Journal of Information Processing Vol.24 No.2 247–254 (Mar. 2016)

[DOI: 10.2197/ipsjjip.24.247]

Regular Paper

Combining Communication Patterns & Traffic Patterns to Enhance Mobile Traffic Identification Performance

Sophon Mongkolluksamee[1,a)] Vasaka Visoottiviseth[1,b)] Kensuke Fukuda[2,c)]

Received: June 1, 2015, Accepted: November 5, 2015

Abstract: The bandwidth of a mobile network is limited and exhausted very fast with the huge number of mobile devices and applications. In order to manage and utilize the limited bandwidth, precise mobile application identification is required. In this work, the combination of communication patterns extracted from graphlet and traffic patterns represented by packet size distribution is studied for enhancing the performance of identifying mobile traffic. There are no privacy concerns for identifying traffic with our technique; it is also effective against the complexities of mobile traffic. The real traffic of five famous mobile applications (Facebook, Line, Skype, YouTube, and Web) is used in our evaluation. The identification performance is high (0.96) of F-measure even considering only a random 50 packets of traffic in a 3-minute duration. While identifying applications, the effect of other mixed background traffic is also studied and mitigated by filtering out short lived flows with a flow duration condition. The high identification performance is still maintained after this filtering process.

Keywords: graphlet, mobile application, application identification, communication patterns, traffic classification, random forest

1. Introduction

The demand for using mobile networks is increasing while the mobile network has a limited bandwidth. The bandwidth is exhausted very fast with the huge number of mobile devices and wide variety of applications. The increasing of mobile data traffic is confirmed by Cisco's report that it enlarged from 1.5 exabytes per month in 2013 to 2.5 exabytes per month in 2014 [2]. To handle this huge traffic, mobile carriers have to fully understand the characteristics of their traffic in order to manage and optimize the bandwidth usage. Identifying mobile application traffic is one of the necessary steps. For example, it can help to offload the tolerant or low priority application traffic to other networks and provide a Quality of Service (QoS) to some specific applications.

The complexity of mobile traffic means that the identifying of application traffic is not easy. Current mobile applications rely on new Internet technologies, such as cloud services. Using these services causes the communication pattern and destination host of these applications to be uniform. Mobile applications also usually tunnel their traffic over HTTP/HTTPS [3], [4], [5], [6], [7], [8], use dynamic ports and encrypt their traffic. Moreover, a mobile network frequently leads to the loss of a link and also a high bit error rate due to an instability in the wireless channel [9]. As a result, existing techniques cannot work well on classifying mobile traffic.

The current state-of-the-art traffic classification can be categorized into four approaches [10]: port-based, payload-based, statistics-based (or flow feature-based), and host behavior-based approaches. The tunneling traffic, dynamic ports and encryption are main obstacles that cause the port-based and payload-based approaches be ineffective in classifying mobile traffic. Moreover, user privacy is affected when accessing payload contents. The problems mentioned here may not affect statistics-based and host behavior-based approaches that extract traffic features from network and transport layers for identifying traffic. However, the instability of a mobile network directly affects traffic features especially a time-related feature that will affect the accuracy of a flow feature-based approach. The communication patterns [11], [12] between host/application and the destination host are extracted for identifying the traffic type in the host-behavior-based approach. As a result of the popular use of cloud services such as Google application program interface (API), Facebook API, and content delivery network (CDN) services, several mobile applications commonly communicate with the same cloud servers. Therefore, it is difficult to distinguish between these applications simply from the communication pattern.

The combination of the statistics-based and the host-behavior-based approaches for gaining high accuracy in mobile traffic identification is highlighted in this study. The host behaviors or communication patterns are observed from graphlets, while the packet size distribution is selected as the feature in the statistics-based approach. This study relies on the real traffic trace from five popular mobile applications captured from the mobile device: Facebook, Line, Skype, YouTube, and Web. The experiment results confirm that this approach gains a high performance of F-measure

[1] Faculty of ICT, Mahidol University, Nakhon Pathom, Thailand
[2] National Institute of Informatics/Sokendai, Tokyo 101–8430, Japan
[a)] sophon.mon@student.mahidol.ac.th
[b)] vasaka.vis@mahidol.ac.th
[c)] kensuke@nii.ac.jp

This paper is an extended version of work published in Ref. [1].

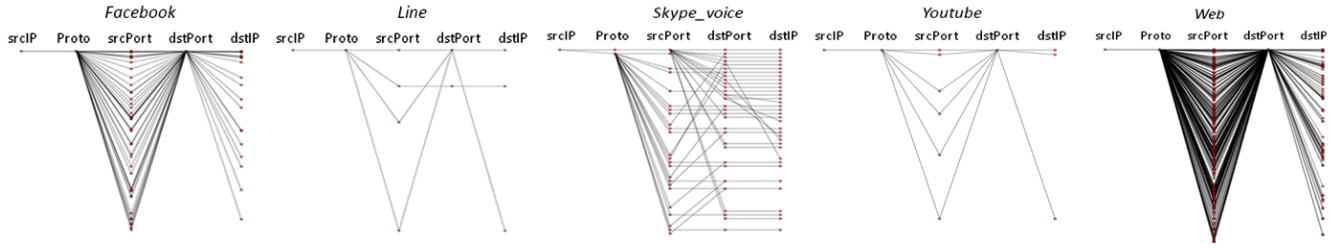

Fig. 1 Examples of 5 application graphlet.

with low variance for identifying these applications. However, the mixture of target traffic and background traffic will decrease the identification performance. We introduce a mitigation technique for reducing the effect of background traffic. We show that the application identification performance with background traffic is still high with our mitigation technique.

2. Preliminary

Identifying mobile applications requires traffic features which tolerate the complexities of mobile traffic. We leverage on communication patterns, which are learned from grahplet, and packet size distribution.

