剧增加，且远大于出流量的带宽。

本文中利用皮尔逊相关系数来计算流量特征之间的关系。皮尔逊相关的别名是积差相关（或者称之为积矩相关），命名来源是 20 世纪英国的描述统计学派先驱皮尔逊。对于计算变量之间的线性相关性，他想到了一种方法，即假设当前存在两个变量 X、Y，再通过下列公式计算 X、Y 之间的皮尔逊相关系数：

$$\rho_{X,Y} = \frac{cov(X,Y)}{\rho_X \rho_Y} = \frac{E(XY) - E(X)E(Y)}{\sqrt{E(X^2) - E^2(X)}\sqrt{E(Y^2) - E^2(Y)}} \tag{4-1}$$

## 4.3　如何向 RNN 中加入特征图信息

在 RNN 的训练过程中，加入有效的额外辅助信息往往能够提升训练效果，例如在机器翻译领域，加入文章的关键词、摘要、作者信息等。通常加入信息的方式有以下几种。

1. 直接将额外信息向量与当前特征向量叠加。原向量为 $p = (p_1, p_2, ..., p_n)$，额外

信息向量为 $e = (e_1, e_2, ..., e_n)$，最终输入向量为 $w = (p_1 + e_1, p_2 + e_2, ..., p_n + e_n)$。这种方法需要保证额外信息向量与原向量维度相同。

2. 将额外信息向量与当前特征向量拼接。原向量为 $p = (p_1, p_2, ..., p_n)$，额外信息向量为 $e = (e_1, e_2, ..., e_m)$，最终输入向量为 $w = (p_1, p_2, ..., p_n, e_1, e_2, ..., e_m)$。也就是增加输入的维度，缺点是通常要求额外信息的特征与原特征类型保持一致，例如均为词向量。
3. 增加一个额外的隐藏层，分别使用不同的矩阵进行变化，将结果用 $tanh$ 函数映射到所需的维度。相当于增加一个普通循环神经网络模型和额外信息模型的感知器，然后加载到输出层上。即：

$$h'_t = tanh(W(p_t) + W(e) + b^{h'_t}) \tag{4-2}$$

因此，本文采用第三种方式将额外的特征关系图信息与原有特征结合起来。

### 4.3.1　神经网络：修改标题

经过数十年的发展，神经网络有非常多的种类，按照网络中是否包含循环可以分为前馈神经网络和循环神经网络。

1. 前馈神经网络：前馈神经网络是一种单元之间连接不形成循环的神经网络。在这种网络中，信息从输入到输出正向流动。前馈神经网络如果只有一层输入节点、一层输出节点，不包含隐藏层，那么被称为单层感知器（Single Layer Perceptron）。若网络由多层计算单元组成，以前馈方式相互连接，则被称为多层感知器（Multi Layer Perceptron）。此外还有一类前馈神经网络，卷积神经网络 (CNN)。卷积神经网络与普通的神经网络非常相似，它们是由具有可学习权重和偏差的神经元组成的。在卷积神经网络（CNN，或 ConvNet 或移位不变或空间不变）中，单元连接模式的灵感来自于视觉皮层的组织，单元在一个被称为感受场的受限空间区域内对刺激做出反应。感受场部分重叠，覆盖了整个视场。单元响应可以用卷积运算在数学上近似。
2. 循环神经网络在循环神经网络 (RNN) 中，单元之间的连接形成了一个定向循环 (它们向前传播数据，同时也向后传播数据，从较后的处理阶段到较早的阶段)。这使得它能够表现出动态的时间行为。与前馈神经网络不同，RNNs 可以利用其内部存储器处理任意输入序列。这使得它们适用于未分割、连接的手写识别、语音识别和其他一般序列处理器等任务。

假设一个神经元接收 $D$ 个输入 $x_1, x_2, ..., x_D$ 令向量 $x = [x_1; x_2; ...; x_D]$ 来表示这组输入，净输入也叫净活性值（Net Activation）. 并用净输入（Net Input）$z \in \mathbb{R}$ 表示一个神经元所获得的输入信号 $x$ 的加权和，

$$z = \sum_{d=1}^{D} w_d x_d + b \\ = w^T x + b \tag{4-3}$$

其中 $w = [w_1; w_2; ...; w_D] \in \mathbb{R}^D$ 是 $D$ 维的权重向量，$b \in \mathbb{R}$ 是偏置。

净输入 $z$ 在经过一个非线性函数 $f(\cdot)$ 后，得到神经元的活性值（Activation，

$$a = f(z) \tag{4-4}$$

其中非线性函数 $f(\cdot)$ 称为激活函数。

一个人工神经元的结构如图 4.1所示：

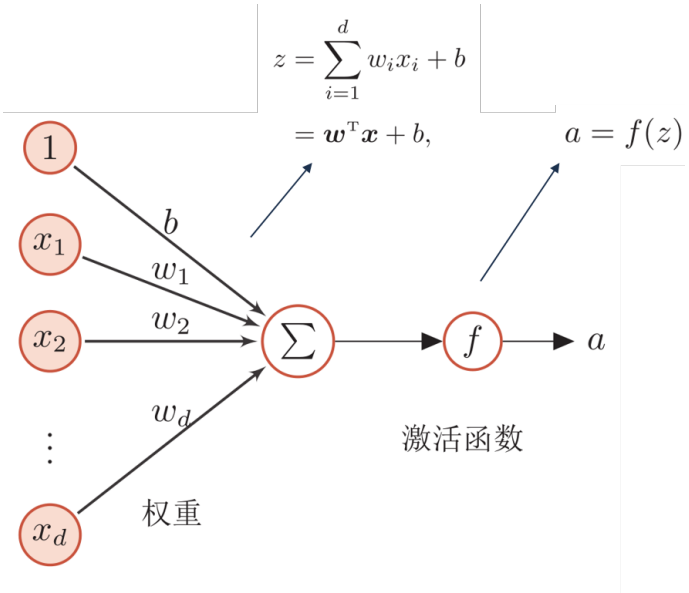


图 4.1　人工神经元模型

在图 4.1中，$\vec{x}$ 为输入向量，$w$ 和 $b$ 分别是权重和偏移，

神经网络主要由以下几部分组成：

- 输入节点（输入层）。在这一层中不进行任何计算，它们只是将信息传递给下一层（大部分时间是隐藏层）。
- 隐藏节点（隐藏层）。中间处理或计算在隐藏层中完成的，然后将输入层的权重（信号或信息）传递给下一层（另一个隐藏层或输出层）。一个神经网络也可以不包含隐藏层。
- 输出节点（输出层）。此层为神经网络的最末层，隐藏层所输入的信息或信号会被这一层接收。再通过激活函数，最终将得到合理范围内的理想数值，例如用于分类的 softmax 函数。
- 连接和权重。神经元之间会有边进行连接，每条边会有一定的权重。即每个连

