When Smartphones become the Enemy: Unveiling Mobile Apps Anomalies through Clustering Techniques

Pedro Casas (1), Pierdomenico Fiadino (2), Alessandro D'Alconzo (1)
(1) AIT Austrian Institute of Technology, (2) Eurecat Technology Centre of Catalonia
(1) pedro.casas@ait.ac.at, (2) pierdomenico.fiadino@eurecat.org

ABSTRACT

The ever-increasing number of mobile devices connected to cellular networks is heavily modifying the traffic observed in these networks. The traffic volumes and patterns generated by smartphones pose novel challenges to cellular network operators. One of these challenges relates to the automatic detection and diagnosis of unforeseen network traffic anomalies caused by specific devices and apps. Synchronized apps generating flashcrowds, device-specific traffic misbehaviors impacting network performance and end-users Quality of Experience (QoE), and other similar anomalies need to be rapidly detected and diagnosed. In this paper we characterize a new type of anomalies impacting cellular networks, caused by the multiple, constantly-connected apps running in smartphones and other end-user devices. We additionally devise a novel detection and classification technique based on semi-supervised Machine Learning (ML) algorithms to automatically detect and diagnose anomalies of this class with minimal training, and compare its performance to that achieved by other well-known supervised learning classifiers. The proposed solution is evaluated using synthetically generated data from an operational cellular ISP, drawn from real traffic statistics to resemble both the real cellular network traffic and the characterized type of anomalies.

CCS Concepts

•**Networks → Network measurement; Network monitoring;** *Network performance analysis;*

Keywords

Anomaly Detection and Classification; Network Measurements; Machine Learning; DNS Traffic; Cellular ISP.

The research leading to these results has been partially funded by the Vienna Science and Technology Fund (WWTF) through project ICT15-129, "BigDAMA".

AllThingsCellular'16, October 03-07, 2016, New York City, NY, USA
© 2016 ACM. ISBN 978-1-4503-4249-0/16/10. . . $15.00
DOI: http://dx.doi.org/10.1145/2980055.2980058

1. INTRODUCTION

During the last decade, a plethora of new, heterogeneous Internet services have become highly popular, imposing new challenges to network operators. In particular, cellular network operators have witnessed an astonishing increase of heterogeneous mobile devices (smartphones, tablets, M2M devices such as telemeters, etc.) running such services. The applications supported by these devices introduce new traffic patterns which are potentially harmful to the network. For example, due to their traffic characteristics, applications that provide continuous online presence (e.g., messaging services) might severely impact the signaling plane of the network, especially in cellular networks [3]. In such a complex scenario, it is of vital importance to promptly detect and classify the occurrence of abrupt changes that could result in anomalies for some of the involved stakeholders.

In this paper we propose a simple yet effective approach to detect and classify network and traffic anomalies using Machine Learning (ML) techniques. The literature offers multiple types of ML-based classifiers, covering a very wide range of approaches and techniques [21]. Our work brings two main contributions: firstly, we present a semi-supervised approach for detecting and diagnosing anomalous traffic patterns linked to different device classes and applications, based on clustering techniques; secondly, we analyze one of these anomalies in which an Apple service outage results in iPhone and other Apple devices flooding the network with connection attempts, providing tangible evidence of the potential harms introduced in this new apps context.

While the approach we consider is not tied to a specific type of network and can be generalized to any kind of communication system, results presented in this paper consider the analysis of data captured in an operational cellular network, and therefore use some features exclusively available in cellular contexts. From our operational experience and our previous studies [2], app-specific anomalies are particularly visible in the DNS traffic of a network, as most of current apps and services distributed by omnipresent Content Delivery Networks (CDNs) extensively rely on a heavy usage of DNS for content access and location. As so, abrupt changes in the DNS query count can be considered as a symptom of such anomalies. Besides DNS query counts, our approach relies on the availability of related *meta-data*, which can also be observed in the analyzed cellular ISP. These meta-data may include information related to the end-device (e.g., device manufacturer, Operative System), the access network (e.g., Radio Access Technology – RAT, Access Point Name – APN, IP address of the DNS resolver), and the requested service (e.g., requested Fully Qualified Domain Name – FQDN).