2.1 Communication Pattern Based Traffic Feature

Communication patterns present the characteristics of host/application when it interacts with the other hosts. Communication patterns cannot easily be changed by the network conditions. Therefore, they are appropriate for using in mobile traffic identification. BLINC [11], which is a state-of-the-art behavior-based technique, uses graphlet for capturing and virtualizing communication patterns of applications. Graphlet consists of three basic elements: (1) several columns related to the network attributes, (2) a node that represents the unique occurrence of a network attribute, and (3) an edge forming links between two nodes in adjacent columns. The graphlet's column usually conforms to the attributes of 5-tuple of flow/packet ordered by source IP address (srcIP), protocol (proto), source port (srcPort), destination port (dstPort), and destination IP address (dstIP). An example of graphlet of the five selected applications is presented in **Fig. 1**.

Since the graphlet cannot be directly used by machine learning techniques, Himura et al. [12] solved this problem by developing a technique to extract numerical features from a shape of graphlet. This technique will be used for extracting a communication pattern of mobile application traffic in this study.

The five types of shape-related information are:

(1) n_i: the number of nodes in graphlet column i, where i is a column index.
(2) o_{ij}: the number of one edge (or one degree) nodes in direction $i:j$, where i, j are column index and $j = \pm i$
(3) $\mu_{i:j}$: the average number of edges in direction $i:j$, where i, j are column index and $j = \pm i$.
(4) $\alpha_{i:j}$: the maximum number of edges in direction $i:j$, where i, j are column index and $j = \pm i$.
(5) $\beta_{i,i+1}$: the backward degree of the maximum degree node.

2.2 Statistics-Based Network Traffic Related Features

Traffic statistical features can be categorized into size-related and time-related features. As a result of the unstable nature of a mobile network, the size-related features are more appropriate than the time-related features for identifying mobile traffic.

Different applications have different patterns of packet usage in terms of the number of packets and size of packets in transferred data. For example, messaging applications transfer a small number of small packets in their communication, while streaming applications transfer a high volume of large packets. Valenti et al. take advantage of this fact to create Abacus algorithm [13] for characterizing traffic patterns of P2P-TV applications. The numbers of exchanged packets with other peers in a specific time window are counted into a histogram. The bin size of the histogram is in base-2 logarithm scale. For example, the frequency of the number of exchanged packets between $(2^{i-1}, 2^i]$ is presented in bin_i, and the last bin (bin_B) is used for counting the frequency between $(2^{B-1}, \infty)$. The value of each bin is normalized by the total number of exchanged packets and used as a signature of each application. In this study, the packet size distribution of mobile application traffic will be applied with this technique.

3. Dataset

We create our dataset by collecting packet traffic of five famous mobile applications (Facebook, Line, Skype, YouTube, and Web) directly from a mobile device (Android 4.2.2) via tcpdump. Since we intend to provide a technique that mobile carriers can apply to measure their mobile network, only the traffic over 3G network is considered. The WiFi network on the mobile device is disabled and only the traffic through 3G network is allowed. The bandwidth of 3G network in this study is 6.92 Mbps in download and 0.42 Mbps in upload. The packets from the applications are separately captured in the packet trace file (PCAP) format for 30 minutes and stored on local memory. The application versions and activities are presented in **Table 1**.

Normally, many applications concurrently run on a mobile device. However, due to the limitation of mobile devices such as battery, computing power, and screen size, users usually interact with only one application at a time and the others are in background mode. The active application is not only one that generates traffic. Other applications in the background mode also generate traffic as well. The study on application traffic in background mode of Qualcomm [14] found that news applications generate four to six requests per hour, location based applications generate two to three per hour, and social networking applications generate one to four per hour. The traffic of background mode applications is lower than the active application traffic; however, the effect of background traffic for identifying mobile application traffic is not negligible.

Table 1 Mobile application specification.

apps	version	activity
Facebook (FB)	15.0.0.20.16	Comment, like, view images
Line (LN)	4.9.1	messaging and sticker
Skype (SK)	4.9.0.45564	Voice calling
YouTube (YT)	6.0.13	Watching video
Web (WB)	4.2.2 (Android browser)	Surfing without video

Table 2 TCP network flow summary (no background traffic).

apps	#flow	#total byte	#total packet	#byte /flow	#packet /flow
FB	78	1,133 K	2,111	13,794	26
LN	31	159 K	450	4,072	13
SK	69	40 K	406	504	5
YT	66	8,563 K	9,703	193,748	210
WB	293	2,475 K	4,850	9,154	17

Note: All values are an average from the 100 of 3-minute traces

Table 3 UDP network flow summary (no background traffic).

apps	#flow	#total byte	#total packet	#byte /flow	#packet /flow
FB	12	3,011	23	250	2
LN	23	647	5	254	2
SK	100	1,780 K	16,640	21,544	197
YT	7	1,454	12	211	2
WB	35	7,037	60	208	2

Note: All values are an average from the 100 of 3-minute traces

Therefore, we prepare two different datasets: (1) the active application traffic without background traffic and (2) the active application traffic with background traffic.

3.1 No Background Traffic

Only one of the selected five applications runs in active mode during the traffic capturing period (30 minutes). All unwanted applications and notifications are closed in order to prevent generating unwanted traffic from background mode applications. Moreover, the traffic from Android OS itself is also filtered out by using the IP address of Google's servers. The traffic of each application is captured for ten of 30-minute traces. Falaki et al. [3] reported that the interaction period between user and mobile application is about 10-250 seconds per session. Therefore, each of our 30-minute traces is sliced to ten of 3-minute trace files. Finally, we have 100 of 3-minute trace files per application.

