Effective and Real-time In-App Activity Analysis in Encrypted Internet Traffic Streams

Junming Liu
Rutgers University, USA
jl1433@rutgers.edu

Yanjie Fu
Missouri U. of Science and Technology, USA
fuyan@mst.edu

Jingci Ming
Rutgers University, USA
jingci.ming@rutgers.edu

Yong Ren
Futurewei Tech. Inc, USA
Yong.Ren1@huawei.com

Leilei Sun
Tsinghua University, China
Dalian University of Technology
leisun@mail.dlut.edu.cn

Hui Xiong*
Rutgers University, USA
hxiong@rutgers.edu

ABSTRACT

The mobile in-App service analysis, aiming at classifying mobile internet traffic into different types of service usages, has become a challenging and emergent task for mobile service providers due to the increasing adoption of secure protocols for in-App services. While some efforts have been made for the classification of mobile internet traffic, existing methods rely on complex feature construction and large storage cache, which lead to low processing speed, and thus not practical for online real-time scenarios. To this end, we develop an iterative analyzer for classifying encrypted mobile traffic in a real-time way. Specifically, we first select an optimal set of most discriminative features from raw features extracted from traffic packet sequences by a novel *M*aximizing *I*nner activity similarity and *M*inimizing *D*ifferent activity similarity (MIMD) measurement. To develop the online analyzer, we first represent a traffic flow with a series of time windows, which are described by the optimal feature vector and are updated iteratively at the packet level. Instead of extracting feature elements from a series of raw traffic packets, our feature elements are updated when a new traffic packet is observed and the storage of raw traffic packets is not required. The time windows generated from the same service usage activity are grouped by our proposed method, namely, recursive time continuity constrained KMeans clustering (rCKC). The feature vectors of cluster centers are then fed into a random forest classifier to identify corresponding service usages. Finally, we provide extensive experiments on real-world traffic data from Wechat, Whatsapp, and Facebook to demonstrate the effectiveness and efficiency of our approach. The results show that the proposed analyzer provides high accuracy in real-world scenarios, and has low storage cache requirement as well as fast processing speed.

*Corresponding Author

KDD '17, August 13-17, 2017, Halifax, NS, Canada
© 2017 Association for Computing Machinery.
ACM ISBN 978-1-4503-4887-4/17/08. . . $15.00
https://doi.org/10.1145/3097983.3098049

CCS CONCEPTS

• **Information systems** → **Mobile information processing systems**; **Data stream mining**; **Clustering and classification**;

KEYWORDS

Time Series Segmentation; Internet Traffic Analysis; In-APP Analytics; Service Usage Classification

1 INTRODUCTION

With the wide spread use of mobile devices, the messaging apps have been providing new ways of communication, such as text messaging, muti-media messaging, broadcast messaging, real-time chat, location sharing and moment post. This, in turn, motivates the emergence of in-App service usage analytics, which aims for understanding in-App user behaviors. Indeed, in-App service usage analytics has recently become critical for mobile companies to enhance user experiences and for Internet providers to provide intelligent network resource distributions [7]. A key task of In-App service usage analytics is to effectively classify mobile Internet traffic into different usage categories in a real-time manner. However, this is a challenging task due to the increasing adoption of secure protocols for in-App services. Moreover, as an online traffic analyzer, it should be able to handle millions of traffic flows from many users simultaneously. This requires fast processing speed and low storage cache requirement.

There are traditional packet inspection methods for the Internet traffic classification. These methods analyze the protocol types, port numbers or protocol signatures with the limited use of cache memory [9, 22, 29]. However, due to the limited available information and the complexity of packet encryption methods, the packet inspection method is not applicable for mobile messaging Apps with secure communication protocols. Recently, some researchers have tried to tackle these issues with data mining solutions [2, 6, 7], which first segment an Internet traffic flow into single-usage subsequences and then discover distinctive traffic patterns for single-usage classification. While the patterns discovered can help improve the classification accuracy, the feature extraction process is very slow and both of the segmentation algorithm and classifiers require the storage of a large set of raw traffic packet data. Moreover, the segmentation algorithm at the packet level is not accurate, and thus the segmented subsequence is not single usage due to the packet length variance. As a result, the features extracted from the inaccurately segmented subsequence are from mixed service usages that deteriorate classification performance.

To address these challenges, in this paper, we develop an online iterative mobile app traffic analyzer that comprises of a recursive time continuity constrained KMeans clustering (rCKC) algorithm for traffic flow segmentation and a Random Forest classifier for segmented traffic classification based on time window representation with adaptable features. Specifically, we first discretize the incoming traffic flows with a series of time windows, each of which is represented by the iterateable feature vector selected by our MIMD feature selection. The iterateable feature vector has the property that each feature element can be updated and the packet is released once a new traffic packet is received. The time window series are then grouped together by the rCKC segmentation algorithm that the time windows within one cluster represent the traffic flows generated from the same service usage. Finally, the cluster center feature vector is fed into a Random Forest classifier for service usage activity classification. As a result, the requirement for cache memory is reduced by more than 1000 times and the processing speed is increased by more than 10 times compared with existing studies. Moreover, we also demonstrate the applications of our method on two types of mobile Apps: Social Media Apps such as Facebook[1] and messaging Apps such as Wechat[2] and Whatsapp[3]. As shown in the results, the proposed analyzer provides high classification accuracy in real-world online scenarios, and has low cache requirement as well as fast processing speed.

2 PROBLEM FORMULATION

In this section, we first define some preliminaries. Then, we introduce the problem of Mobile App Traffic segmentation and classification.

2.1 Preliminaries

Here, we first introduce some preliminaries, which will be used throughout this paper.

Definition 1: *Internet traffic flow.* An internet traffic flow TF consists of a sequence of encrypted internet packets denoted by $TF = (\{t_i, P_i\})_{i=1}^{I}$ where I is the total number of packets and P_i represents the i-th packet received at time t_i. Each packet contains limited information including the packet length $P_i.l$, sender IP address $P_i.SIP$, receiver IP address $P_i.RIP$ and protocol type $P_i.Pr$. An internet traffic flow may contain one or more mobile user's usage activities.

