Machine-Learning Based Approaches for Anomaly Detection and Classification in Cellular Networks

Pedro Casas*, Pierdomenico Fiadino†, Alessandro D'Alconzo*

*AIT Austrian Institute of Technology
{forename.surname}@ait.ac.at

†Eurecat Technology Centre of Catalonia
pierdomenico.fiadino@eurecat.org

***Abstract*—Despite the long literature and assorted list of proposed systems for performing detection and classification of anomalies in operational networks, Internet Service Providers (ISPs) are still looking for effective means to manage the ever-growing number of network traffic anomalies they face in their daily business. In this paper we address the problem of automatic network traffic anomaly detection and classification using Machine Learning (ML) based techniques, for the specific case of traffic anomalies observed in cellular network measurements. We devise a simple detection and classification technique based on decision tress, and compare its performance to that achieved by other supervised learning classifiers well known in the ML literature (e.g., SVM, neuronal networks, etc.). The proposed solution is evaluated using synthetically-generated data from an operational cellular ISP, drawn from real traffic statistics to resemble the real cellular network traffic. Furthermore, we compare the achieved performance against other well-known detectors in the literature (e.g., distribution-based, entropy-based), and propose a multi-detector approach to increase the overall system performance in a number of case studies.**

***Index Terms*—Anomaly Detection; Root Cause Analysis; Network Measurements; Statistical Analysis; Machine Learning; DNS Traffic; Cellular ISP.**

I. Introduction

During the last decade, a plethora of new, heterogeneous Internet-services have become highly popular and omnipresent, imposing new challenges to network operators. The complex provisioning systems used by these services induce continuous changes that impact both operators and customers. Indeed, efficient traffic engineering becomes a moving target for the operator [1], and management of Quality of Experience (QoE) gets more cumbersome, potentially affecting the end customer [2]. Furthermore, 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 [4]. 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 supervised Machine Learning (ML) techniques. Supervised ML offers algorithms which can learn from and make predictions on data, building models from labeled input data to take data-driven decisions instead of static ones. ML techniques provide a promising alternative for detecting and classifying anomalies based on an initially large set of traffic descriptors or features, specially given the possibility to perform data-driven feature selection to finally select a more relevant and powerful set of features for the detection and classification process. ML has been largely used in the field of automatic network traffic classification, and to a lesser extend also applied in the anomaly detection domain. While studies have shown that a number of different algorithms are able to achieve potentially high detection and classification accuracy, most of them have neglected the impacts of accurate feature selection to improve results and achieve more compact systems (i.e., systems which rely on less input parameters). Indeed, using a large set of traffic descriptors as input for a ML-based system is not always the best choice, as it may negatively impact classification results. For example, using more descriptors increments the dimensionality of the problem, introducing sparsity issues. At the same time, using irrelevant or redundant features may diminish performance in the practice, by adding noise to the overall process.

To achieve our goal, we propose a detection and classification system based on well-known decision trees [5]. A decision tree is a classification algorithm that classifies instances by repeatedly partitioning the input space, so as to build a tree whose nodes are as pure as possible (i.e., they contain instances of a single class). The literature offers multiple types of ML-based classifiers, covering a very wide range of approaches and techniques [28]. Many of the approaches offer "black-box" solutions, for which it becomes very challenging to understand the reasons of a particular classification result, and in particular to understand the input features leading to such a result. Decision trees are therefore a very appealing option; they are simple yet very fast and effective. Indeed, classification speed is a paramount asset when thinking in large-scale monitoring scenarios, and decision trees are well-known for their speed. They are also very easy to interpret, and directly provide filtering rules. In addition, decision trees explicitly show the importance of different features, as the learning algorithm automatically performs feature selection by choosing the most discriminating features. This is a paramount advantage as compared to other ML approaches, as decision trees are more robust to noisy or loosely correlated-to-class input features. Last but not least, previous work [21] has shown that decision trees outperform other ML-based standard algorithms for the sake of traffic classification, thus additionally supporting our decision. The C4.5 decision tree is the most frequently used algorithm [6], so we have conceived the proposed system based on such trees.

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. Anomalies observed in a cellular network are normally different from those observed in typical fixed-line networks [3], given that the type of end-user traffic is quite different due to the overwhelming usage of smartphone apps. Given that the types of anomalies we are interested in are mostly related to issues impacting the end customers of a cellular

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

network, we particularly focus on the study of application-specific anomalies. Such anomalies refer to the occurrence of anomalies linked to popular applications (e.g., YouTube, Facebook, WhatsApp, etc.) and/or application providers (e.g., Google, Facebook, Apple Services, etc.). From our operational experience and our previous studies [3], application-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 queries count can be considered as a symptom of such anomalies [3]. 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. II presents the characterization of the cellular traffic and the generation of synthetic datasets used for evaluation purposes. Sec. III describes the proposed anomaly detector and classifier, and briefly overviews the additional ML-based and statistical-based approaches used in the rest of the paper. In Sec. IV we discuss the obtained results, considering both the detection and classification of anomalies. Sec. V briefly reviews the related work. Finally, Sec. VI concludes this work. This paper builds on our previous work on statistical detection and diagnosis of anomalies [22], particularly extending the approach by using ML-based techniques, not used in [22].

