Machine and Deep Learning Approaches for IoT Attack Classification

Alfredo Nascita, Francesco Cerasuolo, Davide Di Monda,
Jonas Thern Aberia Garcia, Antonio Montieri, Antonio Pescapè
University of Napoli "Federico II" (Italy)
alfredo.nascita@unina.it, {fran.cerasuolo, dav.dimonda, j.garcia}@studenti.unina.it, {antonio.montieri, pescape}@unina.it

***Abstract*—In recent years, Internet of Things (IoT) traffic has increased dramatically and is expected to grow further in the next future. Because of their vulnerabilities, IoT devices are often the target of cyber-attacks with dramatic consequences. For this reason, there is a strong need for powerful tools to guarantee a good level of security in IoT networks. Machine and deep learning approaches promise good performance for such a complex task. In this work, we employ state-of-art traffic classifiers based on deep learning and assess their effectiveness in accomplishing IoT attack classification. We aim to recognize different attack classes and distinguish them from benign network traffic. In more detail, we utilize effective and unbiased input data that allow fast (i.e. "early") detection of anomalies and we compare performance with that of traditional (i.e. "post-mortem") machine learning classifiers. The experimental results highlight the need for advanced deep learning architectures fed with input data specifically tailored and designed for IoT attack classification. Furthermore, we perform an occlusion analysis to assess the influence on the performance of some network layer fields and the possible bias they may introduce.**

***Index Terms*—Internet of Things, Machine Learning, Deep Learning, Network Security, Attack Classification.**

I. Introduction

The Internet of Things (IoT) is experiencing an extraordinary growth and is currently applied in a variety of domains. The number of connected devices is growing every day and it is estimated to reach 30.9 billion by 2025.[1] This phenomenon has led to a rapid increase in traffic generated in IoT networks. Despite their widespread use, IoT devices still expose severe vulnerabilities, including—to name a few—unprotected network services, lack of encryption or access control, and insufficient protection for sensitive data. Consequently, the number of attacks against IoT devices is growing at a rapid pace, and there is an urgent need for tools capable of recognizing attacks to properly react and enforce countermeasures.

In this context, intrusion detection systems are of paramount importance as they are used to monitor network devices and determine whether they are source or target of attacks. Various techniques have been developed to identify cyber-attacks and, nowadays, a large number of tools increasingly leverage Machine Learning (ML) and Deep Learning (DL) approaches to be more effective and efficient.

Going into details, this work focuses on *Attack Classification (AC)*, a process aiming at simultaneously modeling malicious and legitimate behaviors as a multi-class classification task. In other words, we aim to infer the specific attack and distinguish malicious from benign network traffic. In fact, while Traffic Classification (TC) is a well-established task, paramount to most management and planning actions, an assessment of modern classification approaches based on ML and DL for performing AC is still an open problem, above all in IoT networks constituting a scenario challenged by aforementioned specific characteristics of IoT devices.

In view of these considerations, in the present work, we apply advanced ML and DL approaches for AC in IoT networks. The study is carried out on the recently-published IoT-23 dataset [1], encompassing the traffic generated with both infected and legitimate real-hardware IoT devices. The specific contributions are summarized hereinafter. (a) We categorize the *state-of-the-art in ML/DL-based AC* toward its effective application in *IoT networks*, providing also a taxonomy of the most-related literature. (b) We perform *preprocessing operations* on the IoT-23 dataset to mitigate class imbalance and extract *unbiased input data*; starting from criteria derived from ML/DL-based TC [2], *we design and extract effective input for AC based on the nature of malicious traffic analyzed*. (c) We assess the performance of *state-of-the-art advanced DL architectures* borrowed from TC and *first shifted to AC domain* (an 1D-CNN, a hybrid 2D-CNN+LSTM, and the multimodal architecture MIMETIC [3]), and compare them with *traditional ML algorithms* including both simple and ensemble methods. (d) We perform an *occlusion analysis* to understand the reasons behind deep models' working and to particularly evaluate the possible bias introduced when considering input data extracted at network layer.

II. Related Work

The present section summarizes the most related works facing Anomaly Detection (AD) and AC via ML- and DL-based traffic classifiers. Specifically, we consider recent papers published in the last **five years** and categorize them by highlighting the key aspects of each work as reported in Tab. I. Firstly, we consider only papers facing AD (i.e. a binary task: benign vs. malicious), AC (i.e. a multi-classification task: identification of the specific malicious attack), or both **tasks** via **DL** approaches or exploiting the IoT-23 dataset [1].