接将神经元 $i$ 的输出传递给神经元 $j$ 的输入, 每个连接被赋予一个权重 $W_{ij}$。

- 激活函数。这个函数的作用在于将非线性特征引入到神经网络当中。同时它会将值的范围紧缩至更小，所以一个 Sigmoid 激活函数的值区间为 [0,1]。深度学习中有很多激活函数，如 Sigmoid、Tanh、ReLU 、Softplus、Softmax 等。表4.1为常见的激活函数。

表 4.1　激活函数

| 名称 | 表达式 | 导数 |
|---|---|---|
| Sigmoid | $f(x)=\frac{1}{1+e^{-x}}$ | $f'(x)=f(x)(1-f(x))$ |
| Tanh | $f(x)=\frac{2}{1+e^{-2x}}-1$ | $f'(x)=1-f(x)^2$ |
| ReLU | $f(x)=max(0,x)$ | $f(x)=\begin{cases}0 & \text{x<0}\\ 1 & \text{x>=0}\end{cases}$ |
| Softplus | $f(x)=log(1+e^x)$ | $f'(x)=\frac{1}{1+e^x}$ |
| Softmax | $S_i=\frac{e_i}{\sum_j e_j}$ | |

- 学习规则。学习规则是一种规则或算法, 它修改神经网络的参数, 以使网络的给定输入产生一个有利的输出。这个学习过程通常相当于修改权重和阈值。

### 4.3.2　RNN

下图 4.2是循环神经网络的示意图。该图显示了一个循环神经网络被展开成一

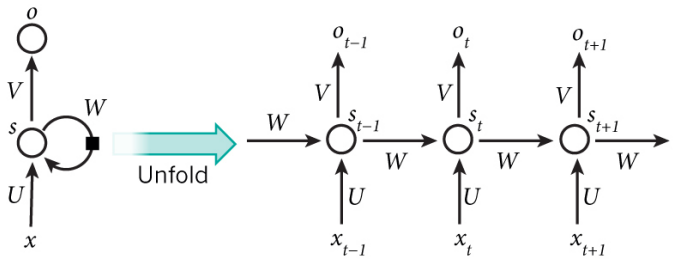


图 4.2　循环神经网络

个完整的神经网络。例如，如果输入序列是一个由 5 个单词组成的句子，那么网络就会被展开成一个 5 层神经网络，每个单词一层。这个网络在 t 时刻接收到输入 $x_t$ 之后，隐藏层的值是 $s_t$ ，输出值是 $o_t$ 。关键一点是，$x_t$ 的值不仅仅取决于 $s_t$ ，还取决于 $s_{t-1}$ 。用公式表示如下：

$$\begin{aligned} O_t &= g(V \cdot S_t) \\ S_t &= f(U \cdot X_t + W \cdot S_{t-1}) \end{aligned} \tag{4-5}$$

$x_t$ 表示第 $t$ 步的输入，例如 $x_1$ 表示时刻 1 的特征向量。$s_t$ 表示第 t 步隐藏层

状态，也就是网络中的“记忆”。$s_t$ 是由前一层的隐藏层状态和当前层的输入计算得到。函数 $f(\cdot)$ 通常是一个非线性函数，如 tanh 或者 ReLU。

### 4.3.3　LSTM

普通 RNN 有不能处理长依赖的问题，因此 Hochreiter 提出了一种长短期记忆网络-LSTM，以及它的变种，它是一种特殊的 RNN，适用于学习长期依赖。现在 LSTM 已经被广泛应用于各个领域。

所有 RNN 都具有链式形式。在普通的 RNN 中，这种循环是一种非常简单的结构，比如简单的 tanh 层。

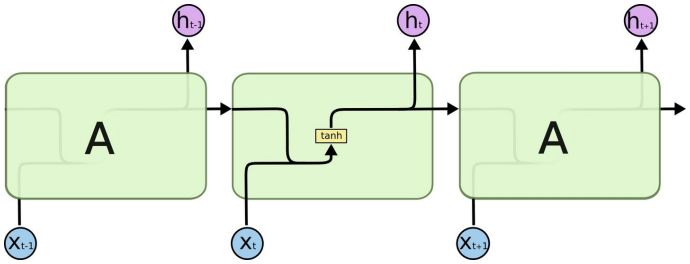


图 4.3　普通 RNN 结构

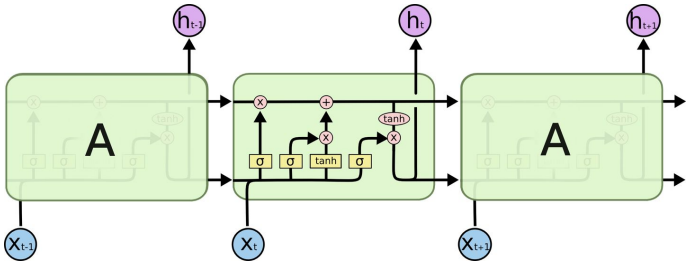


图 4.4　LSTM 结构

LSTM 中的第一步是决定从单元状态中丢弃什么信息。通过加入一个称为忘记门的 $\sigma$ 层完成。遗忘门会读取上一时刻的输出 $h_{t-1}$ 和当前时刻的输入 $x_t$，计算出一个维度为 n 的向量 $f_t$，该向量的值均在 $(0,1)$ 之间。1 表示“完全保留”上个神经元的状态信息，0 表示“完全舍弃”。

$$f_t = \sigma(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f) \tag{4-6}$$

下一步是确定该神经元的哪些新状态信息被存放在单元状态中。这里包含两个部分。第一，sigmoid 层，即“输入门层”，决定 LSTM 单元将更新哪些值。然后，tanh 层创建一个新的候选值 $z_t$ 的向量，该向量可以加入到下一层单元状态中。

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{4-7}$$

$$\widetilde{C}_t = tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{4-8}$$

最后一步是将旧单元状态 $c_{t-1}$ 更新为新状态 $c_t$。把旧状态与遗忘门 $f_t$ 相乘，丢弃掉之前需要丢弃的信息。接着与新状态进行相加。综合得出该神经元的输出的状态，也即更新单元的状态。

