**文章编号**:1007-5321(2017)02-0102-04 **DOI**:10.13190/j.jbupt.2017.02.017

# 基于用户代理及互联网知识的应用识别

桂小林, 刘 军, 乔媛媛, 池沐聪, 雷振明
(北京邮电大学 网络体系构建与融合北京市重点实验室, 北京 100876)

**摘要**: 为了在网络端识别安装在移动设备中的应用名称,提出了一种新的应用识别方法. 该方法能根据模糊或不完整的用户代理(UA)特征字符串识别应用名称. 首先,从超文本传输协议请求报文中获得 UA 特征;然后,从互联网中获取该 UA 对应的特征文本;最后,对该特征文本进行分析进而识别应用名称. 基于大规模网络端采集的网络流量数据进行实验,结果表明所提方法的识别效果优于其他已知方法.

**关 键 词**: 应用识别; 互联网知识; 用户代理; 超文本传输协议
**中图分类号**: TN911.22 **文献标志码**: A

# Identifying Apps Based on User-Agent and Web-Knowledge

GUI Xiao-lin, LIU Jun, QIAO Yuan-yuan, CHI Mu-cong, LEI Zhen-ming
(Beijing Key Laboratory of Network System Architecture and Convergence, Beijing University of Posts and Telecommunications, Beijing 100876, China)

**Abstract**: A new method was proposed to identify apps via analyzing network traffic. This method can identify apps according to incomplete and ambiguous features of user-agent. Firstly, the user-agents from hyper text transfer protocol headers was extracted; then, the related text feature was obtained from Internet by using the corresponding user-agent; finally, the apps by analyzing the obtained text features was identified. Experiments are conducted by using hyper text transfer protocol traffic traces collected from Gn interface. It is shown that the proposed techniques are superior to the known methods.

**Key words**: application identification; web-knowledge; user-agent; hyper text transfer protocol

由于超文本传输协议(HTTP,hyper text transfer protocol)流量在移动互联网中的占比较大[1-2],理解其构成与使用特征对运营商进行网络规划、流量监控、市场分析至关重要. 现有的 HTTP 流量精细化识别方法可以分为 3 类:1) 将流量分为较细粒度的业务类型[3-4];2) 将流量关联到具体的网站[5];3) 将流量关联到具体应用[6-9]. 目前,第 3)类方法仅停留在应用字符串特征的提取,而对于大部分的中文应用,这些字符串特征通常是模糊的或者是不完整的,例如,Host 字段包含“qq”既有可能来自腾讯 QQ 客户端的请求,也有可能来自微信客户端的请求;UA 字段为“amap”的应用名称不是“amap”,而是“高德地图”. 提出了一种结合用户代理(UA,user agent)特征及该特征对应的网络知识的方法来识别应用名称.

## 1 应用识别系统

### 1.1 问题定义

输入数据是一个 HTTP 请求记录集合,可表示为 $R=\{r_1,r_2,r_3\cdots\}$,一条访问记录 $r_i=\{i,u\}$ 包含该请求多个维度的信息,在此仅列出与本研究内容相关用户标识 $i$ 及用户代理 $u$. 抽取用户代理的特

**收稿日期**: 2016-01-29
**基金项目**: 高等学校学科创新引智计划(111 计划)项目(B08004); 国家自然科学基金项目(61601414)
**作者简介**: 桂小林(1984—), 男, 博士生, E-mail:guixiaolinde@gmail.com; 雷振明(1951—), 男, 教授, 博士生导师.

征并对这些特征按用户数从高到底排序得到一个 UA 特征集合 $X=\{x_1,x_2,x_3,\cdots\}$. 取 $x_i$ 在搜索引擎中搜索,取排名前 5 的搜索结果的标题作为 $x_i$ 的特征文本$\{t_{i1},t_{i2},t_{i3},t_{i4},t_{i5}\}$. 将该特征文本按一定规则分词,并统计各个词出现次数(用 $n$ 表示)得到词的集合 $Y_i=\{\langle y_{i1},n_{i1}\rangle,\langle y_{i2},n_{i2}\rangle,\cdots\}$. 在得到分词结果以后,应用识别问题就转化成从一系列候选词中选择一个词来作为 UA 特征所对应的应用名称,用式(1)表示为

$$\Gamma(x_i)=T(Y_i) \tag{1}$$

其中:$\Gamma$ 为 $x_i$ 所对应的应用名称,$T$ 为选择词语的规则.

## 1.2　系统结构

应用识别系统框图如图 1 所示,各模块的功能分别描述如下.