On the other hand, regarding the works using **datasets** different than IoT-23, we can notice that most of the less

[1]IoT connected devices worldwide from 2010 to 2025 - www.statista.com.

TABLE I

RELATED WORKS USING MACHINE AND DEEP LEARNING APPROACHES FOR ANOMALY DETECTION AND ATTACK CLASSIFICATION. THE PAPERS ARE ORDERED BY YEAR. THE LAST ROW SUMMARIZES THE PRESENT WORK. ACRONYMS MEANING IS REPORTED AT THE BOTTOM OF THE TABLE.

Ref	Year	Dataset	Task	TO	DL	MM	Raw	Input Data	Technique
[4]	2017	NSL-KDD	AD&AC	B	●	○	○	Flow-based statistics	RNN
[5]	2018	NSL-KDD	AD&AC	B	●	○	○	Flow-based statistics	DNN
[6]	2018	NSL-KDD	AD&AC	B	●	○	○	Flow-based statistics	CNN
[7]	2018	NSL-KDD, KDD-Cup-99	AC	B	●	○	○	Flow-based statistics	NDAE
[8]	2019	Bot-IoT	AD	B	●	○	○	Flow-based statistics	SVM, RNN, LSTM
[9]	2019	KDD-Cup-99, NSL-KDD, UNSW-NB15, Kyoto 2006+, WSN-DS, CICIDS2017, ADFA	AD&AC	B	●	○	○	Flow-based statistics	DNN, LR, NB, KNN, DT, AB, RF, SVM
[10]	2019	NSL-KDD, UNSW-NB15, CICIDS2017	AD&AC	B	●	●	○	Flow-based statistics	MDAE, LSTM, DNN, NB, SVM
[11]	2020	Bot-IoT, CSE-CIC-IDS2018	AD&AC	B	●	○	○	Flow-based statistics	DNN, RNN, CNN, RF, NB, SVM, MLP, RBM, DBN, DBM, DAE
[12]	2020	UNSW-NB15, CICIDS2017	AD&AC	B	◐	○	○	Flow-based statistics	LR, SVM, DT, RF, MLP, KNN, BG, BS, ST
[13]	2020	IoT-23	AD	B	●	○	○	Flow-based statistics	LSTM+RNN
[14]	2020	Bot-IoT	AD&AC	B	●	●	◐	Biflow/packet-based L3/L4 fields	M2-DAE
[15]	2021	ToN_IoT, IoT-23	AC	B	◐	○	◐	Flow-based features	BS, RF, MLP
[16]	2021	IoT-23	AD	B	●	○	○	Time window-based statistics	TFNN, DT, RF, AB
[17]	2021	IoT-23	AD&AC	B	●	○	○	Flow-based statistics	2D-CNN+LSTM
[18]	2021	NSS Mirai IoT-23 (Mirai)	AC	F	○	○	◐	Flow-based features	FIM, KNN, RF, SVM
[19]	2021	Bot-IoT, IoT Network Intrusion, MQTT-IoT-IDS2020, IoT-23	AD&AC	B	●	○	○	Flow-based statistics	1D-CNN, 2D-CNN, 3D-CNN
This work	2021	IoT-23	AC	B	●	●	●	L4/L2 unbiased payload Per-packet header fields	MIMETIC, 1D-CNN, 2D-CNN+LSTM, NB, DT, RF, BG

Task: Anomaly Detection (AD), Attack Classification (AC). **TO**: Traffic Object: Biflow (B), Flow (F). **DL**: Deep Learning. **MM**: Multimodal. **Raw**: Raw input data. **Technique**: AdaBoost (AB), Bagging Classifier (BG), Boosting Classifier (BS), Convolutional Neural Network (CNN), Deep AutoEncoder (DAE), Deep Boltzmann Machine (DBM), Deep Belief Network (DBN), Deep Neural Network (DNN), Decision Tree (DT), Frequent Itemset Mining (FIM), KNN (K-Nearest Neighbors), Linear Regressor (LR), Long Short-Term Memory (LSTM), Multimodal Deep AutoEncoder (MDAE, M2-DAE), MultiLayer Perceptron (MLP), Naïve Bayes (NB), Non Symmetric Deep AutoEncoder (NDAE), Random Forest (RF), Recurrent Neural Network (RNN), Restricted Boltzmann Machine (RBM), Stacked Denoising AutoEncoder (SDAE), Stacking Classifier (ST), Support Vector Machine (SVM), Transformer-based Neural Network (TFNN).
"+" symbol indicates hybrid architectures; ● present, ○ lacking, ◐ partial.

recent ones [4, 5, 6, 7, 9] leverage KDD-Cup-99 or NSL-KDD. Unfortunately, such datasets hardly exhibit a current real-world network traffic profile, particularly considering that they were collected more than two decades ago. Moreover, despite the higher number of statistical features provided by the refined NSL-KDD dataset with respect to KDD-Cup-99, the use of such handcrafted statistics nullifies a key advantage of DL, that is the automatic extraction of knowledge from raw traffic data without the need of human-expert intervention.