II. Anomaly Modeling and Data Generation

We have evaluated different anomaly detectors (C4.5 DT, DAD and H-EWMA, see Sec. III) 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 [11]. Such an approach does not only allow to extensively analyze the performance of the framework 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. To illustrate the procedure, next we explain how to generate semi-synthetic background DNS traffic, as well as how to replicate some the observed DNS-related anomalies.

In the paper we take as main anomaly detection feature the distribution of number of devices across DNS query counts, i.e., the counting of the devices issuing a given number of DNS requests within each time bin. In fact, perturbations in this distribution indicate that a device sub-population deviates from the usual DNS traffic patterns, thus pointing to potential anomalies. For the sake of classifying the anomalies, we take the distributions of query count across the fields described in Tab. I.

Field Name	Description
Manufacturer	Device manufacturer
OS	Device operating system
APN	Access Point Name
FQDN	Fully Qualified Domain Name of remote service
Error Flag	Status of the DNS transaction

Table I
Features used in the analysis.

A. Construction of Semi-synthetic Background Traffic

The procedure for constructing the semi-synthetic dataset is conceived with the objective of maintaining as much as possible the structural characteristics of the real, normal operation (i.e., anomaly-free) traffic, while eliminating possible (unknown) anomalies present in real traces. Exploring real traces, we observed that the traffic yields some fundamental temporal characteristics. In particular, the traffic is non-stationary due to time-of-day variations. This effect is not limited to the number of active devices, but rather applies to the entire distribution. Distribution variations depend on the change of the applications and terminals mix, which in turn induce modifications in the traffic patterns. Furthermore, we found that, besides a strong 24-hours seasonality, the traffic exhibits a weekly pseudo-cycle with marked differences between working days and weekends/festivities [7]. Finally, traffic remains pretty similar at the same time of day across days of the same type.

The first step of the construction procedure consists of manually labeling and removing possible anomalous events. However, as the complete ground truth is unknown in real traffic, we cannot completely rely on individual labeling of alarms. Therefore, we have to accept that minor anomalies may go undetected if their effect is comparable with purely random fluctuations. Then, the dataset is transformed to eliminate possible residual (unknown) anomalies present in the real traffic, while preserving the above mentioned structural characteristics. The transformation procedure is described as follows.

Let consider a real dataset spanning a measurement period of a few weeks, for a total of m consecutive one-day intervals (e.g., $m = 28$ in our case). Each one-day period starts and ends at 4:00 am local time: this is the time-of-day where the number of active devices reaches its minimum (considering a single time-zone). Denote by m_W and m_F the number of working and festivity (W- and F-) days, respectively, in the real dataset (e.g., $m_F = 8$ and $m_W = 20$), and by K the total number of 1-min time bins ($K = 28 \cdot 24 \cdot 60 = 40320$). For each device i consider the vector $\mathbf{d}_i \equiv \{c_i^{\tau_0}(k),\, k = 1, 2, \ldots, K\}$ at the minimum timescale ($\tau_0 = 1$ minute) across the whole real trace duration, where each element $c_i^{\tau_0}(k)$ is the list of the DNS measurements related to device i at time k. For those time bins where device i is inactive, the corresponding element in $\mathbf{d}_i$ is empty. We now divide this vector into m blocks, each one corresponding to a single one-day interval. Each block is classified as W- or F-block based on the calendar day. At this point we apply a random *scrambling* within the W class: each W-block element of $\mathbf{d}_i$ is randomly relocated at the same time position selected among all W-days. The same scrambling is applied independently to the F-blocks. In this way we obtain a new vector $\widetilde{\mathbf{d}}_i$ where the position of the blocks has been scrambled, separately for W- and F-blocks, but the time location and the F/W intervals have been maintained. Finally, from the set of scrambled vectors $\widetilde{\mathbf{d}}_i$ we can derive a new set of distributions for each time bin k and timescale τ, for all the considered traffic features.

The dataset obtained in this way retains certain characteristics of the real dataset, while others are eliminated. The most important change is that the random scrambling of the individual components $\mathbf{d}_i \rightarrow \widetilde{\mathbf{d}}_i$ results in the *homogenization* of the individual daily profiles — separately for W- and F-days. This eliminates any minor residual local anomaly that survived the manual labeling by spreading it out

across all one-day intervals of the same F/W type. In other words, all W-days in the new dataset share the same (synthetic) aggregate daily profile. Same applies to F-days. Note however that the synthetic dataset retains the most important characteristics of the real process. In the first place, it keeps the time-of-day variations of the number of active devices. Secondly, the semi-synthetic dataset maintains the differentiation between the two classes of W- and F- days, although it eliminates any differentiation *within* each class (e.g., between Saturday and Sunday). Thirdly, it keeps the differentiation between distributions for different time-of-day. The result of the procedure is an anomaly-free DNS dataset *structurally similar* to the real trace.

B. Modeling and Generation of Synthetic Anomalies

During six months of experimentation we encountered a few recurring large-scale DNS traffic anomalies. Investigating these events we found some common traits and we 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. In both the 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. 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. As a simple example of such types of anomalies, we have observed events in which Apple devices running a specific version of iOS lost their persistent connectivity to certain servers providing the Apple push-notification service (which is the core of the remote notifications used in virtually every iOS App), resulting in a surge of DNS requests to locate new servers, and the resulting "scanning" of the complete IP address space of Apple push-notification service. Such an event was perceived by the cellular ISP as an internal sort of DDoS attack, as a large population of their own customer devices starting "bombarding" the network, starving resources at the access in some specific regions.