[1]https://www.facebook.com/mobile/
[2]http://www.wechat.com/en/
[3]https://www.whatsapp.com/

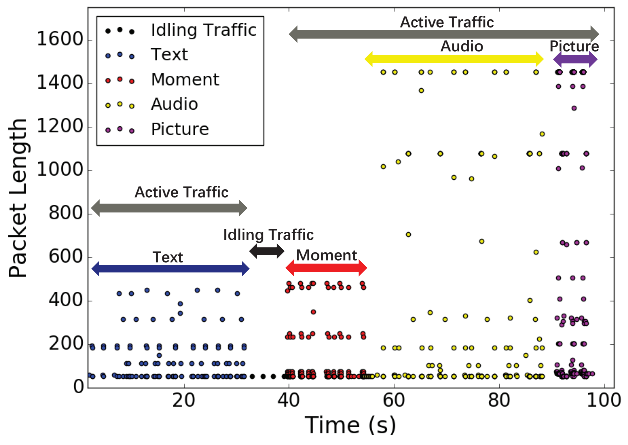


Figure 1: Examples of Wechat traffic flow

Definition 2: *Traffic Segment.* A traffic segment $S =< s_0, s_t >$ is a subsequence of an internet traffic flow from time s_0 to s_t. Segment operation $T(S)$ extracts the segment time duration $T(S) = s_t - s_0$. Operation $V(S)$ extracts the packet volume $V(S) = \sum_{P_i \in S} P_i.l$.

Figure 1 shows an example of traffic flow consisting of two activate traffic flows separated by a segment of idling traffic. The idling traffic represents the traffic flows of background packets when no service usage is activated. The second activate traffic flow includes three segments of traffic: browsing moment, sending audio note, and sending picture.

Definition 3: *Time Window Representation.* Instead of segmenting an internet traffic flow by comparing adjacent packets which is time-consuming and also introduces large variance, we divide a traffic flow into a set of continuous time windows. A time window W_n records a portion of traffic sequence starting from t_0^n to $t_{w_n}^n$. The size of a time window τ is fixed, s.t. $t_{w_n}^n - t_0^n \leq \tau$. There is a time gap Δ, s.t. $0 \leq t_0^{n+1} - t_{w_n}^n \leq \Delta$, between two adjacent time windows to ensure the smoothness of time window feature vector F_n.

2.2 Problem Definition

Given an incoming traffic flow $TF = (\{t_i, \bar{P_i}\})_{i=1}^{I}$, the mobile traffic analyzer is built to segment and classify a sequence of in-App usage activities denoted by $(\{b_n, e_n, \boldsymbol{u}_n\})_{n=1}^{N}$, where b_n, e_n, and $\boldsymbol{u}_n$ respectively represent the begin time, the end time, and the activity class. Essentially, there are two major tasks: traffic flow segmentation and traffic segment classification:

(1) *traffic flow segmentation*: the objective of our first task is to segment the origin traffic flow into multiple traffic segments $TF = \{S_1, S_2, ...S_n\}$, such that the traffic packets within each segment S are generated from the same single-usage activity.

(2) *traffic classification*: the second task is to classify the major usage activity given a segmented traffic.

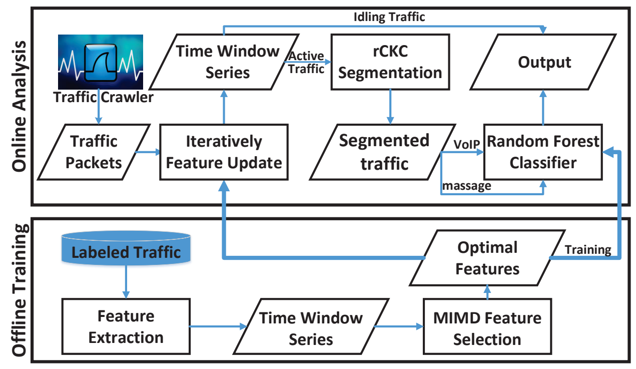


Figure 2: The Framework Overview.

2.3 Framework Overview

Figure 2 shows the framework of our proposed method which consists of two processes: offline training and online analysis. **Offline Training**. The offline process is conducted to learn the traffic patterns, test segmentation algorithm and train classifiers with labeled traffic flows. As shown in Figure 2, the offline process consists of four parts:
1) *Labeled traffic collection.* We collect the traffic flows of different mobile apps from a group of volunteers and employees of leading ICP provider company. Each traffic flow contains a sequence of packets with the corresponding packet receiving time, packet length, and packet protocol type. The service usage types, their start time and end time are reported by the data collectors.
2) *Traffic patterns extraction.* Mobile traffic patterns are extracted from a multi-view perspective: packet length related features and packet time delay related features. Totally 30 features are extracted as a full feature set in this part.
3) *MIMD feature selection.* To reduce the workload of feature extraction and save cache space for feature vectors, 6 most discriminative traffic patterns are selected as the optimal feature set using the MIMD feature selection criteria.
4) *Classifier Training.* The selected feature vectors and their labeled service activities are used for the Random Forest Classifier training. In this classifier, the major protocol types (TCP or UDP) are first used to filter the traffic into two major categories: VoIP (video call and voice call), and other usages (other usage activities including text, picture, audio, moment, etc). The usage services of each major category are then classified respectively.
Online Analysis. The online process is designed to analyze incoming streaming traffic flows. Different from the Offline process, the online process constructs the time windows and their feature vectors in an iterative way at traffic packet level. For each received traffic packets, its information (time stamp, packet length and protocol type) is collected to update the feature vector of the current time window. And the packet is released without the requirement of cache storage. Then the rCKC segmentation separates time windows into different groups and labels the time boundaries of single usage activity traffic segment. The segmented traffic can be easily inferred by incorporating the Random Forest Classifier learned in the offline process.

3 DATA COLLECTION

Table 1: Usage Activities of Different Mobile Apps

U#	Wechat	Whatsapp	Facebook
0	Audio	Audio	Moment
1	Location	Picture	Video upload
2	Picture	Voice Call	Video watch
3	Short Video	Text	Picture
4	Video Call	Short Video	New Video Upload
5	Moment	Location	
6	Text		
7	Voice Call		

To guarantee the data quality and facilitate our experiments, we design and deploy the data collection platform for off-line model training. With the cooperation of leading ICP provider company, we recruit volunteers and employees who are equipped with our specially configured smartphones with brand new Android or IOS. The traffic flows are captured by WireShark[4], a well-known traffic packet sniffing and analyzer tool, and we utilize it to crawl the packet information (packet time stamp, packet size, and protocol type) of traffic flows transmitted through our specific Internet access point. Meanwhile, data collectors report their corresponding usage types and activate time range as ground-truth. Finally, both the internet traffic and the reported usages are organized, archived and stored in our data repository.

4 METHODOLOGY

In this section, we first present our MIMD feature selection framework to select the most discriminative features. Then we provide the technical details of our rCKC algorithm for mobile APP traffic flow segmentation and Random Forest Classifier for segmented traffic classification.

