The Role of Machine Learning in Cybersecurity

GIOVANNI APRUZZESE* and PAVEL LASKOV*, University of Liechtenstein, Liechtenstein
EDGARDO MONTES DE OCA and WISSAM MALLOULI, Montimage, France
LUIS BÚRDALO RAPA, S2 Grupo, Spain
ATHANASIOS VASILEIOS GRAMMATOPOULOS and FABIO DI FRANCO, ENISA, Greece

Machine Learning (ML) represents a pivotal technology for current and future information systems, and many domains already leverage the capabilities of ML. However, deployment of ML in cybersecurity is still at an early stage, revealing a significant discrepancy between research and practice. Such a discrepancy has its root cause in the current state of the art, which does not allow us to identify the role of ML in cybersecurity. The full potential of ML will never be unleashed unless its pros and cons are understood by a broad audience.

This article is the first attempt to provide a holistic understanding of the role of ML in the entire cybersecurity domain—to any potential reader with an interest in this topic. We highlight the advantages of ML with respect to human-driven detection methods, as well as the additional tasks that can be addressed by ML in cybersecurity. Moreover, we elucidate various intrinsic problems affecting real ML deployments in cybersecurity. Finally, we present how various stakeholders can contribute to future developments of ML in cybersecurity, which is essential for further progress in this field. Our contributions are complemented with two *real* case studies describing industrial applications of ML as defense against cyber-threats.

CCS Concepts: • **Security and privacy**; • **Computing methodologies** → **Machine learning**;

Additional Key Words and Phrases: Cybersecurity, incident detection, machine learning, artificial intelligence

ACM Reference format:
Giovanni Apruzzese, Pavel Laskov, Edgardo Montes de Oca, Wissam Mallouli, Luis Búrdalo Rapa, Athanasios Vasileios Grammatopoulos, and Fabio Di Franco. 2023. The Role of Machine Learning in Cybersecurity. *Digit. Threat.: Res. Pract.* 4, 1, Article 8 (March 2023), 38 pages.
https://doi.org/10.1145/3545574

1 INTRODUCTION

With the rising complexity of modern information systems and the resulting ever increasing flow of big data, the benefits of **Artificial Intelligence (AI)** are now widely recognized. Specifically, **Machine Learning (ML)** methods [85] are already deployed to solve diverse real world tasks—especially with the advent of deep learning [98]. Fascinating examples of practical achievements of ML are machine translation [168], travel and vacation

*G. Apruzzese and P. Laskov contributed the most to the paper.

Authors' addresses: G. Apruzzese and P. Laskov, University of Liechtenstein, Vaduz, Liechtenstein; emails: {giovanni.apruzzese, pavel.laskov}@uni.li; E. M. de Oca and W. Mallouli, Montimage, France; emails: {edgardo.montesdeoca, wissam.mallouli}@montimage.com; L. B. Rapa, S2 Grupo, Spain; email: luis.burdalo@s2grupo.es; A. V. Grammatopoulos and F. Di Franco, ENISA, Greece; emails: gramthanos@gmail.com, fabio.difranco@enisa.europa.eu.

2576-5337/2023/03-ART8 $15.00
https://doi.org/10.1145/3545574

recommendations [77], object detection and tracking [139], and even various applications in healthcare [57]. Furthermore, ML is rightly considered to be a technology enabler, as it has shown great potential in the context of telecommunication systems [114] or autonomous driving [8].

Nevertheless, modern society is increasingly relying on **Information Technology (IT)** systems—including autonomous ones—which are also actively leveraged by malicious entities. Digital threats are, in fact, continuously evolving [90], and according to Gartner attackers will have sufficient capabilities to *harm or kill* humans by 2025 [3]. To prevent such incidents and mitigate the plethora of risks that can target current and future IT systems, defensive mechanisms require the capability to quickly adapt to the (i) mutating environments and (ii) dynamic threat landscape.

Coping with such a twofold requirement via static and human-defined methods is clearly unfeasible, and deployment of ML in cybersecurity is inescapable. Not surprisingly, abundant work addressed integration of ML in cybersecurity, as evidenced by recent survey papers (e.g., References [23, 36, 71]) and technical reports (e.g., References [35, 106]). Despite impressive results in research settings, however, the development and integration of ML in production environments is progressing at a slow pace. A recent survey [93] shows that although over 90% of companies already use some AI/ML in their defensive tools, we observe that most of these solutions still leverage "unsupervised" methods (e.g., References [2, 97]) and mostly for "anomaly detection." Such observation demonstrates a drastic discrepancy between research and practice, especially in comparison with other domains where ML has already become an indispensable asset.

The peculiarity of the security domain is that all operational decisions—made by the top management—are about the tradeoff between losses and losses [83]. In simple terms, the rationale is "paying x to avoid paying $y \gg x$." Investment in security should be justified by the prevention of substantially higher *but ultimately unpredictable* losses from security incidents. Hence, *decision makers* must have a clear understanding of the (i) benefits, (ii) problems, and (iii) challenges of a cybersecurity solution before endorsing their adoption in practice. However, the current state of the art of ML for cybersecurity fails to deliver such understanding. Taken individually, research papers—commonly claiming to outperform previous work—often lead to contradictory results. For instance, Reference [166] shows that deep learning methods outperform "traditional" ML methods, but the opposite is claimed in Reference [134] in the exact same setting. Furthermore, existing literature surveys related to ML in cybersecurity do not provide a holistic coverage suitable for operational decisions. Some of them are too technical and hence tailored for ML experts (e.g., Reference [180]), others focus only on research efforts neglecting real-world implications (e.g., Reference [23]) or have a limited scope (e.g., only deep learning [36]). As a result, the role of ML in cybersecurity is portrayed in a highly fragmented way, thus hindering deployment of ML in practice—despite its great potential for cybersecurity.

We attempt to rectify this problem. Specifically, this article is the first effort to provide a comprehensive analysis of the role of ML in cybersecurity. We distill scientific knowledge and industrial experience related to deployment of ML *within the entire domain of cybersecurity*. One of our goals is to make the current state of the art *understandable to any reader*, irrespective of their prior expertise in cybersecurity or ML. We also take this opportunity to clarify many *misconceptions* related to ML in the context of cybersecurity. We highlight the benefits of using ML in cybersecurity by listing all the tasks where it outperforms or provides novel capabilities with respect to traditional security mechanisms. We also elucidate the intrinsic problems of ML in the cybersecurity context. Such an analysis reveals the challenges that require the joint contribution of all relevant stakeholders to improve the quality of ML-driven security mechanisms.

Let us explain how we achieve our objective and outline the structure of our article, which comprises several self-contained sections. We begin (Section 2) by introducing the key concepts of the ML paradigm in a *notation-free* form. We also define the intended audience of this article and outline the differences of our work from previous literature surveys and reports.

Then, in Section 3, we present the most emblematic application of ML in security: cyberthreat detection. We distinguish between three broad areas: network intrusion detection, malware detection, and phishing detection,

which is common in related literature [23, 163]. The goal of this section is to highlight the *added value* of ML with respect to traditional detection mechanisms.

Next, in Section 4, we elucidate the cybersecurity tasks *orthogonal to threat detection* that can exploit the capabilities of ML to analyze unstructured data. In contrast to detection problems that require (costly) labels, raw data are abundant in cybersecurity and can also be exploited via ML. For instance, alerts can be filtered to remove annoying false alarms or compressed into more manageable reports. Furthermore, information from diverse sources can be cross-correlated to anticipate novel attacks or to identify the weak-spots of a given organization. The goal of this section is to illustrate that there exist many (and vastly unexplored) additional areas in which ML can be deployed to enhance the security of modern systems.

We continue (Section 5) by emphasizing the intrinsic problems of cybersecurity applications of ML. Some of these problems (e.g., concept drift, adversarial examples, confidentiality) are fundamental and arise from the contrasting assumptions of cybersecurity and ML. Further problems are specific to either in-house development (e.g., hidden maintenance costs) or commercial products (e.g., limited scope and transparency). The goal of this section is highlighting that ML is not perfect and *real* deployments involve many tradeoffs, which must be known (to decision makers), mitigated (by ML engineers), and addressed (in future work).

As our main constructive contribution, we outline the impending challenges of ML in cybersecurity in Section 6. Solving these challenges will strongly facilitate the operational deployment of ML in cybersecurity. However, it requires the joint effort of (i) regulatory bodies, (ii) corporate executives, (iii) ML engineers and practitioners, and (iv) the scientific community. Our takeaway is that rectifying the current immaturity of ML in cybersecurity requires *a radical re-thinking* of future technological developments. For instance, research efforts should focus on more pragmatic results instead of merely "outperforming the state of the art." However, such efforts necessitate an increased availability of real data whose disclosure requires authorization by senior management, as well as potentially new regulations that enable public release of such data.

To establish a connection between research and practice, we discuss two *real* industrial applications of ML in cybersecurity in Section 7. We note that commercial security products are typically provided as "black boxes" with little technical details about the actual implementation of ML. This section sheds light into the operational tradeoffs and "tricks of the trade" needed to meet the practical needs of the customers. These case studies are provided with the contribution of Montimage and S2Grupo.[1]

This article is a result of collaboration among researchers, industry practitioners, and policy-makers. Our findings reflect the insights from both recent technical reports and scientific literature. To the best of our knowledge, no previous work combines such a broad scope with our heterogeneous intended audience.

Contribution. Our main goal is to foster the deployment of ML in cybersecurity by bridging the gap between research and practice. Specifically, our article makes the following contributions:

- it provides an overview of the *benefits* and *problems* of ML in the *entire* cybersecurity domain;
- it considers the twofold perspective of the *research* and *industrial* community;
- it identifies many *misconceptions* that are becoming common in this field;
- it highlights how (i) regulatory bodies, (ii) corporate executives, (iii) engineers, and (iv) the research community can contribute to *future developments* of ML in cybersecurity.
- it elucidates two *real deployments* of ML products.

Furthermore, this article is meant to be understandable by *any* reader, irrespective of their technical expertise.

2 BACKGROUND AND MOTIVATION

To set up the stage for our article, we first introduce the main concepts of Machine Learning in a simplified way, accessible to any reader (Section 2.1). Our goal is to present the established terminology as well as the common

[1]The names of all authors, companies and vendors were anonymised during the reviewing process.

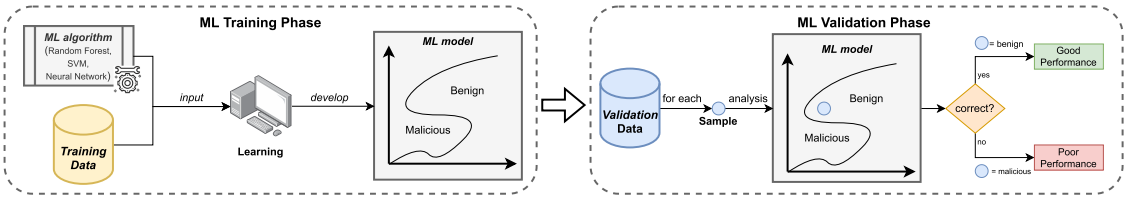

Fig. 1. Machine Learning development. After collecting some training data and analyzing such data via an ML algorithm, an ML model is obtained. Such an ML model must be tested via some validation data. If the performance of such an assessment is appreciable, then the ML model can be deployed in production.

classes of existing ML methods. We then define the scope and target audience of this article (Section 2.2) and highlight the differences of our effort with respect to previous work (Section 2.3).

2.1 Soft Introduction to Machine Learning

The goal of ML is to develop machines[2] that automatically learn to make decisions. The learning is done through a *training* phase: By instructing a computing device to analyze some "existing" (training) data via a given ML *algorithm*, a ML *model* is developed. Such a model incorporates all the knowledge learned during the training phase and implements a function to make decisions on "future" data. Before a ML model can be deployed in an operational environment, its performance must be assessed. To this end, some "validation" data are processed by the ML model and its predictions are either analyzed by humans or compared with some known ground truth. We can hence define a ML *method* as "the process for developing a ML model by using ML algorithms on some training data." An exemplary workflow of the training and validation phases is schematically depicted in Figure 1.

A crucial factor in the development of ML models is the notion of *labels*, which represent the target value for a prediction function on a given sample (e.g., benign or malicious). Depending on the availability of labels, ML methods can be classified into *supervised* and *unsupervised*. Supervised methods explicitly require labelled training data. In some cases, such labels occur naturally[3]; otherwise, acquiring labels involves dedicated manual verification. However, unsupervised methods either do not require labels or involve limited supervision. For instance, in *reinforcement learning* the ML model is built through a feedback mechanism[4] that is completely automated.

An orthogonal classification of ML methods is between *shallow* and *deep* learning. Deep learning refers to ML methods based on *neural networks*, which typically require more computational power and larger training datasets compared to shallow ML methods—requirements that could only be met in the recent years [98]. Let us point out the first *misconception*: Deep learning is not not necessarily better than shallow ML. Indeed, when the data to analyze have a small number of features, shallow ML can attain similar performance as deep learning [23], but the latter still requires more resources and the results are more difficult to interpret (e.g., Reference [14]). In contrast, the advantages of deep learning lie in its ability to deal with data with high complexity, such as images, unstructured text, or when temporal dependencies must be taken into account. In all such cases, shallow ML simply cannot be used. Deep learning methods can be supervised or unsupervised and can also leverage reinforcement learning—e.g., the popular generative adversarial networks [17].

[2]The notion of a "machine" refers to a software component that can be deployed on any computing device, even in the cloud.
[3]In time-series forecasting [44], the learning is done by analyzing the past history of a given phenomenon, which is used to make the future predictions. Such history can be seen as the training data, where each element is associated to its timestamp and its known value (i.e., the label).
[4]Such a mechanism only requires defining the "actions" that can be taken by the ML model and the "reward" that should be provided to the ML model depending on the effects of its actions on the "environment."

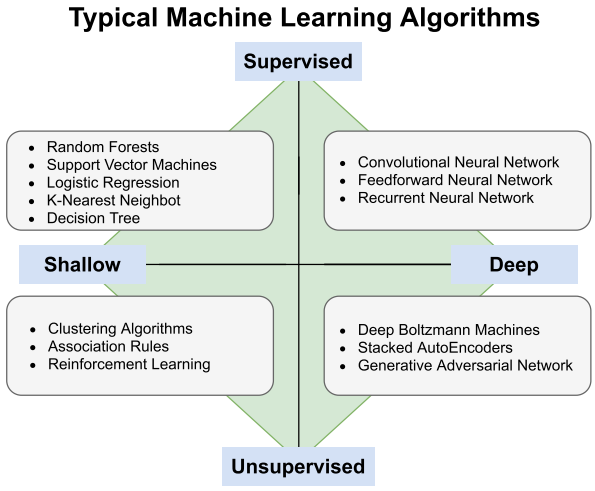

Fig. 2. Typical machine learning algorithms. An algorithm can be "deep" if it relies on *neural networks*; otherwise, it is "shallow." Algorithms requiring *labelled* data are used for "supervised" tasks; otherwise, they can be used also in "unsupervised" tasks.

Table 1. Typical ML Performance Metrics

Metric	Description	Formula
Accuracy (Acc)	It denotes the percentage of correct predictions among all predictions. It is misleading in the presence of imbalanced distributions (common in cybersecurity).	$Acc = \frac{TP+TN}{TP+TN+FP+FN}$
Detection Rate (DR)	It measures the capacity of identifying attacks, but it does not consider false alarms. It is also known as *Recall*, or *True Positive Rate*.	$DR = \frac{TP}{TP+FN}$
FP Rate (FPR)	It represents the percentage of incorrect "positive" predictions. Useful for measuring the amount of false alarms.	$FPR = \frac{FP}{FP+TN}$
Precision ($Prec$)	It denotes the percentage of correct predictions among all "positive" samples. High values implies low false positives, but nothing can be said about false negatives.	$Prec = \frac{TP}{TP+FP}$
F1-score ($F1$)	It combines *Precision* and *Detection Rate* into a single metric. Useful for "overviews," but it is difficult to interpret.	$F1 = \frac{TP}{TP+0.5(FN+FP)}$

Cyber-threat detection represents a binary classification problem: Samples are positive (malicious) or negative (benign). Accuracy is useful for cybersecurity tasks where such a distinction is not possible. Acronyms: True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN).

We provide an overview of some of the most popular ML algorithms in the above-mentioned categories in Figure 2. For a more comprehensive description, we refer the reader to Reference [23].

Finally, let us briefly address the performance assessment of ML models. The most common quality measure is the *accuracy* metric, which represents the percentage of correct predictions made by the ML model. However, *accuracy can be misleading* in the presence of imbalanced data distributions, which is typical in cyber-threat detection, because malicious activities tend to be rare events and are (hopefully) overshadowed by benign samples. In such a context, it is common to differentiate between "positives" (i.e., malicious activities) and "negatives" (i.e., benign activities). The performance can then be measured by taking into account the correct (i.e., True Positives and True Negatives) and incorrect (i.e., False Positives and False Negatives) predictions generated by a given ML model. A complete list of the most common performance metrics is in Table 1. Note that performance assessment pertains to ML *models* and not methods. Depending on the specific setting—e.g., the training data, the ML algorithm, its parameters—a ML method may yield many ML models, each having a different performance.

2.2 Scope and Target Audience

The scope of our article is bridging the gap between scientific research and operational practice of ML in cybersecurity. We do so by unifying in a single document the benefits, problems, and future challenges of ML in the entire cybersecurity domain. Our article is meant to be understandable by any reader that is interested in ML and its relationship with cybersecurity.

Specifically, we address the following three classes of target readers:

- *Decision makers (e.g., Corporate Executives, Chief Information Security Officers)* who need to understand the state of the art. This article should allow more sound decisions on the adoption of ML and its integration into existing systems to enhance the productivity of security operation centers.
- *Security professionals (e.g., security consultants and administrators, digital forensics experts)* who should understand the operational issues affecting ML applications in cybersecurity. Such understanding is crucial for a reliable operation of such instruments in practice, as well as for transparent marketing and assessment of commercial ML solutions.
- *Researchers and Engineers* who are interested in devising novel ML solutions for cybersecurity, improving existing ML systems, or mitigating some of their limitations. The open issues and challenges presented in this article should guide future developments of ML for cybersecurity.

The takeaways of this article leverage the contribution and take into account the standpoints of *all* the above-mentioned classes of readers. For example, experienced engineers may be aware of the shortcomings of ML, but they may not know how such issues are received by decision makers. At the same time, security professionals may know how ML is used, but they can benefit from understanding the most significant future developments in this field.