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 did not introduce a modification in the distribution of the Error Flag. Note that although E_2 type anomalies are of relatively low intensity, their identification is important as, in our experience, they may lead to problems on the signaling plane.

Tab. II summarizes the characteristics of the two 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 $d = 1h$, starting at $t_1 = 9:00$. 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 type E_2 is analogous, but differs in the selection of the anomalous terminals (same OS, but not necessarily same manufacturer). The Error Flag is unaffected in this case.

Type	E_1	E_2
Start time t_1	9:00	13:00
Duration d	1h	2 days
Involved devices D	10%	5%
Back-off time	5 sec	180 sec
Manufacturer	single popular	multiple
OS	single (with sub-ver)	single (with sub-ver)
Error flag	+5% timeout	—
FQDN	top-2LD	top-2LD

Table II
ANOMALOUS DNS TRAFFIC FEATURES FOR TYPES E_1/E_2.

III. ANOMALY DETECTION AND CLASSIFICATION TECHNIQUES

In this section we describe the proposed anomaly detection and classification approach based on decision trees, focusing on the specific features used as input. Besides this decision tree based approach, we consider two statistical-based detection approaches in our study: Distribution-based Anomaly Detection (DAD) and Entropy-based Anomaly Detection, using an Exponentially Weighted Moving Average change detector (H-EMMA). In addition, we consider five standard supervised ML-based approaches previously used in the literature for classification comparison purposes: Multi-Layer Perceptron (MLP) Neural Networks, Naive Bayes (NB), Random Forest (RF), Support Vector Machines (SVM), and Locally-Weighted-based Learning (LWL). We briefly describe all these approaches next.

A. DT - Decision Tree-based Detection and Classification

DTs [5], [6] define a classification technique based on a tree graph, where inner nodes correspond to a condition on a feature and leaves are the outcome (i.e., the class). A DT represents a very popular classification algorithm due to its simplicity (it can be easily converted into a rule-based classification system) and readability (it can be graphically represented). We shall see these advantages in the evaluations. The training follows a top-down greedy algorithm that works by iteratively splitting the nodes, using either the Gini Index or the Information Gain as optimization metric. For the results presented in our approach we employed the popular C4.5 implementation, which uses the latter as metric for training.

The proposed approach uses as input a set of features which are derived from the DNS measurements and the additional data indicated in Tab. I. Tab. III describes the specific set of 36 features, which are computed for every 10-minutes long 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. The training of the decision tree-based classifier is done on top of synthetically generated datasets, which are by default labeled datasets. In the evaluations we describe the specific dataset used in this paper.

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

Table III
INPUT FEATURES FOR THE C4.5 DT-BASED DETECTOR/CLASSIFIER.

Before moving on to the description of the remaining algorithms, we make a note on the relevance and challenges associated to the selection of features. It is clear that the selection of features used by any detection and classification system plays a major role in its performance: indeed, even the best algorithm would completely fail if the input features for classification are loosely correlated to the underlying classes. The set of features presented in Tab. III, as well as the base measurements described in Tab. I are derived from expert knowledge, based on the specific types of anomalies under study. While in section IV we evaluate the impact of supervised feature selection to choose the best features for the DT classifier, the question on how to select a set of proper input features in a completely unsupervised manner remains. Unsupervised feature selection is highly challenging, as no ground-truth is available for the task, and even if different partial approaches exist, it still represents an open research problem.

Techniques such as Principal Components Analysis (PCA) have been used in the past to blindly obtain better input features, but PCA has many drawbacks and technical limitations (e.g., it assumes linear manifolds where data lies), and is not particularly appealing in the practice, as obtained features are transformations of the base ones. In [29] we have explored the possibility of using clustering techniques to blindly select a set of proper input features for the sake of network intrusion detection. In particular, given a initial and generic set of features defining the so-called *feature space*, we have applied Sub-Space clustering techniques to identify the best sub-spaces where data can be correctly analyzed. In the future we plan to apply a similar direction in the particular problem of this paper.

B. DAD - Distribution-based Anomaly Detection

Distribution-based detectors generally consider the temporal analysis of the empirical probability distribution of certain features, using some notion of similarity between the observed distribution and a set of (anomaly-free) distributions which describe the normal-operation behavior, i.e., the baseline. In this paper we rely on a DAD algorithm we have conceived and introduced in [10], which employs a powerful heuristic approach to build the baseline, taking into account the structural characteristics of the analyzed traffic such as time of day variations, presence of pseudo-cyclic weekly patterns, and long term variations. The comparison between the analyzed distribution and the baseline is performed using a symmetric and normalized version of the well-known *Kullback-Leibler* divergence. The anomaly detection test checks if the average distance among distributions exceeds an upper bound, which is periodically updated based on past observations.

C. H-EWMA - Entropy-based Anomaly Detection

A particularly popular approach for detecting anomalies in network traffic is the one represented by entropy-based analysis [8], [9]. The entropy of a feature captures in a single number the dispersion of the corresponding probability distribution, thus it is highly appealing for the analysis. However, such a compression necessarily looses relevant information about the higher distribution moments of the analyzed feature, limiting the detectability of some anomalies. Entropy-based detectors work by flagging abrupt changes in the time series of the empirical entropy of the analyzed features. We consider the well-known, yet effective, Exponentially Weighted Moving Average (EWMA) algorithm for detecting such changes.

