杭州电子科技大学

硕 士 学 位 论 文

题　目:流统计特征在网络流量分类中的应用研究

研 究 生　朱 文 波

专　　业　计 算 机 应 用 技 术

指导教师　徐　明　教 授

完成日期　2014 年 12 月

杭州电子科技大学硕士学位论文

流统计特征在网络流量分类中的应用研究

研 究 生： 朱文波

指导教师： 徐 明 教授

2015 年 3 月

Dissertation Submitted to Hangzhou Dianzi University
for the Degree of Master

Application Research of Flow Statistical Features on Network Traffic Classification

Candidate: Zhu Wenbo

Supervisor: Prof. Xu Ming

March, 2015

杭州电子科技大学

学位论文原创性声明和使用授权说明

原创性声明

本人郑重声明： 所呈交的学位论文，是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不含任何其他个人或集体已经发表或撰写过的作品或成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。

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。

论文作者签名： 日期： 年 月 日

学位论文使用授权说明

本人完全了解杭州电子科技大学关于保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属杭州电子科技大学。本人保证毕业离校后，发表论文或使用论文工作成果时署名单位仍然为杭州电子科技大学。学校有权保留送交论文的复印件，允许查阅和借阅论文；学校可以公布论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存论文。（保密论文在解密后遵守此规定）

论文作者签名： 日期： 年 月 日

指导教师签名： 日期： 年 月 日

摘　要

准确识别网络流量的类型，是网络监控与管理领域的基础问题，也是研究中的热点与难点之一。随着网络上的流量越来越多，流量的类型越来越复杂，传统基于端口的分类技术与基于负载信息签名的分类技术越来越无法满足网络分类的要求与发展，因此基于流统计特征的网络流量分类技术也逐渐受到了更多的重视。本文在分析与总结国内外相关研究的基础之上，对基于流统计特征的网络流量分类技术进行了深入研究。本文的主要工作如下。

一方面，提出了一种基于网络流统计特征的分类方法。该方法从只包含数据包的原始流量中提取网络流与基于流的统计特征，随后使用基于 K-means 的聚类机器学习算法从网络流中获取流量簇，最后使用相关流与多数投票相结合的方式，对流量簇的类型进行标记。基于公开数据集的实验表明，本方法可以达到较高的分类精度与 F-measure 值。

另一方面，提出了一个基于流统计特征网络流量分类的一般性工作流程框架，并在此基础之上提出了一种网络流量分类优化方法。首先通过分析与总结前人相关成果，提出了一个基于流统计特征网络流量分类的一般性工作流程框架。其次在此一般性工作流程框架基础之上，先使用过滤器策略对分类过程中的统计特征进行筛选，再将封装器策略与启发式特征搜索算法相结合，应用序列前向搜索得到一个最优特征集，最终使用此最优特征集完成分类工作。基于公开数据集中不同数据片段的实验表明，该方法总是可以得到一组稳定的最优特征集；基于最优特征集的分类实验表明，优化后的分类方法在保证分类准确率的前提下，可以获得更好的时间效率。

综上所述，本文提出了一种基于流统计特征的网络流量分类技术，构造了一个一般性的工作流程框架，并基于此框架提出了一种网络流量分类的优化方法，借助本优化方法可以有效提升分类速度。

关键词：网络流量分类，统计特征，特征选择，启发式搜索

ABSTRACT

Identify the type of network traffic accurately is a fundamental problem in network monitoring and management field. And it is also one of the hotspot and difficulty topics of research. With more and more traffic transferred on the network, the flow types are more and more complicated. The traditional classification technology using port number or payload information is getting unable to meet the requirements and development of network traffic classification. So, the network traffic classification based on flow statistical features attracts more and more attention. Based on the summary of researchers at domestic and abroad, this paper make a deeply study on network traffic classification based on flow statistical features. The works of this paper are as follows.

On the one hand, we put forward a network traffic classification method based on statistical features. The method includes only the flow statistical features extracted from the raw traffic data. Then we use the clustering machine learning method based on K-means to obtain the flow clusters from the raw data. Finally, we combine the relevant flow and majority voting method to label the type of flow cluster. Experiments based on public datasets show that the method can reach high precision and F-measure value.

On the other hand, we propose a general framework about the working procedure of the statistical feature based network traffic classification, and put forward a optimization method for traffic classification on this framework. First of all, with the analysis and summary of the previous research result, we put forward a general working framework about the flow statistical based network traffic classification,. Secondly, based on this general framework, we use the filter strategy for feature selection in the first and the wrapper strategy of heuristic sequential forward searching in the next, find out an optimal feature subset at last. And we apply the optimal feature subset in traffic classification. Public datasets indicate this method can always get a stable optimal set. The classification results indicate the optimal feature subset could obtain a better time efficiency under the same classification precision.

To sum up, this paper presents a network traffic classification method based on flow statistical features and constructs a general framework of traffic classification. Based on this framework, an optimization method for network traffic classification is proposed. By means of the optimization method, the classification speed can be improved effectively.

Keywords： network traffic classification , statistical feature, feature selection, heuristic searching

目　录

第1章　绪论

1.1 研究背景与意义

互联网于 1969 年出现，从仅供美国军方使用的 ARPAnet 开始，经过四十余年的发展与扩充，逐渐变成了一张连通全球的网络。互联网取得了巨大的成功，在当今社会，互联网已经成为了一种重要的公共基础设施，在各个领域中发挥着越来越大的作用、占据着越来越重要的地位。它对人类文明传播产生了巨大的影响，在人类文明中充当着发展的助推剂。

互联网底层依托于 TCP/IP 技术，而在应用层则是开放、自由的，即经由互联网传输的数据对于 TCP/IP 协议来说是透明的。随着互联网的发展，可依托于互联网实现的功能与完成的工作越来越多，比如新闻浏览、游戏娱乐、文件下载、即时通讯、在线支付和商务交易等。因此，通过互联网传送的信息类型是由各种类型混杂而成的。据调查显示，几乎每天都会有由新应用产生的数据信息出现在互联网当中。

在许多情况下，网络服务供应商（Internet Service Provider, ISP）希望能对积极、有效地管理与控制用户通过各类应用产生的流量，对特定用户或特定应用进行各种保护或限制，对不同类型的流量给予不同的优先级，实现差异化的服务质量（Quality of Service, QoS）管理。那么，如何按照应用的类型对流经网络的流量进行分类，就成为了一个重要的研究课题。比如，对于某个特定的 ISP 来说，通过点对点（Point to Point, P2P）应用传输的数据，就占到了全部流量的百分之六十到百分之八十[1-2]。如果能够正确、有效的识别此类型的流量，并对 P2P 流量提供实施一定的限制，如降低 P2P 流量的转发优先级等，就可以为通过其他类型传输数据的网络用户提供更好的服务质量，利用已有带宽资源得到最优的服务效果[3-4]。因此，对网络流量的有效分类，是实现 QoS 管理的前提条件之一。

近年来，互联网的安全形势日益严峻，尤其是针对特定机构、公司与组织的攻击行为越来越多。攻击者不仅会单纯地使用各类病毒、木马和蠕虫等恶意程序进行攻击，而且会使用更复杂的方式对目标实施高级持续性威胁（Advanced Persistent Threat, APT）。由于防火墙、入侵检测系统等专用安全设备主要依赖于预设的规则与行为特征对入侵行为进行检测，因此不能对 APT 攻击进行全面有效的防范。而通过对网络流量的检测与分类，不仅能够实现对传统恶意程序的检测，还可以在一定程度上应对特定类型与模式的 APT 攻击。由于 APT 攻击普遍针对涉及国防安全、商业机密的机构进行，因此全方位的提升安全防护水平对此类机构

是十分必要的。

在大数据行为分析十分流行的今天，我们也可以借助其中的方法与工具，对网络流量进行分析，并从中获得一些抽象层次更高、价值内涵更丰富的信息。比如，通过分析特定类型的网络流量，我们可以得知该类型网络流量的主要用户、用户的网络状况、用户对该类型应用的使用习惯与使用偏好等通过原先的技术能力与技术手段无法获知的事实与情况。这不仅有利于帮助 ISP 提供更精细化、具体化与个性化的服务，也能帮助特定类型应用的提供者获得更多网络层面的用户信息。

现如今，在网络上传输的数据量十分巨大，网络结构也十分复杂，谁也无法完全记录网络上传输的所有信息。但是对感兴趣的网络节点的监控与管理，则是必不可少的。而其所依赖的关键技术——网络流量分类技术——必将扮演核心的角色，受到深入的研究，得到长久的发展[5]。

1.2 相关研究现状

目前，网络流量分类问题受到了十分广泛的探讨与研究。由于各类网络应用具有自身的特性，因此对网络流量的区分粒度也存在很多种类。但总的来说，主要使用以下三种层次的分类粒度[6]：

(1) 包级别（Packet Level）

主要关注包（Packet）的特征，主要特征有数据包大小、连续数据包的到达时间间隔等，这主要是网络层（IP 层）上的概念。

(2) 流级别（Flow Level）

主要关注流（Flow）的特征，一般来说，流通常由一个五元组来唯一描述，五元组由源 IP 地址、源端口号、目的 IP 地址、目的端口号、传输层协议五个字段组成。一个流可以是 TCP 流，也可以是 UDP 流，主要特征有流中的包个数、流的大小等，这主要是运输层上的概念。

(3) 流量组合级别（BoF Level）

主要关注网络端点之间的流量组合（Bag of Flow, BoF），通常由一个端点 IP 地址、端点端口号、传输层协议组成的三元组来唯一描述，通常适用于更粗粒度的、骨干网上的流量统计特性，主要特征有 BoF 中的流的个数等。这主要是应用层上的概念。

在上述三种粒度中，流级别（Flow Level）的流量分类受到了最为广泛的研究。由于网络流量分类的根本目的在于从运输层数据中区分出各类应用在运输层上产生的流量，所以在运输层数据的基础上进行分类的流级别分类技术获得了最广泛的关注与最深入的研究。

目前，对网络流量进行分类的方法主要包括三类[7]：基于端口号的方法、基于负载签名信息的方法，以及基于流统计特征的机器学习分类方法。下面对这三类分类方法逐个进行介绍。

(1) 基于端口号的分类方法

常用熟知端口[8]由 IANA 分配，一般来说，我们可以依赖端口号，对应用层的数据进行直接地识别与分类。这种分类方法的思想非常简单，既然运输层的端口号是用来标识应用层协议，那么就直接依赖 TCP 或 UDP 连接中目的端口号进行分析，将熟知的端口号与其对应的应用类型建立起一个一一对应的映射关系。随后，分类器只需要找到一个运输层连接中一个数据包，并从这个数据包中取出目的端口号字段，就可以完成对网络流量类型的分类过程。这种分类方法的实现原理非常简单，只需要一张映射表即可，适用场景主要包括高速网络上的实时分类，如基于端口号的过滤器式防火墙。比如 Linux 中的网络流量管理工具 iptables 的底层原理，就是基于端口号的流量分类。

2003 年，Fraleigh 等[9]使用此方法构造了 IPMON 系统，并使用此系统对 Sprint 公司（一家美国大型 ISP）的骨干网进行了深入细致的研究，详细对比了 Web、P2P 和未知流量占总流量的比重，还对全网的平均往返时延、失序率等数据进行了统计，将使用此方法进行的分类研究工作做到了极高的水准。

但是很显然地，这种分类方法在很多方面都会受到制约，比如，目前很多 P2P 类型的应用在传输数据时，会使用随机端口或动态端口技术；当今，还有许多熟知端口列表之外的应用在互联网上产生大量的流量；此外，对网络层负载信息的加密，也会导致分类器难以获得运输层信息。基于以上种种原因，仅使用端口号进行分类的技术路线一般只能获得较低的分类准确率[10]。

2004 年，Sen 等[11]对 Gnutella、DirectConnect 和 Kazaa 等多个 P2P 协议进行的分类实验表明，按照各个协议划分，使用熟知端口号进行传输的 P2P 类型的流量，只占到总流量的 28%-62%不等。

2005 年，Moore 等[12]使用一个从校园网上采集的流量数据集，对 FTP、SSH 和 IMAP 等超过 20 种的常见协议进行分类。实验结果表明，使用 IANA 熟知端口列表实现的，基于端口号的分类，可以正确识别的流量仅有 69%，还有 30%的流量完全无法被识别，只能被归类为未知流量。

(2) 基于负载签名信息的分类方法

为了避免基于端口号的分类方法的种种弊端，基于负载签名信息的分析方法逐渐被研究者所重视[13]。该方法的基本思想是，通过分析数据包中的负载信息部分是否包含已知应用的特定签名信息，来完成对整个流的分类。由于直接审查了流量的内容，因此本方法与使用端口号的方法相比，可以取得的分类精确率一般更高。另一方面，由于某些协议只在一个流的前几个数据包中含有特定的签名信

息，因此该分类方法一般只使用一个流的前几个数据包中的负载信息，这也保证了本方法整体上的高效性。

2004 年，Sen 等[11]使用人工分析的方法，从 Gnutella、eDonkey、DirectConnect、BitTorrent 和 Kazaa 等五种被广泛使用的 P2P 协议当中，提取了基于正则表达式的签名信息，并使用这些签名信息对这 5 种 P2P 协议进行了精确地识别。实验表明，该方法对 P2P 协议的分类结果中，假否定率（*FN*）最多只占该类型总流量的 9%。

2008 年，Park 等[14]提出了一套自动提取与生成签名信息的算法。通过签名切分、合并与重连接的方法，使算法无需对所有流量进行穷举搜索，就可以较容易的得到准确的应用程序签名信息。随后作者将使用此算法生成的签名与使用人工方法得到的签名进行对比，可以发现使用两种方法得到的签名信息基本一致，甚至找到了使用其他方法时未能找到的签名信息。最后的实验结果分析表明，使用本方法进行流量分类，可以达到 97%的全局准确率。

基于负载签名信息的分类方法是一套行之有效的方法，Linux 下的著名流量过滤工具 L7-filter[15]就是基于负载签名信息实现的，其中的 L7 代表 Layer 7，即 OSI 模型中的应用层。它能使 Linux 的 iptables 工具支持应用层的流量分类监控,较为精确地管理与控制由 P2P、IM 等应用产生的流量。

虽然基于负载签名信息的分类方法避免了基于端口号的分类方法所带来的一些问题，但仍存在一定的局限性与未解决的问题：本方法只能识别那些已知类型的流量，如果事先没有存储该类型的签名，就无法识别该类型；如果签名信息发生一定程度的变形，本方法也可能无法正确的完成分类；只能识别非加密类型的流量；它的流量识别过程比基于端口的分类方法复杂，并需要使用较大的存储空间完成对签名信息的存储；此外，本方法还可能涉及安全性方面的问题，甚至可能引起对隐私权的侵犯。[6]

(3) 基于流统计特征的机器学习分类方法

随着对网络流量分类问题研究的发展，一种基于流量统计特征来识别流量应用类型的新方法被提出[16]。通过观察，对于特定类型的应用产生的数据包与数据流，存在一定的统计特征，这些特征主要包括数据流的持续时间、数据包之间间隔时间、包的长度和流的字节总数等等。我们假设这些特征都是唯一的，且可以用这些特征来有效的识别流量的类型[17-18]。由于特征种类多、数据规模大、系统复杂度高，通过简单的映射关系或查找关系完全无法实现本方法。因此，很自然地，基于机器学习的方法被引入到了流量分类问题中。

机器学习的过程主要包括下面三个步骤：首先是构建分类模型，即建立具体的分类器；其次是对分类器的训练或调适；最后是使用分类器完成分类工作。目前，在流量分类领域使用的机器学习方法主要包括有监督的机器学习方法、无监督的机器学习方法和半监督的机器学习方法共三大类。下面就前人完成的实验与

使用的方法进行简单的介绍。

有监督的机器学习方法种类很多，每种方法具备不同的特性，适用的问题也不尽相同。2005 年，Moore 等[19]使用朴素贝叶斯法构建了一个网络流量分类器，分类器中选取了流持续时间、端口号、包间隔时间和包间隔时间的傅里叶变换等特征。之后将数据集中的流量采用人工标注的方式，得出训练样本中的网络流量的基准分类。最后使用此分类器对 WWW 类型的流量进行分类，得到了高达 98% 的真肯定率（*TP*）。

2009 年，Este 等[20]对每个网络类型分别使用支持向量机（Support Vector Machine, SVM）算法，为每种类型单独构造了一个 SVM 分类器。SVM 是有监督的机器学习中最常见、最通用、也较为行之有效的方法。SVM 的基本工作原理是：使用核函数，将分布于非欧空间的样本点变换至欧式空间中，或完成相反的空间转换；再通过超平面对欧式空间样本点的划分，将对待分类的样本进行二分类，即判定样本点是某特定类型或不是某特定类型；使用松弛变量等方法，将分类器的误判率（包括假肯定率 *FP* 与假否定率 *FN*）控制在一定的范围内。这篇文章中，作者将对多个网络流量类型的多分类问题，转换为多个二分类问题，即分别判断某个流量是否为类型 *A*、是否为类型 *B*……直至对所有预设类型的判断结束，不符合所有预设类型的流量即可判为未知流量。若一个网络流符合多个预设类型，则对其进行更进一步的处理。本方法在分类之前，需要使用数据集的一部分作为训练集，并使用数据集的其他部分作为验证集来计算分类的准确率。基于 LBNL 数据集[21]中分类结果显示，使用这个方法对各个类型进行分类的真肯定率可以达到 69%-100%。

通过上述研究可以看到，有监督的机器学习方法依赖于训练集对模型的调适，因此训练集中标记类型的正确与否将直接影响分类结果，而一般来说训练集中的标记都需要人工标注完成。为了保证分类结果不受人工影响，学者们进一步的提出了无监督的机器学习方法。无监督的机器学习方法一般称为聚类，聚类算法一般又可以分为软聚类和硬聚类[22]，二者的区别是，软聚类算法只给出落入各聚类簇的概率，而硬聚类算法直接给出最终落入的聚类簇。

2004 年，McGregor 等[23]人使用包间隔时间、包大小、流的总字节数和连接持续时间等作为统计特征，通过无监督的机器学习对 TCP 流量进行了分类。作者首先使用期望最大化（Expectation Maximization, EM）软聚类算法，通过多次反复计算得出对每个流的最大似然估计，并选取概率最大的估计聚类簇作为流的结果类型。作者随后又对各网络流量类型的聚类簇进行了可视化分析，并得到了聚类簇与端口号之间的关系。

2006 年，Erman 等[24]比较了 K-均值（K-Means）、基于密度的空间噪声聚类（Density Based Spatial Clustering of Applications with Noise, DBSCAN）和自动类

（Auto Class, AC）这三种硬聚类算法在网络流量分类问题中的效果。实验表明，K-均值和空间噪声聚类的分类准确率较自动类算法的准确率低，其中自动类算法可以达到平均 92%的准确率。

非监督的学习方法虽然保证了分类结果不受人工影响，但是标记过程大多依赖于机器，因此没有一个统一的标准来对准确性进行度量与比较。为此，学者们又提出了半监督的机器学习方法。半监督的机器学习方法使用的数据集包括已标记样本和未标记样本两部分。首先使用聚类算法将待分类数据分入不同的簇中，然后通过包含已标记样本的流，完成从簇到类别的映射。

2013 年，Park 等[25]利用网络通信的局部性原理，提出了 SSIP（Same Server IP）的概念。SSIP 的思想是，相同的 IP 地址上，一般会提供同一种类型的服务，因此也会产生同一种类型的流量。利用这种思想，就可以使得已知的分类结果被重复使用。作者使用从校园网上采集获得的数据集，通过对比发现，服务器 IP 记录一般只有不超过 5 万条记录，而同一时间的数据流最多可以达到 20 万条。使用 SSIP 作为已标记的分类样本，不仅可以对未标记分类样本进行有效的标记，还可以使用尽量少的空间完成标记工作。