4.1 MIMD Feature Selection

A set of 28 feature elements has been analyzed in our previous work [7] which provides the best usage activity classification accuracy. Here we list the 28 feature elements (more details of the feature element definitions can be found in [7]): **Packet length related features:** packet length maximal, minimal, mean, variance, kurtosis, skewness, hopping count, length of longest monotone (including increasing and decreasing) subsequences, size percentiles (4 subranges), 3 set of forward variances, 3 sets of backward variance, 3 most frequent subsequences; **Packet time delay related features:** time delay maximal, minimal, mean, variance, kurtosis, skewness. In addition, we add two more features: the traffic packet density (number of packet per second) and traffic speed (average

[4]www.wireshark.org

packet size per second) as our full feature set. However, although these feature elements can help achieve a high traffic flow classification accuracy, some of them are redundant and most of them are not iterateable. In other words, in order to extract the full set of features, the traffic analyzer needs to store all the raw traffic data, which makes the analyzer inapplicable as an online streaming traffic tool. In the following, we denote the feature set of 30 features as "full feature set" and conduct a feature selection algorithm to select the most representative feature set as our "optimal feature set". Besides that, the percentage of packets using TCP or UDP protocol is calculated for VoIP traffic filtering in order to reduce the problem complexity of the multi-class classification.

Motivated by the internal clustering validation measures [18, 25, 30], the optimal feature set for segmentation is selected to better group the packets that are generated from the same activity (Compactness) and to separate the packets from different activities (Separation) with the lowest time consumption. In this the following, we describe the two criteria of our recursive MIMD feature selection for the clustering based traffic flow segmentation.

Feature Similarity Function. After Z-score feature standardization, each feature element is rescaled to form a standard normal distribution. The similarity distance between any two feature vectors F, F' of dimension N are defined as:

$$SD(\mathbf{F}, \mathbf{F'}) = \frac{1}{N} \sum_{n=1}^{N} e^{-(F_n - F'_n)^2} \tag{1}$$

Maximal Inner Activity Similarity (IAS). For the packets generated from the same activity, their lengths and time intervals will vary. The features should not be sensitive to the inner cluster packet variance. In other words, for the time windows representing the continued activity, they should have high similarities in feature domain such that the segmentation algorithm will not mistakenly separate them as time windows from different activities. Mathematically, given a traffic flow from a single usage activity a_i represented by n_i^a time windows, the Maximal Inner Activity Similarity requires the maximal feature distance between the time windows $\mathbf{F}_{i,k}$, $k = 1, 2, ..., n_i^a$ and their segment center $\bar{\mathbf{F}}_i^a = \frac{1}{n_i^a} \sum_{k=1}^{n_i^a} \mathbf{F}_{i,k}$:

$$\max IAS(A; \mathbf{F}), \quad IAS(a_i; \mathbf{F}) = \frac{1}{n_i^a} \sum_{k=1}^{n_i^a} SD(\mathbf{F}_{i,k}^a, \bar{\mathbf{F}}_i^a) \tag{2}$$

Minimal Different Activity Similarity (DAS). The DAS measures how distinct that the feature vector representing one usage activity is from that representing another. For the packets generated from different activities, the selected features should be able to discriminate them by a large distance in the feature domain.

$$\min DAS(A \neq A'; \mathbf{F}), \quad DAS(a_i, a'_j; \mathbf{F}) = SD(\bar{\mathbf{F}}_i^a, \bar{\mathbf{F}}_j^{a'}) \tag{3}$$

Objective Function. We define the objective function Φ to combine the average IAS and DAS calculated from our offline single activity traffic flows and optimize IAS and DAS simultaneously.

$$\max \Phi(IAS, DAS)), \quad \Phi(IAS, DAS) = IAS - DAS \tag{4}$$

Recursive Addition MIMD feature selection. When the full feature set is large, considering all feature subset is time costly. Therefore, we propose a recursive addition MIMD feature selection to select the optimal feature set. Specifically, we start with one feature that can maximize $\Phi(IAS, DAS)$ and recursively add one additional feature that can maximize the objective function. The addition step is repeated until the full set is reached. At last, we select the optimal feature set that has the maximal objective compared to all feature sets of different dimensions.

4.2 Traffic Flow Segmentation

Time series segmentation has been investigated to reduce the complexity of time series analysis. However, traditional time series segmentation algorithms such as Fourier Transforms, Wavelets, Symbolic Mappings and Piecewise Linear Representation are not applicable for mobile APP traffic time series due to the high density of Internet traffic flows and big variance of packet lengths. Motivated by the problem of image segmentation that the segmented pixel region must be spatial connected, in this paper, we propose a recursive Constrained KMeans Clustering (**rCKC**) algorithm with time continuity constraints for mobile App Internet traffic segmentation. To the best of our knowledge, such kind of constrained clustering algorithm has not been studied ever before in this problem.

After the time window representation, the problem of the traffic flow segmentation is to group a sequence of time windows $\{w_i(\mathbf{F}; t_i), i = 1, 2, ..., N\}$ into multiple clusters within which the time windows are the observation of traffic flows generated from the same usage service. In addition to the general clustering objective that within each cluster the distances between each time window are small and the distance between cluster centers are large, we add the time continuity constraints that the time windows within a cluster are timely connected as a link (w_i, w_{i+1}). Algorithm 1 presents the proposed CKC algorithm that recursively segments the origin traffic flows: Step 1 checks if the IAS of segment S is below the single activity IAS threshold δ. If the segment is identified as a single activity segment, no segment split is required and the feature vector of this segment center is output for classification. A segment with small IAS indicates a high probability that this segment contains more than one activity and should be split into sub-segment. The threshold δ is chosen as the lower bond of IAS that is calculated by the feature selection off-line. The segment split starts in step 4 with initiating K sub-segment centers by maximizing the distance between adjacent segment centers $D(c_j, c_{j+1})$. Step 7 assigns the time windows to each segment centers such that the average distance from the time windows to its segment center is minimized. Step 8 recalculates the segment center. Step 7 and 8 are repeated until the segment centers do not change and the segment is separated into K sub-segments $\{S_j, j = 1, 2, ..., K\}$. The algorithm acts on each sub-segment to check if further segment split is necessary in step 9 and 10. In our experiments, we set $K = 2$.

