frontiers | Frontiers in Computer Science

TYPE Original Research
PUBLISHED 13 May 2024
DOI 10.3389/fcomp.2024.1321238

OPEN ACCESS

EDITED BY
Eduard Babulak,
National Science Foundation (NSF),
United States

REVIEWED BY
Naresh Thaneeru,
Kakatiya University, India
Dinesh Sharma,
Manipal University Jaipur, India

*CORRESPONDENCE
Steve Moyle
steve@moyle.info

RECEIVED 13 October 2023
ACCEPTED 29 April 2024
PUBLISHED 13 May 2024

CITATION
Moyle S, Martin A and Allott N (2024) XAI Human-Machine collaboration applied to network security.
Front. Comput. Sci. 6:1321238.
doi: 10.3389/fcomp.2024.1321238

XAI Human-Machine collaboration applied to network security

Steve Moyle[1]*, Andrew Martin[1] and Nicholas Allott[2]

[1]Department of Computer Science, Oxford University, Oxford, United Kingdom, [2]NquiringMinds, Southampton, United Kingdom

Cyber attacking is easier than cyber defending—attackers only need to find one breach, while the defenders must successfully repel all attacks. This research demonstrates how cyber defenders can increase their capabilities by joining forces with eXplainable-AI (XAI) utilizing interactive human-machine collaboration. With a global shortfall of cyber defenders there is a need to amplify their skills using AI. Cyber asymmetries make propositional machine learning techniques impractical. Human reasoning and skill is a key ingredient in defense and must be embedded in the AI framework. For Human-Machine collaboration to work requires that the AI is an ultra-strong machine learner and can explain its models. Unlike Deep Learning, Inductive Logic Programming can communicate what it learns to a human. An empirical study was undertaken using six months of eavesdropped network traffic from an organization generating up-to 562K network events daily. Easier-to-defend devices were identified using a form of the Good-Turing Frequency estimator which is a promising form of volatility measure. A behavioral cloning grammar in explicit symbolic form was then produced from a single device's network activity using the compression algorithm `SEQUITUR`. A novel visualization was generated to allow defenders to identify network sequences they wish to explain. Interactive Inductive Logic Programming (the XAI) is supplied the network traffic meta data, sophisticated pre-existing cyber security background knowledge, and one recurring sequence of events from a single device to explain. A co-inductive process between the human cyber defender and the XAI where the human is able to understand, then refute and shape the XAI's developing model, to produce a model that conforms with the data as well as the original device designers programming. The acceptable model is in a form that can be deployed as an ongoing active cyber defense.

KEYWORDS

eXplainable AI, network security, IoT security, symbolic machine learning, inductive logic programming

1 Introduction

1.1 Cyber security and its challenges

Defending assets from cyber attack remains challenging. There are many asymmetries that disadvantage the defender, including rare symptoms of compromise are buried within burgeoning log files of (most likely) normal behavior; specification of the assets are rarely complete and accurate; public knowledge of exploits are always out of date, with the continual emergence of new zero-day attacks; and systems may have a vulnerable component, but the deployment context and configuration may make it unexploitable. This leaves the defender having to reason in the complex world where the information they have access to cannot be relied upon. Ultimately, skilled cyber defenders admit that it is almost impossible for them to give any strong security guarantees (Meer, 2015).

There is a severe lack of human skilled cyber defenders. An international body that supports cyber defenders estimates that there are more than three million fewer defenders than necessary [(ISC)2, 2022]: "To adequately protect cross-industrial enterprises from increasingly complex modern threats, organizations are trying to fill the worldwide gap of 3.4 million cybersecurity workers." Organizations with cybersecurity resources are those who can afford it or must have it—industries like banking, finance, government and military. The shortfall identifies only skills of industrial enterprises—it is greater still in the small business and the domestic settings. It is here where many devices are simply installed from their packaging and left operating in an undefended environment, with little or no attention.

Real computer networks can be arbitrarily complex. For example, detailed analysis of a small organisation's network, where data was collected for half a year, showed that 181 devices were connected within its networks (see Section 2.1 for further details). Some of the devices are for normal operational duties (for example laptops, network routers, door swipe access controllers), others are defensive in nature (for example anti-virus software and vulnerability scanners). All devices pose a risk of expanding the organisation's *attack surface*.

In larger organizations the asset registers and design documentation can soon fall behind those of the true implementations. Systems architects often cannot agree what is and what is not actually running in their environments. Cyber defenders are keen to incorporate both *controls* (for example network firewalls) and *monitoring* as part of their defensive toolkit. Monitoring can be *passive* like the continuously running network security monitoring system Zeek (Zeek, 2021), or it can be *active*, like vulnerability scanners (e.g., OpenVAS, OpenVAS, 2023) where probes for *known* vulnerabilities are commissioned.

Designing a secure device is fundamentally challenging. Internet-of-Things (IoT) devices are often resource constrained so as to meet demanding low-power and low cost specifications. Security of the device is a lower priority. Certifying the level of security for a particular device design is also challenging and costly[1] [e.g., (the now super-ceded) Orange Book (United States Department of Defense, 1985) and Common Criteria (ISO/IEC, 1999)]. A certified secure device at installation time allows us to *presume* that it is *good* at that time, but some time later a *vulnerability* in that version of the device is discovered. This does not automatically mean the device is *bad*—the vulnerability needs to be *exploitable*, and that a path-way to exploitation is present.

Installers and operators of devices rarely know precisely how the device should behave. There is a growing appreciation for published device information from the device manufacturers themselves helping to improve security (Grayeli and Mulugeta, 2020; Megas et al., 2022). The *Distributed Device Descriptors (D3)* standard (ManySecured, 2021) allows both manufacturers and interested parties to specify what resources a device class requires, and the behavior that it produces. This provides the possibility to perform run-time comparisons of the behavior of an instance of the device class and its actual, monitored behavior. Deviations from specification can be caused by the device going *bad*, or simply that the specification has errors of omission or commission.

The number of computable programs is countably infinite (Turing, 1937). If we consider a crude partitioning of all programs into *good programs* and *bad programs* it is obvious that each of these partitions are themselves countably infinite. It follows that it is unrealistic, in general, to be able to guarantee that a persistent attacker will not be able to entice the computer-based systems to execute bad programs. It is easier to intercept communications between devices than to access the runtime environment of the devices. Considering the messages between computing devices, these too, are countably infinite. This makes it difficult, in the general case, to use external eavesdropping of devices' communications for accurately predicting whether the messages themselves are *sinister* or *benign*. One can also appeal to the notion of *trust* or not trusting anything one has not produced oneself [See Ken Thompson's ACM Turing Award Lecture (Thompson, 1984)] to appreciate the daunting cyber security challenges.

Standards and processes exist that allow systems to have their security properties rigorously asserted (United States Department of Defense, 1985; ISO/IEC, 1999). Although these are typically only used for high-end systems (i.e., rarely for consumer devices and IoT) they do provide empirical evidence of the existence of some sliding scale of *easy-to-secure* and *hard-to-secure* device designs. As a cyber defender, are there devices which operate in a manner that make defending them *easier* than other devices? Can we determine by observing devices externally which of them are (likely to be) easier to defend? By studying their eavesdropped connections and messages to other devices, can easier to defend devices' predictable behaviors be defended by blocking novel behaviors (or at least alarming or alerting with the novel events occur)?

1.2 Machine intelligence and explainable AI

ChatGPT and other systems powered by large language models have captured the public imagination in recent times. However, not everyone considers that these are general purpose artificial intelligences (Wolfram, 2023). In his 1950 paper (Turing, 1950), Turing goes beyond his pragmatic imitation test of machine intelligence[2] and considers the question, "Can machines think?." Turing quickly dismisses nine common philosophical objections to his question about thinking machines, and then focuses on three strategies which might lead to the creation of a thinking machine. Broadly from Muggleton (2014) these are (1) AI by programming, (2) AI by ab initio machine learning, and (3) AI using logic, probabilities, learning, and background knowledge.

Ab initio machine learning is the current state of the art of AI. It takes recorded examples of some phenomena that we wish to forecast and use algorithms for the production of models or code (see Flach, 2012). *Ab initio* refers to the way the models are updated when the new data arrives—typically by discarding the model and

1 The costs of security certification for systems is prohibitively high, and worse, only assure the current version under evaluation. Awarded certifications do not apply to future bug fixes or feature enhancements.

2 *Machine Intelligence* was the term Turing used. It was not until 1959 that John McCarthy introduced the term *Artificial Intelligence* (McCarthy, 1959).

starting the learning again, but with the additional data to utilize. Many of the current techniques give excellent forecasting accuracy when interpolating with novel inputs. This gives an illusion of intelligence. In reality these are systems typically trained to perform one, narrow, task.

Michie[3] categorizes machine learning styles as *weak*, *strong*, and *ultra strong* (Michie, 1988). His weak criterion of Machine Learning is a "system [that] uses sample data (training set) to generate an updated basis for improved performance on subsequent data." Indeed deep neural network models satisfy this criteria. He goes on to define the Ultra-strong criterion[4] of Machine Learning as a "system [that] satisfies [the] weak criterion and also can communicate its internal updates in explicit *and operationally effective* symbolic form." Beyond the standard accuracy of its forecasts, this requires that the machine learning generated model can be communicated to another agent[5]; and that the agent can be *coached* to improve its own performance on the same task that the model is used for. In other words "Don't simply give me the answer, explain it to me and teach me how to use it."

Under this lens of Michie's *Ultra-strong Machine Learning*, Deep Learning and its *black-box* style models fall short[6]. Inspecting the matrices of weights that represent their models are not comprehensible, nor are they in a symbolic form. Some researchers have found ways to identify what it is that the Deep Learning has focused on to when making its forecasts. For example in object recognition from digital images, is it the polar bear or the snowy background? People argue that this is a step to explaining the A.I.[7]—but it clearly fails Michie's stronger definitions. NIST (2021) describe a multi-stakeholder practical framework for eXplanatory AI (XAI) based on four principles : (1) explanation, (2) meaningful, (3) explanation accuracy, and (4) knowledge limits. Each of these can be seen to be encompassed, either explictly or implicitly, within Michie's *Ultra-strong Machine Learning*.

Returning to Turing's view (Turing, 1950) on the matter of the construction of an AI. His third, and preferred alternative is: *AI using logic, probabilities, learning and background knowledge*. Turing states that "…one might have a complete system of logical inference 'built in.' …the store would be largely occupied with definitions and propositions. The propositions would have various kinds of status, e.g., well-established facts, conjectures, mathematically proved theorems, statements given by an authority, expressions having the logical form of proposition but not a belief-value." In a sense, Turing is suggesting that underpinning the learning should be a compendium of existing knowledge—in a logical form that can be used to reason with. Such an approach provides benefits over the ab initio machine learning. (1). Existing knowledge can be utilized as "Background Knowledge." This has benefits for the learner—in that it is not forced to relearn every thing every time new examples appear. (2). New Learned knowledge can be independently verified. In Turing's world, it is possible to ask the learner after the fact, "What it is that you have learned?" The learner can then answer, for example, by emitting its learned model in logic. Even if the model is not perfect, it is in a form that can be tested, debugged, and then adopted into the background knowledge for future learners to use.

Going beyond systems that simply offer good predictions, Explainable Artificial Intelligence (or XAI) as described in Gunning et al. (2021) can be seen as patching a gap in contemporary black-box machine learning techniques, so as to overcome limitations in communicating the machine learning results to humans. However, Turing, in 1950, had already provided a pathway to *Explainable Artificial Intelligence*. Michie (1988), too, was insistent on machine learning results being communicated in a comprehensible manner. Comprehensible communication remains challenging, even between humans, and continues to be researched (e.g., Muggleton et al., 2018).