2014 年，Zhang 等[26]实现了一个健壮的网络流量分类系统框架。所谓健壮，就是系统能有效的识别未知类型的流量，而不会把这些流量错误的分入已知类型当中。本文第一次提出了 BoF（即流量组合，上文有介绍）的概念。本文给定了一个三元组：端点 IP 地址、端点的端口号和传输层协议。对于所有的具有相同的此三元组的流，我们即认为是一个流量组合。使用聚类算法分别标记流量组合中所有的流，再选择大多数流被标记的类型作为整个流量组合的类型，即可实现从未标记流量到已标记流量的转换。

半监督的学习方法一般只利用少量的标记样本即可实现对聚类簇进行类型标记，并可以使用多种方法有效的完成标记工作。这样做不仅可以减少由于人工标记引入的误差，还可以压缩聚类过程消耗的时间，提高分类算法的性能。但由于这类方法提出的时间较晚，因此还有一定的研究空间。

1.3 论文研究内容与方法

本文使用基于流统计特征的网络流量分类方法，对网络流量的分类方法进行了探讨，并对已有分类方法进行了一定程度的优化。本文主要进行了以下几个方面的研究：

第一，对网络流量分类问题进行了深入的探讨，讨论了分类问题的一般性，并结合网络流量分类的应用背景分析了特殊性；

第二，描述了网络流量分类问题中涉及的相关技术，主要包括如何使用

libpcap/winpcap 开发包对网络流量进行捕获、存储与处理，如何使用无监督的机器学习算法对网络流量进行分类，并简要介绍了各类特征选择的方法；

第三，构建了网络流量分类问题的一般性框架，给出了基于流统计特征的网络流量分类系统模型，描述了如何有效的标记聚类簇的网络流量类型。之后提出了一组常用的候选特征集合，并使用此特征集合在上述系统模型中完成了实际的网络流量分类工作。

第四，针对常见的网络流量统计特征集中缺乏高阶统计量的问题，引入了偏度、峰度等高阶统计量，分析了使用高阶统计量的原因，构造了与流行为有关的高阶网络流量统计特征，并使用包含有高阶统计特征的特征集作为分类工作的初始特征集。

第五，针对分类问题中的特征选择子问题，对网络流量分类的问题的一般性框架进行了一定的优化。结合前人在特征选择领域上的研究，本文主要使用启发式搜索的特征选择，对分类中所使用的特征进行选取，筛去了有重复信息的冗余特征，排除了对分类效果有负面影响的特征，保留了直接影响分类结果的核心特征，作为最有特征子集。最优特征子集相对原始特征集来说，规模小、与分类器结合更紧密，使用此最优特征子集，可以使分类问题在一定程度上得到简化，从而减少分类时间。实验结果从分类精度与分类时间两方面表明，该方法有效的对原方法进行了优化。

1.4 论文组织结构

本文共分为五章，组织结构如下[27]：

第一章，绪论。介绍论文的研究背景和意义，回顾了网络流量分类领域当前的研究工作与最新进展，提出了本文主要研究内容和方法，并说明了论文的组织结构。

第二章，网络流量分类相关技术。本章首先给出了网络流量的概念与相关操作方式，随后介绍了特征选择领域中常用的几类特征搜索方法，接着对机器学习算法进行了一定的介绍，最后给出了本章小节。

第三章，基于流统计特征的网络流量分类技术。本章首先主要对基于流统计特征的网络流量分类模型进行了介绍，随后提出了网络流量分类的一般性工作流程框架，最后使用了常见的特征集与聚类算法，在上述工作框架下实现了基本的分类工作。

第四章，基于特征选择的网络流量分类技术优化。本章首先介绍了特征选择算法的评价函数，并详细的说明了一种启发式搜索策略的工作过程，随后使用多个特征选择策略结合的方式完成了特征集优化的工作，最后通过实验验证了选取

得到的特征子集在分类工作中的高效性。

第五章，总结与展望。主要总结了本文的取得的成果与做出的贡献，也提出了本文中的不足之处，并给出了进一步研究的方向与对未来工作的展望。

第2章　网络流量分类相关技术

在网络流量的分类问题中，主要涉及网络流量处理、分类算法和特征选择算法等相关技术，其中，网络流量处理技术与分类器技术是网络流量分类工作中的关键技术，这两项技术将直接对分类结果产生根本性的影响。本章系统的介绍了网络流量分类问题中使用的各类技术，并对各类技术按照一定的关系进行归类。此外，本章还介绍了一些常见的、成熟的网络流量处理工具，并对这些工具的使用方法进行了简要的介绍。

2.1 网络流量

2.1.1 网络流量的概念与特点

网络流量一般是指经过交换机、路由器和网关等网络设备的数据。由于这些数据往往只使用网络设备中的缓存，并立即通过接口被发送至其他网络设备，对这些数据的接收与发送就如同流动的车辆，所以一般我们称之为网络流量。

网络中的数据包一般都带有时间戳信息，另一方面，网络中的数据包往往只会存入网络中各类设备的缓存中，此数据包一旦被发往某个接口并完成此发送过程，这个数据包就会被从该设备的缓存中清除，因此网络流量具有即时性与易失性的特点。一般来说，一个数据包只能在特定的时间被捕获，即一旦错过此数据包，就无法在未来重新捕获与原数据包完全一致的数据包。

2.1.2 网络流量的捕获与存储

一般来说，有两种网络流量捕获架构，第一种是在网络设备上直接捕获，另一种是使用镜像技术将流量发送至专用的捕获机。这两种捕获方式各有利弊，下面逐一介绍这两种方式。

(1) 直接捕获

在网络设备上直接捕获网络流量的架构，一般会在网络服务器等非专用、可编程的网络设备下部署。这类设备一般具备开放式的操作系统，因此可以在操作系统上直接部署网络流量捕捉程序。对于现代服务器来说，由于此类设备的网络通信功能一般依赖于各类网卡来完成，网卡往往会插入到 PCI 或 PCI-E 插槽中，因此这类设备一般都会在操作系统中使用一定的内存地址[28]。通过直接访问相应的内存地址，即可获取通过该网卡传输的所有数据包。基于上述方法，之前的研

究者开发了一些实用工具与相关的类库，通过使用这些工具或调用类库中的方法，我们就可以实现对网络流量的捕获。下面介绍最常见的网络流量类库 libpcap。

libpcap 是由加州大学伯克利分校劳伦斯（Lawrence, UCB）实验室与其他贡献者联合开发的一个基于 C 语言的网络流量处理类库，他们在 1994 年 6 月 20 日推出了最早的 v0.0 版[29]，这个版本提供了一些最基础的捕捉与分析的功能。在随后的 20 年之内，libpcap 的版本持续更新，逐步适配了更多的平台与系统。当前的 libpcap 更新到了 1.6.2 版本，并且受到 BSD 3-Clause 开源发布许可证的保护。其中，对于 Windows 平台，就有 winpcap 这个免费、公开的类库。Windows 平台下的用户可以通过在编译时连接类库中提供的 wpcap.lib 文件，并将类库中提供的头文件通过 include 预编译命令引入到自己的 C 语言程序代码中，即可完成对 winpcap 的调用，进而可以使用类库中提供的丰富的数据包捕捉与其他数据包处理功能[30]。这里详细展示如何在 VS2010 环境中引入 winpcap 这个类库：

首先，我们应当将下载并解压缩 winpcap 类库。用户可以在此地址下载 winpcap 的开发包 wpd（http://www.winpcap.org/install/default.htm），并将下载得到的压缩包文件解压缩至本地硬盘。本例中，笔者将开发包解除压缩后存放在了“F:\LabTool\WpdPack_3_0\wpdpack”这个路径下。

之后，我们应当在 VS2010 的 C/C++语言开发环境下的包含目录与库目录中添加 winpcap 开发包中提供的包含文件目录与库文件目录，即“F:\LabTool\WpdPack_3_0\wpdpack\Inlcude”和“F:\LabTool\WpdPack_3_0\wpdpack\Lib”。如图 2.1 所示，在 VS2010 的 C/C++开发环境中，依次点击“视图”-“属性管理器”，即可打开图中左侧的属性管理器窗口，再双击“Microsoft.Cpp.Win32.user”即可打开图中右侧的相关属性页，随后在“VC++目录”标签页中修改“包含目录”和“库目录”两个条目即可。请注意，在修改这两个条目的时，请务必不要将之前的内容删去，而是应当在之前内容之后加入英文分号，并将上述包含文件目录与库文件目录填入分号之后。

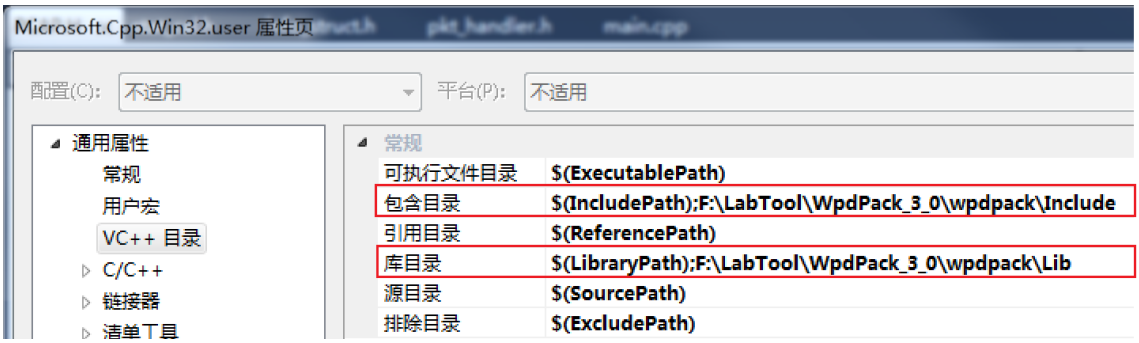

图 2.1 在 VS2010 的 C/C++语言开发环境下引入 winpcap 开发包

最后，我们应当在需要使用 winpcap 类库的解决方案或工程中添加 wpcap.lib 库文件，才能正确的引用 winpcap 类库进行开发工作。具体的操作方法是：右键单

击解决方案资源管理器中的工程名，点击“添加”-“现有项”，然后进入存放 wpd 的目录中，选择 lib 目录下的 wpcap.lib 文件即可。

除了在 Windows 平台下使用 winpcap，我们还可以在 UNIX/Linux 平台下使用原生的 libpcap 类库，具体的使用方法在这里不做介绍。此外，更进一步的，我们还可以在 Linux 平台下直接使用 libpcap 开发组为我们提供的命令行工具 tcpdump[31] 完成流量捕获的工作。tcpdump 工具基于 libpcap 开发，不仅是一个使用 libpcap 开发包对网络流量进行监控的实例，也是一个便于直接使用的工具，更是一个全面展现 libpcap 开发包中各类功能的平台。通过在 Linux 平台下安装 tcpdump 软件包，并调用 tcpdump 命令监听指定的网卡，就可以实现对网络流量的捕获。在 Bash 下，调用 tcpdump 开始流量捕获的具体命令如下：

```
tcpdump -i eth0 -c 100 -w cap.dump
```

此命令可以调用 tcpdump 监听网卡 eth0，并将通过此网卡的前 100 个数据包存入 *cap.dump* 文件中。其中参数-*i* 指定网卡，参数-*c* 指定需要记录或显示的数据包个数，参数-*w* 指定捕捉后存入的文件。tcpdump 命令还有一些常用的参数，如参数-*s* 指定捕获的每个数据包的截断长度字节数，参数-*l* 可以使用管道命令对捕获结果进行重定向等。

上述各类对数据包的捕获与处理方法一般都是针对有一定计算机知识与操作技能的用户而准备的，是否有一种能让所有的计算机用户都能直观的操作与使用的工具呢？答案是有的，这就是大名鼎鼎 Wireshark。Wireshark 的同样有着悠久的历史，其在早年间使用的名称叫做 Ethereal，由 Gerald Combs 于 1998 年开发，并使用 GPL 开源发布许可证进行了发布。由于软件名称的商标权等原因，作者于 2006 年使用 Wireshark 这个新的名称继续开发并维护着之前的项目，并受到了大量用户的青睐[32]。Wireshark 是一款多平台的图形界面网络流量捕捉工具，除了 Windows 平台，还实现了在 Linux Gnome 图形化框架下的图形化。下面就对如何在 Windows 平台下使用 Wireshark 实现网络流量的捕获进行介绍。

首先，打开 Wireshark，直接点击工具栏上的“Start a new live capture”按钮，开始捕捉，如图 2.2 所示。

之后，就可以看到捕获得到的结果，如图 2.3 所示。捕获得到的每个数据包都占据 packet list 中的一行，点击数据包，就可以在下方的 packet details 部分和 packet bytes 部分查看数据包的详细信息，其中 packet details 中显示对数据包的分析与解释，而 packet bytes 部分用 16 进制与 ASCII 码的方式，逐字节的显示数据包中包含的信息。点击或选中 packet details 和 packet bytes 部分的内容，可以在另一部分中对应的高亮显示被点击或选中的部分。

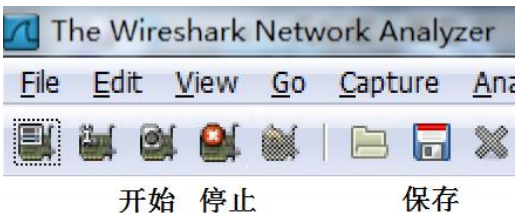

图 2.2 使用 Wireshark 开始、停止捕捉或保存已捕获的网络流量

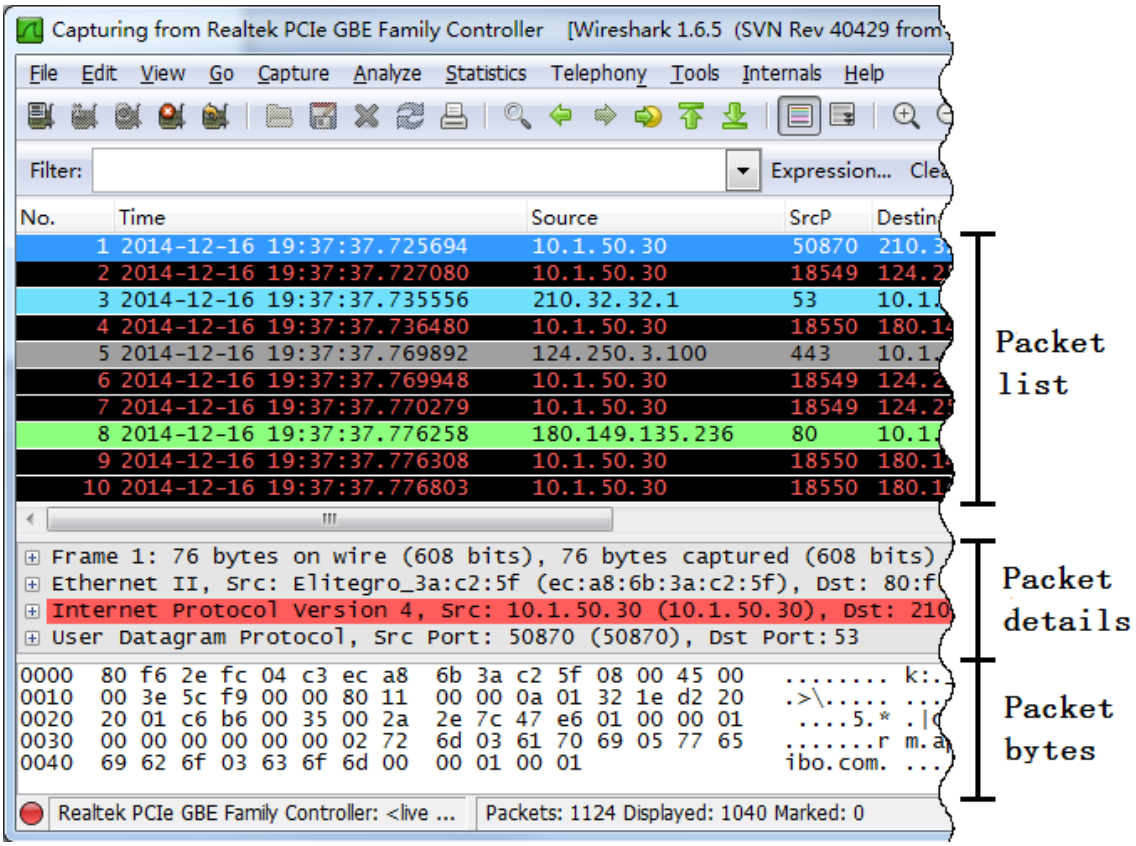

图 2.3 捕捉得到的数据包与第 1 个包及其包含的信息（隐藏了部分信息）

当捕获得到足够多的数据包或达到一定的时长后，即可点击工具栏上的“Stop the running live capture”按钮停止捕获。如果希望能保留本次的捕获结果，那么还可以点击工具栏上的“Save this capture file”按钮来完成对本次捕获结果的保存，保存得到的默认文件类型为 pcap 类型。上述操作对应的按钮如图 2.2 所示。

(2) 基于镜像技术的捕获

有时我们需要在没有开放式操作系统的目标设备上执行网络流量捕获任务，如需要在交换机、路由器等设备上执行网络流量捕获，那么一般会使用镜像技术来实现捕获。端口镜像技术是指，发往特定端口的所有数据包，都会被发往镜像端口，即通过镜像端口可以获取所有通过原端口的数据包。由于该技术往往会被预先嵌入在交换机、路由器等网络设备的固件上，因此我们可以通过配置网络设备的方式来使用此功能。比如在华三 H3C S5120 交换机上，可以通过表 2.1 中的配置命令，来完成对端口 1/0/1 的镜像，并将镜像流量发往端口 1/0/10 上。

表 2.1 在华三 H3C S5120 交换机上应用镜像端口功能的配置命令

```
#进入系统控制界面
<DeviceC> system-view

#创建本地镜像组 mirroring-group 1
[DeviceC] mirroring-group 1 local

#为本地镜像组 1 配置源端口和目的端口，对进出源端口的双向流量进行镜像
[DeviceC] mirroring-group 1 mirroring-port gigabitethernet 1/0/1 both
[DeviceC] mirroring-group 1 monitor-port gigabitethernet 1/0/10

#显示所有镜像组的配置信息
[DeviceC] display mirroring-group all
mirroring-group 1:
    type: local
    status: active
    mirroring port:
        GigabitEthernet1/0/1 both
    monitor port:
        GigabitEthernet1/0/10
```

在完成镜像端口配置之后，我们应当将监控端口与专用的捕获机进行连接，并在捕获机的网卡上开启混杂模式，监听并不加选择的接受所有发往本网卡的数据包，再在捕获机上配合使用前文中介绍的、基于 libpcap/winpcap 开发的各类捕捉工具，才能最终实现对网络流量的捕获。此外，还可以使用专用的数据捕捉卡，通过与镜像端口连接的方式，完成数据捕捉，在此对这种方式不作详述。

2.1.3 网络流量的预处理

网络流量的预处理工作，主要指的是通过对数据包的分析，提取流级别的信息。对于特定的网络流量，如果对数据包只是进行简单的罗列、计数等线性处理，是无法从原始流量中总结、分析与抽象出网络中具体的会话行为的，因为不同的会话并不是按照时间顺序通过网络的，所以原始流量从时间顺序上看，是由多个会话混杂组合而成的。因此，我们必须按照一定的规则，将不同的数据包分别归入不同的会话中，从流级别的视角去分析与提取各类不同会话的特征。因此，对网络流量的预处理，主要分为以下两个部分的技术：第一，是按一定格式读取与

解析原始流量或原始流量文件；第二，是从原始流量的数据包序列中整理得到流，并提取流的内在特征。下面对这两项技术分别进行详细介绍。

(1) 读取原始流量文件

无论是使用直接捕获得到的流量还是原始流量文件进行解析，都要遵循在捕获时所使用的协议格式。对于常见的、由 libpcap/winpcap 捕获得到的 pcap 类型来说，其格式如图 2.4 所示。其中 Pcap Header 位于 pcap 文件的开始，总长度为 24 字节；Pacp Header 之后为这个 pcap 文件中存放的多个数据包，图中包含的数据包个数为 n，每个数据包都由 Packet Header 部分和 Packet Data 部分共同组成，其中 Packet Header 在前，长度为 16 字节，Packet Data 在后，其长度由 Pacp Header 部分的 caplen 字段指示，单位为字节，其内容就是捕获得到的数据包。

