Understanding the Process of Data Labeling in Cybersecurity

Tobias Braun
University of Liechtenstein
tobias.braun@uni.li

Irdin Pekaric
University of Liechtenstein
irdin.pekaric@uni.li

Giovanni Apruzzese
University of Liechtenstein
giovanni.apruzzese@uni.li

ABSTRACT

Many domains now leverage the benefits of Machine Learning (ML), which promises solutions that can autonomously learn to solve complex tasks by training over some data. Unfortunately, in cyberthreat detection, high-quality data is hard to come by. Moreover, for some specific applications of ML, such data must be labeled by human operators. Many works "assume" that labeling is tough/challenging/costly in cyberthreat detection, thereby proposing solutions to address such a hurdle. Yet, we found no work that specifically addresses the process of labeling *from the viewpoint of ML security practitioners*. This is a problem: to this date, it is still mostly unknown how labeling is done in practice—thereby preventing one from pinpointing "what is needed" in the real world.

In this paper, we take the first step to build a bridge between academic research and security practice in the context of data labeling. First, we reach out to five subject matter experts and carry out open interviews to identify pain points in their labeling routines. Then, by using our findings as a scaffold, we conduct a user study with 13 practitioners from large security companies, and ask detailed questions on subjects such as active learning, costs of labeling, and revision of labels. Finally, we perform proof-of-concept experiments addressing labeling-related aspects in cyberthreat detection that are sometimes overlooked in research. Altogether, our contributions and recommendations serve as a stepping stone to future endeavors aimed at improving the quality and robustness of ML-driven security systems. We release our resources.

KEYWORDS

Labeling, ML, Practitioners, User Study, Cyberthreat Detection

1 INTRODUCTION

The never-ending advancements of Artificial Intelligence (AI) in research are in plain sight [32, 34], and Machine Learning (ML) techniques are now becoming increasingly integrated also in operational information systems. Among the plethora of domains in which ML has found a real-world application (e.g., [3, 54]), the one of computer security – and, in particular, **cyberthreat detection** – stands out [7]. On the one hand, by 'training' ML models over some data, it is possible to develop ML-based systems that can mitigate the threat of zero-day attacks—which cannot be countered via conventional signature-based methods [21]. On the other hand, obtaining the data required to devise such data-driven solutions is challenging—especially from an organizational perspective [9].

Indeed, it is well-known that "there is not such a thing as a foolproof system" [14], therefore it is understandable that even ML-powered defenses may fail to detect all attacks. However, while some misclassifications may not raise serious security concerns, others may conceal signs of sophisticated attacks (e.g., [25, 36]), which can lead to an entire organization becoming compromised [7]. Simultaneously, security analysts are often overwhelmed by the excessive amount of false alarms that are raised by data-driven detectors [4]. The sheer reality is that the development of ML models ready for operational cybersecurity requires the collection of data points that pertain to the *specific environment*[1] being monitored [9].

Such a peculiarity hence prevents a reliable 'transfer' of ML models between different environments [10, 15, 19], which intrinsically hinders the advancement (both in research and practice) of ML for security applications.[2] To give an idea, some security companies revealed [7] that deployment of an ML-powered detector required almost one month of data collection (and extensive fine-tuning) done in their customers' network—which are operations that must be performed *manually* and under the *responsibility* of the security company. To make things worse, the process of "obtaining a suitable training dataset" may not only entail the 'collection' of the data but also its 'annotation': in other words, there is a need to associate each data-point to a given *label* that is used during the training phase of the ML model to guide its learning [31]. Such a procedure – required for *supervised* ML methods – necessitates a human who carefully assigns every sample in a dataset to its ground truth.

Due to these reasons, in the security domain, it is now acknowledged that "[data] labeling is expensive" [9, 27], and abundant efforts have attempted to address this issue. For instance, many papers discuss ways to 'optimize' the labeling process (e.g., by proposing active learning strategies [49]), or 'decrease the cost' of labeling (e.g., by assigning coarse labels [59]); others seek to reduce the amount of 'labeled instances' required to develop proficient ML models (e.g., few-shot learning [61]). Finally, some works (e.g., [47]) advise that 'unsupervised' ML methods are more appropriate for cyberthreat detection, due to the absence of a labeling requirement [21]. However, despite a rich literature on this subject, we asked ourselves: "How prohibitive is labeling in practice?"

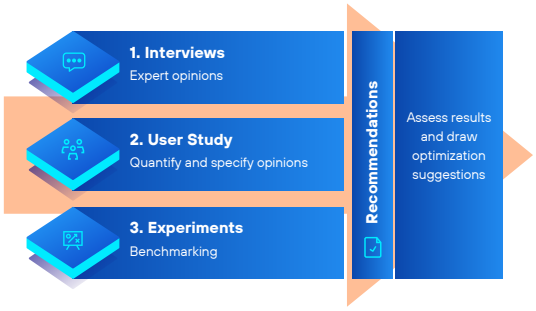


Fig. 1: Overview (and contributions) of our paper.

[1]The requirement for environment-specific data is in **stark contrast** with many other applications of ML [9]. For instance, in visual object recognition, "a cat will always be a cat" and "a dog will always be a dog"; in contrast, in cybersecurity, an IP address (or an URL) can be 'benign' for one organization, and 'malicious' for another one.
[2]The ML-based solutions proposed in many papers – despite showing near-perfect accuracy – have never seen the light of realistic deployment [8]. As a matter of fact, security practitioners see ML (and especially research papers [8]) with skepticism [20].

Perhaps surprisingly, we were unable to find any work that specifically investigated the *labeling problem in itself*. To the best of our knowledge, most existing work assumed that labeling is costly, but no work studied how this process is carried out from an organizational perspective. The only evidence we were able to find was the 2016 paper by Miller et al. [43], which estimated that security companies may have a labeling budget of 80 samples per day. However, the security landscape has changed significantly since 2016 [7], and (some) companies now do have labeling duties. Hence, in this paper, we seek to investigate the problem of data labeling from the viewpoint of security practitioners—with a focus on ML applications for cyberthreat detection.

Contributions. We seek to build a bridge between academic research and industrial practice in the context of data labeling for cybersecurity. To this purpose, after positioning our paper within existing literature (§2), we make the following contributions (Fig. 1):

- As a first step, we carry out **interviews with five security practitioners** having experience in ML development (§3). After qualitatively analyzing the responses, we highlight the challenges associated with data labeling, including pain points, costs, and time expenditures. Intriguingly, *we found that even practitioners cannot estimate the costs of labeling*.
- By using our interviews as a scaffold, we conduct a **user study with security experts from 13 different companies** (§4). Our semi-structured questionnaire delves deeper into the subject of data labeling, and our quantitative analyses reveal how various security companies address this problem for real-world projects. Interestingly, 31% had *never heard of the term "active learning"*, and some stated that *it leads to overconfidence*.
- To validate some of our previous findings, we perform **technical experiments focused on some labeling aspects overlooked in research** (§5). Specifically, we showcase the importance of repeated evaluations in cases of scarce labeled data; the effects of mislabeling; and the pros-and-cons of active learning methods—*which can reach plateaus with no practical benefits at all*.

After discussing our study (§6), we coalesce all of our novel findings into a set of **recommendations and takeaways** (§7) useful to improve the state-of-the-art (for both research and practice). Finally, for scientific reproducibility and to pave the way for future work, we publicly release all our resources [1].

2 BACKGROUND AND MOTIVATION

We summarize some well-known cybersecurity labeling strategies (§2.1), and then compare our paper against related work (§2.2).

2.1 Labeling Strategies (in research)

The starting point of any application of machine learning is **data**, whose role is developing a given ML model, which will then be used to analyze 'unseen' samples. In the context of cyberthreat detection, such *training* data *must* be provided with some reference information (i.e., a 'label') used to distinguish benign from malicious samples. [3] Obtaining such reference information, however, requires some human 'supervision'.[4] Consequently, whenever labeling is required, it is common [9] to introduce some form of labeling *budget*, $\mathcal{B}$, used to associate each sample, x, of a given dataset, $\mathbb{D}$, to its ground truth, y. From an organizational perspective, labeling can be seen as that process which uses $\mathcal{B}$ to obtain $\mathbb{L}$, i.e., a *labeled* dataset (having $\mathbb{L} \subseteq \mathbb{D}$) used to develop an ML model M.