1.3 Machine learning challenges for cyber security

The application of machine learning techniques faces particular challenges in the domain of cyber security. Cyber defense in itself is plagued by many asymmetries that advantage the attacker, the greatest being that the defender must prevent all attacks (many of which are previously unknown), whilst an attacker need only one way of breaking in.

Machine learning—by definition—requires example data[8] to learn from. There is no shortage of data in the cyber world—for example vast quantities of log files are collected relating to systems and their operation. However, most machine learning techniques are *supervised* (for a general description of machine learning practices see Flach, 2012) requiring that each data item is *labeled* with the outcome observed. For this discussion we can use the simple labels of **sinister** and **benign**. Unfortunately, the vast log data is *unlabeled* making supervised machine learning techniques inappropriate. In cases where attacks are analyzed and the log records associated with them labeled (as sinister) there remains a vast imbalance in the training set. Machine learning algorithms give the strongest results when trained on balanced training sets: in this case equal amounts of sinister and benign example records. When the proportion of sinister to benign records is low (e.g., say, one in a hundred—which is likely to be quite an over-estimate in real-world operational environments), the models produced will almost certainly predict a new log record as benign. This leads to false negative errors where the sinister records are identified as benign—making the detection a useless defense. Pushing the learning to focus on the sinister records [e.g., by using *boosting* (Flach, 2012)] will only lead to a model that has a high false positive alarm rate—misidentifying benign records as sinister. Cyber security systems with high false positive alarm rates are considered untrustworthy, and often lead to them being ignored.

3 Donald Michie was a wartime colleague of Alan Turing. They discussed Machine Intelligence whilst playing chess socially.

4 The Ultra-strong criterion of machine learning is an incremental upgrade on the the Strong criterion adding the conjunction "and operationally effective."

5 The agent we are most interested in is a *human*.

6 Deep Neural Network models do manage to pass the less stringent criterion of *Weak* Machine Learning.

7 It seems that AI simply means machine learning in this context.

8 Examples used to learn from are known as the *training [data] set*.

In one sense, there is too much data (unlabeled logs), yet in another sense we don't have enough (sinister labeled logs). To a naive first approximation everything is considered safe. To make progress we must keep expert human cyber defenders in the machine learning cycle, and provide them with tools to make sense of what they see. If humans are key, then so too is XAI, as the cyber defenders will require machine learning frameworks that output models that they find comprehensible.

1.4 Human-Machine collaboration in scientific discovery

The practice of extracting patterns from collected data has a long history[9]. Humans are somewhat distinguished from most other animals by making tools. The usefulness and ease of use of which are important factors. If one seeks a deeper understanding of the tangible world then using the tool of *scientific method* is a good practice. For example, the inductivist incorporation of empirical observations as a basis for constructing general laws [this can be traced back at least to the 17th century (Bacon et al., 1996; Peirce, 1996; Jevons, 2012)].

For Humans to be able to collaborate with the data science tools requires that both agents—the human and the machine—understand a common language. Quite often the scientific expert is not a data analytics expert, so a team-based approach is required. Significant successes have been achieved this way where the actual scientific "discovery" is initially proposed by a machine (learning algorithm). Mutagenesis (King et al., 1996) and drug discovery (Finn et al., 1998) are examples where published results in a discipline were proposed by a machine learning system (see also Gillies, 1996). In these cases it is key that the output of the systems are comprehensible to the experts, requiring translations into the experts' preferred formats.

Considering human cyber defenders, they apply their skill and reasoning in much the same way that a detective solves a murder case. They constantly form hypotheses, collect more evidence, and focus their investigations. They will use *abductive* reasoning to form hypotheses and seek yet-to-be discovered evidence, whilst forming constraints based on the evidence that will rule out incompatible hypotheses and lines of enquiry. They may be able to conclude, based on reasonable probability, the motive, opportunity, and weapon used by a suspect. Once the case is solved, then it is possible to follow a *deductive* reasoning process that explains the case and its artifacts. Furthermore, with the solution of similar styles of cases it is possible to apply *inductive* reasoning to learn general patterns (e.g., Moyle and Heasman, 2003).

We require that the machine can make use of the existing domain knowledge with which the human expert is familiar. A bi-directional shared language is necessary. One that is directly executable is Logic programming, which has a long history of knowledge representation and reasoning. Despite its longevity and many successes standard `Prolog`, with its sound semantics and elegant computation model (e.g., Lloyd, 1984), does not directly support the forms of reasoning that humans often rely on. The more expressive Answer Set Programming (e.g., Lifschitz, 2022), allows a broader range of reasoning, and is not restricted to Logic Programs with single models. Subsets of Logic Programming using constrained natural language are enabling humans to have two-way interactions with systems (Kowalski, 2011; Schwitter, 2020).

Science proceeds where scientists can replicate another's work. A more robust advance is where scientists can refute and argue about another's findings. Indeed, philosophers of scientific method suggest that refutation is key (e.g., Popper, 1935). A strong human-machine system will be one where each party can argue both for and against the communicated understanding of the other. A framework for this scenario is reported in Srinivasan et al. (2022).

A candidate Human-Machine environment would need to support explicit symbolic knowledge, in a comprehensible and explainable form for the human, that is directly executable by the machine, where the state of knowledge can be refuted by each party, through multiple logical reasoning forms (deduction, abduction, and induction). This seems to be similar to Turing's original 1950s preferred specification (Turing, 1950) of what it will take to build a "thinking machine"—what we would now call an XAI. This research reports on efforts to develop techniques for use in an XAI environment specifically to support Cyber Defenders.

1.5 Objectives

Motivated by how difficult it is for professional cyber defenders to keep their environments safe (Meer, 2015), one of the goals is to identify ways to improve the security of Internet-of-Things (IoT) devices in a domestic setting. We wish to determine if it is plausible to use XAI-like tools to reverse-engineer behavioral models (c.f. Bain and Sammut, 1995) (i.e., rules) that a specialized IoT gateway device could utilize to only permit necessary network interactions to IoT devices and their controllers.

The main aim of this research is to test the following hypotheses:

- Hypothesis 1–Identifying **easier-to-secure periods** of operation: That it is possible to use eavesdropped communications information to identify periods of time in a network of devices that are *easier-to-secure* than other periods of time by measuring the volatility of **all** the devices' communications over time.
- Hypothesis 2–Identifying **easier-to-secure devices**: That it is possible to use eavesdropped communications information to identify devices that are *easier-to-secure* than other devices by measuring the volatility of devices' communications.
- Hypothesis 3–Reverse-engineering **behavioral models** of easier-to-secure devices: That it is possible to use eavesdropped communications information from identified *easier-to-secure* devices to build behavioral models of allowed (and disallowed) communication using XAI approaches.

These hypotheses chain together. Without evidence that there are time periods or individual devices that show reduced volatility, then it will be difficult to identify candidates that are easier to

9 Kepler, in the 17th century, is credited with fitting Tycho's planetary observational records to an elliptical orbit.

defend. Hypothesis 3 requires a positive result from Hypothesis 2. Should Hypothesis 3 be valid then a subsequent line of research would be to determine if and how the behavioral models of communication can be extracted and then deployed as a defensive tool for the device(s). This is dealt with in Sections 4, 5. Hypotheses 1 and 2 are tested in Section 3.

1.6 Outline

This is empirical research utilizing observational data collected from a real-world organization. The computer network and the data are outlined in Section 2. Then in Section 3 we consider how to identify *easier-to-secure* devices by studying the eavesdropped connections and messages to other devices. Devices that have low network volatility allow predictable behaviors to be defended by blocking novel behaviors (or at least alarming or alerting with the novel events occur). We thus identify and target one particular IoT-like device to study in detail. A form of behavioral cloning is performed using machine learning of sequences of messages used by the target device in Section 4. A novel tool to visualize the extracted patterns allows a human defender to select interesting recurring sequences generated by the target device for further analysis. Section 5 details an interactive logical induction approach to analyzing fine grained sequence patterns in the presence of existing background knowledge. The results produced conform to the expectations of eXplainable AI—they can be understood by the cyber defender. The discussion, conclusion, and suggestions for future research directions are covered in Sections 6, 7.

2 Observational case-study context

2.1 Network traffic monitoring

Systems can be observed from many perspectives and many different granularities. For example we can observe the internal operation of entire computer systems by executing them from within a virtual machine environment—a form of *white box* analysis. An alternative perspective is to observe the inter-system communications passed into and out-of a computer system—a form of passive *black box* analysis. In this research, out of operational pragmatism, we choose the latter approach, and study the observation logs recorded from eavesdropping network connections and conversations between individual devices.

To collect data for the case study, we implemented the system architecture shown in Figure 1. Network traffic metadata is passively collected by a bespoke edge device consisting of a network TAP (i.e., a device that clones data flowing across a network) and system-on-chip processor running an established Network Security Monitoring system called Zeek (2021). Data is periodically pushed to a secure cloud storage facility. This IoT-like device was developed by integrating publicly available components, in a secure and robust way, and utilizing efficient server-less cloud processing design patterns.

2.2 Materials and methods

This is an empirical enquiry using captured eavesdropped communications data. The data was from a small business from January 1st to June 20th 2021 (inclusive; 172 days) with between 28 and 185 active networked devices per day, and between approximately 63K and 562K logged connection events per day.

The data collected by the Zeek (2021) Intrusion Detection and Network Monitoring System is metadata extracted from the observed network communications. Different Intrusion Detection Systems (Khraisat et al., 2019) utilize different detection approaches ranging from being based on *anomalies* to a being based on *heuristics*. Anomaly approaches can use statistical information derived from simple measures, for example data flow rates. In this work we explore the use of a device-to-device communication volatility (see Section 3, below). Only the communications that pass through the network monitoring device are logged for analysis[10]. The map of the metadata schema is shown in Figure 2. The master data table is `conn`, which keeps a summary record for all connection events observed. The analyses focus on the `conn`ection log. An example record and details of the meta data fields available in the `conn` log are shown in Table 1. Many of the Zeek tables have a similar initial set of columns with event timestamp–`ts`, unique identifier–`uid`, originating IP address and port–`id.orig_h`, `id.orig_p`, responding IP address, and port `id.resp_h`, `id.resp_p` recurring in many tables.

2.3 From raw network to new cyber defender knowledge

Raw network logs are only one source of information. We want to process, transform, and analyze the logs so that they can be used by the Human-XAI to co-create behavioral descriptions of the device being studied. This process is sketched in Figure 3. Note that the cyber defender plays a key role in selecting the device of interest to study (*H1*), providing the background knowledge about the environment and cyber security (*H2*), and interacting directly with the XAI system to produce a new understanding that is mutually acceptable.

3 Simple Good-Turing as device volatility

We first tackle the two initial hypotheses regarding **easier-to-secure** *periods* and *devices* outlined in Section 1.5. In this section we utilize an analytic technique first developed by Alan Turing and Jack Good whilst working as code breakers at Bletchley Park. They developed frequency-based attacks on enemy cipher systems, including determining estimates of character bi-gram probabilities. Their frequency smoothing approach was based on the frequency of frequencies of observed bi-grams[11] in samples of plain-text. The models built by the technique provide an estimate of the *unseen*

10 An eavesdropper can only record what it observes. Encrypted and wireless communications were not explicitly monitored by our device.

