Service Usage Classification with Encrypted Internet Traffic in Mobile Messaging Apps

Yanjie Fu, Hui Xiong, *Senior Member, IEEE*, Xinjiang Lu, Jin Yang, and Can Chen

Abstract—The rapid adoption of mobile messaging Apps has enabled us to collect massive amount of encrypted Internet traffic of mobile messaging. The classification of this traffic into different types of in-App service usages can help for intelligent network management, such as managing network bandwidth budget and providing quality of services. Traditional approaches for classification of Internet traffic rely on packet inspection, such as parsing HTTP headers. However, messaging Apps are increasingly using secure protocols, such as HTTPS and SSL, to transmit data. This imposes significant challenges on the performances of service usage classification by packet inspection. To this end, in this paper, we investigate how to exploit encrypted Internet traffic for classifying in-App usages. Specifically, we develop a system, named CUMMA, for classifying service usages of mobile messaging Apps by jointly modeling user behavioral patterns, network traffic characteristics, and temporal dependencies. Along this line, we first segment Internet traffic from *traffic-flows* into *sessions* with a number of *dialogs* in a hierarchical way. Also, we extract the discriminative features of traffic data from two perspectives: (i) packet length and (ii) time delay. Next, we learn a service usage predictor to classify these segmented *dialogs* into single-type usages or outliers. In addition, we design a clustering Hidden Markov Model (HMM) based method to detect mixed *dialogs* from outliers and decompose mixed *dialogs* into sub-*dialogs* of single-type usage. Indeed, CUMMA enables mobile analysts to identify service usages and analyze end-user in-App behaviors even for encrypted Internet traffic. Finally, the extensive experiments on real-world messaging data demonstrate the effectiveness and efficiency of the proposed method for service usage classification.

Index Terms—In-app analytics, service usage classification, encrypted internet traffic, mobile messaging app

1 Introduction

Recent years have witnessed the increased popularity of mobile messaging Apps, such as WeChat [1] and WhatsApp [2]. Indeed, messaging Apps have become the hubs for most activities of mobile users. For example, messaging Apps help people text each another, share photos, chat, and engage in commercial activities such as paying bills, booking tickets and shopping. Mobile companies monetize their services in messaging Apps. Therefore, service usage analytics in messaging Apps becomes critical for business, because it can help understand in-App behaviors of end users, and thus enables a variety of applications. For instance, it provides in-depth insights into end users and App performances, enhances user experiences, and increases engagement, conversions and monetization. However, a key task of in-App usage analytics is to classify Internet traffic of messaging Apps into different usage types as shown in Table 1.

Traditional methods for traffic classification rely on packet inspection by analyzing the TCP or UDP port numbers of an IP packet or reconstructing protocol signatures in its payload [3], [4], [5]. For example, an IP packet usually has five tuples of protocol types, source address, source port, destination address and destination port. People estimate the usage types of traffic by assuming that messaging Apps consistently transmit data using the same port numbers which are visible in the TCP and UDP headers. However, there are emerging challenges for inspecting IP packet content. For example, messaging Apps are increasingly using unpredictable port numbers. Also, customers may encrypt the content of packets. In addition, governments have imposed privacy regulations which limit the ability of third parties to lawfully inspect packet contents. Moreover, many mobile apps use the Secure Sockets Layer (SSL) and its successor Transport Layer Security (TLS) as a building block for encrypted communications.

To address these challenges, in this paper, we aim at developing data mining solutions for classifying encrypted Internet traffic data generated by messaging Apps into different service usage types. Fig. 1 shows a motivating example about how do we classify WhatsApp and WeChat Internet traffic and identify the corresponding usage types. Note that the traffic patterns of these selected usages in WhatsApp are similar to those in WeChat. Indeed, the network traffic data of mobile messaging encode the unique patterns of both user behaviors and in-App usages. If properly analyzed, these patterns could be a source of rich intelligence for classifying service usages. Furthermore, from the security and privacy perspective, the underlying issue we leverage is that current privacy protection technology conceal the content of a packet, while they do not prevent the detection of networks packets patterns that instead may reveal some sensitive information about the user's preference [6], [7] and behavior [8].

Specifically, we study these patterns from three perspectives: (1) behavioral structure, (2) flow characteristics, and (3) temporal dependencies. First, service usage behaviors in messaging Apps have their unique hierarchical structure.

- *Y. Fu, H. Xiong and C. Chen are with the Management Science and Information Systems Department, Rutgers University, NJ 07102, USA. E-mail: {yanjie.fu, hxiong, cc1063}@rutgers.edu.*
- *X. Lu is with the School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China. E-mail: xjlu@mail.nwpu.edu.cn.*
- *J. Yang is with the Futurewei Tech. Inc, USA. E-mail: jin.yang@huawei.com.*

Manuscript received 10 June 2015; revised 19 Nov. 2015; accepted 23 Dec. 2015. Date of publication 8 Jan. 2016; date of current version 28 Sept. 2016. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TMC.2016.2516020

TABLE 1
Classic Usage Types in Messaging Apps

#	Usage Type
1	text
2	picture
3	audio note
4	stream video call
5	location sharing
6	short video
7	news feed
8	outlier or unknown mixture of usages

We employ the term of *traffic-flow* (Definition 1) to denote the encrypted network traffic (with only time stamp and packet length information being available) generated by mobile messaging Apps, and the terms of *session* (Definition 2) and *dialog* (Definition 3) to represent the segments of *traffic-flow* in different granularity. For example, in web browsing, a *session* is initiated when a user opens the browser, and torn down after the browser being closed. A *session* usually includes multiple *dialogs*, each of which starts from a new tab being opened and lasts until this tab is closed. In one *dialog*, some users may view only one web page while others may view multiple web pages. This example shows a behavioral hierarchy. In other words, a sequence of activities can be divided into multiple *sessions*, and each *session* can be divided into multiple *dialogs*. Similarly, in mobile messaging Apps, a *session* generally starts when a user opens the App and lasts until this user closes it. Each *session* consists of multiple *dialogs*. Most *dialogs* are of single-type usage, such as text, location sharing, voice, or stream video, while other *dialogs* are of mixed usages.

Second, a *traffic-flow* of M observations (packets in this study) usually contains two sequences: (1) an M-size sequence of packet lengths representing the data transmission of service usages and (2) an (M-1)-size sequence of time delays representing the time intervals of consecutive packet pairs. In terms of the packet length, as shown in Fig. 1, different service usages have different global characteristics (e.g., distribution properties such as mean and variance of packet lengths, etc.) and local characteristics (e.g., packet-level features such as forward or backward variances at important observation positions, etc.). For example, texts are more frequently used, shorter in time, and smaller in data size comparing to stream video call, therefore any traffic intervals with flow rate lower than certain thresholds are likely determined as text streams. Aside from global characteristics, local (i.e., packet-level) characteristics are from the fact that the packet lengths of different usage types vary over observation positions in the sequence of packet lengths. For example, Fig. 2a shows the process that a mobile user sends out a text message, and thus generates a pulse in traffic, followed by another pulse representing a text reply. Also, Fig. 2b shows that, in stream video call, most packets are fully loaded (i.e., close to 1,500 bytes) in the sequence of packet lengths. In terms of time delay, different in-App usages adopt different design logics and control flows for function implementation and different network protocols for packet transmission, and thus show unique characteristics of time delay distribution. For example, Fig. 2h shows that, for location sharing, most of packets are sent in the initial phase. However, for short video, data transmission is completed in the end as shown in Fig. 2g.

Third, sequential usages in mobile messaging Apps have not only network traffic characteristics, but also inherent temporal dependencies. For example, stream video call or picture sharing generally starts with a few texts. And each mobile user has his/her personalized patterns of usage transition in mobile messaging Apps. When the traffic flows are short and the defined features are not enough to fully describe the traffic features for classification, we can exploit HMM to capture the temporal dependencies (i.e., usage transition patterns) for enhancing classification accuracy.

Along this line, in this paper, we propose a method to classify in-App service usages using encrypted Internet traffic data by jointly modeling behavioral structure, flow characteristics and temporal dependencies. Specifically, we first segment Internet traffic from *traffic-flow* to *sessions* to *dialogs* by combining hierarchical clustering as well as thresholding heuristics. Besides, we extract the discriminative features of these segmented *dialogs* from two perspectives: (1) packet length and (2) time delay. In addition, we learn a service usage predictor by feeding the extracted features and the reported usage types into the chosen classifiers. Moreover, for those outlier *dialogs* with mixed usages, we exploit a clustering method to further segment these *dialogs* into sub-*dialogs*. Then, we utilize a trained HMM model for disaggregating mixed usage types. Furthermore, we develop a system, named CUMMA, for classifying service usages in mobile messaging Apps using the proposed method. It has been incorporated into the SmartCare service of a company

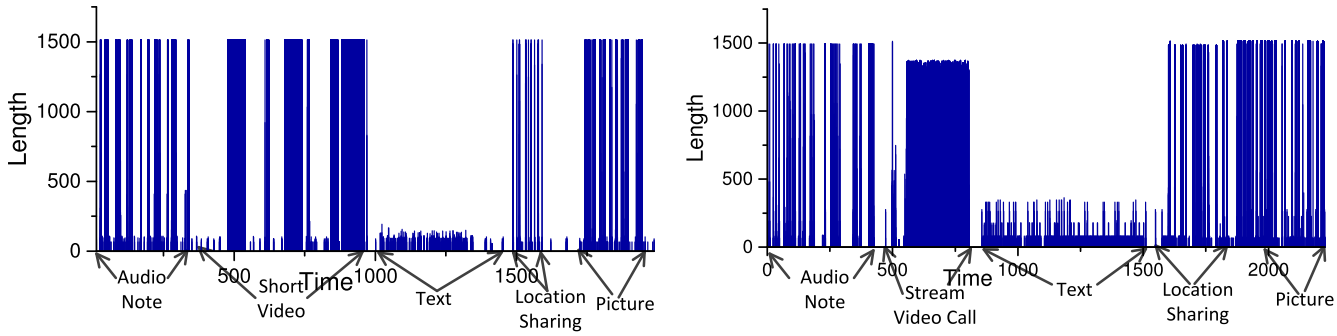


Fig. 1. Example traffic flows of consequent service usages of WhatsApp and WeChat.