2.3 Related Work

With the advent and increasing popularity of ML, abundant works proposed ML solutions for diverse cybersecurity tasks, resulting in hundreds of research papers. Such abundance inspired many literature surveys that aggregate or summarize the state of the art. However, most of such studies may provide a detailed analysis but on a single application, such as cyber risk assessment [137] or IoT security [45]. Others may focus on a specific cyber detection problem, e.g., malware [13, 71, 160], spam [70, 89], or intrusion detection [46, 102]. Some papers do not explicitly focus on ML (e.g., Reference [82]), whereas others do not focus on cybersecurity (e.g., References [81, 127]). Finally, many works only consider specific ML paradigms, such as generative adversarial networks (e.g., Reference [180]), adversarial ML (e.g., References [24, 108]), reinforcement learning [121], or deep learning [36]—the latter of which is not necessarily the best "universal" ML solution for cybersecurity. Such a finding was shown in the well-known work by Apruzzese et al. [23], which has a more limited scope than our article, because they (i) focus on the separation between shallow and deep learning, (ii) do not delve into cybersecurity tasks beyond threat detection, and (iii) only consider scientific works. Indeed, ML has undergone significant advances in cybersecurity since the publication of Reference [23]—as we will show in our study.

All these papers, while being useful for interested and experienced researchers, cannot be appreciated simultaneously by security specialists, executives, and stakeholders—which are included in our target audience. Indeed, the excessive depth or limited scope of prior work does not allow us to grasp the true role (current and future) played by ML in the entire cybersecurity domain. At the same time, technical reports (e.g., References [35, 106]) may be easier to understand by security personnel but are not useful for researchers due to lack of comprehensive guidelines and do not provide much insight on *real* ML deployments.

We aim to close the gap between research and practice of ML in cybersecurity with a single document. To this end, we shape this article so that it is understandable—and usable—by any reader, regardless of their technical

Fig. 3. Pros and cons of supervised and unsupervised ML for cyber threat detection.

competence in ML. With respect to past works, this article represents a "meta-review" of the state of the art[5] that provides a (i) *comprehensive* overview and (ii) *practical* recommendations and research directions (iii) within the *entire* cybersecurity sphere. Moreover, we (iv) clear many misconceptions that are becoming prevalent in this domain. Finally, we (v) address *all* potential stakeholders—which include but are not limited to researchers. To the best of our knowledge, no existing paper unifies all of the above in a single contribution.

3 MACHINE LEARNING FOR THREAT DETECTION

The security lifecycle spans over three processes: prevention, detection, and reaction [171]. The complete prevention of any cyber threat is recognized as an impossible task, whereas the reaction phase assumes that the damage has already taken place. Hence, most security mechanisms (including ML-based ones) focus on threat *detection*. For instance, it is not possible to prevent the creation of a phishing webpage; however, such a threat can be defused by detecting that a given webpage is compromised and alerting the users before they fall victim to a phishing "hook."

The detection of cyber threats can leverage two distinct approaches: *misuse* based and *anomaly* based. The former, also referred to as *signature* or *rule* based, require defining specific "patterns" that correspond to a given threat—under the assumption that future threats will exhibit the same patterns. The latter require defining a notion of "normality" and aim to detect events deviating from such normality—under the assumption that such deviations correspond to security incidents. These two detection approaches are *complementary*: Misuse-based approaches are very precise but can only detect known threats; anomaly-based approaches tend to generate more false alarms but have a better chance against novel attacks.

Before the advent of ML, detection mechanisms required *manual* definition of all the necessary elements for a given approach (either misuse and anomaly based). Aside from being a time consuming and error prone task, such efforts could not cope with the increasing growth of modern environments. Hence, with the progress of data analytics techniques, detection systems began to leverage *data-driven* solutions, such as ML. These solutions not only required less manual effort but, in some cases, even outperformed traditional handwritten detection schemes [46]. In the context of ML, such increased performance is due to the intrinsic ability of ML to learn "weak" signals—unnoticed by human operators—in the analyzed data and use such signals to enhance their detection.

The distinguishing characteristic of ML applications for cyber threat detection (schematically depicted in Figure 3) is whether *supervised* or *unsupervised* ML methods can be deployed. The former can be used as complete detection systems but require labelled data created via some human supervision. The latter do not have a human in the loop but can only perform ancillary tasks.[6] Depending on the data type to analyze, labels may be easier to acquire: For instance, any layman can distinguish a legitimate webpage from a phishing one [99], but distinguishing benign from malicious network traffic is harder [63].

[5]We observe that our article includes almost 200 referenced works. However, most of such works are cited only once, i.e., in the section devoted to the specific problem addressed by the referenced article.
[6]For instance, anomaly detection can be done in an unsupervised fashion, but not all anomalies correspond to security incidents.

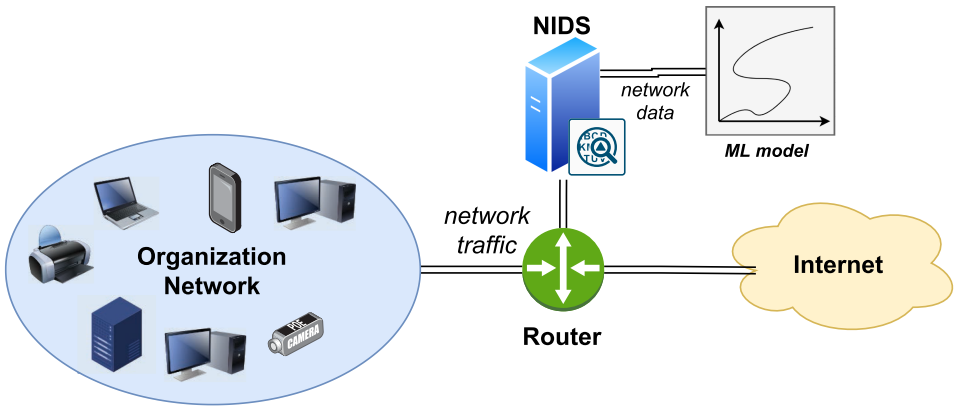

Fig. 4. Typical deployment of a ML-NIDS. The border router forwards all the outgoing/incoming network traffic to a NIDS, which further analyzes such data via a ML model.

It is common to associate ML methods with anomaly detection (even recent papers suffer from such confusion, e.g., Reference [34]). This is a *misconception*, because ML can be used also for misuse-based approaches [91]. Specifically, by analyzing large amounts of data, ML methods can learn the patterns differentiating benign events from malicious ones so as to automatically define the "signatures" for misuse-based approaches. At the same time, ML can be used for anomaly detection by automatically identifying the "normal" activities that correspond to regular behaviors within a given environment.

Let us elucidate some successful applications of ML aimed at the detection of illicit activities that may occur in a modern enterprise. Without loss of generality, we organize this section by distinguishing three broad cyber detection areas: *network intrusion detection* (Section 3.1), *malware detection* (Section 3.2), and *phishing detection* (Section 3.3). There are hundreds of works proposing ML for these tasks, and analyzing all such proposals is outside our scope. Hence, we focus on some interesting and recent applications of ML, emphasizing their *practical* results. Our case studies in Section 7 will consider two exemplary applications of ML for cyberthreat detection.

3.1 Machine Learning in Network Intrusion Detection

One of the cybersecurity areas of main interest to modern enterprises is that of Intrusion Detection, which is accomplished by means of **Intrusion Detection Systems (IDS)**. An IDS can belong to either of two categories: a ***Network* Intrusion Detection System (NIDS)** analyzes activities at the network level, whereas a ***Host* Intrusion Detection System (HIDS)** analyzes activities at the individual host level. In this section, we consider NIDS, because HIDS mostly focus on detecting (local) malware that we discuss in Section 3.2.

Since the early 2010s, many ML solutions have been proposed to improve the effectiveness of NIDS, both in scientific literature [11, 23, 36, 46], and in patents (e.g., References [131, 140]). A NIDS can be deployed anywhere in a network environment and can exploit ML to detect threats against diverse targets, such as cloud, IoT, endpoint devices [93], and even automotive controllers [104]. We report in Figure 4 the typical deployment of a NIDS that leverages the support of ML, which can analyze data of different types, e.g., full **packet-captures (PCAP)**, network flows[7] (NetFlows), **Simple Network Management Protocol (SMNP)**, or even **Domain Name System (DNS)** records. Specifically, with the increasing growth of modern networks, NetFlow analyses are preferred due to many advantages over traditional PCAP, such as reduced privacy concerns, less space required for storage, and faster processing times [176].

[7]Netflow: https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-netflow/.

ML methods based on *unsupervised learning* are particularly appreciated, because acquiring labelled data *for an entire network* is difficult [63]. Among these approaches, we highlight the results obtained by *clustering* methods. For example, in Reference [26] the authors aim to detect attacks by clustering NetFlows with similar temporal behavior and, subsequently, finding the clusters containing hosts that raised alarms from a commercial NIDS based on manual signatures. The results showed a remarkable increase in detection performance[8] with respect to the commercial signature-based NIDS, which only detected three malicious hosts, whereas the integration of ML allowed us to detect 12.

Unsupervised methods can also be used to support the (manual) generation of rules for misuse-based NIDS. In CyberProbe [117], the authors cluster honeypot traffic and create specific rules for each cluster: Such rules allowed us to detect over 75% attacks that were not included in any security feed. Some papers also exploit unsupervised approaches to counter *lateral movement*[9]: The approach in Reference [43] can successfully detect such instances (over 90% recall) with low FP (10%). Finally, NIDS can also benefit from *deep unsupervised algorithms*. As an example, in Kitsune [113] the authors use deep learning to analyze PCAP data and improve the detection rate from below 1% to over 95% while maintaining a low FP rate (below 0.1%). The advantages of unsupervised ML methods make them suitable for commercial products: As an example, the method in Reference [105] is used by Aizoon[10] to support botnet detection via DNS analyses, achieving less than 0.1% FP rate. Our detailed case study in Section 7.1 presents the deployment of unsupervised ML used by Montimage to detect anomalous activities in a modern network.

However, approaches based on *supervised learning*, due to their reliance on good quality labels, are more expensive to deploy but can also provide excellent results. For instance, Exposure [40] leverages labelled DNS records to detect domains involved in malicious activities and achieves less than 10% false alarm rate. A notable effort against botnets is Reference [155], where the authors collect and label some NetFlows, and then use such labelled data to develop a ML botnet detector achieving over 95% precision. Moreover, the work in Reference [19] proposes the usage of probability labels (instead of binary labels) to detect botnet NetFlows that may evade traditional ML-NIDS and reach over 97% precision. Remarkable successes also include deep learning methods, such as the approach in Reference [84], which achieves almost 95% detection rate. In particular, we highlight those solutions that combine deep learning with temporal analyses: A twofold perspective allows us to detect additional malicious patterns that can improve detection performance. For instance, in Reference [56] the F1-score improves from 0.90 to 0.95 when also temporal dependencies are considered. We will present a real deployment of a similar solution in Section 7.2, describing how S2Grupo protects **Industrial Control Systems (ICS)**, showcasing the pros (and cons) of ML with respect to older techniques based on heuristics.

Let us conclude with a remark: *The superiority of deep learning for NIDS is not yet proven*. For instance, the authors of References [134] and [166] both evaluate shallow and deep ML methods on the same dataset (the CICIDS17 [147]): While Reference [166] claims that deep learning outperforms traditional ML, the authors of Reference [134] achieve the opposite result. Specifically, Reference [166] shows a "deep" neural network achieving an F1-score of 0.96 and a "shallow" decision tree achieving an F1-score of 0.95, whereas Reference [134] shows a "deep" neural network also achieving an F1-score of 0.96, but their "shallow" decision tree reaches an F1-score of 0.99. Our stance on this subject is that, under the assumption that deep learning is superior, the marginal improvement does not justify its adoption due to its additional complexity and computational requirements.

3.2 Machine Learning in Malware Detection

The fight against malware is one of the most emblematic challenges of cybersecurity. Because malware affects a specific device, its detection is performed by analyzing data at the host level, i.e., through HIDS. Indeed,

[8] A similar approach has been successfully integrated even in a commercial product, which we cannot name due to NDA.
[9] Lateral Movement: https://www.lastline.com/blog/lateral-movement-what-it-is-and-how-to-block-it/.
[10] https://www.aizoongroup.com/.

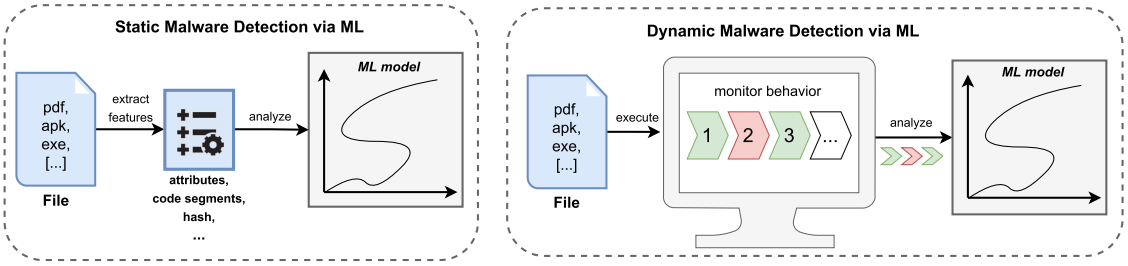

Fig. 5. Malware detection via ML. In static analyses, the properties of a given file are extracted and analyzed by a ML model. In dynamic analyses, the file is executed and the entire behavior is monitored and then analyzed by a ML model.

antiviruses can be considered as a subset of HIDS [94]. A given malware variant is tailored for a given **operating system (OS)**. The popularity of Windows OS made it the most common malware target for more than two decades. However, attackers are now turning their attention to mobile devices running, e.g., Android OS.[11]

Malware detection can use two types of analyses: *static* or *dynamic*. The former aim to detect malware without running any code by simply analyzing a given file. The latter focus on analyzing the behavior of a piece of software during its execution, usually by deploying it in a controlled environment and monitoring its activities. Both static and dynamic analyses, schematically depicted in Figure 5, can benefit from ML.

Static Analysis. These analyses are simple, particularly effective against known pieces of malware, and can be enhanced via ML in many ways. For instance, clustering is useful to identify properties of similar pieces of malware. A similar method is proposed in Reference [80], with the goal of finding a common treatment against all elements in each cluster, and reaches up to 90% precision. In contrast, the authors of Reference [100] leverage clustering to improve the detection of Android malware, and exceed 95% detection rate. Static analyses can be further improved when labelled data are available. An early example is the detection of malicious **Portable Document Format (PDF)** files in Reference [153]: Here, the authors use ML to analyze the structural properties of PDF files, extracting features that yield proficient detection results (over 99% detection rate with less than 0.001% FP rate). Recently, a different approach leverages deep learning to transform executables into images, which are then used to perform the detection: The authors of Reference [87] achieve over 99% accuracy in identifying Windows malware.

Despite these successes, all static malware detection approaches are prone to evasion. This can be easily achieved by modifying the malware executable, which can be implemented without changing its underlying malicious logic. To aggravate the problem, advanced malware variants (e.g., *polymorphic* or *metamorphic*) automatically modify their executables, defeating any static detection approach.

Dynamic Analysis. The combination of dynamic approaches with ML techniques yields effective countermeasures against polymorphic malware. Multiple ML solutions exploit clustering: grouping malware with similar behavior allows us to focus only on those clusters that have not been seen before. For example, Reference [141] proposes a dynamic approach combining clustering and anti-virus scanners to detect and sanitize entire groups of malware variants, achieving almost perfect accuracy against Windows malware. More recently, the work in Reference [15] focuses on Windows malware by leveraging a combination of graph and **Natural Language Processing (NLP)** techniques applied to dynamic API calls and achieves 99.99% accuracy. Some papers even propose deep learning, such as Reference [103], which uses deep neural networks to extract the most relevant dynamic features to classify Android malware, achieving nearly 80% accuracy. Moreover, the authors of Reference [7] apply deep learning to detect Windows *ransomware* and achieve 93% detection rate and 97% precision. An interesting work is HeNet [52], which leverages ML for dynamic malware detection by analyzing *hardware*-specific

[11] https://www.gdatasoftware.com/news/2019/07/35228-mobile-malware-report-no-let-up-with-android-malware.

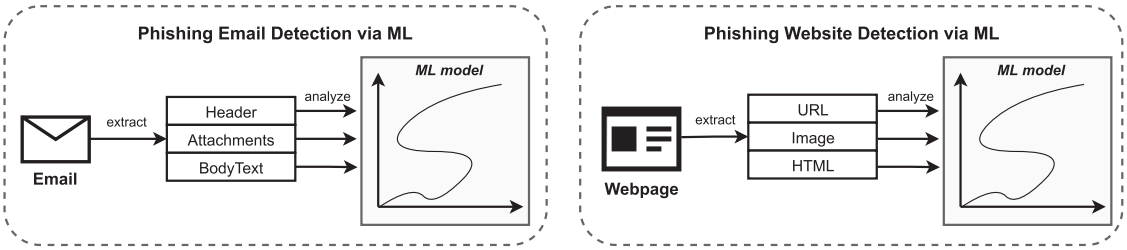

Fig. 6. Phishing detection via ML. For websites, the ML model can analyze the URL, the HTML, or the visual representation of a webpage. For emails, the ML model can analyze the body text, the headers, or the attachment of the email.

(i.e., Intel CPU) data streams, achieving perfect accuracy on real benchmarks. Finally, it is possible to *combine* static with dynamic analyses via ML: This is done in EC2 [49], which combines unsupervised with supervised ML to detect novel android malware, achieving over 90% detection rate.

3.3 Machine Learning in Phishing Detection

Phishing represents one of the most common vectors for penetrating a target network and is still a rampant threat in the cybersecurity landscape [90]. Early detection of phishing attempts is of paramount importance to modern organizations and can greatly benefit from ML. Specifically, we distinguish two different applications of ML to counter phishing attempts: detection of phishing *websites*, where the goal is identifying webpages that are camouflaged to resemble a legitimate website, and detection of phishing *emails*, which either point to a compromised website or induce a response that includes sensitive information. The main difference between these two approaches is the type of analyzed data: For websites, it is common to use the **Universal Resource Locator (URL)** of the webpage, its **Hyper Text Media Language (HTML)** code, or even its visual representation [159]; for emails, it is typical to analyze the text, the header, or the attachments of an email [9]. A schematic representation of such applications is shown in Figure 6, which we now describe in more detail.

Phishing Webpage Detection. Phishing websites are mostly dealt with via *blacklists*. However, such lists quickly become unreliable, because expert adversaries frequently move their phishing hooks from site to site: As shown in Reference [159], over 90% of "squatting" phishing websites are not detected by popular blacklists. ML represents a viable alternative to manual and static blacklisting, and modern web-browsers already leverage its potential [101].