Without loss of generality, we identify the following labeling approaches—discussed, adopted, and proposed in research (and sometimes associated to the term "semi-supervised learning" [9]).

- *Random* (e.g., [5]), i.e., the most naive form of labeling: After obtaining a given dataset, the annotator labels each sample without following any specific strategy, until the budget is depleted.
- *Temporal* labeling (e.g., [6]), whose goal is to label the samples according to their chronological occurrence (until the budget is depleted), to provide a "temporally consistent" testbed (useful to prevent temporal bias which may skew the results [12]).
- *Crowdsourcing* (e.g., [64]), in which the labeling efforts are delegated (by using the available budget) to third-parties. This is a common approach in computer vision [39], which is receiving attention also in cybersecurity [29] (especially when the annotation does not require extensive domain expertise).
- *Synthetic generation* (e.g., [55]), which entails the creation of specific samples whose ground truth is known "a priori".
- *Active learning* (e.g., [49]), which seeks to optimize the labeling procedure by "suggesting" specific samples to the annotator: The idea is to prioritize labeling of those samples that can maximize the learning of the ML model (see Appendix A for more details).

We also mention approaches typically denoted as *self-supervised learning*, which revolve around having the ML model to (iteratively) learn on the (likely inaccurate) predictions that it makes when analyzing unlabeled data (e.g., [38, 56]).

We observe, however, that the practical effectiveness of all the abovementioned strategies is **questionable** or still unclear. For instance, random labeling is, by definition, inefficient (and is the source of experimental bias [9, 12]). Temporal labeling requires accurate timestamps, which are not always available [8]. Crowdsourcing is reliant on the judgment (and 'honesty' [41]) of people who may not be at all interested in the performance of the resulting ML model [39]. Generating data synthetically can be economically viable (since the budget is virtually infinite), but it is difficult to do so in a realistic way (e.g., the generated data may represent threats that are well-known and for which there are already countermeasures [24]) and recent research showed plenty of inaccuracies in some popular datasets [22]. Self-supervised learning has been recently shown to provide almost negligible benefits in cyberthreat detection [9]. Finally, while active learning (AL) has consistently proven to be advantageous [9, 17, 26, 37, 49], it is still unclear *how to reliably use it in practice*: as we will show in this paper (§4), some practitioners are oblivious of the term "active learning".

2.2 Labeling in Practice (related work)

Despite thousands of papers that focus on the interplay between ML and cybersecurity (see, e.g., [7, 32, 53] for literature surveys), we observed that no paper attempted to scrutinize **how labeling is done by security practitioners**.

Indeed, we carried out an extensive review of existing cybersecurity literature, and we found that most works that seek to mitigate the problem of data-labeling simply acknowledge that "labeling is costly in practice", and then proceed to propose a solution that

[3] Cyberthreat detection is $\perp$ to *anomaly detection* (not all anomalies are "a threat" [52]).
[4] Hence the name of *supervised* ML techniques [57].

attempts to alleviate such costs. For example, in 2017, Li et al. [37] showed that using AL (w.r.t. random selection) allows a *phishing detector* to converge faster (in terms of the number of labeled instances required) to its ideal maximum accuracy; a similar finding was made in 2020 by Chen et al. [17] for *malware classification*, and in 2021 by Zhang et al. [63] for network intrusion detection. Yet, all these solutions have been assessed in a laboratory setting, and they did not undergo any form of validation by real practitioners—despite achieving substantial performance improvements.[5] To the best of our knowledge, the few (recent) exceptions are the paper by Van Ede et al. [59], encompassing authors from both industry and academia; and the study by Fredriksson et al. [23]. However, the latter – while providing insight from practitioners – does not pertain to cybersecurity; whereas the former – which proposes a coarse labeling strategy that is validated in a real SOC – only accounts for the perspective of a single security company.

Put simply, scientific literature overlooks the *real-world implications of data-labeling in cybersecurity*—which, to the best of our knowledge, are still unknown. This is a problem because **it prevents one** from determining: (i) whether a given solution is truly applicable to a given context (i.e., does its adoption in practice yield some benefits?); (ii) which methods should be given more attention (i.e., by knowing which methods are used in practice, one can focus on improving such methods); (iii) the overall role of data-labeling in an operational workflow (i.e., do practitioners really care?). Addressing any of these issues is, however, **challenging**— especially from the perspective of a researcher. This is because doing so requires the researcher to go *beyond the lab*, i.e., they must establish some form of collaboration with security professionals—whose practices are often kept hidden (both for their own companies' security, as well as for trade-secrets [28]). For instance, receiving permission to test a given solution on a real security system may be unfeasible for researchers, whereas finding companies who are willing to disclose (parts of) their workflows is tough [42]. In this paper, we aim to overcome all such challenges.

Problem. Despite abundant works claiming that "labeling is costly in cybersecurity", there is no paper that attempted to investigate such a hurdle from the perspective of *practitioners*.

To shed light on the process of data labeling in operational cybersecurity, we reach out to security practitioners (through both expert interviews and user studies) and ask them to share some insights deriving from their daily routines.[6] We then carry out proof-of-concept experiments to validate some of the findings brought to light by our prior analyses. Such a twofold approach is **unusual in related literature**. Indeed, technical papers (e.g., [26]) tend to overlook the perspective of practitioners; whereas papers that investigate the practitioners' viewpoint (e.g., [23]), do not perform any sort of validation—aside from not being focused on cybersecurity.

[5] Albeit, interestingly, a recent work [9] revealed that most comparisons presented shortcomings, thereby questioning whether prior work was effective even in research.
[6] **Ethics:** Our institutions know and approve this research. We follow the Menlo report.

3 EXPERT INTERVIEWS

Our first contribution are the findings of interviews with experts in the field of ML and cybersecurity. We describe the methodology (§3.1); then, we present (§3.2) and discuss (§3.3) our results.

3.1 Method

The goal of these interviews was to identify pain points in the data labeling process, assess its costs and time factors, and gather insights through narrative and directed open-questions [60].

Participants. To identify suitable subject matter experts (SME), we reached out to over 40 companies with expertise in cybersecurity and ML. Companies actively involved in ML programming for cybersecurity applications were specifically targeted, rather than those solely using pre-existing ML solutions. Despite sending hundreds of emails, we found an agreement only with five SMEs, each representing a different company (located in Europe, and having >500 employees). Such SMEs agreed to share some information on their daily routines (due to NDA, we cannot reveal more information on our participants). All these difficulties (common in related studies, e.g., [4, 42]) increase the value of our findings.

Questions. After reaching an agreement with our SME, two authors held various brainstorming sessions aimed at deriving a set of questions that would be used as a basis for the interviews. Specifically, we sought to frame open-ended questions that would facilitate a broad discussion, which allows for uncovering non-obvious issues—while accounting for potential NDA binding the interviewee. Eventually, we concocted eight questions, for which we also anticipated potential answers and prepared likely follow-up questions to delve deeper into specific issues based on the interviewee's responses. In particular, for each question, we predicted between one and five potential answers and defined between three to seven follow-up questions. This laid the foundation for gaining a comprehensive understanding of the practical implications, which formed the basis for subsequent endeavors such as user study and experiments aimed at addressing the identified issues within real-world contexts. Our generic set of questions is available in our public repository [1], but we provide a summary in Table 1.

Table 1: Interview topics

Question No.	Category - Data Labeling
1	Description of the Process
2 - 3	Resource Requirements and Time Expenditure
4 - 6	Improvement Possibilities and Strategies
7 - 8	Future Predictions and Impact

Conduction. The interviews were done (in English) by the same author, who reached out to each SME and agreed on a one-hour timeslot to have a remote interview. We did not prime our SMEs (i.e., we did not send them any questions beforehand), but they were informed that the interview would revolve around labeling practices. The interviews were not recorded, and the interviewer, after asking each question, took plenty of notes. Overall, the interviews were done between December 2022 and March 2023.