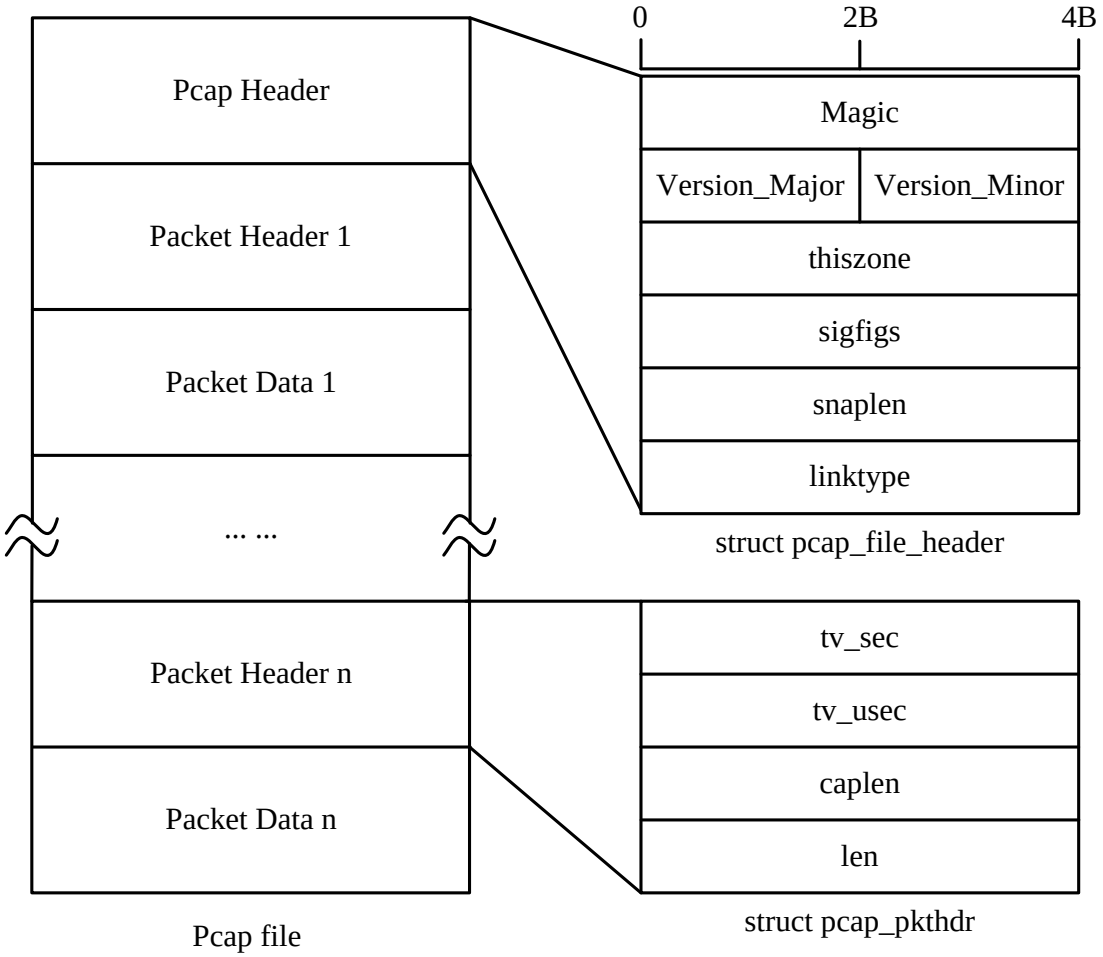

图 2.4 Pcap 文件的格式

图 2.4 中的各个字段的具体定义（C 语言）如表 2.2 所示。

表 2.2 Pcap 文件格式中各个字段的定义

```
struct pcap_file_header {          /* Pcap 文件的头 */
    bpf_u_int32 magic;             /* 魔数，这 4 字节的 16 进制编码为 0xD4C3B2A1 */
    u_short version_major;         /* 主版本号，默认为 0x0200 */
    u_short version_minor;         /* 副版本号，默认为 0x0400 */
    bpf_int32 thiszone;            /* 格林威治时间与当地时间的偏移量，一般为全 0 */
    bpf_u_int32 sigfigs;           /* 时间戳精度，可以使用全 0 */
    bpf_u_int32 snaplen;           /* 数据包的最大长度，一般可以使用 0xFFFF0000 */
    bpf_u_int32 linktype;          /* 数据链路层类型，以太网为 0x01000000 */
};

struct pcap_pkthdr {               /* 数据包的头 */
    struct timeval ts;             /* 时间戳 */
    bpf_u_int32 caplen;            /* 捕获得到的数据包长度 */
    bpf_u_int32 len;               /* 数据包的真实长度 */
};

struct timeval {                   /* 时间戳的结构 */
    long tv_sec;                   /* 秒数，将 1970 年 1 月 1 日 0 时 0 分 0 秒记为 0 */
    long tv_usec;                  /* 微秒数 */
};
```

(2) 提取流的内在特征

从数据包中提取数据流的过程，是一个将线性结构转化为非线性结构的分析过程[33]。在这个过程中，首先需要将顺序排列的数据包按照一定的规则转化成由不同流标签与包信息列表对应的结构，然后对每个流标签对应的包信息列表进行整理计算，得到流标签与流信息一一对应的结构。举例来说，我们一般将<源 IP，源端口号，目的 IP，目的端口号，运输层协议>这个五元组作为标识流的键值，即以此来唯一标识一个流[34]；此外，我们感兴趣的信息是每个数据包的包长度、数据包在应用层的有效负载长度与包的到达时间，那么我们就可以从原始流量的每个数据包中提取得到这些信息并存入相应流标签对应的包信息列表；最后，对每个流标签对应的包信息列表进行计算，得出我们感兴趣的统计信息，如特定流中包长度的最大值、最小值、算术平均值和方差，与相邻数据包到达时间间隔的最大值、最小值、算术平均值和方差等等信息，这些信息是基于包的，但是却属于流的特征，因此可以将这些特征一一对应于每个流的标签。上述过程如图 2.5 所示。

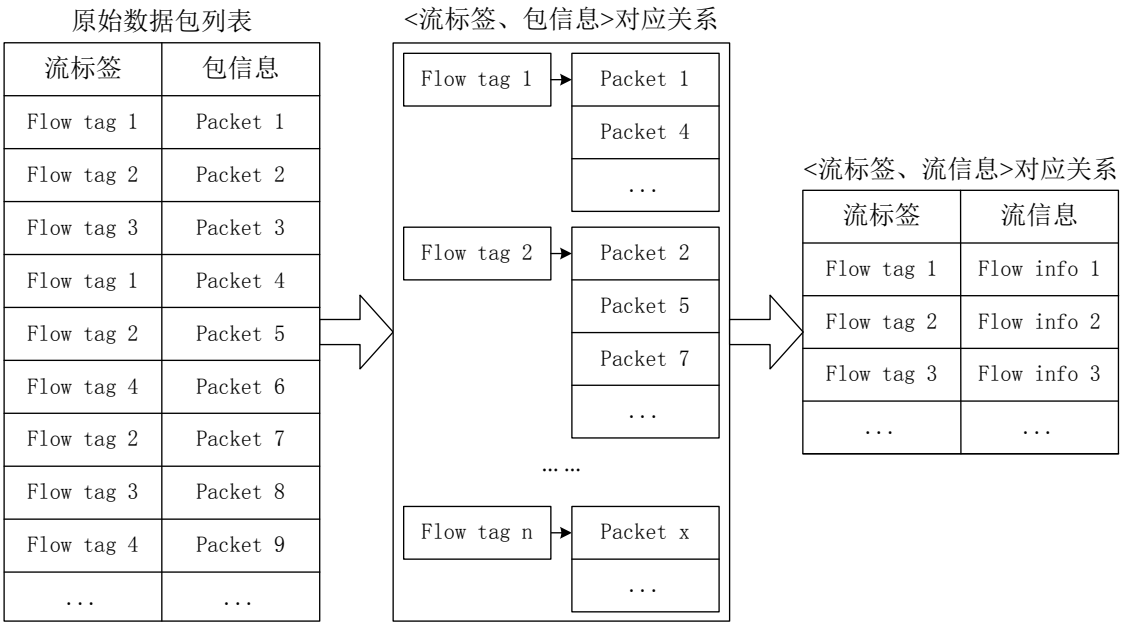

图 2.5 从数据包序列转化为流的过程

2.1.4 网络流量的类型

网络流量的类型，一般指的是生成此数据包或网络流的应用的类型，即 OSI 七层网络模型中位于应用层的程序类型。每一个流，都唯一的对应一个类型，分类工作的根本目的与最终结果，就是将流与产生它的应用类型一一对应起来。

一般来说，这些流量类型中包含一些较为常见的、被视为网络协议的类型，也包括一些常见的或大量用户使用的应用程序。其他的类型中，还包括一些浏览器程序或服务器程序的类型、非常见的应用类型或是目标主机的操作系统类型，也可以包含在对网络流量类型的范围之内。我们可以根据分类的目的，使用不同粒度级别的、不同侧重方向的或者不同应用范围的类型集，来构成目标类型集。表 2.3 列出了常见的网络流量的类型。

表 2.3 网络流量的类型

网络流量的类型	类型名称
被视为网络协议的类型	HTTP、SMTP、FTP、DNS、IMAP、POP、HTTPS、SSH
常见应用程序	WinSCP、BitTorrent、eDonkey、MSN、QQ、Skype
其他	Chrome、Firefox、IE、Quake、WOW、Windows

2.2 特征选择方法

特征选择（Feature Selection）是指从原始特征集中提取一个特征子集，使得此特征子集在某种分类评价标准下达到最优。在分类问题中，特征选择的目的是从原始特征集中提取一个特征子集，使通过此特征子集所构建的分类模型达到与

之前近似的、甚至更好的分类精度。这样做不仅可以提高模型的泛化能力，还可以有效降低分类时间复杂度。

特征选择的问题描述为：给定一个机器学习算法 L 和一个数据集 S，其中 S 来自一个特征集为 $F=\{F_1, F_2, ..., F_n\}$且具有标记类型 $Y=\{Y_1, Y_2, ..., Y_n\}$的样本空间，则一个最优特征子集 F_{opt} 就是使得评价函数 $J=J(L, S, F_{opt})$达到最优的特征子集[35]。

目前，用于特征选择的算法主要有三类：完全搜索、随机搜索和启发式搜索。完全搜索算法通过遍历所有的特征子集，找到使得评价函数取到极值的特征子集。随机搜索算法是在一定的规则下，随机地选取特征子集来判断，最终在这些随机选取的特征子集中选择使评价函数取得极值的特征子集。启发式搜索算法是根据某种启发式策略找到一个使评价函数取得极值的特征子集。但实际上，随机搜索与启发式搜索最终得到的往往都只是全局次优解[36]。

这三种搜索算法中，完全搜索总能得到最优特征子集，但由于其时间复杂度高，所以只有当特征集中特征数量较少时才会使用该方法；随机搜索由于搜索时间或次数受到了限制，所以有可能无法搜索到最优子集，也可能在最优子集出现后仍旧进行搜索；启发式搜索相比完全搜索能有效降低搜索的时间复杂度，而且一般可以在发现最优子集后立即停止搜索，因而启发式搜索是现在最常用的特征选择方法。

2.2.1 完全搜索的特征选择

一般来说，对于一个特征个数为 n 的特征集，所有特征子集的个数为 2^n-1 个，即搜索空间的大小为 2^n-1。对此，Pudil 等[37]证明了这是一个 NP 问题。因此，使用完全搜索算法对所有特征子集进行搜索，是一个具有指数时间复杂度的算法，即此算法的时间复杂度为 $O(2^n)$。这是无法接受的，因为一般的计算机无法在有限的时间内完成对此种规模问题的求解。

下面介绍一个基于完全搜索策略的特征选择方法，即分支限界（Branch and Bound）法。该方法有一个前提假设：存在一个单调的评价函数 J，单调是指，对两个任意特征集 S_1 和 S_2，如果 S_1 是 S_2 的子集，那么 $J(S_1)$必小于（或大于）等于 $J(S_2)$。可以用一个树状结构来描述该算法的运行过程：首先，树的根节点为空集；随后，每一步都创建一个子结点，且树中每个子结点都比它的父节点多一个特征。由于评价函数 J 具备单调性，因此在此基础上搜索满足要求的特征子集，相比于单纯使用完全搜索会节约一定的计算时间，但是从时间复杂度的角度上看，它与不使用任何策略的完全搜索所具备的时间复杂度仍然处于同一数量级。此外，在大多数应用背景下，我们一般无法找到一个完全满足单调性要求的评价函数 J，而且在分类问题中，对评价函数单调性的验证过程一般就是分类过程。因此在选取

评价函数的环节上，分类问题一般不具备使用分支限界法的条件。

2.2.2 随机搜索的特征选择

随机搜索的特征选择算法一般应用在求解非全局最优特征子集的场景中，这是由于搜索方法具有一定的随机性，因此不能保证搜索结果一定是全局最优。举例来说，虽然随机搜索算法有时可以跳出一个局部最优解，但却有可能进入另一个局部最优。

随机搜索算法往往需要依靠带有一定智能的随机搜索策略来实现，这些策略一般指的是一些随机算法中使用的思想与策略，如随机序列算法、模拟退火算法和遗传算法等，这些算法均以一定的概率向前搜索，并结合随机采样过程，来实现算法的随机性。但是随机算法也普遍存在一些的缺点，如较为依赖随机因素，导致实验结果有时无法重现等。下面介绍上述几种常见的随机搜索特征选择算法以及它们的具体工作过程[38]。

(1) 随机序列选择算法

该算法首先产生一个随机的特征子集作为初始特征子集，之后在该子集上每轮添加一个未进入此特征子集中的特征来实现搜索过程，并重新使用评价函数来评价此特征子集，直至评价函数的取值不再继续变大；或者使用相反搜索的策略，每轮移除一个已被选入特征子集中的特征。之后可以再次使用多个随机产生的特征子集，并重复使用上述步骤来搜索基于随机产生的特征子集的最优子集。

可以看到，每次搜索过程的结束就意味着找到了一个局部最优解，但由于随机过程特点，因此我们不能保证该局部最优解一定是全局最优解；而且即使通过多次从不同的随机特征子集开始搜索，也无法保证一定可以得到全局最优解；此外，多个随机特征子集的搜索结果也有可能得到一个相同的最优解，但此时依然不能确定此最优解就是全局最优解。

(2) 模拟退火算法（SA，Simulated Annealing）

模拟退火与随机序列选择算法不同，它没有选取随机化的初始特征子集作为搜索起始点，但是它在搜索的过程中引入了一定的随机因素。模拟退火算法不会简单的排除较当前最优解差的解，而是会以概率化的方式接受一个较差的解。我们希望通过这种方式来跳出局部最优解，甚至到达全局最优解。下面形式化的描述算法。

定义一个评价函数 J，对于两个特征子集 F_1 和 F_2，令 F_1 为 F_2 在搜索过程中的前续特征子集（即通过一定的搜索规则，搜索至 F_1 之后，下一步就应当搜索 F_2），那么如果：

a) 若 $J(F_2)>J(F_1)$，即搜索一步后可以得到更好的结果，则总是接受 F_2；

b) 若 $J(F_2)<=J(F_1)$，即搜索一步后未能得到更好的结果，则以一定的退火概率接受本次的搜索结果，而且这个概率随着搜索的进行逐渐降低，使搜索过程逐步趋向稳定，最终收敛于一个稳定的解。

名词“退火”指的是一种金属的热处理工艺，具体来说，是指是将金属缓慢加热到一定温度并保持足够时间，之后再以适宜的速度冷却。上述第二步中“一定的退火概率”就是参考了金属热处理中的退火过程，这也是模拟退火算法名称的由来。模拟退火在一定程度上避免了随机序列搜索算法容易陷入局部最优的缺点，但是当局部最优解附近的梯度较大，或“退火概率”取值较小，则模拟退火算法也无法求得全局最优解。

(3) 遗传算法（GA, Genetic Algorithms）

遗传算法借鉴了从生物学中遗传的概念。简单的说，遗传算法将要解决的问题模拟为生物进化的过程，通过多代的繁殖过程，并在繁殖中引入基因复制、基因交叉和基因突变等可能改变子代个体属性的行为，逐步选择评价函数取值较优的个体予以保留，同时将评价函数取值较差的个体淘汰。在经过一定代次的遗传后，我们就可以认为经过评价函数选择而得以保存下来的个体都是取值较优的个体，且其中也很可能包含全局最优。

2.2.3 启发式搜索的特征选择

在三类特征选择算法中，完全搜索算法与随机搜索算法虽然都有一定的优点，但是却也都存在相应的不足之处，因此可以考虑将二者的长处进行一定程度的结合，这就是最常用启发式搜索的特征选择算法。启发式搜索策略的特征选择算法主要有 4 种具体的搜索策略，下面予以简单的介绍。

(1) 单特征组合策略

该策略首先求解各个特征独立使用时的评价函数取值，并依据取值排序，然后直接取出排名位于前 n 的特征作为最优的特征子集。但是，这种策略仅当各单个特征满足加性条件或乘性条件的时候才能有效施行。

(2) 序列前向选择策略（Sequential Forward Selection，SFS）

该策略首先初始化最优特征集为空集，之后每次向特征集合中添加一个特征，并保证每次新添加的特征相对其他同代特征最优，最终当最优特征集达到预设的特征数量要求或满足预设的评价函数要求时，就将此时所得到的最优特征集作为最终特征选择算法运行的结果。此外还可以在每次向特征集合中添加特征时加入 f 个特征，其中 f >1，这就是广义序列前向选择策略，该策略可以被视为 SFS 策略的加速版本[39]。

(3) 序列后向选择策略（Sequential Backward Selection，SBS）

该策略与SFS策略恰好相反，在运行开始时从包括所有特征的特征集合出发，之后每次从特征集合中移除一个特征，并保证移除某个特征后的新子集相对其他同代特征集最优，最终当达到预设的停止条件时，就可将此时所得到的特征子集作为最终的最优特征集合。此外也可以每次从特征集中移除 b 个特征，其中 $b>1$，则就是广义序列后向选择策略，该策略可以被视为 SBS 策略的加速版本。

(4) 增 f 去 b 选择策略

这种策略在搜索过程加入回溯过程。举例来说，如果 $f > b$，则该算法首先使用 SFS 策略将 f 个特征加入到特征集中，之后再使用 SBS 策略移除 b 个特征；如果 $f < b$，则算法首先使用 SBS 算法从一个包含所有特征的集合中依次移除 b 个特征，之后再使用 SFS 策略向特征集中添加 f 个特征。该策略实际上是 SBS 策略和 SFS 策略的一种组合与折衷。此外还有广义增 f 去 b 选择策略，即使用广义 SFS 和 SBS 策略替代前述 SFS 和 SBS 策略。此外，对增 f 去 b 选择策略还有一种改进方式，就是在选择算法的不同步骤中使用不同的 f、b 值，实际中的 f、b 值也可以依据具体的特征来选取，这种策略叫做浮动搜索策略，在特定场合中可以获得良好的效果。

2.3 基于迭代重分配的聚类算法

2.3.1 聚类算法概述

聚类算法即无监督的机器学习算法，聚类算法只能做到将数据点分入若干个不同的簇，并且使得每个簇内数据点之间的相似度尽量大，不同簇内数据点之间的相似度尽量小[40]。无监督的分类算法在分类之前不需要专用的样本集来指定每一个类所具备的具体特征，分类过程只需要考虑数据点本身的属性、坐标值和向量值即可。这种分类方法适用于无法预先确定分类结果中的类型个数的情形，也适用同一种类型具备多种特征的情形。

一般来说，聚类算法可以分为四类，即划分聚类方法、层次聚类方法、基于约束的聚类算法和高维数据的聚类算法[41]。在机器学习领域比较常见的聚类算法，都属于划分聚类算法。在划分聚类算法内部，又可以细分为三类，即基于密度的聚类、基于网络的聚类和基于迭代重分配的聚类。