Compared to malware or network intrusion detection, works proposing *unsupervised* ML against phishing websites are less prevalent. An example is Reference [185], exploiting clustering to support the detection of phishing websites and achieving over 95% accuracy. In contrast, *supervised* ML is abundant, because verifying the legitimacy of a webpage is relatively simple, which facilitates labelling procedures and allows us to develop complete ML detectors [55, 99, 159]. Some works use ML to analyse features extracted from a given URL. It is interesting to note that while the authors of Reference [33] use up to 130 features to achieve 99% detection rate, other works (e.g., Reference [144]) use fewer than 30 features and achieve similar results. Other proposals leverage third-party information provided by reputable sources (e.g., DNS records), which can be derived from the URL: An example is PhishMon [122], which achieves nearly 96% accuracy while maintaining a low 1% rate of false positives. Some papers consider the twofold perspective provided by the analysis of both URL- and HTML-based features, which is advantageous when the single URL is not enough to identify a webpage as phishing or not. For example, the work in Reference [32] achieves 95% detection rate by combining these two data types. A significant work is Reference [55], which combines the inspection of the underlying HTML code of the webpage with image processing techniques (based on deep learning) to identify compromised websites: The results show over 95% detection rate at the cost of 1% false-positive rate. Finally, Reference [159] uses all of the above (images,

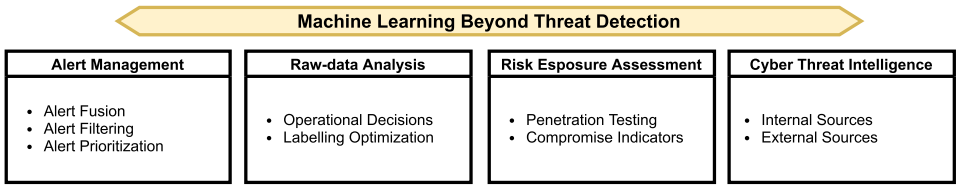

Fig. 7. Additional tasks that can be addressed via ML in cybersecurity. All such tasks mostly involve dealing with raw and unstructured data from heterogeneous sources and provide fertile ground for ML.

HTML and URL): For nearly 1,000 squatting phishing websites, manual blacklisting only detected 9%, whereas ML detected 70% of such phishing attempts.

Phishing Email Detection. One of the earliest applications of ML for cybersecurity involves the detection of unsolicited emails (also often referred to as "spam"). Recent advances in NLP can be leveraged by ML to analyze the body of an email and identify malicious intent [39].

Only few proposals leverage *unsupervised* ML, such as Reference [61], which achieves over 95% detection rate. However, as it is the case for phishing website detection, acquiring ground-truth labels for emails is a trivial task, which facilitates the deployment of *supervised* ML used by email providers to enhance their automatic filters [89]. For instance, Reference [9] analyzes the text of an email and reaches almost 99% accuracy with less than 0.01% false-positive rate. The authors of Themis [67] exploit deep learning to analyze both the text and the header of an email and exceed 99% accuracy. Finally, we mention the work in Reference [73], where the authors leverage supervised ML to detect *spear-phishing* attacks by analyzing an email from different perspectives and achieve over 90% detection rate at the cost of 1% false positives. Attachments can also be analyzed by any malware detection technique (Section 3.2).

As a small digression, we mention that the fight against phishing (and spam) has recently moved to Online Social Networks. This setting exhibits many similarities with the detection of phishing in emails, as it also involves NLP techniques. As an example, the authors of Reference [172] use deep learning to detect malicious tweets and obtain promising results with almost 95% detection rate but with a 5% false-positive rate. Similarly, MalT^P [96] specifically focuses on tweets luring victims to phishing websites, achieving over 95% detection rate and nearly 90% precision.

Takeaway. Using ML for cyberthreat detection has proven to be greatly successful (e.g., References [52, 113, 159]).

4 BEYOND DETECTION: ADDITIONAL ROLES OF MACHINE LEARNING IN CYBERSECURITY

Besides threat detection, there are many additional roles that ML can cover in cybersecurity. Indeed, modern environments constantly generate massive amounts of data, which may come from heterogeneous sources—including the very same ML models described in Section 3. Analyzing such data via (additional) ML can provide insights that further improve the security of digital systems.

Without loss of generality, we classify all these complementary roles of ML in four tasks: alert management (Section 4.1), raw-data analysis (Section 4.2), risk exposure assessment (Section 4.3), and cyber threat intelligence (Section 4.4). We now describe each of these tasks, schematically summarized in Figure 7.

We highlight an enticing characteristic shared by most ML applications described in this section: They do not require extensive and human-guided labelling procedures and hence belong to the *unsupervised* ML category. The potential of using raw data almost "as-is" makes all the ML methods discussed in this section readily applicable in many real scenarios.

4.1 Alert Management

It is well known that developing the "perfect" detection system is not possible (with or without ML). Hence, to prevent the automated execution of actions based on wrong predictions, the output of detection systems usually comes in the form of *alerts*. Depending on such alerts (e.g., their relevance, the involved hosts, or their number) a more appropriate response can be taken. However, modern environments generate thousands of alerts every hour (as shown in, e.g., References [26, 175]), making manual triaging an impossible task. To address this problem, ML can be used for *filtering*, *prioritization*, a or even *fusion* of alerts into more general events.

- *Alert Filtering.* By definition, an alert is not necessarily malicious, and a significant percentage of alerts correspond to false alarms. Because being notified by many irrelevant alerts is impractical and annoying, ML can help in filtering redundant alerts, e.g., because they are related to the same underlying problem. An example is Reference [157], which is specifically tailored for false alarms generated by ML-NIDS: Its effectiveness on real botnet traffic is a remarkable reduction of 75% of the time spent on triaging of false alerts, outperforming non-ML mechanisms by 45%.
- *Alert Prioritization.* If security administrators face too many alerts, then prioritization techniques can be applied to identify the most critical alarms. ML is beneficial, as it can automatically "learn" the most relevant ranking criteria with limited supervision. For instance, the very recent work in Reference [164] shows that ML correctly ranks the most sensitive alerts at the top position in 95% of the cases.
- *Alert Fusion.* The most intuitive way to manage large amounts of alerts is to *aggregate* similar alerts and then to find *correlations* between these groups to identify causal relationships relevant for security tasks. For instance, ASSERT [125] leverages clustering to identify which are the preferred protocols and network ports targeted by malicious activities. Their results highlight that modern attacks are increasingly relying on the Remote Desktop Protocol, as it enables lateral movement activities through pivoting [28].

All of the techniques above can be *combined* together. In this context, we mention the alert management solution in Reference [110] exploiting deep learning to condense and prioritize alerts: The resulting platform was tested and found usable by real security analysts.

4.2 Raw-data Analysis

The cybersecurity domain must deal with heterogeneous systems, each generating raw data of different nature (e.g., logs, reports, alerts). Such a setting represents a fertile ground for ML, whose capabilities could be leveraged to maximize the opportunities provided by such raw data. We can differentiate two areas of application of ML in this context: the support of *operational decisions* via log data analyses and the use of (unlabelled) data to *optimize labelling* efforts and foster deployment of supervised ML.

Operational Decisions. The abundance of *log* data in modern information systems makes ML promising in the context of operational security. The importance of log data analysis became evident after several high-profile security incidents that involved stealthy exfiltration of confidential data.[12] Beehive [178] was one of the first (unsupervised) ML systems focused on knowledge extraction from heterogeneous log data (generated by proxy, **Dynamic Host Control Protocol (DHCP)**, or **Virtual Private Network (VPN)** servers). The goal was combining all these logs in an anomaly detection fashion: Data points not associated with "typical" log patterns represented "incidents" that required manual intervention. Beehive was evaluated on two weeks of log data at the EMC Corporation and detected almost 800 incidents, 65% of which related to true security incidents (malicious activities or policy violations). In comparison, non-ML methods performed much worse, as they were only capable of detecting 8 correct incidents (with a recall of just 1%). Despite being unsupervised ML, Beehive still required manual feature engineering: The most relevant pieces of information from every log source had to be

[12] An example is the well-known RSA incident: https://www.theregister.co.uk/2011/04/04/rsa_hack_howdunnit/.

determined via expert knowledge. Such a problem was overcome with the advent of deep learning. A prominent example is DeepLog [64], which analyzes heterogeneous log data (e.g., Hadoop, or OpenStack logs) with a similar objective as Beehive. DeepLog achieves impressive results in a lab environment, with close to 100% detection rate after training on only 1% of the available data.

Labelling Optimization. Many threat detection techniques (Section 3) rely on supervised ML, which may require huge amounts of labelled data. Such a requirement prevents their applicability in real scenarios, because manual labelling can be prohibitive—especially in Network Intrusion Detection. In contrast, unlabelled data are common in cybersecurity, and many efforts proposed *semi-supervised learning* methods to increase the "return" of small sets of labelled data and hence enable deployment of fully supervised ML methods [29]. For instance, the botnet detector in Reference [184] reaches an F1-score of 0.83 with only 2,400 labels; in contrast, the detector in Reference [21] reaches and F1-score of 0.95 on the same network scenario but requires millions of labelled samples. A parallel line of research leverages the so-called *active learning* paradigm. The idea is to use a ML model (trained on a small labelled dataset) to "suggest" which samples should be labelled in a (large) unlabelled dataset to maximize its "learning rate." As an example, Reference [183] shows that it is possible to save significant labelling effort (from 30% up to 90%) by providing the ground truth of only a restricted amount of samples. An intriguing property of active learning is that it can be used even for already-deployed ML models by following the so-called *lifelong learning* principle: For instance, Tesseract [130] can boost its performance from 57% to 70% after being retrained on 700 samples "actively labelled" by a human expert.

4.3 Risk Exposure Assessment

Although the complete prevention of any cyber attack is an unreachable objective, a system can be significantly strengthened by focusing on its weak spots and anticipating the most likely threats. In this context, ML can help for several tasks, such as penetration testing or estimation of compromise indicators.

Penetration Testing. By automatically "attacking" existing security systems, ML can be a great asset for vulnerability assessment. For instance, Reference [74] apply *reinforcement learning* to synthetically craft attacks against traditional NIDS: The ML approach found the same amount of vulnerabilities in *half* the time of manual inspection and achieved a speedup of 90% with respect to a random attack procedures. More recently, Reference [21] adopted a deep reinforcement learning approach to automatically evade, and then harden, an ML-based botnet detector. Similarly, Reference [161] assessed the vulnerabilities of databases to SQL-injection attacks crafted via ML. There are even proposals of dedicated ML-assisted *platforms* for performing all such assessments [50]. According to a recent survey [111], the potential of ML for penetration testing is still vastly unexplored.

Estimation of Compromise Indicators. It is possible to use ML to estimate the most likely compromised hosts in a given system. The authors of Reference [177] study a corporate environment, using ML to analyze information from heterogeneous sources, such as the behavior of each individual host and of the entire network—as reported by end-point protection devices (McAfee) or even personal information on the specific user of each host. The findings revealed that visits to "business" websites represented the most common indicator of a compromised host (almost 30%), with second place for "travel" websites (nearly 15%)—this is intriguing, considering that such activities were performed during working hours. A potential opportunity is combining ML with honeypots (with a different scope than in Reference [117]): Such a strategy is exploited in Reference [72] to identify which hosts are more likely to be infected by botnet malware. Finally, Facebook exploits ML to identify fake accounts by correlating different sources [173], allowing us to reduce such annoyance by nearly 30%.

4.4 Threat Intelligence

The main task of threat intelligence is to collect and analyze information for *anticipating* novel attacks. This is clearly a powerful instrument for keeping defenses up-to-date in a proactive approach [39]. However, we observe

that a crucial aspect in the protection of enterprises revolves around the value of the items being considered: Hence, ML methods for cyber threat intelligence should be configured so as to prioritize the protection of the most business-critical infrastructures. Failure to take this into account may limit the usefulness of ML.

Nevertheless, applications of ML for threat intelligence can leverage either *internal* or *external* data sources (or both).

Internal Sources. Foreseeing future attack strategies via ML can be done with exclusive reliance on internal corporate data. For instance, Reference [158] leverages ML to artificially create alerts corresponding to past cyberattacks and then use such alerts to study an attacker's behaviour—potentially by using additional ML solutions. As an example, SAGE [116] exploits ML to compress over 300k individual alerts in less than 100 "attack graphs" representing the specific steps of an entire offensive strategy. Another possibility is to use deep learning to "disassemble" some code executables, allowing us to identify some potentially malicious patterns that can reappear in future malware: For instance, EKLAVIA [54] achieves a remarkable 80% accuracy in such a task. Finally, internal and external data sources can be mixed: The authors of Reference [88] exploit historical malware information (provided by Symantec) to foresee how future malware could affect a corporation, and their ML solution provided up to 4 times as many correct predictions as non-ML baselines.

External Sources. It is possible to use ML for the so-called open source intelligence. For example, the authors of Reference [146] focus on security incidents mentioned on Twitter. Their ML approach identified many malicious activities occurring in 2016, such as the Mirai botnet (October 2016) or the data breach at AdultFriendFinder (November 2016), where over 400 million accounts were exposed. Similarly, the deep learning method in Reference [165] analyzed tweets to study the development of ransomware attacks. It is also possible to use information from security feeds, such as the **Common Vulnerability Score (CVS)** stored on well-known databases.[13] For instance, in Reference [51] the authors use ML to predict the CVS with almost 1 week earlier than traditional cybersecurity feeds. Prediction of the CVS with ML can also be done via darkweb data as shown in Reference [12]. The authors use ML to crawl underground forums and correlate meaningful information with vulnerability descriptions. By validating the results via third-party signatures (e.g., Symantec), the proposed ML method successfully predicted the exploitability for about 40% of recorded vulnerabilities compared to about 10% of common feeds. Automated analyses via ML of underground forums (in different languages) aimed at uncovering "cyber-criminal markets" are also performed in Reference [135], allowing us to infer the prices of malicious exploits. Finally, we even mention the existence of *patents* that leverage ML to predict cyberattacks in modern environments [126].

Takeaway. There are many tasks complementary to threat detection that can be covered by ML. The main challenge lies in obtaining relevant information from unlabelled (e.g., References [29, 125]) or unstructured data coming from heterogeneous sources (e.g., References [12, 64, 173]). Such a challenge, however, also represents an intriguing opportunity.

5 INTRINSIC PROBLEMS OF MACHINE LEARNING IN CYBERSECURITY

As shown in Sections 3 and 4, ML can cover a plethora of roles in cybersecurity. Yet, in this specific domain, unleashing the full benefits of ML *in practice* is difficult. This difficulty stems from the underlying conflict between (a) the intrinsic characteristics of the cybersecurity domain and (b) the fundamental assumptions of ML.

Understanding such a conflict is crucial for a comprehensive assessment of all the tradeoffs pertaining to ML-based cybersecurity solutions. Therefore, we now discuss the *intrinsic problems of ML in cybersecurity*, for which we provide an overview in Figure 8. Specifically, we begin by presenting the problems affecting *any* ML solution for cybersecurity (Section 5.1); then, we elucidate the problems of ML solutions developed *in-house* (Section 5.3);

[13] An example is the CVE database, storing vulnerabilities as well as their exploitance likelihood: https://cve.mitre.org/.

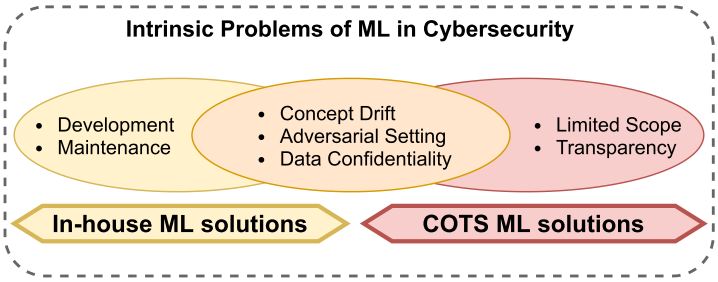

Fig. 8. Problems of ML in cybersecurity. Some are specific to either in-house solutions or to commercial-off-the-shelf (COTS) ML products. Others are shared by both of these categories.

and we conclude with the problems related to the adoption of **commercial-off-the-shelf (COTS)** ML products (Section 5.3).

We stress that all problems herein described are *intrinsic*: They can be mitigated to some degree, but the current state of the art does not allow us to completely resolve them.

5.1 General Problems of ML in Cybersecurity

Machine Learning follows the **independent, identically distributed random variables (iid)** principle [65]. Such a principle states that the data analyzed during the development of the ML model will be similar to the "future" data that the ML model will analyze after its deployment. If the iid assumption is not met, then the deployed ML model will exhibit a different performance than the expected one (measured during development). Such an iid principle impairs ML deployment in cybersecurity, because it interferes with three characteristics of this domain: the *concept drift*, the *adversarial setting*, and the *data confidentiality*. Let us elaborate on each of them.

Concept Drift. Modern systems are continuously evolving: new devices, services, and even users are added (or removed) every day. All such mutations contrast the iid assumption, preventing the reliable application of ML *in the long term*, because the training data quickly become obsolete. This problem is often referred to as *concept drift*, and while it can affect any application of ML, some domains are less touched by it. For instance, in computer vision "a cat will always be a cat," allowing us to use a ML model trained on the same data for decades—e.g., the ImageNet dataset (collected in 2011) is still used today [139]. This is not the case in cybersecurity and especially for threat detection: The *environment* constantly changes, and the *adversaries* also adapt. A schematic representation of the concept drift is shown in Figure 9.

For example, a new vulnerability may be discovered, meaning that some samples previously considered as benign should be treated as malicious[14]; a new segment may be attached to a network, with a considerably different behavior than the other segments, hence generating a lot of (false) anomalies; finally, attackers can devise novel strategies that cannot be detected by existing mechanisms (e.g., zero-day exploits). As a matter of fact, many research efforts highlighted the significant performance degradation of ML detectors in the presence of concept drift [18, 86]. The only practical remedy to concept drift is through constant update of ML systems with new data (labelled if supervised ML is used) that reflects the current trends.

Adversarial Setting. The cybersecurity domain implicitly assumes the presence of adversaries. Although most attacks are "stationary" (which explains why signature-based methods are still widely employed), motivated adversaries constantly refine and change their offensive strategies. Aside from the risk of zero-day attacks,

[14]For instance, hundreds of apps in well-known marketplaces were recently found to be malicious [92].

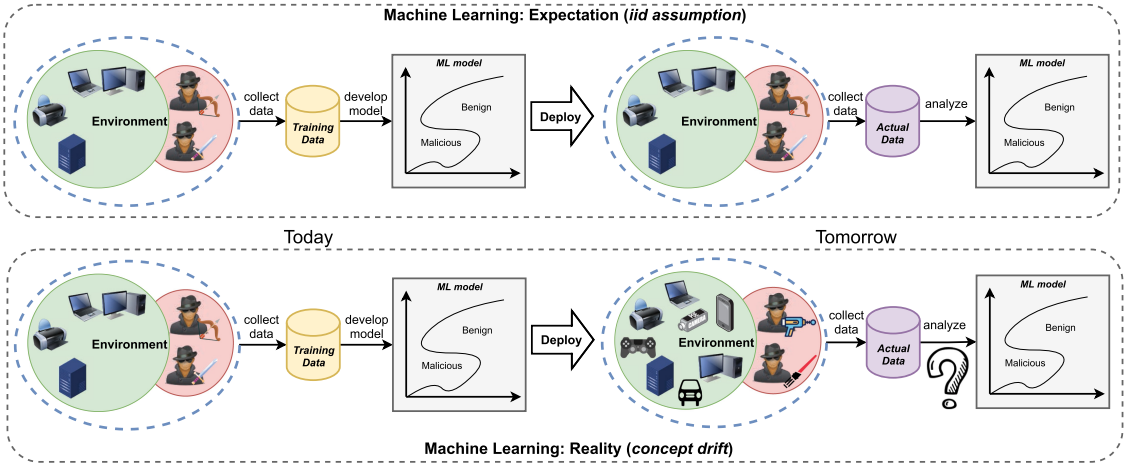

Fig. 9. Machine leaning in the presence of concept drift. The ML model expects that the data will not deviate from the one seen during its training. In cybersecurity, however, the environment evolves, and adversaries also become more powerful.

