Employing Deep Ensemble Learning for Improving the Security of Computer Networks against Adversarial Attacks

Nowroozi, Ehsan; Mohammadi, Mohammadreza; Savas, Erkay; Mekdad, Yassine; Conti, Mauro

DOI
10.1109/TNSM.2023.3267831

Publication date
2023

Document Version
Final published version

Published in
IEEE Transactions on Network and Service Management

Citation (APA)
Nowroozi, E., Mohammadi, M., Savas, E., Mekdad, Y., & Conti, M. (2023). Employing Deep Ensemble Learning for Improving the Security of Computer Networks against Adversarial Attacks. *IEEE Transactions on Network and Service Management*, *20*(2), 2096-2105. https://doi.org/10.1109/TNSM.2023.3267831

Employing Deep Ensemble Learning for Improving the Security of Computer Networks Against Adversarial Attacks

Ehsan Nowroozi, *Senior Member, IEEE*, Mohammadreza Mohammadi, *Member, IEEE*, Erkay Savaş, *Member, IEEE*, Yassine Mekdad, *Member, IEEE*, and Mauro Conti, *Fellow, IEEE*

***Abstract*—In the past few years, Convolutional Neural Networks (CNN) have demonstrated promising performance in various real-world cybersecurity applications, such as network and multimedia security. However, the underlying fragility of CNN structures poses major security problems, making them inappropriate for use in security-oriented applications, including computer networks. Protecting these architectures from adversarial attacks necessitates using security-wise architectures that are challenging to attack. In this study, we present a novel architecture based on an ensemble classifier that combines the enhanced security of 1-Class classification (known as 1C) with the high performance of conventional 2-Class classification (known as 2C) in the absence of attacks. Our architecture is referred to as the 1.5-Class (cmb-classifier) classifier and is constructed using a final dense classifier, one 2C classifier (i.e., CNNs), and two parallel 1C classifiers (i.e., auto-encoders). In our experiments, we evaluated the robustness of our proposed architecture by considering eight possible adversarial attacks in various scenarios. We performed these attacks on the 2C and cmb-classifier architectures separately. The experimental results of our study showed that the Attack Success Rate (ASR) of the I-FGSM attack against a 2C classifier trained with the N-BaIoT dataset is 0.9900. In contrast, the ASR is 0.0000 for the cmb-classifier.**

***Index Terms*—Adversarial machine learning, counter-forensics, secure classification, deep-learning security, adversarial examples, adversarial attacks, ensemble classifiers, cybersecurity.**

Manuscript received 18 September 2022; revised 7 February 2023; accepted 12 April 2023. Date of publication 18 April 2023; date of current version 6 July 2023. The associate editor coordinating the review of this article and approving it for publication was D. Pezaros. *(Corresponding author: Ehsan Nowroozi.)*

Ehsan Nowroozi is with the Faculty of Engineering and Natural Sciences, Computer Engineering Department, Bahcesehir University, 34349 Istanbul, Turkey (e-mail: ehsan.nowroozi@eng.bau.edu.tr).

Mohamamdreza Mohammadi is with the Department of Mathematics, Security and Privacy Research Group, University of Padua, 35121 Padua, Italy (e-mail: mohammadreza.mohammadi@studenti.unipd.it).

Erkay Savaş is with the Faculty of Engineering and Natural Sciences, Sabanci University, 34956 Istanbul, Turkey (e-mail: erkays@sabanciuniv.edu).

Yassine Mekdad is with the Cyber-Physical Systems Security Lab, Department of Electrical and Computer Engineering, Florida International University, Miami, FL 33174 USA (e-mail: ymekdad@fiu.edu).

Mauro Conti is with the Department of Mathematics, Security and Privacy Research Group, University of Padua, 35121 Padua, Italy, and also with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: mauro.conti@unipd.it).

Digital Object Identifier 10.1109/TNSM.2023.3267831

I. Introduction

THE DEVELOPMENT of reliable Machine Learning (ML) techniques with a particular reference to Computer Networks applications is gaining popularity. Such technology can provide satisfactory performance in the face of an adversary trying to impede an accurate analysis. In this context, Digital Forensics tries to gather data on the history of documents, their authenticity, source, the process they underwent, and other factors. However, complicated forensics tasks often need statistical features and modeling. Therefore, forensic scientists have to consider different ML techniques. Whereas disabling ML/DL-based forensic analysis turns out to be a simple task. In this case, the adversary is trying to employ different Counter Forensics techniques to erase the main traces that the ML detector relies on for distinguishing between pristine and malicious examples [1], [2], [3], [4], [5]. In recent years, various Counter Forensics approaches have been developed to bypass forensics analysis and are often commonly referred to as adversarial attacks. As a consequence, the primary objective of an adversary is to impede detection by removing or altering evidence of illicit processing and making the counterfeit example look authentic [3], [4]. The information collected by the adversary concerning the machine to be attacked will have a significant impact on the adversary's strategies and the countermeasures employed by the forensic specialist to counter them [4]. In this study, we explore Computer Networks detection by considering different DL models, which is a forensic analysis used to identify if an example has been subject to any manipulation. Therefore, the analysis is often known as a binary decision, meaning that the examples are divided into two categories, referring to pristine and malicious examples. The most common approach in ML for this problem is to train the detector with both malicious and pristine examples. The objective of the training procedure is to divide the training examples so that they may be appropriately classified. Also, the detector should work well when encountering new situations that were not seen during training. The problem with the procedure described above is that the analyzer has to choose between two classes of cases, even when it comes across examples that are very different from those shown during the training phase, possibly because they do not belong to either.

(a) 2C (b) 1C (c) Cmb-classifier

Fig. 1. The 2C, 1C, and cmb-classifier's decision margins are visually presented. Red samples are considered pristine, whereas black dots signify malicious examples. The attacker attempts to relocate the black samples from the malicious region within the red region, a) The attacker takes full advantage of the availability of empty spaces to cause a missed detection error; as a result, the 2C classifier is vulnerable, b) Since they define a closed region enclosing samples from only one class, 1C classifiers are inherently more resilient to attacks with minor classification errors (a green circle in the figure highlights the examples that have a missed detection error), c) the cmb-classifier is designed to exhibit similar resilience against attacks as the 1C classifier, but the acceptance region is better formed than in the 1C classification scenario.

As soon as the examples fit into any of the aforementioned classifications, the classifier will return a true response depending on the instances seen during training. Given that these characteristics were absent during the training phase, the classifier would likely make a random decision. This is unlikely to be an issue under normal work conditions because the classifier model will never require handling such abnormal conditions. However, under adversarial settings, the adversary might use the availability of unpopulated areas of the example space to induce a low-cost classification error (we refer to it as distortion). Figure 1(a) depicts the above scenario, in which black example points are displaced with minimum distortion into an unpopulated red region. The abovementioned problem may be addressed using a 1C classifier when the adversary's goal is unidirectional [6]. In this case, only examples from the specific class are used to teach the ML how to decide if an example belongs to that class. The significant distinction between a conventional 2C and a 1C classifier is that the latter divides the example region into a closed region with examples from the class considered during training and a complement region including all additional examples. As depicted in Figure 1b, an attacker intending to move an example from the outside (black) region into the closed area containing the target class's examples (red region) can no longer take full advantage of unpopulated portions of the example area.