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{4-9}$$

### 4.3.4　GRU

循环门单元（Gated Recurrent Unit，GRU），由 Cho, et al. (2014) 提出。它组合了遗忘门和输入门到一个单独的“更新门”中。它也合并了 cell state 和 hidden state，并且做了一些其他的改变。结果模型比标准 LSTM 模型更简单，

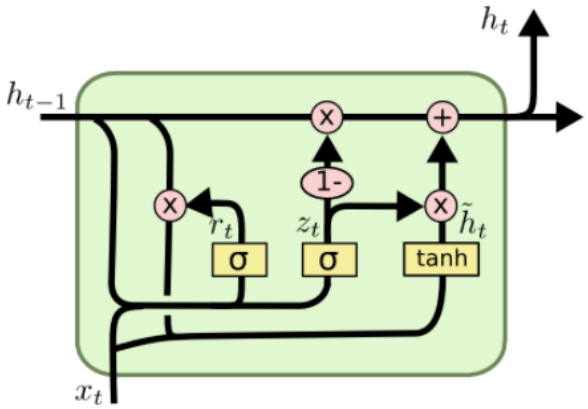


图 4.5　GRU 结构

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{4-10}$$

$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{4-11}$$

$$\widetilde{h}_t = tanh(W \cdot [r_t * h_{t-1}, x_t]) \tag{4-12}$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \widetilde{h_t} \tag{4-13}$$

由图中结构可以看出，GRU 是通过一个循环神经网络和“门”机制来不断更新内部参数，GRU 算法的伪代码如下表所示。

**算法 4.1** *GRU*

**输入：** t
**输出：** 内部参数
1: 初始化 t 时刻单元状态
2: **for** t ← 1 to $T$ **do**
3: $output_t$ = t
4: $state_t = output_t$
5: **end for**

**算法 4.2** *LSTM 层*

**输入：** 一组按时间排列的向量组
**输出：** 按时间排列的向量组
1: $\vec{C}_0 = \vec{0}$
2: $\vec{h}_0 = \vec{0}$
3: **for** $t$ ←1 to $T$ **do**
4: $output_t = activation(dot(W, input_t) + dot(U, state_t) + b)$
5: $state_t = output_t$
6: **end for**

## 4.4　基于关系图结构的 RNN

有向权重图 $G = (V, E, W)$,$V$ 表示特征节点的集合，其中 $|V| = N$, $E$ 表示特征间的关联关系，即图中的边，$W \in R[N * N]$ 为特征节点的相似度的加权邻接矩阵。将流量表示为 G 的一个图信号，P 为每个节点特征数，则-时间 t 观察到的图信号。那么流量预测目的就是：给定 G 下，学得一个函数将 T' 个历史图信号映射到未来 T 时刻的图信号：

$$h[X^{t-T+1}, X^{t-T+2}, ..., X^t; G] \Rightarrow [X^{t+1}, X^{t+2}, ..., X^{t+T}] \tag{4-14}$$

$$r^{(t)} = \sigma(\Theta_r \star G[X^{(t)}, H^{t-1}] + b_r) \tag{4-15}$$

$$u^{(t)} = \sigma(\Theta_u \star G[X^{(t)}, H^{t-1}] + b_u) \tag{4-16}$$

$$C^{(t)} = tanh(\Theta_C \star_G [X^{(t)}, (r^{(t)} \odot H^{(t-1)})] + b_c) \tag{4-17}$$

$$H^{(t)} = u^{(t)} \odot H^{(t-1)} + (1 - u^{(t)}) \odot C^{(t)} \tag{4-18}$$

其中 $X^{(t)}, H^{(t)}$ 表示在时间 $t$ 的输入和输出，$r^{(t)}$ $u^{(t)}$ 分别是在时间 $t$ 的复位门和更新门。$\star_G$ 表示扩散卷积，并且是对应滤波器的参数。与 GRU 相似，DCGRU 可用于构建递归神经网络层，并使用反向传播进行训练。

本文中我们利用循环神经网络（RNN）对时间依赖性进行建模。值得指出的是，本文同时使用了门控循环单元（GRU），它能够捕捉到时间序列中距离较大的依赖关系。在 GRU 的矩阵乘法中，我们加入了前文提到的特征关系图（Feature Graph）。

通常，计算卷积会很费时。但是，如果 $G$ 稀疏，则可以使用总时间复杂度 $O(K \mid \varepsilon \mid \ll O(N^2)$ 递归稀疏矩阵乘法来有效地进行计算。

$$hl = [hl_1, hl_2, \cdots, hl_{n-f+1}] \tag{-}$$

## 4.5　算法性能评估

为了验证 FG-RNN 模型的有效性，本节首先在清华大学无线校园网真实场景下的数据集上开展有效性评估，随后在异常检测领域下的两个经典的公开数据集上进行实验，分别对比了 FG-RNN 和多个模型的评估效果，最后分析了模型参数的敏感性。