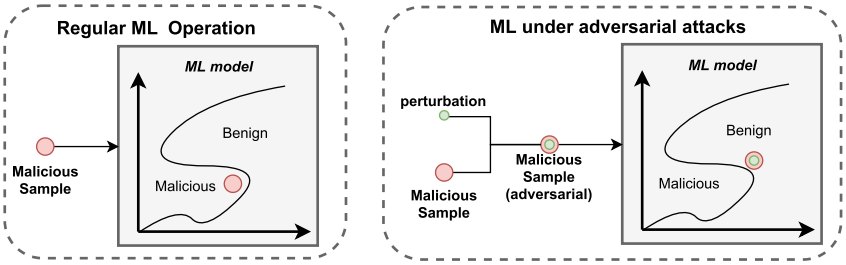

Fig. 10. Typical adversarial attack against a deployed ML model. By inserting tiny perturbations in the input data, it is possible to fool a ML model and induce an incorrect prediction.

deployment of ML also exposes to the threat of *adversarial samples* [154], which specifically target ML systems. Such a threat, schematically depicted in Figure 10, involves applying tiny "perturbations" to some input data with the goal of compromising the predictions of a ML model. Even imperceptible modifications can affect proficient cybersecurity ML detectors. For instance, Reference [25] evaded 20 ML botnet detectors by appending a few bytes of junk data to some network communications; whereas References [133] and [154] showed a similar effect against ML malware detectors. Even commercial products are affected, such as Google Chrome's phishing detector [101]. There exist a wide array of strategies to carry out attacks based on adversarial samples, which can affect either the pre- or post-deployment phase of a ML model [24, 154]. Despite the proposal of many countermeasures against adversarial samples, (e.g., References [19, 76]), no universal solution has been found so-far, and some mechanisms can even decrease the baseline performance (as shown in References [21, 59]). The best defense, according to Biggio and Roli [39], is a *proactive* approach: The adversary must be anticipated and evaluated (and, possibly, countered) *before* ML deployment.

To further stress the importance of such a threat, let us clear two *misconceptions*:

- it is common to refer to adversarial samples as "illegitimate." Such a notation is wrong from a security standpoint: Any sample (adversarial or not) analyzed by a ML model is considered as *legitimate* (i.e., trusted) by the underlying system that forwarded such a sample to the ML model. What is illegitimate is the *attack*, i.e.,

the application of a perturbation that is specifically crafted to thwart a ML model—but not the adversarial sample.[15]
- in related literature, it is common to search for the "minimal" perturbation that allows a sample to thwart a target ML model. However, real attackers are not subject to such constraint.[16]

The latter observation is crucial for demystifying the effectiveness of the so-called *certified defenses* [138], which only work if the perturbation is minimal or restricted within a very small boundary.

Confidentiality. The cybersecurity domain is characterized by its sensitivity to data-privacy, representing a strong barrier for long-term reliance on ML. Let us provide a few examples. The increasing usage of *encryption* can make some ML systems simply unusable. For instance, a ML-NIDS that inspects the payload of HTTP traffic will not work if the traffic is encrypted via HTTPS—and HTTPS is increasingly replacing the insecure HTTP protocol worldwide. Such a problem can also affect other use-cases of ML, such as phishing email detectors: If the emails are encrypted (e.g., via PGP), then it is impossible to analyze their contents with ML. Another problematic scenario can involve the analysis of confidential data: The constant changes in data regulation (e.g., the GDPR [167]) make it difficult to identify data that can be reliably used in the long term. For instance, consider the approach in Reference [177] (cf. Section 4.3), which leverages (among others) user information to estimate the infection risk. Such an approach could not be applied today without the explicit consent of all the users of a company. Moreover, both of these issues (confidential and encrypted data) also impair *labelling* procedures, because it is not possible to (manually) verify the ground truth of a sample if such a sample cannot be "seen" by a human expert. Finally, it is understandable that enterprises do not want to publicly disclose their data, generating an overall shortage of publicly available datasets that can be used to evaluate ML systems [147]. Although this latter problem primarily affects *research*, it also implicitly affects *practice*, because showing a ML system that works in different settings can foster its adoption in real scenarios. We discuss potential solutions to the limited data availability in Section 6.2.

5.2 Problems with in-house development of ML Systems

Despite the problems presented in Section 5.1, an organization may be willing to create a completely in-house ML solution. In this case, the organization can fully control the scope, data, and overall suitability of the resulting ML model. However, such an advantage comes at a price: The ML model must be first *developed* and must also be *maintained*. Both of these procedures are challenging in cybersecurity.

Initial Development. Developing the initial ML model requires three steps: (i) selecting an ML algorithm, (ii) finding the right data, and (iii) fine-tuning the performance. In some domains, these steps are almost straightforward. For instance, in computer vision it is established that deep learning algorithms outperform others; moreover, suitable data (potentially labelled) are easier to acquire—either because they are publicly available (e.g., ImageNet [139]) or because they can be cheaply produced (e.g., the popular captchas [47]). Unfortunately, none of these advantages apply to cybersecurity. For instance, some research works show that deep learning is worse (e.g., References [21, 134]) while others claim the opposite (e.g., References [123, 166]). Similarly, there is confusion with respect to which *features* should be taken into account (cf. Section 3.3 where Reference [33] use 130 features and Reference [144] use 30, achieving similar performance). Finding the right data is also inherently more challenging in cybersecurity. Such a challenge includes acquiring data of *high quality* and in the *right amount*. Labelling requires expert knowledge, and according to Reference [112] a company cannot afford

[15]To provide a concrete example, let us consider [22]: It is legitimate to increase the size of network communications, but it is illegitimate to do so with the intent of thwarting a ML model. However, a ML model considers all analyzed samples as trusted, because the ML model is oblivious of the intent of the data generation process.
[16]For instance, in Reference [22] adding 1 KB of data is more effective than adding only 1B. Hence, a real attacker is more likely to add 1 KB than just 1 B.

to label more than 80 malware samples *per day*. For reference, the initial deployment of Tesseract [130] required 50,000 labelled samples. Unlabelled data may be easier to acquire, but as shown in Section 4 it can come from heterogeneous sources and be in different formats, requiring a detailed preprocessing pipeline to collect, store, and forward such data to the ML model. Furthermore, the iid assumption (cf. Section 5.1) prevents a reliable use of data originating from different environments [149], hence even the (few) publicly available data can have questionable effectiveness. Finally, a common *misconception* is thinking that the performance of a ML model is linearly dependant on the size of its training data[17]: In some cases, *smaller datasets can yield to superior* ML models—we will show this in our case studies (Section 7.1). Nevertheless, any given dataset must also be *balanced*: In real environments, a malicious event is a rare occurrence and a given dataset should reflect such distribution [170].

To aggravate all of the above, it is not possible to determine *a priori* which combination (algorithm, features, dataset, balancing) yields the best performance after deployment. Hence, empirical and time-consuming evaluations—by training and testing multiple ML models—are always a necessity. As a result, finding the most optimal tuning for real deployments may require a huge amount of manual effort by trial-and-error.

Constant Maintenance. To mitigate the disruptive effects of concept drift (Section 5.1), it is fundamental to continuously update a given ML solution with data reflecting the current trends. Such procedures are costly but can be alleviated via lifelong learning solutions (cf. Section 4.2). However, a common *misconception* is that "update" procedures simply entail finding (and, if necessary, labelling) new data. This is an underestimation, because such procedures also necessitate (i) deciding what to do with "old" data and (ii) finding the "sweet spot" that yields the adequate performance. Indeed, maintaining old data can be detrimental in some cases (e.g., if some "benign" samples are discovered to be "malicious"), but completely removing it can also adversely affect the performance (e.g., some "old" phenomena can reappear in the future). Nonetheless, even small changes in the training data can decrease the performance of an ML system (e.g., this is the fundamental principle of poisoning attacks [24]). These issues require additional manual labour through trial-and-error.

A potential mitigation for all such tuning operations (both pre- and post-deployment) may come in the development of techniques focused on *explaining* the decisions of ML systems (e.g., Reference [107]), which are currently difficult to interpret—especially for deep learning [14]. This is an intriguing direction of research, which has very recently also touched the area of adversarial ML (e.g., References [16, 60])

5.3 Problems of Commercial-off-the-Shelf ML Products

Developing an in-house ML model may be prohibitive (e.g., in terms of computational or human resources), and COTS solutions represent a viable alternative. In this case, all the operations presented in Section 5.2 must be performed by the product vendor. However, we point out two drawbacks of such COTS solutions, aggravated in cybersecurity scenarios. Specifically, such solutions implicitly have a *limited scope*, and they may (inadvertently) suffer from lack of *transparency*.

Limited Scope. Relying on third-party solutions limits any end-user to their intended scope, meaning that some tasks simply cannot be accomplished with products currently on the market. For instance, any commercial ML model cannot be trained on the exact data used by an organization—at least initially. The organization can allow the vendor to collect their data and use such data to refine the ML model; however, this may not be possible due to confidentiality reasons (Section 5.1). Therefore, some commercial solutions can be used only if the deployment environment (of the organization) resembles the pre-deployment environment (of the vendor) used to generate the data for the corresponding ML model. For example, phishing websites are malicious "everywhere," meaning that it is possible to *transfer* [179] ML phishing detectors. However, such a transfer cannot be easily done for other cybersecurity tasks, such as NIDS [27]. This is because *every network is unique* [149], and a malicious

[17] According to the founder of Deep Learning, Andrew Ng, this is also becoming true for Deep Neural Networks [152].

behavior in one network can be benign in a different network. Due to such an issue, most COTS products leverage (unsupervised) ML and mostly for anomaly detection (e.g., References [2, 97]).

Lack of Transparency. COTS solutions come as a "black box," and the decision to deploy such solutions depends on their advertised performance. This fact leads to many issues, all sharing a common culprit: the cost of *misclassifications* in cybersecurity. In some domains, incorrect predictions do not have severe consequences: For instance, a recommender ML system (e.g., the one in AirBnb [77]) that makes an incorrect recommendation is not a cause of concern. In contrast, in cybersecurity a single FN can be the difference between a compromised and a secure system. At the same time, both employees and security analysts are annoyed by false alarms, which can even be exploited by attackers to conceal more severe threats [53]. By considering the performance metrics reported in Table 1 (cf. Section 2.1), we remark that each metric focuses on a single aspect, and even good scores can be meaningless if not contextualized.[18] Nonetheless, even if a COTS ML solution is fully transparent (i.e., all metrics are reported and contextualized), the performance will always refer to the environment of the *vendor*, which is likely to differ from the real deployment setting. Finally, we mention that—to the best of our knowledge—no COTS ML solution (including those not pertaining to security tasks) reports its robustness to potential adversarial attacks, which is a severe deficiency in cybersecurity scenarios.

Takeaway. In cybersecurity, ML can provide great benefits but also presents many risks due to the intrinsic adversarial setting and the dynamic ecosystem. Such risks must be taken into account today and should be addressed by future works.

6 THE FUTURE OF MACHINE LEARNING IN CYBERSECURITY

We have elucidated the benefits (Sections 3 and 4) as well as the problems (Section 5) of ML for cybersecurity. There are potentially infinite ways to advance the state of the art, such as increasing existing performance (e.g., Reference [62]), mitigating known issues (e.g., the poor explainability [16, 145]), as well as development of novel applications of ML in cybersecurity (e.g., the integration of quantum computing [75]).

As a constructive step forward, this section highlights which future developments can *completely revamp* the state of the art of ML in cybersecurity. Although every improvement is appreciated, we believe that the existing gap between research and practice can only be closed by the joint contribution of four players: *regulatory bodies*, *corporate executives*, *engineers*, as well as the *research community*. Specifically, we identify four future challenges that—if properly addressed—can revolutionize ML in cybersecurity. We now elucidate these challenges (schematically shown in Figure 11), explaining their root causes and our recommended course of action for each of the four "players" indicated above.

6.1 Certification (Sovereign Entities)

The 2020 EU White Paper on Artificial Intelligence [1]—also followed by a 2021 US DHS report [4]—indicates *trustworthiness* as one of the key requirements for future ML applications. Especially, emphasis is put on "high-risk" scenarios, where deployment of ML should conform with pertinent legal requirements. Cybersecurity applications naturally qualify as high risk, and hence procedures that certify the *performance* and *robustness* of ML systems should be developed and enforced by regulatory bodies. Let us elaborate.

[18] As an example, consider a detector evaluated on a dataset containing 9,990 benign samples and 10 malicious samples: Accuracy of 99.99% can be obtained by only detecting 1 malicious sample (of 10), despite its inability to detect 90% of the attacks. Another example is an FPR of 1%: It may appear low, but if the environment generates 300k alarms (as in Reference [116]), then the FPR corresponds to 3,000 false alarms. Note that also the inverse is true: An increment of just 1% in the TPR can be either an almost negligible or an extremely significant performance boost.

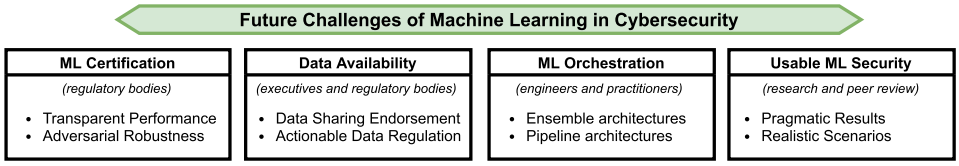

Fig. 11. Future challenges of machine learning in cybersecurity. Addressing all such challenges requires the cooperation of four players: regulatory bodies, corporate executives, engineers, and researchers.

Performance Certification. Comprehensive testing represents the only instrument for performance verification of a ML system. However, despite hundreds of works, there is a *lack of standardized evaluation protocols*. This is a problem especially for COTS products, as performance assessments may be carried out in biased environments or may consider unfair comparisons that inflate the results to favor a given ML solution. Meaningful assessments must consider the realistic distribution of data and take into account the (likely) temporal shift. Traditional cross-validation techniques, typical for ML in the computer vision domain, should be used only for tuning: Specifically, the performance should be validated via statistical tests. Establishing standardized evaluation protocols would foster pragmatic and fair comparisons, promoting overall ML deployment in practice. Nevertheless, the full details of such operations (e.g., the data used, the evaluation methodology, and the final results) should be transparent to the customers of COTS ML systems.

Robustness Certification. The increased interest toward ML led to (scientific) investigations of its robustness in adversarial scenarios, bringing to light the vulnerability to adversarial examples (Section 5.1). Yet no universal solution has been found so far, with some defenses being broken in the time span between their appearance as a preprint and their publication as a peer-reviewed article.[19] The first step to solve this problem is to *acknowledge that no ML solution is flawless*. Indeed, to quote a recent survey on the cybersecurity perspective of European stakeholders [69]: "security of ML and adversarial attacks was not mentioned as one of the key challenges by the interviewees," which epitomizes that such a threat is not perceived by the end-users of ML solutions. To address these issues, assessments of adversarial robustness must become mandatory in evaluations of *any* ML-based solution for cybersecurity. The most likely security risks, and their potential consequences, should be known *before* real ML deployments. Moreover, all the details of such assessments should be transparently provided.

Recommendation: To ensure better transparency and reliability, regulatory bodies must enforce the development and adoption of standardized procedures that certify the performance and robustness of ML systems.

6.2 Data Availability (Executives and Legislation Authorities)

The effectiveness of any ML solution depends on the data used to train the corresponding ML model. However, among the toughest challenges faced by ML in cybersecurity is *finding appropriate data*. Despite the recent interest in ML led to the release of more open datasets (e.g., References [10, 31, 142]), such datasets exhibit limitations [163]. For instance, inaccurate labels, fast obsolescence, small and synthetic environments (e.g., Reference [115]), or even flawed generation process—as shown in Reference [66]. All these problems can only be mitigated to some degree (e.g., Reference [37]) and cannot be solved by the scientific community. The lack of adequate data (i.e., *real* and *labelled*) makes evaluations of ML conducted in research environments to be of questionable value, preventing sound assessments of ML capabilities, and ultimately hindering its deployment. Addressing the shortage of data is possible, but it requires the joint intervention of industrial stakeholders and

[19]For instance, defensive distillation was proposed in 2016 [128] and broken few months later [48].

regulation authorities: The former should promote *data sharing*, and the latter should devise more *actionable data regulation* policies.

Data Sharing. A solution to the lack of adequate data is the promotion of data-sharing practices. In cybersecurity, some portions of data can be easily shared: For instance, Sophos has recently released over 20M (labelled) malware samples [78]; similarly, the recent CrimeBB dataset [129] contains 1 million accounts crawled from darkweb forums for 10 years. In contrast, other pieces of data (especially *benign* data) are more confidential and hence their *disclosure requires explicit permission from corporate executives*. Acquiring such permission is a tough barrier, especially due to privacy and secrecy issues. However, we observe that sensitive information can be anonymized (e.g., Reference [136]), and recent advances in *federated learning* overcame such problems [57].

There indeed exist some success stories of data-sharing *platforms* focused on security information, such as the EU-OF2CEN project [151]. Similar platforms represent a great *opportunity* for some companies, as they open the doors to a new market entirely dedicated to ML datasets, potentially with (updated) ground truth (e.g., Reference [150]). From this perspective, a promising initiative is STIX CyBox [143]: Its goal is creating a threat intelligence platform shared by multiple parties, facilitating the entire process of incident detection and response. Nonetheless, such platforms must (i) contain unbiased data—otherwise, there a the risk of manipulating future developments [109]—and (ii) comply with the existing regulation, hence requiring the involvement of the respective authorities.

Actionable Data Regulations. The strategical importance of data gave birth to multiple regulations that "protect" data owners and limit abuse of sensitive information. Despite ensuring more privacy rights, such regulations introduced additional constraints on data gathering and processing, resulting in yet another barrier to ML developments—both for research and practice. Specifically, the (already costly) *data-labelling procedures are crucially affected by such regulations* (Section 5.1). Even if action is taken by executives to disclose their corporate data, existing regulation policies are difficult to interpret and likely to change in the future [124]: For instance, information that can be shared "today" may not be shareable "tomorrow," hindering long-term projects. However, we observe that some GDPR compliant data-sharing platforms exist (e.g., Reference [79]). Hence, the regulatory authorities should promote such efforts even in the cybersecurity context, for instance by providing actionable policies that ensure the compliance of (open) data in the long term.