Consequently, we explicitly flag when DL architectures are effectively fed (also partially) with **raw** input data. In more detail, with the exception of recent works [14, 15, 18] extracting different engineered **input data** fields—including not only flow-based statistics, but also features related to connection metadata, attack-stage activities, higher-layer protocol behaviors, etc.—from raw PCAP traces, we can notice that all the other works counter-productively apply DL to manually-extracted traffic statistics. Interestingly, in addition to those leveraging already preprocessed datasets (e.g., KDD-Cup-99, NSL-KDD, Kyoto 2006+, WSN-DS, see [9] for details), most works manually extract statistics on the sets of packet/payload lengths, inter-arrival times, etc. from raw traffic via tools such as CICFlowMeter[2] or Zeek[3]. Furthermore, the choice of using input data encompassing PCAP metadata (e.g., timestamps or ad-hoc IDs) or biased fields (e.g., local IP addresses or source/destination ports) is likely to introduce bias inflating AC performance [2]. Regarding the traffic segmentation adopted, we observe that flows and especially bidirectional flows (briefly biflows) are the most-common choices as **traffic objects**. On the contrary, referring to the specific AC **technique** applied, several types of DL algorithms are leveraged, including Deep Neural Networks (DNNs), different AutoEncoders (AEs), both 1D-, 2D-, and 3D-Convolutional Neural Networks (CNNs), and variants of Recurrent Neural Networks (RNNs) as Long Short-Term Memory (LSTM). In addition to our, only two other papers [13, 17] exploit the composition capabilities of hybrid DL architectures. We also underline that, in Tab. I, we have partially flagged the **DL** column for the works [9, 15] that employ MultiLayer Perceptron (MLP) without specifying if in the deep or shallow variant. Additionally, several ML approaches are commonly used as

[2]https://github.com/CanadianInstituteForCybersecurity/CICFlowMeter
[3]https://zeek.org/

baselines for comparison, usually showing poorer performance than DL ones. They include both single ML classifiers (e.g., Naïve Bayes and Decision Tree) and ensembles (e.g., Random Forest and AdaBoost).

Finally, we can notice that, with the exception of the works [10, 14] both using a variant of the MultiModal Deep AutoEncoder (MDAE), the present work is the only one that investigates the performance of a *state-of-art multimodal (**MM**) architecture* [3] for AC and compare its performance against both single-modal DL proposals and ML algorithms (single and ensembles). Moreover, the present work uniquely considers different types of *unbiased raw input data* (cf. Sec. III) for feeding such DL architectures and also conducts a careful *occlusion analysis* to evaluate the potential bias introduced by input data pertaining to layers lower than L4.

III. Experimental Setup

A. Deep Learning-based Attack Classification

In this work, we aim to perform traffic classification of attacks conducted against IoT devices, briefly **AC**. Formally, given a **traffic object** [2], namely an aggregation of traffic packets sharing common proprieties, AC consists in assigning a label among L classes (each corresponding to benign traffic or to a specific attack) within the set $\{1, \cdots, L\}$. To this aim, we leverage state-of-art traffic classifiers based on both **single-modal** and **multimodal DL architectures** [2, 3], ensuring that they are fed with *unbiased* input data, which do not inflate performance and then do not lead to misleading outcomes.

In more detail, we exploit a single-modal 1D-CNN having a base architecture (i.e. layers and related hyperparameters) as that proposed in [20]. For each traffic object, we feed the 1D-CNN with the first N_b bytes of transport-layer payload (`PAY`) or of network-layer header and payload (`NET`) after removing biased fields (see Sec. IV-C). `PAY` and `NET` input types are arranged in a *byte-wise* fashion and normalized within $[0, 1]$.