D. RF - Random Forrest-based Classification

RF is an ensemble technique based on multiple instances of decision trees, each one based on a different part of the training set, randomly selected. These instances are called bootstrapped samples. The final outcome is generally decided by majority voting among all the bootstrapped samples.

E. SVM - Support Vector Machines Classification

SVMs are non-probabilistic binary classifiers [28]. SVM is considered one of the most powerful supervised classification algorithm. It works by representing each feature vector in a multidimensional space and trying to find a linear separation (i.e., an hyperplane) for the classes. In some cases, however, a linear separation of the space is not possible, hence it uses the so-called kernel trick, which implicitly increases the dimensionality of the space, resulting in an easier separation in a much higher dimensional space, due to the increased sparsity.

F. NB - Naïve Bayes Classification

NB is a very simple classifier based on Bayesian statistics [28]. Despite its simplicity, it is widely used as it is very efficient in a number of scenarios, especially in high-dimensional datasets. It works by assuming that features are mutually independent, which is not true in most cases, hence the adjective naïve. This assumption allows for an easy calculation of the class-conditional probabilities, using maximum likelihood estimation.

G. LWL - Locally-Weighted-based Learning Classification

LWL is another Bayes classifier [28]. It overcomes the limitations of NB, i.e., the assumption of feature independence, by learning local models. LWL constructs a new Bayes model using a weighted set of training instances at classification time.

H. MLP - Multi-Layer Perceptron Classification

MLP is an artificial neural network composed of multiple layers of neurons, each of them generally represented by a non-linear function [28]. The layers are fully connected in a feed-forward scheme. Each neuron employs an activation function that maps the weighted inputs to the output that is passed to the following layer. The weights, originally set to random values, are iteratively adjusted during the training phase.

IV. Anomaly Detection and Classification - Evaluation

In this section we evaluate the proposed C4.5 DT approach, firstly by comparing its detection performance against that achieved by the two statistical-based detection approaches (DAD and H-EMMA), and then by comparing its classification performance against the rest of the ML-based approaches. We additionally evaluate the impact on classification performance of performing feature selection on the inputs used by the C4.5 DT approach.

For evaluation purposes and following the data generation approach described in Sec. II, we construct a fully labeled dataset consisting of a full month of synthetically generated cellular network DNS measurements, reported with a time granularity of 10 minutes. The dataset contains normal operation traffic, with multiple instances of the aforementioned E_1 and E_2 anomalies.

A. Detection Performance: C4.5 DT vs. DAD and H-EWMA

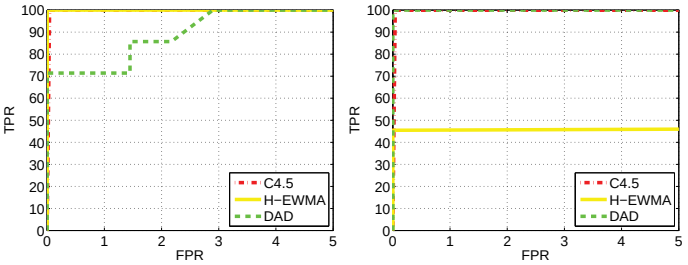
Let us first get an initial picture of the detection capabilities of the C4.5 DT approach, by injecting only two instances of each type of anomaly on top of the normal operation traffic. The training and evaluation of the C4.5 DT is performed by 10-fold cross validation: at each evaluation round, one anomaly of each type is injected in the training subset, and one anomaly of each type is injected in the validation subset. Detection performance is evaluated in a time bin basis and not in an event basis: this means that there are 6 anomalous time slots for the type E_1 anomaly and $6 \times 24 \times 2 = 288$ for the type E_2 one that need to be correctly detected by the algorithms, and not a single instance per anomaly type. We follow such a direction as we are not only interested in detecting the occurrence event, but also its full span/duration. We additionally evaluate the DAD and the H-EWMA detectors, using the distribution of number of devices across DNS query counts as monitoring feature. Note that both detectors are meant to be applied in single features, but could in principle be applied to multiple features in parallel (e.g., see Fig. 2). The purpose of this simple evaluation is to show that the achieved detection performance is similar, or even better, than that obtained by more traditional, statistical-based approaches.

Fig. 1 depicts the Receiver Operating Characteristic (ROC) curves obtained for (a) the E_1 type anomaly and (b) the E_2 type anomaly using the 3 detectors. The C4.5 DT achieves almost perfect detection performance in both cases, even slightly outperforming the DAD detector for the E_1 anomaly. The H-EWMA detector also achieves perfect detection for the E_1 anomaly, but completely fails to detect all the instances of the E_2 one, due to its low intensity.

Recall that the C4.5 DT uses a much broader set of inputs for the analysis, whereas in this evaluation, only one single feature is analyzed by both DAD and H-EWMA. To make a fairer comparison, Fig. 2 presents the detection results achieved by DAD and H-EWMA on all the additionally impacted features (FQDN, error code, OS and manufacturer for E_1 and both OS and FQDN for E_2). Drawn observations remain the same. As a first conclusion, we can see that the C4.5 DT approach offers a detection performance comparable or even slightly better to that achieved by DAD in both anomaly types. Being DAD a very powerful detector (i.e., it evaluates the complete empirical distribution of the input features), results are highly relevant.