其中多分类任务的准确率指标为对于每个标签，分别计算 precision，然后取不加权平均。

### 4.5.1　实验参数设置

在本节中，

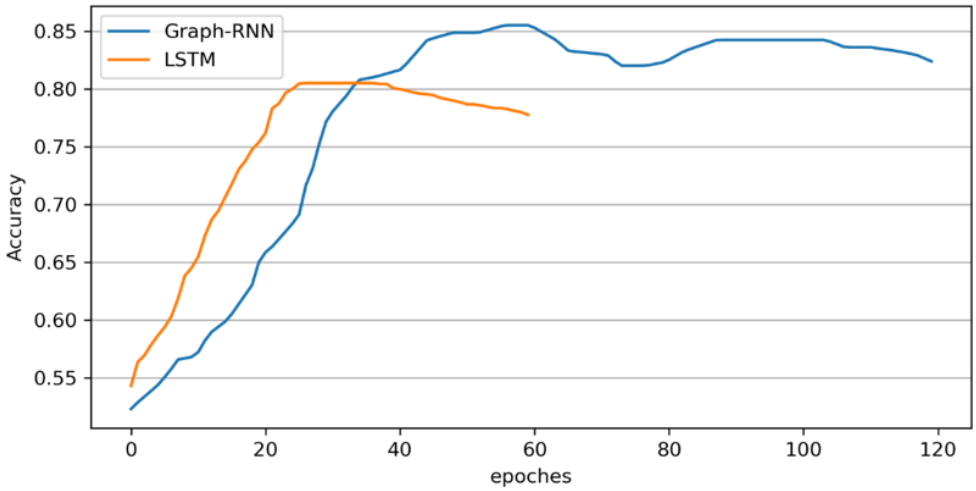


图 4.6　结果 1

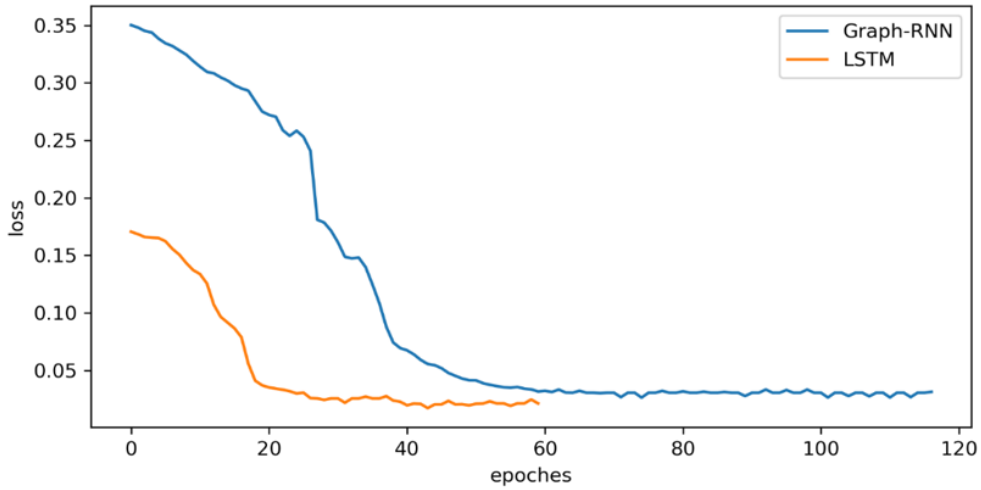


图 4.7　结果 1

TODO: 格式待调整

### 4.5.2　基于开源数据集的检测结果

### 4.5.3　基于真实数据的检测结果

我们用神经网络最后一层输出的情况，来观察整个模型预测、获得损失和学习的流程：

**算法 4.3** *FG-RNN*

**输入：** 一组按时间排序的向量组 $\vec{x}_1$,$\vec{x}_2$,...,$\vec{x}_T$, 特征间关系图 $G$ 以及它的邻接矩阵 $W$
**输出：** 按时间排序的向量组 $\vec{h}_1$,$\vec{h}_2$,...,$\vec{h}_T$

通过 Xavier 方法初始化 $q_{net1}$, $q_{net2}$ and $p_{net}$ 的所有参数
**while** $\mathcal{L}$ 没有收敛 **do**
    利用公式计算每一个节点 $v$ 的 $q_\phi(\boldsymbol{y}_v|\boldsymbol{A},\boldsymbol{X},Y_L)$ 以及 $h_v^{K-1}$
    **for** i ← 1 to $m$ **do**
        从分布 $q_\phi(Y_U|\boldsymbol{A},\boldsymbol{X},Y_L)$ 中采样 $Y_U$
        在采样得到的 $Y_U$ 基础上，利用公式计算 $q_\phi(\boldsymbol{z}|\boldsymbol{A},\boldsymbol{X},Y)$
        **for** j ← 1 to $n$ **do**
            从分布 $q_\phi(\boldsymbol{z}|\boldsymbol{A},\boldsymbol{X},Y)$ 中采样 $\boldsymbol{z}$
            在采样得到的 $\boldsymbol{z}$ 基础上用公式计算 $p_\theta(\boldsymbol{y}_v|\boldsymbol{A},\boldsymbol{X},\boldsymbol{z})$
        **end for**
    **end for**
    用公式计算目标函数 $\mathcal{L}(\theta,\phi)$
    用梯度下降法更新 $q_{net1}$，$q_{net2}$ 和 $p_{net}$ 的所有参数
**end while**

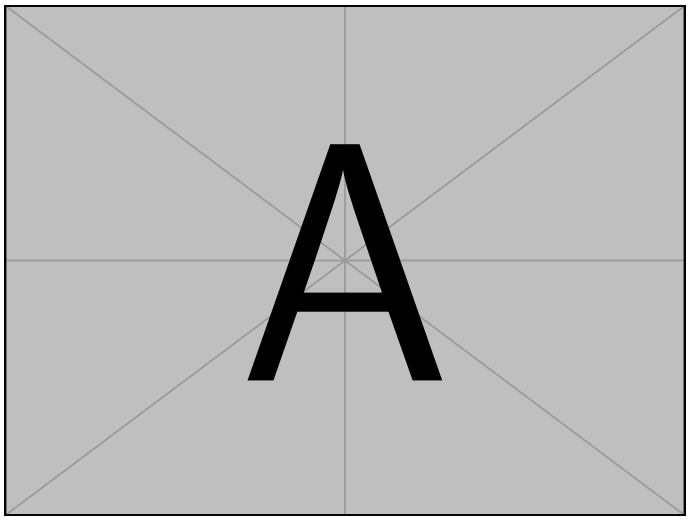


图 4.8　Example figure

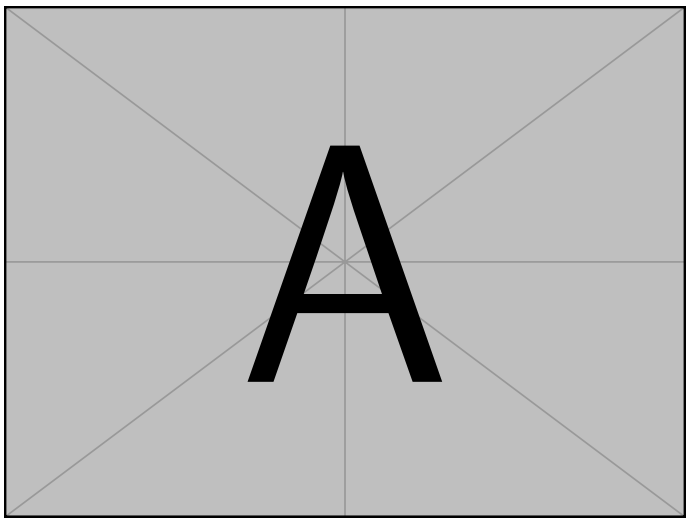


图 4.9　Example figure

# 第 5 章　流量检测系统的设计与实现

## 5.1　引言

本章结合前四章的分析和结论，将基于特征关系图的 RNN 模型应用到实际流量环境中，验证其真实有效性。因此本文针对清华大学校园网的流量环境，设计了一个流量异常检测系统。该系统应该满足以下几点要求：