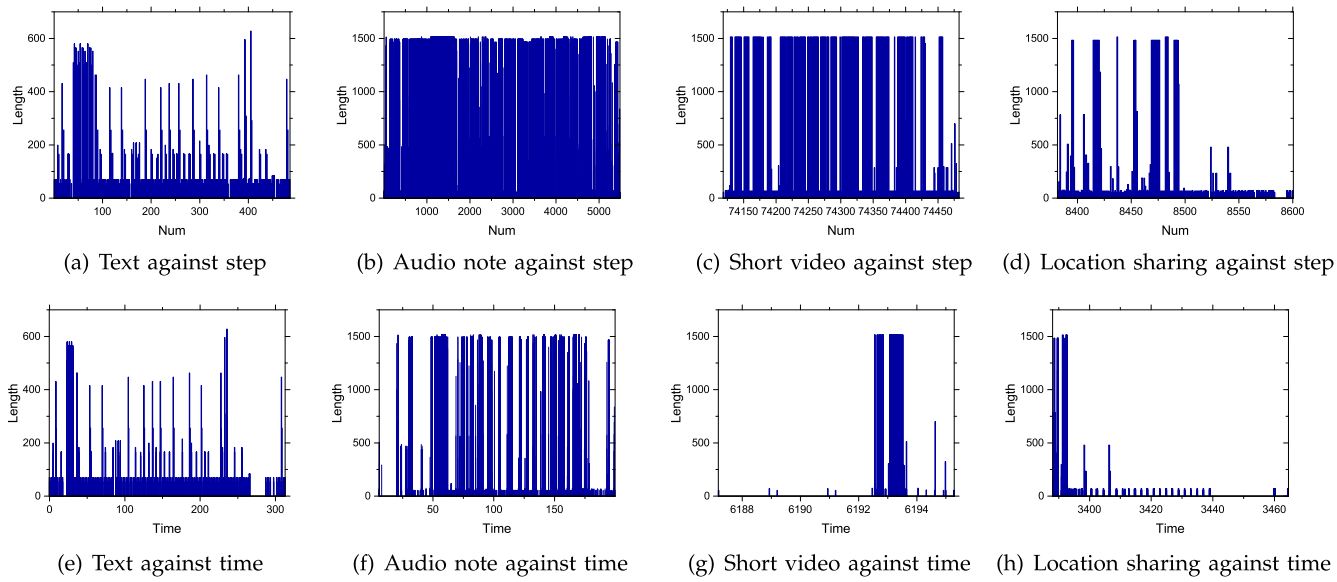


Fig. 2. Packet length sequences of different usage types with respect to observation positions (i.e., step) and observation times (i.e., time).

for the purpose of enhancing end-user experiences. Also, we illustrate a real world business application for scoring quality of experiences with CUMMA. Finally, we conduct comprehensive performance evaluation on real world traffic data of WeChat and WhatsApp, and the experimental results demonstrate the effectiveness (96.1 percent accuracy for WeChat and 97.5 percent accuracy for WhatsApp) and efficiency of our method.

2 Preliminaries

Before delving into the details, we first introduce the definition of *traffic-flow*, which denotes the encrypted network traffic generated by mobile messaging App.

Definition 1 (Traffic-Flow). *A traffic-flow* F *generated by mobile messaging App contains a time range* $T_F = [t^F_{start}, t^F_{end}]$, *and a sequence of packets* $seq = \{p_1, p_2, \ldots, p_M\}$ *where* M *is the number of observations. For each element in* seq, *only the packet length is available. Therefore, we have* $F = (\{t_m, l_m\})_{m=1}^{M}$, *where* l_m *represents the length of the* m*th packet at the* m*th time stamp* t_m, *and* $t^F_{start} = t_1, t^F_{end} = t_M$.

In the following, we present the problem statement, and then provide an overview of the system architecture.

2.1 Problem Statement

The sequential usages in mobile messaging Apps can generate large amount of encrypted Internet traffic data. Service usage classification of mobile messaging traffic typically segments and classifies *traffic-flows* into a sequence of usage activities, each of which is described by a triple consisting of a start time, an end time, and a usage type. In this study, encrypted Internet traffic refers to *traffic-flows* whose packets only have time stamps and packet lengths available.

Formally, given a *traffic-flow*, the proposed system should return a sequence of in-App usage activities, $(\{b_n, e_n, \pm \boldsymbol{u}_n\})_{n=1}^{N}$, where b_n, e_n, and $\pm \boldsymbol{u}_n$ refer to the begin time, the end time, and the single usage or the mixed usages of the nth usage activity. Essentially, there are two major tasks: (1) identifying the time intervals of sequential usage activities and (2) inferring the types of sequential messaging usages.

2.2 The Overview of System Framework

Fig. 3 presents the framework of our CUMMA system for classifying service usages using encrypted Internet traffic data. There are four modules in the system framework: (1) traffic segmentation, (2) traffic feature extraction, (3) usage type prediction, and (4) outlier detection and handling.

2.2.1 Traffic Segmentation

We first collect the network traffic data of different usages in messaging Apps using the data collection platform introduced in Fig. 5. After collecting these benchmark data, we perform a two-stage segmentation with these *traffic-flows* from coarse-grained level named *session* to fine-grained level named *dialog*. Specifically, we first define two types of granularity of traffic segmentation: (1) *session* and (2) *dialog*.

Definition 2 (Session). *A* session s *is a subset of* traffic-flow F *(i.e.,* s $\subseteq$ F*). In addition,* s *contains a time range* $T_s = [t^s_{start}, t^s_{end}]$, *and* I_s *adjacent packets* $seq^s = \{p_{m_{i_1}}, p_{m_{i_2}}, \ldots, p_{m_{I_s}}\}$ *(*$seq^s \subseteq seq$*) which satisfies* $t^s_{start} = t_{m_{i_1}}, t^s_{end} = t_{m_{I_s}}$ *and there is no other session* s^* *that makes* $T_s \subseteq T_{s^*}$. *Consequently, we*

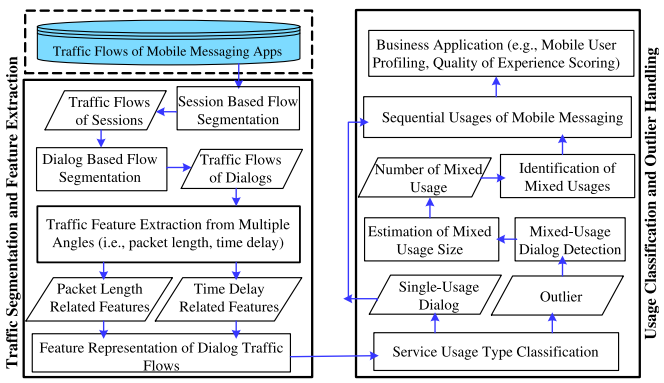


Fig. 3. The framework of in-App service usage classification with encrypted Internet traffic.

have $s = (\{t_{m_{i_s}}, l_{m_{i_s}}\})_{i_s=i_1}^{I_s}$, *where* $l_{m_{i_s}}$ *is the length of packet* $p_{m_{i_s}}$ *at the time stamp* $t_{m_{i_s}}$. *Moreover,* $\forall i_s \in \{i_1, i_2, \ldots, I_s - 1\}$, *we have* $(t_{m_{i_s+1}} - t_{m_{i_s}}) < \delta$, *and* $\forall i_s \in \{i_1, i_2, \ldots, I_s\}$, $l_{m_{i_s}} > \phi$, *where* δ *and* ϕ *are predefined time threshold and packet length threshold for segmenting* sessions *from* traffic-flow.

Note that, $t^F_{start} \leq t^s_{start}, t^F_{end} \geq t^s_{end}$. With segmenting from *traffic-flow* to *sessions*, we have $S = \{s_1, s_2, \ldots, s_I\} = (s_i)_{i=1}^{I}$ and $S \subseteq F$, where s_i is the ith *session* and I is the number of *sessions*. In the context of in-App usages, a *session* generally starts when a user open the App and lasts until a user close it. In a *session*, a user can try different usages. For example, one opens his/her WhatsApp and starts texting his/her friend at 5 PM, then has video chat with his/her mother from 5:25 PM to 5:45 PM, finally focuses on reading news feeds until 6 PM. Consequently, the generated traffic data from 5 PM to 6 PM is a *session*.

Definition 3 (Dialog). *A* dialog d *is a subset of* session s, *which contains a time range* $T_d = [t^d_{start}, t^d_{end}]$, J_d *adjacent packets* $seq^d = \{p_{i_{j_1}}, p_{i_{j_2}}, \ldots, p_{i_{J_d}}\}$ $(seq^d \subseteq seq^s)$, *and corresponding usage activity of mobile messaging App. Moreover, we have* $d = (\{t_{i_{j_d}}, l_{i_{j_d}}\})_{j_d=j_1}^{J_d}$ *where* $l_{i_{j_d}}$ *is the length of packet* $p_{i_{j_d}}$ *at time stamp* $t_{i_{j_d}}$, *and* $t^d_{start} = t_{i_{j_1}}, t^d_{end} = t_{i_{J_d}}$.

Here, we note that $t^s_{start} \leq t^d_{start}, t^s_{end} \geq t^d_{end}$. After segmenting from *session* to *dialogs*, we have $D = \{d_1, d_2, \ldots, d_J\} = (d_j)_{j=1}^{J}$ and $D \subseteq S$, where d_j is the jth *dialog* and J is the number of *dialogs*. In the above example for Definition 2, the generated Internet traffic of video call from 5:25 PM to 5:45 PM can be treated as a *dialog*. In most cases, a *dialog* corresponds to a single-type usage, such as text, location sharing, voice, or stream video, while other *dialogs* are of mixed usages.