3.1.1 Dataset Details

Six characteristics of each application traffic are studied: (1) the destination port usage, where we consider the mobile device as a source, (2) the number of network flows, (3) the total number of bytes transferred, (4) the total number of packets, (5) the number of bytes per flow, and (6) the number of packets per flow. These characteristics are extracted from TCP and UDP protocol traffic. Other protocols are filtered out. We also focus on bi-directional traffic flows with the same 5-tuple satisfying one of following conditions: (1) no active packet longer than 30 seconds, (2) occurring of a TCP FIN packet, and (3) flow duration in excess of 60 seconds. TCP/UDP flow information for each application is presented in **Table 2** and **Table 3**. The important destination port usage is shown in **Table 4**.

3.2 Dataset with Background Traffic

Three of 30-minute traffic traces for each selected application which contains background traffic are captured. The background data consists of two different sources. The first one is notifications of applications in background mode for Facebook, Line and Skype. The second one is traffic of Android OS that frequently communicates with Google's servers. In addition, three of 30-minute traffic traces for background traffic (Android, Facebook, and Line) are separately captured for studying the characteristics of background traffic. The 30-minute traces are split to 3-minute traces with the same procedure as non-background traffic cases.

Table 4 Destination port usage (no background traffic).

apps	TCP port:usage(%)	UDP port:usage(%)
FB	443:97.8%, 80:1.93%	53:100%
LN	443:76.04%, 80:23.95%	53:97.5%
SK	High:84.38%, 443:7.94%	High:94.26%
YT	443:82.49%, 80:16.47%	53:100%
WB	80:85.36%, 443:14.5%	53:100%

High = port number higher than 5,000
Note: the usage found from flows of the 100 of 3-minute traces

Table 5 Background traffic characteristics.

App	Total packet per flow	Total byte per flow
Andorid OS	66% less than 10 packets 34% on average 21 packets	66% less than 1,000 bytes 34% on average 4,477 bytes
Facebook	64% less than 10 packets 36% on average 28 packets	54% less than 1,000 bytes 46% on average 6,632 bytes
Line	80% less than 10 packets 20% on average 14 packets	80% less than 1,000 bytes 20% on average 1,699 bytes

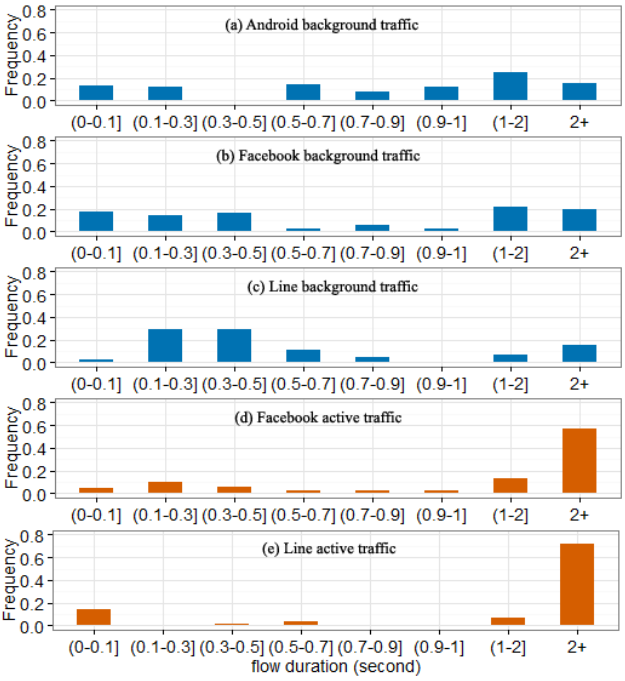

Fig. 2 Background traffic flow duration.

3.2.1 Dataset Details

The mixture of traffic will affect the performance of identifying the target application especially for the low volume of traffic application such as Line. The characteristics of background traffic are studied in order to mitigate this issue. Using the packet level information it is difficult to distinguish between target traffic and background traffic. We analyze flow durations, the total number of packets per flow, and the total bytes per flow. The bi-directional network flow of each background traffic type is extracted with the

same as criteria in the non-background traffic case. Since there is a small amount of UDP traffic in background traffic, we will consider only the TCP traffic. Flow characteristics (total packets per flow and total bytes per flow) of the three background traffic types are listed in **Table 5**. The flow duration of the background traffic comparing with those of active traffic of Facebook and Line is presented in **Fig. 2**. From the figure, we find that the flow durations of background traffic are mostly less than 2 seconds. Thus, we will use this as a threshold for filtering out background traffic in this study.

4. Methodology

The identification method consists of three main steps. In the first step, non TCP/UDP packets are filtered out. TCP and UDP packets are forwarded to the second step for extracting the traffic features. The second step consists of two processes: (1) graphlet-based feature extraction and (2) packet size related feature extraction. The graphlet-based feature extraction generates a shape-based feature vector (35 dimensions). The packet size distribution extraction generates two feature vectors for TCP and UDP protocols. There are in total 12 dimensions for each protocol. All feature vectors are concatenated to be a final feature vector. The 59-dimension final feature is passed to a machine learning classifier in the last step to identify application.

4.1 Feature Extraction

4.1.1 Graphlet-based Features Extraction

The shape-based feature vectors [12] are extracted from the shape of the graphlet. They represent the communication pattern of an application to the other hosts. A graphlet, which is formed by the TCP and UDP packets, has five columns in this study. These columns order by the 5-tuple of network flow, which are srcIP, proto, srcPort, dstPort, and dstIP respectively. The shape-based features are extracted from the five shape-related information of each graphlet column.

The graphlet-based feature vectors are presented between index 1 to 35 in **Table 6**. Feature 1 to 4 relate to the srcIP column. The features of the protocol column are index 5 to 13. The feature 14 to 22 are features of the srcPort column. The index 24-31 and 32-35 are features of the dstPort and dstIP column respectively.