Additionally, we consider a single-modal hybrid architecture based on a combination of 2D-convolutional and LSTM layers (named HYBRID hereinafter) [21]. HYBRID has as input M informative unbiased fields extracted from the header of the first N_p packets (`HDR`) of each traffic object. Based on suggestions of previous works [2, 21], we extract the following per-packet header fields: (i) the number of bytes in transport-layer payload, (ii) the direction $\in \{-1, 1\}$, (iii) the TCP windows size (equal to 0 for UDP packets), (iv) the time elapsed since the arrival of previous packet (i.e. the inter-arrival time), (v) the time-to-live (TTL), and (vi) the TCP flags (as a six-bit vector denoting the presence (1) or absence (0) of each flag). `HDR` input type is scaled with a quantile transformer to follow a uniform distribution that spreads out the most frequent values and reduces the impact of outliers.

Finally, we employ the multimodal traffic classifier (named MIMETIC) we have recently proposed in [3] to capitalize the intrinsic heterogeneity of traffic data via the intermediate fusion of highly-structured features extracted via DL. MIMETIC consists of two branches each corresponding to a different input modality: a 1D-convolutional branch fed with `PAY`/`NET` input type and a branch based on a bidirectional Gated Recurrent Unit fed with `HDR` input type. The intermediate features extracted by each branch are then concatenated and further elaborated via a shared dense layer. MIMETIC is trained via a two-phase procedure consisting of pre-training of individual modalities and fine-tuning of the whole architecture.

All DL-based classifiers are completed with a softmax (i.e. a dense layer with L nodes and softmax activation) and are trained for a total of 90 epochs (for MIMETIC, we consider 25 epochs for pre-training of each modality and 40 epochs for fine-tuning) to minimize categorical cross-entropy loss. Also, we leverage the Adam optimizer (with batch size of 32 for single-modal classifiers and of 128 for MIMETIC) and the early-stopping technique (with patience of 15 epochs and minimum delta of 0.01) to prevent overfitting. In case the instances are *longer* or *shorter* than the considered fixed-length input data (i.e. N_b or N_p), to constrain them to the designed length, we truncate or pad with zeros, respectively.[4]

Furthermore, we consider as baselines **ML-based traffic classifiers** fed with 70 *handcrafted biflow-based features*, namely the number of packets, the packet rate, and statistics (i.e. min, max, mean, standard deviation, variance, mean absolute deviation, skewness, kurtosis, and percentiles) computed on the set of packet lengths, inter-arrival times, TTLs, and TCP window sizes. We evaluate four state-of-art ML-based classifiers: (i) Gaussian Naïve Bayes (NB), (ii) Decision Tree (DT, with max depth of 16 and entropy feature-split criterion), (iii) Random Forest (RF, with 200 estimators and entropy feature-split criterion) and (iv) Bagging classifier (BG, with 100 DT estimators). For completeness, we have also evaluated three feature selection techniques (i.e. Recursive Features Elimination without and with Cross-Validation and Sequential Features Selection) without finding appreciable differences.[5] Therefore, in the following, we employ the aforementioned feature set without further modifications. Finally, we underline that such approaches are applicable only in *"post-mortem" AC*, as opposed to all DL classifiers employed that are fed with input types which are naturally suited for *"early" AC*.

B. Dataset Description and Preprocessing Operations

We employ the IOT-23 dataset [1] collected at the Stratosphere Laboratory of the Czech Technical University between 2018 and 2019. Overall, IOT-23 encompasses 23 traffic traces (i.e. PCAP files named scenarios) captured within a controlled IoT network environment with unrestrained network connection: 20 traces are related to malicious traffic and 3 to benign traffic. In more detail, each scenario corresponds to a specific malware sample or to benign traffic. A *Raspberry Pi* infected with a certain malware generates malicious traffic, while the 3 benign scenarios are generated each with a different real-hardware IoT device: a *Philips HUE Smart Led Lamp*, an

[4] To distinguish actual from padded zeros, we add 1 to all bytes of `PAY`/`NET` input before normalization (by dividing each byte by 256) and zero-padding.

[5] We use a validation set (20% of training set) for parameter tuning and feature selection, while we consider Scikit-learn (https://scikit-learn.org/) default values for parameters not explicitly mentioned before.

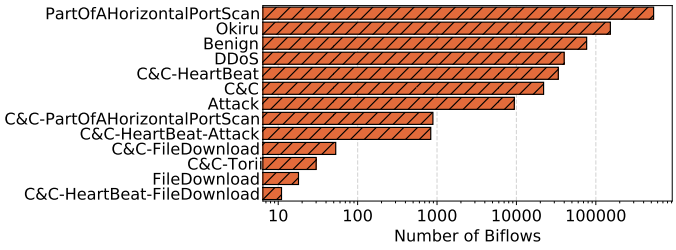


Fig. 1. Number of per-class biflows (in log scale) of preprocessed IoT-23 dataset. For details on label meaning, please refer to [1].