Given the high accuracy advantage of 2C classifiers and the robustness of 1C classifiers, as shown in Figures (a) and (b), a system that leverages the accuracy and robustness of 2C and 1C classifiers is required. As a result, the cmb-classifier's main objective is to maintain both benefits: (i) accuracy of detection, and (ii) robustness, as shown in Figure 1c. In this study, we propose the cmb-classifier, which has comparable robustness against numerous adversarial attacks as the 1C classifier has; however, the acceptance regions are significantly more formed in comparison to the 1C (see Figure 1c). Note that the accuracy in the absence of attacks is similar to the 2C classifier. This characteristic is proven in our study by considering two well-respected datasets: the N-BaIoT and RIPE datasets. Despite its security appeal, the 1C classifier has limitations. These constraints in 1C may not employ any information about the examples relating to a particular two classes. A 1C may be developed by evaluating only pristine examples (i.e., 1C-Leg), but this ignores details about the specific traces left in a malicious example. In contrast, a 1C classifier may be developed by evaluating only malicious examples (i.e., 1C-Mal). Consequently, in the absence of attacks, the efficiency of 1C classifiers is expected to be lower than that of a 2C architecture; however, when compared to the approach in [7], [8], we achieved good performance by including DL architectures. The authors in [7], [8] proposed a multi-classifier model based on a Support Vector Machine (SVM) that includes the benefits of both models, combining the greater accuracy of 2C classifiers with the inherent security of 1C classifiers. We considered this scenario and introduced the cmb-classifier, a novel architecture based on a combination of multiple DL architectures (e.g., CNN as a 2C classifier, pristine auto-encoder as one 1C-Leg, malicious auto-encoder as one 1C-Mal, and dense classifier). In Table I, we list all the acronyms and notations considered in our study.

A. *Contributions*

Our contributions are outlined in the following:

- We propose an ensemble architecture, known as the cmb-classifier, by employing deep ensemble learning in computer networks. The cmb-classifier architecture consists of four classifiers. Namely: a 2C classifier (i.e., CNN), a 1C classifier (i.e., trained with pristine examples), and a 1C classifier (i.e., trained with malicious examples), followed by a final dense classifier.
- We perform eight adversarial attacks with different parameters against 2C (i.e., scenario 1) and the cmb-classifier (i.e., scenario 2) classifiers separately: the I-FGSM, FGSM, JSMA, L-BFGS, PGD, BIM, DeepFool, and C&W attacks. We considered two well-known datasets in computer networks while performing these attacks: the RIPE-Atlas [9] and N-BaIoT [10] datasets.

TABLE I
LIST OF ABBREVIATION

Acronym	Description
CNN	Convolutional Neural Network
2C / 1C	2-Class Classification / 1-Class Classification
PK / LK	Perfect Knowledge / Low Knowledge
DL / ML	Deep Learning / Machine Learning
SVM	Support Vector Machine
GAN	Generative Adversarial Networks
IoT	Internet of Things
DDoS	Distributed Denial of Service
ASR	Attack Success Rate
Max. dist	Maximum distortion
PSNR	Peak Signal-to-Noise Ratio
L_1 dist	L_1 distance
FGSM	Fast-Gradient-Sign-Method
I-FGSM	Iterative-Fast Gradient-Sign-Method
JSMA	Jacobian-based Saliency-Map-Attack
LBFGS	Limited-Memory Broyden-Fletcher-Goldfarb-Shanno
PGD	Projected-Gradient-Descent
BIM	Basic-Iterative-Method
C&W	Carlini and Wagner

- The impact of the selected adversarial attacks on the resilience of the 2C classifier and the cmb-classifier architecture is systematically investigated. These attacks are considered during the testing phase, and we refer to them as exploratory evasion attacks.
- We compared the security of the cmb-classifier architecture with 2C. To elaborate more, we demonstrate that the ASR for the I-FGSM adversarial attack with the attack strength factor 0.1 is 0.9900 for 2C and 0.0000 for the cmb-classifier, respectively.

B. Organization

Our study is organized as follows. Section II review relevant studies that examined the 1C property in various tools in the cybersecurity domain. In Section III, we provide the experimental setup and present our proposed framework. Following that, we present the experimental findings in Section IV and discuss the adversarial strategies employed against the 2C classifier (i.e., CNN). We summarize our research and potential directions for further research in Section V.

II. Related Works

In this section, we provide related works on 1C classifiers in the cybersecurity domain.

Although the use of 1C classifiers in cybersecurity and Digital Forensics applications is not innovative, they may be found in various forensic tools. For example, in [11], the researchers utilized 1C for video forgery identification to provide an effective system in complex environments such as social networks. The scientists employed a technique based on *auto-encoders* trained on pristine data. When there is a considerable reconstruction error between both the outputs and inputs, *auto-encoders* behave like 1C. One class classification (known also as 1C) is typically used for outlier detection in a variety of scenarios if a robust statistical characterization under abnormal conditions is not available. Consider the problem of recognizing acoustic diversity [12] or predicting network intrusion [13]. Another work in [14] provides a strategy for an adversarial intrusion detection system that integrates several 1Cs to improve the complexity of exploratory attacks. In [8], the authors considered an ensemble model with the combination of SVM blocks to improve the security of a detector against adversarial attack. Although they achieved high security against adversarial attacks, a proposed model completely breaks against noise addition in the input.

The *Open set* problems, which have been investigated in various Digital Forensics as well as security-oriented tools, are another type of 1C classification wherein LK of the domain is provided at the training phase and unidentified classes might be presented to a method during testing [15], such as open set authentication of IoT [16], incremental open set intrusion detection [17], android malware detection [18], and DDoS attack detection [19]. Later, 1C classifiers were integrated with Generative Adversarial Networks (GANs) to build detectors that operate on the hypothesis that there are not many malicious examples available. This is true for [20], which is utilized for intrusion detection systems, as well as [21], which is used to identify unknown IoT infections.

Apart from considering 1C classification in various applications for improving detection accuracy and robustness, a second approach consists of building an architecture that might resist various attacks. In [22], the authors demonstrated the absence of attack transferability in Computer Networks, and only a few attacks are transferable between source and target networks. As a result, they explored several deep architectures as a target network to limit attack transferability. The major disadvantage of this method is that it only responds to a few adversarial attacks. In another related work, the authors in [23] developed a secure architecture using feature randomization to mitigate attack transferability between networks. The major disadvantage of this strategy is that it only works against a limited number of adversarial attacks, the same as previous research work. The system fails if the adversary considers an attack with a high attack parameter. In Table II, we provide an overview that summarizes past and present research on the various machine and deep learning applications in different applications that considers 1C classifiers. To the best of our knowledge, the security domain aspect in computer networks has not been considered in most published research in this area. We explore many application domains that used 1C classifiers. To create a model with high accuracy and security, we aim to combine the advantages of several classifiers for the first time in DL. Using prominent datasets as well as black-box scenarios, we also intend to carry out comprehensive evaluations.

III. Evaluation Methodology

In this section, we specify the detection task. Afterward, we discuss the cmb-classifier framework, including adversarial techniques on the 2C and 1C. Then, we describe the considered datasets as well as the cmb-classifier's training process.