Table 6 Feature index and feature name.

index	name	index	name	index	name	index	name
1	n_1	16	$o_{3:4}$	31	$\beta_{4:3}$	46	T_bin_{10}
2	$o_{1:2}$	17	$\mu_{3:2}$	32	n_5	47	T_bin_{11}
3	$\mu_{1:2}$	18	$\mu_{3:4}$	33	$o_{5:4}$	48	U_bin_0
4	$\alpha_{i:j}$	19	$\alpha_{3:2}$	34	$\mu_{5:4}$	49	U_bin_1
5	n_2	20	$\alpha_{3:4}$	35	α_{54}	50	U_bin_2
6	$o_{2:1}$	21	$\beta_{3:2}$	36	T_bin_0	51	U_bin_3
7	$o_{2:3}$	22	$\beta_{3:4}$	37	T_bin_1	52	U_bin_4
8	$\mu_{2:1}$	23	n_4	38	T_bin_2	53	U_bin_5
9	$\mu_{2:3}$	24	$o_{4:3}$	39	T_bin_3	54	U_bin_6
10	$\alpha_{2:1}$	25	$o_{4:5}$	40	T_bin_4	55	U_bin_7
11	$\alpha_{2:3}$	26	$\mu_{4:3}$	41	T_bin_5	56	U_bin_8
12	$\beta_{2:1}$	27	$\mu_{4:5}$	42	T_bin_6	57	U_bin_9
13	$\beta_{2:3}$	28	$\alpha_{4:3}$	43	T_bin_7	58	U_bin_{10}
14	n_3	29	$\alpha_{4:5}$	44	T_bin_8	59	U_bin_{11}
15	$o_{3:2}$	30	$\beta_{4:3}$	45	T_bin_9		

Note: T=TCP, U=UDP

4.1.2 Packet Size Distribution Feature Extraction

Since applications are different in terms of packet size usage, the distribution of packet size is useful for distinguishing applications. Abacus algorithm [13] is selected for capturing the packet size distribution of mobile applications in this study. The histogram has 12 bins from bin_0 to bin_{11}. The number of bins is derived from the MTU (1,500 bytes) of Ethernet standard. The packet size between $(2^{k-1}, 2^k]$ are counted into bin^k, where k is a bin index. The last bin is used to count packet size between $(2^{10}, \infty)$ bytes. The feature vectors of the TCP packet size are presented by index 36 to 47 and the UDP packet size are presented by index 48 to 59 in Table 6.

4.2 Machine Learning Classification

Based on the intensive evaluation over 179 classifiers in Ref. [15], Random Forest (RF) [16] and Support Vector Machine (SVM) with Gaussian kernel show the best performance overall. We selected RF for its high accuracy and a small number of parameters. Moreover, it provides an importance metric of each feature which helps us to interpret and understand the results. It also helps selecting the set of features for further improving our proposed technique. RF is an ensemble learning method based on masses of decision tree. The randomForest package in R is selected for this study. This package has two main parameters: ***ntree*** and ***mtry***. The ***ntree*** parameter is used to specify the number of trees in the forest. The number of input variables for each split is presented with ***mtry***. The setting of these two parameters directly affects the classification performance. From our study of parameter setting, we found that the most appropriate values of ***ntree*** and ***mtry*** are 500 and the square root of $\boldsymbol{p}$, where $\boldsymbol{p}$ is the total number of variables, respectively. The detail of parameter tuning is described in Section 6.

5. Case Study

Some examples of communication and traffic pattern of selected applications are presented in this section, i.e., graphlet and packet size histogram. Both graphlet and histogram are created from traffic during 3 minutes of non-background traffic dataset.

5.1 Graphlet-Based Feature

Examples of graphlet for each selected application are presented in Fig. 1. They illustrate the communication pattern of applications with other hosts. Facebook's graphlet shows the behavior of the client-server application. It has a dense shape because the Facebook application creates many connections to Facebook services that host on the server of Akamai, a well-known cloud service provider. It varies high port numbers to communicate with the server's port such as port number 80, 443, and 53. The shape of Line's graphlet is sparse. Line application creates a few connections to a few servers. Line will create additional connections to cloud servers such as Akami and Amazon in order to send or receive a Line sticker. For Skype, which is a peer-to-peer (P2P) application, its graphlet shape reveals that there is no fix source/destination port in communication. In addition, considering the graphlet shape of YouTube and Line, they are very similar. YouTube communicates with a few of YouTube's servers. Thus,

distinguishing them with only the communication pattern is not an easy task. The most dense client-server graphlet shape is the web. It confirms the normal behavior of the Web that connects many servers in order to display web pages composed of several pieces of files.

5.2 Distribution of Packet Size

Examples of the histogram of packet size distribution of five applications are shown in **Fig. 3**. Facebook, Line, and Web are TCP-based applications. YouTube can dynamically use TCP and UDP to achieve the best performance for users but it mainly uses TCP in this experiment environment. In TCP signaling packets are small (e.g., 40-64 bytes), and the number of them directly depends on the number of connections. YouTube has the lowest number of signaling packets as shown in bin6 of in Fig. 3 (g), because it connects to a limited number of servers. Facebook and Web have higher signaling packets as shown in Fig. 3 (a) and (i) respectively. In terms of packet size, Facebook , YouTube, and Web use larger than 1,024 bytes packet to transfer data. These packets are counted into bin11 of the histogram. Line uses small packet sizes to carry the user message therefore the TCP signaling packets and data packets are mostly counted into bin6 as shown in Fig. 3 (c). These four applications use UDP for DNS service. Due to the fact that the DNS packet size is less than 512 bytes, the distribution of UDP of these applications is between bin6 to bin9. The packet size distribution of Skype presented in Fig. 3 (e), (f) is notable. It uses both small TCP (32-128 bytes) and UDP (64-256 bytes) packets in its communication. For both of TCP and UDP, Facebook and Web show a similar pattern of packet size usage, therefore using only the packet size distribution might make it difficult to distinguish these two applications.