3.2 Main Findings

After carrying out the interviews, we qualitatively analyzed all the notes taken (we cannot share such notes due to NDA). We organize

our main findings in three areas (summarized in Table 2): challenges of data labeling, possible improvements (in the short-term), and avenues for future work. Let us present each of these at a high level.

Table 2: Interview results.

Challenges	Suggested Improvements	Future Work
Sensitive Data	Iterative Labeling	Self-explainable ML Models
Time Expenditure	Active Learning	Early labeling
Financial Costs	Integration of data labeling into Company Routines	
Lack of Ground Truth		
Continuous Process		
Manual Task		

Challenges of Data Labeling. Our SMEs admitted to facing many challenges during their daily routines w.r.t. data labeling. Among these, we mention the following six.

- *Sensitive data*: Labeling sensitive data poses challenges due to strict privacy regulations. Systems that ensure no direct human interaction with the data are needed to maintain confidentiality.
- *Time expenditure*: The time spent on data labeling varies depending on the data type and system dynamics. The complexity is increased by system changes within a short period. Estimating the exact percentage of time spent on labeling is difficult but it consumes a significant amount of time, especially in supervised methods for threat modeling or identifying malicious patterns.
- *Costs of data labeling*: Data labeling constitutes a significant portion of the overall costs associated with developing an ML model. However, discussing specific cost numbers is challenging and SMEs have limited knowledge about the costs due to its ongoing nature and budget allocations.
- *Lack of ground truth*: especially for some cyberthreats (e.g., APT [36]), it is hard even for a SME to provide a reliable label (i.e., is an attack taking place or not?). Iterative labeling is necessary due to difficulties in differentiating between different threat types and the discovery of new patterns.
- *Ongoing nature of data labeling*: Data labeling is an ongoing process due to software updates and changes in the environment or threat landscape. Labeled datasets become obsolete and relabeling is necessary based on factors such as data type and problem dynamics. This issue is often aggravated by the (well-known) likely "alert fatigue" [4].
- *Manual labeling*: Human expertise is crucial in the labeling process. Involving domain experts with a deep understanding of the cybersecurity domain is necessary for accurate labeling.

Perhaps interestingly, our interviewees never mentioned "crowdsourcing" (§2.1). This may be because their security companies handle sensitive data that cannot be offloaded to third parties.

Improvement Possibilities: According to our interviewees, there are ways to improve the current process of labeling in the cybersecurity domain. Three, in particular, were identified,

- *Iterative labeling*: Iterative labeling is beneficial for cyberthreat detection. This approach allows continuous adaptation, although it requires reevaluation when new discoveries arise.
- *Active learning*: Most companies were not familiar with the concept of active learning (AL). However, those who were aware[7] of AL recognize its potential to enhance the efficiency and effectiveness of the data labeling process.
- *Integration into company routines*: Companies strive to seamlessly integrate data labeling into their routines to ensure accurate labeling. However, widespread implementation and standardized processes are still lacking.

We anticipate that the findings above inspired us to perform our experimental campaign focused on active learning.

Looking ahead. Our interviewees made two intriguing remarks that may revolutionize the process of data labeling in cybersecurity.

- Data labeling in cybersecurity will be influenced by developments in *AI explainability*. Currently, many ML models operate as black boxes, lacking transparency in their decision-making processes. This hinders trust in the outputs of these models and the ability to justify security decisions to stakeholders. The demand for more explainable ML models in cybersecurity is growing, potentially reducing the reliance on human experts for data labeling. Models that can provide credible explanations for their decisions may eliminate the need for human verification. In contrast, more advanced models could offer additional context to assist human experts, reducing the time required for labeling.
- *Commencing data labeling early* is crucial for cost efficiency and improved learning. Even if better labeling methods emerge in the future, starting with pre-labeled data prevents starting from scratch. Early data labeling expedites the learning cycle and enhances the quality of labeling, leading to long-term cost savings. Companies should ensure their systems allow easy validation or dismissal of data with a single click. However, they must also guard against dismissing results without thorough scrutiny. Proper data labeling is essential to avoid future complications. Effectively mastering data labeling is an ongoing process involving labeling, learning from errors, and repeating the cycle. Therefore, it is advantageous for companies to initiate data labeling as early as possible.

Disclaimer: The statements above stem from our own re-elaboration of the (sparse and unstructured) answers we received during our interviews, and they reflect the opinion of SMEs.

3.3 Interpretation and Takeaways

We now attempt to interpret the responses received by our interviewees, aiming to derive some actionable takeaways.

First, **there is a huge gap between scientific research and industrial practice**. This is evidenced by the following:

- Most of our interviewees did not know the term "active learning", despite being common in related literature [9, 51].
- Despite labeling playing a crucial role in ML development (which had been known for decades [44, 57]), companies do not have established workflows for doing so in practice.
- Reaching out to practitioners was hard, since only 5 out of 40 companies accepted to participate in our interviews (a problem encountered also by other studies [42]).

Given the above, our paper is a step in the right direction.

Second, **the costs of labeling are unknown to both researchers and practitioners alike**. Whenever we asked an interviewee to provide an estimate of such costs (either in terms of allocated resources, or time spent labeling) we have never received a clear

[7] Interestingly, SOC analysts did not know the term, but were unconsciously using AL since alerts in a SOC can be labeled and are also provided with a 'confidence' score.

answer. What we find intriguing is that our interviewees belonged to large and well-known security companies—and we were expecting that the labeling workflow was at least somewhat structured; in contrast, the reality is that (at least according to our interviewees) these procedures are done manually and occasionally. Hence, we advocate for companies to take the problem of labeling more seriously (even their employees are implicitly requesting it!).

Third, **labeling is not easy even for SMEs**. This finding is crucial, especially in the cybersecurity context, since it suggests that – in reality – the data used to train operational ML models may present abundant errors. As such, from the perspective of a researcher, assuming that a given 'benchmark' dataset contains labels that are 100% accurate (which is the de-facto standard[8] in ML papers) may be overly optimistic. We argue that to represent a more realistic scenario, it is necessary to synthetically create some 'polluted' data-points which can simulate human labeling errors.

We conclude this section by reporting some insightful remarks that, despite being orthogonal to data labeling, provide further evidence of the 'disconnection' between research and practice in the context of ML security (and which complement [4]).

Reports from SOC analysts. According to some SMEs who are familiar with research, there is a discrepancy between academic assertions and the practical use of ML, particularly of "Deep Learning", in cybersecurity. Despite recent studies claiming extensive application of Deep Learning (e.g., [40]), the insights from SMEs suggest otherwise. Indeed, according to SMEs, a significant portion of cyberthreat detectors relies on rule-based methods, with ML being used only for complex scenarios.[a] Furthermore, when anomalies or changes occur in the network environment, SMEs emphasize the need to revert to simpler ML models and start afresh. This highlights the importance of efficient labeling and raises concerns about the efficacy of deep learning methods, which require larger (labeled) datasets.

[a] Allegedly, ML is only applied to approximately 20% of the incoming samples, while rules govern decision-making for around 70%.

4 USER STUDY

Drawing on the insights gleaned from the interviews (§3), we carry out a user study to further elucidate the key issues tackled by this paper. We begin by describing the adopted methodology (§4.1), and then present the results (§4.2).

4.1 Method

Contrary to the interviews, the user study entails a semi-structured questionnaire [2], meant to be answered asynchronously by a different set of participants.

Questionnaire. We designed a questionnaire having 15 questions inquiring about the role of labeling (within the participant's company) and about predictions on future developments of ML. Each question is accompanied by a set of 3 to 5 potential answers or a designated space for respondents to provide a custom response. The last three (out of 15) questions are formulated as open prompts. We provide our questionnaire in our repository [1]. To encourage participation and ensure a reasonable completion time, the user study is designed to be completed within five minutes. Participants were always given the possibility of not answering some questions.