The remainder of this paper is organized as follows: Sec. 2 briefly reviews the related work. In Sec. 3 we overview and analyze the occurrence of an app-generated anomaly in an operational cellular ISP. Sec. 4 describes the proposed clustering-based anomaly detector and classifier, and briefly overviews the underlying concepts. Sec. 5 presents the characterization of the cellular traffic and the generation of synthetic datasets used for evaluation purposes. In Sec. 6 we discuss the obtained results, considering both the detection and classification of anomalies, additionally comparing the achieved performance to that of other ML-based systems proposed in the literature. Sec. 7 concludes this work.

2. RELATED WORK

There has been considerable amount of research on anomaly detection in network traffic. A large set of papers apply concepts and techniques imported from fields like Data Mining [10] and Machine Learning [11]. Focusing on statistical-based methods, most work rely on the analysis of scalar time-series, typically of total volume. They adopt various techniques like Discrete Wavelet Transform [12], CUSUM [13] and others. It is commonly accepted that information-theoretic concepts, and in particular entropy measures, are well-suited for anomaly detection [4, 5]. Distribution-based approaches such as [7] are intrinsically more powerful, as they look at the entire distribution, rather than only at some specific mode or aggregation. A comprehensive survey on multiple anomaly detection techniques applied to different fields beyond network communications is available in [9]. In terms of ML-based approaches for classifying anomalies, the field of automatic traffic analysis and classification trough ML techniques has been extensively studied during the last half-decade. A standard non-exhaustive list of supervised ML-based approaches includes the use of Bayesian classifiers [14], linear discriminant analysis and k-nearest-neighbors [15], decision trees and feature selection techniques [16], and support vector machines [17]. Unsupervised and semi-supervised learning techniques have also been used before for traffic analysis and classification, including the use of k-means, DBSCAN, and AutoClass clustering [18], sub-space clustering techniques [20, 22], and a combination of k-means and maximum-likelihood clusters labeling [19]. Closely linked to this work, we have recently addressed the same detection and classification problem using supervised ML based techniques [23]. The main differences of current paper are on both the application of semi-supervised, clustering-based techniques, as well as the usage of a smaller, non-bootstrapping based set of input features. We refer the interested reader to [21] for a detailed survey on the different ML techniques applied to automatic traffic classification.

3. APP-GENERATED ANOMALIES

We start by the analysis of a real case study based on the diagnosis of a large scale anomaly observed in an operational cellular network. Fig. 1 shows the time series of the total DNS query count observed in the network for two consecutive days. Two anomalous spikes are observed between 9:00 and 11:00 of the second day. Note that the scale of the anomaly is highly significant, suggesting a potentially major impact in the network. During the anomaly we also observe an increase of DNS query timeouts, pointing to a degradation in the performance of the DNS servers due to the steep increase of requests. Fig. 2 depicts the same information as before, but discriminating devices by OS type - note that we just consider the two most popular OS types, namely iOS and Android. Interestingly, while Android devices do not cause any abnormal increase in the number of DNS queries as compared to normal operation (day before), iOS devices seem to be the ones causing the aforementioned spikes, pointing to an anomaly potentially linked to iOS only.

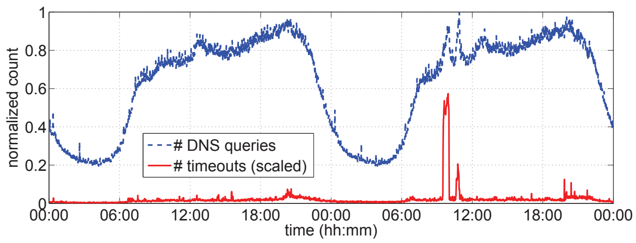


Figure 1: DNS query counts and timeouts during an Apple-related anomaly.

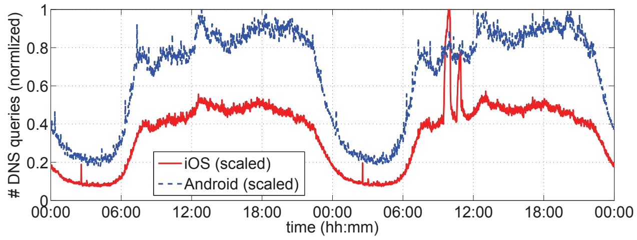


Figure 2: DNS query counts per device OS.