In this study, we considered two class classifications for pristine and malicious examples: hypothesis H0 refers to the case of pristine examples provided and without any additional

TABLE II
DIFFERENCE BETWEEN OUR WORK AND EXISTING WORKS IN 1C CLASSIFIER

Application Domain	Classifier	Considered Datasets	Advantages	Disadvantages
Network Intrusion Detection [13]	-Bi-GAN	-NSL-KDD -CIC-DDoS2019	-Less training overhead	-High false positive rate
Open set [16]	-CNN -OpenMAX	-ADS–B	-Applicable on real world problems	-Low robustness of the features
Android malware detection [18]	-Different classifiers	-Private Dataset	-Increase high detection rate	-Sensitive to adversarial attacks
Intrusion detection [19]	-DNN -CNN -LSTM	-CIC-DDoS2019	-Good inference time -Smaller number of trainable parameters	-Sensitive to adversarial attacks
Computer Networks [22]	-CNN	-N-BaIoT -DGA -RIPE Atlas	-Considering adversarial attacks	-Responds to a few adversarial attacks
This study	-CNN -Auto-encoder	-N-BaIoT -RIPE	-Improve a Computer Networks security -Robust against noise addition	-Need to investigate against causative attacks (future research work)

processing. In contrast, hypothesis H1 refers to the case of modified or altered examples. In this case, in a real-world scenario, the attacker is constantly eager to apply adversarial attacks to H1 to prevent a correct detection. In this paper, we consider the most well-known and widely used adversarial attacks against CNN (2C) models in the DL literature and apply them to the cmb-classifier architecture to evaluate its robustness. All of these strategies are applicable in both white-box and black-box scenarios. These attacks include the FGSM attack, the I-FGSM attack, the JSMA attack, the L-BFGS attack, the PGD attack, the BIM attack, the DeepFool attack, and the C&W attack. We note that these attacks have been discussed in detail in previous studies [22]. In addition, we suppose that the attacker's purpose is to evade detection of the manipulation, i.e., to cause a missed detection error. In Counter Forensics, the most standard attack is an integrity violation attack [24]. With P_{MD} and P_{FA}, respectively, we can represent the probabilities of a missed detection error, which is the possibility that a malicious example will be misidentified for a clean example, and a false alarm probability, which is the possibility that a clean example will be wrongly identified as a tampered example.

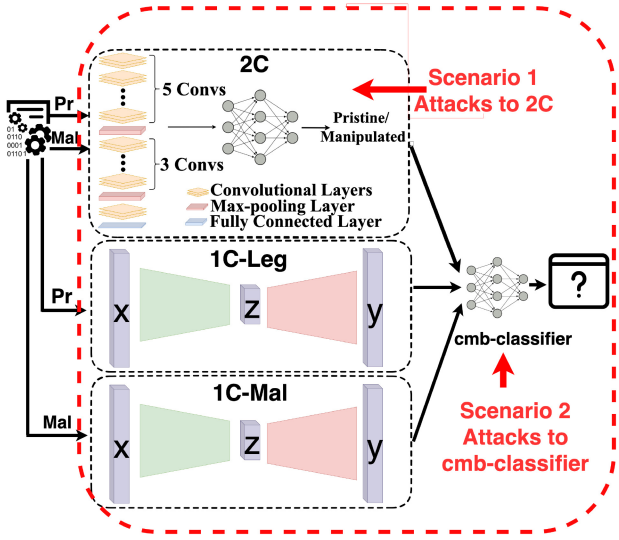


Fig. 2. The framework of the cmb-classifier employed in this study is as follows. The input data contains both pristine and malicious examples. In this diagram, we considered CNN architecture as a 2C, considering auto-encoder "1C-Leg" trained with pristine examples, auto-encoder "1C-Mal" trained with malicious examples, and final dense classifier, also known as a classifier combination. Consequently, the outputs of these three classifiers are employed to train a dense classifier.

A. Architecture of the cmb-Classifier

The cmb-classifier's architecture used in this study by considering the ensemble method is shown in Figure 2. Three classifiers were trained in parallel using dataset examples: a 2C (here, CNN) trained with pristine and malicious instances from both classes from the N-BaIoT and RIPE-Atlas datasets independently, and two 1Cs (here, two auto-encoders), one trained with pristine and the other with malicious examples. The results of these classifiers are subsequently processed by a final dense classifier, which makes a final decision.

As stated in the introduction, in the absence of attacks, 2C classification techniques can achieve high accuracy; nevertheless, they do not generalize properly to examples that were improperly represented during the training process, enabling the adversary to carry out his attack by exploiting the unexplored areas of the features space. This scenario is illustrated in Figure 1a, where the adversary takes full advantage of the availability of empty spaces to induce a missed detection error, exposing the 2C vulnerable to attacks. However, 1C is inherently more robust to adversarial attacks since they provide a closed region that only includes examples through a specific class, commonly called the H0 class. Figure. 1b illustrates this impact, showing that a larger distortion is required to move an example from the H1 (malicious examples) region to the H0 (pristine examples) region. As a consequence, the acceptance region in Figure 1c (referred to as cmb-classifier) has comparable robustness and performance against adversarial attacks as those obtained by the 1C (see Figure 1b) and 2C classifier (see Figure 1c). To clarify more regarding Figure 2, we present two possible scenarios:

- *Scenario 1:* The attacker exploits several adversarial attacks against the 2C classifier. Afterward, we analyze the performance of all classifiers using adversarial examples generated by the 2C classifier.
- *Scenario 2:* In this scenario, the attacker performs a variety of adversarial attacks against the entire secure cmb-classifier's architecture.

B. Network Architecture

We adopted the network configuration provided by [25] for the 2C classifier. This network has nine convolutional layers

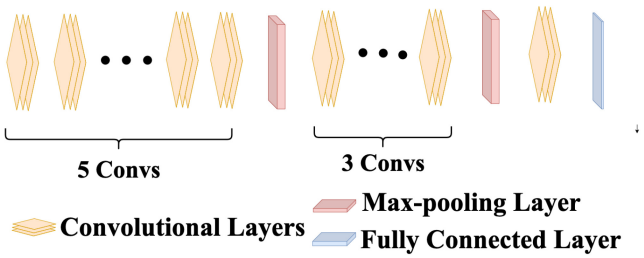


Fig. 3. The proposed 2C (CNN) architecture's pipeline.

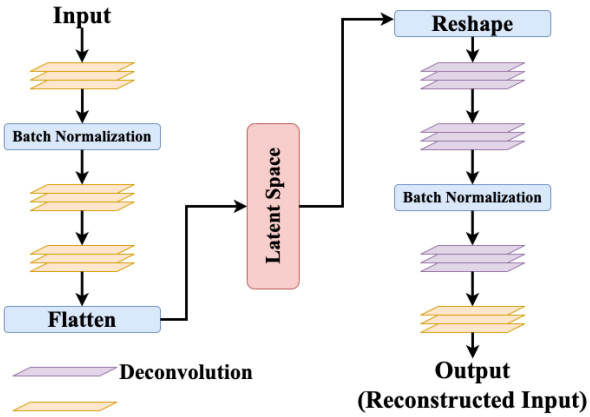


Fig. 4. The proposed 1C (auto-encoders) architecture's pipeline.

followed by max-pooling layers, and a dense layer followed by a sigmoid layer. The 2C architecture's process is depicted in Figure 3. For all convolutions, we employ a kernel size of 3×3 and a stride of 1, and for max-pooling with a stride 2 with a kernel size of 2×2. Only one dense layer is employed, which reduces the number of parameters. The network has several convolutions to transmit every neuron before the first max-pooling layer effectively. Adjusting the stride to one, the best spatial information will remain. We employ this network configuration because it offers a high level of accuracy over training.