Later, we perform a hierarchical segmentation based on the definitions of *session* and *dialog*: (1) we first segment each *traffic-flow* into *sessions* using a thresholding method; (2) then we segment each *session* into *dialogs* by a bottom-up hierarchical clustering based method mixed with thresholding heuristics. The above method can segment the raw Internet traffic into *dialogs*. Note that most *dialogs* contain single usage type, and only a few are the mixture of multiple usage types.

2.2.2 Traffic Feature Extraction and Usage Type Prediction

Given a set of *dialogs*, our objective is to identify their usage types. To this end, we first mine the discriminative features of the network traffic data from two perspectives: (1) packet length and (2) time delay. We then feed these segmented *dialogs*, each of which with a traffic feature vector and a reported usage type, into a robust classifier (i.e., random forest) for training and use the trained classifiers to predict the usage types of new traffic data.

2.2.3 Outlier Detection and Handling

Although the trained classifier can predict most *dialogs*, it is possible that some *dialogs* are classified as outliers. It thus is necessary to devise an approach for detecting and handling outliers as shown in the right part of Fig. 3. Specifically, we first categorize these outlier *dialogs* into two categories: (1) true outliers and (2) *dialogs* with mixed usage types. While the *dialogs* of true outliers are generally ignored, we exploit a clustering based method to segment mixed *dialogs* into multiple consecutive sub-*dialogs* of single-type usage. Since these sub-*dialogs* are short and only feature extraction method cannot fully describe network traffic data, we propose to explore the temporal dependencies between consecutive user-generated packets, and utilize a trained HMM model to predict the usage types of these sub-*dialogs*.

3 Usage Classification of Mobile Messaging Traffic

In this section, we detail the four major modules of the proposed system for classifying service usages in mobile messaging App.

3.1 Traffic Segmentation

In this section, we develop a two-stage algorithm: (1) from *traffic-flow* to *session* and (2) from *session* to *dialog*, to perform traffic segmentation.

From *Traffic-Flow* **to** *Session*. Given a *traffic-flow* $F = (\{t_m, l_m\})_{m=1}^{M}$, we aim to segment the sequence of observations into multiple *sessions*. Specifically, we first collect the background traffic on the condition that there is no service usage activities in the targeted App. As can be seen from Fig. 1b, background traffic are those packets whose lengths are extremely small, typically ranging from 50 to 70. Bases on this pattern, we remove the background traffic from the original traffic. Later, we further filter out all zero-traffic intervals longer than certain threshold, and obtain multiple *sessions* $S = (s_i)_{i=1}^{I}$.

From *Session* **to** *Dialog*. Since each *session* may contain multiple consecutive *dialogs*, duration based thresholding method is not enough for *dialog* level segmentation. We therefore adopt the idea of hierarchical clustering and devise a bottom-up based segmentation algorithm. For each *session* $s_i \in S$, let $V(d)$ denote the traffic volume (i.e., the sum of all packet lengths) of *Dialog* d.

- Initialization. We first divide the time duration of the *session* s_i into multiple small intervals (e.g., 1.5 second per interval). We then filter out zero-traffic intervals and treat all the non-zero intervals as a sequence of initial *dialogs* $D = (d_j)_{j=1}^{J}$.
- Merging heavy-heavy *dialog* pairs. We define a traffic volume threshold α. For each continuous *dialog* pair (d_j, d_{j+1}), if $V(d_j) > \alpha$ and $V(d_{j+1}) > \alpha$, then merge d_j and d_{j+1}. Repeat this until no such pair exists.
- Merging light-heavy *dialog* pairs. For each continuous *dialog* triple (d_{j-1}, d_j, d_{j+1}) where $V(d_j) < \alpha$, if $V(d_{j-1}) > \alpha$ and $V(d_{j-1}) > V(d_{j+1})$, then merge d_{j-1} and d_j; if $V(d_{j+1}) > \alpha$ and $V(d_{j+1}) > V(d_{j-1})$, then merge d_j and d_{j+1}.
- Merging peak *dialog* pairs. For each continuous *dialog* pair (d_j, d_{j+1}), merge d_j and d_{j+1},if $V(d_j) > \beta$, $V(d_{j+1}) > \beta$, and $dist(d_j, d_{j+1}) < \gamma$ where β refers to a traffic volume threshold of peak *dialogs* and $dist(d_j, d_{j+1})$ refers to the time difference between the start time of d_j and the start time of d_{j+1}.

By the above three merging operations, we try to keep adjacent intervals in one *dialog* if they possibly correlate to one usage type or mixed usage types. This is to avoid the situation where the network traffic of one usage type is divided into two *dialogs* of same usage types, which is unrecoverable in our approach. If two nonparallel activities are mistakenly grouped into one *dialog*, they can still be identified in the following module called outlier detection and handling (refer to Section 3.4). This module outputs a sequence of *dialogs* $D = (d_j)_{j=1}^{J}$ where each *dialog* contains a single usage or mixed usages.

3.2 Traffic Feature Extraction

An Internet traffic of a *dialog* encodes two kinds of information: (1) the sequence of packet lengths and (2) the sequence of time delays. Here, we introduce the discriminative features we have extracted from two perspectives: (1) packet length and (2) time delay.

3.2.1 Packet Length Related Features

Generally, the sequence of packet lengths exhibits the patterns of data flow, and thus can reflect different usage types.

- *Overall descriptive statistics:* It is vital to exploit overall descriptive statistics, as they can describe the basic properties of packet length distribution from multiple aspects. Given a sequence of packet lengths, we extract the first order and the second order descriptive statistics (i.e., standard deviation, median, minimum, skewness, kurtosis, and standard error of packet length) as features.
- *Variances in backward and forward directions:* The variance of packet sizes is a signature of in-App behaviors. Although some sequences may have low variation, this feature set can capture the fine-grained variances in terms of two different directions at a specific quantile. This fine-gained measurements can help discern in-App behaviors. Given a sequence of packet lengths, we first select three representative observation positions (i.e., first quartile, second quartile and third quartile packet indexes) from the sequence. Then, for each selected position, we split the sequence into two subsequences with respect to backward and forward directions. Finally, we calculate the variances of these subsequences (i.e., 3*2=6 subsequences in the experiment) as features, Here, the variance of packet length is given by $\mathrm{Var} = \frac{\sum_{x \in X}(x-\bar{x})^2}{n-1}$, where X is a packet length sequence and $n = |X|$.
- *Percentages of packets with length in K equal-sized ranges:* Different in-App behaviors result in different ranges of packet lengths. For example, the packet length of video stream is larger the packet length of text messages. This feature set discretizes the distribution of packet lengths into several equal-sized ranges. In this way, we reduce the noise led by the small fluctuations in packet lengths. Given a sequence of packet lengths, we first identify the minimum and maximum values of IP packet lengths. We then split the range from minimum to maximum into K equal-sized subranges. For each subrange, we calculate the number and the percentage of packets in this subrange. Finally, we obtain a K-size feature vector, each of which represents the percentage of packets with length in the kth subrange.
- *Hopping counts:* This feature set captures the file-grained stability of traffic flow. For example, unlike stream video or voice call, text messages contains many length packets with randomized lengths, where hopping counts can effectively describe the "pulse" of packets. We count the number of packets whose lengths are greater than the lengths of their next packets with a significant margin. We take this number as a feature to characterize the fluctuation of the sequence.
- *Lengths of longest monotone subsequences:* This feature set describes the sizes of the gradient descent or ascent patterns in a sequence, and the transition patterns from gradient descent to gradient ascent. We examine the longest monotone (including increasing and decreasing) subsequences of a *dialog*, and use the lengths of these two subsequences to describe the tendency and skewness of the network traffic.
- *top-N frequent fixed-size continuous subsequences and the corresponding frequencies:* By mapping packet length ranges into letters, we can regard a traffic flow as a sequence of letters. This feature set illustrates the frequent "letter" pattern in traffic flows, generated by in-App protocols, which indirectly shows the data proceeding logics of App designer. Similarly, we first identify the range of packet lengths, and split this range into K equal-sized subranges. For each element in the sequence of packet lengths, we replace this element with k if its value is in the kth subrange. In this way, we map this sequence into a new string sequence. With this string sequence, we identify all the continuous subsequences with size ranging from 3 to 20 and calculate their corresponding number of occurrences in this string sequence. Finally, we use the top-N numbers of occurrences as features ($N = 5$ in the experiment).

3.2.2 Time Delay Related Features.

In addition, we can describe a *dialog* from the perspective of time delay. Specifically, we extract the time interval for every two consecutive packets, and thereby obtain a sequence of time delays. Indeed, the sequence of time delays characterizes the patterns from two aspects, i.e., implementation or design logics of in-App usages and protocols of data transmission, which can help discriminate usage types as well. Similarly, we can adopt the definitions of packet length related features (i.e., overall descriptive statistics) and extract the features from the sequence of time delays.

3.3 Usage Type Prediction

The feature extraction module outputs a set of vectorized *dialogs* $D = (\boldsymbol{d}_j)_{j=1}^{J}$ where each is represented by a feature vector. To predict usage types, we exploit the classification based methods. Generally speaking, a *dialog* is a long time series of Internet traffic. Thus, a well-defined feature set is

enough to capture the characteristics of different usage types. For simplicity, we neglect the temporal dependencies between consecutive *dialogs* and treat these segmented *dialogs* as a set of independent training instances. Later, all the features are mapped and scaled to real type values in order to fit a classifier. Here, we exploit the ensemble power of Random Forest. Specifically, we apply the general technique of bagging, to tree learners. Let B denote the number of trees, given a set of labeled *dialogs* $D = (\boldsymbol{d}_j)_{j=1}^{J}$ with usage types $U = (u_j)_{j=1}^{J}$ where $u_j \in 1, 2, \ldots, K$ (since we have defined seven usage types, $K = 7$ in this study), we repeatedly select a random sample with replacement of the labeled *dialogs*, and respectively fit B decision trees to the B samples. After training, predictions for unknown *dialog* $d^{'}$ can be made by averaging the predictions from all the individual decision trees on the *dialog* $d^{'}$. In particular, random forest gives the probability estimation for each class k as follows

$$P(k|d^{'}) = \frac{1}{B}\sum_{b=1}^{B} P_b(k|d^{'}),$$

where $P_b(k|d^{'})$ is the probability estimation of usage type k given by the bth tree. It is estimated by computing the ratio that usage type k gets votes from the leaves of the bth tree. The overall decision function of random forest is defined as

$$u^{'} = argmax_k P(k|d^{'}).$$

Finally, each *dialog* will be labeled as a single usage type or outlier/unknown mixture of usages.