B. Classification Performance vs ML Approaches

We move on now to the evaluation of the classification capabilities of the C4.5 DT approach. In this section we only consider the ML-based approaches described in Sec. III, as the statistical-based detectors are not meant for classification. Still, in order to improve the classification performance of the C4.5 DT approach, we consider



(a) Anomaly type E_1. (b) Anomaly type E_2.

Figure 1. ROC curves for the detection of anomalies type E_1 and E_2.

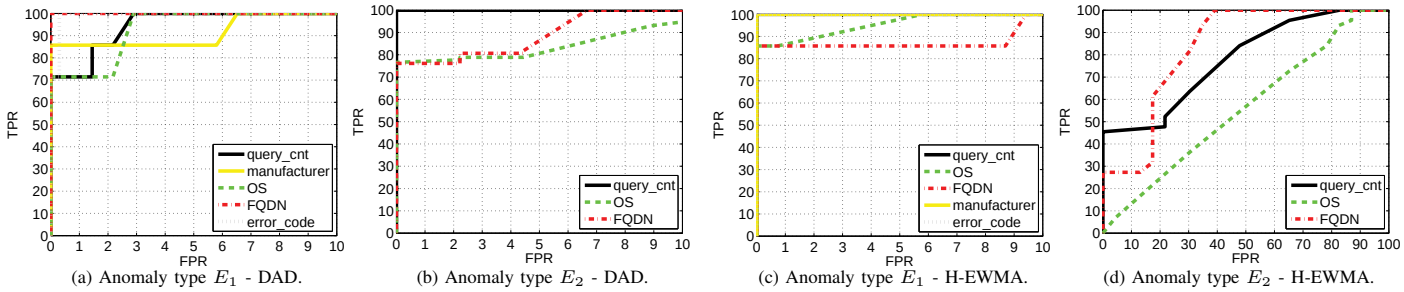
the output of the statistical anomaly detectors as additional input features for the ML-based classifiers. This is a common and powerful approach followed in the ML domain, and permits to integrate the valuable information produced by both detectors into the classification process, resulting in a sort of multi-way classification approach. We therefore include 12 additional features to the synthetic dataset: the distance metric obtained by the DAD approach and the output of the EMWA algorithm, for all the 6 fields depicted in Tab. III (the names of these features follow the notation `field_name_adtool` for the DAD approach and `field_name_ewma` for the H-EWMA one). To perform a better evaluation, we extend the labeled dataset by introducing multiple instances of E_1 and E_2 anomalies but with a different fraction of the device population involved in the anomaly, going from 0.1% to 20%. 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 service outages, responding with a surge in the number of DNS queries. We take two different intensities for this anomaly type, considering a population of 12% and 3% respectively (size of virtual-operator customer populations, as observed from our real measurements). The duration of E_3 anomalies is 1 hour. In total we include 16 different variations of anomalies, 7 of type E_1 , 7 of type E_2 and 2 of type E_3. Each time bin is assigned a class, either normal - label 0, or anomalous - label 1, 2 or 3 for the three anomaly types respectively.

We use the well-known Weka Machine-Learning software tool [23] to calibrate the six ML-based algorithms and to perform the evaluations. Reported results refer to optimal parameter settings, after thorough testing. We address the interested reader to the survey [28] and to the Weka documentation [23] for additional information on the different configuration parameters of each algorithm.

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.

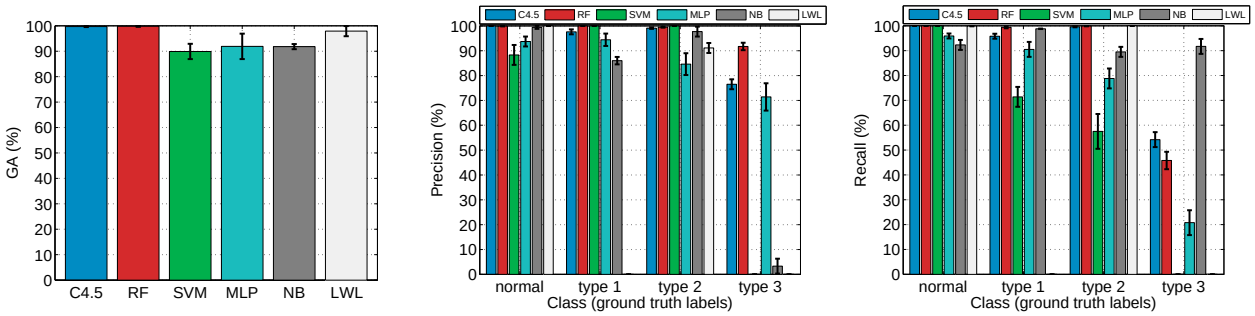
$$\mathrm{R}_i = \frac{TP_i}{n_i}, \quad \mathrm{P}_i = \frac{TP_i}{TP_i + FP_i}, \quad \mathrm{GA} = \frac{\sum_{i=1}^{M} TP_i}{n} \tag{1}$$

1) Results and Discussion: Fig. 3 reports the performance of the six compared classifiers in the classification of all the 10-minutes time bins. All the evaluations presented use 10-fold cross-validation,



(a) Anomaly type E_1 - DAD. (b) Anomaly type E_2 - DAD. (c) Anomaly type E_1 - H-EWMA. (d) Anomaly type E_2 - H-EWMA.

Figure 2. ROC curves for the detection of anomalies type E_1 and E_2 for the DAD and H-EWMA anomaly detectors, considering all the impacted features.