由于在网络流量分类中，EM 算法、最近邻算法、K-中心点（K-Medoids）算法和 K-均值（K-Means）算法被广泛采用，而这几种算法均属于基于迭代重分配的聚类算法，故本节将对此类算法进行详细介绍。

2.3.2 EM 算法与最近邻算法

基于迭代重分配的聚类方法主要通过多次迭代的方式，逐步求解最优化的聚类结果，反复的将数据集中各数据点重新分入各聚类中心点所属簇，并通过计算平方差距离等方式判定每次分配后的结果是否为最优，或比较多次分配的结果以判断是否达到稳定。此类算法中，最基础的，也是最早被提出的两种算法是 EM 算法与最近邻算法。

EM（Expectation Maximization）算法由 Arthur 等提出[42]，本方法在每一轮中首先使用隐含变量的现有估计值，计算模型的最大似然估计值，随后再使用上一步得到的最大似然估计值调整模型，重新估计原先的隐含变量。这两个步骤交替进行，经过多轮迭代，直到对隐含变量的估计值收敛为止。本方法能处理大量的、具有复杂结构的记录，并且可以产生较为合理聚类结果。

最近邻（NN, Nearest Neighbor）算法的思想较 EM 算法更直接，本方法首先计算目标数据点与其周围已被分入聚类簇的数据点之间的距离，并对此距离进行排序，随后通过比较目标数据点在一定范围内的各类型数据点的个数，来决定的目标点的应该分入的簇。但是这种聚类方式的时间复杂度较高，一般来说可以达到 $O(N^2)$量级，其中 N 为数据集中数据点的总个数。

2.3.3 K-均值算法

K-均值算法即 K-means 算法，这类算法是目前为止实现版本最多、工作最稳定的一种聚类算法。在 K-均值算法中，每个聚类簇均由该类中所有数据点的几何平均或算术平均来代表，并且将其为聚类中心。该方法虽然不直接给出每个数据点的具体的类别属性，但对于具有相似属性的数据点，该方法可以很好地将它们聚集在一起，形成一个紧密的簇结构。

但是 K-均值算法也存在一些缺陷。首先，在 K-均值算法中的聚类簇的数量 k 需要在分类之前指定。对此，Zhang 等[26]通过搜索，验证了在使用 10 倍于、甚至 20 倍于网络流量类型个数的簇个数 k 时，才能获得较好的分类效果。但对于一般性的问题，k 值的选取则是一个困难的问题，需要对每个问题进行单独的讨论与实验，才能得出与目标问题相适应的 k 值。

第二，在 K-均值算法中，聚类结果对初始种子（Seed）存在一定程度的依赖，能否选取合适的种子也会直接影响最终的聚类结果。一般来说，我们需要使用种子值来对聚类模型进行初始化，随后才能开始迭代的过程。如果不能选择适当的初始值，那么聚类结果可能会比较不理想，甚至有可能使算法陷入局部极小值等，导致不平衡情况的出现。

2.3.4 K-中心点算法

K 中心点算法即 K-medoids 算法，该方法主要使用簇中的某个特殊点来代表该聚类簇。一般而言，而 K-中心点算法不是以簇内的点的平均值计算的中心点的，而是选择距离平均值最近的数据点作为该簇的中心点，这与 K-均值算法是不同的。由于 K-均值算法需要单独给出簇中心点的定义，而且 K-均值算法一般不适用于发现高维非凸面体形态的簇，或者簇间规模差别很大的簇，但是 K-中心点算法能较为有效的处理以上这些异常类型的数据，得到较好的效果。K-中心点算法在早期有两个版本，分别是 PAM 算法和 CLARA 算法[43]，经过发展，最终演化成了现在的 K-中心点算法。

2.4 本章小结

本章主要介绍了与网络流量分类紧密相关的几种技术，具体包括以下几点。

第一，网络流量。首先简要介绍了网络流量及网络流量的特点，之后详细说明了网络流量的捕获、存储、处理与分析的过程，说明了 libpcap/winpcap 在网络流量捕获与处理工作中的重要作用。

第二，特征选择方法。首先介绍了特征选择的目的；随后分别展示了三类常用的特征选择方法：完全搜索的特征选择、随机搜索的特征选择和启发式搜索的特征选择方法，并对启发式搜索的特征选择使用的四种策略进行了深入的介绍。

第三，基于迭代重分配的聚类算法。首先从介绍一般性的聚类算法入手，并对各种聚类算法从聚类原理上分类，之后由浅至深的介绍了几种基于迭代重分配的聚类算法，并仔细讨论了这些算法长处、不足以及适用范围。

第3章　基于流统计特征的网络流量分类技术

3.1 引言

随着在网络上传输的流量越来越多，流量的类型越来越复杂，基于端口的分类技术与基于负载信息签名的分类技术越来越无法满足网络分类的要求与发展，因此，基于流统计特征的分类技术逐渐受到了更多的重视。本章首先介绍了使用机器学习方法进行网络流量分类的系统，同时给出了一组常用的网络流量统计特征。随后重点描述了对使用相关流法与多数投票法对聚类算法产生的流量簇进行标记的方式。最后我们基于上述工作框架完成了网络流量的分类实验，并对实验结果进行了详细的讨论。

3.2 网络流量分类系统

3.2.1 基于机器学习的分类系统简介

在一个典型的机器学习的分类过程中，输入是由一组实例构成的，其中的每一个实例都由一个向量来描述，而且这个向量定义在一个固定的特征空间内；输出是由类型标签或者簇构成，其中类型标签定义在类型空间内，而由于簇是无类型的，所以一般不受定义空间的限制。

在网络流量分类中，作为输入的实例是一个网络流。一个网络流可以由一个五元组唯一描述，即网络流的键值为五元组<源 IP，源端口，目的 IP，目的端口，运输层协议>。此外，在某些研究工作中[44-45]，研究者们也会使用双向网络流作为输入实例的类型，而双向流依然可以使用上述的五元组进行描述。

上述五元组只能作为唯一标识网络流的依据，而不是作为输入实例的特征空间。特征空间一般是指网络流的统计特征，一般来说，可以包括流中的包个数，字节数，流存续时间，包大小的均值、方差，包间隔时间的均值、方差等。对于 TCP 类型的流来说，还可以对带有控制标记的包进行统计，如 ACK 包的总数，ACK 包之间的时间间隔，PSH 包的总数等。这里需要注意的是，流统计特征一般不包括端口号信息，也不包括数据包负载部分的内容信息。

对于分类工作来说，其具体输出结果主要与分类过程中使用的的机器学习算法有关。如果我们使用有监督的机器学习算法，则首先应当对分类模型进行训练。随后在分类阶段，分类算法可以依据对样本集的学习结果，直接判定某个流量的

类型，即直接将此流量直接分入某个具体类型中。所以在使用有监督的机器学习算法时，我们可以直接得到某个流量的类型。

但是样本集的构造具有一定的困难性，这主要是由于基准类型的获取较为困难；其次，因为同一类型的网络流量可以在属性空间内具有多个相互分离的样本群，即两个差异化程度很大的样本，也有可能属于同一种网络流量的类型，所以在选取样本时，我们很难保证将同一流量类型的所有相互分离的样本群全部选中；此外，样本集所选取的样本很难做到普适性与一般性，不一定能有效的代表特定的类型。基于以上的原因，在使用机器学习算法对网络流量分类时，我们更倾向于使用无监督的机器学习算法，即聚类算法。

在使用聚类算法对网络流量进行分类时，我们得到的输出一般是聚类簇。这些聚类簇本身不包含任何网络流量类型信息，我们仅认为所有处于同一个聚类簇内的流，必定属于同一种网络流量类型。因此，如何合理、准确的标记聚类簇整体的流量类型，也成为了一个难点。

最后简要的讨论分类器算法。基于以上的讨论，一般来说，我们使用的有监督的机器学习算法主要包括朴素贝叶斯法（Naïve Bayes）[46-47]、支持向量机[48-50]等，无监督的机器学习算法主要包括最近邻算法、K-均值算法和 K-中心点算法。在网络流量分类分类问题中，假设候选的网络流量类型的种类个数为 N，一般我们在使用有监督的机器学习算法时，往往把 N-分类问题转换成为 N 个二分类问题，即为每一种流量类型单独训练一个分类模型；而在使用无监督的机器学习算法时[51-52]，我们则不需要考虑单独处理每个流量类型。

3.2.2 常用的统计特征

基于流统计特征的网络流量分类中，存在一个至关重要的环节，就是统计特征的选取。可以说，统计特征的选取将会直接影响分类的效果与性能。[53]

上文提到，统计特征主要可以包括流中的包个数，字节数，流持续时间，包大小的均值、方差，包间隔时间的均值、方差等与流行为有关的特征，对于 TCP 类型的流来说，还可以包括流中 ACK 包的总数，ACK 包之间的时间间隔，PSH 包的总数等与 TCP 状态相关的特征。本文在此将常用的统计特征列出，如表 3.1 所示，表中列出的特征都可以在分类工作中作为候选特征使用。

3.3 对聚类簇的标记方法

我们在前一节中提到，如何对聚类产生的流量簇进行标记，以得到簇对应的网络流量类型，是一个难点问题。一般来说，我们可以使用相关流法与多数投票

法相结合的方式，完成对流量簇的标记。本节就将针对此问题进行详细的讨论，下面逐一介绍这两种方法。

表 3.1 常用的统计特征

特征类型	特征描述
基于流行为的	流的持续时间（Duration of flow）
	流中的包个数（Number of packets）
	流中的总字节数（Volume of bytes）
	流中各包大小（Packet size）的最大值、最小值、均值和方差等
	流中各包负载大小（Payload size）的最大值、最小值、均值和方差等
	流中相邻包时间差（Time interval）的最大值、最小值、均值和方差等
基于 TCP 状态的	流中 ACK 位有效的包个数
	流中 PSH 位有效的包个数
	收到第一个返回的 ACK 包之前发送的字节数
	流中 ACK 位有效且携带的 SACK 信息的包个数
其他	RTT 样本（RTT samples）的个数
	流中第一个与最后一个携带负载信息的包的时间差

3.3.1 相关流法

相关流（Correlated flows）的概念最早提出由 Wang 等人[44]于 2011 年提出，并由 Zhang 等人[54]详细说明。相关流的概念是基于一个假设提出的，这个假设认为：在一段时间之内，存在一个固定的端点，对外仅提供同一类型的服务，那么使用<端点 IP，端点端口号，运输层协议>这个三元组即可唯一的标识这个固定的端点。如果多个流的五元组中包含共同的三元组，那么就可以认为这些流必定属于同一流量类型，并且我们称这些流之间互为相关流。一个相关流的例子如图 3.1 所示，图中的四个流互为相关流，我们也称这四个流为一组相关流。

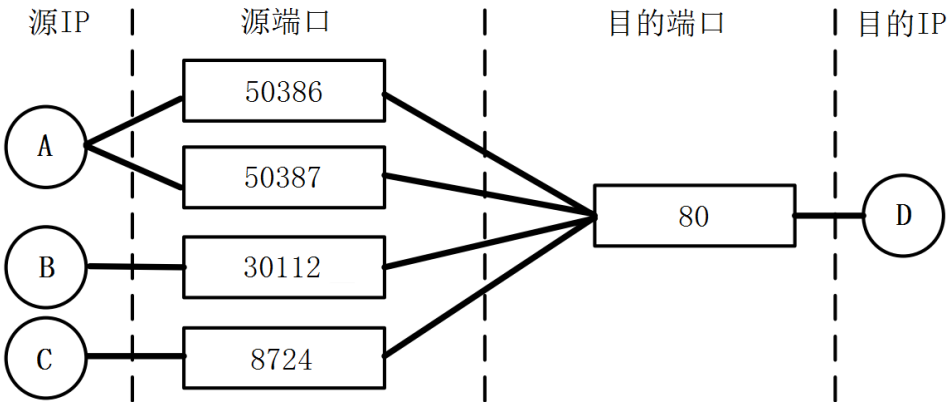

图 3.1 相关流（Correlated flows）

一般来说，上述假设是普遍成立的，因为在网络上，大多数提供特定服务的服务器都是如此工作的。这些服务器往往只提供固定的服务，也只使用固定的 IP、端口组合，即只使用固定的套接字。所以具有相同三元组的流可以被认为是同一类型的。

为了利用这种相关关系，我们可以利用流标签扩散的方式来将相关性传递到多个未标记的数据包。假设有一个流集合 $A=\{a_1, a_2, ..., a_n\}$，且集合 A 中的所有流都已经被标记，另有一个流集合 $B=\{b_1, b_2, ..., b_m\}$，且集合 B 中的所有流都未被标记。通过使用上述的三元组，将集合 B 中所有与集合 A 中元素相关的元素进行标记，具体的标记过程如算法 3.1 所示。此算法即基于相关流的流标签扩散算法。

算法 3.1 流标签扩散

Algorithm 3.1 Flow label propagation

Input: small flow set ***A*** with all flows labeled, large flow set ***B*** unlabelled
Output: extended set ***E*** with labeled flows.

```
1  E ← A;      // Create the output flow set
2  for i←1 to n do
3      for j←1 to m do
4          check and compare 3-tuple of a_i and b_j; //a_i ∈ A, b_j ∈ B
5          if a_i and b_j share the same 3-tuple then
6              b_j.label←a_i.label;
7              put b_j into E;
8          end if
9      end for
10 end for
11 return E;
```

在标记结束之后，可以得到集合 E，即上述算法的输出，其中 E 定义如下：

$$E = A \cup \{b_j \in B \mid \exists a_i \in A \text{ is related to } b_j\} \tag{3.1}$$

此时，对于集合 E 中的所有流量，我们都可以视为获得了类型标签，所以可以将集合 E 称为已标记集。此外，可以看出：

$$E \subseteq (A \cup B) \tag{3.2}$$

即 E 是集合 A、B 交集的子集。由于集合 A 是全部带有标记的，这就说明集合 B 中可能有经过流标签扩散后仍然未获得类型标签的流。

2013 年，Park 等[25]通过研究表明，在一个由实验者自行采集的非公开数据集上，约有 38%的流量可以通过相关流法进行标记。此外，如果以一小时为单位，可以发现每小时内的相关流的组数远远小于流的总个数，其中流的总个数最低为相关流组数的 2 倍，最高可以达到将近 6 倍，如图 3.2 所示。由于我们总是使用标识相关流的三元组或标识连接的五元组作为存储流数据的键值，如果键值的个

数可以被显著地减少，那么就可以有效降低存储流数据所使用的空间。

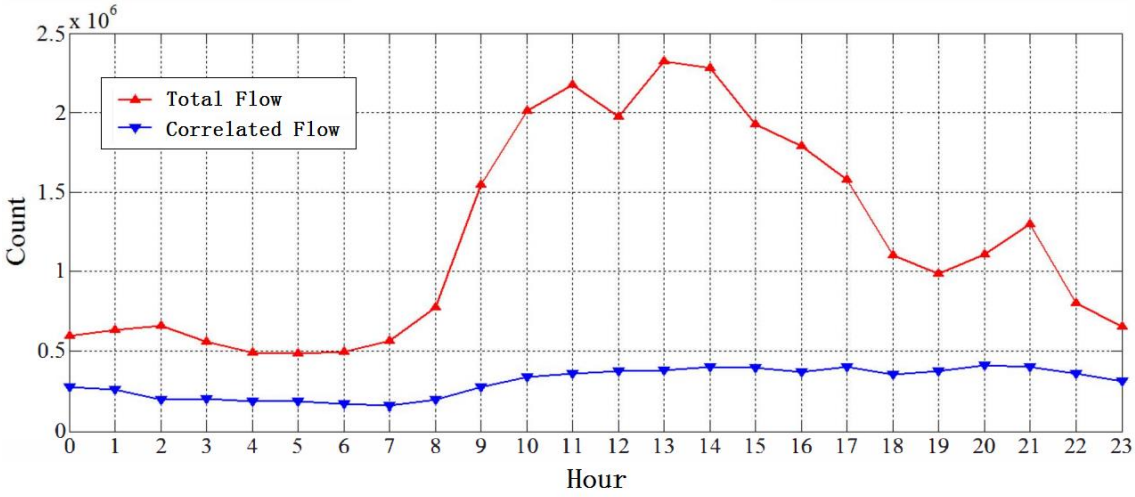

图 3.2 总体流数目与相关流组数的对比

3.3.2 多数投票法

多数投票（Majority Voting）法是一种使用簇内流量的类型标签，对整个簇的类型进行标记的方法。简单的说，一个流量簇中，哪个类型的流量最多，我们就可以认为这个簇属于哪个类型。

令 $C=\{W_1, W_2, ..., W_n\}$为一个未被标记的流量簇，$T=\{T_1, T_2, ..., T_m\}$为簇中所有流的标记类型集，即簇中所有流的标记类型为 $T_j \in T$，其中 j=1, 2, ..., m。此时，使用多数投票法可以将流量簇 C 标记为 T_{most}，其中 T_{most} 的定义为：

$$\sum_{i=1}^{n} T_{most} = \max_{j=1,2,...,m} \sum_{i=1}^{n} T_{ij} \tag{3.3}$$

其中，T_{ij} 表示流 W_i 与标记类型为 T_j 的关系，这个关系的定义如下：

$$T_{ij} = \begin{cases} 1 & \text{if } W_i \rightarrow T_j \\ 0 & \text{if } W_i \not\rightarrow T_j \end{cases} \tag{3.4}$$

其中$\rightarrow$标识符代表流 W_i 的类型为 T_j，$\not\rightarrow$ 标识符代表流 W_i 的类型不为 T_j。在此定义中，如果我们认为流 W_i 的类型标记 T_j 一定是可靠的，我们就可以直接使用如式 3.4 所示的 0-1 二值化投票；如果我们对流 W_i 的类型标记 T_j 不做完全的接受，则可以使用[0, 1]这个闭区间内的数值分别作为流 W_i 的类型为 T_j 与类型不为 T_j 时 T_{ij} 的取值，且两种情况下的概率取值之和为 1，即可实现概率化投票。

多数投票法的示意图如图 3.3 所示[55]。通过配合使用多数投票法与相关流法，就可以完成对流量簇类型的标记工作：首先通过扩散算法标记大多数流，再使用多数投票法完成对簇类型的标记。另外对于未知类型，我们只需要将其单独作为一种类型即可，如果簇中的未知类型的流量占到大多数，我们就可以将相应簇的