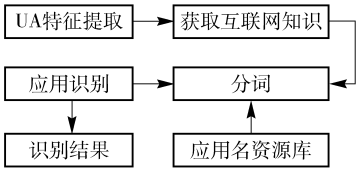


图 1　应用识别系统框图

1) UA 特征提取. HTTP 协议规定客户端在发出 HTTP 请求时,需要提供客户端的名称标识符,该标识符就是 HTTP 报文头部的 UA 字段. 除此之外,某些服务器还需要客户端的其他信息,如手机类型、客户端操作系统类型等. 因而,用户代理字段可能包含多个字符串,而需要的只是 UA 特征字符串. 因此,需要滤除 UA 字段中的无用信息,如 release、version 等字符串.

2) 获取互联网知识. 从 UA 特征文件中取出用户数排名前 100 的应用 UA,然后将这些 UA 作为关键字逐个地在搜索引擎中搜索,对于每一个 UA 的搜索结果,取前 5 位的标题字段作为该 UA 的特征文本. 测试了百度、谷歌搜索引擎,发现采用谷歌搜索引擎效果较好.

3) 应用名资源库. 为了避免分词算法将应用名分成不同的词语,如将“百度贴吧”分成“百度”“贴吧”两个词语,实现了一个从应用商店中爬取应用名称的程序,并构建一个应用名资源库存储爬取的所有应用名称,定义该资源库中的词的词性为 appName. 因而,当分词工具检测到一个字符串与应用名资源库中的某个词语相同,就会将其分成一个词语.

4) 分词[10]. 将从互联网上获取的 UA 特征文本转化成词语. 采用的中文分词工具来自 github 开源项目 Ansj,该开源项目是在 TREC Novelty 比赛中排名第一 ICTCLAS 中文分词系统的 Java 实现. 该分词工具的输入为句子,输出为词/词性对.

对于每一个应用特征,统计其特征文本中各词语出现次数. 由于词性为 appName 的词是已知的应用名称,这些词是该 UA 特征对应的应用名称的可能性更大,因此在计算这些词的出现次数时应该赋予一个较大的权重 $w$. 式(2)用来确定各个词语出现次数,$O$ 表示已知的应用名集合.

$$\Phi_1(x_m,y_n)=\begin{cases}\Phi(x_m,y_n)w, & x_m\in O\\ \Phi(x_m,y_n), & x_m\notin O\end{cases} \tag{2}$$

为了更清楚地描述应用特征与分词结果的对应关系,用表 1 来描述指纹与应用名对应关系. 表的行名是应用字符串特征,用 $x$ 表示;表的列名表示应用名称,用 $y$ 表示. $m$ 表示该表的行号,$n$ 表示该表的列号,当 $m=n=1$ 时,$\Phi(x_1,y_1)=55$,其含义是当使用 Micromessenger 作为关键词在谷歌中搜索时,在排名前 5 的搜索结果标题中,“微信”字符串出现的次数是 55 次.

**表 1　应用字符串与应用名称对应表**

| 字符串 | 微信 | QQ | 优酷 |
|---|---|---|---|
| Micromessenger | 55 | 4 | 0 |
| QQ | 3 | 30 | 0 |
| Youku | 1 | 0 | 22 |

5) 应用识别. 由于不同应用字符串所对应的候选应用名称出现次数差别较大,为了便于对不同应用字符串统一设置选词门限,应该将同一应用字符串所对应的词出现的次数用式(3)标准化. 式(3)表明,对于某一个应用字符串 $x_m$,其特征文本中的词 $y_n$ 出现的次数越多,$\Psi$ 值越大,$x_m$ 的应用名称为 $y_n$ 的可能性越大.

$$\Psi(y_n)=\frac{\Phi_t(x_m,y_n)}{\sum\Phi_t(x_m,*)} \tag{3}$$

在寻找应用的 UA 特征过程中,发现同一个应用可能有多个 UA 特征,如微信应用相关的报文中出现的 UA 可能有 Micromessenger、Wechat、tencent-connect. 这些 UA 特征可以分为两类,一类是可以用来唯一确定该应用名称的 UA 特征,称之为专用代理,如 Micromessenger 与 Wechat;另一类是为实现应用的某一功能而存在的功能代理,如 tencentcon-

nect,它是腾讯公司用于各应用之间共享用户数据的代理. 由于功能代理没有实际的应用与其对应,利用功能代理在搜索引擎中搜索,难以搜索到相应的应用名称,从而降低了应用名称识别的准确率. 为了提高识别准确率,需要将这些功能代理滤除. 由于功能代理没有具体的应用名称与其对应,用其在谷歌中搜索而获得的词语的出现次数分布通常比较均匀,因而功能代理对应的词语不确定性程度更大. 利用这个特征,用式(4)计算的阈值滤除应用字符串中的功能代理.

$$H(\Psi(y_n)) = -\sum_i \Psi(y_n^i)\mathrm{lb}\Psi(y_n^i) \qquad (4)$$

式(4)的意义在于,对于任一字符串 $x_m$,若根据 $\Psi$ 值计算出来的熵值越大,说明该 UA 为功能代理的可能性越大. 设置阈值 $\lambda$ 将这些功能代理滤出.