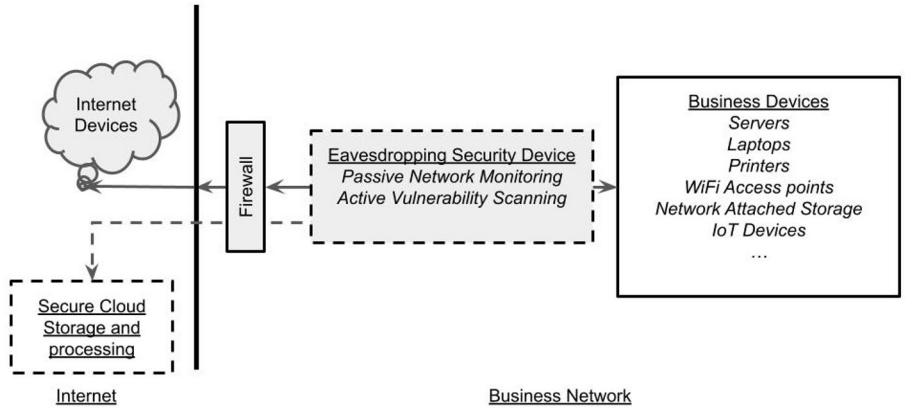

FIGURE 1
System Context diagram: a network monitoring edge device passively records metadata about observed network connections from and to an office with networked devices. Logs are securely stored and processed in the cloud.

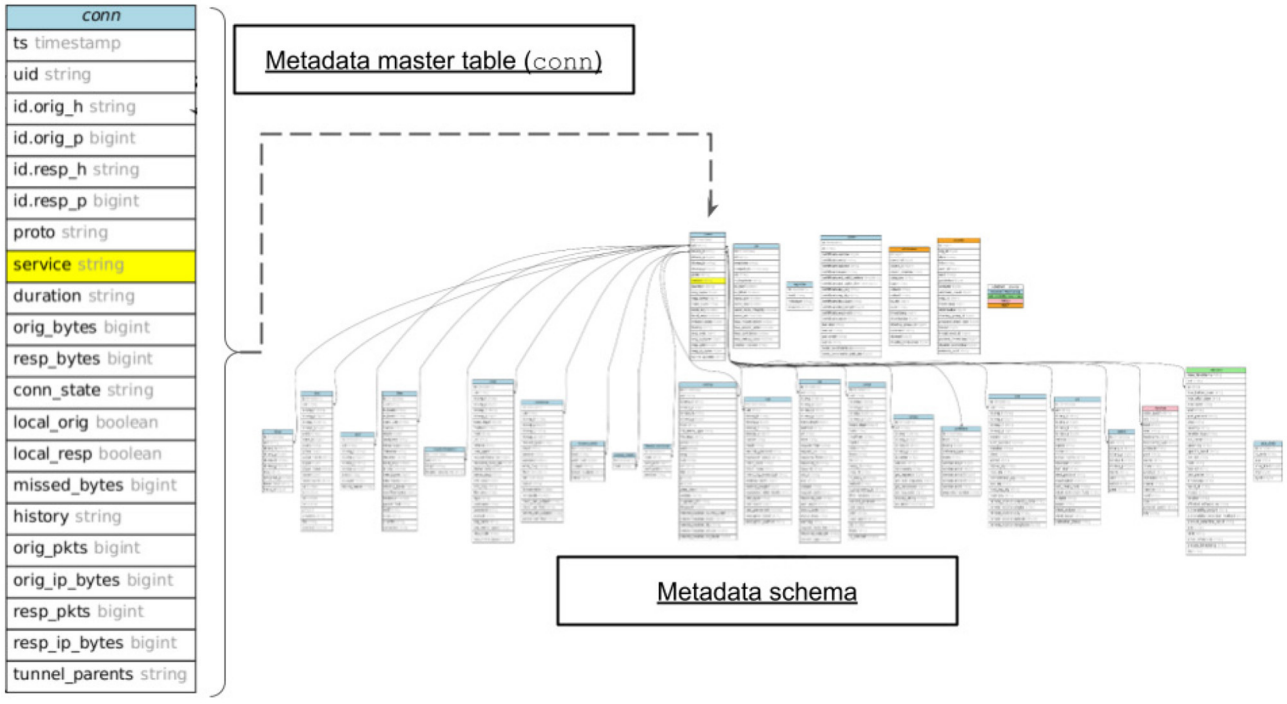

FIGURE 2
Eavesdropped network communications metadata data schema. The left hand side depicts the table name/columns for the `conn` master metadata table recorded by Zeek (2021). The right-hand side shows the complete metadata schema map (for context only). Note the many subordinate metadata tables, which provide metadata details for each `service` recorded in the `conn` table (e.g., `http`, `dns`, `dhcp`, ...).

probability mass from a sample. In this research we use a simplified form of the algorithm **SGT** from Gale and Sampson (1995) to fit Good-Turing frequency models on samples of real-world network traffic data.

11 Legend has it that Good (Good, 1953) cunningly published the technique with a motivating example relating to ornithology so as to avoid censorship.

3.1 Good-Turing frequency estimation

The original motivation for the Good-Turing Frequency estimation relates to what can be sensibly inferred *beyond* a sample of nominal objects relating to objects not present in the sample. It takes as input the counts of objects in the sample, and then uses the counts of the counts (spectra) to fit a log-log regression equation. For each model, the method returns the *slope* and the *intercept* of the regression equation, as well as a goodness-of-fit measure (r^2). The fitted model can be used to provide smoothed

TABLE 1 Column names, data types, and examples of fields from the Zeek conn metadata log.

Column name	Data type	Example	Selected descriptions
ts	timestamp	2020-12-31 23:59:22	Timestamp of connection
uid	string	CSuVAz2v3MtaiPkaja	Unique ID (key)
id.orig_h	string	10.0.100.55	Source IP address
id.orig_p	bigint	51475	Source IP port
id.resp_h	string	192.29.32.24	Destination IP address
id.resp_p	bigint	80	Destination IP port
proto	string	tcp	Protocol
service	string	http	Service type
duration	string	0 days 00:01:00.052789000	
orig_bytes	bigint	282	Originator message size
resp_bytes	bigint	510	Response message size
conn_state	string	SF	
local_orig	boolean	TRUE	
local_resp	boolean	FALSE	
missed_bytes	bigint	0	
history	string	ShADadFf	
orig_pkts	bigint	6	
orig_ip_bytes	bigint	534	
resp_pkts	bigint	4	
resp_ip_bytes	bigint	682	
tunnel_parents	string	(empty)	
year	smallint	2020	
month	smallint	12	
day	smallint	31	

NB (1) Some values have been altered to ensure confidentiality. For a complete description of fields see: https://docs.zeek.org/en/master/logs/conn.html. NB (2) The `year`, `month`, and `day` fields are additional helper fields provided independently from Zeek to improve database storage and indexing.

estimates of probabilities of objects occurring in the population, including an estimate of the *unseen probability mass* **P0**. This method works even when the underlying population cardinally is very large (or indeed infinite)[12]. We are predominantly interested in the parameter **P0**—and using it as as an indicator of *volatility*.

12 This makes the technique useful in computational linguistics where the number of potential sentences in a natural language is for all practical purposes infinite.

3.2 Analyzing network traffic with SGT

For all analyses we used data collected from the real-world network described in Section 2.2. We are interested in the originators of network messages and responders, originator-responder pairs, as well as gross network traffic, and that of specific devices. We aim to determine whether the Simple Good-Turing (SGT) models show evidence of utility in measuring network event volatility from daily `conn` data. The process for analyzing the data, particularly the SGT models, is outlined in Figure 4.

Recall that **Hypothesis 1** (see Section 1.5) concerns *volatility* for which we use the SGT estimate of unobserved species of network communication events, $P0_{events}$. Does $P0_{events}$ vary over time? If there is a variation, can we determine *low volatility* and *high volatility* periods of network traffic? A summary of the SGT analysis of the network traffic follows[13]. Figure 5 shows daily statistics for aggregate counts of source and destination connections (individually and as pairs) observed, as well as the number of devices.

The following observations are from the 172 day period for which data was collected. The daily number of devices (max. 185/min. 28) and connections (max. 561.7K/min. 63.7K) are correlated with the organizations work patterns, with noticeably lower counts on work-weekends. Network connection rates (weakly positively) correlate to low values of $P0$ (*low volatility*). The number of originating devices is about forty-fold lower than the responding devices[14], and that the values of originating and responding $P0$ do not appear to be tightly coupled.

From this, we adopt the conclusion that the SGT $P0$ measure can be used to identify time periods of higher and lower of network traffic volatility in real-world data.

Hypothesis 2 (see Section 1.5) Identifying *individual devices* that are easier-to-secure than others by using the SGT $P0$ value. For each device the SGT $P0_{Device}$ was calculated for each day. The mean $\hat{P}0_{Device}$ for the 172 day period along with the standard deviations were also calculated (see footnote[13]). Both daily and mean $P0_{Device}$ values vary significantly. with the mean SGT metric varying from lowest device to highest device of two orders of magnitude. The standard deviations also allows the identification of devices that are consistently low volatility.

Ranking the devices from lowest to highest $\hat{P}0_{Device}$ shows a sigmoid-shaped plot (see Figure 6). Figure 7 focuses in on the lowest twelve lowest scoring devices. For comparison, the period run-charts for a low and a high value $\hat{P}0_{Device}$ are shown in Figure 8.

We adopt a second conclusion that the SGT $P0$ measure can be used to identify individual device of higher and lower of network traffic volatility in real-world data.

13 For full details refer to the supporting materials (Moyle et al., 2023b).

14 For the 243 day period from 2021-01-01 to 2021-06-21 the sum of the count of each days originating devices O was 16,372 and for the responding devices R was 621,389 giving a ratio of $R/O = 37.95$.

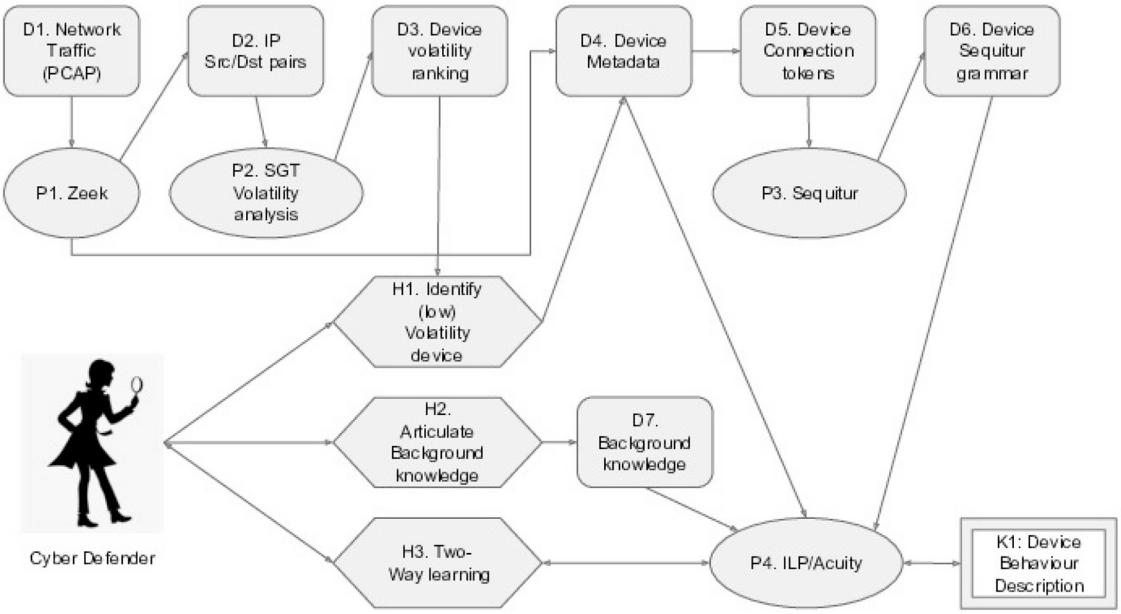

FIGURE 3
Process flow diagram: Data Elements (D) start with eavesdropped Network Traffic (D1), and are processed by processing elements (P) with the human Cyber Defender's interactions (H), using eXplainable AI, to produce new knowledge (K) culminating in the understanding of device behavior that can be used to improve security further. The data transformations *D1* to *D6* are along the top line of the figure; below are the processes *P1* to *P4* that perform the transformations. The human cyber defender plays their crucial role in *H1*, *H2*, and *H3*; with the ultimate outcome produced as new knowledge in *K1*.