Amazon Echo Home, and a *Somfy Smart Doorlock*. IoT-23 authors have manually analyzed each PCAP file and defined a set of handcrafted rules to label Zeek biflows. Specifically, the labels describe the relation between malicious flows and malicious activities performed, while non-malicious traffic is labeled as "benign". For brevity, we refer to IoT-23 website [1] for the details on the labels used for malicious traffic.

To exploit the biflow-level labels provided with the IoT-23 dataset, we parse the raw PCAP files and aggregate packets into *biflows*[6] thus being our traffic objects. It is worth noting that this is also the most common choice of the vast majority of state-of-art works tacking AC or AD (cf. Tab. I).

IoT-23 exhibits a class imbalance problem, with the four most highly-populated classes having more than $15M$ biflows (compared with other ones having no more than $40k$ samples), while three minority classes present less than 10 biflows. Therefore, based on such per-class occurrences, we have randomly down-sampled (without replacement) the following classes to the 0.25% of the original dataset[7]: (i) *PartOfAHorizontalPortScan*, (ii) *Okiru*, (iii) *DDoS*, and (iv) *Benign*. On the other hand, we have removed the classes whose number of biflows is less than 10: (i) *C&C-Mirai*, (ii) *Okiru-Attack*, and (iii) *PartOfHorizontalPortScan-Attack*. Overall, employed dataset contains $870.6k$ biflows. Figure 1 shows their distribution among the 13 final classes. We underline that—although preprocessing operations guarantee a sufficient number of biflows for each class and reduce the computational burden—given the unbalanced per-class share of samples, such a dataset represents a realistic and challenging evaluation testbed.

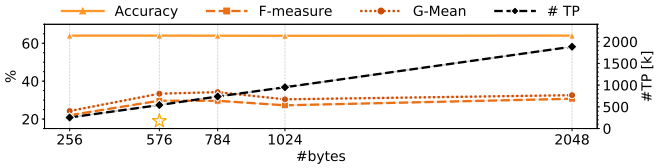
IV. Experimental Evaluation

Our evaluation is based on a *stratified hold-out technique*: we split the dataset into training (75%) and test (25%) sets. Performance is evaluated in terms of *accuracy* (the ratio of correctly classified samples), *macro* (i.e. arithmetically-averaged over classes) *F-measure*, and *macro G-mean*.[8]

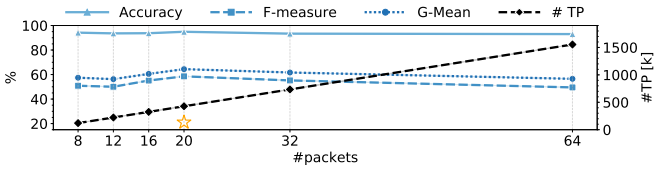
[6]A bidirectional flow or *biflow* is defined as a stream of packets sharing the same quintuple (i.e. transport-level protocol, source and destination IP addresses and ports) regardless the direction of communication.

[7]We have chosen to down-sample the *whole* dataset, as opposed to the sole training set, since the latter choice would have biased the overall accuracy (evaluated on the test set) toward the performance of the majority classes.

[8]F-measure is the harmonic mean of precision (the per-class ratio of decisions being correct) and recall (the class-conditional accuracy), while G-mean is the geometric mean of recall and specificity (the fraction of per-class actual negatives correctly classified as such).



(a) 1D-CNN.



(b) Hybrid.

Fig. 2. Accuracy [%], F-measure [%], G-mean [%], and number of trainable parameters [#TP] of 1D-CNN (a) and Hybrid (b) when varying the input dimensions N_b and N_p, respectively. The best value is highlighted via a $\star$.

TABLE II
Performance measures [%] of ML and DL classifiers.

Traffic Classifier	*Accuracy*	*F-Measure*	*G-Mean*
NB	37.60	25.44	55.64
DT	**95.62**	75.92	80.92
BG	95.46	76.98	**82.96**
RF	95.49	**77.28**	82.81
1D-CNN (PAY)	64.03	29.69	33.39
Hybrid (HDR)	94.88	58.33	64.40
Mimetic (PAY+HDR)	95.73	67.41	73.97
Mimetic (NET+HDR)	**99.93**	**91.70**	**93.50**
Mimetic (NET$\star$+HDR)	99.99	95.89	97.42

Unbiased per-metric best performance and overall best ML and DL classifiers (based on F-measure) are highlighted in boldface. NET$\star$ includes biased fields.