For 1C-Leg and 1C-Mal auto-encoders, we employed a simple CNN-based auto-encoder. We used convolutional layers, batch normalization layers, and a fully-connected layer for the encoding part. For decoding, we adopted a fully-connected layer, de-convolutional (transposed convolutions) layers, a batch normalization layer, and a convolutional layer. In addition, we have a latent space between the encoder and the decoder, which has just one fully-connected layer. The latent space is the output of the encoder and is used as input for the decoder. In the encoder, the network has some convolutions to capture the most important features of the input samples. For decreasing the size of feature maps, the strides of all convolutions were set to two. To build a compressed representation of input samples, auto-encoders can be utilized. For this reason, we exploit this network architecture because it allowed us to have extremely good similarity for the original input and network output, i.e., reconstructed input using latent space features. The 1C model is illustrated in Figure 4. We set 3×3 kernel sizes and strides of two for all convolutions and deconvolutions. For the latent space, we considered a dense layer with a size of 512 that may help us to increase the effectiveness of auto-encoders in the cmb-classifier.

Three dense layers are considered to perform the classification task for the dense network. The first dense layer is created by concatenating the extracted flatten layer of the 2C classifier with the size of 1728 to latent spaces, i.e., the output of the encoder part of two 1C classifiers that size of each one is 512. The concatenation task will generate a dense layer with a size of 2752 neurons. As we have two classes (pristine and malicious), layers of size 128 and 2 are considered for the second and third dense layers, respectively, which are suitable for our classification task.

C. Datasets

We employed large real-world datasets comprising pristine and malicious examples to assess the effectiveness of the cmb-classifier capability in computer networks. Our study utilized the N-BaIoT and RIPE datasets to train the CNN model (2C) and auto-encoders (1Cs). This well-known dataset has been used to achieve various cybersecurity objectives lately. Employing deep auto-encoders [26], the N-BaIoT dataset was also utilized to monitor botnet attacks in the Internet of Things (IoT) devices, IoT detection methods [27], and Federated Learning for identifying and mitigating IoT attacks [28]. The following paragraphs provide a detailed explanation of the dataset.

N-BaIoT dataset: This dataset combines malicious and benign traffic from nine business IoT devices using port mirroring. The N-BaIoT dataset contains approximately seven million examples with 115 properties. As a result of the evolution of IoT cyber-attacks, the adversary relies on botnets to exploit such vulnerabilities, transforming the IoT into a vulnerable Internet [29].

RIPE-Atlas dataset: The RIPE Atlas project evaluates network packets using Internet-aware sensors [30]. It is used to continually monitor network or client visibility from various locations. Furthermore, it can perform ad-hoc connection assessments to examine the network, fix any faults discovered, and check the DNS server availability.

Utilizing traffic data, the RIPE dataset first attempts to determine the type of applications. Two distinct architectures are being considered to explore these datasets: Recurrent Neural Networks and CNN.

D. Experimental Setup

We employed 29000 samples from the N-BaIoT dataset for training, 10000 for verification, and 10000 for testing, per class to train a 2C classifier. We employed the same dataset split strategy for training 1C-Leg, where only pristine examples were considered, while only malicious examples were considered for 1C-Mal. The dense classifier (also known as a cmb-classifier) is fed by concatenating a flattening layer from a 2C and two latent layers from 1C-Leg and 1C-Mal. In this regard, the 2C flatten layer size is 1728, the latent layer size from the 1C-Leg is 512, and the 1C-Mal is 512. As a consequence, the input size fed to the cmb-classifier is 2752. Concerning the RIPE dataset, we considered 30000

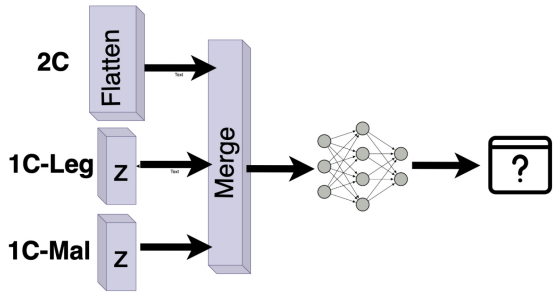


Fig. 5. The cmb-classifier training pipeline.

examples for training a 2C classifier, 10000 for validation, and 10000 for testing, per class. We utilized the same dataset split strategy for training 1C-Leg that we employed for N-BaIoT, where only pristine examples were considered, whereas only malicious examples were considered for 1C-Mal. Furthermore, the cmb-classifier size is 2752, the same as the N-BaIoT cmb-classifier.

To clarify the training of the cmb-classifier, we utilized the features from the flattening layer (2C) and the features from the latent space of auto-encoders (1Cs). Figure 5 exhibits the training pipeline considered for the cmb-classifier.

All features are then merged and structured into a dense network. In all cases, the input size is always 64×64. We believe that this set of features is sufficient to generalize a CNN, auto-encoders, and a dense network. We employed TensorFlow and Python to create our classifiers, using the Keras API. In our experiments, we employed the Intel CoreTM i7 processor 10750H, the GeForce NVIDIA 2060 RTXTM with GDDR6 6GB GPU, and DDR4 32GB, and Ubuntu operating system version 20.04 were used in our tests. We provided the Python code for the simulation on Github [31]. For the cmb-classifier, 500 training cycles have been carried out across all classifiers. We used the Adam solution with a momentum of 0.99 and a learning rate ($lr = 10^{-4}$). For training and validation, the batch size for the N-BaIoT and RIPE datasets is 16.

E. Security Assessment

The cmb-classifier's security is evaluated by examining the study's validity under adversarial attacks. Compared to those achieved by 2C under the same attacks, these outcomes are close to each other. To prove that the cmb-classifier model provides better security than the 2C classifier, this study aims to demonstrate that system compromise causes more distortion in the scenarios that are being attacked. We especially took into account the threat model below:

- The **attacker's objective** is to alter a malicious example so that the feature representation is transferred into the pristine area, leading to a missed detection error.
- By using terminology from [7], we highlight a PK where the adversary has full knowledge of the model. This refers to considering the **attacker's knowledge** scenario.
- In terms of **attacker capabilities**, we concentrate on exploratory attacks [32], that is, attacks conducted during the testing phase. This category comprises most of the Counter Forensics approaches proposed in the literature.

TABLE III
TEST ACCURACY AND PRECISION VALUES OF ALL THE CLASSIFIERS

	Dataset	2C	1C-Leg	1C-Mal	Cmb-classifier
Test Accuracy	N-BaIoT	99.98%	98.86%	98.80%	100%
	RIPE	99.76%	98.32%	98.51%	100%
Precision	N-BaIoT	99.86%	98.73%	98.78%	99.99%
	RIPE	99.80%	98.28%	98.32%	99.99%

IV. EXPERIMENTAL RESULTS AND DISCUSSION

We present experimental results in 2C and the cmb-classifier separately in this section. First, we present the accuracy rate of all classifiers in the absence of attacks. Then, we conduct several adversarial attacks against 2C and evaluate the performance of each classifier. Finally, we employed the same adversarial attacks on the cmb-classifier in a black-box setting to understand its robustness fully. All experiments were carried out using two datasets, N-BaIoT and RIPE.