1: $FirstDay \leftarrow$ `2021-01-01`
2: $LastDay \leftarrow$ `2021-06-21`
3: **for all** Day such that $FirstDay \leq Day \leq LastDay$ **do**
4: select every `id.orig_h` observed on that Day
5: Produce a count-of-counts frequency histogram F
6: Fit F to an SGT model and return $Slope$, $Intercept$, r^2, and $P0$
7: Return also the total number of unique devices D and the total number connections C
8: **end for**

FIGURE 4
Outline process for calculating Simple Good-Turing (SGT) analysis based on `id.orig_h` data recorded in the `conn` network flow meta-data table. The process is analogous for generating SGT trends for `id.orig_resp` models and `id.orig_h–id.orig_resp` pairs models.

3.3 SGT discussion

Like a skilled cyber defender, we would like to understand what the actual devices are and their roles. However, passive network eavesdropping rarely produces definitive identity information. The MAC address of a device allows us to guess the manufacturer of the device[15]. Information gleaned[16] from active vulnerability scanning can help. Combining both such sources allows us to form an opinion about one particular high scoring $\hat{P}0_{Device}$. The conjecture is that it is a WiFi access point or router which is always on and interacting with a vast number of devices on behalf of its connecting devices.

By studying the hardware *MAC* address of the third lowest scoring $\hat{P}0_{Device}$ device, and combining the information from the device's vulnerability scan (that we were fortunate to already have with the data) we determine its hardware and operating system. This is fortuitous, as it directly identifies the IoT-like network monitoring device that was placed in the network. We also make some tentative guesses that amongst the lowest five volatility scoring devices are a router, a network-attached-storage device, and a printer. Having identified an IoT device using the SGT analysis of network connection information Section 4 details how the data can be used to perform behavioral cloning.

With respect to the first hypothesis regarding whether SGT $P0$ can be used to identify fluctuations in gross network network-as-whole traffic over time, the answer is a cautious *Yes*. With respect to the second hypothesis regarding whether $P0$ can be used to identify individual devices that show markedly lower (or higher) $P0$, the answer is a more confident *Yes*.

15 The MAC address is not 100% trustworthy as it can be spoofed.

16 A website for performing MAC Address lookups is: https://www.macvendorlookup.com/.

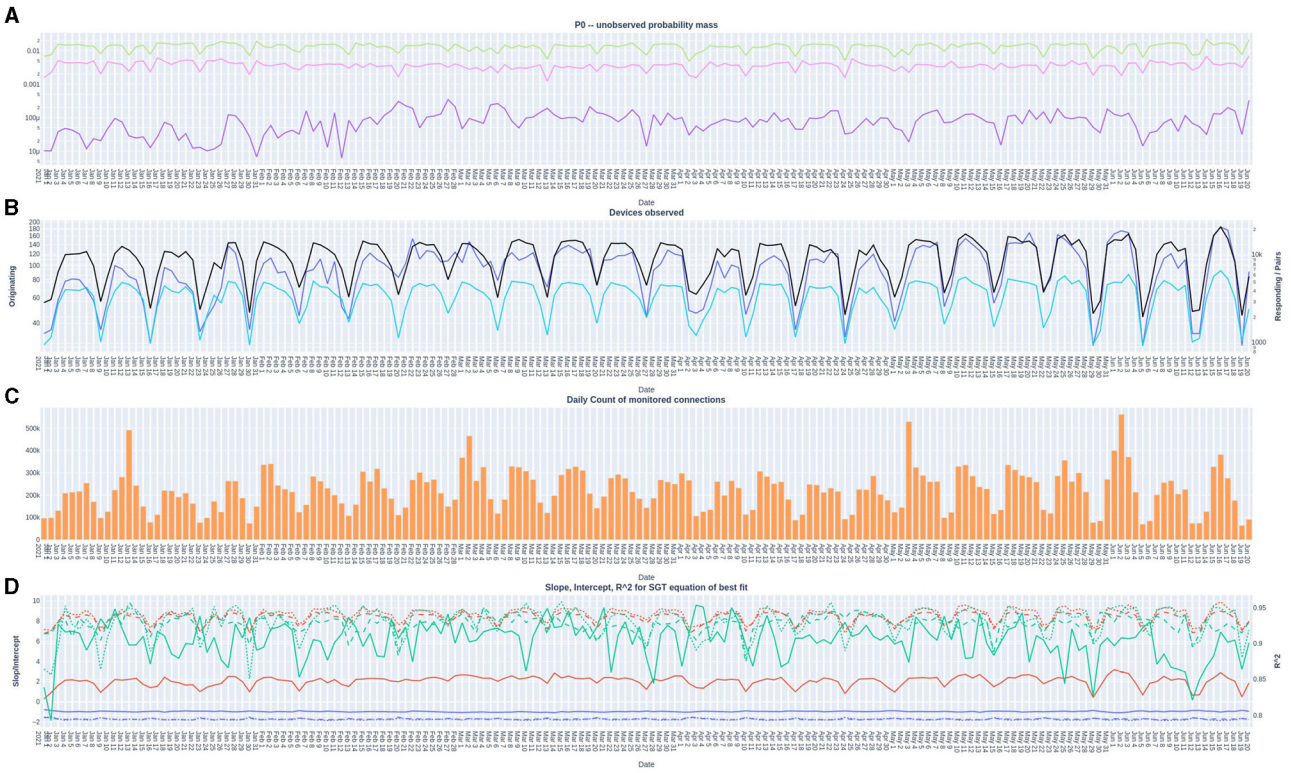

FIGURE 5
Simple Good-Turing frequency analysis of individual Originating IP addresses, individual Responding IP addresses, and combined Originating-Responding IP address pairs by day. **(A)** *P*0 SGT probability mass of unseen addresses (a proxy for volatility) Originating IP addresses (purple); Responding IP addresses (pink); Originating-Responding IP address pairs (green). **(B)** Daily count of IP addresses Originating IP addresses (dark blue); Responding IP addresses (light blue); Originating-Responding IP address pairs (black). **(C)** Orange bars number of monitored connections. **(D)** Model fit parameters intercept (red), slope (blue), r^2 (green) model for Originating IP addresses (solid lines); Model for Responding IP addresses (dashed lines); Originating-Responding IP address pairs (dotted lines).

4 Sequence grammar induction and device behavioral cloning

Behavioral cloning is the machine learning process of recording the performances of skilled agents and using induction algorithms on the traces of the behavior so as to generate models that behave in a similar manner as the original agent (Bain and Sammut, 1995). Typically the agents under consideration are skilled humans, but here we are interested in inducing models of device behavior. For this purpose we study the network event traces of an identified low SGT *P*0 volatility IoT-like device. We use the compression and grammatical induction system SEQUITUR (Neville-Manning and Witten, 1997).

One way to consider machine learning is that it is the (algorithmic) extraction of patterns from data. Another view is that machine learning is about extracting generalizations from specific examples. The related Minimum Description Length (MDL) principle is that "what is learned by a machine learning scheme is a kind of theory of the domain from which the examples are drawn" (Frank et al., 2016). From Ockham's Razor we prefer the shortest theory with the same explanatory power, which means that MDL encourages the selection of the shortest theory that allows us to reconstruct the original example data. This is similar to the objectives of a compression algorithm that allows us to regenerate the original (data) file from the generalized (or compressed) file.

In this research we use the SEQUITUR compression algorithm from Neville-Manning and Witten (1997). It is a lossless compression algorithm with a computational complexity linear in the size of the input and is particularly well suited to long sequences. Furthermore, the compressed output contains a human-readable data structure—a hierarchical grammar—that explicitly encodes for frequent sub-sequences from the original file.

4.1 Analyzing the network traffic of a single device with SEQUITUR

From the 185 devices observed we selected the device with IP address `10.0.0.145` which (1) had very low volatility; and (2) we could identify it (as the IoT-like network monitoring). The overall data transformation process is depicted in Figure 3. Here we focus on the sequence mining using SEQUITUR. For full details of the analyses including source listings refer to Moyle et al. (2023b). The outline process is:

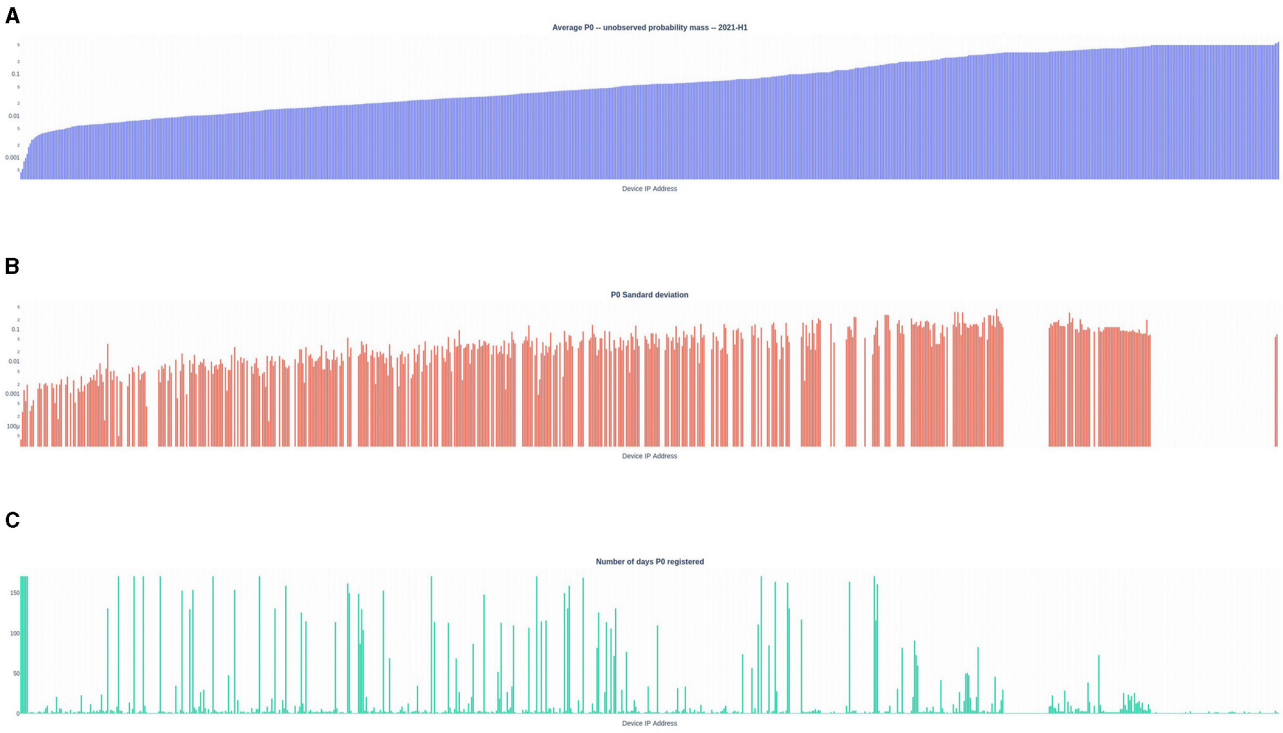

FIGURE 6
Simple Good-Turing frequency analysis summary of individual internal IP addresses observed daily rank ordered by mean P0 value for the period. The device with lowest average volatility is the left-most data-point; and the device with the highest average volatility is the right-most data-point. **(A)** blue bars – the period average $P0_{conn}$ each bar represents a single `id.orig_h` (e.g., an IP address); **(B)** red bars – the period standard deviation of $P0_{conn}$; **(C)** green bars – the number of days the `id.orig_h` is active in the period (maximum = 171 days). Note that the **(A, B)** are log scales, while the **(C)** is linear scale.