1. 实时性。如果一个系统检测延迟很高，那么即使其准确率同样很高，这个检测结果也将毫无意义。因为异常检测的目的是在攻击的早期阶段就将其发现，并且作出应对。
2. 高效性。校园网用户规模大，流量峰值高，该系统需要能够高效处理海量的流量数据。
3. 鲁棒性。该系统应该保证能够在复杂的网络环境面前依然有较好的检测效果，例如发生突出事件或者新的异常类型时依然能够有一定的检测效果。
4. 可扩展性。该系统的各个模块之间的耦合度应尽可能低，一方面便于调试，另一方面可以便捷的验证其他算法模型。

本文的流量异常检测系统按照上述需求设计如图5.1所示，可以分为三个模块：输入模块，模型模块，输出模块。其中输入模块将流量数据和 SOC 标签数据整合成特征数据；模型模块分为离线训练和在线检测两部分，离线训练部分定期产出更新好参数的模型，在线检测部分会接受当前的特征数据进行判别；最后输出模块将判别结果根据历史信息进行异常等级划分，给运维人员下一步提供参考。

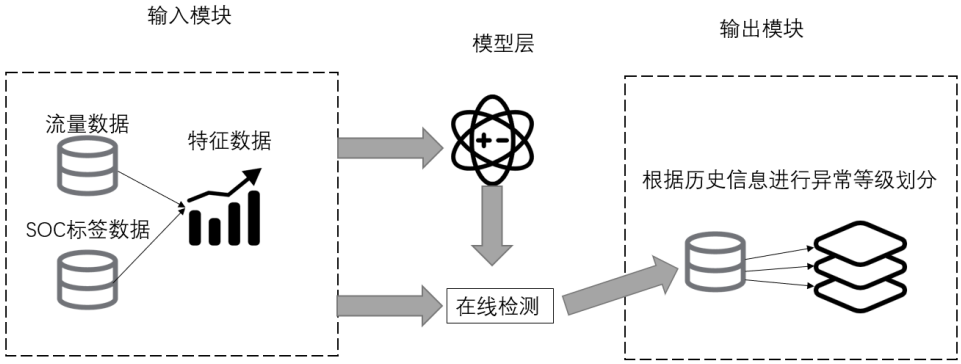


图 5.1　系统架构图

## 5.2　spark 平台介绍

Apache Spark 是由 UC Berkeley AMP Lab 开源的用于大规模数据处理的统一分析引擎[33]，现如今已经成为 Apache 软件基金会的顶级开源项目。Spark 原理与 Hadoop 类似，都是基于 MapReduce 进行分布式计算，但是功能更加丰富，且支持 SQL 查询、流式处理、图计算等功能。Spark 具有以下几个特点：

- 速度快，Spark 的速度比 Hadoop 要快 100 倍以上，图5.2是一项逻辑回归任务在 Hadoop 和 Spark 两个系统的运行时长对比。这是因为 Spark 基于内存计算，而 Hadoop MapReduce 必须进行读取和写入磁盘，IO 操作比内存操作更加耗时。
- 易用性。Spark 提供了 80 余种高级 API，可以轻松编写高度并行的应用程序。Spark 支持 Java, Scala, Python, R, and SQL 等多种编程语言。
- 兼容性。Spark 可以运行在 Hadoop 模式、Mesos 模式、Standalone 独立模式或 Cloud 中，并且还可以访问各种数据源，包括本地文件系统、HDFS、Cassandra、HBase 和 Hive 等。

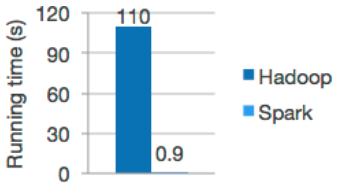


图 5.2　Logistic regression in Hadoop and Spark

如图5.3所示，Spark 项目包含多个独立组件。其最核心的模块为深蓝色的 Spark Core，其定义了核心 API，如弹性分布式数据集（Resilient Distributed Datasets，RDD），该模块也负责调度、监控计算集群中的计算任务，具有内存管理，故障恢复，与管理系统、存储系统交互等功能。为了满足分布式计算系统的可扩展性，Spark 支持在各种集群管理器之上运行，例如图中灰色的模块 Hadoop YARN，Apache Mesos 以及浅蓝色的 Spark 自身的“独立调度程序”（Standalone Scheduler）。在 Spark Core 之上，具有四个组件：Spark SQL、Spark Streaming、MLlib、GraphX。本文中的系统主要使用 Spark Streaming 组件。

## 5.3　系统整体设计

下图 5.6是 Spark Streaming 作业的执行流程，具体流程分为以下几步：

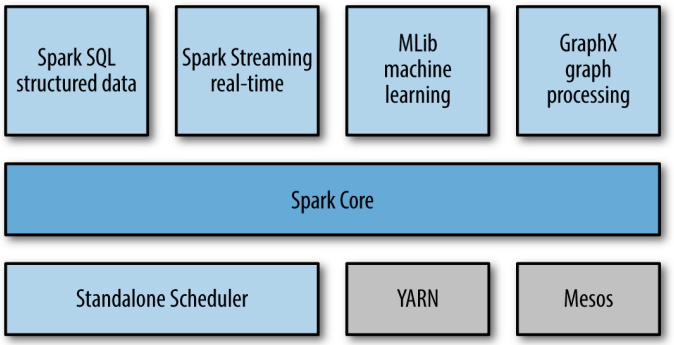


图 5.3　Spark 组件

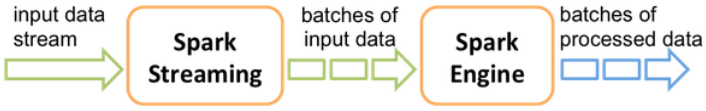


图 5.4　spark 工作原理

## 5.4　流式数据特征抽取

Spark Streaming 支持 Scala、Java 和 Python，本文中使用 Python 提交任务。使用第三章中的特征提取方法，输入为由 kafka 传入的流量数据，提取的结果都是传输层的一些统计信息，以一个 TCP 流或一个 UDP 流为一个单位。TCP 流以 FIN 标志为结束，UDP 以设置的 flowtimeout 时间为限制，超过时间就判为结束。在一个 TCP 流中有很多个数据包，先三次握手而后传输信息再四次挥手。统计一个流中的统计信息作为提取的特征。且统计的特征都分前后向，规定由源地址到目的地址为正向，目的地址到源地址为反向，例如某条 TCP 流的源 ip 地址为 192.168.31.100，源端口号为 174，目的 ip 地址为 183.232.231.174，目的端口号为 443，则为该流构建一个标志叫 Flow ID:192.168.31.100-183.232.231.174-46927-443-6, 由源地址、目的地址、源端口、目的端口、协议号组成。流量特征提取的整体流程图如下：