To ensure consistency and comparability of the results, the questionnaire begins by clarifying the meaning of "manual labeling", and by defining the term "project"[9] (this term occurs frequently in the questionnaire). The first questions (Q) focus on uncovering the role of labeling in the participants' daily routines; for instance, Q3 asks "what percentage of the whole project is dedicated to labeling?" with possible answers being "less than 10%/between 10% and 20%/between 30% and 50%/more than 50%". We also inquire about a participant's opinion on "active learning". The last questions ask about expectations on the future role of supervised ML and explainability in the cybersecurity domain (both of which are linked to data labeling). We hosted the questionnaire on an unpublished website, and we never asked for participants' sensitive or personally identifiable information (the questionnaire is anonymous).

Participants. We set ourselves the goal of having an increased number of participants for the user study (w.r.t. the expert interviews). Hence, between March and June 2023, we reached out again to dozens of SMEs, each representing a unique security company, asking if they were willing to partake in a short survey about their labeling practices. We restricted all communication to emails: as soon as an SME agreed to participate in our research, we sent them a link to the questionnaire. Since the procedure was asynchronous, we do not know the identity of our respondents (who may have delegated colleagues with better expertise). Nonetheless, by the end of June 2023, we received 13 responses to our user study—all belonging to SMEs representing different security companies.[10]

4.2 Results and Takeaways

With few exceptions, all 13 SMEs responded to every question. We display the responses for three questions (Q3, Q5, Q7) in Fig. 2; the full results are in our repository [1]. Due to space limitations, we only discuss the three most relevant findings.

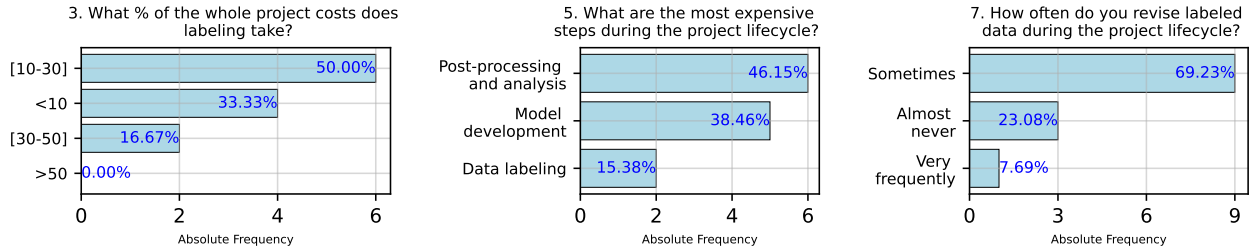
First, Q3 (Fig. 2a) shows varying perspectives on the time dedicated to data labeling, with some SMEs estimating it to be more than 30% of a project's life-cycle, while others believed it to be between 10 and 30%, or even less than 10%. Intriguingly, Q1 reveals that the projects of 54% of participants take [4–6] months, while those of 31% take more than 6 months, and the remaining 15% have projects that last [1–4] months (this supports the observation in [8] that cybersecurity is mostly outsourced). However, Q5 (Fig. 2b) indicates that labeling is (overall) "less expensive" than other maintenance operations such as post-processing and analysis. This finding aligns with Q7 (Fig. 2c), which reveals that most practitioners hardly revise previous labels. Hence, we conjecture that **labeling is done mostly at the beginning** and that – while important – after the ML model has been developed, labeling duties are overshadowed by other tasks (we invite the reader to look at Q6 in our repository).

Second, about active learning (in Q10), 4 (31%) of our respondents have "never heard of it", while 4 (31%) "are using it" and 5 (38%) are "using something similar". Further, some remarked that **AL can**

[8] This can explain why the near-perfect accuracies of ML models in research environments still trigger skepticism in security practitioners [8].

[9] We defined a project as "the development of an ML model that yields appreciable detection performance after its deployment".

[10] We believe that the higher response rate w.r.t. the interviews to be due to the survey being less time-consuming (~2 minutes) than the interview (1+ hour).



(a) Responses to Q3. **(b) Responses to Q5.** **(c) Responses to Q7.**

Fig. 2: Some responses of our user study (we provide the full results in our public repository [1]).

lead to overconfidence, because the expert will only inspect the samples suggested by the ML model, thereby potentially overlooking samples that conceal traces of serious threats. This observation is intriguing and may explain why AL is not yet widespread in security—which is a field in which even a single misclassification can lead to an entire system becoming compromised.

Third, about future prospects, Q15 reveals that our participants have mixed views on the popularity of supervised ML (w.r.t. unsupervised ML): 5 (38%) participants believe that "it is unlikely that more supervised ML will be deployed", whereas the remaining predicted that it would be either "very popular" (3, 23%) or "more used, but not by much" (5, 38%). Regardless, **the expectation is that supervised ML will remain used in cyberthreat detection**, which urges the development of effective labeling strategies.

5 TECHNICAL EXPERIMENTS

As a last contribution, we now perform proof-of-concept experiments revolving around some of our prior findings. We first examine the impact of various amounts of training data on the performance of a (supervised) ML-based detector (§5.1). Then, we investigate how human labeling errors can affect the quality of an ML model (§5.2). Finally, we assess the benefits of active learning (§5.3).

Dataset. The cyberthreat detection landscape is large, since it encompasses, e.g., malware, phishing, and network intrusion detection—all of which being domains for which many ML-ready datasets exist [9]. However, recent studies revealed that publicly available datasets for network intrusion detection are flawed [22], whereas many recently proposed malware detectors in research entail deep learning—which our SME regarded with skepticism (see end of §3.3). Hence, we focus these experiments on the problem of *phishing website detection*, given that (i) many works (e.g., [11, 45, 58]) showed that "shallow" ML algorithms outperform those based on deep neural networks; and that, for phishing detection, (ii) we are more confident that the labels are correct [9]. Among the many datasets containing phishing and benign websites (e.g., [11, 45]), we chose the well-known one by Chiew et al. [18]. It contains 10 000 samples (webpages), equally split between benign (taken from Alexa top) and phishing (taken from OpenPhish and PhishTank).

Disclaimer: The goal of our evaluation is to guide future research by suggesting some "viewpoints" often neglected in related literature (but relevant in practice). We do not claim technical novelty, but our results can serve as a benchmark (we share our code [1], which also includes the hyperparameters and low-level details).

5.1 Training Size Impact (Baseline)

As a starting point, we study the performance of an ML-detector as a function of the amount of labeled data used during its training phase. While similar studies have been carried out in the past (even for phishing website detection—e.g., [9, 37, 62]), we are not aware of works that performed such an evaluation on our chosen dataset. Furthermore, related papers on phishing detection perform their experiments on "private" data (such as [37, 62]). Hence, our testbed represents a valuable benchmark for future work.

Setup. We embrace the recommendations of [9]. First, we take our dataset [18] (having 5k benign/phishing samples), $\mathbb{D}$, and extract a test partition, $\mathbb{E}$, containing 20% of $\mathbb{D}$ (i.e., 1k benign/malicious samples). The remaining 80% samples of $\mathbb{D}$ are then treated as data usable for training, $\mathbb{T}$. For developing our ML detector, we rely on the random forest (RF) algorithm—which has been shown to consistently outperform other types of classification algorithms for phishing website detection (e.g., [18, 37, 45, 58]). Since in this experiment we want to measure the impact of different amounts of labeled data, we train ML models by randomly sampling from $\mathbb{T}$ at 1% increments, spanning from 1% (i.e., 80 labeled samples) to 100% of $\mathbb{T}$ (i.e., 8 000 samples); for consistency, we ensure that every subset is balanced. Hence, for every considered subset of $\mathbb{T}$, we train an RF classifier and assess its performance on $\mathbb{E}$. Finally, to provide a statistically significant benchmark, we repeat the sampling 30 times (i.e., we develop 3 000 ML models: 30 trials×100 subsets of $\mathbb{T}$): according to [9], simulations of scarce amounts of labeled data (drawn from a large labeled dataset) may present sampling bias which must be accounted for by repeating the draw many times (which is not done, e.g., in [62]). Inspired by this observation, we will *compare the results of a 'single' trial with the results (averaged) of* 30 *trials*. We measure the performance of every assessment via common evaluation metrics, i.e., accuracy, recall, precision, and F1-score; for simplicity, we only consider accuracy in this section (we report the other metrics in our repository [1]).