- Select a day's Zeek data for the identified device (2021-01-10, consisting of 3,613 events);
- Build string tokens combining the `conn` table fields: `id.origh_h` – source IP address, `id.resp_h` – destination IP address, `id.resp_p` – destination port number, `prot` – identified protocol, and `service` – identified service for each network connection event that the device has.
- Hash the string tokens to a unique 9 digit number as connection event IDs.
- Input the sequence of 3,613 connection event IDs into `SEQUITUR` to generate the hierarchical compression grammar.

```
RuleID: 118

Rule: [head(118),token(271311686),rule(162)]

Expansion: [271311686,847213017,847213017,271311686]
Length: 4
Frequency in training: 21/3613
Frequency in the grammar: 19

271311686:  10.0.0.145 10.0.0.137 (name_not_found) 53 udp dns
847213017:  10.0.0.145 3.209.99.56 (ec2-3-209-99-56.compute-1.
    amazonaws.com) 22 tcp -
847213017:  10.0.0.145 3.209.99.56 (ec2-3-209-99-56.compute-1.
    amazonaws.com) 22 tcp -
271311686:  10.0.0.145 10.0.0.137 (name_not_found) 53 udp dns
```

Listing 1 `SEQUITUR` **Grammar Rule 118 details. A reformatting of the raw** `SEQUITUR` **grammar in that it includes the original mapping between the network connection event metadata information and their unique 9 digit number IDs.**

4.2 Hierarchical grammar output

One of the strengths of the `SEQUITUR` approach is that it produces human comprehensible symbolic output. As it is designed for compression, the expansion process must be able recreate the original input. The first item in the `SEQUITUR` output is the *start rule* (*S*0); which is then followed by a series of *grammar rules* each with a unique rule identifier (R_i). The start rule—typically very long—contains instructions for where to expand the grammar rules. Each of the grammar rules contains grammatical structure as well as the original string elements that it encodes for.

`SEQUITUR` takes our input of 3,613 connection IDs and produces one start rule, and 129 grammar rules. Consider Grammar Rule 118 as an example (see listing 1). To expand *S*0, simply replace every occurrence of `118` with `271311686, 847213017, 847213017, 271311686`.

4.2.1 Visualization of sequence information

Mapping grammar rule numbers and connection event IDs to unique colors, leads to a natural visualization. For example The full expansion for rule 118 can be visualized by overlaying

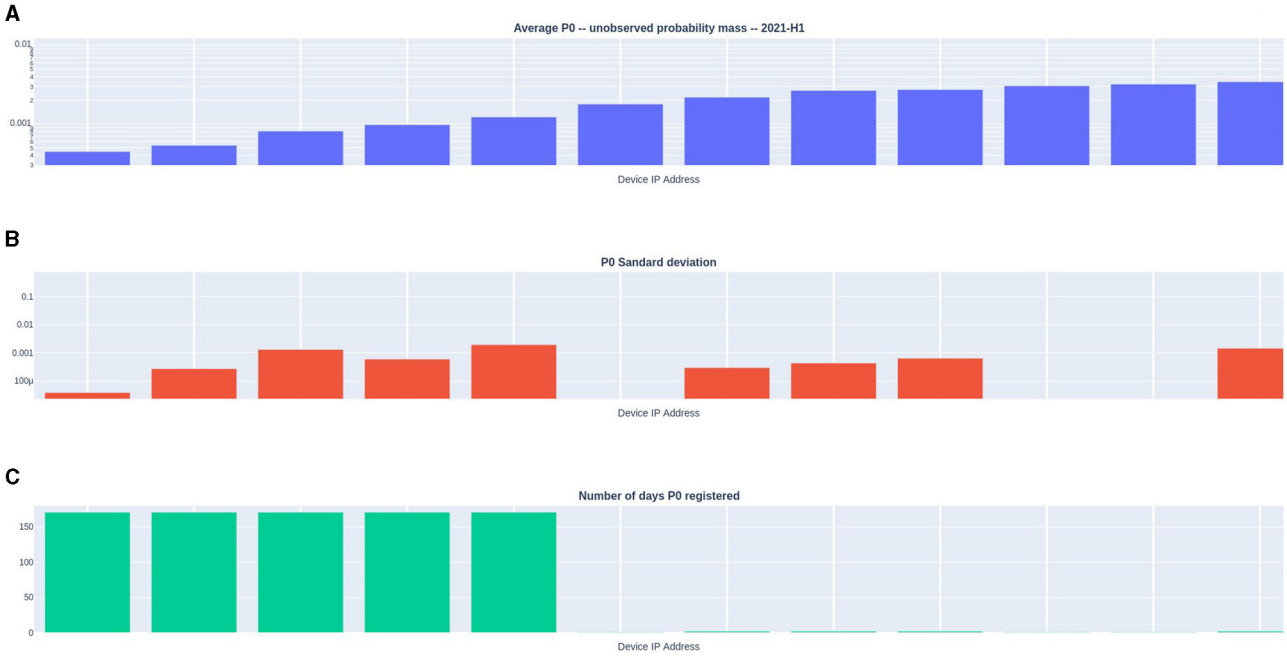

FIGURE 7
Simple Good-Turing frequency analysis summary of individual internal IP addresses observed daily. Rank ordered by mean P0 value for the period Zoomed in on the lowest 12 ranked P0 IP addresses. **(A)** blue bars – the period average $P0_{conn}$ each bar represents a single `id.orig_h` (e.g., an IP address); **(B)** red bars – the period standard deviation of $P0_{conn}$; **(C)** green bars – the number of days the device was operating. Note that the **(A, B)** are log scales, while the **(C)** is linear scale.

the colored *tiles* for IDs 271311686 and 847213017, on the rule color tile rectangle for Rule 118 as depicted in Figure 9. A simple tool was developed for visualizing sequences and the grammar rules using such a color tiling scheme.[17] It provides a way of browsing the original input sequence, guided by information overlaid from the SEQUITUR sequence extraction and some other sources.

In studying the *sequence* of events we drop the timestamps and focus only on the *ordering* of events.[18] Having discarded absolute temporal information and relied on event ordering to produce a visualization, we incorporate some adornments to convey some absolute temporal information. These include (1) to identify grammar rule sub-sequences that recur at the same relative time, we add a simple black strike-through marker to the visualization to indicate which is the first token tile to occur after the a new hour; (2) to indicate a beginning of a group of events that are clustered[19] within a pre-defined tolerance with a red underline. These features are shown in Figure 10.

17 The prototype has been coded in the *XPCE* (Wielemaker and Anjewierden, 2002) graphical extensions to *swi-prolog* (Wielemaker et al., 2010).

18 Dropping timestamps for sequence visualization purposes is also used in Nguyen et al. (2018).

19 The algorithm [from tyrex (2021)] has a single parameter ϵ which is used for identifying clusters in one-dimension.

4.3 Sequence grammar induction discussion

The use of SEQUITUR for the sequence mining of network connection events was successfully incorporated into a visualization enabling a human to browse and identify repeating sequences generated by a low volatility IoT-like device. The internal hierarchical grammar induced by SEQUITUR has been helpful. However, we have not yet *explained* any of the recurring sequence patterns, but it is a key step in the behavior cloning process which is completed in the logical inductions that follow in Section 5.

5 Logical XAI for cyber defense

5.1 Logic for human-computer interaction and reasoning

Mathematical logic has a long history including being used by the pioneers of AI (McCarthy, 1959; Robinson, 1965; Barrett and Connell, 2005). Indeed, when it comes to a strategy for building a *Thinking Machine* Turing sees logic as a fundamental component (Turing, 1950). Logic provides a way of explicitly encoding knowledge with well defined semantics. A form of logic primarily for knowledge sharing and reasoning has been used for more than 20 years to power the WC3's *Semantic Web* (Koivunen and Miller, 2001).

The forms and power of reasoning vary. Most often the form of logical reasoning used is *deduction*, being a relatively

FIGURE 8
Simple Good-Turing frequency analysis summary of internal IP addresses 10.0.0.8 (high P0 values) and 10.0.0.145 (low P0 values)observed daily. Blue line – device with IP address `10.0.0.145`, Red line – device with IP address `10.0.0.8`. The P0 axis is linear (not logarithmic as in some other figures).

efficient and safe way of drawing conclusions from known trusted information. Deductions take *general laws* and *cases* to derive sound *results*. However humans are able to reason with incomplete, and sometimes contradictory evidence, more akin to *abductive* reasoning. Abductions take known *general laws* and *results* to speculate on unknown *cases*. Abductive Logic, and its implementations (e.g., Arias et al., 2018) are beginning to make contributions to cyber defense research (Moyle et al., 2023a).

5.2 Machine learning in logic: inductive logic programming

A third form of reasoning, *induction*, takes as input *cases* and *results* and hypothesizes *general laws*. Inductive Logic Programming (ILP) (Muggleton and Raedt, 1994) is a form of symbolic machine learning where all aspects of the learning problem are specified and modeled as fragments of logic programs—typically as `Prolog` programs. The resulting models are *comprehensible* for humans. This makes the approach well suited to interactions with domain experts in their quest to explain some phenomena.

A simple tutorial illustration demonstrates how Inductive Logic Programming differs from Deductive Logic Programming is shown in Figure 11. Background knowledge encoded in logic (in the example these are known relationships in the family domain; but in general can be anything); examples are the positive/negative examples to be explained (also in logic) (in the example there are two positive examples of the relationship grandfather—and no negative examples); the hypothesis or explanatory model (to be generated as a fragment of logic). The outputs (models), being in logic, are easily comprehensible. ILP, however, is poor in domains with numerical or probabilistic information, and it is computationally expensive making the use of very large datasets problematic. It is normally used in a batch mode [refer to Ko (2000) and Moyle and Heasman (2003) for batch mode ILP in cyber security domains].

In this research we use Srinivasan's `ALEPH` ILP system (Srinivasan, 2001) as it provides an extensive set of features and capabilities. It requires configuration parameters to be specified that relate to the specific machine learning context and task. Normally, this process is done in a batch fashion, but in this work, an interaction with a domain expert, in which they constantly feed in more guidance and expertise as the search for an explanation that is acceptable to them converges. `ALEPH` has some support for such an interactive usage mode, but extended features are provided by the `ACUITY` wrapper system (Ray et al., 2016; Ray and Moyle., 2021). The formation of the explanation is a **co-discovery** with the ILP system being guided by a (cyber security network forensics) domain expert. For such an interaction between human and machine, it is

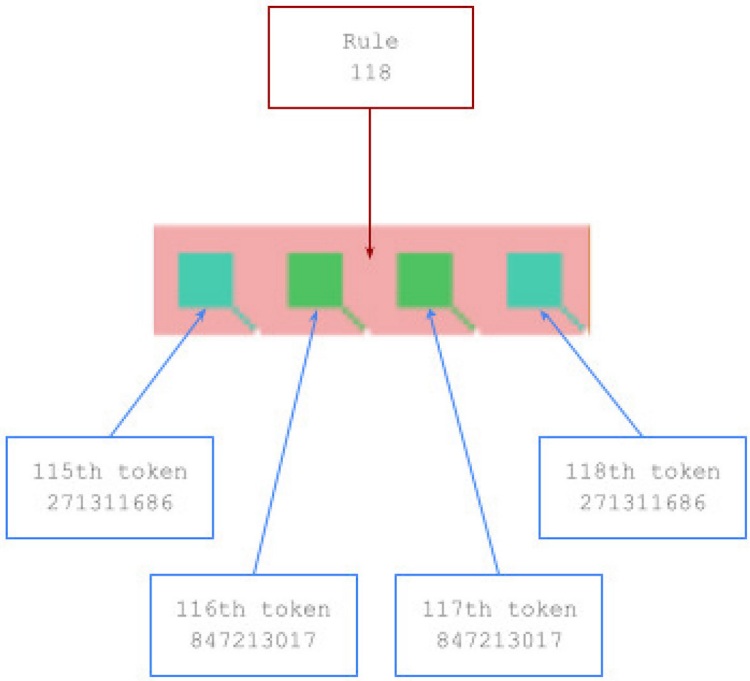

FIGURE 9
Simple SEQUITUR sequence visualization for example grammar rule 118. The backgound color pink is specific to the grammar rule, while the colors of the square tiles are unique to each of the underlying network connection IDs (or tokens). The left-to-right order of the tokens is from the original input sequence.