从 pcap 文件中逐个读取 packet，将每个数据包添加到对应的流中在 currentFlows 存储当前还未结束的所有 TCP、UDP 流。在添加的过程中不断地更新每个流的统计特征，最终将统计特征写入 csv 文件。判断新加入的数据包是否属于当前所有未结束的流，如果属于当前流则判断正向还是反向，之后判断时间是否超时、不超时则判断是否含有 FIN 标志，如果两者都不满足，则声明一个 BasicFlow 对象，根据 id 从 currentFlows 中拿到与当前数据包对应的流，调用 addPacket 将该数据包加入到对应流中。如果前面判断不在当前所有未结束的流中，则直接创建一个新的流，里面只含当前数据包，存入到 currentFlows 中。如果属于当前某个未结束的流，且超时或存在 FIN 标志，则说明当前 flow 结束，超时则从 currentFlows

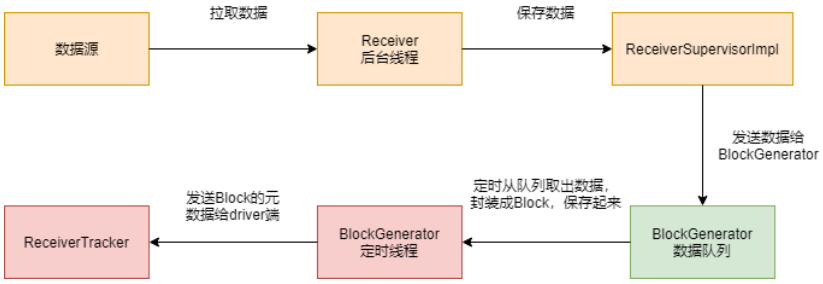


图 5.5　Spark 数据源

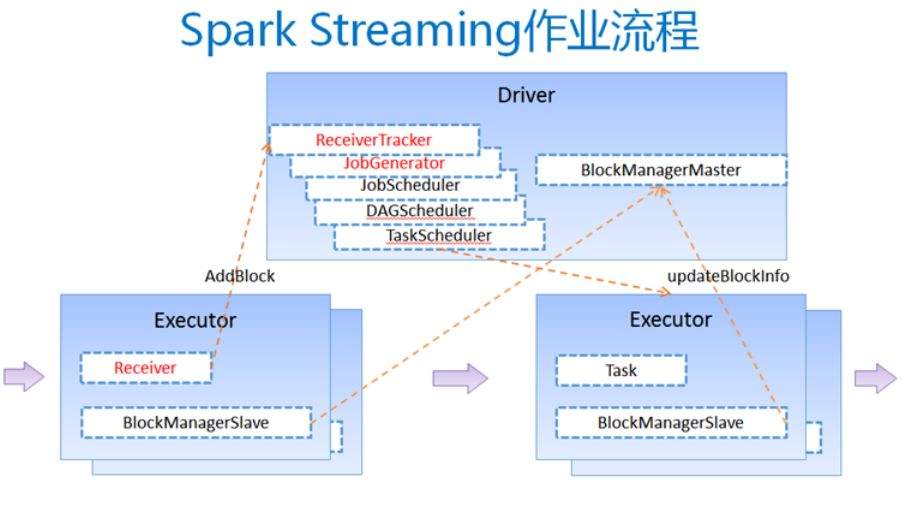


图 5.6　spark streaming 执行流程

中移除对应流，新建 flow 存入 currentFlows 中，含 FIN 标志则直接从 currentFlows 中移除对应流。结束的 flow 直接调用 onFlowGenerated 函数将流打印存储起来。