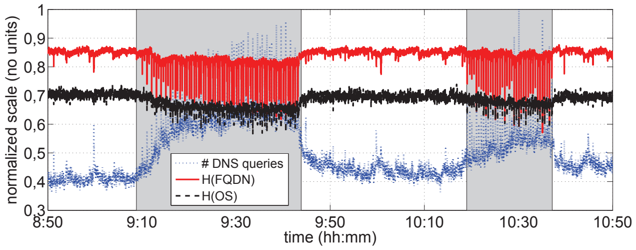


Figure 3: Entropy of features revealing an anomaly linked to OS and specific requested FQDNs.

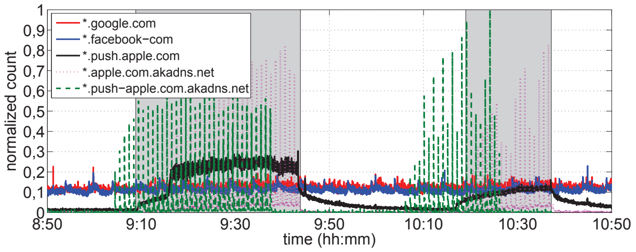


Figure 4: FQDNs requested by the affected devices.

Fig. 3 provides a closer look into the anomaly, comparing the time series of the total DNS requests count and the

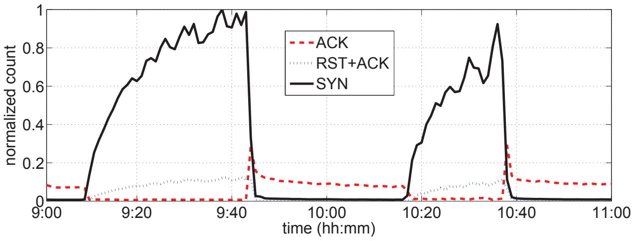


Figure 5: TCP flags during the anomaly.

entropy of two selected features: FQDN and OS. We use the empirical entropy of these features, namely $H(\text{FQDN})$ and $H(\text{OS})$ as a better means to visualize changes in the distribution of such features during the anomaly [4]. Both features present a very high correlation with the spikes in the DNS count, suggesting again that the issue might be due to specific OS - iOS in this case, querying for certain services - i.e., FQDNs. Fig. 4 reports the mostly requested FQDNs during the anomaly. While some of the top FQDNs present a stable behavior - **.facebook.com* and **.google.com*, the FQDNs **.apple.com.akadns.net*, **.push-apple.com.akadns.net*, and **.push.apple.com* show a significant increase, pointing to a problem in the push notification service used by Apple, and hosted by the Akamai CDN. This service is the one providing persistent connectivity for iOS devices. The conclusion obtained from this analysis is that, for some unknown reason, iOS devices lost connectivity to the servers providing the push notification service, resulting in a heavy increase of DNS queries to locate new servers, and the resulting "scanning" of the complete IP address space of the Apple push-notification service.

We also evaluate the effect of this anomaly on the lower transport layer. The # of SYN, ACK, and reset (RST+ACK) packets generated inbound and outbound for the anomalous traffic subset is depicted in fig. 5. To an increasing amount of connection attempts by the impacted devices (increase in the SYN count), servers refuse/reset the connection, implying that the requested service is temporary unavailable. During the anomaly, the amount of ACK packets drops to zero, meaning that none of the devices was able to establish a connection. The scheduling strategy used in cellular networks at the Radio Access Network (RAN) assigns higher bandwidth to users exceeding a certain traffic rate. Therefore, the high amount of connection attempts can directly result in a waste of network resources at the RAN, which could even impact the performance of those users not responsible for this specific anomaly. In this perspective, the Apple-related anomaly looks very similar to an internal DDoS attack, as a large population of the customer devices started "bombarding" the network, starving resources at the access in some specific regions. Next, we devise an approach to automatically detect and classify this type of anomalies, using clustering techniques.