A. Experimental Results

Table III provides the test accuracy and precision values of the 2C, 1C pristine, 1C malicious, and the combination dense classifier (acronym to cmb-classifier). We see that the performance of the cmb-classifier increase slightly compared to 2C.

The average ASR on 2C based on CNN networks and dense networks utilized in the cmb-classifier are represented in the results when we applied different adversarial attacks. In addition, we took into account the average PSNR on 500 samples from the test folder, the average L1 distortion (L_1 dist), and the average maximum absolute distortion (Max. dist) as other metrics. In addition, we provide running attack timings in seconds while performing adversarial assaults on CNN and dense networks.

For the I-FGSM attacks, we considered the strength attack factors ε to 0.1, 0.01, and 0.001, and we employed the same attack parameters for the FGSM. We fixed the number of steps S to 10. The strength factor θ for JSMA adversarial attack is set at 0.1 and 0.01. In the case of other attacks such as DeepFool, LBFGS, BIM, and PGD, we considered the default attack variable, which we believe is sufficient to fool a 2C network. Finally, we looked into C&W adversarial attacks with various confidence factors such as 0, 50, and 100.

1) Performance Under Attacks: In this part, we evaluate the effectiveness of the 2C and the cmb-classifier in the face of attacks. We started by deploying all attacks on 2C and then evaluated the cmb-classifier-1.5C to see how it performed. Subsequently, to evaluate the security level of the cmb-classifier, we conducted adversarial attacks directly on it.

Attack against 2C (Scenario 1): For detecting task N-BaIoT and RIPE pristine examples from malicious examples, we first conduct the different adversarial attacks against 2C to demonstrate that this classifier is intrinsically sensitive to all adversarial attacks. As predicted, the attack was consistently successful in producing an incorrect classification, and after running, all malicious examples are classified as pristine. Table IV, and V reported when the eight different adversarial attacks were employed against the 2C classifier and when

TABLE IV
EXPERIMENTAL RESULTS FOR ADVERSARY ATTACKS ON A 2C (TRAINED WITH N-BAIOT DATASET). THE TIME OF RUNNING ATTACKS ON THE MODELS IS REPORTED IN SECONDS IN THIS TABLE

Attack Type	PSNR	L_1 dist	Max. dist	ASR	Exe. Time
I-FGSM, ε= 0.1	17.617	29.691	40.733	0.9900	315.518
I-FGSM, ε= 0.01	17.996	28.557	39.101	0.9500	3044.2015
I-FGSM, ε= 0.001	17.997	28.410	38.927	0.9463	29805.6208
FGSM, ε= 0.1	17.277	30.856	40.712	0.9800	29.6365
FGSM, ε= 0.01	17.414	30.396	40.098	0.9756	46.8270
FGSM, ε= 0.001	17.546	29.931	39.470	0.9500	152.3631
JSMA, θ= 0.1	17.432	7.025	178.500	0.9900	3513.4855
JSMA, θ= 0.01	**Fails**	**Fails**	**Fails**	**Fails**	**Fails**
DeepFool, Default	23.489	7.162	175.386	1.0000	221.2673
LBFGS, Default	24.145	7.143	132.766	0.9700	2825.4893
BIM, Default	17.905	28.823	38.980	0.9800	559.6410
PGD, Default	17.915	28.876	38.747	0.9800	2279.5689
C&W, c = 0	24.130	6.981	131.851	1.0000	9012.3253
C&W, c = 50	23.60	7.099	127.714	1.0000	7968.1794
C&W, c = 100	23.50	7.100	127.8402	1.0000	7464.7506

TABLE V
EXPERIMENTAL RESULTS FOR ADVERSARY ATTACKS ON A 2C (TRAINED WITH RIPE DATASET). THE TIME OF RUNNING ATTACKS ON THE MODELS IS REPORTED IN SECONDS IN THIS TABLE

Attack Type	PSNR	L_1 dist	Max. dist	ASR	Exe. Time
I-FGSM, ε= 0.1	16.415	27.541	40.883	0.9900	423.618
I-FGSM, ε= 0.01	16.763	26.998	40.765	0.9863	5440.1006
I-FGSM, ε= 0.001	16.761	26.986	39.601	0.9556	39805.1309
FGSM, ε= 0.1	16.107	28.733	38.510	0.9900	35.5140
FGSM, ε= 0.01	16.212	28.190	38.190	0.9856	56.9280
FGSM, ε= 0.001	16.440	27.721	36.255	0.9800	250.1616
JSMA, θ= 0.1	16.322	7.125	187.652	0.9832	4510.3905
JSMA, θ= 0.01	**Fails**	**Fails**	**Fails**	**Fails**	**Fails**
DeepFool, Default	39.515	8.252	180.165	1.0000	301.1695
LBFGS, Default	29.450	8.120	143.855	0.9800	39215.4990
BIM, Default	20.905	29.763	40.129	0.9900	439.5009
PGD, Default	20.816	30.521	40.556	0.9700	3289.4699
C&W, c = 0	22.140	5.871	132.751	1.0000	8212.5360
C&W, c = 50	20.41	6.098	136.604	1.0000	7968.2414
C&W, c = 100	20.52	6.510	136.6512	1.0000	7464.7706

the models were trained using the N-BaIoT and RIPE-Atlas datasets.

The findings in this table relate to the average ASR on 2C, average PSNR on 500 examples, average *L1* distortion, average maximum absolute distortion (Max. dist), and average running attacks time in seconds, as was stated earlier. The adversarial attack was effective if the average ASR threshold was higher than 50%.

The I-FGSM attack has a predefined number of steps *S* for attacks, which is 10 where SSS, which refers to search step size and is related to the optimum strength of attacks; therefore, the optimal strength is considered in the [ϵ_s : 0.1, 0.01, and 0.001] range. A powerful attack is often considered when setting a higher ϵ_s, but we need to apply high distortion in the input example; therefore, we will have a high PSNR. In our studies, we examined ϵ_s = 0.1, 0.01, and 0.001, which mislead a 2C architecture also that average PSNR remains around 17db, implying that attacks must apply higher distortion. This attack scenario also happens for the FGSM; even though there are no steps in this attack, it can mislead a network with a close average PSNR, similar to I-FGSM. The BIM technique is an enhancement of the FGSM approach. The approach is to iteratively adjust the value one step at a time and prune the resulting value to verify that it is within the prescribed range of the original sample. Regarding BIM, we investigated using a default parameter since the default parameter is sufficient to deceive the 2C classifier. PGD adversarial attack, similar to FGSM, is concerned with determining the perturbation that maximizes the loss function under given *l* distortion constraints. Regarding PGD, we considered the default parameter, which we believe is sufficient to mislead a 2C architecture.

The JSMA's parameter T is set at 7. The method's number of iterations is set to 2000 by default. The relative alternation per pixel is fixed at 0.01 and 0.1. We did not examine parameters lower than 0.01 since JSMA already fails to deceive a 2C architecture with a parameter of 0.01. We utilized the default attack setting for L-BFGS, which attempts to estimate the optimal solutions of the optimization method that the attack should handle to determine the lowest perturbation in an adversarial example that causes the error in predictions (typical gradient-descent approach).