Algorithm 1 $rCKC(S = \{w_i, i = 1, 2, ..., N\}, K)$

```
Require: Input: $\{w_i(F_i; t_i), i = 1, 2, ..., N\}$
 1: if $IAS(S) > \delta$ then
 2:     output $S = \{w_i, i = 1, 2, ..., N; F(S)\}$
 3: else
 4:     Initial: $C^0 = \arg\max_{c_j \in \mathbf{W}} \sum_{j=1}^{K} DAS(c_j, c_{j+1})$
 5:     while $C^p \neq C^{p+1}$ do
 6:         $p \to p + 1$ %next iteration
 7:         $b^p = \arg\max_{b^p} IAS(S(w_{b^p} : w_N))$;
 8:         $C_1^p = \frac{1}{b^p} \sum_{i=1}^{b^p} (w_i)$
 9:     end while
10:     for $j = 1 : K$ do
11:         rCKC($S_j$, $K$)
12:     end for
13: end if
```

4.3 Segmented Traffic Classification

By feature extraction, we can output a set of vectorized traffic segments $S = (\boldsymbol{s}_j)_{j=1}^{J}$ where each is represented by a feature vector. To predict C types of service usages, we exploit the ensemble power of Random Forest with filtering (denoted as FRF). Specifically, by observing the traffic patterns, the traffic density and traffic speed can directly discriminative the VoIP activity (video call and voice call) from the rest. We first apply the filtering technology that separates the VoIP activity directly and the classification for the rest is conducted using the general technique of bagging of tree learners. By doing this, the C class classification task is simplified as a $C' < C$ class classification problem. Let B denote the number of trees, given a set of labeled traffic segments $S = (\boldsymbol{s}_j)_{j=1}^{J}$ with usage types $U = (u_j)_{j=1}^{J}$ where $u_j \in 1, 2, .., C'$, we repeatedly select a random sample with replacement of the labeled traffic segments, and respectively fit B decision trees to the B samples. After training, predictions for unknown traffic segment d' can be made by averaging the predictions from all the individual decision trees on the traffic segment d'. In particular, the random forest gives the probability estimation for each class c as follows:

$$P(c|s') = \frac{1}{B} \sum_{b=1}^{B} P_b(c|s') \tag{5}$$

where $P_b(c|s')$ is the probability estimation of usage type c given by the b-th tree. It is estimated by computing the ratio that usage type c gets votes from the leaves of the b-th tree. The overall decision function of random forest is defined as:

$$u' = argmax_c P(c|s') \tag{6}$$

5 EXPERIMENTAL RESULTS

To validate the efficiency and effectiveness of our proposed method, extensive experiments are performed on real-world traffic data from three mobile applications: Wechat, Whatsapp, and Facebook. For each application, we first collect a set of single activity traffic for feature selection and classifier training, and another set of traffic flows containing two activities (including one type activity with idling traffic) for traffic flow segmentation & classification testing. After the off-line model training and testing, an online performance test is conducted. All experiments are conducted on a PC with an Intel(R) Core i7-4790 CPU, 3.6 GHz, and 16 GB RAM running 64-bit Windows 10 system.

Table 2: Statistics of the WeChat Training data.

#	Usage Type	Records	Packets	Traffic	Tra/min
1	audio	136	44K	23.53M	208K
2	location	112	119K	31.79M	348K
3	picture	100	132K	103.2M	986K
4	sight	63	163K	141.11M	1.33M
5	video call	100	1,170K	239.76M	2.17M
6	moment	67	7K	1.18M	50K
7	text	229	30K	4.5M	32K
8	voice call	105	265K	32.54	758K

Table 3: Statistics of the Whatsapp Training data.

#	Usage Type	Records	Packets	Traffic	Tra/min
1	audio	176	72K	26.62M	436K
2	picture	197	178K	141.5M	2.26M
3	call	194	143K	21.64M	287K
4	text	202	34K	3.22M	42K
5	video	173	483K	472M	11.06M
6	location	80	11.52K	8.03M	47.77K

Table 4: Statistics of the Facebook Training data.

#	Usage Type	Records	Packets	Traffic	Tra/min
1	moment	101	40K	21.65M	607K
2	videoup	75	21K	6.56M	238K
3	video watch	108	1,216K	1,326M	42M
4	picture	97	57K	51M	2.26M
5	new video	77	844K	825M	10M

5.1 Experimental Data

Table 2, 3 and 4 show the basic statistics of our collected single activity traffic data of different types of service usages with respect to WeChat, WhatsApp and Facebook. In addition to this single activity traffic data, we collect two-activity traffic data that contains a segment of one activity traffic flow followed by another segment of traffic generated from a different service usage. Each activity-activity pair has 30 records and the time duration for each segment ranges from 5 seconds to 120 seconds.

5.2 Feature Engineering

Feature selection. The recursive addition MIMD feature selection progress is presented in Figure 3. As the dimension of the optimal feature set increases from 1 to 30 (full set), both of the IAS (red circle dot line) and the DAS (blue circle dot line) firstly increase and then decrease. The IAS reaches its maximum at the feature set of dimension 7 when the inner cluster time windows have the least variations. However, the DAS increases significantly which makes different segments less discriminative. By combining these two criterions, we find the objective (black circle dot line) Φ reaches its maximum at the feature set of dimension 6. Moreover, the cross-validation score (green square dot line) at dimension 6 is slightly lower (0.55%) than its highest value at dimension 25. Considering a higher dimension of features requires more memory and time consumption in feature extraction. Here in our following experiments, we use the feature set of dimension 6 as our optimal feature set. From a time window observation of N packets $\{(t_1, P_1), (t_2, P_2), ..., (t_N, P_N)\}$, the feature elements are defined and calculated as follows:

1 Percentile 25% P_{25}: percentage of packets with packet length smaller than 25% maximum packet length L_{max}. Mathematically, $P_{25} = \frac{1}{N}\sum_{i=1}^{N}\delta(P_i.l < 25\%L_{max})$. Where $\delta(.)$ is the delta function that equals to 1 if the condition is true.
2 Percentile 75% N_{75}: percentage of packets with packet length larger than 75% maximum packet length. Mathematically, $P_{75} = \frac{1}{N}\sum_{i=1}^{N}\delta(P_i.l > 75\%L_{max})$.
3 Top frequent continuous subsequence TCS: the highest repeating frequency of packet subsequence of length 3 (see detailed calculation in [7]).
4 Packet length variance var: the variance of packet length: $var = \frac{1}{N}(\sum_{i=1} NP_i.l^2) - (\frac{1}{N}\sum_{i=1}^{N} P_i)^2$
5 Traffic density TD: the average number of packets per unit time (second): $TD = \frac{N}{t_N - t_1}$.
6 Traffic speed TS: the average packet lengthes per unit time (second): $TS = \frac{\sum_{i=1}^{N} P_i.l}{t_N - t_1}$.

The ratio R_{pr} of packets number using UDP N_T and TCP N_U is collected for VoIP traffic filtering. After we determine the optimal feature set, the rCKC stopping threshold $\delta = 0.813$ is chosen to be the lower bond of IAS at dimension 6.

Iterative feature update. The iteratively time window feature update is illustrated in Algorithm 2. To achieve the feature iteration, two sets of small temporal vectors tem, tem' are iteratively updated. Once the analyzer receives a packet P, step 2 determines if a time window of packets (τ seconds of traffic flow) are observed. step 3 updates the tem vector to fill more packet information if the time window is not full. Step 5 updates the bottom half time window information for next time window updates. Once the time window is filled up, the feature vector is calculated and stored in Step 7∼9. The bottom half time window information is used as the top half of the next new time window. After these updates, the packet P is released. As a result, the raw packets are not stored in the cache which significantly reduces the storage requirement. Once the memory stores a number of time window features, these time windows are used for traffic analysis (segmentation and classification) and the memory is released for new incoming time windows.