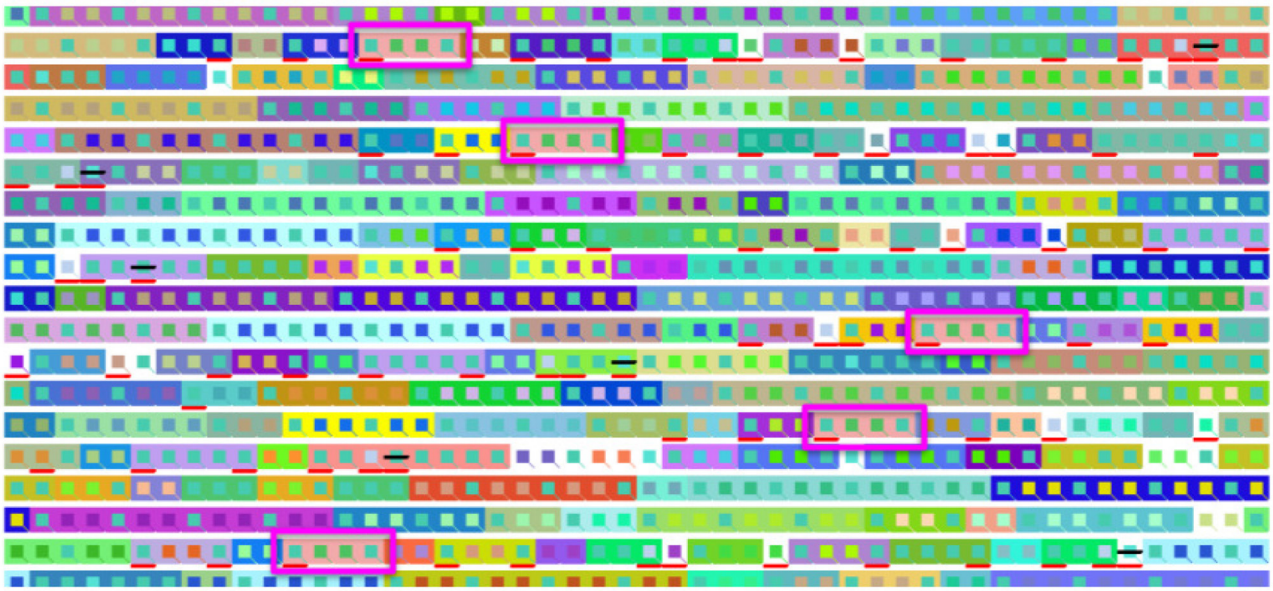

FIGURE 10
A simple SEQUITUR-based sequence visualization. Individual event tokens are represented by colored squares. The sequence of events flows left-to-right, top-to-bottom. Underlayed on the event tokens are colored rectangles representing recurring sub-sequence patterns extracted by SEQUITUR rules. Tool tips (not shown) provide detail information for selected event tokens. Adornments relating to absolute timestamps are black strike-throughs (e.g., which token is nearest to the start of an hour). Red underlines denote the start of a cluster of events. Note that the purple highlighted borders are manually drawn, to show the occurrences of rule 118 in the context of the the input sequence of tokens. The events are from 00:00 UTC to approximately is 06:00 UTC. Each instance of rule 118's sub-sequence is in its own cluster ($\epsilon = 5.0$ seconds), and *not* at the top-of-the-hour.

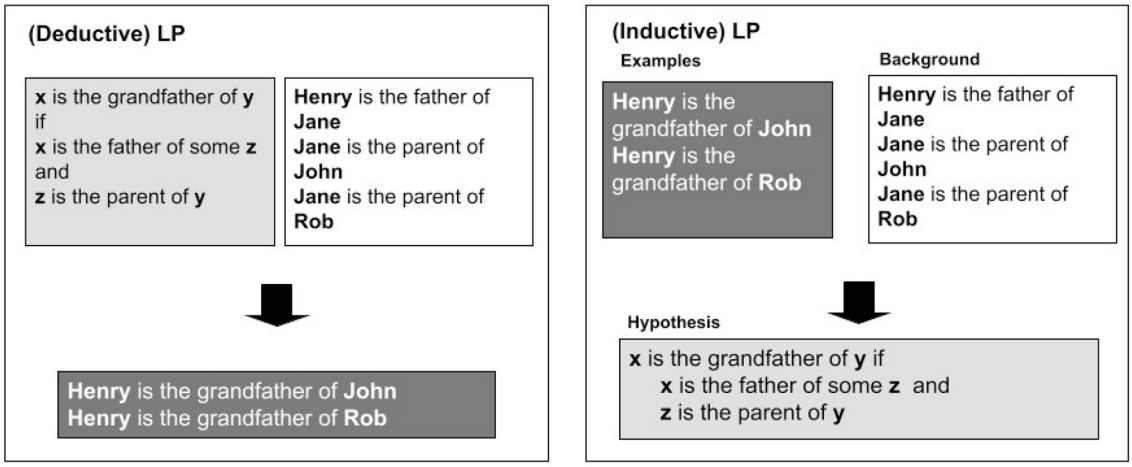

FIGURE 11
Tutorial illustration of Inductive Logic Programming. The **left**-hand panel – *(Deductive) LP* – depicts the normal deductive usage of Logic Programming where sound conclusions are derived from general rules about the grandfather relationship and specific facts. The **right**-hand panel – *(Inductive) LP* – depicts the (potentially unsafe) production of a general law about the grandfather relationship, produced from specific examples of grandfathers and other associated background facts.

important that the both parties can comprehend (Michie, 1988) any intermediate results, and react to what has been produced in a way so as to refine the outputs into an *acceptable*[20] explanatory theory.

An outline of the interactive ILP process is:

1. Assemble data, background knowledge, and settings (as fragments of a Logic Program).
2. **Select** a specific exemplar to be explained.
3. Let `ALEPH` build a most specific clause and search the lattice with respect to the input data, background knowledge, constraints, and settings – outputting a *proposed explanation* as a single `Prolog` rule.
4. **If** the *proposed explanation* is acceptable, **then** remove all the other examples that are also explained by the acceptable explanation, **otherwise** the human communicates to `ACUITY` that the *proposed explanation* is unacceptable, and makes changes to settings and other constraints, then repeat step 2.
5. **If** more examples exist go back to step 2 and select another unexplained exemplar; **otherwise** output the collection of all rules and other knowledge accumulated in the session as the *explanatory theory*.

5.3 Experiment to logically explain NetMon data

Having identified a single IoT-like device operating within the operational network (see Section 3.3), we wish to use a Human-XAI approach in the explanation of a sequence of related network events (see: Section 4.2.1) that can be incorporated into a (partial) specification of what the device *normally* does. Often, a key challenge for securing IoT devices is first identifying their presence on the network (as asset registers do not exist, or they have not been properly maintained). The analysis of the data in Section 3.3 identified that the *volatility* of network connections to/from a device provides an indication, relative to other devices on the same observed network, of whether it belongs to the class of *IoT-like* devices. From that analysis the device with IP address `10.0.0.145` had low volatility and was selected for further the explanatory analysis.

5.3.1 `ACUITY` Data preparation

The data transformation process is depicted in Figure 3. The following summarizes the process for building the inputs to the XAI machine learning system `ACUITY`. It takes as a precursor, the data transformations described in Section 4.1:

- Use the sequence visualization to identify interesting (repeating) subsequences as examples for XAI machine learning.
- The following items are then utilized by the XAI machine learning system:
  - The raw Zeek `conn`ection table event records;
  - The extracted Zeek `dns` and `dhcp` table event records;
  - The sequence pattern data extracted by the `SEQUITUR` system (see Section 4.2)
  - The hierarchical grammar and supporting data elements (from `SEQUITUR`).
  - `Prolog` code providing logical relationships between entities.

A noteworthy inclusion of cyber defender expertise into the XAI background knowledge relates to the concept of *beaconing*. This is a common design pattern in IoT-like devices that regularly

20 We do not provide a strong definition of acceptability here, only that the domain expert is satisfied with the explanation arrived at.

contact a central service for any pending commands[21]. A fragment of Prolog was coded to determine if there was a time around which repeating sequences clustered with respect to the minute hand on the clock. This specification was made available in the learning process. Detailed Data Understanding and Data Preparation information can be found in Moyle et al. (2023b).

5.3.2 Human-Machine XAI explaining a SEQUITUR sequence

This section outlines a proof-of-principle experiment using ACUITY to explain one of the subsequences extracted by SEQUITUR from network monitoring data of the device at IP address 10.0.0.145 on 2021-01-01. The exemplar under study is grammar Rule 8 which is a sequence of events containing four elements with the token hashes [363783709, 100233664, 100233664, 363783709], and occurs 18 times in the twenty-four hour period under study.

Following the process outlined in Section 5.2 leads to a co-discovery conversation between the cyber defense expert and the machine learning system in which seven interactions occur: an example to be explained is selected[22]; the first suggested explanation is rebutted; the expert defines an over-general constraint; a counter example is added to refute another explanation; constraints are added based on what the expert finds most interesting (twice); and the explanation (known as the theory–see listing 2) is accepted. For complete details see Moyle et al. (2023b).

```
[theory]

[Rule 1] [Pos cover = 1 Neg cover = 0]
exp_seq(A) :-
   rule_start_zeek_uuid(A,B), next_to(B,C), next_to(C,D
     ),
   has_port_service(D,ssh),
   has_dest_name(D,'ec2-3-209-99-56.compute-1.amazonaws
     .com'),
   beaconing_every_hour(A,10).


[positives]
exp_seq(8).

[negatives]
exp_seq(sk_1_1_rule_id).
exp_seq(1).


true .
```

Listing 2 Explanatory Theory of grammar rule 8 for the IoT-like eavesdropping device learned with ACUITY.

The theory to explain the sequence that is generated starting from the single example of SEQUITUR grammar rule 8 consists of the single positive example (exp_seq(8)), two expert introduced examples ("this is not an example of a grammar rule 8 sequence"), and a single (ALEPH rule). This rule can be directly translated from the Prolog rule in listing 2 into English:

> The sequence of events is explained if: the sequence grammar rule starts with a Zeek UUID (*B*) and that the next event after *B* is *C* and the next event after *C* is *D*. The event *D* connects to port service ssh at the server 'ec2-3-209-99-56.compute-1.amazonaws.com', at the beaconing frequency of 10 minutes past the hour.