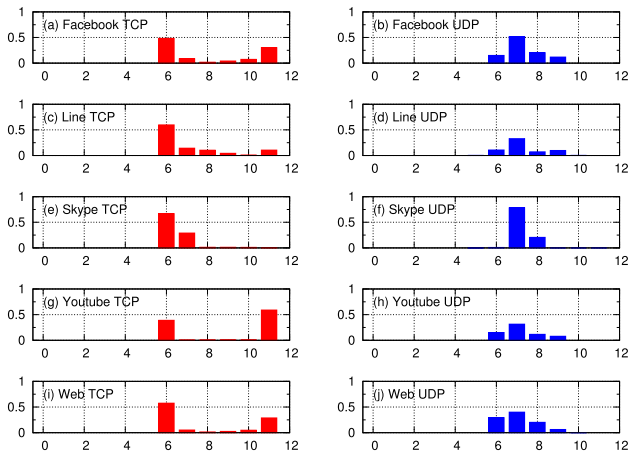

Fig. 3 TCP and UDP packet size distributions for 5 applications (x-axis: bin index, y-axis: PDF).

6. Parameter Tuning

Setting an appropriate value of parameters is significant to the classification performance of the RF algorithm. This section investigates the best setting of ***ntree*** and ***mtry***, which are two main parameters for randomForest package in R, for classifying mobile traffic. Setting ***ntree*** to 500 and ***mtry*** to $\sqrt{p}$, where p is the total number of variables, are the default of these two parameters. To find the best setting of these two parameters for our dataset, the value of ***mtry*** and ***ntree*** are varied. ***mtry*** is varied from 1 to p and ***ntree*** is set to three difference values: 100, 500, and 1,000. The minimum Out-OF-Bag (OOB) error is used as an indicator for the best combination of these two parameter settings. RF uses a bootstrapping technique to select the different training sample from the original sample for growing each tree in the forest. About one-third of the original sample is left out, called out-of-bag sample, and used for testing. The average of misclassification of OOB sample is the OOB error estimate.

Figure 4 presents the OOB error estimated from varying ***mtry***=1 to 59 (the total number of features), and ***ntree***=100, 500, and 1,000. The vertical dashed line presents the default of ***mtry*** ($\sqrt{59} \sim 7$). Each point is the average of 10 experiments. The dataset, here, is traffic without background dataset. This dataset is applied with randomly-selecting-50-packet scheme which is detailed in Section 8.1. The OOB errors of setting ***ntree*** equal to 500 and 1,000 are low and very close to each other. The OOB error is very high when setting ***mtry***=1 and becomes low when setting ***mtry***=6 to 15, and then increases again for large ***mtry***. The number of trees in the forest directly affects the time and resource consumption in the training phase. For this reason the default value of ***ntree***=500 and ***ntree***= $\sqrt{p}$ is the optimum setting for this experiment.

7. Effective Traffic Features

The random forest algorithm provides the ranking of important variables (or features) in a classification. Understanding importance features is crucial for selecting the set of features. It also helps to interpret and understand the result. We set the best parameters (***mtry***=7 and ***ntree***=500) as shown in Section 6.

Figure 5 shows the importance scores of each feature obtained by 100 runs. A higher value of the importance represents more discriminative power. The x-axis represents the feature index, which corresponds to Table 6, and the y-axis presents the impor-

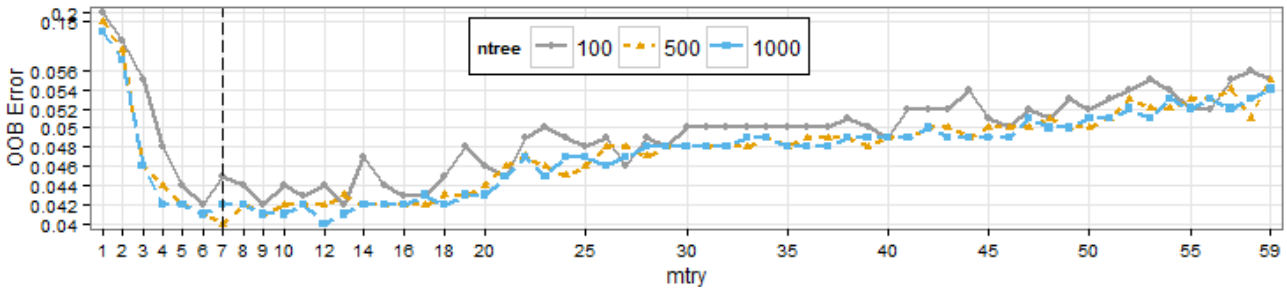

Fig. 4 Random Forest parameter tuning.