Recommendation: To address the shortage of adequate data, companies should be more willing to share data originating in their environments (e.g., Reference [151]), whereas regulation authorities should promote such disclosure by defining proper policies and incentives [124].

6.3 Usable Security Research (Scientific Community)

The combination of ML and cybersecurity is a fertile opportunity for research, and recently inspired many related papers. Such a trend, however, is a double-edged sword. On the good side, the rising scientific interest demonstrates the high potential of ML for cybersecurity. On the bad side, such abundance can be *detrimental* for real ML deployments, as it may raise more questions rather than provide answers. Specifically, we identify two problems of existing research: the *lack of pragmatic results* and the *limited consideration of realistic scenarios*.

Pragmatic Results. One of the primary goals of research is to "outperform the state of the art." In the context of ML, such a goal requires us to propose a novel ML method and then show that this method achieves a better performance than prior works—an objective that can be achieved without providing any "true" contribution to the state of the art. For example, by slightly changing the training data it is possible to achieve a superior performance; similarly, an existing solution may be sub-optimally reproduced (by using, e.g., a different dataset,

or different tuning parameters). Note that all such "flaws" can be *unconsciously* introduced by researchers.[20] This phenomenon, also referred to as *benchmark lottery* [58], results in an overall confusion on what really works best and impairs real ML deployments. Among the main culprits of such a phenomenon is the *poor reproducibility* of researches, as very few works disclose the entire information required to replicate their experiments. Therefore, novel researches cannot properly reproduce previous works, and the peer-review process cannot assess whether the experimental protocol is correct and unbiased. At the same time, however, we point out that most scientific venues do not allow (or require) inclusion of any supplementary and technical resource. Hence, even researchers must face a difficult decision about what low-level information should be included in the actual submission—which is subject to page limitations.

Recommendation: The peer-review process should facilitate and enforce the inclusion of the material for replicating ML experiments. At the same time, such a material should be evaluated to ensure its correctness—potentially by a separate set of reviewers with more technical expertise.

Realistic Security Scenarios. As a direct consequence of the benchmark lottery phenomenon, many research papers simply focus on providing "better numbers" than past work, overlooking the assumptions made by such past work. In the context of cybersecurity, this is a problem, because realistic circumstances must be considered, and any result that stems from unrealistic scenarios is of questionable value. For instance, there is a *superficial treatment of training data*: Only few papers (e.g., References [18, 130]) consider the *concept drift*, which is intrinsic in cybersecurity; moreover, many recent papers (e.g., Reference [156]) still use *outdated datasets*, such as the NSL-KDD, which is over 20 years old and does not reflect any current environment. The result is that all papers propose ML methods that achieve near-perfect performance—but what is the practical impact of all such research? We acknowledge that public (labelled) data are difficult to acquire, but over the past few years several datasets have been openly released (e.g., References [115, 147]). The impression is that the cybersecurity setting is turning into a yet-another research playground where new ML methods are evaluated on some "security-related" data, but realistic security considerations are only made in the introduction to provide some justification for a given publication venue. Specifically, there is a *lack of realistic threat models*. Such lack is epitomized in the emerging field of adversarial ML (Section 5.1), where most attacks against security systems assume extremely powerful opponents. For instance, the authors of Reference [20] show that the majority of attacks against ML-NIDS require adversaries with direct access to the ML-NIDS itself, which is an assumption that violates the basic security principles. Similarly, Reference [133] show that adversarial attacks have a different effectiveness when the opponent cannot manipulate the data-processing pipeline (which is usually not accessible). Hence, it is not surprising that the industrial stakeholders are either confused or do not care about adversarial examples—as evidenced by two recent surveys [69, 95] and the detailed case study in Reference [42].

Recommendation: Future researches on ML applications for cybersecurity should have a closer connection with the real world. The assumed *threat model* should be realistic, the *dataset* should resemble recent trends, and the *concept drift* should be taken into account.

6.4 Orchestration of Machine Learning (Engineers)

ML is not meant to fully replace existing systems or human experts. Rather, it should provide an additional "perspective" that can be used to identify otherwise overlooked phenomena. However, ML methods exhibit huge variance (e.g., different performance [134, 162] or adversarial robustness [25]), and a single ML solution cannot

[20] A recent paper describing the pitfalls of ML in cybersecurity is Reference [30].

protect against all threats that can target modern organizations.[21] Addressing all such issues is possible by *orchestrating* diverse ML solutions. Indeed, any ML model (irrespective of its goal) ultimately represents just a single component of a cybersecurity system—which can be a "hybrid" system that leverages also non-ML techniques. However, such orchestration requires the expertise of *ML engineers*, who must coordinate different outputs to extract actionable information. Specifically, ML (and non-ML) models can be combined either in an *ensemble* or in a *pipeline* architecture, depending on the final goal of the system.

Ensemble architecture. One of the most proficient ways to combine different ML models is the so-called *ensemble* [38]. The idea is leveraging many simplified learners *with a common goal*: Each ML model of the ensemble analyzes the same data but by focusing on a specific problem. For instance, it is possible to create ML-NIDS using ensembles of ML models, in which each model has the same goal (i.e., intrusion detection) but focuses on a specific threat (e.g., botnet or **Denial of Service (DoS)** attacks [113]). Despite the proven performance benefits of such architectures, a tough challenge faced by engineers is the *lack of standardized feature sets* that can be used to devise all such systems. Each model of the ensemble must ultimately analyze the same data, and, depending on the features provided as input, the performance can greatly differ (as shown in Reference [41]). Our industrial case studies in Section 7 consider a similar architecture.

Pipeline architecture. When the system envisions ML models having systematically *different inputs and outputs*, they must be organized in a pipeline architecture. For example, it is possible to create an ensemble of ML models for threat detection (Section 3) and then use their outputs for threat intelligence (Section 4). Similar systems already exist, either as COTS products (e.g., SIEM[22] or SOARS[23]) or as scientific proposals: For instance, ARCUS [175] is a security-focused orchestration platform that could benefit from the integration of many of the ML solutions discussed in this article. However, such architectures are challenging to implement by engineers: *Each individual component is affected by all the issues* presented in Section 5, therefore multiplying their impact.

Recommendation: Orchestrating complex systems that use (combinations of) ML and non-ML solutions is beneficial for cybersecurity. Hence, ML engineers and practitioners should clearly highlight how to combine all such components to maximize their practical effectiveness.

7 CASE STUDIES: INDUSTRIAL APPLICATIONS OF ML FOR CYBERSECURITY

As a final contribution of this article, we present two case studies that showcase real and successful industrial applications of ML in cybersecurity. Many commercial products are advertised as leveraging ML. Yet most of these products are provided as black boxes, preventing any understanding of how ML is actually applied *in practice*. Specifically, our case studies involve the two following scenarios:

- using ML for detecting Cache Poisoning Attacks against Named Data Networks. The approach is integrated in a NIDS developed by Montimage (Section 7.1);
- combining sequential deep learning with non-ML methods for protecting Industrial Control Systems. The approach is integrated in a cybersecurity device developed by S2Grupo (Section 7.2).

Both of these solutions use ML for anomaly detection with limited supervision. Our goal is to provide a high-level overview on such "black box" ML systems by elucidating their internal functionalities.[24]

[21]The most exemplary use-case are zero-day attacks, which can easily evade supervised ML methods: Zero-day samples cannot—by definition—be included in the training data. Anomaly detection through unsupervised ML is more feasible but at the cost of many false positives.
[22]System Information and Event Managers: https://www.forcepoint.com/cyber-edu/siem.
[23]Security Orchestration Automation and Response Systems: https://www.rapid7.com/solutions/security-orchestration-and-automation/.
[24]The commercial nature of such systems—which are built on the end-users data—makes some low-level details to be protected by NDA, but explicit requests can be made by contacting the respective vendors.

7.1 Detection of Cache Poisoning Attacks in Named Data Networks

Information Centric Networking (ICN) is a revolutionary paradigm in the context of communications: While most of the Internet follows a host-to-host perspective, ICN adopts a host-to-content vision [6]. The ICN architecture is more suitable for massive content diffusion (e.g., video streaming), representing the major use cases of modern networks. Despite providing multiple benefits in terms of bandwidth efficiency and scalability, ICN can fall victim to DoS attacks and, in particular, to poisoning attacks [5, 174]. In this case study, we analyze a real ML detection system that protects against such attacks targeting ICN architectures. The specific techniques are integrated into the Montimage Monitoring Tool,[25] which is a module of the IDS framework developed by Montimage [118, 169].

Scenario and Challenges. This case study focuses on the well-known ICN approach of **Named Data Networking (NDN)** [181]. This NDN approach leverages a pull-based mechanism using two kinds of packets: *Interest* (a request for a content) and *Data* (the response with the content). When a given user wants to retrieve some content, the user (i) specifies the desired content's name (e.g., "/data/video.mp4") in an Interest, (ii) sends such Interest through the NDN network, and (iii) receives the corresponding Data—which can be provided either by the content *producer* or by any intermediate NDN node storing a copy of such Data. The practical implementation of NDN exposes to the risk of new security attacks, such as the **Content Poisoning Attack (CPA)** [174]. In CPA, a malicious *producer* (content creator) colludes with a malicious *consumer* (a user requesting content) to force any NDN node on their path to insert malicious content in their **content storage (CS)**, hence causing poisoning attacks. This results in nodes answering some requests with such malicious content: For example, a victim may ask for a specific webpage and instead be redirected to a malicious phishing website. CPA are a dangerous threat to NDN, as shown in Reference [119]: Analyses on real system highlighted that identifying CPA is *impossible* via static and human-based approaches. This is due to the intrinsic characteristics of NDN, as each node in the network topology reacts differently. Moreover, NDN are also susceptible to **Interest Flooding Attacks (IFA)**, which represent a variant of DoS in which the NDN is "flooded" with interest requests [148] for existing or even non-existing content that can disrupt the distribution of content. Although IFA are easier to identify than CPA, countering *both* IFA and CPA is challenging and requires the usage of more dynamic analytical techniques—such as ML.

Montimage ML-Solution. The ML-solution developed by Montimage leverages *ensembles* of ML models organized in a **Bayesian Network Classifier (BNC)** [120]. The intuition is that detection of CPA is only possible by monitoring the behaviour of each node in a NDN network—and, specifically, by analyzing and cross-correlating the evolution of different metrics for each node.

Such a goal is achieved by means of specific probes deployed on each node and monitoring its complete activity. In particular, each probe collect metrics related to the Data plane of NDN: CS, **Pending Interest Table (PIT)**, Faces. The latter, in particular, are an abstraction of a communication channel that NDN uses for packet forwarding. Such abstraction represents data coming from diverse "faces," i.e., overlay tunnels over TCP and UDP, delivery of NDN network layer packets (e.g., Interest, Data packets), inter-node communication channels that send packets to other nodes, and intra-node communication channels that send packets to another process on the same node.

The information captured by these probes is then analyzed by ensembles of *micro-anomaly-detectors*, each focusing on deviations from the normal behaviour of a single metric captured by each probe. It is true that CPA can impact many metrics and in different ways, raising hundreds of (likely) false alarms by each micro-detector. However, *correlating* all the alarms with a BNC allows us to (i) increase the detection performance while (ii) mitigating the high rate of false alarms generated by individual micro-detectors.

[25] https://montimage.com/products/MMT_DPI.html.

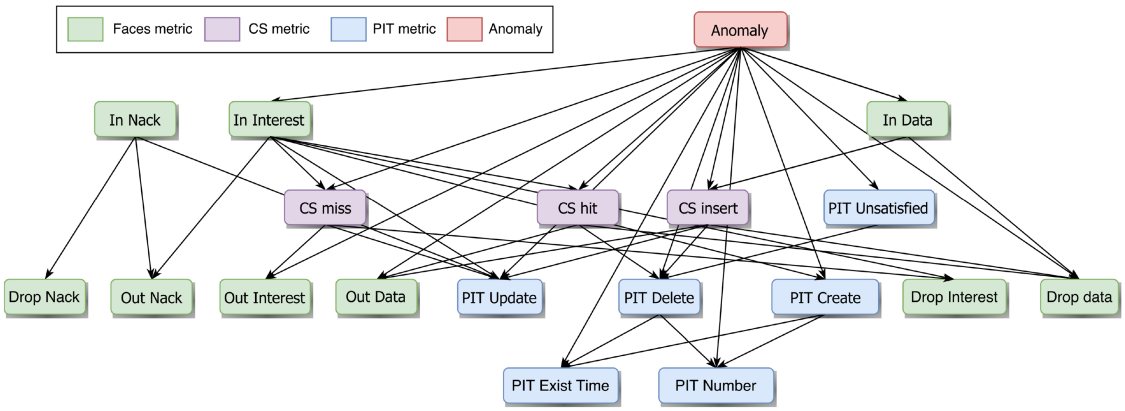

Fig. 12. Architecture of the Bayesian network classifier adopted by Montimage to detect CPA in NDN. Each node represents a micro detector that focuses on a single metric. The Anomaly node correlates the output of all other NDN nodes.

A schematic representation of the considered BNC is provided in Figure 12: the "anomaly" node (denoted in red) represents the anomalies that can occur in the entire NDN, whereas the remaining nodes represent the individual micro-detectors. Hence, each node focuses on a single metric, specifically Faces, CS, or PIT (denoted in green, purple, and blue in Figure 12). The (directed) edges in the BNC represent the causal relationships between the Anomaly node and a metric (or pairs of metrics). An edge connects the "causing" node to the "affected" node. The causal relationships are deduced based on the processing of each packet arriving to the NDN node.

Evaluation and Results. It is necessary to conduct a preliminary assessment of the learning efficency of the BNC before its deployment. This is because NDN generate a lot of traffic, and even though the BNC can "condense" the raised alarms it is still important that such alarms—and, specifically, false alarms—are within acceptable levels. To this purpose, Montimage first collects huge amounts of *real* data from the probes and then uses such data (assumed to be benign) to train (and test) a BNC. Specifically, multiple BNC are assessed, each considering a different training size: The goal is finding the optimal size that minimizes the rate of false alarms. The results of such an assessment are reported in Figure 13, showing the misclassification error (as measured via fivefold cross-validation) as a function of the training size. We observe that an optimal value is achieved when the training set contains ~280 samples.[26] For higher values, the error increases due to overfitting (this phenomenon confirms the misconception outlined in Section 5.2). Thus, for the considered deployment scenario, Montimage uses training sets of 280 samples—corresponding to 23 minutes of *real* reportings.

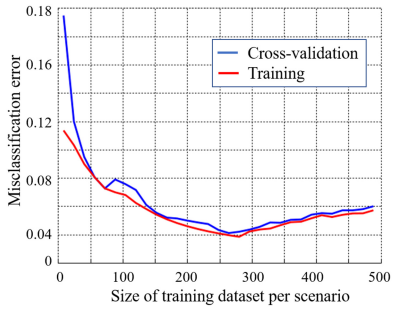

Fig. 13. Preliminary assessment of the BNC to identify the optimal size of the training dataset.

[26]We observe that such samples represent *alarms* corresponding to multiple signals, and not to raw events.

Table 2. CPA Detection Performance for Two Different Routing Strategies and Increasing Attack Payload

Routing Strategy	Attack payload (# Interest/s)	5	10	20	50
Bestroute	% TP	95.0%	95.3%	97.0%	98.3%
	% FP	1.0%	1.0%	1.0%	1.0%
Multicast	% TP	63.3%	72.8%	79.3%	96.3%
	% FP	1.0%	1.0%	1.0%	1.0%

To evaluate the performance in production settings, Montimage reproduces the NDN topology in Reference [182] and creates two distinct environments, each adopting a specific NDN routing strategy: *bestroute* or *multicast*. Then, each environment is monitored for 10 minutes, and the attack is simulated in the last 5 minutes. Specifically, multiple CPA are launched, each considering increasing *payloads*, denoting the number of requests for content (i.e., Interests) per second; in our case, we consider payloads of 5, 10, 20, and 50 Interests per second. In comparison, legitimate clients produce 10 Interests per second (on average): Hence, the malicious traffic ranges from half to five times the legitimate traffic. The traffic generated during such simulations is collected and used to assess the quality of the BNC: The goal is to verify whether the BNC is capable of identifying the CPA, which occurs in the last 5 minutes.

To provide a twofold perspective of the performance (see Section 5.3), Montimage measures the **True-Positive Rate (TPR)** and **False Positive Rate (FPR)** (–cf. Table 1 in Section 2.1). The results of such evaluation, performed on a testing set of 240 samples, are reported in Table 2. We observe that the TPR increases for greater payloads, because the CPA become more conspicuous. Nonetheless, it is appreciable that even CPA with low payload can be effectively detected. Finally, the low FPR is crucial for real deployments as they are annoying to human operators. All such results are due to the advantages provided by the BNC, because BNC use a probabilistic approach that allows us to take into account the underlying random nature of the observed metrics. Such a property makes BNC tailored for multi-variate anomaly detection in *real* environments. In contrast, other ML algorithms present significant drawbacks: For instance, "deep" neural networks are excessively difficult to develop in such settings (also due to their poor explainability), whereas other "shallow" algorithms, such as SVM, simply do not allow us to efficiently represent and correlate all the metrics affected by CPA.

The major limitation of BNC is its intrinsic function as anomaly detector: Indeed, an anomaly is not necessarily malicious. For instance, in a NDN setting, a sudden demand for a video from legitimate users could lead to a temporary increase in traffic, indicating an abnormal activity. To mitigate this problem, Montimage considers four possible "states": *normal state*, *IFA attack state*, *CPA attack state*, and *number of users increase*. Each state is denoted by different "anomalous" combinations taking into account a total of 18 metrics: A similar solution allows us to maintain the FPR to acceptable levels (as shown in Table 2). We take this opportunity to make a crucial remark for real ML deployments: One may believe that defining more "states" and/or increasing the amount of considered metrics leads to better results. However, according to Montimage a similar approach can yield proficient results only in a lab environment, because it induces overfitting, and the true *deployment* performance may suffer excessive FPR.

Finally, an intriguing future development of such an ML solution involves the consideration of "stateful" analyses that take into account the time-axis (as done, e.g., in Reference [56]) and allow to detect even anomalies occurring in the temporal domain. The next case-study by S2Grupo will consider a similar application.

7.2 Combining ML with Non-ML Methods to Protect Industry 4.0 Environments

With the rapid growth of the Industry 4.0 paradigm, industrial environments are even more exposed to **Advanced Persistent Threats (APT)** [132]. Specifically, recent developments of ICS represent an attractive target

for attackers [68]. In this case study, we share the experience in the design and operation of CAIAC,[27] a non-intrusive device that leverages sequential ML to protect ICS against APT and other cyber-threats.