Results. We visualize the results of this experiment in Fig. 3, showing the performance (y-axis) as a function of the size of the training set (x-axis). The green line refers to the average accuracy over the 30 trials, whereas the red line refers to the accuracy of a single (randomly chosen) trial. By observing the green line, we can see that the accuracy (we recall that $\mathbb{E}$ is a balanced dataset) is already above 88% with only 1% of $\mathbb{T}$, and it reaches 95% with 12% of $\mathbb{T}$. This is an intriguing result (which partially echoes those in [9, 37]) since it shows that (at least on this dataset) it is not necessary to

resort on a large labeled dataset to develop proficient ML detectors. Nonetheless, by observing the red line, we see an inconsistent trend (w.r.t. the stable one of the green line): e.g., for the red line, the accuracy for 18% of $\mathbb{T}$ is 97% whereas the green line reaches such a value only for 35% of $\mathbb{T}$. This underscores the importance of carrying out multiple trials, since a single 'lucky' draw may yield to overly-optimistic performance. Finally, we also note that the average accuracy with 100% of $\mathbb{T}$ (i.e., 80% of the original $\mathbb{D}$) is ~99%, a result that aligns with the one by the creators of $\mathbb{D}$ [18] (which confirms the quality of our implemented ML detector).

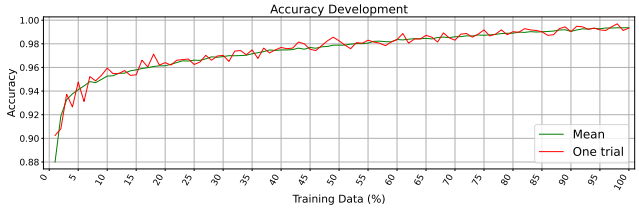


Fig. 3: Performance as a function of the training size. We further compare the average performance (over 30 trials) w.r.t. a single run.

Takeaways.[a] First, large labeled datasets are not always necessary to yield appreciable performance—hence we endorse researchers to experiment with smaller amounts of labeled data. Second, when randomly sampling from small datasets, the performance between multiple and single trials is remarkably different—hence we encourage researchers to repeat their assessments.

[a]These only apply to our dataset and ML algorithm, and we do not generalize.

5.2 Human Error Impact

Next, inspired by some of our findings (revealing that labels may be revised—see §4.2), we seek to evaluate what happens if some of the labels in a given training dataset are incorrect.

Setup. We follow a similar procedure as in the previous experiment (see §5.1). The first difference is that we consider only three (instead of a hundred) subsets of $\mathbb{T}$: 50%, 80%, 100% (i.e., 40%, 64%, 80% of the original $\mathbb{D}$). The second difference entails the method we use to simulate the erroneous labels. Specifically, for any given subset of $\mathbb{T}$ used for training, we 'flip' the class of some of its labels—thereby inducing some form of "self-poisoning" [33]. We consider five *flip ratios* (listed in Table 3), each denoting the percentage of samples of a given class (benign or malicious) whose label is flipped (i.e., a 'benign' sample is assigned to a 'malicious' label, and vice-versa). We repeat all experiments 30 times, averaging the performance—always measured on the same $\mathbb{E}$ (20% of $\mathbb{D}$) for consistency.

Table 3: Flip Ratios—We simulate poisoning by flipping the label ('benign' becomes 'malicious', and vice-versa) of a subset of the training data.

	Malicious	Benign	Poisoning
Clean	0.0	0.0	0%
Poison	0.10	0.10	20%
	0.10	0.20	30%
	0.20	0.10	30%
	0.20	0.20	40%

Results. We visualize the results in Fig. 4, showing the performance (y-axis) for each training subset of $\mathbb{T}$ (group of bars), and for each flip ratio (individual bars); specifically, the light-blue bar is 'clean' (which we use as baseline) and the green ones entail 'poisoning'. Fig. 4(a) focuses on F1-score, whereas Fig. 4(b) on the (absolute) false positives. By observing Fig. 4(a), an intriguing result is that, despite the substantial difference in training data size, the performance barely changes (i.e., each bar of the same color has very similar F1-scores—confirmed by a two-sample statistical t-test). Furthermore, another surprising observation is that the effect on the F1-score is mild: even when 40% of the labeled data is poisoned (dark-green bar), the drop w.r.t. the baseline (light-blue bar) is minor (from 0.97 to 0.92). However, a two-sample statistical t-test confirms that the drop is statistically significant (p-value ≈ 0).

In contrast, by focusing on Fig. 4(b), we see an almost contradictory result: the highest number of false positives is always achieved by the middle bar in each group—which is *not the one with the largest percentage of poisoning* (which is the rightmost bar in each group). Given that in phishing detection false positives tend to be very annoying to end-users, this reveals that practitioners should be extra-cautious when assigning 'malicious' labels (indeed, the middle bar has 20% of benign samples being turned into malicious ones, and 10% of malicious samples being turned into benign ones).

Takeaways.[a] For certain amounts of training data size, mislabeled data (poisoning) leads to negligible performance differences. However, while the F1-score appears to be a 'robust' metric to poisoning, the amount of false positives is affected by a specific type of poisoned classes. Hence, we endorse researchers to pay more attention to the poisoned class. For mitigations, see, e.g., [16].

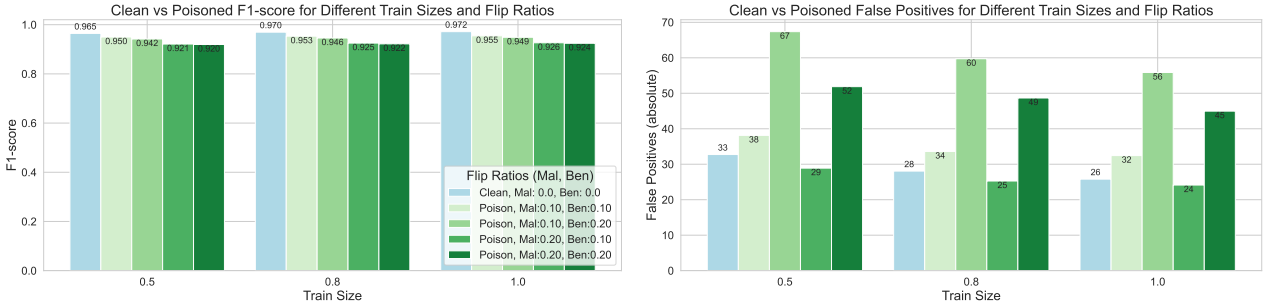
[a]These only apply to our dataset and ML algorithm, and we do not generalize.

5.3 Active Learning Gain

Lastly, we turn the attention to active learning (AL) due to the 'mixed' viewpoint expressed by our SME on this technique (see §3.2 and §4.2). We recall that extensive background on AL is in Appendix A, which discusses the specific method of *uncertainty sampling*—which we will use in our experiments due to its simplicity and demonstrated effectiveness (e.g., [9, 37, 49]).

Setup. We adopt a similar setup as in §5.1. The implementation of AL is similar[11] to the one in [9]. However, the difference lies in our assessment methodology. In particular, we seek to pinpoint the performance gain by assuming a fixed labeling budget but spread over many iterations. To give an idea, assume that an annotator has a labeling budget of 50 samples. In [9] (or also in [37]) the authors compared the gain using, e.g., 50 randomly chosen labeled samples w.r.t. 50 actively suggested labeled samples. In contrast, we want to compare what happens if the 50 actively suggested labeled samples derive from the annotator labeling such samples "all together" w.r.t. doing so by splitting the labeling task in "mini-batches". I.e., labeling a subset of these 50 (e.g., 25), and then using such batch to update the ML model which is then used to provide "updated suggestions", which will finally be provided to the annotator for another (or more) round of labeling. To do this, we 'fix' the labeling budget to the entire $\mathbb{T}$, and then consider different 5 different amounts of labeling iterations, i.e., [2,4,16,32,64]; the first iteration is always randomly

[11]For this experiment we changed the RF algorithm by setting the 'bootstrap' option to False (setting it to True, which we did in §5.1, yielded spurious artifacts here).



(a) Effect on F1-score (averaged over 30 trials).