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

Figure 3. Classification Accuracy, Precision, and Recall for normal operation instances and different anomaly-types' events. The performance of C4.5 DT is almost perfect for normal traffic and anomalies of type 1 and 2, but quality significantly drops for the anomaly type 3.

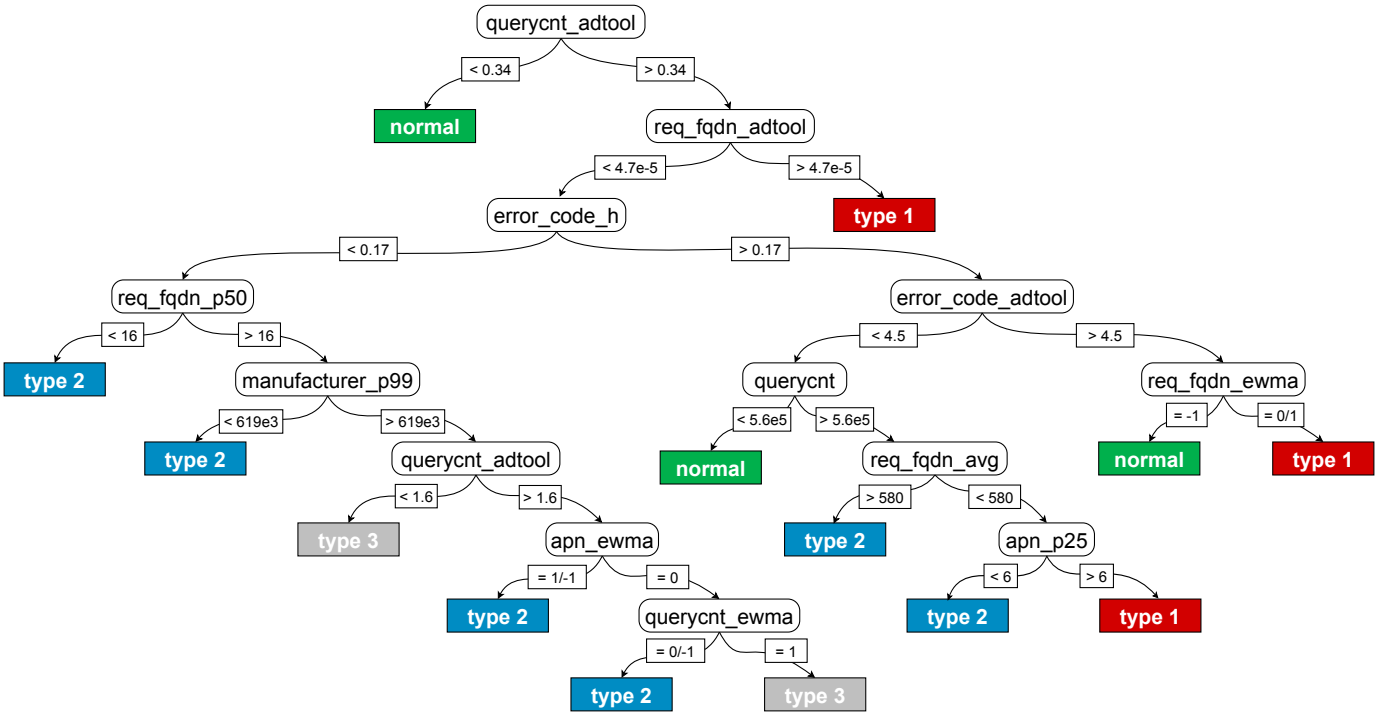
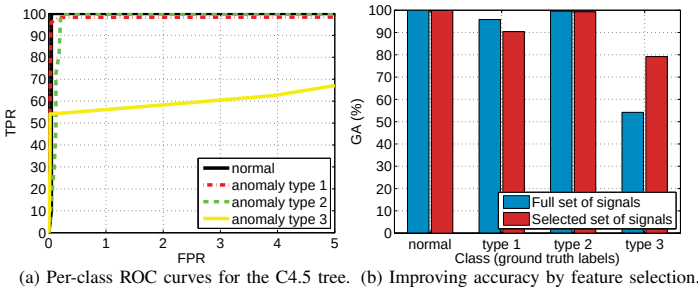


Figure 4. Pruned C4.5 DT model for anomaly diagnosis. C4.5 achieves very high global accuracy, as well as very high precision and recall for normal traffic and type 1, type 2 anomalies. However, this tree is not capable of properly tracking type 3 anomalies. This issue can be solved by performing pre-filtering on the input features, by feature selection techniques.

which means that we train and test the models for 10 different training/testing combination sets, to avoid biased results. To be fare in the comparisons, parameters are set manually for all the models, performing an extensive trial-and-error testing phase to obtain the best results. In addition, for each classifier we run each 10-fold cross validation experiment for 100 consecutive times, using different random splits of the data, an compute the resulting Confidence Intervals – CIs (with a 95% confidence level). Fig. 3(a) depicts the global accuracy obtained by the six approaches. All the models provide very high accuracy, above 90%. The C4.5 decision tree model



(a) Per-class ROC curves for the C4.5 tree. (b) Improving accuracy by feature selection.

Figure 5. Performance of the C4.5 anomaly classifier, and classification enhancement through feature selection.