3.4 Outlier Detection and Handling

We propose a method to handle those *dialogs* classified as outliers or unknown mixture of usages. Indeed, an anomalous *dialog* could be generated by true outliers such as network errors, but it is difficult to differentiate these true outliers only based on packet length and time delay related features. Since we have removed the background noise from the original *traffic-flows* in the phase of traffic segmentation, we assume that these anomalous *dialogs* are generated by mixture of usages. Therefore, we only consider the case of mixed usages.

Given a *dialog* $d_j \in D$ that is classified as a mixture of usages, the objective is to identify the most probable hidden usage mixture $(\{u_{jh}, d_{jh}\})_{h=1}^{H}$. We exploit a Clustering-HMM method to disaggregate a usage mixture consisting of multiple single-type sub-*dialogs*. In the phase of usage type prediction, we have mined many *dialogs* of single usage type, such as text, stream video call, news feed, etc. For simplicity, we only consider a single feature (i.e., packet length) and extract the mean value of packet length of each usage type from the identified single-type *dialogs*. We utilize a clustering method (i.e., KMeans), then set up K centers with these mean packet lengths as prior knowledge, and segment each mixed-usage *dialog* into multiple traffic segmentations where each represents a single-usage sub-*dialog*. Later, we remove the anomalous sub-*dialogs* that have too few packets or are too short in time, so that we segment each mixed *dialog* into multiple sub-*dialogs* where each is assumed to be generated by a single-type usage.

Then, we perform the aforementioned feature extraction and profile each sub-*dialogs* into a feature vector. However, these segmented sub-*dialogs* are shorter than *dialogs* in time. Unlike the phase of usage type prediction, we take into account the temporal dependencies between consecutive sub-*dialogs*, because these temporal dependencies can help enhance the classification accuracy for these short time series of sub-*dialogs*. We thus utilize HMM to classify sub-*dialogs* in a mixed *dialog*. Specifically, for each service usage type $k \in 1, 2, \ldots, K$ (K = 7 in this study), we feed corresponding sequences of packet lengths into a HMM model. Therefore, we obtain seven trained HMM models. Given an sub-*dialog* with unknown usage type d_{jh}, we fit this sub-*dialog* into the seven HMM models, compute the likelihood of this sub-*dialog* for each usage type, and thus obtain seven likelihoods. Finally, we classify this sub-*dialog* as the service usage type of the largest likelihood. And, we use the previous classified *dialogs* to train the HMM model and exploit this trained HMM model to predict the usage types of sub-*dialogs*.

So far, we have performed traffic segmentation, traffic feature extraction, usage classification and outlier handling, we are able to return a sequence of semantically-rich usage activities $(\{b_n, e_n, \pm \boldsymbol{u}_n\})_{n=1}^{N}$, each of which contains start time, end time and usage type.

4 Experimental Results

We evaluate the performance of our developed system with the real world data of Internet traffic collected from mobile messaging Apps.

4.1 Application Deployment

Table 1 shows the defined eight types of service usages in messaging Apps. To collect the Internet traffic data of these usage types, we design and deploy the data collection platform as shown in Fig. 5. In this platform, each mobile volunteer is equipped with a factory unlocked smart phone, a computer, and a packet analyzer, i.e., WireShark.[1] Specifically, we first install a fresh new Android OS on this smart phone, and remove all the Apps except the targeted messaging App (e.g., WeChat or WhatsApp). Then, we setup an Android firewall and grant access permission to the targeted messaging App only. In this way, we can ensure the collected *traffic-flows* are purely from the messaging App. Later, we establish a virtual Wi-Fi access point (AP) which is exclusively connected the smart phone only. In other words, there is only one way for this smart phone to access Internet; that is the AP. In addition, we exploit a well-known packet analyzer, WireShark, to crawl the packet information (e.g., time stamp, packet size, etc.) of messaging traffic data from this AP. Moreover, we set up packet filtering rules and distill the crawled packet information. Meanwhile, mobile volunteers will manually report their corresponding usage types as ground-truth labels. Finally, both the Internet traffic and the reported usage types are organized, archived and stored in our data repository. In this lab data collection method, we assure that only the messaging app traffic was captured. In real world data collection, we will first filter out other traffic

1. https://www.wireshark.org/

TABLE 2
The Characteristics of the WeChat Internet Traffic Data

#	Usage Type	Packets	Bytes	Avg. Traffic /min
1	text	105 K	20 M	24 K
2	picture	294 K	215 M	1,175 K
3	audio note	82 K	37 M	29 K
4	stream video call	2,222 K	734 M	2,850 K
5	location sharing	15 K	5,944 K	149 K
6	short video	70 K	50 M	823 K
7	news feed	595 K	381 M	755 K
8	outlier	1,517 K	59.7 M	194 K

TABLE 3
The Characteristics of the WhatsApp Internet Traffic Data

#	Usage Type	Packets	Bytes	Avg. Traffic /min
1	text	33.724 K	387.373 K	1.222 K
2	audio note	79.626 K	3.967 M	22.709 K
3	picture	85.761 K	6.432 M	52.120 K
4	stream voice call	209.229 K	4.127 M	33.308 K
5	location sharing	11.313 K	289.964 K	3.274 K
6	short video	385.122 K	43.295 M	328.179 K

(e.g., http, ftp, email, etc.) with traffic classification at application or protocol level, which is a thoroughly studied problem. Many communication companies have matured tools for filtering.

4.2 Data Description

We collected the network traffic data by monitoring the service usage activities of mobile volunteers in WeChat and WhatsApp. To collect the WeChat data, we hired fifteen volunteers. Each mobile volunteer was equipped with one Samsung handset (Samsung Galaxy S II) with Android operating system (version 4.0.4). The time period of data collection spanned from December 2, 2014 to December 31, 2014. The volunteers were required to perform the seven types of service usages in their daily life (i.e., morning, noon, afternoon, and night), and manually report the start time, the end time and the label of their in-App usage activities. To collect the WhatsApp data, we additionally hired five volunteers (different from the volunteers for WeChat data collection) and examined their WhatsApp daily usage in the time period from September 25, 2015 to October 18, 2015. Tables 2 and 3 respectively show the basic statistics of the WeChat dataset and the WhatsApp dataset for different types of service usages.

4.3 Evaluation Metrics

We measured the classification performance of the proposed system using the following metrics: (1) *Overall Accuracy* is the ratio of the number of True Positives to the number of True Positives and False Positives for all classes. Formally, overall accuracy is given by $\frac{TP}{TP+FP}$, where TP and FP are the number of True Positives and the number of False Positives for all classes, respectively. We use this metric to measure the accuracy of a classifier on test data for all service usage types. The latter three metrics measures the classification accuracies of each service usage type from different aspects. (2) *Precision* is the ratio of the number of True Positives to the number of True Positives and False Positives with respect to a specific type of service usage. (3) *Recall* is the ratio of the number of True Positives to the number of True Positives and False Negatives with respect to a specific type of service usage. (4) *F-measure* considers both precision and recall in a single metric by taking their harmonic mean. Formally, F-measure is given by $2 \times P \times R/(P+R)$, where P and R are precision and recall, respectively. Besides, to further illustrate the effectiveness of CUMMA for measuring in-App Quality of Experience (QoE, which will be detailed in Section 4.10), we consider the following metrics for ranking performance evaluation.

(1) *NDCG* is the normalized *discounted cumulative gain* (Normalized DCG@K), and DCG@K is given by

$$DCG[K] = \begin{cases} rel_1 & if\ K = 1 \\ DCG[K-1] + \frac{rel_K}{log_2 K}, & if\ K >= 2. \end{cases}$$

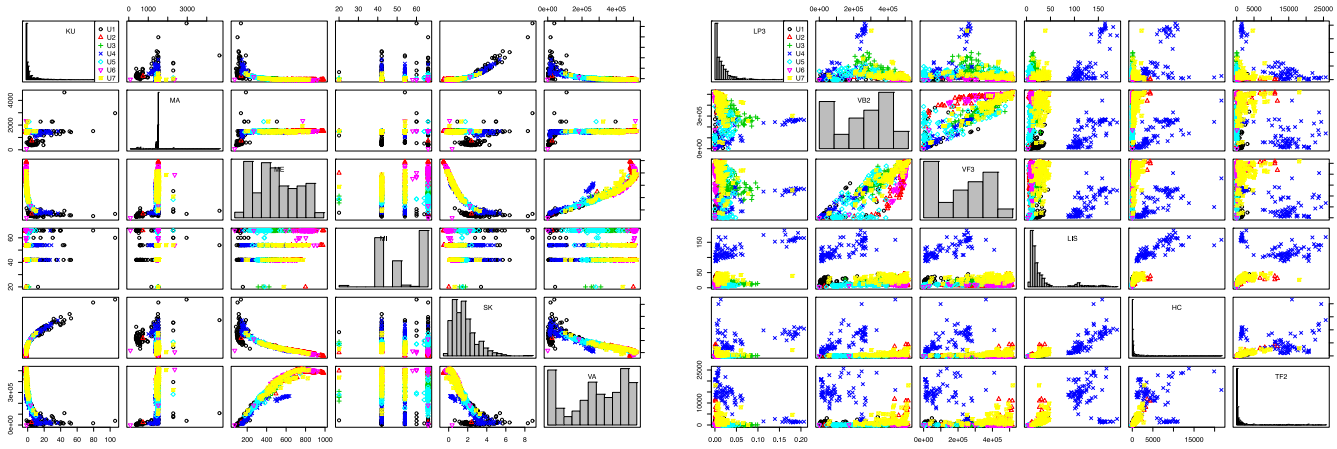
Given the ideal discounted cumulative gain $DCG^{'}$, NDCG at the Kth position can be computed as NDCG[K]= $\frac{DCG[K]}{DCG^{'}[K]}$, where rel_K is the ground-true rating of QoE of the Kth usage activity in the predicted ranking list. The larger NDCG@K is, the higher top-K ranking accuracy is.