5.3.3 Evaluation

We are fortunate to have access to the device manufacturer of the device instance we are studying. The device developers confirm that the device class has code that attempts to establish a connection to the named cloud host every 10 minutes past the hour, as shown in the implemented code fragment for the device's *cron* scheduler in listing 3.

```
    :
*/10 * * * * /path/to/ai-aws-ssh.sh >> /var/log/ai/aws-
    ssh.log 2>&1
    :
```

Listing 3 Beaconing ssh connection to the cloud-based controller in the cron scheduler of device 10.0.0.145.

We further check the coverage of the final theory with respect to the data for the entire day, in which we find that both SEQUITUR grammar rules 8 and 191 are predicted by the final theory. Inspection of grammar rule 191 shows that it is an extension of rule 8 (and is probably produced by an over specification[23] by the compression algorithm).

The cyber defender is sufficiently content in the explanation that they have co-constructed with the XAI system in that (1) it fits their broad understanding and experience; (2) it covers other unnoticed instances; and (3) it does not false alarm. They are also relieved that they have not had to spend much effort in studying detailed logs, and the extensive searching for explanations has been done by the machine.

6 Discussion

Cyber defenders are deluged by data that lacks labels to make machine learning and other artificial intelligence techniques a challenge to apply. Scarce human defenders themselves can be significantly enhanced if they can team up with an XAI that amplifies their skills. This requires both the Human and Machine can share with each other what they know and what it is that they have newly determined. In this research we have used network monitoring meta data as a first source of data. This data is transformed and filtered through a *volatility* measure, which provides the ability to identify both periods of operation as well as individual devices that are easier to defend.

It can be difficult for academic cyber security researchers to obtain real world data. For this research it was possible to utilize approximately six months of operational network data from a modest sized organization. 185 individual devices were observed during the study period. Although the organization had a variety

21 Malware sometimes use the same design pattern for which it is part of their *Command and Control* (or C2) system.

22 The most specific rule based on the inputs and the selected example is 57 literals long, defining the size of the search space for single clauses to be 2.40×10^{32}.

23 We prefer over specialization by the distribution free SEQUITUR system so that the generalizations can be produced by ACUITY.

of devices there was only one fully identified device, the network monitoring system that was installed for this observational case study.

In Section 3 we showed how the *Simple Good-Turing frequency estimator* unobserved probability mass ($P0$) can be applied to the eavesdropped network metadata. Gross patterns of daily network traffic show periods of less volatility (and there is a strong correlation to periods when traffic and volatility can be expected to be relatively less—e.g., on non-workdays). Average daily network volatility varies substantially throughout the 172 day period with a minimum value of 6.1 microns and a maximum value of 331 microns. In the same period, the number of observed devices varies between 28 and 185, whilst the total number of observed network connections in a one-day period varies from 63,710 to 561,742. The ratio of Originating device connections to Responding device connections is about 1 to 40, and that these two sources of $P0$ are not strongly correlated.

A spectrum of volatility exists in real device behavior, giving initial support that it is possible to identify easier-to-defend devices. Without physical access to the devices *post-hoc* identification of network devices is difficult. The analysis of the lowest ranked five devices *suggests* these are devices with repetitive simple behaviors. Unsurprisingly, the network monitoring system itself is a low volatility device, being the third lowest ranked device in the survey. As this device is fully specified and identified in advance, it allows any analyses to be compared to its designed IoT behavior, making it a good target for further detailed study.

The `SEQUITUR` compression algorithm is used to perform an initial phase of *behavioral cloning* to an arbitrary day's worth of metadata traffic containing 3,613 network `conn`ection events. `SEQUITUR` is used as a machine learning algorithm as its internal basis records a *hierarchical grammar*. This consists of grammar rules that encode repeating sequences. The whole original sequence can be visualized using a colored tiling that encodes both the individual events and the repeating sequences that are a component of. The `SEQUITUR` algorithm does not take account of the timestamps, only the sequence of events. This is similar to approaches taken elsewhere also in applied security domains (e.g., (Nguyen et al., 2018)). However, adding back some information into the sequence visualization is helpful: (1) marking the beginning of each hour; and (2) clustering the events along the time dimension using the original timestamps. The human cyber defender can use the visualization to identify some (possibly recurring) sub-sequences that they want to explore more deeply. Each occurring instance of an identified becomes an *example* that can be used for the defender to give to the XAI.

The final phase of generating a behavioral cloning understanding of specific device behavior is performed in Section 5 using the interactive inductive logic programming system `ACUITY` (see Ray and Moyle., 2021). This takes as input the example of the sequence of behavior to be explained, and contextual background knowledge. These inputs are all expressed as fragments of Logic Programs, which are comprehensible to the Human defender, and directly executable by the XAI [which conforms to Michie's description of an *ulta-strong machine learner* (Michie, 1988)]. The defender is able to directly refute the XAI's working hypotheses when needed, to shape an explanation that sufficiently fits both the observed data and tacit expertise of the defender.

The result of theHuman-XAI collaboration session explains the `SEQUITUR` rule 8 pattern from a single example, with one negative example added by the defender, and one negative example inferred through the defender's rebuttal process. The explanation includes the defender's high-level concept of *beaconing* which is a common design pattern used in some IoT devices where they periodically contact a master controller for any pending commands at some regular frequency. It is rendered in English, but has a direct representation as a fragment of `Prolog`, which can be used to reason with, or even deploy as an allowable behavior of the device that can be permitted on the network at run time. This has some similarities logical induction of intrusion detection system rules reported in Ko (2000), but our approach does not require significant, pre-labeled balanced datasets.

Revisiting Michie's version of eXplainable AI which he calls *Ultra-strong Machine Learning* the following tests must be met:

- **[M1]** *[a] system [that] uses sample data (training set) . . .*
- **[M2]** *. . . to generate an up-dated basis for improved performance on subsequent data . . .*
- **[M3]** *. . . and also can communicate its internal updates in explicit . . .*
  - **[M3.1]** *. . . and operationally effective . . .*
  - **[M3.2]** *. . . symbolic form*

The results reported here were generated from a system that used sample data, which originated as network traffic, and was identified and selected by being filtered through volatility, then extracted as recurring sequence patterns. This satisfies Michie's **[M1]** requirement. The induced rule, mentioning beaconing, updated basis was applied to a randomly selected day of that devices network logs, and found that the behavior appears 24 times at the hourly time stamps forecast. This, then satisfies the **[M2]** requirement of improved performance, as without the rule, the system would not be able to identify the behavior in unseen data. Dealing with the combined **[M3/M3.2]**—the cyber defender was able to verify the beaconing rule (indeed they helped create it) in the symbolic natural language formulation above, thus demonstrating that the system was able to communicate its internal update to a human. Finally, the combined **[M3/M3.1]** we see that we can directly utilize the isomorphic `Prolog` form of the beaconing rule, in both identifying the behavior in subsequent network logs, and also as part of a more fine-grained defensive system. Furthermore, we can add this operationalization to the manufacturer's specification for that device class (see Grayeli and Mulugeta, 2020, ManySecured, 2021) for use in other deployment contexts.

Although we have demonstrated XAI Human-Machine collaboration applied to network security, it still falls short of replacing human cyber defenders. The results above utilize restricted reasoning that is limited to deduction and some induction. Cyber defenders constantly form hypotheses, collect more evidence, and focus their investigations. They will use *abductive* reasoning (Peirce, 1996) to form hypotheses and seek

yet-to-be discovered evidence, whilst forming constraints based on the potentially contradictory evidence that will rule out incompatible hypotheses and lines of enquiry. Once the case is solved, then it is possible to follow a *deductive* reasoning process that explains the case and its artifacts. Abductive reasoning is difficult to achieve using standard logic programming techniques (e.g., like `Prolog`). Advances in *Answer Set Programming* (Lifschitz, 2022)) provide for environments that describe their world in a comprehensible form (e.g., logic) whilst permitting missing and or contradictory evidence. Some initial explorations have been made in the field of cyber security in the `s(CASP)` environment (Moyle et al., 2023a). Providing high-level symbolic interaction in natural language continues to be explored in the `s(CASP)` system (see Sartor et al., 2023).

7 Conclusion and future work

To center this research clearly in the realms of eXplainable AI we have chosen to align with Donald Michie's definition of *Ultra-strong Machine Learning* (Michie, 1988). As his definition insists at the outset on comprehensible internal models that are learned by the AI there is no need to reverse-engineer (Gunning et al., 2021) any learned black-box model to understand what it might do when making predictions. Systems that perform within Michie's definition could be termed *Self Explanatory AI*.

Supported by an empirical case study we demonstrated that passively collected Network Monitoring event data can be used to identify the volatility of communications of network devices. It provides further evidence to motivate a systematic approach to the identification of individual devices' network traffic volatility as provided by the Simple Good-Turing frequency estimator (Good, 1953; Gale and Sampson, 1995). We have shown with empirical evidence that there is a spectrum of easier-to-defend through to harder-to-defend network devices. This supports Meer's conjecture (Meer, 2015) that it is very difficult to defend whilst much easier to attack.

The case study has also demonstrated that the symbolic compression algorithm, `SEQUITUR` (Neville-Manning and Witten, 1997), can be used as a sequence pattern extractor from a stream of temporal network events. Furthermore, an elementary method for visualizing the sequence patterns provides a basic human-computer interface for navigating the network data. Humans can explore the network event interaction sequences via visualizations, and select targets for further investigations

The symbolic compression techniques can be used as a form of behavioral cloning (Bain and Sammut, 1995) by extracting hierarchical grammars to describe sequences of network interactions. The grammars provide a component of background knowledge that can be deployed in an interactive inductive logic programming system (`ACUITY`, Ray et al., 2016). Further rich knowledge about the cyber security domain and specific network security context can be added. For example, high-level concepts that cyber defenders understand, like the behavior pattern of device beaconing, can also be incorporated into the discourse. We demonstrated that a cyber defender can have a robust comprehensible direct interaction with the XAI machine learning environment to guide it toward a mutually understood explanation of the sample network phenomenon selected from the event logs.

The resulting knowledge produced was compared with the ground-truth knowledge from the generating device's actual behavior and that was found to be a consistent explanation. Such knowledge could be deployed in an operational role with high confidence to improve individual device security and security of the network.

The XAI systems used for this research are still very rudimentary, they do not yet fulfill the complete standard expected by Srinivasan et al. (2022) in which an XAI environment should to be two-way, requiring the XAI to be further capable of refuting the hypotheses of their human collaborator.

7.1 Future work

There are numerous directions that can be pursued from this research ranging from empirical (e.g., more analysis of the existing data), to theoretical (e.g., how robust is SGT and how does it compare to other volatility measures?).

Is it possible to use the visualization of tilings in other ways? In Axon (2018) sounds are mapped to network traffic data for the purpose of network-security monitoring. The abstraction of data used in Axon (2018) is low level. A higher level of abstraction could be used for sonification by using analogs of the tokens and the token map visualization. For example each rule color (i.e., number) and each token color (token ID) can be mapped to particular tones and perhaps particular instrument timbres.

The proof of concept described has used data from on particular day from a single device. It is claimed that `SEQUITUR` (Neville-Manning and Witten, 1997) is particularly effective when working with longer sequences. The current tooling for cyber defenders is crude and there are many plausible enhancements. The visualizations of the `SEQUITUR` outputs from sequence mining the network event data could be integrated into the `ACUITY` workflows to enhance the ability to explain most, if not all, sequence patterns produced by a single device. Improved user interfaces of the type being explored here (Deane and Ray, 2023) may be beneficial.

We have not yet turned the new knowledge into an widely usable active defense. Having produced behavior explanations, can these be codified and exported in a standard form? Existing standards for IoT behavior specification include *D3* (ManySecured, 2021) and *MUD* (Grayeli and Mulugeta, 2020).