流量类型标记为未知类型。

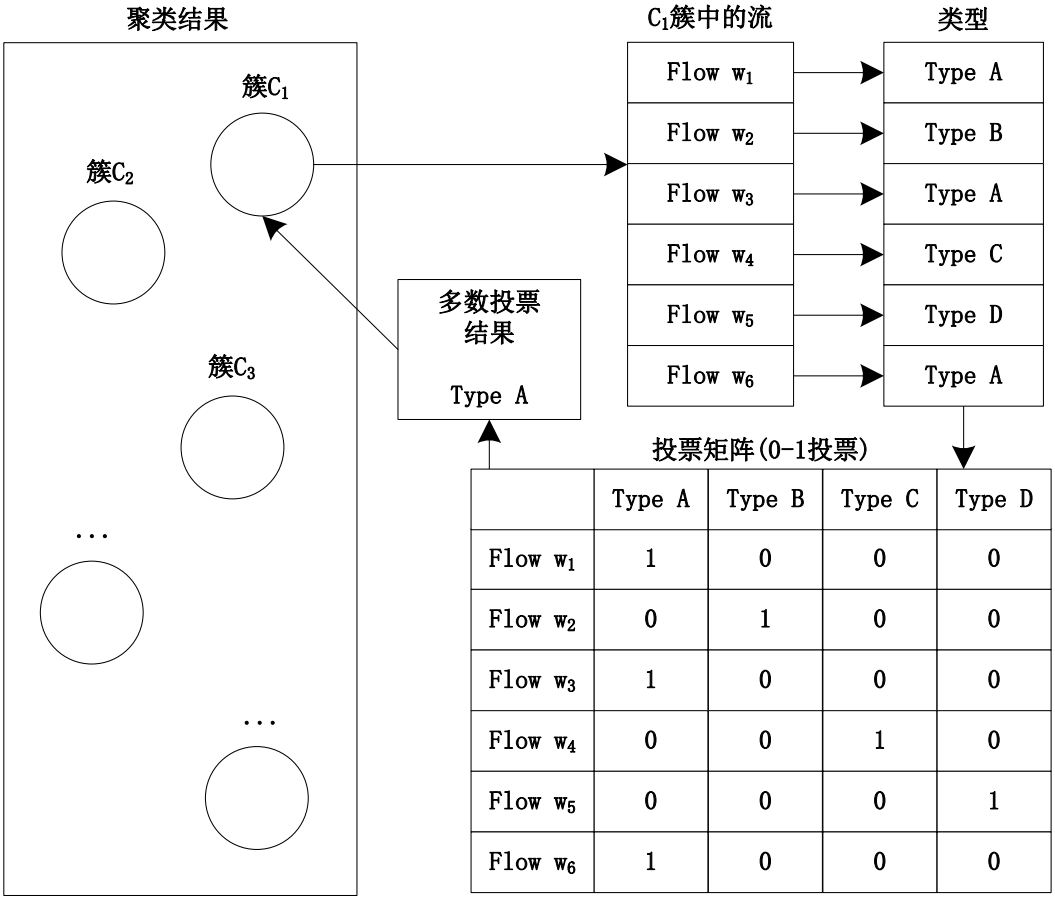

图 3.3 多数投票法示意图

3.4 实验

3.4.1 实验环境与数据集

本实验的环境主要为基于 Windows 的流量分析与分类系统。在本系统中，我们首先使用基于 winpcap 开发的网络流量分析工具，该工具可以从 dump 文件中依次读出原始数据包，并对数据包整理、汇聚进而得到网络流；其次，我们使用 WEKA 中的分类工具[56]，来完成对网络流量的聚类工作；最终，我们使用前文提到的流量簇标记方法对流量簇进行标记，得到各流量簇的网络流量类型。在此我们对 WEKA 智能分类工具进行简单的介绍。WEKA（Waikato Environment for Knowledge Analysis，怀卡托智能分析环境）是由怀卡托大学开发的常用的机器学习与数据挖掘软件，它基于 JAVA 开发，并且免费、开源。WEKA 中包含了数据预处理、回归算法、有监督的分类算法、无监督的聚类算法以及可视化等部分，此外，基于

WEKA 的源码来开发实现新的机器学习算法也非常方便。

数据集方面，我们选取了 WIDE[57]数据集。WIDE 数据集由日本 MAWI 工作组维护的，WIDE 数据集是在一条连接日本与美国的 150Mbps 链路上直接采集的，MAWI 不仅会给出每天一定时段的数据包，还会在每年选取 1-3 天的时间，进行全天候的采集，而且在全天候采集的情况下，也会给出应用层部分的前 40 字节信息，便于研究者对流量进行更加深入、全面的研究。表 3.2 给出了上述数据集的大小、负载字节数和采集时间等信息。

表 3.2 WIDE 数据集基本信息

数据集	大小	包个数	开始时间	持续时长	负载信息
WIDE-08	36.1GB	60.6M	2008-03-18 13:00	1 小时	前 40 字节
WIDE-09	44.3GB	66.2M	2009-03-31 13:00	1 小时	前 40 字节

3.4.2 实验评价标准

为了评价网络流量分类算法的性能，我们选取了分类问题中最常用的几个性能评估指标来对分类性能进行衡量。首先我们引入几个变量：

(1) B_p。B_p 描述在真实的基准情况中，属于类型 p 的流的数量，这个变量一般无法被分类器感知，只能依赖于内建在数据集中的信息进行判断。

(2) C_p。C_p 描述在分类过程结束之后，被分入类型 p 的流的数量，这个变量直接依赖于分类器，隐含的体现了分类器的性能。

(3) R_p。R_p 描述被分类过程正确的分为基准类型 p 的流的数量，显然的，$R_p \leqslant B_p$ 且 $R_p \leqslant C_p$。并且我们还可以观察到，有

$$\sum_p C_p = \sum_p B_p \tag{3.5}$$

之后，我们可以使用上述三个变量，导出如表 3.3 中的几个常用的分类度量指标[58]。

表 3.3 分类问题中的几个常用度量指标

	本身属于类型 p 的数量	本身不属于类型 p 的数量
被分入类型 p 的数量	*TP*	*FP*
未被分入类型 p 的数量	*FN*	*TN*

(4) *TP*（True Positive），即真肯定率。由上，*TP* 可以被定义为 R_p / C_p.

(5) *FP*（False Positive），即假肯定率。由上，*FP* 可以被定义为 $(C_p - R_p) / C_p$.

(6) *FN*（False Negative），即假否定率。由上，*FN* 可以被定义为 $(B_p - R_p) / B_p$.

(7) *TN*（True Negative），即真否定率。这个指标一般不在多分类问题中引入。

最后，我们可以依据上述分类度量指标，导出分类器的效果评估指标。

(8) $Precision_p$，即某特定类型 p 的精确率。一般的，$Precision_p$ 被定义为：

$$Precision_p = \frac{TP}{TP+FP} = \frac{R_p}{C_p} \Big/ \left(\frac{R_p}{C_p} + \frac{C_p - R_p}{C_p} \right) = \frac{R_p}{C_p} \tag{3.6}$$

对于整个分类算法，它的精确率一般可以使用加权平均的方式求得：

$$Precision = \frac{\sum_p Precision_p \cdot C_p}{\sum_p C_p} = \frac{\sum_p R_p}{\sum_p C_p} \tag{3.7}$$

(9) $Recall_p$，即某特定类型 p 的召回率。一般的，$Recall_p$ 被定义为：

$$Recall_p = \frac{TP}{TP+FN} = \frac{R_p}{C_p} \Big/ \left(\frac{R_p}{C_p} + \frac{B_p - R_p}{B_p} \right) \tag{3.8}$$

对于整个分类算法，它的召回率一般可以使用加权平均的方式求得：

$$Recall = \frac{\sum_p Recall_p \cdot B_p}{\sum_p B_p} \tag{3.9}$$

(10) *F-measure*，即 F-度量。F-measure 是一种综合考虑精确率 *precision* 和召回率 *recall* 的性能评估指标，它被广泛的应用于机器学习算法评价领域。一般的，*F-measure* 被定义为：

$$F-measure = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3.10}$$

3.4.3 实验设计与结果分析

实验使用的特征集仅选取基于流行为的特征，这里给出一组其他研究者常用的特征集，如表 3.4，这个特征集包含了最基本的统计量与流行为组合产生的流统计特征。特征集内的特征数目为 10。

表 3.4 基于流行为的统计特征集

流行为的种类	特征描述	特征个数
流中的包	包的总数	1
流传输的字节	字节总数	1
流中各包的大小	最大值、最小值、平均值与标准差	4
相邻包的时间差	最大值、最小值、平均值与标准差	4

实验使用的聚类方法为 K-means 方法，K-means 方法内嵌在 WEKA 中，调用较为方便。对使聚类得到的流量簇，我们结合使用相关流法与多数投票法对其进行标记。对数据集内部的相关流特性进行考察，可以得到如表 3.5 所示的结果，

可以看到相关流在所有流中的比例非常高，这表明使用相关流法进行流标签扩散，可以实现对绝大多数的流进行类型标记。

表 3.5 数据集中的流与相关流

数据集	流的数目	相关流占比
WIDE-08	532K	98.6%
WIDE-09	231K	98.3%

接下来，我们可以得到对上述两个数据集的分类结果，WIDE-08 数据集的分类结果如图 3.4 所示，WIDE-09 数据集的分类结果如图 3.5 所示。我们使用 6 个网络流量类型对数据集进行划分，这六个类型分别是 http（包括 http 和 https）、mail（包括 smtp、imap 和 pop3）、ssh、ssl、dns 和 other（包括其他所有类型）。

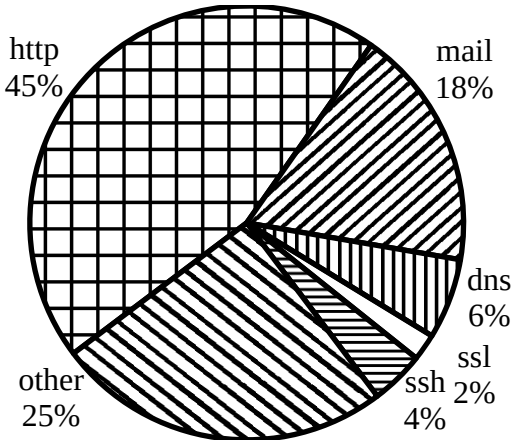

图 3.4 WIDE-08 的分类结果，即类型组成（Breakdown）

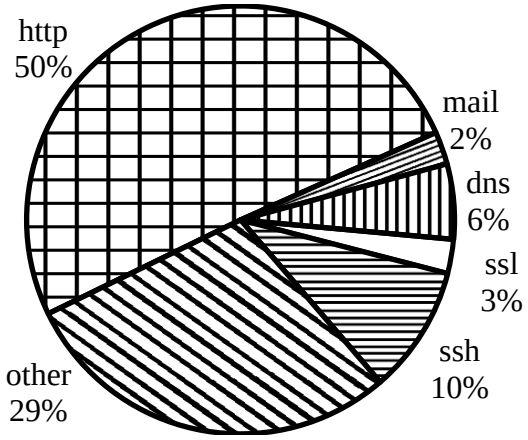

图 3.5 WIDE-09 的分类结果，即类型组成（Breakdown）

我们在上述使用 K-means 算法进行聚类的过程中，选取的类簇个数值 K 为 100。接下来我们可以考虑使用不同的 K 的取值，观察 K 值的变化对分类结果有何影响。

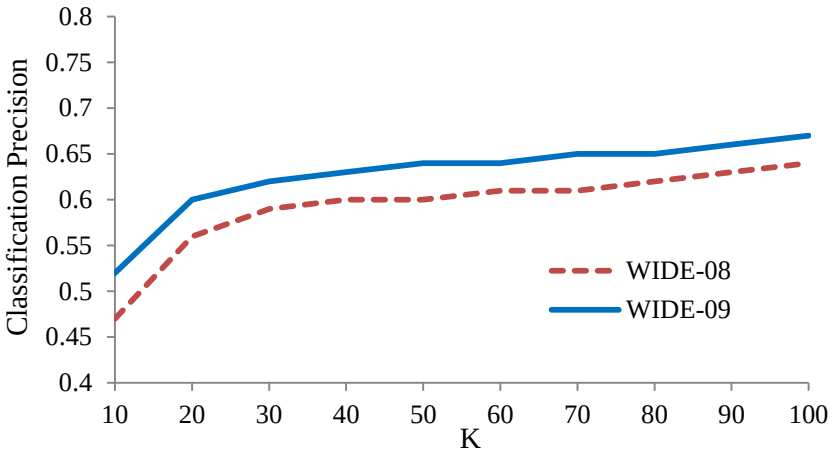

图 3.6 使用 K-means 算法时 K 的取值对分类准确率的影响

图 3.6 展示了两个特征集的分类结果准确率随 K 取不同值而变动的情况。可以看到分类准确率在 K=10 到 100 的范围内（每次 K 取值增加 10），随着 K 的增

长而增长。由于图 3.4 和图 3.5 中的分类结果是在 K=100 时得到的，所以其准确率相对较高。这个结果与 Zhang 等[26]的研究结果相一致。我们对此结果的解释是，当选择较大的 K 值时，聚类簇的簇内部错误率会较低，这表示同一簇内的流量属于同一网络流量类型的概率更高，进而可以使得最终的分类准确率更高。

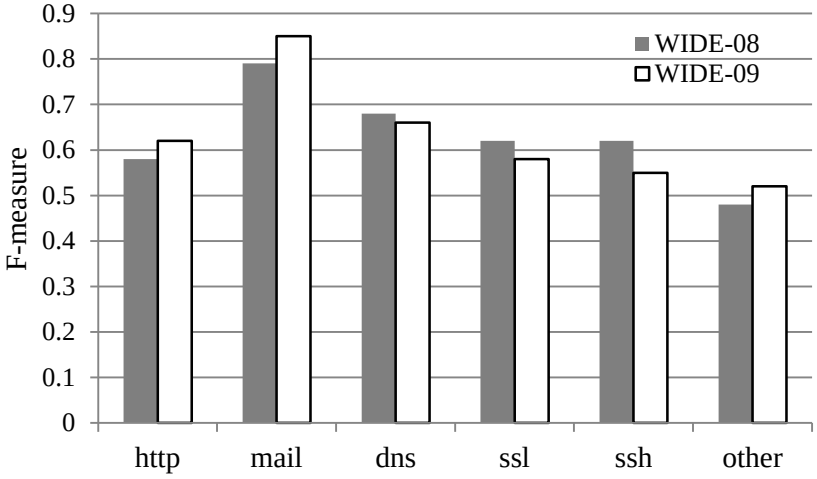

图 3.7 每种流量类型的 F-measure 指标

分类问题中，我们一般会结合分类结果的准确率、召回率等多个指标，使用 F-measure 来衡量分类的效果。图 3.7 分别展示了两个数据集中每种流量类型的 F-measure。可以看到，分类器对 mail 类型的分类效果相对较好，对 other 类型的分类效果相对一般。对此，我们认为在类型组成中比重较小的类型，一般可以获得相对其他类型更高的 F-measure 值，比如 mail 类型；而在类型组成中比重较大的类型，往往由于精确率与召回率都会受到相对较大的影响，所以相对其他类型来说，可以得到的 F-measure 值会更小，比如 http 类型。

3.5 本章小结

本章主要实现并展示了基于流统计特征的网络流量分类工作，具体内容主要包括以下几点。

第一，简要介绍了基于机器学习算法的网络流量分类方法，给出了一组常用的网络流量统计特征。

第二，详细讲解了对聚类簇的两种标记方式，即相关流法与多数投票法，并简要的讨论了两种方法的效果。

第三，完成了基本的网络流量分类实验。首先描述本实验中使用的实验环境与平台，介绍了几个本领域内的公开数据集，随后给出了常用的对分类算法效果进行度量的指标，最后使用特定的聚类方法实现了对网络流量的分类，展示了各数据集的类型成分，得出了在一定范围内 K 取值对聚类算法的影响，并讨论了各流量类型的 F-measure 取值情况。

第4章　基于特征选择的网络流量分类技术优化

4.1 引言

在上一章中介绍的网络流量分类技术中，我们完成了一整套的网络流量分类工作。其中，在分类过程中较为关键的一步，就是要选取一定的特征作为特征集。如何选取特征，什么样的特征更适合网络流量的分类工作，是一个值得讨论与研究的问题。本章首先提出了使用流统计特征实现网络流量分类的一般性框架，即分类工作的技术实施路线。随后介绍了特征选择算法的评价函数，并描述了启发式特征搜索算法的工作原理与序列前向搜索策略的搜索过程。接着本章提出了对网络流量分类中所用到统计特征的改进，即引入具有高阶矩的统计特征。接下来我们使用候选特征集进行特征选择，得到优化后的特征集，并对优化后的分类效果进行了讨论，验证了优化工作的有效性。

4.2 基于流统计特征的网络流量分类一般性工作框架

基于前一章的介绍，我们可以对基于流统计特征的网络流量分类有一个大致的了解，下面我们给出一个用于分类的一般性工作框架，简要描述各个模块的数据处理功能，展示各个部分如何相互关联。一般性工作框架如图 4.1 所示，从上到下依次分为数据采集部分、数据处理部分和流量分类部分，每部分内又包含若干操作步骤，其中画虚线的操作属于可选操作，使用特征选择算法对特征集进行处理、使用原始 dump 文件生成流标签这两项操作都属于可选操作，通过使用这些可选操作，我们可以对整个分类框架进行一定程度的优化。最终得到的分类结果为 Classification Results，在图中使用加粗的斜体字标出。

4.3 启发式搜索特征选择算法

在第二章中，我们已经对特征选择算法做过了简要的介绍。本节将首先讨论特征选择算法的评价函数，随后将详细说明如何在优化工作中使用启发式特征搜索选择算法，并结合序列前向搜索策略与相关的评价函数给出一个简单的特征选择实例。

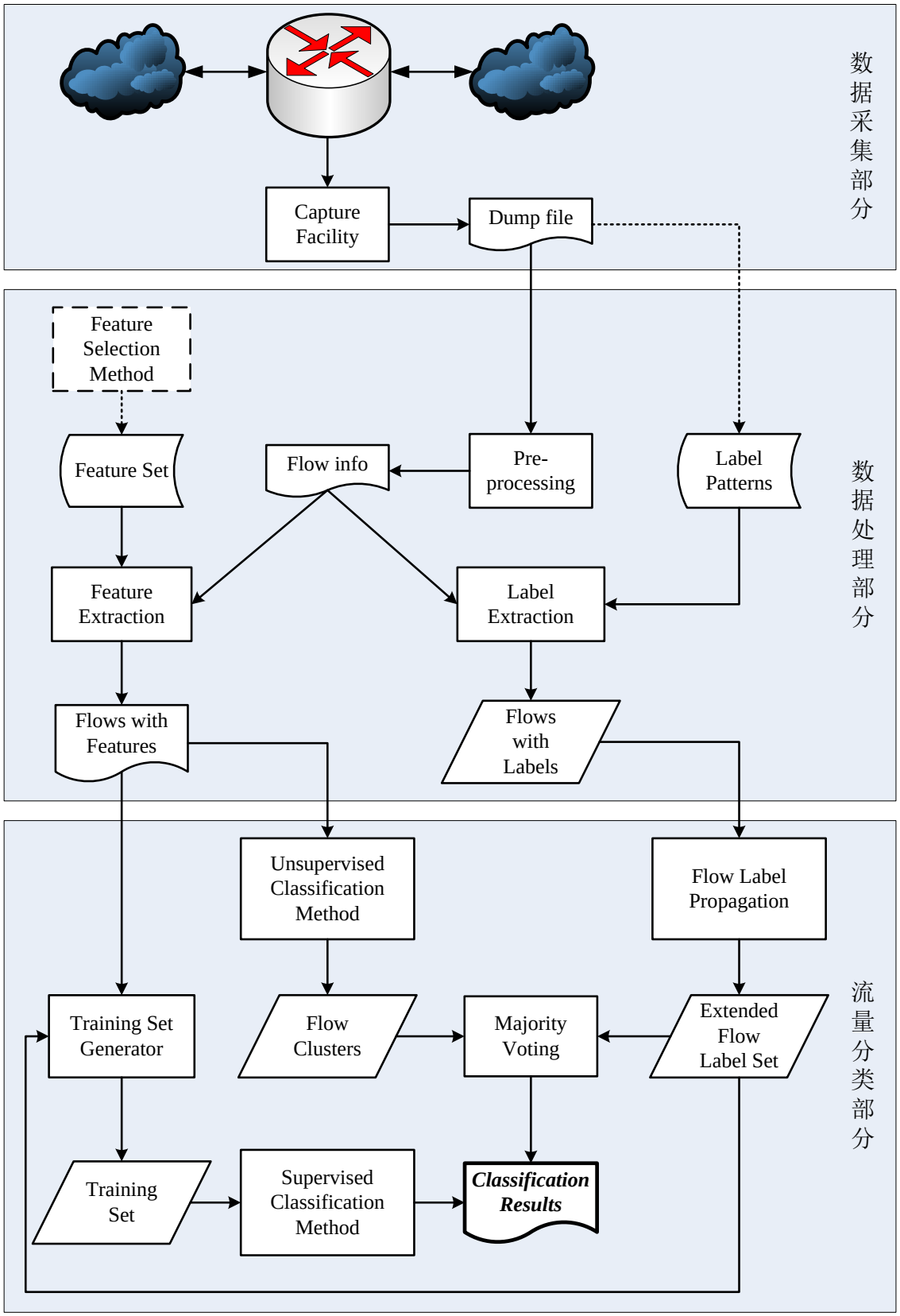

图 4.1 基于流统计特征的网络流量分类一般性工作框架

4.3.1 评价函数

由于使用特征选择对特征集进行筛选的目的是挑选一组最适合分类工作的特征集，因此如何评价某个特征或特征集的优劣，将会直接影响分类算法的工作过程与最终结果。对特征或特征集的评价标准，我们一般称为评价函数。评价函数在特征选择算法中负责评判是否选择或排除某个特征，决定搜索过程是否继续，这些判断结果在特征选择的过程中都有决定性的作用。