(2) *Precision@K and Recall@K*. For measuring QoE, we use a three-level rating system ($3 > 2 > 1$) instead of binary rating, we treat the rating $= 3$ as "high-quality" and the rating < 3 as "low-quality". Given a top-K service usage activities $\mathcal{L}_K$ sorted in a descending order of the predicted QoE scores, the precision and recall are defined as $\mathrm{Precision@}K = \frac{|\mathcal{L}_K \bigcap \mathcal{L}_{=3}|}{K}$ and $\mathrm{Recall@}K = \frac{|\mathcal{L}_K \bigcap \mathcal{L}_{=3}|}{|\mathcal{L}_{=3}|}$, where $\mathcal{L}_{=3}$ are the usage activities whose ratings are equal to three.

(3) *Kendall's Tau Coefficient*. Kendall's Tau Coefficient (or Tau for short) measures the overall ranking accuracy. Let us assume that each usage activity i is associated with a benchmark score y_i and a predicted score f_i. Then, an activity pair $\langle i, j \rangle$ is said to be concordant, if both $y_i > y_j$ and $f_i > f_j$ or if both $y_i < y_j$ and $f_i < f_j$. Conversely, $\langle i, j \rangle$ is said to be discordant, if both $y_i < y_j$ and $f_i > f_j$ or if both $y_i < y_j$ and $f_i > f_j$. Tau is given by $\mathrm{Tau} = \frac{\#_{conc} - \#_{disc}}{\#_{conc} + \#_{disc}}$.

4.4 Baseline Methods and Comparison Settings

In the literature of general traffic classification (i.e., not limited to in-App traffic), researchers typically use (1) *features of descriptive statistics* stand for the standard deviation, median, minimum, maximum, skewness, and kurtosis of the sequence of packet lengths. Therefore, descriptive statistics can be treated as a baseline feature set for comparison. Aside from descriptive features, we further categorized the designed features into five categories as proposed feature sets: (2) *features of length* represent the percentages of packets with length in K equal-sized ranges (K=4 in our experiment). (3) *features of fluctuation* refer to the variances in forward and backward directions at first-quantile, second-quantile, and third-quantile positions in the sequence of packet lengths, as well as hopping counts (in our experiment hopping threshold of packet length is 300 bytes). (4) *features of frequent*



(a) Descriptive statistics

(b) Other selected features such as fluctuation, length, trend, etc.

Fig. 4. An example correlation analysis of different selected features on the WeChat dataset.[2]

patterns represent the lengths of the longest monotone subsequences, and the numbers of top-N frequent fixed-size continuous subsequences (in our experiment N is 3); (5) *features of time delay* represent the descriptive statistics of the sequence of time delays; (6) *combination* represent all the extracted features, which indeed is our proposed method.

To illustrate the predictive power of our proposed analytic framework, we use four different classifiers (i.e., Gradient Boosted Trees, Support Vector Machine, Naive Bayes, KNeighbors) in the phase of usage type prediction (refer to Section 3.3) as baselines for comparison. Note that, in Section 3.3, we propose to exploit the ensemble power of Random Forest, and thus we name our proposed method as (1) *Random Forest - Kmeans - HMM* **(RF)**, where "Kmeans - HMM" stand for the two methods used in the step of outlier handling (refer to Section 3.4). Here, Kmeans is used to separate mixed-usage *dialogs* into sub-*dialogs* and HMM is used to classify consecutive sub-*dialogs* (short traffic). Similarly, for the four baselines, we name the entire framework with Section 3.3 using four baseline classifiers as follows: (2) *Gradient Boosted Trees - Kmeans - HMM* **(GBT)**, (3) *Support Vector Machine - Kmeans-HMM* **(SVM)**, (4) *Naive Bayes - Kmeans - HMM* **(NB)**, and (5) *KNeighbors - Kmeans - HMM* **(KNN)**. We will compare our proposed method (RF) with four baseline methods: GBT, SVM, NB, and KNN.

4.5 Correlation Analysis

We provide a visualization analysis to validate the correlation between the extracted features and the seven usage types using the WeChat dataset. Specifically, we used the scatter-plot matrix for correlation analysis. Each non-diagonal chart in a scatter plot matrix shows the correlation between a pair of features whose feature names are listed in the corresponding diagonal charts. Given a set of N features, there are N-choose-2 pairs of features, and thus the same numbers of scatter plots. The dots represent the *dialogs* or sub-*dialogs* and their colors represent the service usage types. Fig. 4 plots that the colored points representing seven usage types are separately distributed and grouped into different clusters in the 2-dimensional subspaces of different feature pairs. Therefore, we have reason to believe that, when we combine all the extracted features together, the seven usage types can be separated and classified more accurately. The visualization results validate the correctness of our feature design for discriminating in-App usage types.

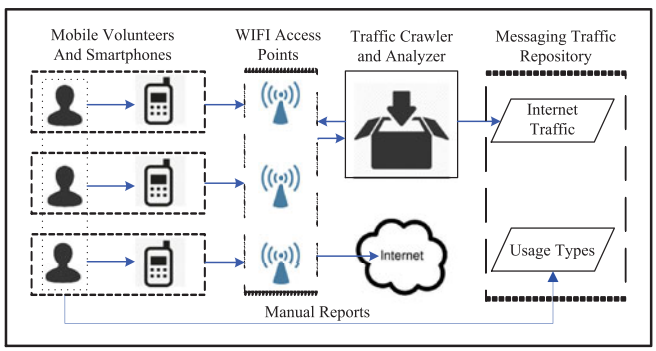
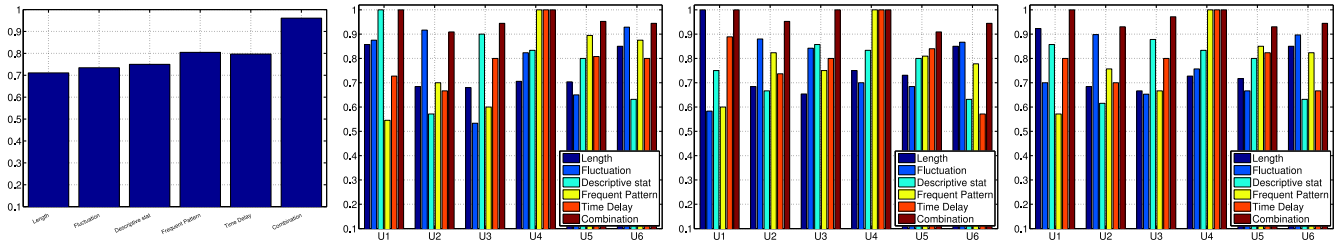


Fig. 5. The data collection platform of mobile messaging Apps.

4.6 A Comparison of Different Features

We compare the effectiveness of different features for service usage classification in our proposed system. In the experiments, we used the Random Forest-Kmeans-HMM method as the default method. For Random Forest, we set the number of trees in the forest as 100, used the gini index to measure the quality of a split, and used bootstrap samples when building trees. For K-Means model, we set the number of clusters as 7 which is also the number of service usage types, set maximum number of iterations for a single run as 300, and set relative tolerance with regards to inertia to declare convergence as $1 * e^4$. To reduce the uncertainty of splitting the data into training and test data, we randomly divided the data into 80 percent for training and 20 percent for testing. We repeated this process five times and extract the average performance for evaluation.

2. U1, U2, U3, U4, U5, U6, U7 respectively denote text, picture, audio note, stream video call, location sharing, short video, and news feed. Also, KU, MA, ME, MI, SK, VA respectively denote kurtosis, max, mean, min, skewness, variance. LP3, VB2, VF3, LIS, HC, TF2 respectively denote the percentage of packets with length in second equal-sized ranges, variance in backward direction at the second quantile position, variance in forward direction at the third quantile position, the length of longest increasing subsequence, hopping count, frequency of top-2 frequent fixed-size continuous subsequences.



(a) Overall accuracy (b) Precision of different usages (c) Recall of different usages (d) F-measure of different usages

Fig. 6. Performance comparison of different features on the WeChat data. U1, U2, U3, U4, U5, U6, and U7, respectively, denote text, picture, audio note, stream video call, location sharing, short video, and news feed.

4.6.1 Results on WeChat Data

Fig. 6 shows the performances of different feature sets (i.e., length, fluctuation, descriptive statistics, frequent pattern, time delay). First, Fig. 6a shows that the combination of all the features achieves the highest overall accuracy, which is larger than 0.96, which proves that fusing multi-source features can better classify usages than single-source features. Besides, Figs. 6b and 6c plot that the classification accuracies of individual feature sets vary on different usage types. For example, descriptive statistics of packet lengths yield better results of classifying text and audio note, while fluctuation features of packet lengths provide high accuracy on picture classification. The features of frequent pattern in packet lengths perform better in classifying stream video call and location sharing than other individual feature sets. Our further study illustrates that time delay related features can strengthen the power of the combination of all features.

4.6.2 Results on WhatsApp Data

Fig. 7 again validates the consistent effectiveness of our designed features using the WhatsApp data. For example, Fig. 7a plots the combination of our proposed features can achieve > 0.97 overall accuracy of all six usage types, while the accuracy of the baseline method (i.e., descriptive features) is only 0.83. Figs. 7b and 7c show the average performances of the combination method for six individual usage types are above other competitors. Finally, Fig. 7d demonstrates a balanced performance between precision and recall for our method .