Scenario and Challenges. This case study highlights the advantages of ML applications for anomaly detection in time-series data. The intuition is that APT leverage zero-day vulnerabilities and hence cannot be detected via misuse-based detection approaches—irrespective of being human or data driven. However, pointwise and static anomaly detection approaches are not enough to detect advanced cyberattacks, and the additional perspective provided by the temporal domain may facilitate the detection of refined offensive strategies [132].

In the specific ICS scenario, there are two crucial requirements that must be met by security systems. First, they should operate in a non-intrusive way, avoiding additional overhead and ensuring the regular functionalities of the ICS: This is a tough requirement, because ICS include hundreds of devices and while excessive false alarms are annoying, slow reaction times may imply a fallout of the entire ICS. Second, they must take into account the complexity and variability of the data in ICS, which is difficult to manage to the intrinsic heterogeneity of ICS. Such a requirement cannot be met just with traditional approaches for time-series anomaly detection based on heuristics: To address this problem, S2Grupo leverages the capabilities of deep learning.

S2Grupo ML-Solution. The ML solution developed by S2Grupo, CAIAC, is an intriguing example of *ML orchestration* (Section 6.4): CAIAC not only leverages the benefits provided by "small" ML models (as done in Section 7.1) but also exploits the potential of non-ML methods for time-series analyses. In particular, the idea is to combine deep learning algorithms, epitomized by **Long-Short Term Memory (LSTM)** neural networks, with statistical approaches for time-series forecasting, such as **Seasonal Autoregressive Integrated Moving Average (SARIMA)**. The result is an ensemble of ML and non-ML models, exploiting the benefits of both approaches and overcoming their limitations: Statistical models can be more manageable, but when the data have high complexity deep learning is superior. Such a design choice is particularly suited for real ICS deployments due to a threefold advantage with respect to "one-size-fits-all" ML architectures. Specifically:

- individual ML models are easier to train, because they must deal only with a tiny portion of the data, resulting in better performance and lower false alarms;
- it allows combining different algorithms, each addressed to a specific problem and data type.
- it makes the resulting system more "future proof," because each ML model can be individually updated, removed, or replaced.

Furthermore, CAIAC is based on *passive* monitoring in near real time, hence preventing excessive information overhead while still allowing timely responses.

Let us explain CAIAC in more detail. The intuition is to analyze the network traffic of the considered ICS from different perspectives, each associated to a specific time series. This time series can differ on the basis of two criteria: the network *metric* (e.g., transmitted packets) and the *granularity* used to aggregate the corresponding metric in time slots of fixed length. All such time series are used to devise multiple ML and non-ML models: The performance of each model can be assessed individually by forwarding its detected anomalies to a higher-level correlation layer (similarly to Reference [132]). The goal of this layer is determining the nature of such anomalies: They can either be legitimate (i.e., a "normal" malfunctioning of a component that must be investigated) or illegitimate (i.e., an attack is taking place). Such a procedure allows us to identify the most suitable models that will be integrated in CAIAC, depending on the pros and cons of each model. Indeed, LSTM models may yield a superior performance but require a training phase, whereas statistical models are easier to develop and only require some tuning. Hence, such (non-ML) models are the preferred choice when they exhibit similar performance to LSTM.

[27] https://s2grupo.es/en/research-development-innovation/industrial-cybersecurity/caiac.html.

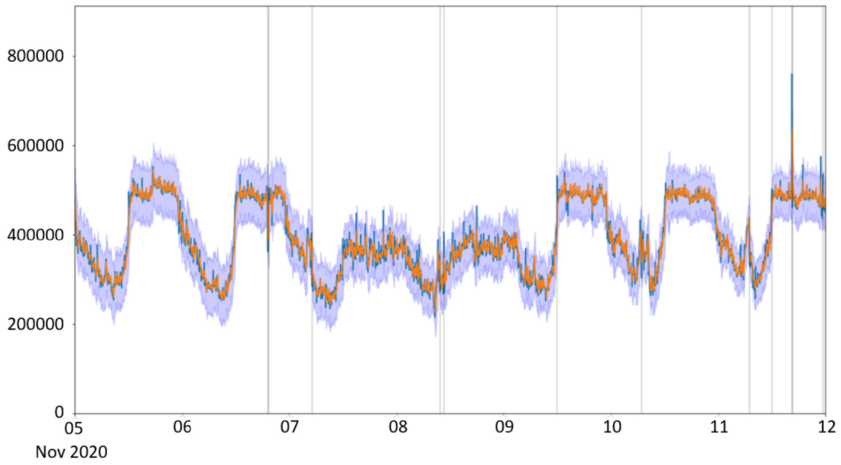

Fig. 14. Anomaly detection with (non-ML) SARIMA, using a sliding window of 30 minutes. The time series represents the transmitted packets (y-axis) within 5-minute slots, over a period of 1 week (x-axis), corresponding to a total of 2K samples. Dark blue correspond to actual values, orange denotes the values predicted with SARIMA, and light blue denotes the confidence interval of SARIMA's predictions. Vertical gray lines correspond to the anomalies detected by SARIMA.

Evaluation and Results. To develop CAIAC, it is necessary to first assess the characteristics of the specific ICS: Indeed, it is not possible to use models trained on different environments (as explained in Section 5.3). Hence, S2Grupo monitors and collects the network traffic of the considered ICS and creates multiple time series, each considering a given metric and granularity. Some metrics are commonly adopted in NIDS (e.g., transmitted packets or bytes, in-/out-degree [132]); others are specific of ICS and require dedicated industrial dissectors that extract the relevant information (e.g., protocol, parameters, command density). Finally, each metric is aggregated in time slots of varying length, from 1 minute to 1 hour.

After this data collection phase, which in the considered setting typically amounts to about 10 GB of data per day, S2Grupo performs the exploratory analysis focused on determining the most proficient (ML and non-ML) algorithms for studying each time series. Let us elucidate the differences between two specific applications of SARIMA and LSTM, starting from the non-ML algorithm.

Specifically, SARIMA analyzes a time series by adopting a sliding window approach: All data points within a given time window are considered by SARIMA to predict a "future" value, which is provided alongside a *confidence range*. We provide an example of SARIMA in Figure 14, showing the time series of the *transmitted packets* aggregated in time slots of *5 minutes*, over a period of 1 week; the sliding window considered by SARIMA is of 30 minutes. The actual values are reported in dark blue, whereas the values predicted via SARIMA are shown in orange; the confidence window of each predicted value is shown in light blue: therefore, actual values that fall outside of this range are treated as anomalous. In particular, vertical gray lines denote the anomalies detected by SARIMA.

From Figure 14, we observe that SARIMA accurately detects *stationary* deviations. However, SARIMA can only detect *non-stationary* changes when they happen within its sliding window. Furthermore, non-stationary (but legitimate) changes that occur after a long stationary interval are falsely detected as anomalies by SARIMA. Despite some incorrect predictions, the considered application of SARIMA obtained a performance that was deemed appropriate for the given task and integrated in CAIAC.

Let us showcase an application of deep learning via LSTM. Since LSTM do not provide a confidence interval for each prediction, S2Grupo developed a custom anomaly threshold that takes into account the deviation between predicted and actual values, as well as the degree of accumulation of such deviation in the past history. An

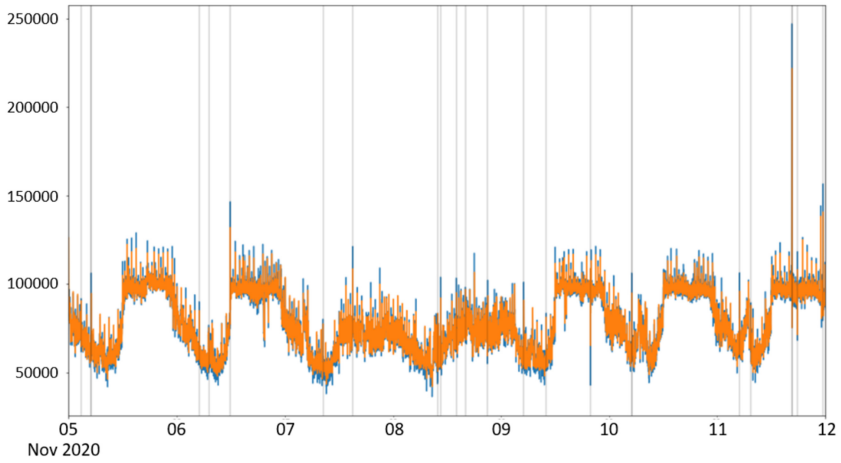

Fig. 15. Anomaly detection with a deep LSTM neural network. The time series represents the transmitted packets (y-axis) within 1-minute slots, over the period of 1 week (x-axis), corresponding to a total of 10K samples. Actual values are shown in blue, and the LSTM predictions are shown in orange. Vertical gray lines denote the anomalies detected by the LSTM.

example of such an LSTM application is given in Figure 15, showing the time series of the transmitted packets (same as Figure 14) but with a time slot of 1 minute. The actual values are shown in blue, whereas the LSTM predictions are in orange. Vertical grey lines denote the anomalies detected by the LSTM, i.e., when the actual values falls outside the given anomalous threshold predicted with the LSTM.

From Figure 15, we can observe that, by reducing the time slot from 5 to 1 minute, the resulting time series is less predictable, making statistical methods unfeasible and requiring the advanced capabilities of deep learning. Indeed, the considered LSTM can detect anomalous values without being affected by non-stationary changes—even after long stationary intervals. This example highlights the capabilities of (deep) ML to deal with data with high dimensionality: The LSTM takes into account a long "past" history, allowing to better infer the "normal" behaviour. In contrast, applying SARIMA on the same time series resulted in very poor results due to the intrinsic variability of the sequence, which forced us to aggregate data in 5-minute time slots.

However, it is important to take into account that the LSTM require a training step, whereas SARIMA only requires some parameter adjustment. In this use-case, the LSTM in Figure 15 was trained with data collected over *3 weeks*. Such a characteristic implies that a similar LSTM model requires at least 3 weeks of data collection, since no previous network traffic data were available to train the model—alongside the additional computational resources to store such data and train the LSTM model (which were within acceptable levels). Hence, CAIAC would initially make use of SARIMA and then replace it after enough data have been collected to develop a more proficient LSTM model.

We can conclude that machine (and deep) learning are powerful instruments for protecting modern ICS, but methods that do not leverage ML are equally important to compensate some of the limitations of ML. As such, future developments should not exclusively focus on ML and overlook the benefits provided by other data-driven methods.

8 CONCLUSION

This article elucidates the role of ML for Cybersecurity by providing a broad and high-level overview of the benefits, problems, and future challenges of ML in this domain. Our article is oriented at the entire cybersecurity sphere, and to make our contribution understandable by a broad audience, we limit technical terms to a minimum.

Table 3. Summary of Security ML Misconceptions Discussed in the Article

#	Misconception	Ref.
1	Deep Learning vs Shallow Learning	Section 2.1
2	Machine Learning and Anomaly Detection	Section 3
3	Legitimacy of Adversarial Samples	Section 5.1
4	Minimal Adversarial Perturbations	Section 5.1
5	Size of training data	Section 5.2
6	Updating ML models with new data	Section 5.2

Moreover, we also clarify many misconceptions (summarized in Table 3) that are becoming common due to the increasing abundance of works that link ML with cybersecurity applications.

After introducing the basic concepts of ML, we provide a concise summary of their applications to detect three types of cyber threats: Malware, Phishing, and Network Intrusions. Then we elucidate some additional cybersecurity areas that can leverage the autonomous capabilities of ML, such as raw-data analysis, alert management, cyber risk estimation, and threat intelligence. What follows is a description of the fundamental problems affecting ML within the specific context of operational cybersecurity, which should be known to weigh the pros and cons of the still-emerging ML solutions. Some of these problems stem from the intrinsic conflicts between the fundamental principles of ML and the cybersecurity domain and can be addressed only by the joint effort of different worlds: regulatory and authoritative bodies, corporate executives and engineers, as well as the entire scientific community. To this end, we highlight the future challenges of ML in cybersecurity, which we integrate by comprehensive recommendations addressed at each of these separate worlds. Finally, we present two case studies of successful—and operational—industrial deployments of ML to counter cyber threats.

This article will hopefully inspire meaningful developments of ML in the cybersecurity domain, laying the foundations for an increased deployment of ML solutions to protect current and future systems.

REFERENCES

[1] 2020. *On Artificial Intelligence—A European Approach to Excellence and Trust.* Technical Report. European Commission.

[2] 2021. Darktrace Industrial Uses Machine Learning to Identify Cyber Campaigns Targeting Critical Infrastructure. Retrieved August 2021 from https://www.darktrace.com/en/press/2017/204/.

[3] 2021. Gartner Predicts by 2025 Cyber Attackers Will Have Weaponized Operational Technology Environments to Successfully Harm or Kill Humans. Retrieved August 2021 from https://www.gartner.com/en/newsroom/press-releases/2021-07-21-gartner-predicts-by-2025-cyber-attackers-will-have-we.

[4] 2021. *S&T Artificial Intelligence and Machine Learning Strategic Plan.* Technical Report. U.S. Department of Homeland Security.

[5] Alexander Afanasyev, Priya Mahadevan, Ilya Moiseenko, Ersin Uzun, and Lixia Zhang. 2013. Interest flooding attack and countermeasures in named data networking. In *Proceedings of the IFIP Networking Conference.* IEEE, 1–9.

[6] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Börje Ohlman. 2012. A survey of information-centric networking. *IEEE Commun. Mag.* 50, 7 (2012), 26–36. https://doi.org/10.1109/MCOM.2012.6231276

[7] Muna Al-Hawawreh and Elena Sitnikova. 2019. Leveraging deep learning models for ransomware detection in the industrial Internet of Things environment. In *Proceedings of the IEEE Military Communications and Information Systems Conference.* 1–6.

[8] Mohammed Al-Qizwini, Iman Barjasteh, Hothaifa Al-Qassab, and Hayder Radha. 2017. Deep learning algorithm for autonomous driving using GoogLeNet. In *Proceedings of the IEEE Intelligent Vehicles Symposium.* 89–96.

[9] Areej Alhogail and Afrah Alsabih. 2021. Applying machine learning and natural language processing to detect phishing email. *Comput. Secur.* 110 (2021), 102414.

[10] Kevin Allix, Tegawendé F. Bissyandé, Jacques Klein, and Yves Le Traon. 2016. Androzoo: Collecting millions of android apps for the research community. In *Proceedings of the IEEE/ACM 13th Working Conference on Mining Software Repositories (MSR'16).* IEEE, 468–471.

[11] Mohammad Almseidin, Maen Alzubi, Szilveszter Kovacs, and Mouhammd Alkasassbeh. 2017. Evaluation of machine learning algorithms for intrusion detection system. In *Proceedings of the IEEE 15th International Symposium on Intelligent Systems and Informatics (SISY'17).* IEEE, 000277–000282.

[12] Mohammed Almukaynizi, Eric Nunes, Krishna Dharaiya, Manoj Senguttuvan, Jana Shakarian, and Paulo Shakarian. 2017. Proactive identification of exploits in the wild through vulnerability mentions online. In *Proceedings of the IEEE International Conference on Cyber Conflict US (CyCon US'17)*. Institute of Electrical and Electronics Engineers Inc., 82–88.
[13] Nisreen Alzahrani and Daniyal Alghazzawi. 2019. A review on android ransomware detection using deep learning techniques. In *Proceedings of the ACM International Conference Management of Digital EcoSystems*. 330–335.
[14] Kasun Amarasinghe, Kevin Kenney, and Milos Manic. 2018. Toward explainable deep neural network based anomaly detection. In *Proceedings of the IEEE International Conference Human System Interaction*. 311–317.
[15] Eslam Amer and Ivan Zelinka. 2020. A dynamic Windows malware detection and prediction method based on contextual understanding of API call sequence. *Comput. Secur.* 92 (2020), 101760.
[16] Abderrahmen Amich and Birhanu Eshete. 2021. Explanation-guided diagnosis of machine learning evasion attacks. *Proceedings of the ACM International Conference on Availability, Reliability and Security Conference*.
[17] Hyrum S. Anderson, Jonathan Woodbridge, and Bobby Filar. 2016. DeepDGA: Adversarially-tuned domain generation and detection. In *Proceedings of the ACM Workshop on Artificial Intelligence and Security*. 13–21.
[18] Giuseppina Andresini, Feargus Pendlebury, Fabio Pierazzi, Corrado Loglisci, Annalisa Appice, and Lorenzo Cavallaro. 2021. INSOMNIA: Towards concept-drift robustness in network intrusion detection. In *Proceedings of the ACM CCS Workshop on Artificial Intelligence and Security*.
[19] Giovanni Apruzzese, Mauro Andreolini, Michele Colajanni, and Mirco Marchetti. 2020. Hardening random forest cyber detectors against adversarial attacks. *IEEE Trans. Emerg. Top. Comput. Intell.* 4, 4 (2020), 427–439.
[20] Giovanni Apruzzese, Mauro Andreolini, Luca Ferretti, Mirco Marchetti, and Michele Colajanni. 2021. Modeling realistic adversarial attacks against network intrusion detection systems. *ACM Digit. Threats: Res. Pract.* (2021).
[21] G. Apruzzese, M. Andreolini, M. Marchetti, A. Venturi, and M. Colajanni. 2020. Deep reinforcement adversarial learning against botnet evasion attacks. *IEEE Trans. Netw. Serv. Manage.* (2020).
[22] Giovanni Apruzzese and Michele Colajanni. 2018. Evading botnet detectors based on flows and random forest with adversarial samples. In *Proceedings of the IEEE International Symposium on Network Computing and Applications*. 1–8.
[23] Giovanni Apruzzese, Michele Colajanni, Luca Ferretti, Alessandro Guido, and Mirco Marchetti. 2018. On the effectiveness of machine and deep learning for cybersecurity. In *Proceedings of the IEEE International Conference on Cyber Conflicts*. 371–390.
[24] Giovanni Apruzzese, Michele Colajanni, Luca Ferretti, and Mirco Marchetti. 2019. Addressing adversarial attacks against security systems based on machine learning. In *Proceedings of the IEEE International Conference on Cyber Conflicts*. 1–18.
[25] Giovanni Apruzzese, Michele Colajanni, and Mirco Marchetti. 2019. Evaluating the effectiveness of adversarial attacks against botnet detectors. In *Proceedings of the IEEE 18th International Symposium on Network Computing and Applications (NCA'19)*. IEEE, 1–8.
[26] Giovanni Apruzzese, Mirco Marchetti, Michele Colajanni, Gabriele Gambigliani Zoccoli, and Alessandro Guido. 2017. Identifying malicious hosts involved in periodic communications. In *Proceedings of the IEEE International Symposium on Network Computing Applications*. 1–8.
[27] Giovanni Apruzzese, Luca Pajola, and Mauro Conti. 2022. The cross-evaluation of machine learning-based network intrusion detection systems. *IEEE Trans. Netw. Serv. Manage.* (2022).
[28] Giovanni Apruzzese, Fabio Pierazzi, Michele Colajanni, and Mirco Marchetti. 2017. Detection and threat prioritization of pivoting attacks in large networks. *IEEE Trans. Emerg. Top. Comput.* 8, 2 (2017), 404–415.
[29] Giovanni Apruzzese, Aliya Tastemirova, and Pavel Laskov. 2022. SoK: The impact of unlabelled data in cyberthreat detection. In *Proceedings of the IEEE European Symposium on Security Privacy*.
[30] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. 2021. Dos and don'ts of machine learning in computer security. In *Proceedings of the USENIX Security Symposium*.
[31] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. 2014. Drebin: Effective and explainable detection of android malware in your pocket. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'14)*, Vol. 14. 23–26.
[32] Mehdi Babagoli, Mohammad Pourmahmood Aghababa, and Vahid Solouk. 2019. Heuristic nonlinear regression strategy for detecting phishing websites. *Soft Comput.* 23, 12 (2019), 4315–4327.
[33] Ram Basnet. 2014. Learning to detect phishing URLs. *Int. J. Res. Eng. Technol.* 3 (2014), 11–24.
[34] Manjula C. Belavagi and Balachandra Muniyal. 2016. Performance evaluation of supervised machine learning algorithms for intrusion detection. *Proc. Comput. Sci.* 89 (2016), 117–123.
[35] Jacopo Bellasio and Erik Silfversten. 2020. The impact of new and emerging technologies on the cyber threat landscape and their implications for NATO. In *Cyber Threats and NATO 2030: Horizon Scanning and Analysis*, 88.
[36] Daniel S. Berman, Anna L. Buczak, Jeffrey S. Chavis, and Cherita L. Corbett. 2019. A survey of deep learning methods for cyber security. *Information* 10, 4 (2019), 122.
[37] Gustavo de Carvalho Bertoli, Lourenço Alves Pereira Junior, Filipe Alves Neto Verri, Aldri Luiz dos Santos, and Osamu Saotome. 2021. Bridging the gap to real-world for network intrusion detection systems with data-centric approach. *Proceedings of the Neural Information Processing Systems*.