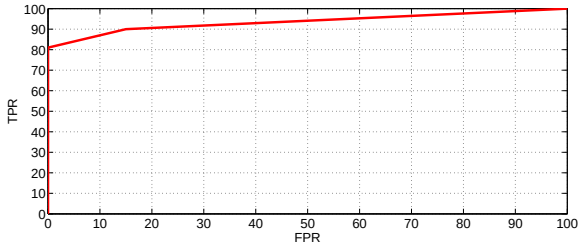


Figure 6. OOS test performance for the C4.5 anomaly detector. The detector is trained with instances of anomaly types E_1, E_2 and E_3, and tested on anomaly of type E_4, not available in the training set.

achieves the same performance as the RF, but the latter uses 20 parallel C4.5 decision trees instead of a single one. SVM, MLP and NB achieve slightly worse performance in terms of accuracy, which is a-priori surprising, as at least SVMs and MLPs have proved to be very good classifiers in previous work. Understanding this under-performance is part of our future work. Regarding CIs, even if the number of anomalous slots for each type of anomaly is rather small, there are no particular bias for most of the classifiers. The MLP classifier shows a slightly higher variance in the results, which might suggest a less robust performance in terms of over-fitting. This comes directly from the MLP structure used in the tests, which has 10 hidden neurons in the intermediate layer, making the tunning more cumbersome and prone to higher bias.

Regarding precision and recall depicted in Figs. 3(b) and 3(c), we can observe that all the approaches systematically fail to properly track the type 3 anomalies. C4.5 achieves high precision and recall for normal and type 1, type 2 anomalies, but also fails to properly isolate type 3 events, resulting in a very low recall. While the problem of unbalanced classes is for sure an issue partially masking these results, the particularities of type 3 anomalies require additional efforts to properly track them. Indeed, as also shown in Fig. 5(a), while the per-class ROC curves obtained for the first 3 classes (0, 1, and 2) by C4.5 are almost perfect (TPR = 100% for a FPR below 1%), the ROC curve for the type 3 events shows poor results. As we see next, we can greatly improved the performance of C4.5 DT for classification of type 3 events by performing features filtering and selection.

Finally, Fig. 4 depicts the obtained C4.5 DT model. As we claimed before, decision tree models provide great insights about the process leading to a specific classification result. Using the model, a network operator can identify those features leading to specific type of anomaly, and better infer on their nature. Features at the higher levels of the tree have more distinguishing power and account for more population size than lower level features. In this model, the root node is the output of the DAD detector on the distribution of DNS requests, showing the paramount role and information provided by such feature. This was in fact one of the main reasons for including the outputs of the DAD and H-EWMA detectors as inputs to the C4.5 DT model. Note also the relevance of the FQDN-related features, which appears in the classification of anomalies of type E_1 and E_2. Note that paths from the root node to leaves representing different anomaly types can be directly expressed as logical rules, which can be ultimately integrated into any kind of rule-based monitoring system.

2) Improving C4.5 DT Performance by Feature Selection: As we said before, using an extensive list of traffic features as input is not always the best strategy, as it may negatively impact classification results. Using more features increments the dimensionality of the feature space, normally introducing undesirable effects such as sparsity. At the same time, using irrelevant or redundant features may diminish performance in the practice. We show next that by carefully addressing the pre-filtering of input features by standard feature selection techniques, we can partially solve the classification problem of the C4-5 DT model, related to type 3 anomalies.

There are different search strategies and evaluation criteria to construct a sub-set of traffic features. Regarding search strategies, the idea is to test different sub-sets of features, studying local changes in the particular evaluation criterion when adding or removing features. The evaluation criterion permits to test the goodness of a particular sub-set. In this paper we apply a widely used evaluation criterion to construct a reduced sub-set of features: correlation-based evaluation. This approach basically selects sub-sets of features that are poorly correlated among each other, but highly correlated to the classes of traffic. As search strategy, we use Best-First (BF) search; BF is similar to a standard greedy exploration, but it has the ability to do backtracking, i.e., it basically keeps the previously evaluated sub-sets so as to avoid local maximum/minimum results when there is no local improvement.

We now evaluate the impact of feature selection on the performance of the C4.5 DT model. By running the proposed technique, we end up with a greatly reduced set of features, going from the initial 48 features to only 4. The resulting set is composed of the following features: querycnt_adtool, apn_avg, req_fqdn_p25, and req_fqdn_adtool. Interestingly, these features have a high correlation to the type E_3 anomaly characteristics, which are directly linked to APN and DNS query counts. Also, the relevance of these features is partially shown by the original tree in Fig. 4, which reinforces the advantages of the self-feature-selection achieved during the training of the DT. To conclude, Fig. 5(b) shows the per-class accuracy obtained by the C4.5 DT model for both input features sets (i.e., the full set of features, and the pruned, 4 features set). While the performance obtained in the classification of type E_1 anomalies is slightly worse when performing feature selection, there is a great improvement in the detection performance for the type E_3 anomalies, partially compensating the initial problems of the C4.5 DT model.