使用该方法得到的 csv 文件如下所示：

## 5.5　模型训练模块的设计与实现

补充一些图

## 5.6　预测输出模块

## 5.7　系统结果展示与分析

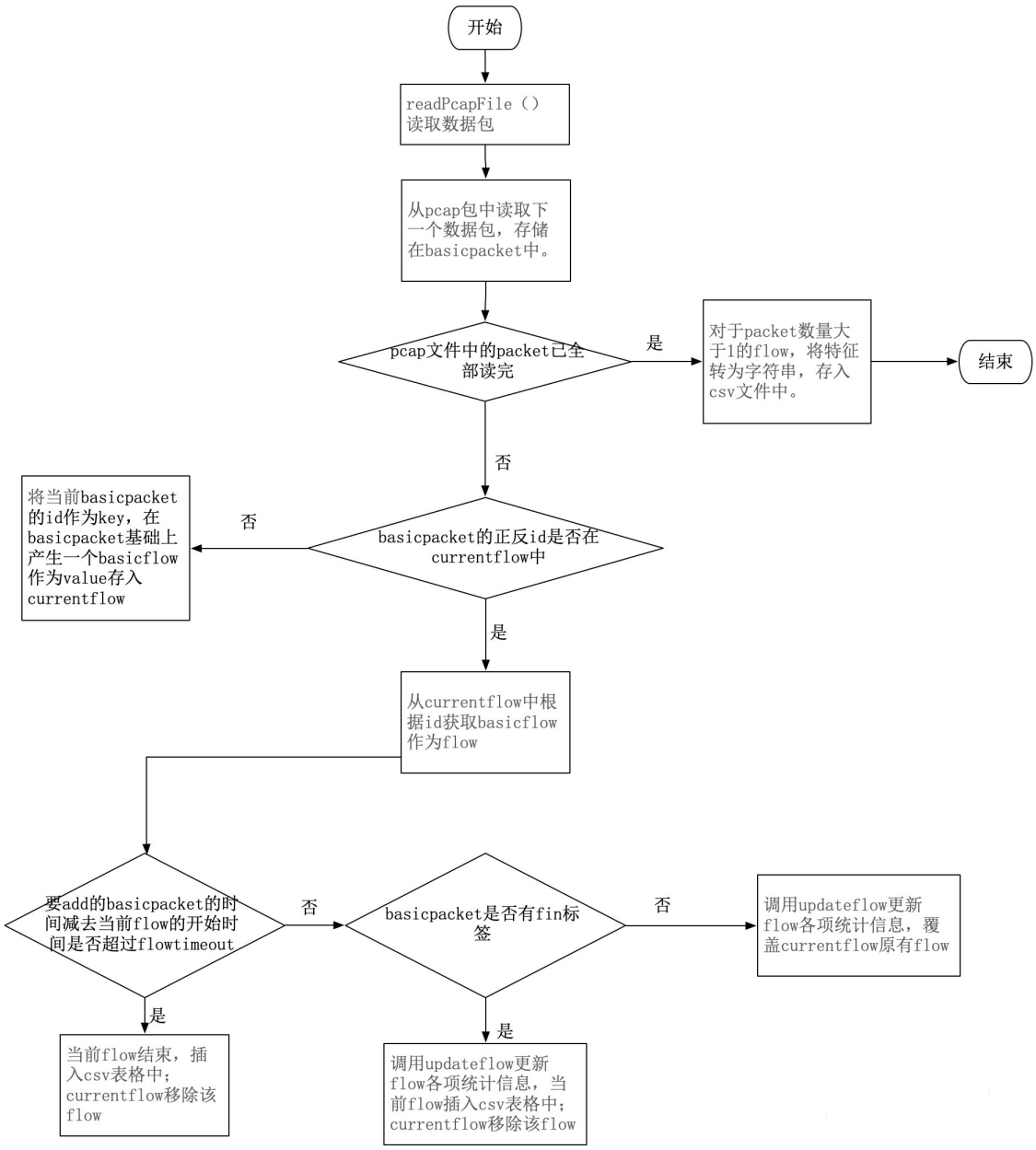


图 5.7　特征提取

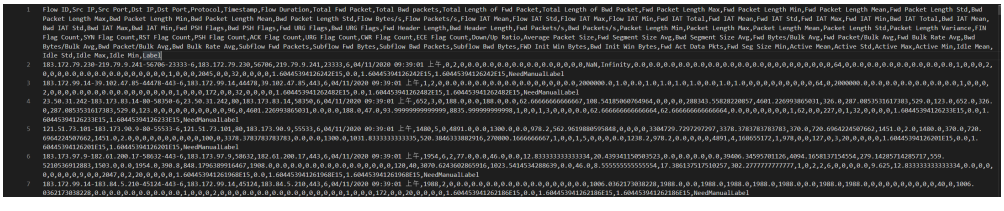

图 5.8　特征文件

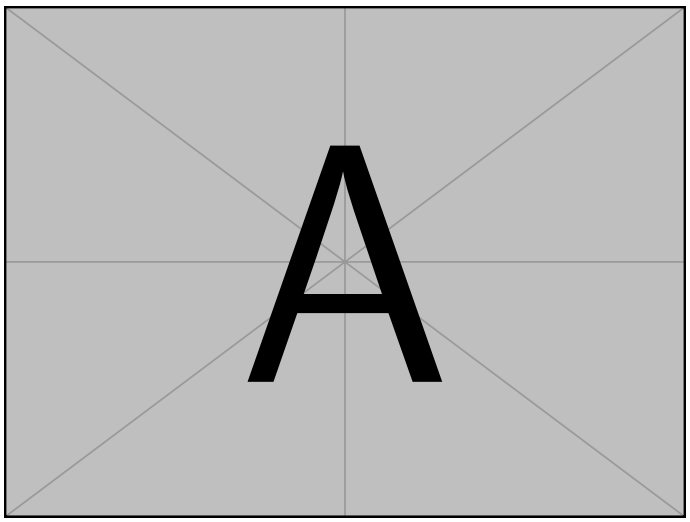


图 5.9　Example figure in appendix

# 第 6 章　总结与展望

## 6.1　总结

随着互联网的规模越来越大，网络流量的规模也急剧膨胀，网络应用的种类日益多样化，因此管理网络的难度也越来越大。传统的基于专家知识、人工生成规则的自动化运维具有很多缺陷，一是无法适应快速变化的网络流量环境，过度依赖规则库，而规则库需要不断人为添加新的规则；二是面对复杂网络情况时，很难根据流量的实时变化动态更新基线。

异常检测作为智能运维的关键环节，现有的异常检测算法具有很多缺陷，如面对海量数据规模时，无法或很难做到实时性；应用类型多导致的流量特征复杂，因此很难达到较高的检测率和较低的误报率；大多数异常检测算法仅能报告是否发生异常，难以对确定异常种类和定位异常来源给出指导性意见。

本文以清华大学校园网为例，进行网络流量异常检测算法和系统的研究。清华大学校园网是全球规模最大，架构最复杂，流量场景最多变的校园网之一，具有以下几个特点：

1. 用户规模大，流量峰值高。每天有 10 万台设备活跃在线，同时在线设备最多为 7 万台，峰值流量约为 30Gbps/s.
2. 用户应用类型多，远比一般的企业网复杂，在网络环境中几百种应用同时使用，这给数据分析带来了很多困难。
3. 异常流量是常态。扫描流量、攻击流量占比多。

总的来说，本文在以下几个方面对流量异常检测进行了研究：

1. 本文设计了一种基于特征之间关系图的循环神经网络算法 (FG-RNN)。在复杂的网络流量环境下，流量特征种类繁多，且特征之间的相关性会随着流量变化而变化。原有的异常检测算法通常直接利用提取的特征进行训练，本文提出的 FG-RNN 算法有效利用了特征之间的相关性信息，将其加入到神经网络的训练过程中。经过在开源数据集、真实数据集下分别进行的实验验证，本文提出的算法检测结果优于现有的基于机器学习、深度学习的算法。
2. 本文对基于特征之间关系图的循环神经网络算法进行了流式改造，使之能够满足当前实时异常检测的需求。为了应对海量的校园网流量数据和高速数据流，本文设计了一个基于 spark streaming 的实时异常检测系统。该系统分为输入模块、模型模块、检测模块三部分，输入模块负责将流量数据进行窗口划分、特征选择、特征抽取、合并计算，得到的特征矩阵与预训练得到的关

系矩阵一同送到模型中进行训练，模型中的参数每 2 小时更新一次，最后由检测模块对当前流量进行判别，并给出异常流量的类别。

3. 基于该实时异常检测系统，在公开数据集 UNSW-NB15 和 CICIDS2017 上用多个不同的衡量指标对 FG-RNN 和现有神经网络算法进行实验对比，以 F1-score 为例，相比其他算法的最好效果，

## 6.2　未来研究和展望

本文通过对清华大学校园网流量进行研究，提出了一种基于特征关系图的循环神经网络算法（FG-RNN），并设计和实现了一个基于 spark streaming 的实时异常检测系统。但是仍然有很多不足之处，本人认为未来的研究可以从以下几个方面着手。

1. 本文 FG-RNN 算法中使用比较简单的基于皮尔森相关系数的关系矩阵计算方式，该方式并不一定是最佳的关系图计算方式。在未来的工作中，可以尝试其他挖掘方法，如 Network Embedding 等。
2. 目前，大多数研究仅关注网络异常的检测和分类，而如何根据检测分类结果定位出异常的源头，指导运维人员进行快速有效地防御，甚至能够根据异常检测的结果自动化防御是未来研究的重点和难点。
3. 由于流量日志数据过于海量，系统给出的异常检测结果往往内容巨大，可能淹没运维人员真正关心的问题。
4. 与商业化的网络运维中心（Network Operation Center）平台中的威胁告警系统相比，基于各种算法的流量检测系统通常人力成本、机器开销更低，但是也有很多不足，未来可以将两者结合。

# 参考文献


[1] 中国互联网信息中心. 中国互联网络发展状况统计报告 [Z].