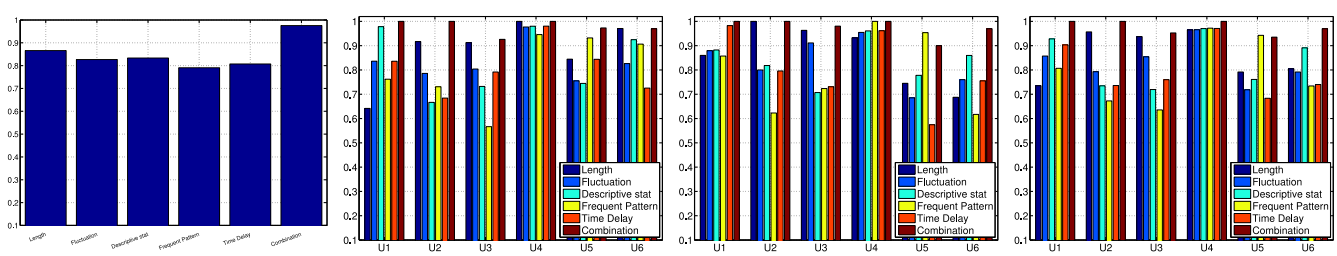
These findings can be helpful for profiling Internet traffic, and thus enabling various mobile applications. For instance, we capture the patterns of Internet traffic from packet length and time delay, typically decomposing a traffic sequence into two sequences of packet lengths and time delay. Moreover, for each decomposed sequence, we define and extract features from multiple perspectives, which can help comprehensively capture the characteristics (e.g., statistical properties, fluctuation patterns, trend patterns, and different directions) of Internet traffic. We even treat the continuous sequence as a string sequence, and capture its patterns in discrete domain. Moreover, these profiling methods can be generalized into other Internet traffic related problems beside in-App analytics.

4.7 A Comparison of Different Classifiers

We compare the effectiveness of different classifiers for classifying in-App service usages in our proposed system. Specifically, we set the number of trees as 100 for random forest. We set the number of estimators as 100, learning rate as 1.0, maximum depth as 1 for gradient boosted trees. We used RBS kernel and set the learning control value as 0.0001 for support vector machine. We randomly divided the data into 80 percent for training and 20 percent for testing. We repeat this process five times and extract the average performance for evaluation.

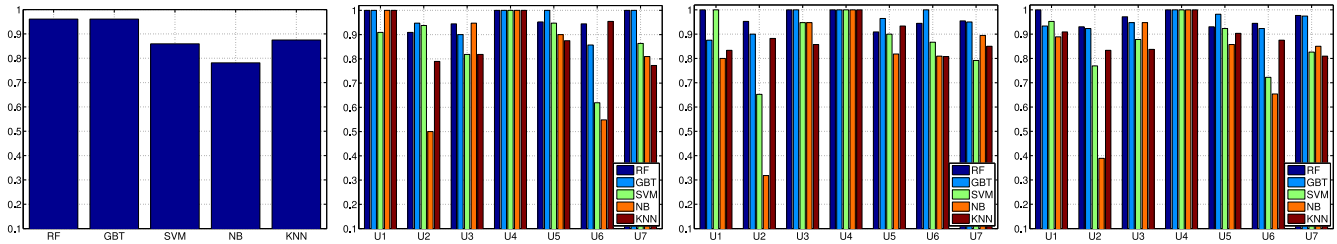
4.7.1 Results on WeChat Data

Fig. 8 plots the results of our framework with different classifiers (e.g., Random Forest, Gradient Boosted Trees, Support Vector Machine, Naive Bayes, K Nearest Neighbors) with respect to overall accuracy, as well as precision, recall and F-measure of different usages. Specifically, Fig. 8a shows the overall accuracies of our proposed method, **RF**, is above 0.96 and significantly outperform the other baseline classifiers. Figs. 8b and 8c show respectively that the precision and recall values of every usage type of our proposed method are larger than 0.9. This results validate that our method can provide a balanced and high accuracy in classifying both mixed usages and individual usages, which illustrate the robustness of our method. By leveraging the power



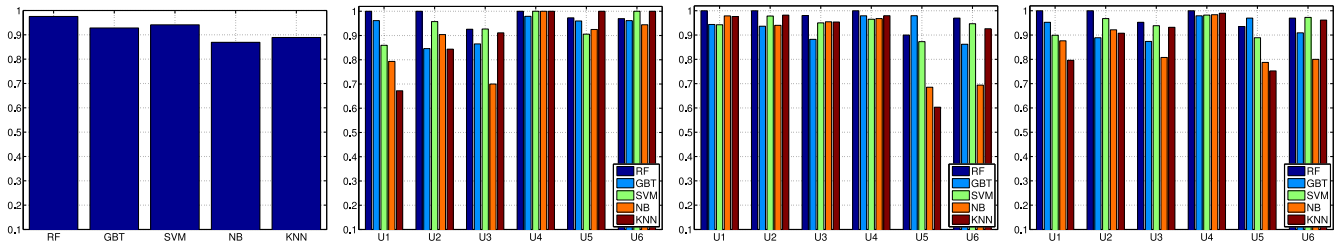
(a) Overall accuracy (b) Precision of different usages (c) Recall of different usages (d) F-measure of different usages

Fig. 7. Performance comparison of different features on the WhatsApp data. U1, U2, U3, U4, U5, and U6, respectively, denote text, audio note, picture, stream voice call, location sharing, and short video.



(a) Overall accuracy (b) Precision of different usages (c) Recall of different usages (d) F-measure of different usages

Fig. 8. Performance comparison of different classifiers on the WeChat data. U1, U2, U3, U4, U5, U6, and U7, respectively, denote text, picture, audio note, stream video call, location sharing, short video, and news feed.



(a) Overall accuracy (b) Precision of different usages (c) Recall of different usages (d) F-measure of different usages

Fig. 9. Performance comparison of different classifiers on the WhatsApp data. U1, U2, U3, U4, U5, and U6, respectively, denote text, audio note, picture, stream voice call, location sharing, and short video.

TABLE 4
The Overall Accuracy of Usage Classification over Different Ratios of Training Data

Ratio of Training Data	0.2	0.3	0.4	0.5	0.6	0.7	0.8
Overall Accuracy on WeChat	0.880273438	0.900223214	0.909375	0.911875	0.922265625	0.927083333	0.9609375
Overall Accuracy on WhatsApp	0.95	0.955944056	0.96576087	0.973641304	0.974509804	0.97173913	0.97398374

of our analytic framework, even KNeighbors also can obtain the precision of 0.772727273 on classifying sending sight. In addition, Fig. 8d plots **RF** and **GBT** continue to be the two best classifiers.

4.7.2 Results on WhatsApp Data

Fig. 9 plots the performance comparison of our method RF and other baseline methods. Specifically, regarding overall accuracy of all six usage types, Fig. 9a clearly illustrate RF significantly outperform other baseline methods, as the accuracy of RF is the highest at 0.975510204. Similarly, a close check will found that the average performances of RF on the six usage types are consistently more robust, stabler, and higher than other baseline methods.

The observations can yield important implications. In our proposed framework, we use ensemble methods, such as Random Forest, as our core predictive engine. Ensemble methods not only combine the power of many weak classifiers, but also increase the randomness of selection training data, and therefore has high robustness and accuracies. The experiments show that if we can properly choose classifiers and accurately design features of Internet traffic, it can significantly increase the overall accuracy for in-App behavior analytics.

4.8 Robustness Check

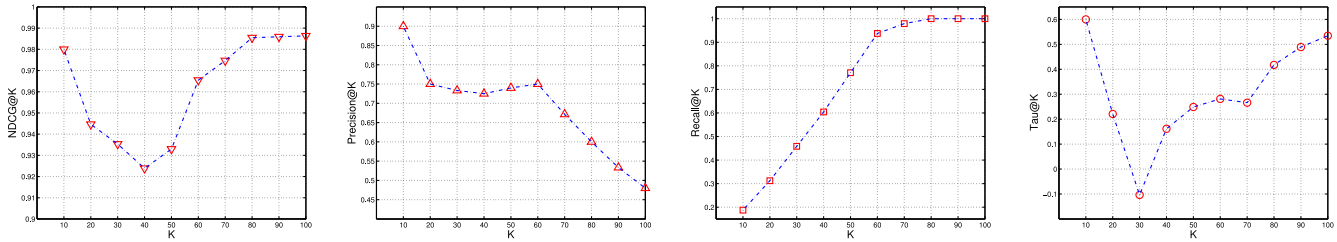
Table 4 shows the sensitivity of overall accuracy over different training data ratio with respect to WeChat and WhatsApp, based on our proposed **RF** method. As can be seen, even we only use 20 percent as training data and 80 percent as test data, our analytic framework can still achieve an overall accuracy of 0.88 in WeChat and 0.95 in WhatsApp. This extreme case shows that, due to the effective feature design and the usage of ensemble classifier, our CUMMA system is robust (i.e., not sensitive) to the ratio of training data. This is an important advantage in practical deployment; that is, even though the collected data are not enough, our system can still accurately classify in-App usages.

4.9 Efficiency

CUMMA is designed to perform offline in-App usage analytics without the requirement of processing traffic in real time, thus efficiency is not a big concern. Table 5 presents the computational performance for major steps in terms of average running time using the data in Table 2. Generally, our proposed analytic framework can be scaled up to more Internet traffic data. Specifically, it is fast, typically less than one minute, to exploit hierarchical clustering for traffic

TABLE 5
The Computational Performance on the WeChat Data

Procedures	Time
Traffic Segmentation (Hierarchical Clustering)	61.6 sec
Traffic Feature Extraction	81 min
Usage Classifier Training (Random Forest)	1.73 sec
Outlier Handling (Kmeans+HMM Training)	23.4 sec



(a) NDCG (b) Precision (c) Recall (d) Kendall's Tau

Fig. 10. Performance of ranking QoE scores of WeChat stream video call.

segmentation and training classifers. Feature extraction is the most time consuming step, because we discretize a continuous sequence into a string sequence, and then extract trendy patterns such as longest increasing or decreasing sequences, which require string matching algorithms.

4.10 The Study of App-Level QoE Scoring

Here, we we propose a two-step method and show how to measure the QoE of WeChat with CUMMA.