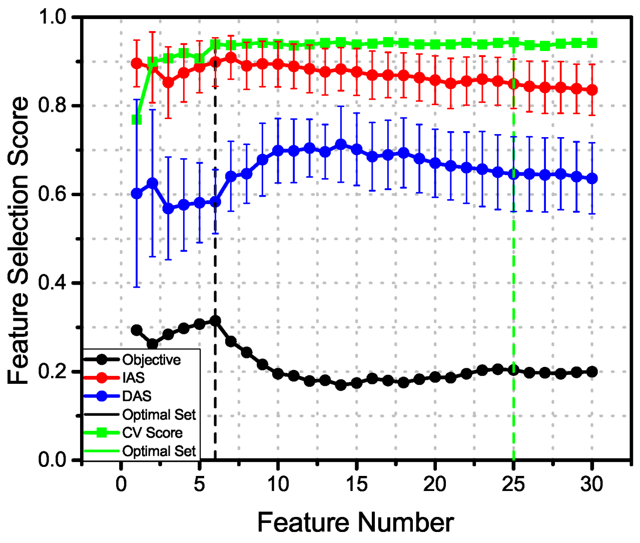


Figure 3: Feature Selection Progress.

The tem vector update function $Update(tem, P)$ works as follows: when a packet P of length $P.l$ arrives, variable N records the number of packets in current window and updates as $N+ = 1$; variable $N_{25}(N_{75})$ records the number of packets within $< 25\%$ ($> 75\%$) maximum packet length and updates as $N_{25}+ = \delta(P_i.l < 380)$, $N_{75}+ = \delta(P_i.l > 1135)$; variable FCS is a dictionary records the accumulate frequency of sequence $(P_{-2}.l, P_{-1}.l, P.l)$; variable N_T, N_U records the number of packet using $TCP(UDP)$ protocol and updates as $N_T+ = \delta(P.pr =' TCP')$, $N_U+ = \delta(P.pr =' UDP')$, where $\delta(.)$ is the delta function which equals to 1 if the condition is TRUE otherwise 0; L, L^2 records the summation of packet length and the square of packet length and updates as $L+ = P.l$, $L^2+ = P.l^2$. The packet is used for updating the tem vector only. Once the packet information is extracted, it is released without storage which significantly reduces the cache storage requirement for online traffic analyzer.

5.3 Baselines & Metric

5.3.1 Study of Single Activity Classifier. The single traffic flow activity classifier proposed in our work is denoted as the hierarchical random forest classifier (**HRF**). To confirm the effectiveness, we use the following four baseline algorithms for performance comparison: 1. Random Forest (**RF**); 2. Support Vector Classifier (**SVC**): The multi-class SVC builds a set of one-to-one classifiers and choose the class that is selected by the most classifiers [8]; 3. K-Nearest Neighbors classifier (**KNN**); and 4. Gaussian Naive Bayesian (**GNB**). The baselines are We measure the classification performance of different classifiers using the evaluation metrics of overall accuracy for all classes, and the Precision, Recall, F-Measure for each individual service usage category.

Algorithm 2 Iteratively update feature and time window

```
Require: Temple variable (initial 0):
    $tem$=$(N, N_{25}, N_T, N_U, L, L^2, TCS)$
    $tem'$=$(N', N'_{25}, N'_T, N'_U, L', L^2, TCS')$
1:  while Receive packet $P$ do
2:    if $P.t - T_0 \leq \tau$ then
3:      $Update(tem, P)$, $T_t = P.t$
4:      if $P.t - T_0 \geq \Delta$ then
5:        $Update(tem', P)$
6:      end if
7:    else
8:      $N_{25} = \frac{N_{25}}{N}$, $N_{75} = \frac{N_{75}}{N}$, $var = \frac{L^2}{N} - \mu^2$
9:      $TCS = \max_{L \in FCS} FCS(L)$
10:     $TS = \frac{L}{T.t - T_0}$, $TD = \frac{N}{T.t - T_0}$, $R_{pr} = \frac{N_U}{N_T}$
11:     Store feature: $N_{25}, N_{75}, var, TS, TD, R_{pr}$
12:     $tem = tem'$, $tem' = \mathbf{0}$, $Update(tem, P)$, $T_0$=$P.t$
13:   end if
14: end while
```

5.3.2 Study of traffic flow analyzer. The mobile traffic flow analyzer proposed in our work compromises of rCKC segmentation and Random Forest Classifier with filtering and is denoted as (**rCKC+FRF**). In order to confirm the effectiveness of our models, we conduct experiments to compare our methods with the following baselines:

- *AC+RF* [6]. This analyzer consists of an Agglomerative Connectivity Constrained Clustering (AC) and a Random Forest Classifier. The segmentation groups the timely adjacent time windows from bottom to up until a specific number of clusters is achieved.
- *CUMMA* [7]. The CUMMA traffic analyzer consists of adjacent packet merging strategy at packet level for traffic segmentation and a Random Forest Classifier for segmented traffic classification.
- *SW+RF* [11]. The sliding window (SW) based segmentation compares similarities between adjacent time windows in feature domain and combine the windows with a local search strategy. The average feature vector of adjacent time windows with high similarities are fed into the Random Forest Classifier.

The metrics we adopt to measure the performance are traffic duration accuracy (**TDA**) and traffic volume accuracy (**TVA**). The traffic duration accuracy evaluates the total correct service usage activity time durations and the traffic volume accuracy evaluates the total packet lengths that are correctly labeled. Given a traffic flow TF containing $\{\hat{S}\}$ traffic segments, their ground truth activities $\{\hat{a}_s\}$, our segmented traffic $\{S\}$ and predicted activities $\{a_s\}$, the two evaluation metrics **TDA** and **TVA** are formally formulated as follows:

$$TDA = \frac{1}{T(TF)} \sum_{S} \sum_{\hat{S}} \delta(a_s - \hat{a}_s) T(S \cap \hat{S}) \quad (7)$$

$$TVA = \frac{1}{V(TF)} \sum_{S} \sum_{\hat{S}} \delta(a_s - \hat{a}_s) V(S \cap \hat{S}) \quad (8)$$

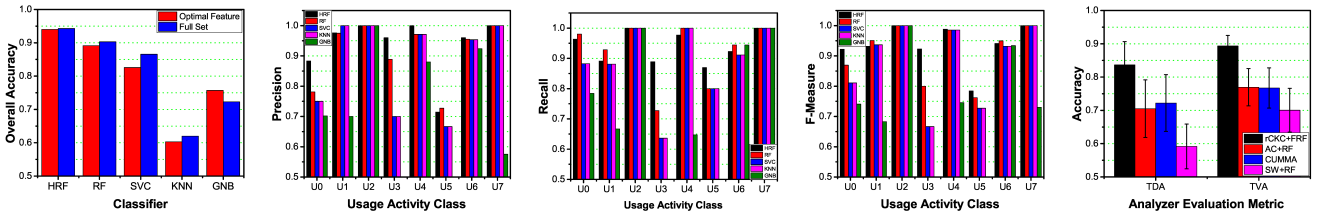
Where $\delta(a_s - \hat{a}_s)$ is the delta function that equals to 1 if a_s equals $\hat{a}_s$ otherwise equals to 0. The intersection operation $S_1 \cap S_2$ captures the traffic segment shared by both segment S_1 and segment S_2.