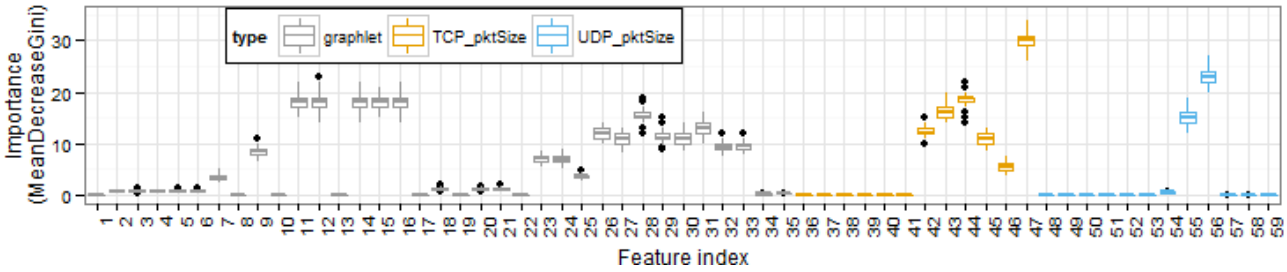

Fig. 5 Important Features.

tance score. For the graphlet feature, there are two outstanding groups whose scores are more than 10. The first group is feature 11, 12, 14, 15, and 16. The second group is feature 26 to 33. The first group presents the source port behavior of each application while the second group emphasizes the relation of the destination port and destination host. For the TCP packet size feature, there are five features (42, 43, 44, 45, and 47) whose importance scores over 10. Among these, feature 47 is the most discriminative. It presents the number of large packets (>1,024 bytes). Regarding the UDP packet size feature, there are only two features (55 and 56) whose importance scores are more than 10. In this dataset, these two features are important because they can be used to differentiate between Skype and DNS.

8. Experiment

8.1 No Background Traffic Experiment Setting

Three main experiments with different packet selection schemes are conducted for evaluating the proposed technique. The performance to identify the mobile applications of all possible sets of features (the graphlet feature, packet size distributions, and the combination of them) is investigated for all experiments. In the first experiment, all packets of 3-minute trace are used to extract communication patterns and packet size distributions. The results of this experiment are a baseline for those in the other experiments. In the second experiment, the first 100 consecutive packets are chosen. The randomly sampling packet scheme is applied in the last experiment. There are three sub-experiments for the last experiment, which are randomly sampling 50, 100, and 200 packets.

To evaluate the proposed technique, 10-fold cross validation is applied for each experiment. The ninety of 3-minute traces of dataset are used for training and the remaining ten traces are used for testing. The cross validation is run ten times. Thus, there are total 100 rounds of each experiment. F-measure of each run of experiment is calculated for measuring the performance of each feature type for classifying the mobile application traffic. The definition of F-measure is shown in Eq. (1). A higher F-measure (close to 1.0) indicates better performance.

$$F\text{-}measure = \frac{2 \times precision \times recall}{(precision + recall)} \quad (1)$$

8.2 With Background Traffic Experiment Setting

We conduct three experiments: random 50 packet, random 100 packet and the first 100 packet. We train the model with non-background traffic dataset and test with background traffic dataset. In order to see the effect of background traffic, we consider two cases: filtered short-lived flows and non-filtered short-lived flows. The key idea of filtering is that background traffic is mostly short-lived and filters out by using flow duration as shown in Section 3. All flows shorter than two seconds are filtered out and the packets of the remaining flows will be used to extract the features vector. Each experiment is run for ten rounds.

9. Results

9.1 No Background Traffic

We compute the average and standard deviation of F-measure from 100 trials of each experiment. **Figure 6** represents the comparison of F-measure of random 50 packets, the first 100 packets, and all packets experiments. The combination of features achieves the highest F-measure for all experiments; and it gains 0.96 even using only 50 packets. Considering the error bar, the combination of features shows the narrowest error bar, i.e., the highest precise result.

Moreover, it is important to discuss costs in the classification process such as classification time, memory usage, and CPU usage. The sampling scheme can balance the costs and the classification accuracy. The appropriate number of sampling packets is investigated by varying the number of sampling packets (50, 100, and 200 packets). **Figure 7** presents the result of the sampling packet experiment. For the larger number of sampling packets, the F-measure of graphlet feature and packet size distribution increase but the combination of the two features achieves the highest and most steady result in the experiments.

As presented in Fig. 6 and Fig. 7, the combination of the features not only improves an overall performance of classification, but also improves the result per individual application. **Figure 8** shows that the graphlet feature is good for classifying Web and Facebook but the packet size distribution works very well for Line, Skype, and YouTube. However, the combination of these two features can fill up weak points in each other and gains the highest F-measure for classifying all applications.

9.2 With Background Traffic

On the one hand, **Fig. 9** reveals that background traffic decreases the identification performance comparing with Fig. 6, as expected. On the other hand, **Fig. 10** shows the better performance with filtering out short flows. In addition, **Fig. 11** confirms that the performance of identifying each application is still high after filtering short flows. In summary, the proposed method achieves high classification ability for five applications with background traffic by introducing the flow duration constraint.

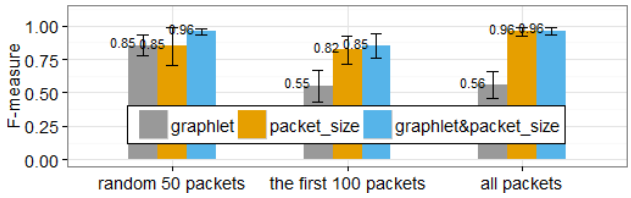

Fig. 6 Comparison of F-measure for random 50 packets, 100 packets and all packets experiments without background traffic (error bar is +/- sd).

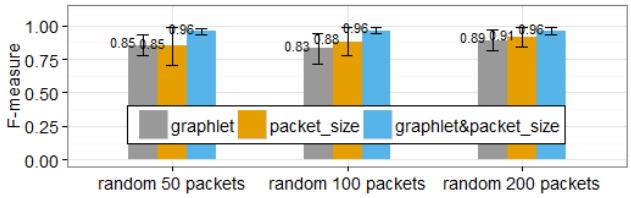

Fig. 7 Comparison of F-measure for random packets experiments without background traffic.