4.10.1 Offline Training

We use the divide-and-conquer strategy. In WeChat, the overall QoE is represented by a joint score of the QoEs of seven usage types denoted by $\mathcal{U} = (\{u_n\})_{n=1}^{n=7}$, as listed in Table 1. The QoE of each usage type is described by several QoE features of Internet traffic. For each usage type $u \in \mathcal{U}$, for example, stream video call, we first design several features that are related to the QoE of u, which is denoted by θ_u. Note that different usage types have different QoE features. Then, we collect the *traffic-flows* of u and corresponding groundtrue QoE ratings (i.e., $2 > 1 > 0$) from mobile users via a survey. Later, we extract the QoE features from these *traffic-flows*. We feed these *traffic-flows*, each of which along with a QoE feature vector (θ_u) and a QoE rating, into a linear regression model for training. We regard the trained model as the QoE predictor of u, denoted by $f_u(\theta_u)$, where $f(\cdot)$ is the trained QoE predictor. We repeat the above procedure for all the seven usage types listed, and thus learn seven QoE predictors.

4.10.2 Online Prediction

Given a *traffic-flow* of a specific user in a time interval, we can analyze and output a sequence of usage activities $\mathcal{L} = (\{\boldsymbol{a}_m\})_{m=1}^{m=M}$, where each $\boldsymbol{a} \in \mathcal{L}$ consists of a segmented network traffic (i.e., a *dialog* d) and an identified usage type u. Then, for each usage activity $\boldsymbol{a}$ labeled as the usage type u, we extract its QoE features θ_u and use the trained QoE predictor f_u to predict its QoE. Finally, we combine the predicted QoEs of all the classified usage activities into a final QoE score using a weighted sum function. Particularly, we draw the empirical distributions of groundtrue QoE ratings with respect to different usage types, i.e., 7 QoE distributions for seven usage types $\{P(u = u_1, \xi), \ldots, P(u = u_7, \xi)\}$, where variable ξ represents possible QoE ratings and $\xi \in f(\cdot)$. For each classified usage activity $\boldsymbol{a} \in \mathcal{L}$, we identify the probability of its predicted QoE rating ξ based on the ground-true QoE distribution of classified usage type u, i.e., $P(u, \xi)$. We treat the computed probabilities as weights, and aggregate all the predicted QoEs into a weighted sum score as the final QoE, given by $\sum_{\forall k_i \in K} f_{u_{k_i}}(\theta_{u_{k_i}}) \times P(u = u_{k_i}, \xi = f_{u_{k_i}}(\theta_{u_{k_i}}))$, where $K = \{k_1, \ldots, k_l\}$.

4.10.3 Results of Stream Video Call

Due to the space limit, we select stream video call as an example to demonstrate the effectiveness of QoE scoring in WeChat. Specifically, we first extract three QoE features of video call: buffering ratio [9], average traffic throughput, and duration time. Particularly, buffering ratio is the ratio of the frequency of video stalls (i.e., interruptions) to the duration time of the stream video call; average traffic throughput is defined as total data size divided by duration time and aims to measure the speed of the traffic. We then normalize the three features with $\frac{log(1+x_i)}{log(1+max(\{x_i\}_{i=1}^{n}))}$, and regress the three features to the QoE score by $f_{u_{videoCall}}(\theta_{u_{videoCall}}) = \alpha \times \frac{1}{r_{nor}^2} + \beta \times t_{nor} + (1 - \alpha - \beta) \times d_{nor}$, where r_{nor}, t_{nor} and d_{nor} refer to normalized buffering ratio, throughput, and duration respectively. We collect 187 WeChat *traffic-flow* samples of stream video call from 15 mobile users, and learn $f_{u_{videoCall}}(\cdot)$ with 87 samples in terms of minimizing mean average precision (MAP[3]), then the estimated parameters are $\alpha = 0.5$, $\beta = 0.2$. Note that, each of the *traffic-flows* we sampled here is associated with only one usage type (i.e., stream video call), and we aim to evaluate the ranking performance of $f(\cdot)$, which is the key factor impacting the accuracy for QoE prediction. Therefore, for the rest 100 samples, we examine the ranking accuracy in terms of NDCG@K, Precision@K, Recall@K and Kendall's Tau (Fig. 10). As can be seen, CUMMA can effectively estimate in-App QoEs.

5 Related Work

5.1 Non-Encrypted and Encrypted Traffic Classification

Traffic classification attempts to discern the type of traffic carried on the Internet. Traditionally, non-encrypted traffic classification is based on direct inspection of packet content. The simplest methods are to infer the application of internet traffic by assuming that most applications consistently use well-known TCP or UDP port numbers. However, port based methods suffer from the adoption of unpredictable port numbers in more and more applications [4]. More studies exploited the payload based methods, typically

3. http://en.wikipedia.org/wiki/Information_ retrieval# Mean_ average_ precision

constructing session and application information from IP packet to classify traffic applications [5]. However, pay load based methods are limited by significant complexity and processing load [10]. For security and privacy reasons, many applications have encrypted data transmission. Therefore, encrypted traffic classification exploits traffic statistics or host behaviors for classifying traffic and can be grouped into two sub-categories: (1) flow feature based approaches and (2) host behavior based approaches [11]. On one hand, the flow feature based approaches correlate the statistical properties of traffic flows to the applications of IP traffic [11]. Particularly, supervised classification methods were applied to predict the traffic class of new coming flows, e.g., naive Bayes [12], Bayesian neural network [13], C4.5 decision tree [14], SVM [15] etc. By contrast, to deal with traffics from unknown Apps, researchers adopted unsupervised methods (clustering) to find cluster structures in unlabeled traffic data and assign any testing flow to the application-based class of its nearest cluster [16], [17], [18]. Besides, some well known clustering algorithms such as k-*means* [19], DBSCAN [20], AutoClass [21] were also applied to traffic classification. On the other hand, the host behavior based approaches exploit the host profiles and the network-wide interaction between different hosts (e.g., constructing traffic dispersion graphs) for classifying traffic applications [22], [23].

5.2 In-App Usage Analysis

Mobile user behavior analysis is typically based on massive user digital footprints, such as voice calls and SMSs [24], digital media consumption (video and audio) [24], and Apps usage [25], [26]. Our work is closely related to in-App usage analysis. For example, Falaki et al. [25] developed a custom logging tool to inspect App usage behavior, and found that the behaviors of people using mobile Apps are substantially diverse. Also, Xu et al. [26] collected anonymized IP-level networking traces in a large tier-1 cellular network and distinguished the traffic from different Apps by exploring signatures from HTTP headers. Moreover, Qian et al. [27] proposed a novel approach to expose the cross-layer interaction among various layers to diagnose usage of mobile Apps. Finally, Tongaonkar et al. [28] utilized the advertising traffic originating from the Apps to identify App usage patterns. The work in [29] created a hash table with traffic data and user actions labels, and classified new traffic by query and matching.

5.3 Time Series Segmentation and Classification

First, our work has a close connection with time series classification. In this field, Bakshi and Stephanopoulos [30] proposed a trend based approach for time series classification. However, their results are hard to interpret. Also, Keogh and Pazzani [31] later proposed a piece-wise representation which is robust to noise. To overcome the obstacle of high dimensionality, Jeng and Huang [32] utilized singular value decomposition to select essential frequencies. Second, the segmentation (or summarization) task aims to create an accurate approximation of time series by reducing its dimensionality while retaining its essential features. One of these representative methods is piece-wise linear approximation (PLA) [33]. In the context of PLA, several methods were proposed, such as sliding windows [33], top-down approach [34] and bottom-up approach [31]. More improved approaches were proposed such as fast greedy algorithms [35] and a statistical method for choosing number of segments [36]. Different from PLA, Palpanas et al. [37] proposed a representation of time series that implicitly handles the segmentation of time series by utilizing user-specified amnesic functions, while Abonyi et al. [38] proposed to group time points by their similarity.

6 Concluding Remarks

6.1 Summary

In this paper, we developed a system for classifying service usages using encrypted Internet traffic in mobile messaging Apps by jointly modeling behavior structure, network traffic characteristics, and temporal dependencies. There are four modules in our system including traffic segmentation, traffic feature extraction, service usage prediction, and outlier detection and handling. Specifically, we first built a data collection platform to collect the *traffic-flows* of in-App usages and the corresponding usage types reported by mobile users. We then hierarchically segment these traffic from *traffic-flows* to *sessions* to *dialogs* where each is assumed to be of individual usage or mixed usages. Also, we extracted the packet length related features and the time delay related features from *traffic-flows* to prepare the training data. In addition, we learned service usage classifiers to classify these segmented *dialogs*. Moreover, we detected the anomalous *dialogs* with mixed usages and segmented these mixed *dialogs* into multiple sub-*dialogs* of single-type usage. Finally, the experimental results on real world WeChat and WhatsApp traffic data demonstrate the performances of the proposed method. With this system, we showed that the valuable applications for in-App usage analytics can be enabled to score quality of experiences, profile user behaviors and enhance customer care.

6.2 Implication

Sequence activity analysis has been, and will always be, a significant but challenging problem in various application scenarios. We exploit the divide-and-conquer strategy and present an incremental analytic framework for in-App behavior analysis. This framework includes traffic hierarchical segmentation, traffic feature extraction, traffic classification, and outlier detection and handling, and thus can be broken into small and testable steps with low complexity and high scalability. In addition, we show how to filter, identify, extract, and utilize the information of behavioral hierarchy, network traffic characteristics, and temporal dependency hidden in the sequences of packet length and time delay of Internet traffic. These traffic profiling techniques can be helpful for other mobile messaging Apps, other traffic related applications, and more generally, sequential analysis. With this framework, we respectively achieve 96 and 97 percent overall accuracy in WeChat and WhatsApp under cross validations. Finally, our work has clear benefits for enabling important applications in examining and improving user experience of mobile Apps.

6.3 Future Work

An interesting direction for future research is to expand the analysis beyond classic in-App service usages (e.g., text, voice note, location sharing, etc.) into other topics of detecting languages, fingerprinting users, fingerprinting devices, and inferring locations. A second direction is to generalize the idea of jointly modeling behavior structure, network traffic characteristics, and temporal dependencies to other sequence classification tasks, for example, browsing activity analysis with browser traffic, which is generated by both serial and parallel HTTP requests when loading a HTML webpage of multimedia data.