(b) Effect on False Positives (absolute—averaged over 30 trials).

Fig. 4: Impact of mislabeling. We simulate human error by flipping the label of some subsets of the training data to see how much the performance changes.

chosen (this is a realistic assumption—§3.2 and [9]). After each iteration, we measure the performance of the resulting updated ML model (always on the same $\mathbb{E}$, for consistency). As usual, we repeat these experiments 100 times for statistical robustness.[12]

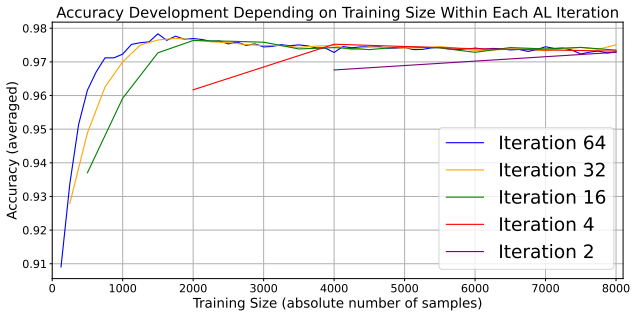


Fig. 5: Impact of Active Learning (avg 100 trials). We compare the gains of labeling the suggested samples "all together" w.r.t. doing so over many iterations—each done by updating the model and suggesting new samples.

Results. We report the results in Fig. 5, which shows the Accuracy (y-axis) as a function of the amount of the training size (x-axis); lines indicate the specific amount of iterations. As usual, we report further visualizations entailing other performance metrics (F1-score, Recall, Precision) and training sizes (50% and 80% of $\mathbb{T}$) in our repository [1]. By observing Fig. 5, we see that the first iteration (which reports the performance after randomly choosing samples) is always much worse than the following ones—which is expected. However, we make an intriguing observation: there is a stark difference between the 'gain' of multiple labeling tasks, w.r.t. fewer ones. To appreciate this, we consider two use cases. **(1) Scarce Budgets**, i.e., when an annotator can label at most 1.5k samples. Let us focus our attention on the blue, yellow, and green lines. For the green line, after one round of active learning (500 random and 500 suggested labels over a single round), the Accuracy is 96%; for the yellow line, after four rounds (250 random, and 750 suggested labels over three rounds) the Accuracy is 97%; for the blue line, after eight rounds (125 random, and 875 suggested labels) the Accuracy is 97.2%. This may indicate that splitting the labeling task into multiple batches may be advantageous. However, this is not true for **(2) Abundant Budgets**, i.e., when an annotator can afford more than 4k labels. Let us compare the blue with the orange line. For the orange line, after one round of active learning, (2k random and 2k suggested labels), the Accuracy is 97.5%. For the blue line, after 32 rounds (125 random and 3875 suggested), the Accuracy is 97.3%. Indeed, after a certain point, the performance saturates and there is no gain in splitting the labeling into multiple batches.

> **Takeaways.**[a] Applying active learning (through uncertainty sampling) by splitting the labeling task into multiple batches is advantageous in the initial development phases, but yields no returns after some saturation points. We endorse researchers to identify these plateaus in other domains and datasets.

[a] These only apply to our dataset and ML algorithm, and we do not generalize.

6 DISCUSSION

Comparison with Prior Work. Our paper shares similarities with prior (peer-reviewed) works that touch the problem labeling in practice. Here, we discuss two of these. **(1)** Fredriksson et al. [23] carry out interviews (in 2019) with five SMEs belonging to two companies (ours belong to five companies) in a single country (ours belong to five countries). However, the role of cybersecurity is unclear: the two companies to which their SMEs belong to are only reported to be "telecommunication providers" and "a company specialized in labeling". Intriguingly, one participant from the latter company reported that "labeling takes 200 times less with active learning", which is in stark contrast to what most of our SMEs reported (both in our interviews and in the user study). **(2)** Koh et al. [35] conduct semi-structured interviews with 21 ML practitioners (4 of whom work in cybersecurity) on the EU AI Act. While their results do not allow to identify the specific responses of the security practitioners, they found that 20% of their interviewees perform a "thorough labeling process", potentially suggesting that some industries do have systematic labeling approaches in place.

Implications for AI Security. Our study reveals that the process of labeling in cybersecurity is still at an early stage and that the

[12] **For example**, 4 iterations means that $\mathbb{T}$ (having 8k samples) is first randomly sampled (by drawing 2k samples) used to train an initial ML model M; this M is then used to suggest 2k samples to an annotator and, after being correctly labeled, will be used to update M (which is now trained over 4k samples). The updated M will then further suggest 2k samples to label, and then be updated again (now with 6k samples). The procedure will be repeated one last time—thereby exhausting the labeling budget of 8k samples (i.e., the full $\mathbb{T}$). After every update of M, we assess its performance on $\mathbb{E}$. We repeat this process 10 times, and then select a new $\mathbb{E}$ (as recommended by [9]) and start again for 10 more times. We average all results. (**Runtime** is in Appendix B.)

many methods proposed in research to deal with this problem are far from being a panacea. This underscores a problem: labeled data is necessary for applications of ML in cyberthreat detection [9], but the immaturity of the currently adopted labeling practices leads to ML-driven security systems with tradeoffs. E.g., incorrectly labeled data leads to self-poisoning [33] which degrades performance (as we also showed in §5.2). Moreover, the lack of a structured approach to labeling security events also hinders updating the ML model with new (correctly) labeled data, thereby exposing it to evasion attacks [7]. Ultimately, this paper is a call to action: by building a bridge between research and practice, novel solutions—at both the technical and organizational levels —could be developed that improve the reliability (in terms of detection performance and generic robustness) of security systems empowered by ML.

Limitations. We conducted open interviews with five SMEs and carried out a semi-structured user study with 13 SMEs—all operating in the field of cybersecurity and ML, and belonging to different companies. The recruitment of experts with experience in both areas proved challenging due to their scarcity, which is common in related studies [4, 35] whose population hardly goes above 20. As such, we do not aim to generalize our findings (but we are not aware of studies focused on security that do so), and our small sample size prevents one from deriving statistically-rooted conclusions. Our experiments are a proof-of-concept and there exist many ways to carry out our evaluation. For instance, we consider only one ML algorithm (despite being the best for the chosen task [18]) and one AL strategy (which is known to work well [9]) and experiment on only one dataset (which is popular [13]) of a subdomain of cyberthreat detection—but this is due to recent works showing the unreliability of prior datasets in other domains [22, 30]. We advocate future work to replicate our experiments on different datasets (and we release our tools to facilitate this [1])

7 CONCLUSIONS AND RECOMMENDATIONS

We investigated the problem of data labeling from the perspective of operational ML security. We interviewed and carried out a user study with security professionals with experience in ML development. Our findings elucidate the hurdles and issues that practitioners face in their daily routines when managing ML-driven security systems. We then carried out technical experiments aimed at showcasing some pragmatic aspects of data labeling which are seldom considered in research on cyberthreat detection.

To improve the security and robustness of their ML systems while reducing data labeling costs, companies can take the following steps: **(1)** *Optimize Label Quality*: Ensure high-quality data labeling from the beginning to avoid costly revisions of previously labeled data, **(2)** *Implement Active Learning (AL):* AL can reduce time and cost by achieving high performance with fewer iterations, reducing the amount of data to be labeled, **(3)** *Set Stop Criteria for AL Cycles:* After a certain number of iterations, a performance plateau may be reached where further labeling efforts may not be cost-efficient. Setting stop criteria can reduce time and cost, and **(4)** *Integrate Data Labeling into Workflows:* Incorporating data labeling into ongoing work processes enhances efficiency and reduces labeling time.

Acknowledgments. We thank the Hilti Corporation for funding, and the practitioners we interviewed for their contributions, availability, and valuable feedback.