4. CLUSTERING-BASED AD

The clustering-based approach introduced in this work, referred to as K-CDA (K-means Clustering-based Detector of Anomalies) has the main advantage of being semi-supervised, which means that the amount of learning data which is required for training/calibration purposes if significantly less than that required by supervised approaches. Given a training set of m measurements consisting each of n features, our method uses first the well known K-means clustering algorithm [6] to partition the complete feature space $X \in \mathbb{R}^{m \times n}$ in a set of K clusters. The centroid of each of these K clusters is then computed, and a label is assigned to each of them, based on majority voting performed on a small sample of ground truth labels among the measurements belonging to each cluster. In particular, we decide the class y of a cluster based on the real label of only 5% of the samples within this cluster, randomly sampled. We have verified that this small fraction is good enough to provide proper detection and classification results. Once all clusters have been labeled, the standard approach to follow for clustering-based classification is to assign, to every new sample, the class of the cluster with the closest centroid [20]. However, given that the real number of clusters K is not known in advance, following such an approach might be counterproductive and lead to less robust results [20].

Indeed, one well-known limitation of using K-means as clustering algorithm is that one needs to define in advance the number of K clusters to identify, which in principle is completely unknown, especially when no labeled data is used. Selecting a small value for K results in bigger and potentially less homogeneous clusters; having a big number of clusters has the advantage of resulting in more homogeneous clusters, but if this number is too big, the analysis and interpretation of results becomes more cumbersome and the advantages of grouping samples together is partially lost. To partially solve this issue, we resort to a simple heuristic, based once again on a majority voting approach. We set K to a value equal to 0.1% of the training sample size, i.e., $K = m/1000$, and then decide on the label of a new sample using a k-NN (Nearest Neighbors) algorithm, computing the distance of the new sample to all the K centroids, and using majority voting on the labels of the k closest centroids. By doing so, we obtain more homogeneous clusters, and limit the impacts of single centroid-based classification. The value of k clearly depends on the value of K: based on empirical testing, we set $k = K/3$ (naturally, all values are rounded to obtain integer numbers for both k and K).

The final ingredient of our approach is on the particular way we compute distances: instead of using a simple Euclidean distance, we compute the per-cluster normalized Mahalanobis distance between every new sample and the K labeled centroids. The Mahalanobis distance takes into account the correlation between samples, dividing the standard Euclidean distance by the variance of the samples belonging to each cluster. In a nutshell, if a cluster has a bigger variance on a certain direction (i.e., feature), then the Mahalanobis distance will make samples closer to this cluster than to other ones with smaller variance, making less compact clusters closer to samples.

In this paper we take as main traffic feature the total number of DNS requests issued within a time bin. As we saw in Sec. 3, perturbations in this feature indicate that a device sub-population deviates from the usual DNS traffic patterns, thus pointing to potential anomalies. For the sake of better detecting and diagnosing the anomalies, we additionally take the distributions of DNS query counts across the fields described in Sec. 1. From these distributions, we compute a set of features describing their shape and carried information, such as various percentiles and entropy values. Tab. 1 describes the specific set of $n = 36$ features, which

Table 1: Input features for K-CDA.

Field	Feature	Description
DNS_query	querycnt	# DNS requests
APN	apn_h	H(APN)
	apn_avg	$\overline{\text{APN}}$
	apn_p{99,75,50,25,05}	percentiles
Error_flag	error_code_h	H(Error_flag)
	error_code_avg	$\overline{\text{Error\_flag}}$
	error_code_p{99,75,50,25,05}	percentiles
Manufacturer	manufacturer_h	H(Manufacturer)
	manufacturer_avg	$\overline{\text{Manufacturer}}$
	manufacturer_p{99,75,50,25,05}	percentiles
OS	os_h	H(OS)
	os_avg	$\overline{\text{OS}}$
	os_p{99,75,50,25,05}	percentiles
FQDN	req_fqdn_h	H(FQDN)
	req_fqdn_avg	$\overline{\text{FQDN}}$
	req_fqdn_p{99,75,50,25,05}	percentiles

are computed for every time bin. The set includes the number of observed DNS requests, as well as multiple percentiles of fields such as associated APN, device OS and manufacturer, requested FQDN and number of DNS error messages. We also take as input the average values of these fields, as well as their entropy, the latter reflecting the dispersion of the observed samples in the corresponding time bin. As we explain in Sec. 5, the training of the K-CDA algorithm is done on top of synthetically generated datasets, which are by default labeled datasets.