In DeepFool, adversarial attacks initialize the sample confined by a classifier's decision boundaries. The area of the decision boundaries determines the sample's class designation. A small vector is conducted with a sample approximated by the polyhedron's boundary in each cycle. The perturbations are then carried out to an example in each iteration to determine the overall perturbation based on initial decision classifier limitations. Regarding C&W adversarial attack, C&W claims that a series of attacks evaluate norm-restricted additive perturbations and completely destroy defensive distillation. It is further shown that when the perturbation is produced using an exposed white-box model, their attack effectively deceives a network that has been defensively distilled under black-box settings. The results in Table IV and Table V show that both attacks can mislead a 2C architecture. In reality, we investigated both attacks in this study since recent studies show that both are among the most powerful attacks in DL.

Performance of 2C attacks against the cmb-classifier and Auto-encoders: In the previous section, we conducted various attacks on the 2C classifier and concluded that most attacks fully fooled a detector. After performing attacks to 2C with 500 samples, we kept attack samples separately to test the cmb-classifier since we wanted to know if the cmb-classifier could determine whether these samples were attack samples. The results of 2C attack samples against the cmb-classifier are shown in Table VI for the N-BaIoT dataset and Table VII for a RIPE dataset. We observe that the attacked samples we achieved against 2C are ineffective since most of the cmb-classifier test accuracy is close to 0%.

Additionally, the architecture of the cmb-classifier includes two auto-encoders, one of which is trained with pristine samples while the other is trained with malicious samples (see

TABLE VI
THE PERCENTAGE OF ATTACKED CASES THAT WERE MISCLASSIFIED. THE ATTACK IS DIRECTED EMPLOYED AT 2C (N-BAIOT DATASET)

Attack Type	1C-Leg	1C-Mal	cmb-classifier
I-FGSM, ε = 0.1	0.0020	0.0100	0.0000
I-FGSM, ε = 0.01	0.0001	0.0020	0.0000
I-FGSM, ε = 0.001	0.0000	0.0000	0.0000
FGSM, ε = 0.1	0.0050	0.0010	0.0000
FGSM, ε = 0.01	0.0001	0.0030	0.0000
FGSM, ε = 0.001	0.0000	0.0000	0.0000
JSMA, θ = 0.1	0.0011	0.0150	0.0000
JSMA, θ = 0.01	**Fails**	**Fails**	**Fails**
DeepFool, Default	0.1812	0.1200	0.0000
LBFGS, Default	0.0010	0.0002	0.0000
BIM, Default	0.0050	0.0001	0.0000
PGD, Default	0.0005	0.0000	0.0000
C&W, c = 0	0.0004	0.0000	0.0001
C&W, c = 50	0.0018	0.0021	0.0061
C&W, c = 100	0.1012	0.1001	0.0070

TABLE VII
THE PERCENTAGE OF ATTACKED CASES THAT WERE MISCLASSIFIED. THE ATTACK IS DIRECTED EMPLOYED AT 2C (RIPE DATASET)

Attack Type	1C-Leg	1C-Mal	cmb-classifier
I-FGSM, ε = 0.1	0.0040	0.0031	0.0000
I-FGSM, ε = 0.01	0.0006	0.0010	0.0000
I-FGSM, ε = 0.001	0.0000	0.0000	0.0000
FGSM, ε = 0.1	0.0300	0.0020	0.0000
FGSM, ε = 0.01	0.0020	0.0010	0.0000
FGSM, ε = 0.001	0.0000	0.0000	0.0000
JSMA, θ = 0.1	0.0800	0.0030	0.0000
JSMA, θ = 0.01	**Fails**	**Fails**	**Fails**
DeepFool, Default	0.0240	0.0120	0.0030
LBFGS, Default	0.0300	0.0010	0.0000
BIM, Default	0.0050	0.0001	0.0000
PGD, Default	0.0009	0.0010	0.0000
C&W, c = 0	0.0081	0.0030	0.0010
C&W, c = 50	0.0041	0.0033	0.0051
C&W, c = 100	0.2021	0.2201	0.0084

Figure 2). For ease of use, we refer to auto-encoders trained on pristine samples as 1C-Leg and those trained on malicious samples as 1C-Mal. Given that we obtained the adversarial samples from the 2C architecture, we tested 1C-Leg and 1C-Mal against these attacks. Attacks over 1C-Leg and 1C-Mal are ineffective, demonstrating that using a constrained acceptance area makes the 1C more resistant to adversarial attacks (see Figure 1b). As a result, the attack is inefficient in causing an inaccurate classification for the cmb-classifier (see Figure 1c) but efficient in causing a missed classification error in 2C (see Figure 1a).

According to Table VI and Table VII, attacking 2C is insufficient to deceive the cmb-classifier as well as both 1C classifiers. According to the values in these tables, the misclassification rate in the cmb-classifier is 0% in most adversarial attacks, and only in C&W, the misclassification rates are 0.0001, 0.0061, and 0070. To better understand, for the I-FGSM adversarial attack in Table VI with a parameter of 0.1, we evaluate all classifiers using attack examples given by 2C; thus, testing 1C-Leg, the misclassification rate is 0.0020, 1C-Mal is 0.0100, and the cmb-classifier is 0.0000. Furthermore, we have seen similar behavior when the cmb-classifier's architecture is trained on a RIPE dataset. As a result, as shown in Table VII, we assess all classifiers using attack examples provided by 2C; so, evaluating 1C-Leg, the misclassification rate is 0.0040, 1C-Mal is 0.0031, and the cmb-classifier is 0.0000. Thereby, we can state that 1C-Leg and 1C-Mal enhance the cmb-classifier in improving security.

TABLE VIII
EXPERIMENTAL RESULTS FOR ADVERSARY ATTACKS ON THE CMB-CLASSIFIER (TRAINED WITH N-BAIOT DATASET). THE TIME OF RUNNING ATTACKS ON THE MODELS IS REPORTED IN SECONDS IN THIS TABLE

Attack Type	PSNR	L_1 dist	Max. dist	ASR	Exe. Time
I-FGSM, ε = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	406.415
I-FGSM, ε = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	4144.1005
I-FGSM, ε = 0.001	**Fails**	**Fails**	**Fails**	**Fails**	37605.4521
FGSM, ε = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	49.4734
FGSM, ε = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	66.8612
FGSM, ε = 0.001	**Fails**	**Fails**	**Fails**	**Fails**	451.2393
JSMA, θ = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	4313.7812
JSMA, θ = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	6349.2156
DeepFool, Default	17.690	13.477	255.000	0.0020	87.4947
LBFGS, Default	19.74	9.670	253.529	0.4100	4915.2149
BIM, Default	**Fails**	**Fails**	**Fails**	**Fails**	450.3401
PGD, Default	**Fails**	**Fails**	**Fails**	**Fails**	6267.9812
C&W, c = 0	19.41	9.796	239.149	0.3200	48992.2740
C&W, c = 50	19.20	9.532	212.341	0.3122	47856.1244
C&W, c = 100	19.12	9.942	239.597	0.3000	47966.0449

Attack against the cmb-classifier (Scenario 2): We have launched adversarial attacks against 2C and evaluated the misclassification rate of the 1C-Leg, 1C-Mal, and the cmb-classifier. We have discovered that attacks in 2C are insufficient to mislead the entire architecture. In this scenario, we will analyze the security of the cmb-classifier model to determine if the attacker can deceive this model or become fail.