[38] Battista Biggio, Igino Corona, Zhi-Min He, Patrick P. K. Chan, Giorgio Giacinto, Daniel S. Yeung, and Fabio Roli. 2015. One-and-a-half-class multiple classifier systems for secure learning against evasion attacks at test time. In *Proceedings of the International Workshop on Multiple Classifier Systems*. Springer, 168–180.

[39] Battista Biggio and Fabio Roli. 2018. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recogn.* 84 (2018), 317–331.

[40] Leyla Bilge, Engin Kirda, Christopher Kruegel, and Marco Balduzzi. 2011. EXPOSURE: Finding malicious domains using passive DNS analysis. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'11)*. 1–17.

[41] Adel Binbusayyis and Thavavel Vaiyapuri. 2019. Identifying and benchmarking key features for cyber intrusion detection: An ensemble approach. *IEEE Access* 7 (2019), 106495–106513.

[42] Franziska Boenisch, Verena Battis, Nicolas Buchmann, and Maija Poikela. 2021. "I never thought about securing my machine learning systems": A study of security and privacy awareness of machine learning practitioners. In *Mensch und Computer 2021*. 520–546.

[43] Atul Bohara, Mohammad A. Noureddine, Ahmed Fawaz, and William H. Sanders. 2017. An unsupervised multi-detector approach for identifying malicious lateral movement. In *Proceedings of the IEEE 36th Symposium on Reliable Distributed Systems (SRDS'17)*. IEEE, 224–233.

[44] Gianluca Bontempi, Souhaib Ben Taieb, and Yann-Aël Le Borgne. 2012. Machine learning strategies for time series forecasting. In *European Business Intelligence Summer School*. 62–77.

[45] Emilie Bout, Valeria Loscri, and Antoine Gallais. 2021. How machine learning changes the nature of cyberattacks on IoT networks: A survey. *IEEE Commun. Surv. Tutor.* (2021).

[46] Anna L. Buczak and Erhan Guven. 2015. A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Commun. Surv. Tutor.* 18, 2 (2015), 1153–1176.

[47] Elie Bursztein, Matthieu Martin, and John Mitchell. 2011. Text-based CAPTCHA strengths and weaknesses. In *Proceedings of the ACM Computer and Communications Security Conference*. 125–138.

[48] Nicholas Carlini and David Wagner. 2016. Defensive distillation is not robust to adversarial examples. arXiv:1607.04311. Retrieved from https://arxiv.org/abs/1607.04311.

[49] Tanmoy Chakraborty, Fabio Pierazzi, and V. S. Subrahmanian. 2017. EC2: Ensemble clustering and classification for predicting android malware families. *IEEE Trans. Depend. Sec. Comput.* (2017).

[50] Sujita Chaudhary, Austin O'Brien, and Shengjie Xu. 2020. Automated post-breach penetration testing through reinforcement learning. In *Proceedings of the IEEE Conference on Communications and Network Security (CNS'20)*. 1–2.

[51] Haipeng Chen, Jing Liu, Rui Liu, Noseong Park, and V. S. Subrahmanian. 2019. VASE: A twitter-based vulnerability analysis and score engine. In *Proceedings of the IEEE International Conference on Data Mining (ICDM'19)*. IEEE, 976–981.

[52] Li Chen, Salmin Sultana, and Ravi Sahita. 2018. Henet: A deep learning approach on intel processor trace for effective exploit detection. In *Proceedings of the IEEE Security and Privacy Workshops*. 109–115.

[53] Howard Chivers, John A. Clark, Philip Nobles, Siraj A. Shaikh, and Hao Chen. 2013. Knowing who to watch: Identifying attackers whose actions are hidden within false alarms and background noise. *Inf. Syst. Front.* 15, 1 (2013), 17–34.

[54] Zheng Leong Chua, Shiqi Shen, Prateek Saxena, and Zhenkai Liang. 2017. Neural nets can learn function type signatures from binaries. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security'17)*. 99–116.

[55] Igino Corona, Battista Biggio, Matteo Contini, Luca Piras, Roberto Corda, Mauro Mereu, Guido Mureddu, Davide Ariu, and Fabio Roli. 2017. Deltaphish: Detecting phishing webpages in compromised websites. In *European Symposium on Research in Computer Security*. Springer, 370–388.

[56] Andrea Corsini, Shanchieh Yang, and Giovanni Apruzzese. 2021. On the evaluation of sequential machine learning for network intrusion detection. In *Proceedings of the International Conference Availability, Reliability, Security*.

[57] Ittai Dayan, Holger R. Roth, Aoxiao Zhong, Ahmed Harouni, Amilcare Gentili, Anas Z. Abidin, Andrew Liu, Anthony Beardsworth Costa, Bradford J. Wood, Chien-Sung Tsai, et al. 2021. Federated learning for predicting clinical outcomes in patients with COVID-19. *Nat. Med.* (2021), 1–9.

[58] Mostafa Dehghani, Yi Tay, Alexey A. Gritsenko, Zhe Zhao, Neil Houlsby, Fernando Diaz, Donald Metzler, and Oriol Vinyals. 2021. The benchmark lottery. In *Proceedings of the Conference and Workshop on Neural Information Processing Systems (NeurIPS'21)*.

[59] Ambra Demontis, Marco Melis, Battista Biggio, Davide Maiorca, Daniel Arp, Konrad Rieck, Igino Corona, Giorgio Giacinto, and Fabio Roli. 2017. Yes, machine learning can be more secure! A case study on android malware detection. *IEEE Trans. Depend. Sec. Comput.* (2017).

[60] Ambra Demontis, Marco Melis, Maura Pintor, Matthew Jagielski, Battista Biggio, Alina Oprea, Cristina Nita-Rotaru, and Fabio Roli. 2019. Why do adversarial attacks transfer? Explaining transferability of evasion and poisoning attacks. In *Proceedings of the USENIX Security Symposium*. 321–338.

[61] Melvin Diale, Turgay Celik, and Christiaan Van Der Walt. 2019. Unsupervised feature learning for spam email filtering. *Comput. Electr. Eng.* 74 (2019), 89–104.

[62] Luis Dias, Simão Valente, and Miguel Correia. 2020. Go with the flow: Clustering dynamically-defined netflow features for network intrusion detection with DynIDS. In *Proceedings of the IEEE 19th International Symposium on Network Computing and Applications (NCA'20)*. IEEE, 1–10.

[63] Jesús E. Díaz-Verdejo, Antonio Estepa, Rafael Estepa, German Madinabeitia, and Fco Javier Muñoz-Calle. 2020. A methodology for conducting efficient sanitization of HTTP training datasets. *Fut. Gener. Comput. Syst.* 109 (2020), 67–82.
[64] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 1285–1298.
[65] Murat Dundar, Balaji Krishnapuram, Jinbo Bi, and R. Bharat Rao. 2007. Learning classifiers when the training data is not IID. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI'07)*. 756–61.
[66] Gints Engelen, Vera Rimmer, and Wouter Joosen. 2021. Troubleshooting an intrusion detection dataset: The CICIDS2017 case study. In *Proceedings of the IEEE Security and Privacy Workshop*. 7–12.
[67] Yong Fang, Cheng Zhang, Cheng Huang, Liang Liu, and Yue Yang. 2019. Phishing email detection using improved RCNN model with multilevel vectors and attention mechanism. *IEEE Access* 7 (2019), 56329–56340.
[68] Cheng Feng, Tingting Li, and Deeph Chana. 2017. Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks. In *Proceedings of the 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'17)*. IEEE, 261–272.
[69] Simone Fischer-Hübner, Cristina Alcaraz, Afonso Ferreira, Carmen Fernandez-Gago, Javier Lopez, Evangelos Markatos, Lejla Islami, and Mahdi Akil. 2021. Stakeholder perspectives and requirements on cybersecurity in Europe. *J. Inf. Secur. Appl.* 61 (2021), 102916.
[70] Tushaar Gangavarapu, C. D. Jaidhar, and Bhabesh Chanduka. 2020. Applicability of machine learning in spam and phishing email filtering: Review and approaches. *Artif. Intell. Rev.* (2020), 1–63.
[71] Joseph Gardiner and Shishir Nagaraja. 2016. On the security of machine learning in malware c&c detection: A survey. *ACM Comput. Surv.* 49, 3 (2016), 59.
[72] José Tomás Martínez Garre, Manuel Gil Pérez, and Antonio Ruiz-Martínez. 2021. A novel machine learning-based approach for the detection of SSH botnet infection. *Fut. Gener. Comput. Syst.* 115 (2021), 387–396.
[73] Hugo Gascon, Steffen Ullrich, Benjamin Stritter, and Konrad Rieck. 2018. Reading between the lines: Content-agnostic detection of spear-phishing emails. In *Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 69–91.
[74] Mohamed C. Ghanem and Thomas M. Chen. 2018. Reinforcement learning for intelligent penetration testing. In *Proceedings of the IEEE 2nd World Conference on Smart Trends in Systems, Security and Sustainability*. 185–192.
[75] Arnaldo Gouveia and Miguel Correia. 2020. Towards quantum-enhanced machine learning for network intrusion detection. In *Proceedings of the IEEE 19th International Symposium on Network Computing and Applications (NCA'20)*. 1–8.
[76] Kathrin Grosse, Nicolas Papernot, Praveen Manoharan, Michael Backes, and Patrick McDaniel. 2017. Adversarial examples for malware detection. In *Proceedings of the European Symposium on Research in Computer Security*. Springer, 62–79.
[77] Malay Haldar, Mustafa Abdool, Prashant Ramanathan, Tao Xu, Shulin Yang, Huizhong Duan, Qing Zhang, Nick Barrow-Williams, Bradley C. Turnbull, Brendan M. Collins, et al. 2019. Applying deep learning to AirBnB search. In *Proceedings of the ACM SIGKDD International Conference Knowledge Discovery and Data Mining*. 1927–1935.
[78] Richard Harang and Ethan M. Rudd. 2020. SOREL-20M: A large scale benchmark dataset for malicious PE detection. arXiv:2012.07634. Retrieved from https://arxiv.org/abs/2012.07634.
[79] Martin Horák, Václav Stupka, and Martin Husák. 2019. GDPR compliance in cybersecurity software: A case study of DPIA in information sharing platform. In *Proceedings of the ACM International Conference Availability, Reliability and Security*. 1–8.
[80] Xin Hu, Kang G. Shin, Sandeep Bhatkar, and Kent Griffin. 2013. Mutantx-s: Scalable malware clustering based on static features. In *Proceedings of the USENIX Annual Technical Conference*. 187–198.
[81] Yupeng Hu, Wenxin Kuang, Zheng Qin, Kenli Li, Jiliang Zhang, Yansong Gao, Wenjia Li, and Keqin Li. 2021. Artificial intelligence security: Threats and countermeasures. *ACM Comput. Surv.* 55, 1 (2021), 1–36.
[82] Martin Husák, Tomáš Jirsík, and Shanchieh Jay Yang. 2020. SoK: Contemporary issues and challenges to enable cyber situational awareness for network security. In *Proceedings of the International Conference on Availability, Reliability and Security*. 1–10.
[83] Mohammad S. Jalali, Michael Siegel, and Stuart Madnick. 2019. Decision-making and biases in cybersecurity capability development: Evidence from a simulation game experiment. *J. Strateg. Inf. Syst.* 28, 1 (2019), 66–82.
[84] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. 2016. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (Formerly BIONETICS)*. 21–26.
[85] Michael I. Jordan and Tom M. Mitchell. 2015. Machine learning: Trends, perspectives, and prospects. *Science* 349, 6245 (2015), 255–260.
[86] Roberto Jordaney, Kumar Sharad, Santanu K. Dash, Zhi Wang, Davide Papini, Ilia Nouretdinov, and Lorenzo Cavallaro. 2017. Transcend: Detecting concept drift in malware classification models. In *Proceedings of the USENIX Security Symposium*. 625–642.
[87] Mahmoud Kalash, Mrigank Rochan, Noman Mohammed, Neil D. B. Bruce, Yang Wang, and Farkhund Iqbal. 2018. Malware classification with deep convolutional neural networks. In *Proceedings of the 9th IFIP International Conference on New Technologies, Mobility and Security (NTMS'18)*. IEEE, 1–5.
[88] Chanhyun Kang, Noseong Park, B. Aditya Prakash, Edoardo Serra, and V. S. Subrahmanian. 2016. Ensemble models for data-driven prediction of malware infections. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining*. 583–592.

[89] Asif Karim, Sami Azam, Bharanidharan Shanmugam, Krishnan Kannoorpatti, and Mamoun Alazab. 2019. A comprehensive survey for intelligent spam email detection. *IEEE Access* 7 (2019), 168261–168295.

[90] Houssain Kettani and Polly Wainwright. 2019. On the top threats to cyber systems. In *Proceedings of the IEEE 2nd International Conference on Information and Computer Technologies (ICICT'19)*. IEEE, 175–179.

[91] Ahsan Al Zaki Khan. 2019. Misuse intrusion detection using machine learning for gas pipeline scada networks. In *Proceedings of the International Conference Security and Management*. 84–90.

[92] Platon Kotzias, Juan Caballero, and Leyla Bilge. 2021. How did that get in my phone? Unwanted app distribution on android devices. In *Proceedings of the IEEE Symposium on Security and Privacy*. 53–69.

[93] Nir Kshetri. 2021. Economics of artificial intelligence in cybersecurity. *IEEE IT Profess.* 23, 5 (2021), 73–77.

[94] Gunupudi Rajesh Kumar, Nimmala Mangathayaru, and Gugulothu Narsimha. 2016. An approach for intrusion detection using fuzzy feature clustering. In *Proceedings of the IEEE International Conference on Engineering & MIS (ICEMIS'16)*. 1–8.

[95] Ram Shankar Siva Kumar, Magnus Nyström, John Lambert, Andrew Marshall, Mario Goertzel, Andi Comissoneru, Matt Swann, and Sharon Xia. 2020. Adversarial machine learning-industry perspectives. In *Proceedings of the IEEE Security and Privacy Workshops*. 69–75.

[96] Eric Lancaster, Tanmoy Chakraborty, and V. S. Subrahmanian. 2018. MALT^P: Parallel prediction of malicious tweets. *IEEE T. Comput. Soc. Syst.* 5, 4 (2018), 1096–1108.

[97] Lastline. 2020. *Using AI to Detect and Contain Cyberthreats*. Technical Report.

[98] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521, 7553 (2015), 436–444.

[99] Jhen-Hao Li and Sheng-De Wang. 2017. PhishBox: An approach for phishing validation and detection. In *Proceedings of the IEEE DASC/PiCom/DataCom/CyberSciTech Conference*. 557–564.

[100] Yuping Li, Jiyong Jang, Xin Hu, and Xinming Ou. 2017. Android malware clustering through malicious payload mining. In *Proceedings of the International Symposium on Research in Attacks, Intrusions, and Defenses*. Springer, 192–214.

[101] Bin Liang, Miaoqiang Su, Wei You, Wenchang Shi, and Gang Yang. 2016. Cracking classifiers for evasion: A case study on the google's phishing pages filter. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 345–356.

[102] Hongyu Liu and Bo Lang. 2019. Machine learning and deep learning methods for intrusion detection systems: A survey. *Appl. Sci.* 9, 20 (2019), 4396.

[103] Zhen Liu, Ruoyu Wang, Nathalie Japkowicz, Deyu Tang, Wenbin Zhang, and Jie Zhao. 2021. Research on unsupervised feature learning for Android malware detection based on Restricted Boltzmann Machines. *Fut. Gener. Comput. Syst.* 120 (2021), 91–108.

[104] Siti-Farhana Lokman, Abu Talib Othman, and Muhammad-Husaini Abu-Bakar. 2019. Intrusion detection system for automotive Controller Area Network (CAN) bus system: A review. *EURASIP J. Wireless Commun. Netw.* 2019, 1 (2019), 1–17.

[105] Pierangelo Lombardo, Salvatore Saeli, Federica Bisio, Davide Bernardi, and Danilo Massa. 2018. Fast flux service network detection via data mining on passive DNS traffic. In *Proceedings of the International Conference on Information Security*. Springer, 463–480.

[106] Dimitris Margaritis. 2020. *Artificial Intelligence Cybersecurity Challenges*. Technical Report. European Union Agency for Cybersecurity.

[107] Daniel L. Marino, Chathurika S. Wickramasinghe, and Milos Manic. 2018. An adversarial approach for explainable ai in intrusion detection systems. In *Proceedings of the IEEE Conference of the Industrial Electronics Society*. 3237–3243.

[108] Nuno Martins, José Magalhães Cruz, Tiago Cruz, and Pedro Henriques Abreu. 2020. Adversarial machine learning applied to intrusion and malware scenarios: A systematic review. *IEEE Access* 8 (2020), 35403–35419.