5.4 Off-line Performance Test

Results on Wechat. The overall accuracy of different classifiers for Wechat single activity classification based on full feature set and optimal feature set is presented in Figure 4(a). As can be seen, our method outperforms the baselines with significant improvement using both feature sets. Moreover, by choosing the optimal feature set, our proposed method achieves an overall accuracy of 94.01%, slightly lower (-0.3%) than the overall accuracy using full feature set. Our HRF achieves a better accuracy than the RF algorithm indicates the VoIP-noVoIP separation can help boost the classifier performance by reducing the classes size. Figure 4(b) $\sim$ 4(d) shows the results of each approach in terms of precision, recall, and f-measure respectively using the optimal feature set. Comparing to the baseline algorithms, our method achieves the highest precision, recall and f-measure, indicating a balanced accuracy for each specific usage service. Among the eight usage activities, all classifiers have the worst performance on U5 (moment). The moment usage activity of Wechat APP is similar to that of browsers, and is the most challenging class for single activity prediction since it contains mixed traffics ranging from browsing (or posting) pictures to text only moment. Our method still achieves a f-measure of 0.7843 for this specific challenging task. Figure 4(e) presents the overall TDA and TVA of different traffic analyzer tests on the 2-activity pair traffic flows and our proposed analyzer significantly outperforms the state-of-the-art baselines with an overall TDA of 0.8366 and an overall TVA of 0.8935.

A more detailed TDA and TVA of 2-activity traffic flows are presented in Figure 5 that contains 8*8 sub-figures labeled by two usage activities. For example, the sub-figure "audio-picture" shows the performance test on two-activity traffic flow including a segment of audio traffic followed by a segment of picture traffic. The diagonal sub-figures shows the performance of our analyzer on short single activity traffic flows. As can be seen, our proposed traffic analyzer outperforms other baselines in all usage service environments.

Results on Whatsapp. The Whatsapp traffic flow has very similar traffic patterns compared to Wechat. The performance tests on Whatsapp is presented in Figure 6. The overall accuracy of our proposed classifier achieves an overall accuracy of 90.47% based on the optimal feature set and 93.56% based on the full feature set. Figure 6(b) $\sim$ 6(d) shows the significance of our classifier in terms of precision, recall and f-measure respectively using optimal feature set. Only 6 usage activities are tested for Whatsapp APP and our proposed classifier achieves the best performance on each single specific usage activity. Among all the 6 service usage classes, the test on Voice Call (U2) has the best performance since it is the only VoIP usage activity in our testing class.



(a) Accuracy (b) Precision (c) Recall (d) F-Measure (e) TDA-TVA

Figure 4: Performance Comparison of Wechat

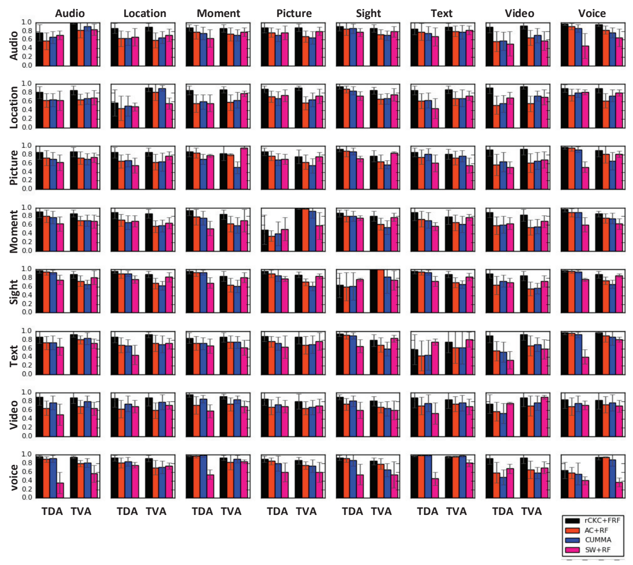


Figure 5: Performance Test on 2-activity traffic flows.

Similar to the Wechat performance test, we test our proposed analyzer on the 2-activity Whatsapp traffic flows. The analyzer performance comparison in Figure 6(e) illustrates that our analyzer improves the accuracy with an overall TDA of 0.9076 and an overall TVA of 0.966.

Results on Facebook. Figure 7 presents the offline test of our proposed classifier on Facebook Internet traffic flows. Different from the messaging APPs, there is no VoIP traffic flow tested on the Facebook APP and our HRF classifier is reduced to the traditional RF classifier. The overall accuracy of the RF classifier achieves the best overall accuracy of 92.48% among all other baselines based the optimal feature set and 95.87% based on the full feature set. The analyzer also works better on Facebook traffic flows with a high TDA(0.9147) and TVA(0.9654) present in Figure 7(e).

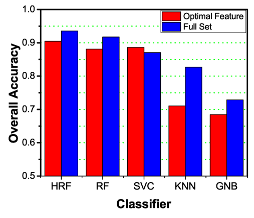
5.5 Online Performance Test

After we train and test our mobile traffic analyzer off-line with simple cases, we implement it online and test a continuous traffic flow of 2126 seconds during which the mobile user keeps switching service usage activities. The incoming traffic with labeled segment is shown in Figure 8(a). The average processing speed (number of packets processed per second) of different traffic analyzers is present in Figure 8(b). As can be seen, our designed analyzer is achieves an average processing speed of 8840 packets per second which is 14.3 times faster than the most competitive baseline, 7.14 times faster than CUMMA. Moreover, by choosing the optimal feature set, the processing speed increase almost 4 times for all analyzers. Figure 8(c) compares the TDA and TVA of different analyzers using the optimal feature set. Our proposed analyzer achieves a TDA of 86.82% and a TVA of 91.21%, which outperforms other baselines with significant margin. Due to the simplicity and local search strategy of SW+RF Internet traffic analyzer, it achieves a faster processing speed (1.94 times faster than our proposed method) but the overall TDA and TVA are only 35.42% and 53.96%.

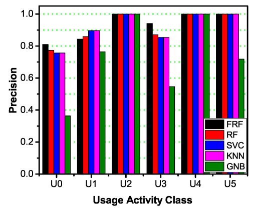
6 RELATED WORK

Below we describe some related studies that have been accomplished on Internet traffic flow analysis.