除此之外,由于网络环境的不确定性,会得到一些错误的字符串特征,这些错误的字符串特征在谷歌中的搜索结果仅包含少量的词语(甚至为空),设置词语个数门限为 $N$,可滤除一部分错误的 UA 特征.

最后,对任意一个 UA 特征 $x_m$,取 $\Psi$ 值最高的 $y_n$ 作为该 UA 对应的应用名称.

## 2 评估方法

对于应用名称的识别,识别的准确程度以及是否正确地将功能代理滤除是本研究最关心的问题. 因此,采用准确率及召回率来衡量识别结果. 其相关符号定义如下.

真正(TP,true positive):UA 特征与应用名称关联正确的数量 $N_{\mathrm{TP}}$;

真负(TN,true negative):UA 特征与应用名称关联错误的数量 $N_{\mathrm{TN}}$;

被滤出的专用代理数量:$N$;

总字符串特征数量:$N_{\mathrm{TP}} + N_{\mathrm{TN}} + N$;

准确率:UA 与应用名称正确关联占总 UA 数量的比例;

召回率:正确识别的 UA 与错误识别的 UA 数之和占总 UA 的比例.

准确率和召回率的定义如下.

$$\text{准确率} = \frac{N_{\mathrm{TP}}}{N_{\mathrm{TP}} + N_{\mathrm{TN}}} \qquad (5)$$

$$\text{召回率} = \frac{N_{\mathrm{TP}} + N_{\mathrm{TN}}}{N_{\mathrm{TP}} + N_{\mathrm{TP}} + N} \qquad (6)$$

## 3 实验评估

为了验证所提方法的有效性,采用现网流量数据进行实验,并与 Dai 等[6-7]提出的方法进行比较.

### 3.1 数据采集

实验数据的采集环境如图 2 所示,数据来自运营商骨干网 Gn 口,其采集过程为:首先,流量监控系统中的相关设备对通过网络中的数据进行 1∶1 全镜像;然后,将一定时间内,如 5 min 五元组(源 IP,目的 IP,源端口,目的端口,协议)信息相同的通信报文记录组合成流的形式,并且每 5 分钟的流记录构成一个文件;最后,这些文件通过数据传递工具程序上传到数据分析集群文件系统(HDFS,Hadoop distributed file system)中. 通过对每一天流记录文件中的 IMEI 字段进行去重计数,可以得到网络中的用户数. 共采集了两天的数据进行实验,这两天数据是取自不同省份的 Gn 口,其基本情况如表 2 所示.

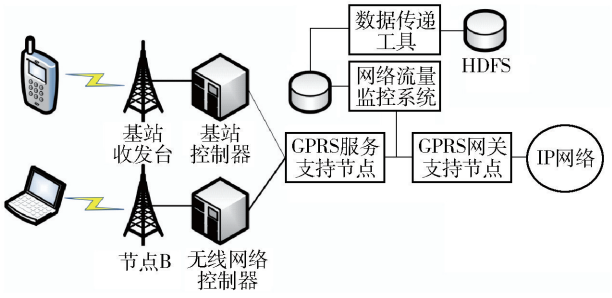


图 2 数据采集环境

**表 2 数据基本情况**

| 数据 | 采集时间 | 流文件/GB | 用户数 | 流数 |
|---|---|---|---|---|
| *A* | 2015-01-24 | 204 | 78 529 833 | 303 963 471 |
| *B* | 2015-04-04 | 137 | 74 577 871 | 204 065 252 |

### 3.2 实验结果与分析

为了验证所提方法的有效性,从数据集 *A* 和 *B* 中选出用户数排名前 1 000 的 UA 进行实验,3 种方法的参数设置如下.

1)对于 Dai 等[6]提出的 NetworkProfiler 方法,设置其结构化相似性参数 Tg 为 0.6;文中没有明确说明是否使用了 UA 字段,但笔者在实现该文算法时使用了该字段,这样可以提高识别的准确率.

2)对于 Miskovic 等[7]提出的 AppPrint 方法,参数设置与该文保持一致.

3)对于所提方法,在实验中发现,当设置参数 $w$、$N$、$\lambda$ 分别为 10、2、6 时,两组数据中识别效果最

优. 图 3 展示了这 3 种方法的识别准确率.