The experimental results of the attacks on the cmb-classifier classifier when trained with an N-BaIoT dataset are shown in Table VIII and in Table IX while trained with a RIPE dataset. In this scenario, the attack usually requires more iteration to enter the pristine region. However, ASR demonstrates that even with high attack iteration, the cmb-classifier is still secure against various adversarial attacks since most of the adversarial attacks completely fail to fool a model. According to the experimental results obtained with the N-BaIoT dataset in Table VIII, all attacks, including I-FGSM, FGSM, JSMA, BIM, and PGD, fail to mislead a network. Only DeepFool, LBFGS, and CW adversarial attacks can fool the cmb-classifier; as we indicated, if ASR is less than 50%, the system

TABLE IX
EXPERIMENTAL RESULTS FOR ADVERSARY ATTACKS ON THE CMB-CLASSIFIER (TRAINED WITH RIPE DATASET). THE TIME OF RUNNING ATTACKS ON THE MODELS IS REPORTED IN SECONDS IN THIS TABLE

Attack Type	PSNR	L_1 dist	Max. dist	ASR	Exe. Time
I-FGSM, ε = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	521.405
I-FGSM, ε = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	5304.2130
I-FGSM, ε = 0.001	**Fails**	**Fails**	**Fails**	**Fails**	66700.5011
FGSM, ε = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	69.0314
FGSM, ε = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	80.9420
FGSM, ε = 0.001	**Fails**	**Fails**	**Fails**	**Fails**	372.1473
JSMA, θ = 0.1	**Fails**	**Fails**	**Fails**	**Fails**	6200.8241
JSMA, θ = 0.01	**Fails**	**Fails**	**Fails**	**Fails**	9215.1270
DeepFool, Default	18.60	14.515	255.100	0.0041	91.3950
LBFGS, Default	20.43	10.310	250.412	0.2010	3420.1860
BIM, Default	20.00	10.421	251.326	0.0020	310.4514
PGD, Default	22.12	12.67	250.120	0.1012	7130.8523
C&W, c = 0	18.21	8.142	221.020	0.2200	48992.2740
C&W, c = 50	18.10	8.411	214.190	0.2101	43871.0109
C&W, c = 100	18.00	8.820	231.475	0.2000	48540.1309

is still secure. To elaborate more, in C&Ws, the ASR is close to 30%. In only one attack with LBFGS, the ASR is 40%, which is still below 50%. As a result, we can state that 1C-Leg and 1C-Mal are quite effective in strengthening the robustness of the cmb-classifier, indicating that the 1C classifiers are more difficult to attack as a consequence of the use of a closed acceptance region. Furthermore, based on the execution running time, attacks, even with a significant distortion into the examples, fail to mislead the cmb-classifier.

According to the experimental results obtained with a RIPE dataset, in Table IX, the cmb-classifier is still robust against various adversarial attacks. All attacks, including I-FGSM, FGSM, and JSMA, completely fail to deceive a network. In Table IX, for the adversarial attacks such as DeepFool and BIM attacks, the ASR is close to 0%, implying that the network is fairly secure against these attacks. On the other hand, with LBFGS, PGD, and C&Ws attacks, ASR is close to 20%, meaning that only 20% of attack examples were misclassified, so the ASR is lower than 50%, suggesting that the system remains secure.

V. CONCLUSION AND FUTURE WORK

According to recent studies, most ML and DL methods are intrinsically vulnerable and fragile to adversarial attacks, posing new serious security threats to cybersecurity tools. The study of the security of ML and DL-based methods in the presence of an adversary becomes important. As a result, in this study, we proposed the cmb-classifier, a multi-classifier architecture, to mitigate the damage caused by an adversary in a PK scenario. Our classification technique successfully takes the benefits of 2C and 1C techniques, providing better security while maintaining 2C classification's exceptional performance in the absence of attacks. We trained our classifier to discriminate between pristine and malicious examples using the prominent datasets namely: N-BaIoT and RIPE-Atlas. The experimental results demonstrated the robustness of the cmb-classifier against several adversarial attacks. In our study, we opted for exploratory types of attacks over causative attacks since most of the Counter Forensics attacks that have been provided currently fit into the category of exploratory attacks.

In future work, we aim to improve the cmb-classifier by employing a block-GAN combination. As a result, it would be interesting to identify the most suitable GAN network for this research study. Another possible future study direction is the development of backdoor or poisoning attacks that adversaries may employ to interrupt the training phase, often known as causative attacks.

REFERENCES

[1] T. Gloe, M. Kirchner, A. Winkler, and R. Böhme, "Can we trust digital image forensics?" in *Proc. 15th ACM Int. Conf. Multimedia*, New York, NY, USA, 2007, pp. 78–86.

[2] R. Böhme and M. Kirchner, "Counter-forensics: Attacking image forensics," in *Digital Image Forensics*. New York, NY, USA: Springer, 2013, pp. 327–366.

[3] E. Nowroozi, M. Barni, and B. Tondi, "Machine learning techniques for image forensics in adversarial setting," Ph.D. dissertation, Dept. Inf. Eng. Math., Univ. Siena, Siena, Italy, 2020.

[4] E. Nowroozi, A. Dehghantanha, R. M. Parizi, and K.-K. R. Choo, "A survey of machine learning techniques in adversarial image forensics," *Comput. Secur.*, vol. 100, Jan. 2021, Art. no. 102092.

[5] E. Nowroozi, Abhishek, M. Mohammadi, and M. Conti, "An adversarial attack analysis on malicious advertisement URL detection framework," *IEEE Trans. Netw. Service Manag.*, early access, Nov. 28, 2022, doi: 10.1109/TNSM.2022.3225217.

[6] S. Kang, "Using binary classifiers for one-class classification," *Expert Syst. Appl.*, vol. 187, Jan. 2022, Art. no. 115920.

[7] B. Biggio et al., "Evasion attacks against machine learning at test time," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discov. Databases*, 2013, pp. 387–402.

[8] M. Barni, E. Nowroozi, and B. Tondi, "Improving the security of image manipulation detection through one-and-a-half-class multiple classification," *Multimedia Tools Appl.*, vol. 79, no. 3, pp. 2383–2408, 2020.

[9] "RIPE-atlas-community/ripe-atlas-data-analysis." Accessed: Apr. 22, 2023. [Online]. Available: https://github.com/RIPE-Atlas-Community/RIPE-Atlas-data-analysis

[10] "UCI machine learning repository: Detection_of_IoT_botnet_attacks_N_BaIoT data set." Accessed: Apr. 22, 2023. [Online]. Available: https://archive.ics.uci.edu/ml/datasets/detection_of_IoT_botnet_attacks_N_BaIoT

[11] D. D'Avino, D. Cozzolino, G. Poggi, and L. Verdoliva, "Autoencoder with recurrent neural networks for video forgery detection," *Electron. Imag.*, no. 7, pp. 92–99, 2017.

[12] E. Marchi, F. Vesperini, F. Eyben, S. Squartini, and B. Schuller, "A novel approach for automatic acoustic novelty detection using a denoising autoencoder with bidirectional LSTM neural networks," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2015, pp. 1996–2000.