评价函数从计算策略上来分，可以将它们分成两类：过滤器（Filter）策略及封装器（Wrapper）策略[59]。过滤器策略主要侧重于通过各个特征之间的相关关系来评价特征集，而封装器策略一般需要与分类器算法结合使用，通过评价从一组特定特征集出发而得到的分类结果来评价这组特征集。下面对这两类策略进行简要的介绍。

(1) 过滤器（Filter）策略

过滤器策略主要使用一些特定的过滤规则，对特征集中的特征进行分析，从特征之间的结构与内部属性入手，对特征集进行调优。常见的过滤器规则可以将特征集中彼此之间相关性较高的特征筛选出去，即削弱特征之间的互信息；也可以通过判断特征集的信息熵，保留信息增益较高的特征；还可以通过类内部的点距离等方式对特征进行筛选。这个筛选的过程一般都是与分类过程以及分类器无关的，因此过滤器也可以被视为预处理。下面给出这几种过滤规则的原理介绍与相关公式。

相关性（Correlation）过滤规则建立在这样一个假设之上：较好的特征子集所包含的特征应当与结果类型的相关度较高，而与特征集中的其他特征相关度较低，即好的特征集内部的特化程度高，冗余程度低。由于一组特征可以由一个向量表示，所以我们可以使用向量之间的线性相关系数（correlation coefficient）来代表特征之间的相关度。特征 X、Y 都是由多个离散的样本组成的，故特征 X、Y 之间的线性相关系数 r_{XY} 的计算方法如下，其中 cov(X, Y)指 XY 间的协方差，var(X)指特征 X 上各样本的方差，X_i、Y_i 分别指 X、Y 两个特征类型的第 i 个样本，$\bar{X}$、$\bar{Y}$ 分别指 X、Y 类型样本的均值：

$$r_{XY}=\frac{\operatorname{cov}(X,Y)}{\sqrt{\operatorname{var}(X)\cdot\operatorname{var}(Y)}}=\frac{\sum_{i=1}^{N}(X_i-\bar{X})(Y_i-\bar{Y})}{\sqrt{\sum_{i=1}^{N}(X_i-\bar{X})^2\cdot\sum_{i=1}^{N}(Y_i-\bar{Y})^2}} \tag{4.1}$$

信息增益（Information gain）过滤规则[60]首先计算某个特征集的信息熵（Information Entropy），之后计算添加了某个新特征之后的条件信息熵（Conditional Entropy），如果条件信息熵较原信息熵的绝对值更大，我们就认为通过加入这个新的特征，可以获得信息增益，那么这个新的特征也随即被保留。

对于特征集 X，假设存在 t 种取值，即$\{x_1, x_2, ..., x_t\}$，且取值 x_i 的出现概率为 p_i，那么特征集 X 的信息熵 $H(X)$的计算方法如下：

$$H(X) = -\sum_{i=1}^{n} p_i \log p_i \tag{4.2}$$

现在给定一个新的特征 Y，那么 Y 相对于 X 的条件信息熵为：

$$H(Y \mid X) = \sum_{i=1}^{n} p_i H(Y \mid X = x_i) \tag{4.3}$$

此时，如果将特征 Y 加入原特征集 X 中，我们将获得信息增益 $IG(Y|X)$:

$$IG(Y \mid X) = H(X) - H(Y \mid X) \tag{4.4}$$

如果信息增益 $IG(Y|X)$取得正值，我们就认为可以将特征 Y 加入特征集 X 中。如果除特征 Y 之外，还存在多个候选特征，那么应当选取加入特征集 X 之后可以使新特征集获得最大信息增益的那个特征加入。

除了上述两种常用的过滤规则外，我们还可以选用类的内部距离作为过滤规则。类的内部距离过滤规则指的是，好的特征或特征集可以使同一类型样本之间的距离尽量的小，不同类型样本之间的距离尽量的大，而较差的特征集则无法满足上述条件。对于距离的定义，我们可以使用欧式距离、马氏距离[61]或曼哈顿距离。对于样本 A、B，下面分别给出 AB 间的这三类距离的计算公式，其中 $ED(AB)$表示欧氏距离，$MD(AB)$表示马氏距离，$MTD(AB)$表示曼哈顿距离，n 表示样本中包含的特征个数，Φ 表示样本间的协方差矩阵[62]：

$$ED(AB) = \|A - B\| = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2} \tag{4.5}$$

$$MD(AB) = \sqrt{(A-B)^T \Phi^{-1} (A-B)} \tag{4.6}$$

$$MTD(AB) = \sum_{i=1}^{n} |a_i - b_i| \tag{4.7}$$

(2) 封装器（Wrapper）策略

封装器策略不考虑特征之间的关系，评价特征集的方式主要是对使用此特征集得出的分类结果进行评价，相当于将一个分类器封装（Wrap）在特征选择算法之内，故有封装器策略的名称。封装器策略与所使用的分类算法有直接关系，特征选择的过程与分类结果有直接联系，当特征选择的搜索过程结束时，最终的分类结果也会同时产生。封装器策略的时间复杂度要比过滤器策略高出很多，而且对于通过使用不同分类器得到的特征子集，相互之间可能存在较大差异，即不同分类器的最优特征子集可能不一致。不过，相对使用过滤器得到的优化特征子集，

通过本策略得到的优化特征子集在特征规模方面会小很多。

封装器策略一般可以使用分类器精确率作为评价函数。对于网络流量分类问题，我们必须能够有效地区分每个流量是否被分入正确的类型中。对于这个问题，我们可以使用多数投票法（见图 3.3）的思想，分别计算每个聚类簇内部的精确率。首先，对于每个流 w_t 其都有对应的真实类型 T_t，对于每个聚类簇 C，我们也可以通过多数投票法得出一个类型 T_C。因此我们可以将簇内精确率 $precision(C)$ 定义为流类型 T_t 等于 T_C 的流个数与簇内流的总个数之比，即:

$$precision(C) = \frac{\text{Count}(T_t = T_C \mid w_t \in C)}{\text{Count}(w_t \in C)} \tag{4.8}$$

其次，对于分类算法的全局精确率，我们可以使用两种方式进行计算。第一种方式是，使用所有簇内精确率的均值，我们称这种精确率为p_m。设通过聚类算法得到n个簇$\{C_1, C_2, ..., C_n\}$，那么p_m就可以被定义为：

$$p_m = \frac{\sum_{i=1}^{n} precision(C_i)}{n} \tag{4.9}$$

此外，还有另一种精确率的计算方式，此种方式下的精确率可以定义为所有正确投票的流个数与所有流个数的比值，我们称这种精确率为p_c。我们将所有流类型与它自身所处簇的类型一致的流，定义为正确投票的流。设分类算法共对t个流进行了分类，则p_c就可以被定义为：

$$p_c = \frac{\text{Count}(correctly\ voted\ w)}{\text{Count}(all\ flows)} \tag{4.10}$$

上述两种方式都可以作为全局准确率的定义，即特征选择的评价函数可以在p_m或p_c两者之间选择。

使用封装器策略时，我们往往需要搜索对大量的特征子集进行搜索，而且需要在每一个特征子集上完成一次分类过程，如果搜索算法选择失当，如使用完全搜索，那么我们将很难在有限的时间内得到有效的特征选择结果。而如果使用随机搜索或启发式搜索等非完全搜索算法，则可以有效降低特征选择算法的时间复杂度。但考虑到随机搜索的不稳定性，我们一般更倾向于使用启发式搜索策略。而且，大量的研究也表明[63-65]，使用启发式搜索与封装器策略相结合的方式，是一种较为实用的特征选择解决方案。

综上所述，本文在特征选择的过程中将结合使用以下策略：首先，我们将对一个较大的特征集进行过滤，筛去其中的冗余特征、干扰特征等无效信息，并有效缩小特征集的规模；随后，我们使用封装器策略，在一个相对较小的特征空间内，利用启发式搜索算法，最终找出最优的特征子集。

4.3.2 启发式搜索算法与序列前向搜索策略

在第二章中，我们介绍过启发式搜索算法。启发式搜索算法在开始搜索时往往从一个特征集出发，每次遵循一定的规则，向上一步得到的特征集中添加或移除一个或多个特征，直到达到了一定的停止条件时，搜索的动作才会停止。一般来说，启发式搜索算法都需要与封装器策略配合使用。

序列前向搜索策略（Sequential Forward Selection，SFS），是一种自底至顶的搜索策略。本策略从空集出发，并将空集作为第 0 代特征集；每次向上一代特征集中添加一个特征来构成下一代特征集，并且在添加时需要满足最优者加入的条件，即加入这个特征之后的特征集在下一代中是最优特征集；一旦出现某一代特征集在加入任何新特征之后都无法达到更优，那么就停止搜索。下面使用形式化的方式描述这个过程。

考虑一个包含 n 个特征的特征集 F_n，我们希望通过使用序列前向搜索策略，在此特征集的基础之上得到一个优化的特征集。设使用序列前向搜索策略时，第 k 代的特征集为 S_k，此时 S_k 中必定包含了 k 个特征。考虑使用封装器评价函数的情形，设这个封装器策略的评价函数为 f。此时我们应当使用 S_k 完成一次分类，并使用评价函数 f 给出对 S_k 的分类结果的评价 $f(S_k)$。

从 S_k 出发，我们可以得出下一代的特征集 S_{k+1}。相对于包含所有特征的集合 F_n，被排除在第 k 代特征集 S_k 之外的特征组成了 S_{k+1} 的候选特征集 $F_k=\{F_1, F_2, ..., F_{n-k}\}$。使用特征集 F_k 中的各个特征与 S_k 组合并完成分类，得到最高评价结果的特征即可以被保留，即并入 S_k 中以生成 S_{k+1}。上述过程可以表示为以下形式：

$$f(S_{k+1}) = f(S_k, F_t) = \max(f(S_k, F_1), f(S_k, F_2), ..., f(S_k, F_{n-k})), t \in \{1, 2, ..., n-k\} \quad (4.11)$$

如上所述，通过序列前向搜索，我们可以得到每一代的特征集 S_k，共计 n 个特征集，令这些特征集的集合为 $S=\{S_1, S_2, ..., S_n\}$。随后使用与上述过程中相同的评价函数 f，从集合 S 中选出评价结果最高的特征集 S_{opt}，一般来说，我们就认为 S_{opt} 即为通过搜索得到的最优特征集。上述过程可以表示为以下形式：

$$f(S_{opt}) = \max(f(S_1), f(S_2), ..., f(S_n)) \quad (4.12)$$

我们注意到，在对包含 n 个特征的集合 F_n 进行序列前向搜索的全过程中，我们共需要搜索 $\frac{n(n+1)}{2}$ 个 F_n 的子集，并使用这些子集完成分类工作，这是一个时间复杂度为 $O(n^2)$的算法。如果再考虑分类过程本身的开销，这样的搜索过程也是无法在有限时间内被完成的。因此我们需要引入一定的停止条件，在合适的时机停止搜索，减少搜索的特征子集的个数。

对于序列前向搜索策略，一个较为实用的停止条件是，如果当下一代中的所

有特征集都无法获得比上一代特征集更高的评价结果时，我们就立即停止搜索，当前这代中的所有候选特征都不被并入上一代特征集中，而是直接将上一代特征集作为特征选择最终的结果，即最优特征集。下面给出上述停止条件的定义。

给定第 k 代特征集 S_k 和评价函数 f，我们可以得出对特征集 S_k 的评价结果 $f(S_k)$，对于 S_{k+1}，其候选特征集为 $F_k=\{F_1, F_2, ..., F_{n-k}\}$，当满足以下条件时，应当停止搜索过程，并且有 $S_{opt} = S_k$：

$$f(S_k) > \max(f(S_k, F_1), f(S_k, F_2), ..., f(S_k, F_t)), t \in \{1, 2, ..., n-k\} \quad (4.13)$$

下面给出一个例子，描述了在一个包含 5 个特征的特征集上，使用序列前向搜索与封装器评价函数进行特征选择的过程，并且在达到式 4.13 所描述的停止条件时，立即停止搜索过程。示例中，我们认为评价函数 $f(S_k)$的值越大，代表对 S_k 的评价结果越好。示例如表 4.1 所示。

表 4.1 使用 SFS 策略时依赖特征函数 $f(S_k)$特征选取过程

代次	特征集	$f(S_k)$	备注
0	**空集**	**0**	此集合即为 S_0，不包含任何特征。
1	$\{F_1\}$	35	
	$\{F_2\}$	40	
	$\{\boldsymbol{F_3}\}$	**45**	此特征集合即为 S_1，特征 F_3 被选入。
	$\{F_4\}$	30	
	$\{\boldsymbol{F_5}\}$	33	
2	$\{F_1, F_3\}$	25	
	$\{F_2, F_3\}$	57	
	$\{\boldsymbol{F_3, F_4}\}$	**60**	此特征集合即为 S_2，特征 F_4 被选入。
	$\{F_3, F_5\}$	40	
3	$\{F_1, F_3, F_4\}$	73	
	$\{\boldsymbol{F_2, F_3, F_4}\}$	**78**	此特征集合即为 $\boldsymbol{S_3}$，特征 $\boldsymbol{F_2}$ 被选入。
	$\{F_3, F_4, F_5\}$	45	
4	$\{F_1, F_2, F_3, F_4\}$	72	本代中没有特征集使评价函数取得更大的值，因此
	$\{F_1, F_2, F_3, F_5\}$	55	没有特征被选入。
结束	$\{\boldsymbol{F_2, F_3, F_4}\}$	**78**	评价函数 $f(S_k)$在 k=3 时取得最大，最优特征集为 S_3。

4.4 对网络流量统计特征的观察

4.4.1 具有高阶矩的统计特征

在上一章中，我们给出了基于统计特征的网络流量分类的常用特征集（见表3.1），包括基于流行为的特征、基于 TCP 状态的特征以及其他特征。在基于流行为的特征中，只有最大值、最小值、平均值与方差四个统计学中常见的统计量与流中的各类统计样本值组合出现，而其他的高阶统计量都被排除特征集之外，而显然地，网络流量的统计特征无法被均值、方差等较为简单的统计量精确描述。此外，由于高阶统计量在多种分类场景中都有较为重要的地位，一些研究表明[66-69]，通过引入高阶统计量可以获得较好的分类或查找效果。基于上述原因，我们可以考虑引入高阶的统计量，以期能够较为全面完整的刻画网络流量的统计特征及其分布形态。

首先我们介绍统计量的矩。一般来说，对于样本 $X=\{X_1, X_2, ..., X_n\}$，存在两类统计量，第一类叫做原点矩，记为 A_k；另一类叫做中心矩，记为 B_k[70]。两类统计量的定义如下，其中 $\bar{X}$ 为 A_1：

$$A_k = \frac{1}{n}\sum_{i=1}^{n} X_i^{\,k}, k = 1,2,3,... \tag{4.14}$$

$$B_k = \frac{1}{n}\sum_{i=1}^{n} (X_i - \bar{X})^k, k = 2,3,... \tag{4.15}$$

可以看到，均值、方差和标准差等常见统计量都可以通过原点矩与中心距导出，如均值（mean）即为 A_1，方差即为 B_2，标准差（standard deviation）即为 $\sqrt{B_2}$ 等。对此，我们也可以说均值的矩为 1，方差与标准差的矩为 2。假设一定类型网络流量的各类样本值符合一定的分布，由于均值只能描述分布的总体位置，方差或标准差只能描述分布中样本相对均值的离散与偏移程度，所以我们需要引入高阶矩的特征来更为详细的描述分布的具体情况。

首先介绍由三阶矩导出的统计量，偏度（Skewness）。偏度可以衡量随机变量分布的不对称性。偏度的值可以为正值、负值或者无定义，在偏度有定义时，其定义域为$(-\infty, +\infty)$。其中偏度为负时（负偏态），代表着在分布形态的左侧比右侧尾部长，大部分的值位于平均值的右侧；偏度为正时（正偏态）则相反；偏度为零时表示平均值两侧的分布形态相对均匀，但这不能说明该分部为对称分布[71]。偏度的定义为：

$$Skewness(x) = \frac{B_3}{\sqrt{B_2^{\,3}}} \tag{4.16}$$

可以看到，偏度是由三阶中心矩与二阶中心矩三次幂的平方根相除得到的。在已知样本 X 的均值 μ 与标准差 σ 的情况下，偏度的定义还可以转化为如下的形式，其中 $E[X]$代表样本 X 的期望：

$$Skewness(x) = E\left[\frac{(X-\mu)^3}{\sigma^3}\right] = \frac{E[X^3]-3\mu\sigma^2-\mu^3}{\sigma^3} \tag{4.17}$$

接下来介绍由四阶矩导出的统计量，峰度（kurtosis）。峰度又称峰态系数。衡量随机变量分布形态在分布峰值处的聚集程度，另一方面，峰度也反映了分布形态中的尾部厚度。峰度在有定义时，其定义域为(0, +∞)。峰度高（尖峰态）就意味着样本在均值周围的聚集程度较高，分布形态在均值处存在一个高峰形态，此时如果方差较大，则可能是由于低频度的、与均值偏离较大的样本引起的；反之，峰度低（平峰态）就意味着样本在均值周围分布较为分散，分布形态的尾部较厚，此时方差一般也会取得较大的值[72]。峰度的定义为：

$$Kurtosis(x) = \frac{B_4}{B_2^2} \tag{4.18}$$

可以看到，峰度是由四阶中心矩与二阶中心矩的平方相除得到的。在已知样本 X 的均值 μ、标准差 σ 以及偏度 SK 的情况下，偏度的定义还可以展开为如下的形式，其中 $E[X]$代表样本 X 的期望：

$$Kurtosis(x) = E\left[\frac{(X-\mu)^4}{\sigma^4}\right] = \frac{E[X^4]-4\cdot SK\cdot\mu\sigma^3-6\mu^2\sigma^2-\mu^4}{\sigma^4} \tag{4.19}$$

由于使用上述的峰度定义时，正态分布的峰度取值为 3，因此还有另一种对于峰度的定义方式，即在上述峰度定义的基础上减 3，以保证正态分布的峰度取值为 0。此时峰度在有定义时的定义域为(-3, +∞)。本文不讨论此类复杂的情况，因此在后文中，所有涉及对峰度的计算的过程，都采用式 4.18 与式 4.19 中给出的定义。

4.4.2 候选特征集

由在上一章中我们给出的基于统计特征的网络流量分类的常用特征集（见表 3.1），我们仅选取基于流行为的统计特征，并结合上一节介绍的高阶统计量，构造了一个特征集，这个特征集中共包含 21 个特征（Complete Features Set, CFS），如表 4.2 所示（其中相邻包时间差一类不包括流的总体情况）。表中各类特征与统计量的组合，即为候选的特征，如包大小的最大值（max_pkt_size），相邻包时间差的标准差（stde_tim_intv）等。我们将在随后的实验中，使用这个特征集中作为特征选择算法的候选特征集。需要注意的是，这 21 个特征只能描述单方向的流，如果需要使用双向流来对网络流量进行分类，那么需要使用一对这样的特征集。

表 4.2 等待接受特征选择的特征集（CFS）

基于流行为的各类特征	特征数目	统计量
包的大小（pkt_size）	7	最大值（max），最小值（min），均值（mean），标准差（stde），偏度（skew），峰度（kurt），流的总体情况（total）。
负载部分的大小（pld_size）	7	
相邻包的时间差（tim_intv）	6	
包个数（pkt_count）。	1	流的总体情况（total）。