NetworkProfiler 对于数据集 $A$ 与 $B$ 的识别准确率在 35% 左右，而 AppPrint 方法的识别准确率在 28% 左右. 由于 NetworkProfiler 及 AppPrint 方法都没有过滤功能 UA 的步骤，因此，这两种方法的召回率都为 100%. 由于 NetworkProfiler 方法使用了 AppStore 中应用唯一的标识符，识别准确率要高于 AppPrint 方法，但是，在数据 $A$ 与 $B$ 中只有 5.3% 的应用产生了这种标识符. 而提出的方法在召回率为 95% 的情况下，准确率能达到 85% 左右. 因此，与 NetworkProfiler、AppPrint 方法相比，在引入了应用字符串的互联网文本特征之后，可以将应用名称识别准确率提升 50% 左右. 为了更直观地展示这 3 种方法的识别结果，表 3 展示了用户数排名前 10 位的应用识别情况.

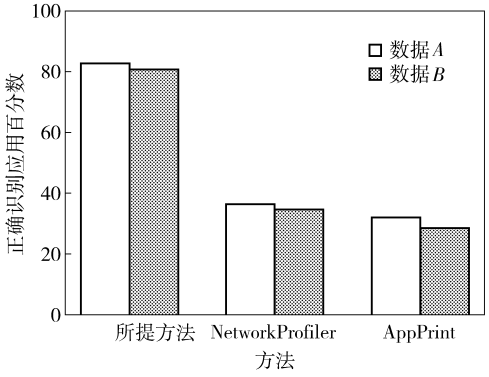


图 3　识别准确率比较

**表 3　前 10 位的应用字符串识别结果比较**

| 应用 | NetworkProfiler | AppPrint | 所提方法 |
|---|---|---|---|
| 微信 | 微信 | Micromessenger，qq | 微信 |
| qq 浏览器 | mqqbrowser | mqqbrowser | qqbrowser |
| qq 空间 | qq 空间 | qzone，qq | qq 空间 |
| 腾讯地图 | soso map | soso map，qq | 腾讯地图 |
| 高德地图 | amap | amap location | 高德地图 |
| safari | safari | safari | safari |
| qq | qq | qq | qq |
| uc 浏览器 | ucbrowser | ucbrowser | uc 浏览器 |
| 支付宝 | alipay | alipay，zhifubao | 支付宝钱包 |
| qq 音乐 | qqmusic | qqmusic，qq | qq 音乐 |

## 4　结束语

利用互联网知识扩充应用特征，进而识别应用名称，并使用运营商现网采集到的网络流量数据对该方法进行评估，证明了该方法的有效性. 虽然所提方法已经达到较高的识别准确率，但是，还有一定的提升空间. 后期将通过增加应用字符串特征（如 URL 的 Host 字段）进而增加特征文本以提升识别准确率. 除此之外，将使用目前最快的大数据处理工具 Spark 框架提升识别效率.

**参考文献：**

[1] Fiadino P, Bar A, Casas P. HTTPTag: a flexible on-line HTTP classification system for operational 3G networks [C]// 2013 IEEE Conference on Computer Communications Workshop. Turin: IEEE, 2013: 71-72.

[2] Gember A, Anand A, Akella A. A comparative study of handheld and non-handheld traffic in campus Wi-Fi networks[C]// Passive and Active Measurement. Berlin: Springer, 2011: 173-183.

[3] Li Wei, Moore A W, Canini M. Classifying HTTP traffic in the new age[C]// ACM SIGCOMM. Seattle: ACM, 2008: 17-22.

[4] 林平，余循宜，刘芳，等. 基于流统计特性的网络流量分类算法[J]. 北京邮电大学学报，2008，31(2)：15-19.
Lin Ping, Yu Xunyi, Liu Fang, et al. A network traffic classification algorithm based on flow statistical characteristics[J]. Journal of Beijing University of Posts and Telecommunications, 2008, 31(2): 15-19.

[5] Liu Jun, Li Tingting, Chen Gang. Mining and modeling the dynamic patterns of service providers in cellular data network based on big data analysis[J]. China Communications, 2013, 10(12): 25-36.

[6] Dai Shuaifu, Tongaonkar A, Wang Xiaoyin, et al. Networkprofiler: towards automatic fingerprinting of Android apps[C]//2013 IEEE Conference on Computer Communications. Turin: IEEE, 2013: 809-817.

[7] Miskovic S, Lee G M, Liao Y, et al. AppPrint: automatic fingerprinting of mobile applications in network traffic [C]//Passive and Active Measurement. Berlin: Springer, 2015: 57-69.

[8] Xu Qiang, Erman J, Gerber A, et al. Identifying diverse usage behaviors of smartphone apps[C]// Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference. Berlin: ACM, 2011: 329-344.

[9] Gui Xiaolin, Liu Jun, Chi Mucong, et al. Fine-grained analysis of cellular smartphone usage characteristics based on massive network traffic[J]. The Journal of China Universities of Posts and Telecommunications, 2016, 23(3): 70-75.

[10] Sun Jian. Ansj 1.4 [EB/OL]. [2014-10-10]. https://github.com/NLPchina/ansj_seg.