[2] Hawkins D M. Identification of outliers: volume 11[M]. Springer, 1980.

[3] Eskin E, Arnold A, Prerau M, et al. A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data[M]. 2002: 77-101.

[4] Hu W, Liao Y, Vemuri R. Robust support vector machines for anomaly detection in computer security.[C]// 2003: 168-174.

[5] Shon T, Kim Y, Lee C, et al. A machine learning framework for network anomaly detection using svm and ga[C]// Proceedings from the sixth annual IEEE SMC information assurance workshop. IEEE, 2005: 176-183.

[6] Kruegel C, Mutz D, Robertson W, et al. Bayesian event classification for intrusion detection[C]// 19th Annual Computer Security Applications Conference, 2003. Proceedings. IEEE, 2003: 14-23.

[7] Hawkins S, He H, Williams G, et al. Outlier detection using replicator neural networks[J]. 2002.

[8] Zhang Z. Hide : a hierarchical network intrusion detection system using statistical preprocessing and neural network classification[C]// Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, United States Military Academy, West Point, NY, 5-6 June, 2001. 2001.

[9] Poojitha G, Kumar K N, Reddy P J. Intrusion detection using artificial neural network[J/OL]. 2010: 1-7. DOI: 10.1109/ICCCNT.2010.5592568.

[10] Cormode G, Thottan M. Algorithms for next generation networks[M]. Springer Science & Business Media, 2010.

[11] Lakhina A, Crovella M, Diot C. Diagnosing network-wide traffic anomalies[J]. ACM SIGCOMM computer communication review, 2004, 34(4): 219-230.

[12] Lakhina A, Papagiannaki K, Crovella M, et al. Structural analysis of network traffic flows[C]// Proceedings of the joint international conference on Measurement and modeling of computer systems. 2004: 61-72.

[13] Lakhina A, Crovella M, Diot C. Mining anomalies using traffic feature distributions[J]. ACM SIGCOMM computer communication review, 2005, 35(4): 217-228.

[14] Nong, Ye, Qiang, et al. An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems (p 105-112)[J]. Quality & Reliability Engineering International, 2010, 17(2): 105-112.

[15] Callegari C, Giordano S, Pagano M, et al. Combining sketches and wavelet analysis for multi time-scale network anomaly detection[J]. Computers & Security, 2011, 30(8): 692-704.

[16] Hamdi M, Boudriga N. Detecting denial-of-service attacks using the wavelet transform[J]. Computer Communications, 2007, 30(16): 3203-3213.

[17] Pascoal C, De Oliveira M R, Valadas R, et al. Robust feature selection and robust pca for internet traffic anomaly detection[C]// 2012 Proceedings Ieee Infocom. IEEE, 2012: 1755-1763.

[18] Fernandes Jr G, Carvalho L F, Rodrigues J J, et al. Network anomaly detection using ip flows with principal component analysis and ant colony optimization[J]. Journal of Network and Computer Applications, 2016, 64: 1-11.

[19] Yeung D S, Jin S, Wang X. Covariance-matrix modeling and detecting various flooding attacks[J]. IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans, 2007, 37(2): 157-169.

[20] Xie M, Hu J, Guo S. Segment-based anomaly detection with approximated sample covariance matrix in wireless sensor networks[J]. IEEE Transactions on Parallel and Distributed Systems, 2014, 26(2): 574-583.

[21] Shannon C E. A mathematical theory of communication[J]. The Bell system technical journal, 1948, 27(3): 379-423.

[22] Behal S, Kumar K. Detection of ddos attacks and flash events using novel information theory metrics[J]. Computer Networks, 2017, 116: 96-110.

[23] David J, Thomas C. Ddos attack detection using fast entropy approach on flow-based network traffic[J]. Procedia Computer Science, 2015, 50: 30-36.

[24] Xie M, Hu J, Guo S, et al. Distributed segment-based anomaly detection with kullback–leibler divergence in wireless sensor networks[J]. IEEE Transactions on Information Forensics and Security, 2016, 12(1): 101-110.

[25] Li G, Wang Y. Differential kullback-leibler divergence based anomaly detection scheme in sensor networks[J]. 2012: 966-970.

[26] Rajasegarar S, Leckie C, Palaniswami M. Hyperspherical cluster based distributed anomaly detection in wireless sensor networks[J]. Journal of Parallel and Distributed Computing, 2014, 74(1): 1833–1847.

[27] Karami A, B G Z. A fuzzy anomaly detection system based on hybrid pso-kmeans algorithm in content-centric networks[J]. Neurocomputing, 2015, 149: 1253-1269.

[28] Carvalho L F, Barbon Jr S, de Souza Mendes L, et al. Unsupervised learning clustering and self-organized agents applied to help network management[J]. Expert Systems with Applications, 2016, 54: 29-47.

[29] Dromard J, Roudière G, Owezarski P. Online and scalable unsupervised network anomaly detection method[J]. IEEE Transactions on Network and Service Management, 2016, 14(1): 34-47.

[30] Syarif I, Prugel-Bennett A, Wills G. Unsupervised clustering approach for network anomaly detection[J/OL]. 2012, 293. DOI: 10.1007/978-3-642-30507-8_7.

[31] Moustafa N, Slay J. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)[C]// 2015 military communications and information systems conference (MilCIS). IEEE, 2015: 1-6.

[32] Özgür A, Erdem H. A review of kdd99 dataset usage in intrusion detection and machine learning between 2010 and 2015[J]. PeerJ Preprints, 2016, 4: e1954v1.

[33] spark. https://spark.apache.org/[Z].