[109] Lennart Maschmeyer, Ronald J. Deibert, and Jon R. Lindsay. 2021. A tale of two cybers-how threat reporting by cybersecurity firms systematically underrepresents threats to civil society. *J. Inf. Technol. Polit.* 18, 1 (2021), 1–20.

[110] Steven McElwee, Jeffrey Heaton, James Fraley, and James Cannady. 2017. Deep learning for prioritizing and responding to intrusion detection alerts. In *Proceedings of the IEEE Military Communications Conference*. 1–5.

[111] Dean Richard McKinnel, Tooska Dargahi, Ali Dehghantanha, and Kim-Kwang Raymond Choo. 2019. A systematic literature review and meta-analysis on artificial intelligence in penetration testing and vulnerability assessment. *Comput. Electr. Eng.* 75 (2019), 175–188.

[112] Brad Miller, Alex Kantchelian, Michael Carl Tschantz, Sadia Afroz, Rekha Bachwani, Riyaz Faizullabhoy, Ling Huang, Vaishaal Shankar, Tony Wu, George Yiu, et al. 2016. Reviewer integration and performance measurement for malware detection. In *Proceedings of the International Conference on Detection of Intrusions and Malware & Vulnerability Assessment (DIMVA'16)*. 122–141.

[113] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. Kitsune: An ensemble of autoencoders for online network intrusion detection. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'18)*, Vol. 5. 2.

[114] Manuel Eugenio Morocho-Cayamcela, Haeyoung Lee, and Wansu Lim. 2019. Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions. *IEEE Access* 7 (2019), 137184–137206.

[115] Nour Moustafa and Jill Slay. 2015. UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In *Proceedings of the Military Communications and Information Systems Conference (MilCIS'15)*. IEEE, 1–6.

[116] Azqa Nadeem, Sicco Verwer, Stephen Moskal, and Shanchieh Jay Yang. 2021. Alert-driven attack graph generation using S-PDFA. *IEEE Trans. Depend. Sec. Comput.* (2021).

[117] Antonio Nappa, Zhaoyan Xu, M. Zubair Rafique, Juan Caballero, and Guofei Gu. 2014. Cyberprobe: Towards internet-scale active detection of malicious servers. In *Proceedings of the Network and Distributed System Security Symposium (NDSS'14)*. 1–15.

[118] Tan Nguyen, Hoang-Long Mai, Guillaume Doyen, Rémi Cogranne, Wissam Mallouli, Edgardo Montes De Oca, and Olivier Festor. 2018. A security monitoring plane for named data networking deployment. *IEEE Commun. Mag.* 56, 11 (2018), 88–94.

[119] Tan Nguyen, Xavier Marchal, Guillaume Doyen, Thibault Cholez, and Rémi Cogranne. 2017. Content poisoning in named data networking: Comprehensive characterization of real deployment. In *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM'17)*. IEEE, 72–80.

[120] Tan N. Nguyen, Xavier Marchal, Guillaume Doyen, Thibault Cholez, and Rémi Cogranne. 2017. Content poisoning in named data networking: Comprehensive characterization of real deployment. In *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM'17)*. IEEE, 72–80. https://doi.org/10.23919/INM.2017.7987266

[121] Thanh Thi Nguyen and Vijay Janapa Reddi. 2021. Deep reinforcement learning for cyber security. *IEEE Trans. Neur. Netw. Learn. Syst.* (2021), 1–17.

[122] Amirreza Niakanlahiji, Bei-Tseng Chu, and Ehab Al-Shaer. 2018. PhishMon: A machine learning framework for detecting phishing webpages. In *Proceedings of the IEEE International Conference Intelligent Security Informatics*. 220–225.

[123] Beny Nugraha, Anshitha Nambiar, and Thomas Bauschert. 2020. Performance evaluation of botnet detection using deep learning techniques. In *Proceedings of the IEEE International Conference Network of the Future*. 141–149.

[124] Livinus Obiora Nweke and Stephen Wolthusen. 2020. Legal issues related to cyber threat information sharing among private entities for critical infrastructure protection. In *Proceedings of the IEEE International Conference on Cyber Conflict (CyCon'20)*.

[125] Ahmet Okutan and Shanchieh Jay Yang. 2019. ASSERT: Attack synthesis and separation with entropy redistribution towards predictive cyber defense. *Cybersecurity* 2, 1 (2019), 1–18.

[126] Ahmet Okutan, Shanchieh Jay Yang, and Katie McConky. 2021. Cyberattack Forecasting Using Predictive Information. (Jan. 21 2021). US Patent App. 16/898,618.

[127] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael Wellman. 2018. SoK: Security and privacy in machine learning. In *Proceedings of the IEEE European Symposium on Security and Privacy*. 399–414.

[128] Nicolas Papernot, Patrick McDaniel, Xi Wu, Somesh Jha, and Ananthram Swami. 2016. Distillation as a defense to adversarial perturbations against deep neural networks. In *Proceedings of the IEEE Symposium on Security and Privacy (SP'16)*. IEEE, 582–597.

[129] Sergio Pastrana, Daniel R. Thomas, Alice Hutchings, and Richard Clayton. 2018. Crimebb: Enabling cybercrime research on underground forums at scale. In *Proceedings of the World Wide Web Conference*. International World Wide Web Conferences Steering Committee, 1845–1854.

[130] Feargus Pendlebury, Fabio Pierazzi, Roberto Jordaney, Johannes Kinder, and Lorenzo Cavallaro. 2019. {TESSERACT}: Eliminating experimental bias in malware classification across space and time. In *Proceedings of the 28th USENIX Security Symposium (USENIX Security'19)*. 729–746.

[131] Roberto Perdisci and Wenke Lee. 2018. Method and System for Detecting Malicious and/or Botnet-related Domain Names. (July 17 2018). US Patent 10,027,688.

[132] Fabio Pierazzi, Giovanni Apruzzese, Michele Colajanni, Alessandro Guido, and Mirco Marchetti. 2017. Scalable architecture for online prioritisation of cyber threats. In *Proceedings of the IEEE International Conference on Cyber Conflicts*. 1–18.

[133] Fabio Pierazzi, Feargus Pendlebury, Jacopo Cortellazzi, and Lorenzo Cavallaro. 2020. Intriguing properties of adversarial ml attacks in the problem space. In *Proceedings of the IEEE Symposium on Security and Privacy*. 1332–1349.

[134] Camila Pontes, Manuela Souza, João Gondim, Matt Bishop, and Marcelo Marotta. 2021. A new method for flow-based network intrusion detection using the inverse Potts model. *IEEE Trans. Netw. Serv. Manage.* (2021).

[135] Rebecca S. Portnoff, Sadia Afroz, Greg Durrett, Jonathan K. Kummerfeld, Taylor Berg-Kirkpatrick, Damon McCoy, Kirill Levchenko, and Vern Paxson. 2017. Tools for automated analysis of cybercriminal markets. In *Proceedings of the 26th International Conference on World Wide Web*. 657–666.

[136] Artur Potiguara Carvalho, Fernanda Potiguara Carvalho, Edna Dias Canedo, and Pedro Henrique Potiguara Carvalho. 2020. Big data, anonymisation and governance to personal data protection. In *Proceedings of the International Conference on Digital Government Research*. 185–195.

[137] Petar Radanliev, David De Roure, Rob Walton, Max Van Kleek, Rafael Mantilla Montalvo, Omar Santos, Peter Burnap, Eirini Anthi, et al. 2020. Artificial intelligence and machine learning in dynamic cyber risk analytics at the edge. *SN Appl. Sci.* 2, 11 (2020), 1–8.

[138] Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Certified defenses against adversarial examples. In *Proceedings of the International Conference on Learning Representations*.

[139] Vignesh Ramanathan, Rui Wang, and Dhruv Mahajan. 2021. PreDet: Large-scale weakly supervised pre-training for detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2865–2875.

[140] Supranamaya Ranjan. 2014. Machine Learning Based Botnet Detection Using Real-time Extracted Traffic Features. (March 25 2014). US Patent 8,682,812.

[141] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, and Pavel Laskov. 2008. Learning and classification of malware behavior. In *Proceedings of the International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*. Springer, 108–125.

[142] Markus Ring, Sarah Wunderlich, Deniz Scheuring, Dieter Landes, and Andreas Hotho. 2019. A survey of network-based intrusion detection data sets. *Comput. Secur.* 86 (2019), 147–167.
[143] Farhan Sadique, Sui Cheung, Iman Vakilinia, Shahriar Badsha, and Shamik Sengupta. 2018. Automated structured threat information eXpression (STIX) document generation with privacy preservation. In *Proceedings of the IEEE Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCon'18)*. 847–853.
[144] Ozgur Koray Sahingoz, Ebubekir Buber, Onder Demir, and Banu Diri. 2019. Machine learning based phishing detection from URLs. *Expert Syst. Appl.* 117 (2019), 345–357.
[145] Wojciech Samek, Thomas Wiegand, and Klaus-Robert Müller. 2017. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. arXiv:1708.08296. Retrieved from https://arxiv.org/abs/1708.08296.
[146] Anna Sapienza, Alessandro Bessi, Saranya Damodaran, Paulo Shakarian, Kristina Lerman, and Emilio Ferrara. 2017. Early warnings of cyber threats in online discussions. In *Proceedings of the IEEE International Conference on Data Mining Workshops (ICDMW'17)*. IEEE, 667–674.
[147] Iman Sharafaldin, Arash Habibi Lashkari, and Ali A. Ghorbani. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the International Conference on Information Systems Security and Privacy (ICISSP'18)*. 108–116.
[148] Salvatore Signorello, Samuel Marchal, Jerome Francois, Olivier Festor, and Radu State. 2017. Advanced interest flooding attacks in named-data networking. In *Proceedings of the IEEE 16th International Symposium on Network Computing and Applications (NCA'17)*.
[149] Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 305–316.
[150] Qiyang Song, Jiahao Cao, Kun Sun, Qi Li, and Ke Xu. 2021. Try before you buy: Privacy-preserving data evaluation on cloud-based machine learning data marketplace. In *Proceedings of the ACM Annual Computer Security Applications Conference*. 260–272.
[151] Paolo Spagnolettia and Andrea Salvia. 2020. Digital systems in high-reliability organizations: Balancing mindfulness and mindlessness. In *Proceedings of the International Workshop Socio-Technical Perspective in Information Systems Development.*
[152] IEEE Spectrum. 2022. *Andrew Ng: Unbiggen AI.* Technical Report.
[153] Nedim Šrndic and Pavel Laskov. 2013. Detection of malicious pdf files based on hierarchical document structure. In *Proceedings of the 20th Annual Network & Distributed System Security Symposium*. 1–16.
[154] Nedim Šrndic and Pavel Laskov. 2014. Practical evasion of a learning-based classifier: A case study. In *Proceedings of the IEEE Symposium on Security and Privacy*. IEEE, 197–211.
[155] Matija Stevanovic and Jens Myrup Pedersen. 2014. An efficient flow-based botnet detection using supervised machine learning. In *Proceedings of the International Conference on Computing, Networking and Communications (ICNC'14)*. IEEE, 797–801.
[156] Tongtong Su, Huazhi Sun, Jinqi Zhu, Sheng Wang, and Yabo Li. 2020. BAT: Deep learning methods on network intrusion detection using NSL-KDD dataset. *IEEE Access* 8 (2020), 29575–29585.
[157] Yuan-Hsiang Su, Michael Cheng Yi Cho, and Hsiu-Chuan Huang. 2019. False alert buster: An adaptive approach for NIDS false alert filtering. In *Proceedings of the ACM International Conference on Big Data*. 58–62.
[158] Christopher Sweet, Stephen Moskal, and Shanchieh Jay Yang. 2020. On the variety and veracity of cyber intrusion alerts synthesized by generative adversarial networks. *ACM Trans. Manage. Inf. Syst.* 11, 4 (2020), 1–21.
[159] Ke Tian, Steve T. K. Jan, Hang Hu, Danfeng Yao, and Gang Wang. 2018. Needle in a haystack: Tracking down elite phishing domains in the wild. In *Proceedings of the Internet Measurement Conference*. 429–442.
[160] Daniele Ucci, Leonardo Aniello, and Roberto Baldoni. 2019. Survey of machine learning techniques for malware analysis. *Comput. Secur.* 81 (2019), 123–147.
[161] Solomon Ogbomon Uwagbole, William J. Buchanan, and Lu Fan. 2017. Applied machine learning predictive analytics to SQL injection attack detection and prevention. In *Proceedings of the IFIP/IEEE Symposium on Integrated Network and Service Management (IM'17)*. 1087–1090.
[162] Maneesh Kumar Verma, Shankar Yadav, Bhoopesh Kumar Goyal, Bakshi Rohit Prasad, and Sonali Agarawal. 2019. Phishing website detection using neural network and deep belief network. In *Recent Findings in Intelligent Computing Techniques*. Springer, 293–300.
[163] Rakesh M. Verma, Victor Zeng, and Houtan Faridi. 2019. Data quality for security challenges: Case studies of phishing, malware and intrusion detection datasets. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 2605–2607.
[164] Kristijan Vidović, Ivan Tomičić, Karlo Slovenec, Miljenko Mikuc, and Ivona Brajdić. 2021. Ranking network devices for alarm prioritisation: Intrusion detection case study. In *Proceedings of the IEEE International Conference on Software, Telecommunications and Computer Networks (SoftCOM'21)*. 1–5.
[165] R. Vinayakumar, Mamoun Alazab, Alireza Jolfaei, K. P. Soman, and Prabaharan Poornachandran. 2019. Ransomware triage using deep learning: Twitter as a case study. In *Proceedings of the IEEE Cybersecurity & Cyberforensics Conference*. 67–73.
[166] Ravi Vinayakumar, Mamoun Alazab, K. P. Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. 2019. Deep learning approach for intelligent intrusion detection system. *IEEE Access* 7 (2019), 41525–41550.
[167] Paul Voigt and Axel Von dem Bussche. 2017. The EU general data protection regulation (GDPR). In *A Practical Guide* (1st ed.). Springer International Publishing, Cham, 3152676.

[168] Lise Volkart, Pierrette Bouillon, and Sabrina Girletti. 2018. Statistical vs. neural machine translation: A comparison of mth and deepl at swiss post's language service. In *Proceedings of the 40th Conference Translating and the Computer*. 145–150.
[169] Bachar Wehbi, Edgardo Montes de Oca, and Michel Bourdellès. 2012. Events-based security monitoring using MMT tool. In *Proceedings of the 5th IEEE International Conference on Software Testing, Verification and Validation (ICST'12)*, Giuliano Antoniol, Antonia Bertolino, and Yvan Labiche (Eds.). IEEE Computer Society, 860–863. https://doi.org/10.1109/ICST.2012.188
[170] Charles Wheelus, Elias Bou-Harb, and Xingquan Zhu. 2018. Tackling class imbalance in cyber security datasets. In *Proceedings of the IEEE International Conference Information Reuse and Integration*. 229–232.
[171] Laurie Williams, Gary McGraw, and Sammy Migues. 2018. Engineering security vulnerability prevention, detection, and response. *IEEE Softw.* 35, 5 (2018), 76–80.
[172] Tingmin Wu, Shigang Liu, Jun Zhang, and Yang Xiang. 2017. Twitter spam detection based on deep learning. In *Proceedings of the Australasian Computer Science Week Multiconference*. 1–8.
[173] Teng Xu, Gerard Goossen, Huseyin Kerem Cevahir, Sara Khodeir, Yingyezhe Jin, Frank Li, Shawn Shan, Sagar Patel, David Freeman, and Paul Pearce. 2021. Deep entity classification: Abusive account detection for online social networks. In *Proceedings of the USENIX Security Symposium*.
[174] Zhiwei Xu, Bo Chen, Ninghan Wang, Yujun Zhang, and Zhongcheng Li. 2015. ELDA: Towards efficient and lightweight detection of cache pollution attacks in NDN. In *Proceedings of the IEEE 40th Conference on Local Computer Networks (LCN'15)*. IEEE, 82–90.
[175] Carter Yagemann, Matthew Pruett, Simon P. Chung, Kennon Bittick, Brendan Saltaformaggio, and Wenke Lee. 2021. {ARCUS}: Symbolic root cause analysis of exploits in production systems. In *Proceedings of the USENIX Security Symposium*.
[176] Aviv Yehezkel, Eyal Elyashiv, and Ol Soffer. 2021. Network anomaly detection using transfer learning based on auto-encoders loss normalization. In *Proceedings of the ACM Computer and Commununications Security Workshop*.
[177] Ting-Fang Yen, Victor Heorhiadi, Alina Oprea, Michael K. Reiter, and Ari Juels. 2014. An epidemiological study of malware encounters in a large enterprise. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 1117–1130.
[178] Ting-Fang Yen, Alina Oprea, Kaan Onarlioglu, Todd Leetham, William Robertson, Ari Juels, and Engin Kirda. 2013. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proceedings of the 29th Annual Computer Security Applications Conference*. ACM, 199–208.
[179] Jiao Yin, MingJian Tang, Jinli Cao, and Hua Wang. 2020. Apply transfer learning to cybersecurity: Predicting exploitability of vulnerabilities by description. *Knowl.-Bas. Syst.* 210 (2020), 106529.
[180] Chika Yinka-Banjo and Ogban-Asuquo Ugot. 2020. A review of generative adversarial networks and its application in cybersecurity. *Artif. Intell. Rev.* 53, 3 (2020), 1721–1736.
[181] Lixia Zhang, Alexander Afanasyev, Jeffrey Burke, Van Jacobson, Patrick Crowley, Christos Papadopoulos, Lan Wang, Beichuan Zhang, et al. 2014. Named data networking. *ACM SIGCOMM Comput. Commun. Rev.* 44, 3 (2014), 66–73.
[182] Lixia Zhang, Alexander Afanasyev, Jeff Burke, Van Jacobson, KC Claffy, Patrick Crowley, Christos Papadopoulos, Lan Wang, and Beichuan Zhang. 2014. Named data networking. *Comput. Commun. Rev.* 44, 3 (2014), 66–73. https://doi.org/10.1145/2656877.2656887
[183] Xiaohan Zhang, Yuan Zhang, Ming Zhong, Daizong Ding, Yinzhi Cao, Yukun Zhang, Mi Zhang, and Min Yang. 2020. Enhancing state-of-the-art classifiers with API semantics to detect evolved android malware. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. 757–770.
[184] Yong Zhang, Jie Niu, Guojian He, Lin Zhu, and Da Guo. 2021. Network intrusion detection based on active semi-supervised learning. In *Proceedings of the IEEE International Conference on Dependable Systems and Networks*. 129–135.
[185] Weiwei Zhuang, Qingshan Jiang, and Tengke Xiong. 2012. An intelligent anti-phishing strategy model for phishing website detection. In *Proceedings of the 32nd International Conference on Distributed Computing Systems Workshops*. IEEE, 51–56.

Received 5 December 2021; revised 25 March 2022; accepted 17 June 2022