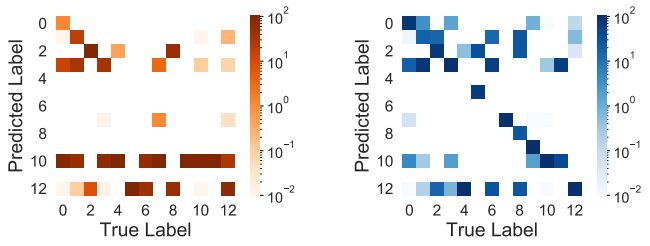
A. Sensitivity Analysis

To tune the input size of considered DL architectures, we perform a sensitivity analysis as depicted in Fig. 2. In detail, we show the AC performance for different values of N_b and N_p attained by single-modal 1D-CNN and Hybrid, respectively, in terms of all considered metrics.[9] Moreover, we report the number of trainable parameters to highlight the input size-complexity trade-off. The best configurations (marked with a $\star$) are obtained with $N_b = 576$ B and $N_p = 20$ packets, for PAY and HDR input, respectively, having also a reasonable number of trainable parameters. Given the above considerations, in the next analyses, we set $N_b = 576$ B and $N_p = 20$ packets for all the DL architectures employed.

B. Performance Overview

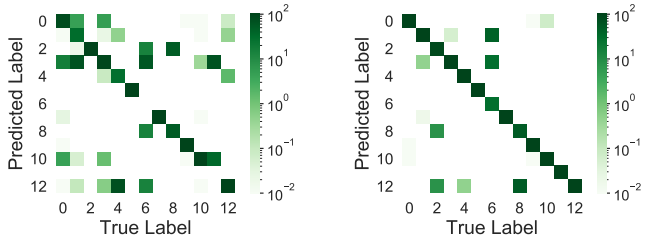
This section describes the performance obtained with ML and DL approaches exploited for AC. Looking at Tab. II, it is apparent that all ML approaches achieve satisfactory performance, with the exception of NB, which shows an accuracy of 37.60% and an F-measure of 25.44%. In particular, the ensemble methods achieve almost the same (best) performance, with RF slightly outperforming BG.

[9]For the multimodal Mimetic classifier, which is fed with both input types, we have not tested all the possible combinations of N_b and N_p, for brevity.



(a) 1D-CNN (PAY).

(b) HYBRID (HDR).



(c) MIMETIC (PAY+HDR).

(d) MIMETIC (NET+HDR).

0 = Benign
1 = C&C
2 = C&C-FileDownload
3 = C&C-HeartBeat
4 = C&C-HeartBeat-Attack
5 = C&C-HeartBeat-FileDownload
6 = C&C-Torii
7 = DDoS
8 = FileDownload
9 = Okiru
10 = PartOfAHorizontalPortScan
11 = C&C-PartOfAHorizontalPortScan
12 = Attack

(e) Categorical Label Coding.

Fig. 3. Confusion matrices of 1D-CNN (a), HYBRID (b), MIMETIC (PAY+HDR) (c), and MIMETIC (NET+HDR) (d) with log scale to evidence small errors. Correspondence between classes and categorical labels (e).

On the contrary, DL architectures exhibit worse results. In fact, although HYBRID still achieves acceptable performance, reaching 94.88% accuracy and 58.53% F-measure, the results of 1D-CNN are remarkably lower. The possible reason is that most of the classes have several zero-payload packets and therefore the 1D-CNN, that is fed with PAY input type, hardly distinguishes the samples and can not classify them correctly.

This phenomenon is further evident from the confusion matrix in Fig. 3a: the 1D-CNN classifies many instances belonging to other classes as *PartOfAHorizontalPortScan* or *Attack* which are among the most represented ones (see Fig. 1). Interestingly, such misclassification patterns are exhibited also by the confusion matrix of HYBRID (Fig. 3b) despite less evident. Conversely, HYBRID shows a higher concentration of erroneous predictions toward *Benign* traffic w.r.t. 1D-CNN.