5. ANOMALY GENERATION

We have evaluated different anomaly detectors for longer than six months in 2014 with DNS traffic from the operational cellular network of a nationwide European operator. While the extensive experimentation allowed us to collect results in a number of paradigmatic case-studies, the number of traffic anomalies observed in the corresponding period was relatively low, limiting as such the performance analysis of the proposed approach exclusively to those few real cases. In principle, one could resort to test traces obtained in a controlled environment (laboratory) or by simulations, but these approaches would miss the complexity and heterogeneity of the real traffic. To bypass this hurdle, we adopted a methodology based on semi-synthetic data, derived from real traffic traces as suggested in [8]. Such an approach does not only allow to extensively analyze the performance of the proposed solution with a large number of synthetic, yet statistically relevant anomalies, but also permits to protect the operator's business sensitive information, as neither real data traces nor real anomalies are exposed. We do not present the details on how to generate semi-synthetic background DNS traffic due to space limitation, but point the interested reader to [1] for further details.

During these months we have encountered a few recurring large-scale DNS traffic anomalies. Investigating these events we found some common traits and conceived a procedure for reproducing them along with their most relevant characteristics. In particular, we identified two exemplary event types, E_1 and E_2 from now on. In both cases, we model an outage of an Internet service for a specific sub-population of devices, which react by repeatedly and constantly issuing DNS queries to resolve the requested service throughout the anomaly, exactly as presented in Sec. 3. Involved devices are identified by fixing a specific OS (with its different versions). Moreover, we aim at modeling the correlation between the selected sub-population and the unreachable service. Therefore, we separately rank the 2nd-Level Domain (2LD) of the FQDNs for anomalous and background traffic, and select the most popular 2LD of the former that is not in the latter.

Table 2: Anomalous DNS traffic features.

Type	E_1	E_2	E_3
Start time t_1	9:00	13:00	18:00
Duration d	2h	1 day	1h
Involved devices D	10%	5%	3%
Back-off time	5 sec	180 sec	20 sec
Manufacturer	single popular	multiple	multiple
OS	single	single	multiple
Error flag	+5% timeout	—	—
FQDN	top-2LD	top-2LD	top-2LD

Event E_1. This type models the case of a short lived (i.e., hours) high intensity anomaly (e.g., 10% of devices repeating a request every few seconds), where all the involved devices are produced by a single manufacturer and run the same OS. In this case, the number of involved terminals and the overall number of additional queries is such to overload the local DNS servers. The latter effect is modeled by increasing the number of `time-out` codes in the Error Flag field.

Event E_2. This type models a long lasting (i.e., days) low-intensity anomaly (e.g., 5% of devices repeating requests every few minutes). Differently from the previous case, the involved terminals are produced by multiple manufacturers, even if they share the same OS. Given the low-intensity, we do not introduce a modification in the distribution of the Error Flag. E_2 anomalies are of relatively low intensity, thus tend to be more difficult to detect than those of type E_1.

Event E_3. We additionally introduce a third class of anomalies type E_3 which models a scenario in which all the customers of certain virtual operators (reflected by specific APNs) are affected by short lived service outages, responding with a surge in the number of DNS queries.

Tab. 2 summarizes the characteristics of the three event types and the actual values used for generating the anomalous dataset in the experiments discussed next. To illustrate the anomaly generation procedure, we consider an event of type E_1 of duration $d = 2h$, starting at $t_1 = 9:00$, following the example presented in Sec. 3. Starting from t_1 at each time bin, $D = 10\%$ of all the active terminals are randomly extracted from the semi-synthetic background traffic, such that the OS is the selected one and the manufacturer is always the same. For each involved terminal, we generate one additional DNS query every 5 seconds, which are then added to the semi-synthetic dataset. The corresponding FQDN is randomly chosen among the domains in the 2LD identified as explained above. Finally, the Error Flag is changed to `time-out` in 5% of the overall DNS queries, so as to model the resolver overload. The last step consists of mangling both the anomalous and the background traffic. The procedure for generating types E_2 and E_3 is analogous, but differs in the selection of the specifically impacted features.