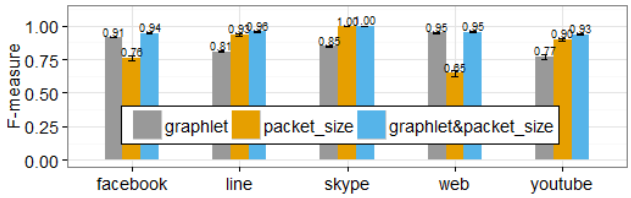

Fig. 8 Comparison of F-measure of each application for random 50-packet experiment (without background traffic).

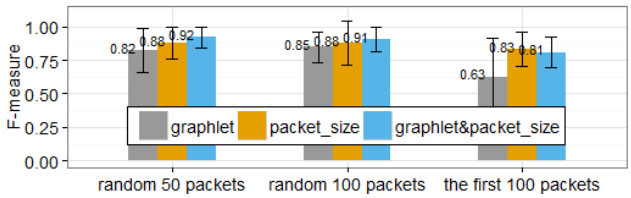

Fig. 9 Comparison of F-measure with background traffic (non-filtered).

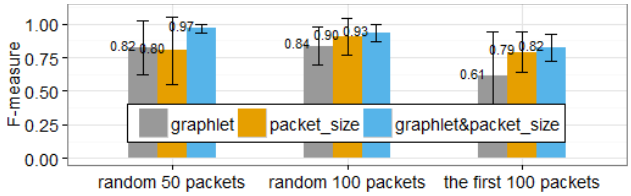

Fig. 10 Comparison of F-measure of short flow filtering.

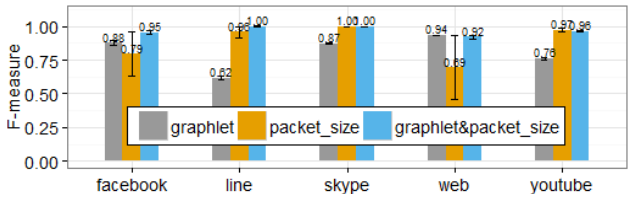

Fig. 11 Comparison of F-measure of random 50 packet short flow filtering.

10. Related Works

Researchers try to identify mobile application traffic for many reasons such as network management [6], [17], [18], [19], [20], security [21], [22], and mobile resources management [7], [23]. Because most of the current mobile applications use HTTP protocol in their communication, they focus on distinguishing mobile applications with HTTP header information. Using user-agent field in HTTP header is the simplest way [5], [6], [17], [18]. Normally, developers are encouraged to put the application information in this field but it is easily relaxed by developers. This may cause this technique to be ineffective. The destination service host in Host field of HTTP herder is another useful source of information. However, the widespread use of cloud services in mobile applications causes these applications to communicate to the same cloud servers. Applications look similar in terms of communication patterns and thus, it is difficult to identify an application based on service host IP address or domain name.

In order to solve previous issues, advance techniques are discussed. Dai et al. create the state machine of Host field and HTTP request pattern for using as a signature to identify applications [6]. Miskovic at el. use the unique number of a token group which is counted from the token (word) in HTTP request header for identifying the application [20]. However, the patterns in HTTP requests are quite rapidly updated because of the changing application functions. By assuming that each application has a deterministic pattern of control in application layer header, Han et al. proposed a technique to find the entropy in an application header from a randomly selected packet to identify an individual application [19]. However, these techniques adopt the payload-based approach which violate the user's privacy and cannot function with the encrypted traffic at all.

11. Discussion and Future Work

Many proposed works focus on the application layer information to improve an identification performance. They consider information from a single viewpoint. In contrast, our technique takes advantage of using information from different viewpoints to gather more characteristics of applications. The packet size distributions reveal microscopic characteristics and the communication patterns provide macroscopic characteristics of traffic. This approach can improve the accuracy and it has no privacy concerns.

Misidentification in our technique primarily originates by unsteadiness in applications traffic, particularly in the starting and nearly complete content downloading period, and instability in the wireless channel. The instability of the wireless network will increase the numbers of reconnection and retransmission in TCP. For example, five percent of Web is misidentified as Line and YouTube. In the nearly complete downloading web page period, the numbers of destination servers and traffic volume become small, then the shape of the graphlet and packet size distribution are similar to those of Line and YouTube.

To apply our technique in the mobile network, gateway is an appropriate location. At this position, network operators can monitor both inbound and outbound traffic to obtain traffic features. Since our technique currently identifies traffic type based on an individual IP address and needs to create a graphlet before extracting required features it may introduce a delay for identification. However, if we consider using only important graphlet features, we can extract features directly instead of creating a graphlet in advance to improve the identification process.

In this study, we focused on five major Android applications.

However, because of the wide variety of applications, users will run many applications at the same time. Such situations make identification difficult. Thus, one important future work is to improve application identification under more complicated situations.

12. Conclusion

This study presented a new technique for identifying mobile application by combining host-behavior-based and statistical-based traffic classification approaches. The shape-based graphlet features which represent the communication pattern of application and packet-size distribution features are both extracted from the real application traffic via 3G network. Then, the machine learning algorithm is applied to the identification. This technique works well under the complexities of mobile traffic without any privacy concerns. Our results highlight that the proposed method achieves high F measure (0.96) and low errors even for 50 randomly sampled packets. The combination of these two features compensates the weak points in each other and improves the identification result. This study also proposed a robust method to mitigate the effect of background traffic by filtering short duration flows (less than 2 seconds) corresponding to the majority of background traffic. The high identification performance is still maintained with this filtering process.

References

[1] Mongkolluksamee, S., Visoottiviseth, V. and Fukuda, K.: Enhancing the Performance of Mobile Traffic Identification with Communication Patterns, *Proc. IEEE COMPSAC'15*, pp.336–345 (2015).

[2] CISCO System: Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014–2019 White Paper (2015).