Acknowledgments

This research was partially supported by Futurewei Technologies, Inc. Also, it was supported in part by Natural Science Foundation of China (71329201) and the Rutgers 2015 Chancellor's Seed Grant Program. Professor Hui Xiong is the corresponding author.

References

[1] (2015). Wechat-free messaging& calling app [Online]. Available: http://www.wechat.com/
[2] (2015). Whatsapp web [Online]. Available: http://web.whatsapp.com/
[3] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "Acas: Automated construction of application signatures," in *Proc. ACM SIGCOMM Workshop Mining Netw. Data*, 2005, pp. 197–202.
[4] T. Karagiannis, A. Broido, M. Faloutsos et al., "Transport layer identification of p2p traffic," in *Proc. 4th ACM SIGCOMM Conf. Internet Meas.*, 2004, pp. 121–134.
[5] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," in n*Proc. 13th Int. Conf. World Wide Webn*, 2004, pp. 512–521.
[6] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Peek-a-boo, i still see you: Why efficient traffic analysis countermeasures fail," in *Proc. IEEE Symp. Security Privacy*, 2012, pp. 332–346.
[7] T. Stöber, M. Frank, J. Schmitt, and I. Martinovic, "Who do you sync you are?: Smartphone fingerprinting via application behaviour," in *Proc. 6th ACM Conf. Security Privacy Wireless Mobile Netw.*, 2013, pp. 7–12.
[8] M. Conti, L. V. Mancini, R. Spolaor, and N. V. Verde, "Can't you hear me knocking: Identification of user actions on android apps via traffic analysis," in *Proc. 5th ACM Conf. Data Appl. Security Privacy*, 2015, pp. 297–304.
[9] A. Balachandran, V. Sekar, A. Akella, S. Seshan, I. Stoica, and H. Zhang, "A quest for an internet video quality-of-experience metric," in *Proc. 11th ACM Workshop Hot Topics Netw.*, 2012, pp. 97–102.
[10] R. Alshammari and A. N. Zincir-Heywood, "Machine learning based encrypted traffic classification: Identifying SSH and skype," in *Proc. IEEE Symp. Computational Intell. Security Defense Appl.*, 2009, pp. 289–296.
[11] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: Myths, caveats, and the best practices," in *Proc. ACM CoNEXT Conf.*, 2008, p. 11.
[12] A. W. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *Proc. ACM SIGMETRICS Perform. Eval. Rev.*, vol. 33, pp. 50–60, 2005.
[13] T. Auld, A. W. Moore, and S. F. Gull, "Bayesian neural networks for internet traffic classification," *IEEE Trans. Neural Netw.*, vol. 18, no. 1, pp. 223–239, Jan. 2007.
[14] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, pp. 5–16, 2006.
[15] A. Este, F. Gringoli, and L. Salgarelli, "Support vector machines for TCP traffic classification," *Comput. Netw.*, vol. 53, pp. 2476–2490, 2009.
[16] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow clustering using machine learning techniques," in *Proc. Passive Active Netw. Meas.*, 2004, pp. 205–214.
[17] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and A. V. Vasilakos, "An effective network traffic classification method with unknown flow detection," *IEEE Trans. Netw. Serv. Manage.*, vol. 10, no. 2, pp. 133–147, Jun. 2013.
[18] J. Zhang, Y. Xiang, Y. Wang, W. Zhou, Y. Xiang, and Y. Guan, "Network traffic classification using correlation information," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 1, pp. 104–117, Jan. 2013.
[19] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, pp. 23–26, 2006.
[20] J. Erman, M. Arlitt, and A. Mahanti, "Traffic classification using clustering algorithms," in *Proc. SIGCOMM Workshop Mining Netw. Data*, 2006, pp. 281–286.
[21] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in *Proc. IEEE Conf. Local Comput. Netw.*, 2005, pp. 250–257.
[22] M. Iliofotou, P. Pappu, M. Faloutsos, M. Mitzenmacher, S. Singh, and G. Varghese, "Network monitoring using traffic dispersion graphs (TDGs)," in *Proc. 7th ACM SIGCOMM Conf. Internet Meas.*, 2007, pp. 315–320.
[23] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: Multilevel traffic classification in the dark," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, pp. 229–240, 2005.
[24] A. Ghose and S. P. Han, "An empirical analysis of user content generation and usage behavior on the mobile internet," *Manage. Sci.*, vol. 57, pp. 1671–1691, 2011.
[25] H. Falaki, R. Mahajan, S. Kandula, D. Lymberopoulos, R. Govindan, and D. Estrin, "Diversity in smartphone usage," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Serv.*, 2010, pp. 179–194.
[26] Q. Xu, J. Erman, A. Gerber, Z. Mao, J. Pang, and S. Venkataraman, "Identifying diverse usage behaviors of smartphone apps," in *Proc. ACM SIGCOMM Conf. Internet Meas. Conf.*, 2011, pp. 329–344.
[27] F. Qian, Z. Wang, A. Gerber, Z. Mao, S. Sen, and O. Spatscheck, "Profiling resource usage for mobile applications: A cross-layer approach," in *Proc. 9th Int. Conf. Mobile Syst., Appl. Serv.*, 2011, pp. 321–334.
[28] A. Tongaonkar, S. Dai, A. Nucci, and D. Song, "Understanding mobile app usage patterns using in-app advertisements," in *Proc. Passive Active Meas.*, 2013, pp. 63–72.
[29] S. E. Coull and K. P. Dyer, "Traffic analysis of encrypted messaging services: Apple imessage and beyond," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, pp. 5–11, 2014.
[30] B. Bakshi and G. Stephanopoulos, "Representation of process trends-iv. induction of real-time patterns from operating data for diagnosis and supervisory control," *Comput. Chemical Eng.*, vol. 18, pp. 303–332, 1994.
[31] E. J. Keogh and M. J. Pazzani, "An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback." in *Proc. 4th Int. Conf. Knowl. Discovery Data Mining*, 1998, pp. 239–241.
[32] S.-L. Jeng and Y.-T. Huang, "Time series classification based on spectral analysis," *Commun. Statist.-Simul. Comput.*, vol. 37, pp. 132–142, 2007.
[33] H. Shatkay and S. B. Zdonik, "Approximate queries and representations for large data sequences," in *Proc. 12th Int. Conf. Data Eng.*, 1996, pp. 536–545.
[34] C.-S. Li, P. S. Yu, and V. Castelli, "Malm: A framework for mining sequence database at multiple abstraction levels," in *Proc. 7th Int. Conf. Inf. Knowl. Manage.*, 1998, pp. 267–272.
[35] J. Himberg, K. Korpiaho, H. Mannila, J. Tikanmaki, and H. T. Toivonen, "Time series segmentation for context recognition in mobile devices," in *Proc. IEEE Int. Conf. Data Mining*, 2001, pp. 203–210.
[36] K. T. Vasko and H. T. Toivonen, "Estimating the number of segments in time series data using permutation tests," in *Proc. IEEE Int. Conf. Data Mining*, 2002, pp. 466–473.
[37] T. Palpanas, M. Vlachos, E. Keogh, and D. Gunopulos, "Streaming time series summarization using user-defined amnesic functions," *IEEE Trans. Knowl. Data Eng.*, vol. 20, no. 7, pp. 992–1006, Jul. 2008.
[38] J. Abonyi, B. Feil, S. Nemeth, and P. Arva, "Fuzzy clustering based segmentation of time-series," in *Proc. 5th Int. Symp. Adv. Intell. Data Anal. V.*, 2003, pp. 275–285.

Yanjie Fu received the BE degree from the University of Science and Technology of China, Hefei, China, in 2008, and the ME degree from the Chinese Academy of Sciences, Beijing, China, in 2011. He is currently working toward the PhD degree in the Management Science and Information Systems Department at Rutgers University. His research interests include data mining and mobile computing.

Hui Xiong (SM'07) received the BE degree from the University of Science and Technology of China (USTC), China, the MS degree from the National University of Singapore (NUS), Singapore, and the PhD degree from the University of Minnesota (UMN). He is currently a full professor and vice chair of the Management Science and Information Systems Department, and the director of the Rutgers Center for Information Assurance at the Rutgers, the State University of New Jersey, where he received a two-year early promotion/tenure in 2009, the Rutgers University Board of Trustees Research Fellowship for Scholarly Excellence in 2009, and the ICDM-2011 Best Research Paper Award in 2011. His general area of research is data and knowledge engineering, with a focus on developing effective and efficient data analysis techniques for emerging data intensive applications. He has published prolifically in refereed journals and conference proceedings (3 books, 60+ journal papers, and 90+ conference papers). He is a co-editor-in-chief of the *Encyclopedia of GIS*, an associate editor of the *IEEE Transactions on Data and Knowledge Engineering (TKDE)*, the *IEEE Transactions on Big Data (TBD)*, *ACM Transactions on Knowledge Discovery from Data (TKDD)*, and *ACM Transactions on Management Information Systems (TMIS)*. He has served regularly on the organization and program committees of numerous conferences, including as a program cochair of the Industrial and Government Track for the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD), a program co-chair for the IEEE 2013 International Conference on Data Mining (ICDM), and a general co-chair for the IEEE 2015 International Conference on Data Mining (ICDM). He is an ACM distinguished scientist and a senior member of the IEEE.

Xinjiang Lu received the BE degree in computing mathmatics from Xinjiang University, Urumqi, China, 2007, the MS degree in software engineering from Norwestern Polytechnical University in 2011, and working toward the PhD degree in computer science at Northwestern Polytechnical University, Xi'an, China. His research interests include data mining and mobile intelligence.

Jin Yang received the BA degree and the ME degree from Tsinghua University, Beijing, China, and the PhD degree of distributed and parallel computing from Imperial College London. He is currently the director of Wireless Big Data Research, Futurewei Inc. His research interests focus on mobile broadband solutions, specially by applying data analytics to new challenges from mobile data applications.

Can Chen received the BA degree from Zhejiang University, Hangzhou, China, 2008, the master of information technology degree from Rutgers University, and is currently working toward the PhD degree in the Management Science and Information Systems Department at Rutgers University. His research interests include researching on data and knowledge engineering.