6. K-CDA PERFORMANCE

In this section we assess the proposed K-CDA approach, evaluating both its detection and classification performance, additionally comparing it against other fully supervised, ML-based approaches. For evaluation purposes, we construct a fully labeled dataset consisting of a full month of synthetically generated cellular network DNS measurements, reported with a time granularity of 5 minutes. The dataset contains normal operation traffic, with multiple instances of the aforementioned E_1, E_2 and E_3 anomalies, as specified in Tab. 2. To perform a better evaluation, we introduce multiple instances of each anomaly type with a different fraction of the device population involved in the anomaly. In total we include 16 different variations of these anomalies, added on top of the 1-month anomaly-free traffic. For each of the anomaly types E_1 and E_2 we include 7 anomaly variations, involving a number of devices going from 0.5% to 20% of the overall population (0.5%, 1%, 2%, 5%, 8%, 10%, 20%). For E_3 we include two different intensities, considering a population of 3% and 12%, which correspond to the actual size of virtual-operator customer populations, as observed from our real measurements. Each time bin is assigned a class, either normal - label 0, or anomalous - label 1, 2 or 3 for the three anomaly types respectively.

6.1 Detection Performance

Let us first get an initial picture of the detection capabilities of the K-CDA approach, by testing the detection and false alarm rates on the complete set of 16 variations of anomalies. To reduce biased results, the training and testing of the K-CDA approach is performed by 10-fold cross validation. Detection performance is evaluated in a time bin basis and not in an event basis: this means that there are 24 anomalous time slots for each anomaly variation of type E_1 (i.e., $7 \times 24 = 168$ time slots in total), $12 \times 24 = 288$ time slots for each variation of type E_2 (i.e., $7 \times 288 = 2016$ in total) and 12 time slots for each variation of type E_3 (i.e., $2 \times 12 = 24$ time slots in total). We follow such a direction as we are not only interested in detecting the occurrence of an event, but also its full span/duration. Note that anomaly classes are highly imbalanced, which in principle imposes major challenges in the training phase. To counterbalance this problem, we resort to a standard oversampling approach, in which we add copies of instances from the under-represented classes (E_1 and E_3) to perform the training.

Fig. 6 depicts the Receiver Operating Characteristic (ROC) curves obtained for the detection of the complete set of anomalies. Fig. 6(a) provides the results obtained for the K-CDA approach, whereas Figs. 6(b-d) show the comparative results obtained for three, well-known supervised based detectors using Neural Networks (MLP), Support Vector Machines (SVM), and C4.5 Decision Trees (C4.5). We selected these other detectors for comparison based on the a-priori good performance shown by their application in previous work on anomaly detection [9] and traffic classification [21]. We use the well-known Weka Machine-Learning software tool to calibrate these ML-based algorithms and to perform the evaluations. We address the interested reader to the survey [21] and to the Weka doc. for additional information on the configuration parameters of each algorithm.

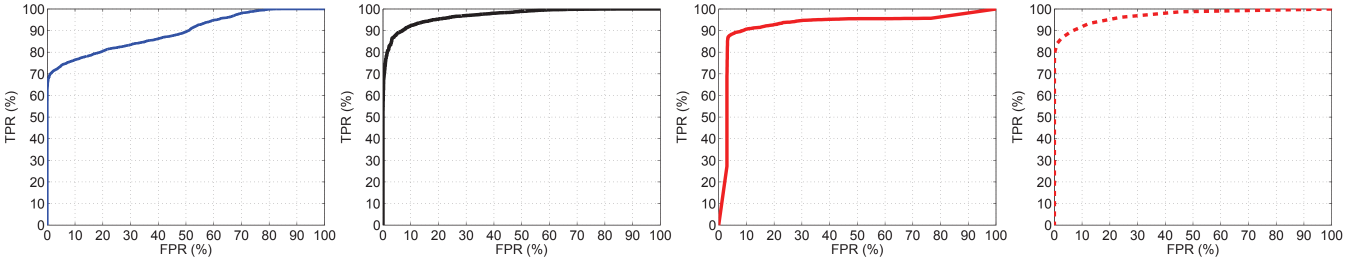
The K-CDA approach can correctly detect around 70% of the anomaly instances with different intensities without false alarms, but it is not capable to properly detect part of the smallest intensity anomalies. The SVM model achieves slightly worse detection performance, resulting in a similar true positives rate but a false alarm rate above 3%. Both the MLP and the C4.5 models achieve better performance, detecting around 80% of the anomalies without false alarms, but they also fail to detect the smallest intensity ones. As a first conclusion, we can claim that the K-CDA detection performance is comparable to that achieved by the supervised models, but using only 5% of labeled samples for training purposes, which is a major advantage.