4.5 实验

4.5.1 数据集与实验设计

本实验的实验环境与 3.4.1 节中描述的实验环境一致，均使用 Windows 平台下的 WEKA 分类工具完成分类。本实验使用的数据集为 WAND 工作组的 ISPDSL-II 数据集[73]，这个数据集是一个公开的数据集，而且本数据集中的每个数据包都携带 4 字节的负载信息。此外，本数据集包含的网络流量文件为 ERF 格式，我们在进行分析之前，首先需要将其转换为 pcap 格式。此外，WAND 工作组还提供了一个开源的 DPI（Deep Packet Inspection）工具 libprotoident[74]，这个工具可以对携带超过 4 字节负载信息的数据包进行检测，并给出该数据包的类型信息。有实验表明[75]，使用此工具得到的网络流量类型的正确率较高，因此我们可以使用 libprotoident 工具在 ISPDSL-II 数据集上获取每个数据包的类型，并以此类型作为基于流统计特征网络流量分类的基准类型与评价标准。表 4.3 是 ISPDSL-II 数据集中部分片段的详细情况，其中的开始时间使用格林威治时间（GMT）。

表 4.3 ISPDSL-II 数据集

片段名	开始时间（GMT）	持续时长	片段大小	包个数	流个数
030946.dsl	2010-01-06 03:09:46	0:20:14	1,817,881K	23,513K	148K
033000.dsl	2010-01-06 03:30:00	0:30:00	2,674,844K	34,701K	207K
040000.dsl	2010-01-06 03:04:00	0:30:00	2,763,105K	35,990K	211K
043000.dsl	2010-01-06 03:04:30	0:30:00	2,691,183K	35,154K	220K
050000.dsl	2010-01-06 03:05:00	0:30:00	2,654,638K	34,457K	207K
053000.dsl	2010-01-06 03:05:30	0:30:00	2,749,030K	35,820K	218K
060000.dsl	2010-01-06 03:06:00	0:30:00	2,670,465K	34,820K	217K
063000.dsl	2010-01-06 03:06:30	0:30:00	2,796,628K	36,623K	215K

对于数据集中的上述 8 个数据片段，分别使用 libprotoident 工具得到各个流的基准类型，随后在这 8 个数据集上进行特征选择。本实验中，特征选择的过程分

为两步：首先使用过滤器策略对特征集中相互之间相关性较高的特征筛去，组成中间特征子集；随后再使用封装器策略对剩余特征进行筛选，使用多数投票法对由特征子集生成的聚类簇进行标记，并选取聚类结果的精确率作为评价函数，找到与分类器结合最好的特征，组成最终的最优特征子集，具体来说，可选 p_m（如式 4.9 所示）或 p_c（如式 4.10 所示）作为评价函数。下面分别介绍这两个步骤。

4.5.2 使用基于相关性的过滤器策略进行特征选择

特征之间的线性相关性可以较好的评价特征之间是否存在互信息，即特征 F_1 与 F_2 是否包括一致的信息。如果特征 F_1 与 F_2 之间的线性相关度较高，则我们认为这两个特征携带的信息有一定的重复性，为了降低冗余度、减少分类器工作中所依赖特征的个数，我们可以从 F_1 与 F_2 这两个特征中删去其中一个，即保留 F_1 或 F_2 即可。但是，如果我们直接对 n 个特征之间的相关性进行考查，那么总共需要进行 $\frac{n(n-1)}{2}$ 次的计算。这个过程的时间复杂度是 $O(n^2)$，这是规模是难以接受的。

对于本章的分类问题来说，我们从表 4.2 中可以看出，可能包含互信息的有两组特征：第一组是包大小（pkt_size）和负载部分大小（pld_size），这一组主要涉及数据长度；第二组是相邻包的时间间隔（tim_intv），这一组主要涉及相邻两个包之间的关系。因此，我们可以单独检查每一组内部的相关性情况。

本实验在 ISPDSL-II 数据集中的 030946.dsl 片段完成，而且我们认为相关性分析的结果对其他数据集也是适用的。下面分别介绍这两组特征的情况。

(1) 对相邻包时间间隔组特征的过滤

表 4.4 展示了对相邻包时间间隔组的特征间相关性（行标题中的特征由列标题特征对应的罗马数字代表）。表中的对角线方向代表了特征与其自身的相关性，故不予计算，仅用“----”代表。对角线右上方的数据与对角线左下方的数据是关于对角线对称的，因为线性相关系数具备可交换性，即 $r_{XY}=r_{YX}$。

表 4.4 相邻包时间间隔组的特征间线性相关性

特征 F_2 \ 特征 F_1		I	II	III	IV	V	VI
(I)	max_tim_intv	----					
(II)	min_tim_intv	0.0845	----				
(III)	mean_tim_intv	**0.6830**	0.4805	----			
(IV)	stde_tim_intv	**0.9393**	0.0327	**0.7728**	----		
(V)	skew_tim_intv	0.1037	-0.0553	-0.1379	-0.0270	----	
(VI)	kurt_tim_intv	-0.0021	-0.0523	-0.0216	0.0007	-0.0077	----

通过表 4.4，我们可以看到 max_tim_intv、mean_tim_intv 与 stde_time_intv 三者之间的相关性均较接近 1，即线性相关。因此我们可以考虑仅保留这三个特征中的一个或两个。实验中，我们选择只将 stde_tim_intv 特征删去，即删去包时间间隔的标准差这一个特征。

(2) 对数据长度有关组特征的过滤

随后我们按照相同的方式，对另一组与数据长度有关的特征进行筛选。这一组特征主要包括与数据包长度和负载部分长度相关的共计 14 个特征，表 4.5 给出了这 14 个特征的编号，表 4.6 给出了这 14 个特征的相关性关系。其中表 4.6 的形式与表 4.4 相类似。

表 4.5 与数据长度有关组特征的编号

编号	特征名称	编号	特征名称
1	max_pkt_size	8	max_pld_size
2	min_pkt_size	9	min_pld_size
3	mean_pkt_size	10	mean_pld_size
4	stde_pkt_size	11	stde_pld_size
5	skew_pkt_size	12	skew_pld_size
6	kurt_pkt_size	13	kurt_pld_size
7	total_pkt_size	14	total_pld_size

表 4.6 与数据长度有关组的特征间线性相关性（单位：千分之一）

F_2 \ F_1	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	----													
2	-30	----												
3	**761**	47	----											
4	**875**	-79	666	----										
5	-310	-46	-583	-305	----									
6	7	5	1	-80	-58	----								
7	49	73	89	-24	-282	378	----							
8	**1000**	-30	**762**	**875**	-311	7	48	----						
9	-90	**988**	-9	-130	-26	5	68	-88	----					
10	-6	0	-3	-5	-5	0	0	-7	3	----				
11	-5	-1	-2	-4	-5	0	1	-10	3	**976**	----			
12	-286	-43	-536	-285	**950**	-5	-253	-288	-24	3	13	----		
13	-2	3	-7	-69	17	**803**	281	-3	3	0	7	227	----	
14	50	73	92	-22	-291	374	**999**	50	69	0	1	-264	271	----

由于表 4.6 涉及 14 个特征，故共有 14*13/2=91 个数据描述特征间的线性相关性。为此，我们设定一个相关系数阈值 r_{thres}=0.75，所有相关系数大于等于 0.75 的

特征对都会被单独讨论。对大于 r_{thres} 的数据，我们在表 4.6 中使用粗体标出。下面对这些特征对进行详细的分析。

首先我们可以看到，特征 1（max_pkt_size）与特征 8（max_pld_size）之间相关性为 1，即完全相同，这是由于两个特征都描述了流中数据长度的最大值。虽然这两个特征与特征 3、特征 4 之间也有较高的线性相关性，但从实践经验来看，数据长度的最大值是一个非常重要的特征，不同的数据长度最大值对不同的聚类簇有较高的区分度，因此我们将这两个特征删去一个。

其次，特征 2（min_pkt_size）与特征 9（min_pld_size）之间相关性为 0.998。这两个特征分别描述流中数据长度最小值，与上述特征 1 与特征 8 的情况相似。但由于特征 2 与特征 9 和其他特征之间的相关性较低，所以我们保留这两个特征中的一个。

随后我们观察到，特征 5 与特征 12，特征 6 与特征 13 之间，特征 7 与特征 14 的相关系数也较高。导致这三对特征相关系数较高的原因与上面的情形类似，因为这些特征对都描述了同一统计量下的包长度与负载长度，因此我们分别保留每个特征对中的一个即可。基于上述原因，我们可以猜测特征 3 与特征 10，特征 4 与特征 11 之间也存在较高的相关性。但实验证明，这两对特征的相关性较低。所以我们需要将这两对特征全部保留。

最后，我们可以看到，特征 10（mean_pld_size）与特征 11（stde_pld_size）之间的相关性较高。我们无法直观的给出产生这个结果的原因，但是我们可以看到特征 3（mean_pkt_size）与特征 4（stde_pkt_size）之间的相关系数也可以达到 0.666，就是说，特征 3 与特征 4 之间也存在着类似于特征 10 与特征 11 之间的现象，只是这对特征的相关系数小于我们预先设定的 r_{thres}。在此，我们可以保留特征 10 与特征 11 中的一个特征。

综上所述，我们可以得到一个经过过滤的特征集，这个特征集与表 4.2 中的特征集相比，共减少了 7 个特征，因此我们使用这个新特征集作为特征选择的中间特征集，作为下一步使用封装器策略进行特征选择的输入特征集。表 4.7 给出了使用过滤器策略后，我们得到的包含 14 个特征的中间特征集（Middle Feature Set, MFS），14 个特征分别对应从 A 到 N 共 14 个字母，其中每个特征分别对应其后方括号内的大写字母。在从两个特征中删去其一的情形中，我们全部选择删去负载部分大小这一类的特征。

4.5.3 使用基于启发式搜索的封装器策略进行特征选择

经过上一小节对特征之间互信息的消除，我们得到了如表 4.7 的中间特征集。本小节在此中间特征集的基础上，使用序列前向搜索策略的启发式搜索，结合封

装了 K-means 聚类算法的封装器，进行进一步的特征选择。

表 4.7 经过过滤器策略选择得到的中间特征集（MFS）

基于流行为的各类特征	特征数目	统计量
包的大小（pkt_size）	7	最大值（max, A）最小值（min, B）， 均值（mean, C），标准差（stde, D）， 偏度（skew, E），峰度（kurt, F）， 流的总体情况（total, G）。
负载部分的大小（pld_size）	1	均值（mean, H)。
相邻包的时间差（tim_intv）	5	最大值（max, I），最小值（min, J）， 均值（mean, K），偏度（skew, L）， 峰度（kurt, M）。
包个数（pkt_count）。	1	流的总体情况（total, N）。

首先我们总共需要对 14 个特征进行选择。由于我们需要在特征选择的过程中，使用封装了 K-means 聚类算法的封装器，所以我们不希望封装器过于复杂，因此我们在 K-means 算法中选取 K=50 来完成分类，这是由于 K=50 时分类器已经可以得到较高的精确率（参考图 3.6），且此时分类器的性能开销相对较小。

本实验在 ISPDSL-II 数据集中的 030946.dsl 片段和 033000.dsl 片段上完成，主要使用 p_m 和 p_c 两类分类精确率作为评价函数。表 4.8 给出了在这两个数据片段上搜索的结果，其中选入的特征一列代表某一代次选入的特征，对于的精确率值为选入此特征后，能够达到的精确率。当选入的特征为“----”时，代表选择过程停止，此次选择的最优特征集即由在停止之前所有被选入的特征组成。

表 4.8 使用序列前向选择进行搜索的结果

代次	030946.dsl				033000.dsl			
	选入特征	p_m	选入特征	p_c	选入特征	p_m	选入特征	p_c
1	C	0.51	A	0.56	C	0.50	A	0.56
2	A	0.62	C	0.64	D	0.59	C	0.66
3	E	0.67	D	0.66	A	0.63	E	0.69
4	D	0.71	J	0.69	E	0.66	D	0.71
5	J	0.74	E	0.71	J	0.71	B	0.74
6	B	0.75	L	0.73	B	0.75	F	0.76
7	----	----	F	0.73	G	0.76	----	----
8			----	----	L	0.77		
9					---	----		

通过表 4.8 可以看出，我们可以在两个数据集片段上，使用两类评价函数，获得成分稳定的、长度相近的特征集。综合考虑表 4.8 中出现的特征，最终我们将表 4.9 中给出的特征集选为最优特征集（Optimal Feature Set）。表 4.9 “出现时

的代次”一列中，“X”表示此特征未出现，在 OFS 中的排名一列的排名仅作为编号，不作为绝对的先后顺序。事实上，从表 4.8 可以看出，使用不同的评价函数在不同的数据集上，得到的特征选择效果并不是绝对一致的，我们最终得到的 OFS 也只是一个掺杂一定随机因素的、综合多次选择结果的近似最优特征集。

表 4.9 最优特征集（OFS）

特征		出现次数	出现时的代次				在 OFS 中的排名
C	mean_pkt_size	4	1	2	1	2	1
A	max_pkt_size	4	2	1	3	1	2
D	stde_pkt_size	4	3	5	4	3	3
E	skew_pkt_size	4	4	3	2	4	4
J	min_tim_intv	3	5	4	5	X	5
B	min_pkt_size	3	6	X	6	5	6
L	skew_tim_intv	2	X	6	7	X	7
F	kurt_pkt_size	2	X	7	X	6	8
G	total_pkt_size	1	X	X	7	X	9

4.5.4 优化后特征集对实验结果的影响

首先，我们在全部 8 个数据片段上，对比了使用 CFS 特征集（如表 4.2）、使用 MFS 特征集（表 4.7）和使用上述最优特征集（OFS）进行分类的分类精度（仅比较 p_c）。三个特征集在各个数据片段上的分类精度如图 4.2 所示。可以看出，使用完全特征集与经过特征选择得到的特征子集进行分类，对分类精确率的影响不大，在个别情况下，使用包含特征个数较少的特征集取得的分类效果甚至更好。

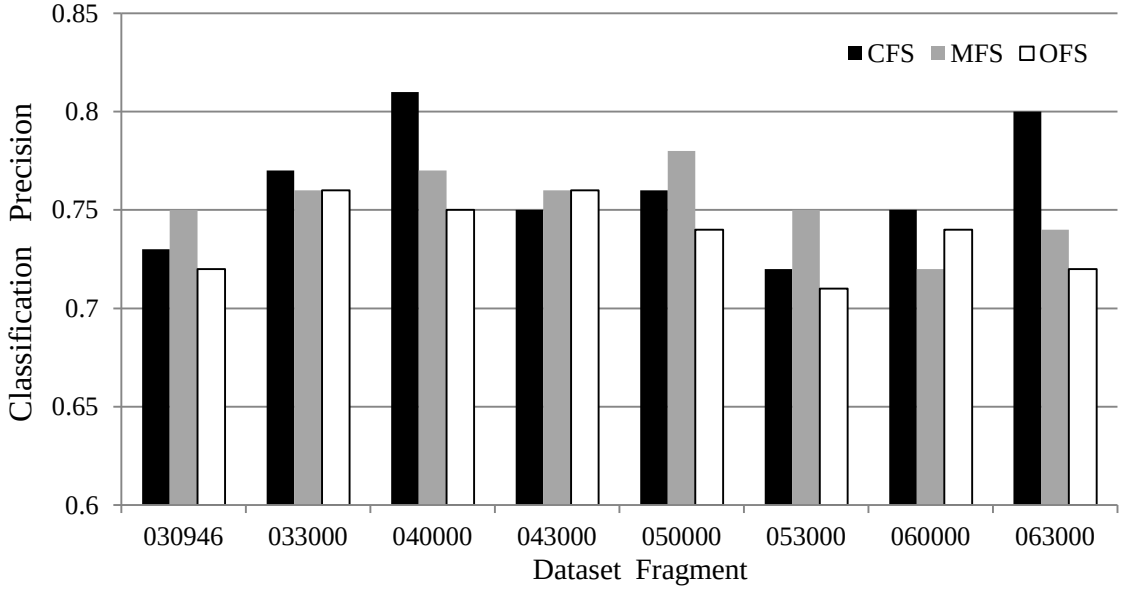

图 4.2 三个特征集在各个数据片段上的分类精度

其次，我们对比使用三个特征集进行分类时所消耗的时间与分类结果的类簇内部距离和，这两项数据均由 WEKA 分类工具在分类结束时给出，对比结果如图 4.3 与图 4.4 所示。

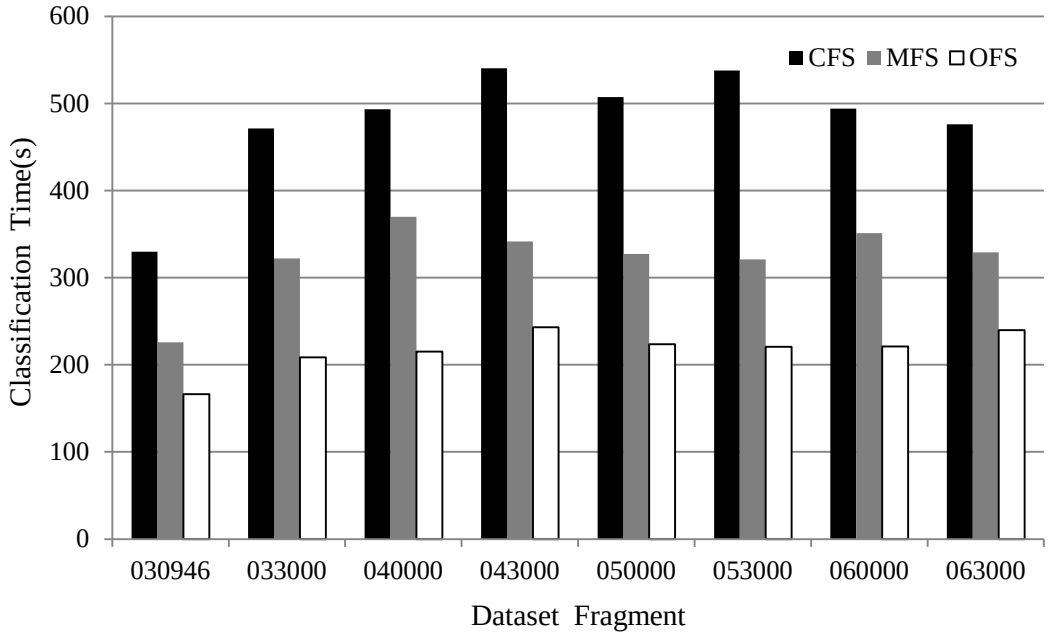

图 4.3 三个特征集在各个数据片段上进行分类时所消耗的时间

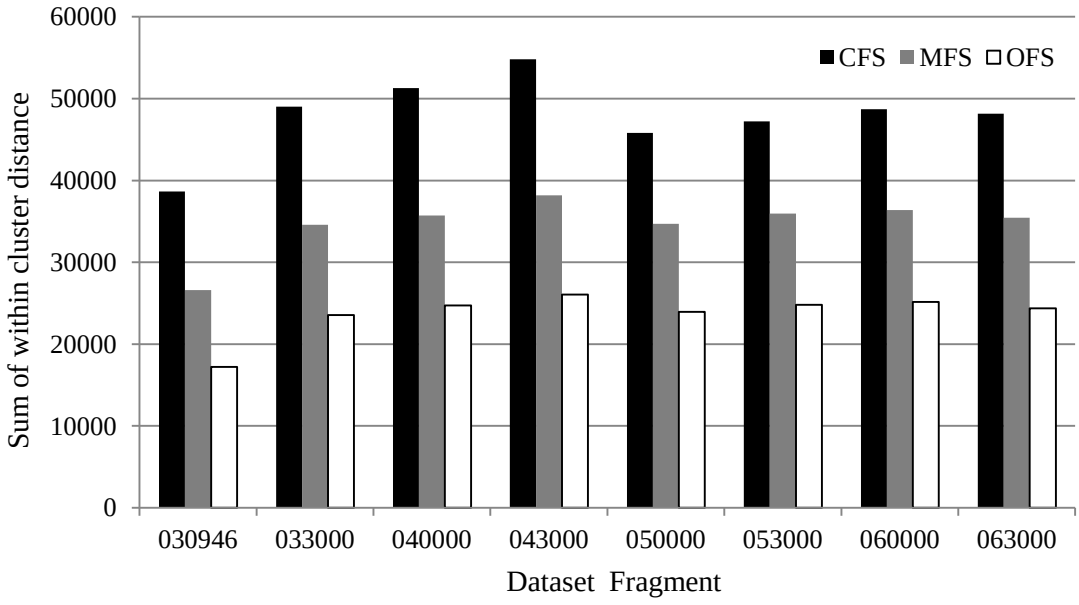

图 4.4 三个特征集在各个数据片段上进行分类时所得到的类簇内部距离和

从图 4.3 与图 4.4 中可以很清晰的看到，依次使用 CFS、使用 MFS 和使用 OFS 进行分类，后者可以较前者在更短的时间内完成分类，而且获得更高的类簇内部距离，即聚类效果更好。