3) Out-of-Sample Performance Evaluation: Even if all the evaluations presented in this paper use 10-fold cross validation to reduce the bias introduced by potential model over-fitting, we performed an out-of-sample (OOS) test to verify the usability of the DT-based approach in a more general scenario. In an OOS test, we use the DT trained with the set of anomalies E_1, E_2 and E_3 to detect a new anomaly type E_4 not available in the training set. In the ideal case, the anomaly type E_4 should be a real anomaly observed in the cellular network, to verify the performance of the detector in the real environment. However, we re-sort once again to synthetic data generation, to protect the ISP privacy. The anomaly of type E_4 belongs to the same class of anomalies as E_1, E_2 and E_3, meaning that devices increase the number of DNS queries, but in this case they do it as part of a short-term (30') moderate flash-crowd, and thus do not generate a marked deviation in the distribution of queries per device. In fact, the numbers of queries is slightly increased by only 2% w.r.t. those issued by the most active (i.e., generating the

most of the queries) devices at peak load time. Given the flash-crowd nature, there is a particularly visible variation in the features related to the FQDN pointing to the requested content. As we did before, we take a different fraction of the device population involved in this new anomaly, going from 0.1% to 20%. Fig. 6 depicts the ROC curve obtained in this test. While detection performance is impacted as compared to the results obtained with the DT presented in Fig. 4 for the other anomaly types, the detector can still correctly recognize more than 80% of the anomalous slots of type E_4 without false positives. This results point to the generalization of the proposed approach for detecting non-observed anomalies of the same class as the one analyzed in this paper. A wider generalization of the approach is out of scope for this paper.

V. Related Work

There has been considerable amount of research about anomaly detection in network traffic. A large set of papers apply concepts and techniques imported from fields like Neural Networks, Self-Organizing Maps [12], Genetic Algorithms [13], Fuzzy Logic [14], Data Mining [15], Machine Learning [16], etc. 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 [17] CUSUM [18] and others.

It is commonly accepted that information-theoretic concepts, and in particular entropy measures, are well-suited for anomaly detection [8], [9]. Distribution-based approaches such as [10] are intrinsically more powerful, as they look at the entire distribution, rather than only at some specific mode or aggregation. The cost is of course a larger amount of data to be processed, and higher complexity of the monitoring platform.

Finally, regarding 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 [19], linear discriminant analysis and k-nearest-neighbors [20], decision trees and feature selection techniques [21], and support vector machines [24]. 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 [25], sub-space clustering techniques [27], and a combination of k-means and maximum-likelihood clusters labeling [26]. We refer the interested reader to [28] for a detailed survey on the different ML techniques applied to automatic traffic classification.

VI. Conclusions

In this paper we have presented a ML-based approach for detection and classification of large scale Internet anomalies based on the analysis of passively captured network data. The approach is based on popular C4.5 decision tress, offering a very powerful and simple to understand and to track technique to both detect and classify anomalies. We believe that such an approach can provide high insights and visibility for daily network operations, specially 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 developed an approach to generate semi-synthetic data, derived from real traffic traces. We believe that this is also a main contribution of our work, as it would help the owners of real data to make such datasets available for the research community without disclosing any privacy or business sensitive information. By relying on Machine Learning techniques, we have shown how to classify the detected anomalies in an automatic fashion. In particular, we investigated a number of supervised classification techniques and the effects of feature selection on classification performance, showing how the C4.5 DT-based approach outperforms the rest. Even more, we compared the detection performance of this approach against other well-known detectors in the literature (e.g., distribution-based, entropy-based), and proposed a multi-detector approach to increase the overall classification performance in a number of case studies. Based on suggestions from the expert reviewers, we are now exploring the possibility of constructing better ML-based models for our problem, relying on traditional model selection strategies and testing different selection criteria.

References

[1] P. Fiadino et al., "Characterizing Web Services Provisioning via CDNs: The Case of Facebook", in *TRAC*, 2014.

[2] P. Casas et al., "When YouTube doesn't Work – Analysis of QoE-relevant Degradation in Google CDN Traffic", *IEEE TNSM*, vol. 11 (4), 2014.

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

[4] A. Aucinas et al., "Staying Online While Mobile: The Hidden Costs", in *ACM CoNEXT*, 2013.

[5] R. Kohavi et al., "Decision-tree Discovery", in *Handbook of Data Mining and Knowledge Discovery*, pp. 267-276, 2002.

[6] C. Kruegel et al., "Using Decision Trees to Improve Signature-Based Intrusion Detection", in *Proc. RAID*, 2003.

[7] P. Romirer-Maierhofer et al., "Device-specific Traffic Characterization for Root Cause Analysis in Cellular Networks", in *TMA*, 2015.

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

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

[10] 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), pp. 245-269, 2010.

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

[12] A. Mitrokotsa et al., "Detecting denial of service attacks using emergent self-organizing maps", in *IEEE ISSPIT*, 2005.

[13] M. Ostaszewski et al., "A non-self space approach to network anomaly detection", in *IEEE IPDPS*, 2006.

[14] W. Chimphlee, et al., "Integrating genetic algorithms and fuzzy C-means for anomaly detection", in *IEEE Indicon*, 2005.

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

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

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

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

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

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

[21] N. Williams el al., "A Preliminary Performance Comparison of Five Machine Learning Algorithms for Practical IP Traffic Flow Classification", in *ACM CCR*, vol. 36 (5), pp. 5-16, 2006.

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

[23] "Weka Data Mining", at http://www.cs.waikato.ac.nz/ml/weka/.

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

[25] J. Erman et al., "Traffic Classification using Clustering Algorithms", in *MineNet*, 2006.

[26] J. Erman et al., "Semi-Supervised Network Traffic Classification", in *ACM SIGMETRICS*, 2007.

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

[28] 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.

[29] P. Casas et al., "Unsupervised Network Intrusion Detection Systems: Detecting the Unknown without Knowledge", in *Computer Communications*, vol. 35 (7), pp. 772-783, 2011.