6.2 Classification Performance

We move on now to the evaluation of the anomaly classification capabilities of the K-CDA approach. We additionally evaluate the three aforementioned ML-based classifiers. To evaluate and compare the performance and virtues of the classification models, we consider three standard metrics: Global Accuracy GA, Recall and Precision. GA indicates the percentage of correctly classified instances (time bins) among the total number of instances. Recall R_i is the number of instances from class $i = 0, \ldots, 3$ correctly classified (TP_i), divided by the number of instances in class i (n_i). Precision P_i is the percentage of instances correctly classified as belonging to class i among all the instances classified as belonging to class i, including true and false positives (FP_i). Recall and precision are two widely used performance metrics in classification. Precision permits to measure the fidelity of the classification model regarding each particular class, whereas recall measures the per-class accuracy.

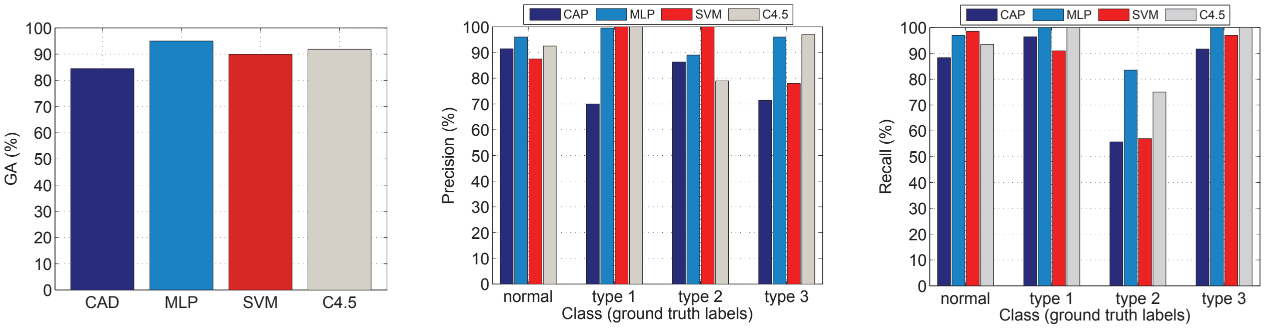
$$\mathrm{R}_i = \frac{TP_i}{n_i}, \quad \mathrm{P}_i = \frac{TP_i}{TP_i + FP_i}, \quad \mathrm{GA} = \frac{\sum_{i=1}^{M} TP_i}{n} \tag{1}$$

Fig. 7 reports the performance of the four compared classifiers in the classification of all the 5-minutes time bins. To limit biased results, all the evaluations presented use 10-fold cross-validation. Reported results refer to optimal parameter settings, after thorough testing. According to Fig. 7(a), MLP, SVM and C4.5 achieve high overall classification accuracy, above 90% in the three cases, and with a slightly better performance for the MLP model. The K-CDA approach achieves slightly worse results, but in any case reaches almost a 85% of accuracy, quite close to the other models. In terms of precision and recall, depicted in Figs. 7(b) and 7(c) respectively, the MLP model outperforms the other classifiers in all the classes, evidencing the nice properties introduced by such a model. As already evidenced in the results presented in Fig. 6(a), the K-CDA approach performs quite similarly to the SVM and C4.5 models, and in particular, shows the same limitations to correctly detect anomalies of type E_2, achieving a recall as low as 55% for this class. The K-CDA approach also shows limitations to correctly isolate anomalies of types E_1 and E_3, showing a precision close to 70% in both cases. Due to page-length limitations we do not include the confusion matrix for each classifier, but the main source of under-performance for the K-CDA approach comes from misclassifying anomalies into another anomaly class, and not due to completely missing them. All in all, results clearly suggest that the proposed K-CDA anomaly detection and classification approach is accurate and comparable to other fully supervised approaches, and that it can diagnose the studied anomalies quite well. As future work, we shall dig deeper into feature selection approaches to improve the performance of the approach with the lowest intensity anomalies of type E_2.



(a) *K*-CDA ROC. (b) MLP ROC. (c) SVM ROC. (d) C4.5 ROC.

Figure 6: Detection performance. *K*-CDA performs similarly to fully supervised approaches.