Nevertheless, MIMETIC (exploiting PAY+HDR input) achieves an accuracy of 95.73% and an F-Measure of 67.41%, witnessing the effectiveness of employing a multimodal classifier: MIMETIC attains an improvement of $\approx 9\%$ F-measure over HYBRID. However, since the results obtained with the 1D-CNN suggest that PAY input is not sufficiently informative for AC, we extend it by considering the first $N_b = 576$ B of network-layer header and payload (NET input, see Sec. III-A). Specifically, in Tab. II, we report the performance of MIMETIC fed with both NET+HDR and NET$\star$+HDR, namely obfuscating all the fields that could introduce bias or not, respectively (cf. Sec. IV-C). The NET+HDR unbiased configuration is then the one we consider for a fair evaluation of MIMETIC. Indeed, in such a case, the performance reached is much higher than that attained with the PAY input: MIMETIC provides an almost ideal accuracy, an F-measure of $\approx 92\%$, and a G-mean $> 93\%$.

The significant F-measure improvement (about $+25\%$) attained by MIMETIC when passing from having PAY+HDR to NET+HDR as input is also proven by comparing the confusion matrices in Figs. 3c and 3d. In fact, we can observe a remarkable reduction of error patterns when using NET+HDR, with predictions showing a higher concentration toward the diagonal (where predicted class equals the actual one).

TABLE III
PERFORMANCE MEASURES [%] WHEN OCCLUDING NET INPUT FIELDS.

Occluded Field	*Accuracy*	*F-Measure*	*G-mean*
No Occlusion	99.96	94.93	96.19
Src IP	66.77	43.11	55.41
Dst IP	86.57	41.67	47.14
Src port	94.06	90.90	93.19
Dst port	51.87	57.29	64.02
IP checksum	99.95	96.84	98.02
L4 checksum	99.95	92.11	94.16
Src & dst IPs	64.97	24.40	32.41
Src & dst ports	49.95	34.19	41.64
All fields	64.36	16.63	20.73

C. Occlusion Analysis

Occlusion analysis is a type of perturbation analysis that studies the effect of occluding part of the input on the output of DNNs. Herein, we investigate how AC performance of 1D-CNN changes when occluding part of NET input (i.e. the first N_b B of network-layer header and payload).[10] Operationally, for each packet we replace, at inference time, the value of one or multiple fields of the network-layer header/payload with zeros (i.e. a non-informative value). By observing how the AC performance changes, we can infer the impact of the field and also evaluate the bias it could introduce in the AC process.

Table III reports AC performance obtained when occluding one or more fields. Specifically, IP addresses and transport-layer ports are commonly considered as biased inputs that could inflate (traffic) classification performance [2]. IP and transport (viz. L4) checksums are also obfuscated because they are computed including the above-mentioned fields.

From the inspection of results we can notice that occluding source and destination IP addresses leads to the most significant performance drop. In fact, a certain source IP address is often (almost) exclusively associated to a specific attack, being the address of the infected device generating the malicious traffic, and thus could (erroneously) be representative for such an attack. Similar considerations are less straightforward for the destination IP address. In this regard, since biflows group packets sharing the quintuple regardless of their direction, the

[10]We do not perform occlusion analysis considering MIMETIC (NET+HDR) because directly evaluating the occlusion effect of the sole NET input would not have been possible.

field obfuscated depends on the packet direction. Therefore, when we obfuscate the destination IP address for upstream packets, we are still hiding an address that is uniquely associated with a given attack as discussed above. Hence, we evaluate also the results attained when occluding both IP addresses of each packet and observe a more significant performance drop, up to $\approx -24\%$ F-measure.

Analogous considerations apply to source and destination ports, with the former showing less impact on results. Indeed, while the set of destination ports has a reduced cardinality and some are related more frequently to certain attacks, the source port assumes almost all distinct values. In line with the observations regarding IP addresses, we can notice an equally significant performance degradation when obfuscating both ports. For checksums, we have smaller changes, probably because in this case the classifier can still exploit all other fields on which checksums are calculated. Finally, as expected, occluding all the above-mentioned fields causes the most significant performance drop down to 16% F-measure.

Such observations suggest that the fields considered in occlusion analysis influence AC performance since they carry information related to the capture scenario, witnessing the importance of not taking them into account for a fair evaluation.

V. Conclusions and future perspectives

IoT devices expose several vulnerabilities, thus the effective and timely identification of attacks against them calls for novel intelligent tools. In this paper, we performed AC via state-of-art DL approaches based on both single-modal and multimodal architectures, and compared them with more traditional ML approaches, encompassing both single and ensemble methods.