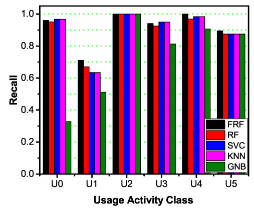
Internet Traffic Segmentation. The mobile Internet traffic flow segmentation is basically a time series segmentation. In the past, several high level time series segmentation algorithms have been proposed, including Fourier Transformations [10], Wavelets [3], Symbolic Mappings [20], hierarchical bottom to up [13], top-down approach [15] and the most frequently used Piecewise Linear Representation (PLR) [14, 19]. These algorithms, based on time series element similarities, are supposed to support fast and dynamic implementation [12]. However, for the problem of Internet Streaming traffic flow segmentation, these algorithms are not applicable due to the large variance of the adjacent packet length and high traffic densities. To reduce the effect of packet level variance, a number of data mining techniques are proposed to extract time series patterns from a subset of traffic flows (time window representation) and calculate the traffic similarities based on the subflows. The most efficient one is the sliding window algorithm [4] that attempts to approximate the incoming subflows with similar traffic patterns. This algorithm works fast and can reduce the effect of packet length variances, however, as a local similarity search based algorithm, it suffers from the stability of Internet traffic flows. Most recently, a group of data mining researchers [1, 6, 16, 17, 23, 24] segment time series with clustering based algorithms in different time



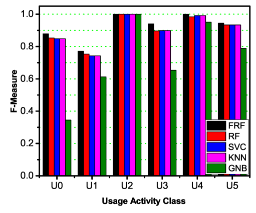
(a) Overall



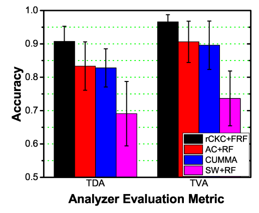
(b) Precision



(c) Recall

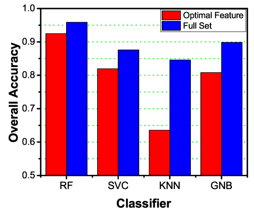


(d) F-Measure

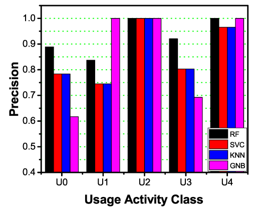


(e) TDA-TVA

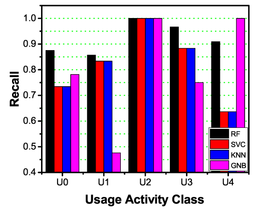
Figure 6: Performance Comparison of Whatsapp



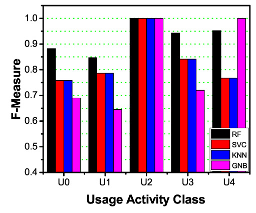
(a) Overall



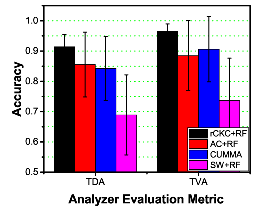
(b) Precision



(c) Recall

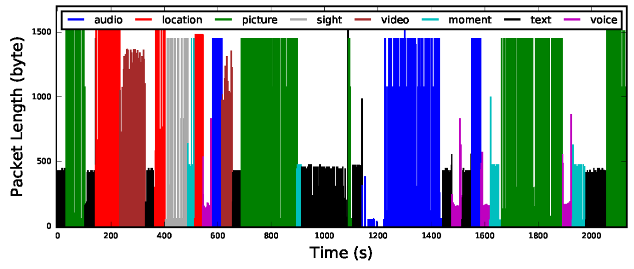


(d) F-Measure

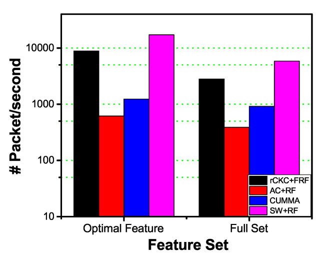


(e) TDA-TVA

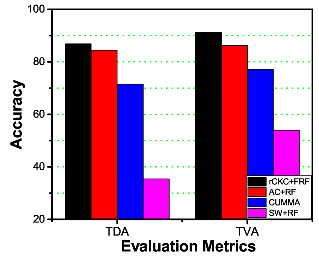
Figure 7: Performance Comparison of Facebook



(a) Ground Truth Traffic Flow



(b) Efficiency



(c) Accuracy

Figure 8: Online Efficiency and Effectiveness Test

series analysis problems, however, most of these clustering based segmentation algorithms require a pre-specified number of segments and a set of complex high dimension feature vectors that can hardly be calculated iteratively as online time series segmentation algorithm. We further improve the time series segmentation algorithm with our rCKC algorithm that constructs time windows iteratively and finds the optimal number of segment recursively.

Mobile Service Activity Classification. Our work has a close connection with mobile service classification [27]. Falaki, etc. in [5] proposed a custom logging tool so as to enhance the classification of mobile App service usages. The authors also observed that the behaviors of people using mobile Apps are substantially diverse. Xu, etc. [28] collected the networking traces at anonymized IP-level from a large tier-1 cellular network and classified the traffic from different Apps by exploring signatures from HTTP headers. A novel approach that exploits the cross-layer interaction among various network layers for classifying usage of mobile Apps was developed by Qian, etc [21]. The work in [26] used the advertising traffic originating from the Apps as usage patterns for the classification of mobile service activities. The performances of these classifiers are limited due to the big variance of the traffic patterns extracted at packet level. Finally, our previous work in [7] proposed a step-by-step analytic method including traffic clustering, traffic classification using existing classifiers which provide a promising mobile service usage classification accuracy. However, since this classifier relies on a complex set of traffic patterns, the processing speed of this classifier is too low to meet the online efficiency requirement.

7 CONCLUSION

In this paper, we developed an online mobile app traffic analyzer for classifying mobile app traffic into different types of service usages. Specifically, we first proposed an optimal set of packet-level iterable features to reduce the cache memory requirement and to improve the processing speed. The received traffic packets from incoming traffic flows are then transferred into a series of time windows represented by the feature vectors, which is selected according to our MIMD criteria. Then, with the consideration of time continuity constraints, the time windows are clustered together with our rCKC clustering algorithm, by which the time windows generated from the same service activity traffic flow are grouped together. In this way, the service usage activities are identified for each segmented traffic flow. Finally, we provided the extensive experiments on real-world traffic flow data from Wechat, Whatsapp, and Facebook. The results show that the proposed method has the advantages of low storage cache requirement and fast processing speed. Also, our method can classify mobile app traffic with high accuracy in a real-time manner.

8 ACKNOWLEDGMENTS

This research was supported in part by the Natural Science Foundation of China (71329201) and Futurewei Technologies, Inc.