(a) Global Accuracy. (b) Precision. (c) Recall.

Figure 7: Classification Accuracy, Precision, and Recall for normal operation instances and anomalies.

7. CONCLUSIONS

In this paper we have presented *K*-CDA, a clustering based approach for detection and classification of anomalies generated by apps, based on the analysis of passively captured network data. The approach is based on semi-supervised techniques, which has the main virtue of requiring less labeled data for calibration purposes. This is a paramount advantage in practical solutions, as labeling data is a complex, cumbersome and error-prone approach, and very difficult to follow in operational scenarios. We believe that ML-based approaches can provide high insights and visibility for daily network operations, especially in current context where traffic complexity keeps growing. Given the general lack of large-scale ground-truth datasets to test the performance of systems like ours, we have used an approach to generate semi-synthetic data, derived from real traffic traces. We believe this could help the owners of real data to make such datasets available for the research community without disclosing any privacy or business sensitive information.

8. REFERENCES

[1] P. Fiadino et al., "RCATool - A Framework for Detecting and Diagnosing Anomalies in Cellular Networks", in *ITC*, 2015.

[2] M. Schiavone et al., "Diagnosing Device-Specific Anomalies in Cellular Networks", in *CoNEXT Student Workshop*, 2014.

[3] A. Aucinas et al., "Staying Online While Mobile: The Hidden Costs", in *CoNEXT*'13.

[4] G. Nychis et al., "An Empirical Evaluation of Entropy-based Traffic Anomaly Detection", in *ACM IMC*, 2008.

[5] B. Tellenbach et al., "Beyond Shannon: Characterizing Internet Traffic with Generalized Entropy Metrics", in *PAM*, 2009.

[6] A. K. Jain, "Data Clustering: 50 Years Beyond K-Means", in *Pattern Recognition Letters*, vol. 31 (8), 2010.

[7] A. D'Alconzo et al., "Distribution-based Anomaly Detection in 3G Mobile Networks: from Theory to Practice", *Int. Journal of Net. Mng.*, vol. 20 (5), 2010.

[8] H. Ringberg et al., "The need for simulation in evaluating anomaly detectors", in *ACM CCR*, vol. 38 (1), pp. 55-59, 2008.

[9] V. Chandola et al., "Anomaly Detection: a Survey", in *Com. Sur.*, vol. 41 (3), 2009.

[10] G.Prashanth, et al., "Using random forests for network-based anomaly detection", in *IEEE ICSCN*, 2008.

[11] Y. Li et al., "An efficient network anomaly detection scheme based on TCM-KNN algorithm and data reduction mechanism", in *IAW*, 2007.

[12] M. Raimondo et al., "A peaks over threshold model for change point detection by wavelets", *Statistica Sinica*, vol. 14, 2004.

[13] P. Lee, et al., "On the Detection of Signaling DoS Attacks on 3G Wireless Networks", in *IEEE INFOCOM*, 2007.

[14] A. Moore et al., "Internet Traffic Classification using Bayesian Analysis Techniques", in *Proc. ACM SIGMETICS*, 2005.

[15] M. Roughan et al., "Class-of-Service Mapping for QoS: a Statistical Signature-Based Approach to IP Traffic Classification", in *IMW*, 2004.

[16] N. Williams el al., "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification", in *CCR*'06.

[17] S. Valenti et al., "Accurate, Fine Grained Classification of P2P TV Applications by Simply Counting Packets", in *TMA*, 2009.

[18] J. Erman et al., "Traffic Classification using Clustering Algorithms", in *MineNet*'06.

[19] J. Erman et al., "Semi-Supervised Network Traffic Classification", in *Sigmetrics*'07.

[20] P. Casas et al., "MINETRAC: Mining Flows for Unsupervised Analysis & Semi-Supervised Classification", in *ITC*, 2011.

[21] T. Nguyen et al., "A Survey of Techniques for Internet Traffic Classification using Machine Learning", in *IEEE Comm, Surv. & Tut.*, vol. 10 (4), pp. 56-76, 2008.

[22] P. Casas et al., "Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge", in *Com. Comm.*, vol. 35 (7), 2011.

[23] P. Casas et al., "Machine-Learning Based Approaches for Anomaly Detection and Classification in Cellular Networks", in *TMA*, 2016.