[3] Falaki, H., Lymberopoulos, D., Mahajan, R., Kandula, S. and Estrin, D.: A first look at traffic on smartphones, *Proc. IMC'10*, pp.281–287 (2010).

[4] Maier, G., Schneider, F. and Feldmann, A.: A first look at mobile hand-held device traffic, *Proc. PAM'10*, pp.161–170 (2010).

[5] Xu, Q., Mao, Z.M., Arbor, A., Erman, J., Park, F., Gerber, A., Pang, J. and Venkataraman, S.: Identifying Diverse Usage Behaviors of Smartphone Apps, *Proc. IMC'11*, pp.329–344 (2011).

[6] Dai, S., Tongaonkar, A., Wang, X., Nucci, A. and Song, D.: NetworkProfiler: Towards automatic fingerprinting of Android apps, *Proc. INFOCOM'13*, pp.809–817 (2013).

[7] Wei, X., Gomez, L., Neamtiu, I. and Faloutsos, M.: Profiledroid: Multi-layer profiling of android applications, *Proc. Mobicom'12*, pp.137–148 (2012).

[8] Lee, S.W., Park, J.S., Lee, H.S. and Kim, M.S.: A study on Smartphone traffic analysis, *Proc. APNOMS'11*, pp.1–7 (2011).

[9] Goff, T., Moronski, J., Phatak, D. and Gupta, V.: Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments, *Proc. INFOCOM 2000*, Vol.3, pp.1537–1545 (2000).

[10] Kim, H., Claffy, K., Fomenkov, M., Barman, D., Faloutsos, M. and Lee, K.Y.: Internet traffic classification demystified: Myths, caveats, and the best practices, *Proc. CoNEXT'08*, pp.1–12 (2008).

[11] Karagiannis, T., Papagiannaki, K. and Faloutsos, M.: BLINC: Multilevel traffic classification in the dark, *ACM SIGCOMM Computer Communication Review*, Vol.35, pp.229–240 (2005).

[12] Himura, Y., Fukuda, K., Cho, K., Borgnat, P., Abry, P. and Esaki, H.: Synoptic graphlet: Bridging the gap between supervised and unsupervised profiling of host-level network traffic, *IEEE/ACM Trans. Networking*, Vol.21, No.4, pp.1284–1297 (2013).

[13] Valenti, S., Rossi, D., Meo, M., Mellia, M. and Bermolen, P.: Accurate, fine-grained classification of P2P-TV applications by simply counting packets, *Proc. TMA'09*, pp.84–92 (2009).

[14] Qualcomm: Managing Background Data Traffic in Mobile Devices (White paper) (2012).

[15] Fernández-Delgado, M., Cernadas, E., Barro, S. and Amorim, D.: Do We Need Hundreds of Classifiers to Solve Real World Classification Problems?, *Journal of Machine Learning Research*, Vol.15, pp.3133–3181 (2014).

[16] Breiman, L.: Random Forests, *European Journal of Mathematics*, Vol.45, pp.5–32 (2001).

[17] Choi, Y., Chung, J.Y., Park, B. and Hong, J.W.K.: Automated classifier generation for application-level mobile traffic identification, *Proc. NOMS'12*, pp.1075–1081 (2012).

[18] Ham, H.-S. and Choi, M.-J.: Applicaion-level traffic analysis of smartphone users using embedded agents, *Proc. APNOMS'12*, pp.1–4 (2012).

[19] Han, X., Zhou, Y., Huang, L., Han, L., Hu, J. and Shi, J.: Maximum entropy based IP-traffic classification in mobile communication networks, *Proc. IEEE WCNC'12*, pp.2140–2145 (2012).

[20] Miskovic, S., Lee, G.M., Liao, Y. and Baldi, M.: AppPrint: Automatic Fingerprinting of Mobile Applications in Network Traffic, *Proc. PAM'15*, pp.57–69 (2015).

[21] Stöber, T., Frank, M., Schmitt, J. and Martinovic, I.: Who do you sync you are?: Smartphone fingerprinting via application behaviour, *Proc. WiSec'13*, pp.7–12 (2013).

[22] Fukumoto, N., Ano, S. and Goto, S.: Passive Smart Phone Identification and Tracking with Application Set Fingerprints, *Proc. Asia-Pacific Advanced Network*, pp.41–48 (2013).

[23] Qian, F., Wang, Z., Gerber, A., Mao, Z., Sen, S. and Spatscheck, O.: Profiling resource usage for mobile applications, *Proc. MobiSys'11*, pp.321–334 (2011).

Sophon Mongkolluksamee is a Ph.D. candidate in the Faculty of Information and Communication Technology, Mahidol University. He received his master's degree in computer science at Mahidol University, Thailand, in 2007. He was an assistant researcher at NECTEC from 2008 to 2012. Also, he was an internship student at NII, Japan in 2011. His research interests are Internet traffic analysis and network security.

Vasaka Visoottiviseth is an assistant professor at Mahidol University, Thailand. She received her Ph.D. degree in computer engineering from Nara Institute of Science and Technology in 2003, M.E. and B.E. degrees from Tokyo University of Agriculture and Technology in 1999 and 1997, respectively. Her current research interests are mobile and wireless computing, Internet traffic measurement and classification, and network security.

Kensuke Fukuda is an associate professor at the National Institute of Informatics (NII). He earned a Ph.D. in computer science from Keio University in 1999. He worked in NTT laboratories from 1999 to 2005, and joined NII in 2006. He was a visiting scholar at Boston University in 2002 and a visiting scholar at the University of Southern California/Information Sciences Institute in 2014–2015. He was also a researcher of PRESTO JST (Sakigake) in 2008–2012. His current research interests are Internet traffic measurement and analysis, intelligent network control architectures, and the scientific aspects of networks.