The experimental results exploit a dataset of malicious and benign traffic generated via real-hardware IoT devices. While ML classifiers achieve satisfactory results (up to 77% F-measure with RF), they are applicable only for "post-mortem" AC, as opposed to DL ones that show poorer performance when leveraging configurations derived from traditional TC (i.e. not properly targeted to the specific peculiarities of AC in IoT networks). Particularly, DL classifiers fed with transport-layer payload exhibit unsatisfactory AC results ($\approx 30\%$ F-measure). Hence, taking into account the specific nature of considered malicious/benign traffic, we exploit information extracted from network-layer header and payload, taking care to obfuscate some fields that could introduce bias in performance. Experimental results demonstrate that the multimodal MIMETIC approach fed with such an input is the best performing one in terms of all considered measures ($\approx 92\%$ F-measure), being also suited for "early" AC. The final occlusion analysis helped us to shed light on the working principles of DL architectures, highlighting also the impact of the fields that are known to introduce bias in results, witnessing the importance of not considering them in operational scenarios.

Future works will explore DL methods more in detail, in particular by evaluating hierarchical architectures where AC is done through a multi-stage approach. We plan also to apply Few-Shot Learning to AC to exploit its benefits in unbalanced datasets with limited samples for certain attack classes, and to extend the interpretability analysis to further investigate the behavior of DL approaches when facing AC.

References

[1] S. Garcia, A. Parmisano, and M. J. Erquiaga. (2020) Iot-23: A labeled dataset with malicious and benign iot network traffic. [Online]. Available: https://www.stratosphereips.org/datasets-iot23

[2] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Toward effective mobile encrypted traffic classification through deep learning," *Neurocomputing*, vol. 409, pp. 306–315, 2020.

[3] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapè, "MIMETIC: mobile encrypted traffic classification using multimodal deep learning," *Elsevier Computer Networks*, vol. 165, p. 106944, 2019.

[4] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21 954–21 961, 2017.

[5] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for internet of things," *Future Generation Computer Systems*, vol. 82, pp. 761–768, 2018.

[6] K. Wu, Z. Chen, and W. Li, "A novel intrusion detection model for a massive network using convolutional neural networks," *IEEE Access*, vol. 6, pp. 50 850–50 859, 2018.

[7] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on emerging topics in computational intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[8] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-iot dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019.

[9] R. Vinayakumar, M. Alazab, K. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, 2019.

[10] H. He, X. Sun, H. He, G. Zhao, L. He, and J. Ren, "A novel multimodal-sequential approach based on multi-view features for network intrusion detection," *IEEE Access*, vol. 7, pp. 183 207–183 221, 2019.

[11] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *Journal of Information Security and Applications*, vol. 50, p. 102419, 2020.

[12] M. M. Rashid, J. Kamruzzaman, M. M. Hassan, T. Imam, and S. Gordon, "Cyberattacks detection in iot-based smart city applications using machine learning techniques," *International Journal of Environmental Research and Public Health*, vol. 17, no. 24, p. 9347, 2020.

[13] M. Woźniak, J. Siłka, M. Wieczorek, and M. Alrashoud, "Recurrent neural network model for iot and networking malware threat detection," *IEEE Trans. on Ind. Info.*, vol. 17, no. 8, pp. 5583–5594, 2020.

[14] G. Bovenzi, G. Aceto, D. Ciuonzo, V. Persico, and A. Pescapé, "A hierarchical hybrid intrusion detection approach in iot scenarios," in *IEE GLOBECOM'20*, 2020, pp. 1–7.

[15] T. M. Booij, I. Chiscop, E. Meeuwissen, N. Moustafa, and F. T. den Hartog, "Ton_iot: The role of heterogeneity and the need for standardization of features and attack types in iot network intrusion datasets," *IEEE IoT Journal*, 2021.

[16] R. Kozik, M. Pawlicki, and M. Choraś, "A new method of hybrid time window embedding with transformer-based traffic data classification in iot-networked environment," *Pattern Analysis and Apps.*, pp. 1–9, 2021.

[17] A. K. Sahu, S. Sharma, M. Tanveer, and R. Raja, "Internet of things attack detection using hybrid deep learning model," *Computer Communications*, 2021.

[18] K. L. K. Sudheera, D. M. Divakaran, R. P. Singh, and M. Gurusamy, "Adept: Detection and identification of correlated attack stages in iot networks," *IEEE IoT Journal*, vol. 8, no. 8, pp. 6591–6607, 2021.

[19] I. Ullah and Q. H. Mahmoud, "Design and development of a deep learning-based model for anomaly detection in iot networks," *IEEE Access*, vol. 9, pp. 103 906–103 926, 2021.

[20] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *IEEE ISI'17*, 2017.

[21] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.