REFERENCES

[1] 2024. *Our Repository.* https://github.com/hihey54/sac24_labeling
[2] Omolola A Adeoye-Olatunde and Nicole L Olenik. 2021. Research and scholarly methods: Semi-structured interviews. *JACCP* (2021).
[3] Hojjat Aghakhani, Lea Schönherr, Thorsten Eisenhofer, Dorothea Kolossa, Thorsten Holz, Christopher Kruegel, and Giovanni Vigna. 2023. VenoMave: Targeted poisoning against speech recognition. In *IEEE SaTML*.
[4] Bushra A Alahmadi, Louise Axon, and Ivan Martinovic. 2022. 99% False Positives: A Qualitative Study of Soc Analysts' Perspectives on Security Alarms. In *USENIX Sec.*
[5] Giuseppina Andresini, Annalisa Appice, Luca De Rose, and Donato Malerba. 2021. GAN augmentation to deal with imbalance in imaging-based intrusion detection. *Future Generation Computer Systems* 123 (2021), 108–127.
[6] Giuseppina Andresini, Feargus Pendlebury, Fabio Pierazzi, Corrado Loglisci, Annalisa Appice, and Lorenzo Cavallaro. 2021. Insomnia: Towards concept-drift robustness in network intrusion detection. In *2021 ICACT*. 111–122.
[7] Giovanni Apruzzese, Pavel Laskov, Edgardo Montes de Oca, Wissam Mallouli, Luis Brdalo Rapa, Athanasios Vasileios Grammatopoulos, and Fabio Di Franco. 2023. The role of machine learning in cybersecurity. *ACM DTRAP* (2023).
[8] Giovanni Apruzzese, Pavel Laskov, and Johannes Schneider. 2023. SoK: Pragmatic Assessment of Machine Learning for Network Intrusion Detection. In *EuroS&P*.
[9] Giovanni Apruzzese, Pavel Laskov, and Aliya Tastemirova. 2022. SoK: The impact of unlabelled data in cyberthreat detection. In *IEEE EuroS&P*.
[10] Giovanni Apruzzese, Luca Pajola, and Mauro Conti. 2022. The cross-evaluation of machine learning-based network intrusion detection systems. *TNSM* (2022).
[11] Giovanni Apruzzese and VS Subrahmanian. 2022. Mitigating Adversarial Gray-Box Attacks Against Phishing Detectors. *TDSC* (2022).
[12] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. 2022. Dos and don'ts of machine learning in computer security. In *USENIX Security*.
[13] Abdul Basit, Maham Zafar, Xuan Liu, Abdul Rehman Javed, Zunera Jalil, and Kashif Kifayat. 2021. A comprehensive survey of AI-enabled phishing attacks detection techniques. *Telecommunication Systems* (2021).
[14] Nicholas Carlini. 2021. Poisoning the unlabeled dataset of {Semi-Supervised} learning. In *30th USENIX Security Symposium (USENIX Security 21)*. 1577–1592.
[15] Marta Catillo, Andrea Del Vecchio, Antonio Pecchia, and Umberto Villano. 2022. Transferability of machine learning models learned from public intrusion detection datasets: the cicids2017 case study. *Software Quality Journal* (2022).
[16] Abraham Chan, Arpan Gujarati, Karthik Pattabiraman, and Sathish Gopalakrishnan. 2022. The fault in our data stars: studying mitigation techniques against faulty training data in machine learning applications. In *Proc. IEEE DSN*.
[17] Chin-Wei Chen, Ching-Hung Su, Kun-Wei Lee, and Ping-Hao Bair. 2020. Malware family classification using active learning by learning. In *ICACT*.
[18] Kang Leng Chiew, Choon Lin Tan, KokSheik Wong, Kelvin SC Yong, and Wei King Tiong. 2019. A new hybrid ensemble feature selection framework for machine learning-based phishing detection system. *Inf. Scie.* (2019).
[19] Gustavo de Carvalho Bertoli, Lourenço Alves Pereira Junior, Osamu Saotome, and Aldri Luiz dos Santos. 2023. Generalizing intrusion detection for heterogeneous networks: A stacked-unsupervised federated learning approach. *Comp. Secur.* (2023).
[20] Markus De Shon. 2019. Information Security Analysis as Data Fusion. In *FUSION*.
[21] Luis Dias, Simão Valente, and Miguel Correia. 2020. Go with the flow: Clustering dynamically-defined netflow features for network intrusion detection with DynIDS. In *IEEE NCA*.
[22] Gints Engelen, Vera Rimmer, and Wouter Joosen. 2021. Troubleshooting an intrusion detection dataset: the CICIDS2017 case study. In *IEEE S&P Workshops*.
[23] Teodor Fredriksson, David Issa Mattos, Jan Bosch, and Helena Holmström Olsson. 2020. Data labeling: An empirical investigation into industrial challenges and mitigation strategies. In *PROFES 2020*. Springer, 202–216.
[24] Sebastian Garcia, Martin Grill, Jan Stiborek, and Alejandro Zunino. 2014. An empirical comparison of botnet detection methods. *Comp. Secur.* (2014).
[25] Ibrahim Ghafir, Mohammad Hammoudeh, Vaclav Prenosil, Liangxiu Han, Robert Hegarty, Khaled Rabie, and Francisco J Aparicio-Navarro. 2018. Detection of advanced persistent threat using machine-learning correlation analysis. *FGCS* (2018).
[26] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. 2009. Active learning for network intrusion detection. In *AISec*. 47–54.
[27] Jorge Luis Guerra, Carlos Catania, and Eduardo Veas. 2022. Datasets are not enough: Challenges in labeling network traffic. *Computers & Security* (2022).
[28] David R Hannah. 2005. Should I keep a secret? The effects of trade secret protection procedures on employees' obligations to protect trade secrets. *Organ. Sci.* (2005).
[29] Jiwon Hong, Taeri Kim, Jing Liu, Noseong Park, and Sang-Wook Kim. 2020. Phishing url detection with lexical features and blacklisted domains. *Adaptive autonomous secure cyber systems* (2020), 253–267.
[30] Paul Irolla and Alexandre Dey. 2018. The duplication issue within the drebin dataset. *J. Comp. Vir. Hack. Tech.* (2018).