REFERENCES

[1] Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. 2015. Time-series clustering–A decade review. *Information Systems* 53 (2015), 16–38.

[2] Tengfei Bao, Huanhuan Cao, Enhong Chen, Jilei Tian, and Hui Xiong. 2012. An unsupervised approach to modeling personalized contexts of mobile users. *Knowledge and Information Systems* 31, 2 (2012), 345–370. https://doi.org/10.1007/s10115-011-0417-1

[3] Kin-Pong Chan and Ada Wai-Chee Fu. 1999. Efficient time series matching by wavelets. In *Data Engineering, 1999. Proceedings., 15th International Conference on.* IEEE, 126–133.

[4] Tak chung Fu. 2011. A review on time series data mining. *Engineering Applications of Artificial Intelligence* 24, 1 (2011), 164 – 181. https://doi.org/10.1016/j.engappai.2010.09.007

[5] Hossein Falaki, Ratul Mahajan, Srikanth Kandula, Dimitrios Lymberopoulos, Ramesh Govindan, and Deborah Estrin. 2010. Diversity in smartphone usage. In *Proceedings of the 8th international conference on Mobile systems, applications, and services.* ACM, 179–194.

[6] Y. Fu, J. Liu, X. Li, X. Lu, J. Ming, C. Guan, and H. Xiong. 2016. Service Usage Analysis in Mobile Messaging Apps: A Multi-label Multi-view Perspective. In *2016 IEEE 16th International Conference on Data Mining (ICDM).* 877–882. https://doi.org/10.1109/ICDM.2016.0106

[7] Y. Fu, H. Xiong, X. Lu, J. Yang, and C. Chen. 2016. Service Usage Classification with Encrypted Internet Traffic in Mobile Messaging Apps. *IEEE Transactions on Mobile Computing* 15, 11 (Nov 2016), 2851–2864. https://doi.org/10.1109/TMC.2016.2516020

[8] Mikel Galar, Alberto Fernández, Edurne Barrenechea, Humberto Bustince, and Francisco Herrera. 2011. An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. *Pattern Recognition* 44, 8 (2011), 1761 – 1776. https://doi.org/10.1016/j.patcog.2011.01.017

[9] Patrick Haffner, Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang. 2005. ACAS: Automated Construction of Application Signatures. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Mining Network Data (MineNet '05).* ACM, New York, NY, USA, 197–202. https://doi.org/10.1145/1080173.1080183

[10] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. 2001. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems* 3, 3 (2001), 263–286.

[11] E. Keogh, S. Chu, D. Hart, and M. Pazzani. 2001. An online algorithm for segmenting time series. In *Proceedings 2001 IEEE International Conference on Data Mining.* 289–296. https://doi.org/10.1109/ICDM.2001.989531

[12] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. 2001. An online algorithm for segmenting time series. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on.* IEEE, 289–296.

[13] Eamonn J Keogh and Michael J Pazzani. 1998. An Enhanced Representation of Time Series Which Allows Fast and Accurate Classification, Clustering and Relevance Feedback.. In *KDD*, Vol. 98. 239–243.

[14] Victor Lavrenko, Matt Schmill, Dawn Lawrie, Paul Ogilvie, David Jensen, and James Allan. 2000. Mining of concurrent text and time series. In *KDD-2000 Workshop.* 37–44.

[15] Chung-Sheng Li, Philip S Yu, and Vittorio Castelli. 1998. MALM: a framework for mining sequence database at multiple abstraction levels. In *Proceedings of the 7th CIKM.*

[16] T Warren Liao. 2005. Clustering of time series data?a survey. *Pattern recognition* 38, 11 (2005), 1857–1874.

[17] T. Warren Liao. 2005. Clustering of time series dataąła survey. *Pattern Recognition* 38, 11 (2005), 1857 – 1874. https://doi.org/10.1016/j.patcog.2005.01.025

[18] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu. 2010. Understanding of Internal Clustering Validation Measures. In *2010 IEEE International Conference on Data Mining.* 911–916. https://doi.org/10.1109/ICDM.2010.35

[19] Sanghyun Park, Sang-Wook Kim, and Wesley W Chu. 2001. Segment-based approach for subsequence searches in sequence databases. In *Proceedings of the 2001 ACM symposium on Applied computing.* ACM, 248–252.

[20] C-S Perng, Haixun Wang, Sylvia R Zhang, and D Stott Parker. 2000. Landmarks: a new model for similarity-based pattern querying in time series databases. In *Data Engineering, 2000. Proceedings. 16th International Conference on.* IEEE, 33–42.

[21] Feng Qian, Zhaoguang Wang, Alexandre Gerber, Zhuoqing Mao, Subhabrata Sen, and Oliver Spatscheck. 2011. Profiling resource usage for mobile applications: a cross-layer approach. In *Proceedings of the 9th international conference on Mobile systems, applications, and services.* ACM, 321–334.

[22] B. Raahemi, W. Zhong, and J. Liu. 2008. Peer-to-Peer Traffic Identification by Mining IP Layer Data Streams Using Concept-Adapting Very Fast Decision Tree. In *2008 20th IEEE International Conference on Tools with Artificial Intelligence*, Vol. 1. 525–532. https://doi.org/10.1109/ICTAI.2008.12

[23] Stan Salvador and Philip Chan. 2004. Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms. In *Tools with Artificial Intelligence, 2004. ICTAI 2004. 16th IEEE International Conference on.* IEEE, 576–584.

[24] Allou Samé, Faicel Chamroukhi, Gérard Govaert, and Patrice Aknin. 2011. Model-based clustering and segmentation of time series with changes in regime. *Advances in Data Analysis and Classification* 5, 4 (2011), 301–321.

[25] Pang-Ning Tan et al. 2006. *Introduction to data mining.* Pearson Education India.

[26] Alok Tongaonkar, Shuaifu Dai, Antonio Nucci, and Dawn Song. 2013. Understanding mobile app usage patterns using in-app advertisements. In *Passive and Active Measurement.* 63–72.

[27] Hannu Verkasalo. 2009. Contextual patterns in mobile service usage. *Personal and Ubiquitous Computing* 13, 5 (2009), 331–342.

[28] Qiang Xu, Jeffrey Erman, Alexandre Gerber, Zhuoqing Mao, Jeffrey Pang, and Shobha Venkataraman. 2011. Identifying diverse usage behaviors of smartphone apps. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference.* ACM, 329–344.

[29] S. Zander, T. Nguyen, and G. Armitage. 2005. Automated traffic classification and application identification using machine learning. In *The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)l.* 250–257. https://doi.org/10.1109/LCN.2005.35

[30] Ying Zhao and George Karypis. 2002. Evaluation of Hierarchical Clustering Algorithms for Document Datasets. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management (CIKM '02).* ACM, New York, NY, USA, 515–524. https://doi.org/10.1145/584792.584877