[13] W. Xu, J. Jang-Jaccard, T. Liu, F. Sabrina, and J. Kwak, "Improved bidirectional GAN-based approach for network intrusion detection using one-class classifier," *Computers*, vol. 11, no. 6, p. 85, 2022.

[14] R. Perdisci, G. Gu, and W. Lee, "Using an ensemble of one-class SVM classifiers to harden payload-based anomaly detection systems," in *Proc. 6th Int. Conf. Data Mining (ICDM)*, 2006, pp. 488–498.

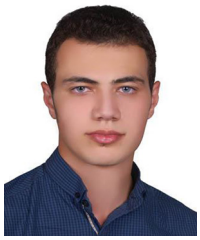
[15] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1757–1772, Jul. 2013.

[16] K. Huang, J. Yang, P. Hu, and H. Liu, "A novel framework for open-set authentication of Internet of Things using limited devices," *Sensors*, vol. 22, no. 7, p. 2662, 2022.

[17] J. Henrydoss, S. Cruz, E. M. Rudd, M. Gunther, and T. E. Boult, "Incremental open set intrusion recognition using extreme value machine," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, 2017, pp. 1089–1093.

[18] Z. Sawadogo, G. Mendy, J. M. Dembelle, and S. Ouya, "Android Malware classification: Updating features through incremental learning approach (UFILA)," in *Proc. 24th Int. Conf. Adv. Commun. Technol. (ICACT)*, 2022, pp. 544–550.
[19] D. Akgun, S. Hizal, and U. Cavusoglu, "A new DDoS attacks intrusion detection model based on deep learning for cybersecurity," *Comput. Secur.*, vol. 118, Jul. 2022, Art. no. 102748.
[20] D. Gumusbas and T. Yildirim, "AI for cybersecurity: ML-based techniques for intrusion detection systems," in *Advances In Machine Learning/Deep Learning-Based Technologies*. Cham, Switzerland: Springer, 2022, pp. 117–140.
[21] Z. Moti et al., "Generative adversarial network to detect unseen Internet of Things malware," *Ad Hoc Netw.*, vol. 122, Nov. 2021, Art. no. 102591.
[22] E. Nowroozi, Y. Mekdad, M. H. Berenjestanaki, M. Conti, and A. EL. Fergougui, "Demystifying the transferability of adversarial attacks in computer networks," *IEEE Trans. Netw. Service Manag.*, vol. 19, no. 3, pp. 3387–3400, Sep. 2022.
[23] M. Barni, E. Nowroozi, B. Tondi, and B. Zhang, "Effectiveness of random deep feature selection for securing image manipulation detectors against adversarial examples," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, 2020, pp. 2977–2981.
[24] D. Dasgupta, Z. Akhtar, and S. Sen, "Machine learning in cybersecurity: A comprehensive survey," *J. Defense Model. Simulat.*, vol. 19, no. 1, pp. 57–106, 2022.
[25] M. Barni, A. Costanzo, E. Nowroozi, and B. Tondi, "CNN-based detection of generic contrast adjustment with JPEG post-processing," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, 2018, pp. 3803–3807.
[26] Y. Meidan et al., "N-BaIoT—Network-based detection of IoT botnet attacks using deep autoencoders," *IEEE Pervasive Comput.*, vol. 17, no. 3, pp. 12–22, Jul.-Sep. 2018.
[27] F. Abbasi, M. Naderan, and S. E. Alavi, "Anomaly detection in Internet of Things using feature selection and classification based on logistic regression and artificial neural network on N-BaIoT dataset," in *Proc. 5th Int. Conf. Internet Things Appl. (IoT)*, 2021, pp. 1–7.
[28] V. Rey, P. M. Sánchez Sánchez, A. Huertas Celdrán, and G. Bovet, "Federated learning for malware detection in IoT devices," *Comput. Netw.*, vol. 204, Feb. 2022, Art. no. 108693.
[29] K. Angrishi. "Turning Internet of Things (IoT) into Internet of Vulnerabilities (IoV): IoT botnets." 2017. [Online]. Available: http://arxiv.org/abs/1702.03681
[30] H. Tahaei, F. Afifi, A. Asemi, F. Zaki, and N. B. Anuar, "The rise of traffic classification in IoT networks: A survey," *J. Netw. Comput. Appl.*, vol. 154, Mar. 2020, Art. no. 102538.
[31] E. Nowroozi. "SPRITZ-1.5C." 2022. [Online]. Available: https://github.com/ehsannowroozi/SPRITZ-1.5C/blob/main/README.md
[32] C. J. Hernández-Castro, Z. Liu, A. Serban, I. Tsingenopoulos, and W. Joosen, "Adversarial machine learning," in *Security and Artificial Intelligence*. Cham, Switzerland: Springer, 2022, pp. 287–312.

Ehsan Nowroozi (Senior Member, IEEE) received the Ph.D. degree from the University of Siena, Italy, in 2020, following his Ph.D., he was a Postdoctoral Fellow with Siena and Padua Universities, Italy, and Sabanci University, Turkey. He is an Assistant Professor with the Faculty of Engineering and Natural Sciences, Department of Computer Engineering, Istanbul's Bahcesehir University's. His primary research interests are in cybersecurity in a particular reference to adversarial machine learning and adversarial multimedia forensics. He is also a Reviewer for several journals, including IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.



Mohammadreza Mohammadi (Member, IEEE) received the bachelor's degree in computer engineering (software-network) from Bu-Ali Sina University, Hamedan, Iran, in 2019. He is currently pursuing the master's degree in ICT with the University of Padua. His main research interest is in the area of machine learning, cybersecurity, IoT, and computer vision. In addition, he works on research contexts which are combination of industrial IoT security and artificial intelligence, and Intrusion detection systems and healthcare systems.

Erkay Savaş (Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from the Electronics and Communications Engineering Department, Istanbul Technical University in 1990 and 1994, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, Oregon State University in June 2000. He has been a Faculty Member with Sabanci University since 2002. His research interests include applied cryptography, data and communication security, security and privacy in data mining applications, embedded systems security, and distributed systems. He is a member of ACM, the IEEE Computer Society, and the International Association of Cryptologic Research K. Derya et al.

Yassine Mekdad (Member, IEEE) received the M.Sc. degree in cryptography and information security from the Mohammed V University of Rabat, Morocco. He was Guest Researcher with the SPRITZ Research Group, University of Padua, Italy. He is currently working as a Research Scholar with the Cyber-Physical Systems Security Lab, Florida International University, Miami, FL, USA. His research interest principally cover security and privacy problems in the Internet of Things, Industrial Internet-of-Things, and cyber-physical systems. Furthermore, he works on research problems at the intersection of the cybersecurity and networking fields with an emphasis on their practical and applied aspects.

Mauro Conti (Fellow, IEEE) received the Ph.D. degree from the Sapienza University of Rome, Italy, in 2009. He is Full Professor with the University of Padua, Italy. He is also affiliated with TU Delft and the University of Washington, Seattle. His research in the area of Security and Privacy is also funded by companies, including Cisco, Intel, and Huawei. He published more than 450 papers in topmost international peer-reviewed journals and conferences. He is the Editor-in-Chief for IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY and has been an Associate Editor for several journals, including IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT. He is a Fellow of YAE and a Senior Member of ACM.