[31] Robert J Joyce, Edward Raff, and Charles Nicholas. 2021. A framework for cluster and classifier evaluation in the absence of reference labels. In *AISec*.
[32] Nektaria Kaloudi and Jingyue Li. 2020. The ai-based cyber threat landscape: A survey. *ACM Computing Surveys (CSUR)* 53, 1 (2020), 1–34.
[33] Zeliang Kan, Feargus Pendlebury, Fabio Pierazzi, and Lorenzo Cavallaro. 2021. Investigating labelless drift adaptation for malware detection. In *AISec*.
[34] Davinder Kaur, Suleyman Uslu, Kaley J Rittichier, and Arjan Durresi. 2022. Trustworthy artificial intelligence: a review. *ACM Computing Surveys (CSUR)* (2022).
[35] Fiona Koh, Kathrin Grosse, and Giovanni Apruzzese. 2024. Voices from the Frontline: Revealing the AI Practitioners' viewpoint on the EU AI Act. In *HICSS*.
[36] Antoine Lemay, Joan Calvet, François Menet, and José M Fernandez. 2018. Survey of publicly available reports on advanced persistent threat actors. *Comp. Secur.* (2018).
[37] Jhen-Hao Li and Sheng-De Wang. 2017. PhishBox: An approach for phishing validation and detection. In *IEEE DASC/PiCom/DataCom/CyberSciTech*.
[38] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. 2021. Self-supervised learning: Generative or contrastive. *IEEE Transactions on knowledge and data engineering* 35, 1 (2021), 857–876.
[39] Alexandra Sasha Luccioni and David Rolnick. 2023. Bugs in the data: How ImageNet misrepresents biodiversity. In *AAAI Conference on Artificial Intelligence*.
[40] Samaneh Mahdavifar and Ali A Ghorbani. 2019. Application of deep learning to cybersecurity: A survey. *Neurocomputing* (2019).
[41] Tenga Matsuura, Ayako A Hasegawa, Mitsuaki Akiyama, and Tatsuya Mori. 2021. Careless participants are essential for our phishing study: Understanding the impact of screening methods. In *Proceedings of the 2021 EuroUSEC*. 36–47.
[42] Jacqueline Meyer and Giovanni Apruzzese. 2022. Cybersecurity in the Smart Grid: Practitioners' Perspective. In *ICSS Workshop (co-located with ACSAC)*.
[43] 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 *Proc. Int. Conf. DIMVA*. 122–141.
[44] Eric Mjolsness and Dennis DeCoste. 2001. Machine learning for science: state of the art and future prospects. *Science* (2001).
[45] Biagio Montaruli, Luca Demetrio, Maura Pintor, Battista Biggio, Luca Compagna, and Davide Balzarotti. 2023. Raze to the Ground: Query-Efficient Adversarial HTML Attacks on Machine-Learning Phishing Webpage Detectors. In *AISec*.
[46] Luis Muñoz-González, Javier Carnerero-Cano, Kenneth T Co, and Emil C Lupu. 2019. Challenges and Advances in Adversarial Machine Learning. *Resilience and Hybrid Threats* (2019), 102–120.
[47] Antonia Nisioti, Alexios Mylonas, Paul D Yoo, and Vasilios Katos. 2018. From intrusion detection to attacker attribution: A comprehensive survey of unsupervised methods. *IEEE Communications Surveys & Tutorials* 20, 4 (2018), 3369–3388.
[48] David Pape, Sina Däubener, Thorsten Eisenhofer, Antonio Emanuele Cinà, and Lea Schönherr. 2023. On the Limitations of Model Stealing with Uncertainty Quantification Models. *ESANN*.
[49] Feargus Pendlebury, Fabio Pierazzi, Roberto Jordaney, Johannes Kinder, and Lorenzo Cavallaro. 2019. {TESSERACT}: Eliminating experimental bias in malware classification across space and time. In *USENIX Security 19*. 729–746.
[50] Bahman Rashidi, Carol Fung, and Elisa Bertino. 2017. Android malicious application detection using support vector machine and active learning. In *CNSM*.
[51] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Brij B Gupta, Xiaojiang Chen, and Xin Wang. 2021. A survey of deep active learning. *ACM computing surveys (CSUR)* 54, 9 (2021), 1–40.
[52] William Robertson, Giovanni Vigna, Christopher Kruegel, Richard A Kemmerer, et al. 2006. Using generalization and characterization techniques in the anomaly-based detection of web attacks. In *NDSS*.
[53] Iqbal H Sarker, ASM Kayes, Shahriar Badsha, Hamed Alqahtani, Paul Watters, and Alex Ng. 2020. Cybersecurity data science: an overview from machine learning perspective. *Journal of Big data* 7 (2020), 1–29.
[54] Avishag Shapira, Alon Zolfi, Luca Demetrio, Battista Biggio, and Asaf Shabtai. 2023. Phantom Sponges: Exploiting Non-Maximum Suppression to Attack Deep Object Detectors. In *IEEE/CVF Winter Conf. Appl. Comp. Vision*.
[55] Iman Sharafaldin, Habibi Lashkari, and Ali A Ghorbani. 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp* (2018).
[56] Thalles Silva, Helio Pedrini, and Adín Ramírez Rivera. 2023. Self-supervised Learning of Contextualized Local Visual Embeddings. In *VIPriors 4*.
[57] Robin Sommer and Vern Paxson. 2010. Outside the closed world: On using machine learning for network intrusion detection. In *IEEE S&P*.
[58] Ke Tian, Steve TK Jan, Hang Hu, Danfeng Yao, and Gang Wang. 2018. Needle in a haystack: Tracking down elite phishing domains in the wild. In *IMC*.
[59] Thijs Van Ede, Hojjat Aghakhani, Noah Spahn, Riccardo Bortolameotti, Marco Cova, Andrea Continella, Maarten van Steen, Andreas Peter, Christopher Kruegel, and Giovanni Vigna. 2022. Deepcase: Semi-supervised contextual analysis of security events. In *2022 IEEE SP*. 522–539.
[60] Susan C Weller, Ben Vickers, H Russell Bernard, Alyssa M Blackburn, Stephen Borgatti, Clarence C Gravlee, and Jeffrey C Johnson. 2018. Open-ended interview questions and saturation. *PloS one* (2018).
[61] Congyuan Xu, Jizhong Shen, and Xin Du. 2020. A method of few-shot network intrusion detection based on meta-learning framework. *IEEE TIFS* (2020).
[62] Jun Yang, Pengpeng Yang, Xiaohui Jin, and Qian Ma. 2017. Multi-classification for malicious URL based on improved semi-supervised algorithm. In *IEEE CSE*.
[63] Yong Zhang, Jie Niu, Guojian He, Lin Zhu, and Da Guo. 2021. Network Intrusion Detection Based on Active Semi-supervised Learning. In *DSN-W*.
[64] Yanqiao Zhu and Kai Yang. 2019. Tripartite active learning for interactive anomaly discovery. *IEEE Access* 7 (2019), 63195–63203.

A BACKGROUND ON ACTIVE LEARNING

The fundamental principle of *active learning* (AL) is to optimize the labeling procedure by "suggesting" to a given (human) annotator which (unlabelled) samples should be provided with their ground truth. The intuition is that some samples are "more informative" than others: by having an ML model be trained on such samples, it is possible to improve its learning in a cost-effective way [9].

Formally, given an ML model M_0 (having performance μ_0) and an unlabelled dataset $\mathbb{U}$, AL methods seek to identify which samples $x_a \in \mathbb{U}$, when used to update M_0 (after being correctly labeled), yield an ML model M_a whose performance μ_a is superior to the performance $\bar{\mu}$ of another ML model $\bar{M}$ obtained by updating the original ML model M_0 with any other sample $\bar{x} \in \mathbb{U}$ (with $\bar{x} \neq x_a$).

Among the many methods [51] encompassed by AL, a popular one is **uncertainty sampling** [50], which leverages the predictions of a pre-trained ML model M_0 as a guide for the "suggestions". The idea is that M_0 is likely to learn 'more' from samples that it cannot properly recognize.[13] Hence, by computing the "uncertainty" (e.g., [48]) of M_0 on the samples $x \in \mathbb{U}$, it is possible to optimize the labeling by having the annotator focus only on those samples that have the highest uncertainty by M_0. Such a procedure has been shown to be significantly more efficient than random sampling [9].

From a research perspective, it is possible to simulate the above-mentioned workflow as follows. First, given a (labeled) dataset $\mathbb{D}$, the researcher must reserve a subset $\mathbb{E}$ used for performance evaluation; and isolate a (small) portion which is considered to be labeled (i.e., $\mathbb{L}$), and then consider the remaining samples[14] as unlabelled (i.e., $\mathbb{U}$). Next, the researcher must train an ML model M_0 on $\mathbb{L}$, compute its performance μ_0 on $\mathbb{E}$, and use M_0 to analyze the samples in $\mathbb{U}$, ensuring to store the confidence (or uncertainty) of each prediction (which can be discarded). Then, the researcher must order the resulting samples according to their confidence, and use the given labeling budget $\mathcal{B}$ to 'move' the samples with the lowest confidence (or highest uncertainty) from $\mathbb{U}$ to $\mathbb{L}$ (thereby obtaining $\mathbb{L}_a$), but *by assigning them with the correct label* (which the researcher knows). Finally, the researcher must re-train M_0 on $\mathbb{L}_a$ (obtaining M_a), and assess the resulting performance μ_a on $\mathbb{E}$. Ideally, μ_a should be largely superior to μ_0, and superior to the $\bar{\mu}$ of any $\bar{M}$ yielded by re-training M_0 on any updated version of $\mathbb{L}$ obtained by using the same budget $\mathcal{B}$ through random sampling from $\mathbb{U}$. This process can be repeated many times, each time taking some samples from $\mathbb{U}$ and moving them to $\mathbb{L}$.

B RUNTIME

We perform our experiments on an Intel i9-12900H CPU (6 cores @ 5GHz) with 64GB of RAM. The runtime for performing all the experiments in §5.1 was 666 seconds; for §5.2, it was 146s; for §5.3, it was 8445s. More details are in our repository [1].

[13] In a sense, this is the principle of adversarial training [46].
[14] Importantly, $\mathbb{E} \cap (\mathbb{L} \cup \mathbb{U}) = \varnothing$