最后，我们给出使用最优特征集对各个数据片段进行分类的结果，即各个数据片段的类型组成，如表 4.10 所示。

表 4.10 各个数据片段的类型组成（单位：百分之一）

数据片段 / 类型	030946	033000	040000	043000	050000	053000	060000	063000
Unknown	33.88	34.82	34.26	34.90	35.72	35.24	32.33	31.07
HTTP/HTTPS	18.85	18.05	17.62	17.66	16.70	18.15	18.14	17.73
SSH/SSL/FTP MAIL/OTHER	5.81	6.99	7.10	6.52	6.65	6.91	6.53	7.04
DNS	1.63	1.68	1.66	1.68	1.57	1.42	1.47	1.03
P2P	39.83	38.45	39.36	39.25	39.37	38.29	41.53	43.12

4.6 本章小结

本章主要提出了一种基于特征选择的网络流量分类技术优化方法，具体内容主要包括以下几点。

第一，结合前人的研究贡献，给出了一个基于流统计特征的一般性网络流量分类框架，框架主要说明数据如何流动、受到何种方式的处理以及各个处理模块如何以一定的顺序协同工作，清晰的展示了分类过程中各个步骤之间的关联，为对分类过程进行局部性的优化提供了条件。

第二，介绍了特征选择算法中的两类评价函数及其使用场景，将评价函数与启发式特征选择算法结合，形式化的描述了序列前向搜索策略的工作过程。

第三，在网络流量分类中引入了具有高阶矩的统计量，给出了常见高阶矩统计量偏度与峰度的计算方法，通过与基于流行为的各类特性相结合，得出了一个包含高阶统计特征的原始候选特征集。

最后，给出本实验所使用的数据集与相关的基准类型生成工具，随后通过在数据集中的不同数据片段上进行分类，并使用特征选择，得到的最优特征集，此外我们还对照未经优化特征集，对最优特征集的分类精度与分类时间进行了评价与讨论。

第5章　总结与展望

5.1 研究工作总结

通过网络传输数据的应用越来越多，对网络流量类型的识别工作也越来越复杂。针对这个问题，本文主要使用了各种分类器算法、特征选择算法，完成基于流统计特征的分类工作。主要工作内容总结如下。

第一，详细介绍了网络流量分类问题产生的背景以及当前的研究现状，对不同的分类技术进行归类，并描述了当前研究中遇到的主要问题。

第二，总结了基于统计流特征的网络流量分类中涉及到的各类技术，详细介绍了网络流量的基本特性与对原始网络流量数据的处理方法，并给出了在分类之前需要使用的特征选择算法与进行分类时需要使用的各种机器学习算法。

第三，详细描述了网络流量分类框架，并使用基础框架完成了网络流量分类的初步工作，介绍了分类工作使用的数据集，以及对分类结果的度量。

第四，通过对统计特征的观察与分析，提出了基于高阶统计量的网络流量统计特征，并对高阶统计量的作用与计算方式进行了展示。

第五，提出了一种基于特征选择的网络流量分类技术优化方法。该方法引入启发式特征选择算法，通过使用特征子集进行分类并评判各个特征子集的分类结果，实现了对最优特征集的选取。

5.2 未来工作展望

随着网络规模的日益扩大，网络流量越来越多，产生网络流量的各种应用也层出不穷，对各种各样的网络流量类型的识别也会越来越显现出其重要性。但是本文给出的方法还存在以下几个方面的不足，希望在下一阶段研究当中能对此有所改进。

第一，本文尚未解决对未知类型流量的精确识别，因此可以在未来使用对特定类簇的标记与识别方法，来完成对未知类型流量的识别。

第二，本文仅仅使用了基于流统计特征的网络流量分类方法，未来可以考虑与基于负载签名信息的网络流量分类方法相结合，提升分类准确率。

第三，本文提出的分类方法是一个离线的分类方法，未来可以将本分类方法改进为一个在线的实时分类系统，进一步扩大本方法的应用场景。

致　　谢

月雅湖畔，春去冬来。青春伴着光阴匆匆流逝，转眼间我已经来到了我研究生生涯的最后一段时光。回首两年半之前的我，从研究生复试结束后就开始计划着自己的研究生生活，希望自己能在研究生期间不断进步。转眼间现在就要毕业了，不敢说自己完成了当初全部的计划，但至少我努力做到了问心无愧。在这两年半的研究生生活中，我从老师、同学和家人身上得到了很多，这都是我一生当中最为宝贵的财富。

首先，我要衷心地感谢我的导师徐明教授。徐老师治学态度之严谨、专业学识之广博、工作热情之饱满，始终让我难以望其项背。人们常说，要想学会做事，须先学会做人，我以为，徐老师在做事和做人这两件事上，都为我做出了最好的榜样，在读研究生的这些日子里，我也一直以徐老师为目标，不仅希望能从他身上学到更多的东西，也希望自己能有一天变得像他一样，成为一个正直的、严谨的、一丝不苟的人。此外，无论在专业学习上的还是在日常生活中，徐老师都给予了我莫大的鼓舞，让我受益匪浅。我还记得徐老师在指导我研究方向时的谆谆教诲，也时常记起徐老师对我读研生活的热切鼓励。独自离家的这两年半中，徐老师给我的帮助与支持，批评与训诫，每每回想，都犹在耳畔，无时无刻不鞭策着我，让我前行。可以说，在研究生期间，我所获得的所有成绩与徐老师的关心与照顾都是分不开的。在此，我祝愿徐老师身体健康、工作顺意。

我也要感谢实验室的郑宁研究员。尽管郑老师公务缠身、日程繁忙，但他也会经常抽空与我们交流。郑老师见解独到、谈吐风趣，每次与郑老师的见面交流都会让人感觉如沐春风，常常让人有茅塞顿开之感。在此，我要感谢郑老师提出的许多宝贵建议，这些建议激发了我的灵感，带我找到进步的方向。

我还要对实验室的徐建教授、张海平副教授和任一支老师表示深深地感谢，他们在每周的研讨班上督促、监督着我们，平时的生活中关心、照顾着我们，感谢他们为实验室带来多元化的学术风气。

另外，我还要感谢实验室的孙偏偏、丁璐、杨洁、代光英、陈宇、肖涛、舒伟欣和王金龙同学，感谢他们在这两年半的时间与我相互陪伴、共同成长。感谢孔飞、尧俊和王黠等已经毕业的师兄师姐们对我的关怀与指导，也要感谢张祝宣、曹俊等还在继续拼搏的师弟师妹们对我的理解与信任。愿我们的同窗之情长存。

感谢我的家人在生活上给予我最大的信任与理解，是他们多年来默默的支持、

诚挚的鼓励和殷切的期望让我有了坚定的信心和强大的动力。他们为我付出了太多，让我总是能够感受到亲情的温暖。

最后，感谢杭州电子科技大学的所有老师与同学；祝愿计算机学院和网络互连与网络安全实验室蒸蒸日上，取得更多优异的成绩；祝福杭州电子科技大学的明天会更好。

参考文献

[1] 林平.网络流量的离线分析[D].北京邮电大学, 2010.

[2] 陈均华,吴春明,姜明.互联网业务特性概述[J].信息工程大学学报, 2009,1(10):41-44.

[3] 焦琳.P2P流量识别方法研究[D].南京邮电大学, 2007.

[4] 吴敏.P2P网络流量控制管理若干关键技术研究[D].南京邮电大学, 2011.

[5] 渠文龙.模式识别算法在网络流量分类中的应用[J].软件, 2013,3(34):72-79.

[6] 熊刚,孟娇,曹自刚等.网络流量分类研究进展与展望[J].集成技术, 2012,1(1):32-42.

[7] 李君.互联网流量分类与识别方法研究[D].南京邮电大学, 2009.

[8] IANA. Service Name and Transport Protocol Port Number Registry[EB/OL]. 2014. http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml.

[9] Chunk Fraleigh, Sue Moon, Bryan Lyles, et al. Packet-level traffic measurements from the Sprint IP backbone[J]. IEEE Network, 2003, 6(17): 6-16.

[10] 王硕.对几种网络流量分类方法的分析与改进[D].浙江大学, 2012.

[11] Subhabrata Sen, Oliver Spatscheck, Dongmei Wang. Accurate, scalable in-network identification of p2p traffic using application signatures[C]. Proceedings of the 13th international conference on World Wide Web. ACM, 2004: 512-521.

[12] Andrew W. Moore, Konstantina Papagiannaki. Toward the Accurate Identification of Network Applications[C]. Proceedings of Passive and Active Measurement Workshop. Springer Berlin Heidelberg, 2005: 41-54.

[13] Michael Finsterbusch, Chris Richter, Eduardo Rocha, et al. A Survey of Payload-Based Traffic Classification Approaches[J]. IEEE Communications Surveys & Tutorials, 2014, 2(16): 1135-1156.

[14] Park Byung-Chul, Won Young J, Kim Myung-Sup, et al. Towards automated application signature generation for traffic identification[C]. IEEE Network Operations and Management Symposium. IEEE, 2008: 160-167.

[15] Application Layer Packet Classifier for Linux[EB/OL]. 2009. http://l7-filter.sourceforge.net/.

[16] 胡婷,王勇,陶晓玲.网络流量分类方法的比较研究[J].桂林电子科技大学学报, 2010,3(30):216-219.

[17] Thuy T. T. Nguyen, Grenville Armitage. A survey of techniques for internet traffic classification using machine learning[J]. IEEE Communications Surveys & Tutorials, 2008, 4(10): 56-76.

[18] Zuev Denis, Moore Andrew W. Traffic Classification Using a Statistical Approach[C]. Springer Berlin Heidelberg, 2005: 321-324.

[19] Andrew W. Moore, Denis Zuev. Internet traffic classification using bayesian analysis techniques[C]. Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems. ACM, 2005: 50-60.

[20] Alice Este, Francesco Gringoli, Luca Salgarelli. Support Vector Machines for TCP traffic classification[J]. Computer Networks, 2009, 14(53): 2476-2490.

[21] Lawrence Berkeley National Laboratory ICSI. LBNL/ICSI Enterprise Tracing Project[EB/OL]. 2005. http://www.icir.org/enterprise-tracing/.

[22] 何震凯.基于聚类分析的网络流量分类研究[D].湖南工业大学, 2009.

[23] Anthony McGregor, Mark Hall, Perry Lorier, et al. Flow Clustering Using Machine Learning Techniques[C]. Springer Berlin Heidelberg, 2004: 205-214.

[24] Jeffrey Erman, Martin Arlitt, Anirban Mahanti. Traffic classification using clustering algorithms[C]. Proceedings of the 2006 SIGCOMM workshop on Mining network data. ACM, 2006: 281-286.

[25] Park Jun-Sang, Yoon Sung-Ho, Kim Myung-Sup. Performance improvement of payload signature-based traffic classification system using application traffic temporal locality[C]. 15th Asia-Pacific Network Operations and Management Symposium. IEEE, 2013: 1-6.

[26]Jun Zhang, Xiao Chen, Yang Xiang, et al. Robust Network Traffic Classification[J]. IEEE/ACM Transactions on Networking, 2014, 99(PP): 1.

[27] 冯神柱.路网轨迹数据的压缩存储技术研究[D].杭州电子科技大学, 2013.

[28] Andrew Moore, James Hall, Christian Kreibich, et al. Architecture of a network monitor[C]. Passive & Active Measurement Workshop. Springer, 2003.

[29] Tcpdump. libpcap-changes[EB/OL]. 2014. http://www.tcpdump.org/libpcap-changes.txt.

[30] 刘文涛.网络安全开发包详解[M].北京:电子工业出版社, 2005:484页.

[31] Tcpdump. TCPDUMP/LIBPCAP public repository[EB/OL]. 2010. http://www.tcpdump.org/.
[32] Chris Sanders,诸葛建伟.Wireshark数据包分析实战[M].北京:人民邮电出版社, 2013:263页.
[33] 王存浩.基于细粒度特征的网络流量管理系统设计与实现[D].厦门大学, 2012.
[34] 钟麟.IPv6流标签的应用探讨[C].2004年中国西部青年通信学术会议.成都, 2004:110-114.
[35] 计智伟,胡珉,尹建新.特征选择算法综述[J].电子设计工程, 2011,9(19):46-51.
[36] 亢婷,魏立力.一种基于粗糙集理论的启发式特征选择算法[J].计算机工程与应用, 2008,30(44):77-79.
[37] Pavel Pudil, Jana Novovičová, Josef Kittler. Floating search methods in feature selection[J]. Pattern Recognition Letters, 1994, 11(15): 1119-1125.
[38] Manoranjan Dash, Huan Liu. Feature selection for classification[J]. Intelligent Data Analysis, 1997, 1-4(1): 131-156.
[39] 饶淑琴.特征选择算法研究及稳定性分析[D].中山大学, 2009.
[40] 贺玲,吴玲达,蔡益朝.数据挖掘中的聚类算法综述[J].计算机应用研究, 2007,1(24):10-13.
[41] Mitchell Tom. Machine Learning[M]. New York: McGraw-Hill, 1997.
[42] Arthur P. Dempster, Nan M. Laird, Donald B. Rubin. Maximum likelihood from incomplete data via the EM algorithm[J]. Journal of the Royal Statistical Society. Series B (Methodological), 1977: 1-38.
[43] Leonard Kaufman, Peter J. Rousseeuw. Finding Groups in Data: An Introduction to Cluster Analysis[M]. John Wiley, 1990.
[44] Wang Yu, Xiang Yang, Zhang Jun, et al. A novel semi-supervised approach for network traffic clustering[C]. 5th International Conference on Network and System Security. IEEE, 2011: 169-175.
[45] Zhang Jun, Xiang Yang, Zhou Wanlei, et al. Unsupervised traffic classification using flow statistical properties and IP packet payload[J]. Journal of Computer and System Sciences, 2013, 5(79): 573-585.
[46] Zhang Jun, Chen Chao, Xiang Yang, et al. Internet Traffic Classification by Aggregating Correlated Naive Bayes Predictions[J]. IEEE Transactions on Information Forensics and Security, 2013, 1(8): 5-15.
[47] Francesco Gringoli, Alice Este, Luca Salgarelli. MTCLASS: Traffic classification

on high-speed links with commodity hardware[C]. IEEE International Conference on Communications. IEEE, 2012: 1177-1182.

[48] 顾成杰,张顺颐.基于改进SVM的网络流量分类方法研究[J].仪器仪表学报, 2011,07:1507-1513.

[49] Seonhwan Hwang, Keuchul Cho, Junhyung Kim, et al. Traffic Classification Approach Based on Support Vector Machine and Statistic Signature[C]. Internet of Things, Smart Spaces, and Next Generation Networking. Springer Berlin Heidelberg, 2013: 332-339.

[50] Yang Baohua, Hou Guangdong, Ruan Lingyun, et al. SMILER: Towards Practical Online Traffic Classification[C]. Proceedings of the 2011 ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems. IEEE Computer Society, 2011: 178-188.

[51] Zhang Jun, Xiang Yang, Wang Yu, et al. Network Traffic Classification Using Correlation Information[J]. IEEE Transactions on Parallel and Distributed Systems, 2013, 1(24): 104-117.

[52] Yu Wang, Yang Xiang, Jun Zhang. Network traffic clustering using Random Forest proximities[C]. IEEE International Conference on Communications. IEEE, 2013: 2058-2062.

[53] 张艺濒,张志斌,赵咏等.TCP与UDP网络流量对比分析研究[J].计算机应用研究, 2010,6(27):2192-2197.

[54] Zhang Jun, Chen Chao, Xiang Yang, et al. Classification of Correlated Internet Traffic Flows[C]. IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications. IEEE, 2012: 490-496.

[55] 汪为汉.IPv6网络流量分类识别技术研究[D].重庆大学, 2012.

[56] Weka 3 - Data Mining with Open Source Machine Learning Software in Java [EB/OL]. 2013. http://www.cs.waikato.ac.nz/ml/weka/.

[57] MAWI Working Group Traffic Archive[EB/OL]. 2014. http://mawi.wide.ad.jp/mawi/.

[58] 王黠.基于微博的位置推测技术研究[D].杭州电子科技大学, 2013.

[59] 毛勇,周晓波,夏铮等.特征选择算法研究综述[J].模式识别与人工智能, 2007,2(20):211-218.

[60] Xiaobo Zhou, Xiaodong Wang, Edward R. Dougherty. Construction of genomic networks using mutual-information clustering and reversible-jump Markov-chain

Monte-Carlo predictor design[J]. Signal Processing, 2003, 4(83): 745-761.

[61] S. J. Roberts, Rudolf Hanka. An interpretation of Mahalanobis distance in the dual space[J]. Pattern Recognition, 1982, 4(15): 325-333.

[62] 许潭潭.基于内容的数据碎片类型识别技术研究[D].杭州电子科技大学, 2013.

[63] Iñaki Inza, Pedro Larrañaga, Rosa Blanco, et al. Filter versus wrapper gene selection approaches in DNA microarray domains[J]. Artificial Intelligence in Medicine, 2004, 2(31): 91-103.

[64] Xiaobo Zhou, Xiaodong Wang, Edward R. Dougherty. Gene selection using logistic regressions based on AIC, BIC and MDL criteria[J]. New Mathematics and Natural Computation, 2005, 01(1): 129-145.

[65] Momiao Xiong, Xiangzhong Fang, Jinying Zhao. Biomarker identification by feature wrappers[J]. Genome Research, 2001, 11(11): 1878-1887.

[66] Keito Kito, Takahiro Murakami. Estimation of an Impulse Response Using Kurtosis[C]. Springer Berlin Heidelberg, 2014: 26-34.

[67] Zhang Xing, Lyu Siwei. Using Projection Kurtosis Concentration of Natural Images for Blind Noise Covariance Matrix Estimation[C]. 2014 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2014: 2870-2876.

[68] Anteneh Ayanso, Paulo B. Goes, Kumar Mehta. Range query estimation with data skewness for top-k retrieval[J]. Decision Support Systems, 2014, 0(57): 258-273.

[69] Walter Briec, Kristiaan Kerstens, Ignace Van de Woestyne. Portfolio selection with skewness: A comparison of methods and a generalized one fund result[J]. European Journal of Operational Research, 2013, 2(230): 412-421.

[70] Wikipedia. Moment[EB/OL]. 2014. http://en.wikipedia.org/wiki/Moment_(mathematics).

[71] Wikipedia. Skewness[EB/OL]. 2014. http://en.wikipedia.org/wiki/Skewness.

[72] Wikipedia. Kurtosis[EB/OL]. 2014. http://en.wikipedia.org/wiki/Kurtosis.

[73] WAND Group. WAND ISPDSL-II Trace[EB/OL]. 2013. http://wand.net.nz/.

[74] WAND Group. WAND Libprotoident[EB/OL]. 2013. http://research.wand.net.nz/software/libprotoident.php, https://github.com/wanduow/libprotoident.

[75] Valentín Carela-Español, Tomasz Bujlow, Pere Barlet-Ros. Is Our Ground-Truth for Traffic Classification Reliable?[C]. Springer International Publishing, 2014: 98-108.

附录

作者在读期间发表的学术论文及参加的科研项目

科研项目：

[1] Android 智能手机用户行为重构分析系统。浙江省新苗计划项目 2013.3-2014.9。

[2] 基于内容的网络安全监察与预报服务系统(2010C11050)。浙江省科技厅重大科技专项重点项目，2012.06-2013.11。

软件著作：

[1] Android 智能手机用户行为重构分析系统 V1.0。登记号：2014SR029696，2013.11。

[2] 基于内容的网络安全监察与预报服务系统 V1.0。登记号：2014SR024667，2013.09。