The system has a restricted power of reasoning as it is using standard logic programming deductive techniques which are limited to simple deductions appropriate when there is one single consistent logical *model*, whilst coping with inconsistencies and missing data are common in human reasoning. Using developments in *Answer Set Programming* (Lifschitz, 2022) and *s(CASP)* (Arias et al., 2018) for a more comprehensive reasoning basis will keep the formalism human comprehensible, and extend the power.

The above list is not exhaustive, but provides some future research and development directions. For a more detailed list see Moyle et al. (2023b).

Data availability statement

The raw data supporting the conclusions of this article will be made available by the authors, without undue reservation.

Author contributions

SM: Conceptualization, Data curation, Investigation, Methodology, Software, Visualization, Writing – original draft, Writing – review & editing. AM: Project administration, Writing – review & editing. NA: Funding acquisition, Project administration, Resources, Writing – review & editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This research was funded in whole or in part by Innovate-UK 105592: CyberStone: Collaborative Secure IOT Gateway (ManySecured).

Acknowledgments

The authors gratefully acknowledge the feedback provided by the reviewers. SM gratefully acknowledges Ashwin Srinivasan for his long support and inspiration for Figure 11.

Conflict of interest

NA was employed by NquiringMinds.

The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Arias, J., Carro, M., Salazar, E., Marple, K., and Gupta, G. (2018). Constraint answer set programming without grounding. *Theory Pract Logic Progr.* 18, 337–354. doi: 10.1017/S1471068418000285

Axon, L. (2018). *Sonification for network-security monitoring*. PhD thesis, University of Oxford.

Bacon, F., Urbach, P., and Gibson, J. (1996). Novum organum. *Br. J. Philos Sci.* 47, 125–128. doi: 10.1093/bjps/47.1.125

Bain, M., and Sammut, C. (1995). "A framework for behavioural cloning," in *Machine Intelligence*, 15.

Barrett, L. and Connell, M. (2005). Jevons and the logic 'piano'. *Rutherford J.* 1, 2005–2006.

Deane, O., and Ray, O. (2023). "Interactive model refinement in relational domains with inductive logic programming," in *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces, pages* 127–129. doi: 10.1145/3581754.3584150

Finn, P., Muggleton, S., Page, D., and Srinivasan, A. (1998). Pharmacophore discovery using the inductive logic programming system progol. *Mach. Learn.* 30, 241–270. doi: 10.1023/A:1007460424845

Flach, P. (2012). *Machine Learning: The Art and Science of Algorithms That Make Sense of Data*. Cambridge, USA: Cambridge University Press. doi: 10.1017/CBO9780511973000

Frank, E., Hall, M. A., and Witten, I. H. (2016). *Data Mining: Practical Machine Learning Tools and Techniques* (Fourth Edition). Burlington, MA: Morgan Kaufmann.

Gale, W., and Sampson, G. (1995). Good–turing frequency estimation without tears. *J. Quant. Ling.* 2, 217–237. doi: 10.1080/09296179508590051

Gillies, D. (1996). *Artificial Intelligence and Scientific Method*. Oxford and New York: Oxford University Press. doi: 10.1093/oso/9780198751588.001.0001

Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika* 40, 237–264. doi: 10.1093/biomet/40.3-4.237

Grayeli, P., and Mulugeta, B. (2020). "Mitigating network-based attacks using mud-improving security of small-business and home iot devices," in *RSA Conference*, 41–50.

Gunning, D., Vorm, E., Wang, J. Y., and Turek, M. (2021). Darpa's explainable ai (xai) program: a retrospective. *Appl. AI Lett.* 2:e61. doi: 10.1002/ail2.61

(ISC)2 (2022). *Cybersecurity Workforce Study*. Available online at: https://media.isc2.org/-/media/Project/ISC2/Main/Media/documents/research/ISC2-Cybersecurity-Workforce-Study-2022.pdf?rev=1bb9812a77c74e7c9042c3939678c196 (accessed May 03, 2024).

ISO/IEC (1999). *Common Criteria for Information Technology Security Evaluation. Common Criteria Project Sponsoring Organisations*. Version 2.1, adopted by ISO/IEC as ISO/IEC International Standard (IS) 15408 1-3. Available online at: https://csrc.ncsl.nist.gov/cc/ccv20/ccv2list.htm (accessed May 03, 2024).

Jevons, W. (2012). *The Principles of Science: A Treatise on Logic and Scientific Method*. Washington, DC: Ebsco PsychBooks.

Khraisat, A., Gondal, I., Vamplew, P., and Kamruzzaman, J. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. *Cybersecurity* 2, 1–22. doi: 10.1186/s42400-019-0038-7

King, R., Muggleton, S., Srinivasan, A., and Sternberg, M. (1996). Structure-activity relationships derived by machine learning: the use of atoms and their bond connectivities to predict mutagenicity by inductive logic programming. *Proc. Natl. Acad. Sci. USA*. 93, 438–442. doi: 10.1073/pnas.93.1.438

Ko, C. (2000). "Logic induction of valid behavior specifications for intrusion detection," in *2000 IEEE Symposium on Security and Privacy, Berkeley, California, USA, May 14-17, 2000* (IEEE Computer Society), 142–153.

Koivunen, M.-R., and Miller, E. (2001). "W3C semantic web activity," in *Proceedings of the Semantic Web Kick-off Seminar in Finland*. Available online at: https://www.w3.org/2001/12/semweb-fin/w3csw (accessed May 03, 2024).

Kowalski, R. (2011). *Computational Logic and Human Thinking: How to Be Artificially Intelligent*. Cambridge: Cambridge University Press. doi: 10.1017/CBO9780511984747

Lifschitz, V. (2022). *What Is Answer Set Programming*? University of Texas. Available online at: https://www.cs.utexas.edu/users/vl/papers/wiasp.pdf (accessed May 03, 2024).

Lloyd, J. (1984). *Foundations of Logic Programming*. Cham: Springer-Verlag. doi: 10.1007/978-3-642-96826-6

ManySecured (2021). *Distributed Device Descriptors (D3)*. Available online at: https://specs.manysecured.net/d3/ (accessed May 03, 2024).

McCarthy, J. (1959). "Programs with common sense," in *Proceedings of the Teddington Conference on the Mechanization of Thought Processes* (London: Her Majesty's Stationary Office), 75–91.

Meer, H. (2015). "What got us here wont get us there," in *Black-Hat Europe*.

Megas, K., Fagan, M., Marron, J., Watrobski, P., and Cuthill, B. (2022). *Profile of the iot core baseline for consumer iot products*. Internal report, National Institute

of Standards and Technology, Gaithersburg, MD. NIST Interagency/Internal Report (NISTIR).

Michie, D. (1988). "Machine learning in the next five years," in *Third European Workshop Session on Learning (EWSL '88)*, eds. D. Sleeman, and J. Richmond (London: Pitman), 107–122.

Moyle, S., Allott, N., and Manslow, J. (2023a). "Modelling cyber defenses using s(casp)," in *Proceedings of the International Conference on Logic Programming 2023 Workshops co-located with the 39th International Conference on Logic Programming (ICLP 2023), London, United Kingdom, July 9th and 10th, 2023, volume 3437 of CEUR Workshop Proceedings. CEUR-WS.org*.

Moyle, S., and Heasman, J. (2003). "Machine learning to detect intrusion strategies," in *Knowledge-Based Intelligent Information and Engineering Systems*, eds. V. Palade, R. J. Howlett and L. Jain (Berlin, Heidelberg: Springer), 371–378. doi: 10.1007/978-3-540-45224-9_52

Moyle, S., Martin, A., and Allott, N. (2023b). *Identifying iot-like devices and using collaborative xai to understand their cyber security behaviour*. Technical report. Available online at: https://ora.ox.ac.uk/objects/uuid:ae1353c9-97c8-4356-9f17-73c2633a740f (accessed May 03, 2024).

Muggleton, S. (2014). Alan turing and the development of artificial intelligence. *AI Commun.* 27, 3–10. doi: 10.3233/AIC-130579 (accessed May 03, 2024).

Muggleton, S., and Raedt, L. D. (1994). Inductive logic programming: theory and methods. *J. Logic Progr.* 19, 629–679. doi: 10.1016/0743-1066(94)90035-3

Muggleton, S. H., Schmid, U., Zeller, C., Tamaddoni-Nezhad, A., and Besold, T. (2018). Ultra-strong machine learning: comprehensibility of programs learned with ilp. *Mach. Learn.* 107, 1119–1140. doi: 10.1007/s10994-018-5707-3

Neville-Manning, C., and Witten, I. (1997). "Linear-time, incremental hierarchy inference for compression," in *Data Compression Conference (DCC '97)*, 3–11.

Nguyen, P., Turkay, C., Andrienko, G., Andrienko, N., Thonnard, O., and Zouaoui, J. (2018). Understanding user behaviour through action sequences: from the usual to the unusual. *IEEE Trans Visual. Comput. Graph.* 25, 2838–2852. doi: 10.1109/TVCG.2018.2859969

NISTN. (2021). *Four principles of explainable artificial intelligence*. Technical Report NIST IR 8312, U.S. Department of Commerce, Washington, D.C.

OpenVAS. (2023). *OpenVAS - Open Vulnerability Assessment Scanner*. Available online at: https://openvas.org/ (accessed May 03, 2024).

Peirce, C. (1996). *Elements of logic*. Collected papers of Charles Sanders Peirce/Peirce, C.

Popper, K. R. (1935). *The Logic of Scientific Discovery*. London, UK: Routledge.

Ray, O., Hicks, S., and Moyle, S. (2016). "Using ilp to analyse ransomware attacks," in *26th International Conference on Inductive Logic Programming (ILP)*, *CEUR 1865*, 54–59.

Ray, O., and Moyle., S. (2021). "Towards expert-guided elucidation of cyber attacks through interactive inductive logic programming," in *13th International Conference on Knowledge and Systems Engineering (KSE)* (IEEE Press). doi: 10.1109/KSE53942.2021.9648769

Robinson, J. A. (1965). A machine-oriented logic based on the resolution principle. *J. ACM* 12, 23–41. doi: 10.1145/321250.321253

Sartor, G., Dávila, J., Fidelangeli, A., and Pisano, G. (2023). "(re)integration of logical english and s(casp)," in *Proceedings of the International Conference on Logic Programming 2023 Workshops co-located with the 39th International Conference on Logic Programming (ICLP 2023), London, United Kingdom, July 9th and 10th, 2023, volume 3437 of CEUR Workshop Proceedings. CEUR-WS.org*.

Schwitter, R. (2020). "Lossless semantic round-tripping in peng asp," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, 5291–5293. doi: 10.24963/ijcai.2020/773

Srinivasan, A. (2001). *The Aleph Manual*. Available online at: https://web.comlab.ox.ac.uk/oucl/research/areas/machlearn/Aleph/ (accessed May 03, 2024).

Srinivasan, A., Bain, M., and Coiera, E. (2022). One-way explainability isn't the message. *arXiv preprint arXiv:2205.08954*.

Thompson, K. (1984). Reflections on trusting trust. *Commun. ACM* 27, 761–763. doi: 10.1145/358198.358210

Turing, A. M. (1937). On computable numbers, with an application to the entscheidungsproblem. *Proc. London Mathem. Soc.* 42, 230–265. doi: 10.1112/plms/s2-42.1.230

Turing, A. M. (1950). Computing machinery and intelligence. *Mind* 59, 433–460. doi: 10.1093/mind/LIX.236.433

tyrex (2021). *1D Number Array Clustering. Stackexchange question answered Jul 6 '21 at 13:02 by tyrex*. Available online at: https://stackoverflow.com/questions/11513484/1d-number-array-clustering (accessed May 03, 2024).

United States Department of Defense (1985). *Department of Defense, Trusted Computer System Evaluation Criteria*. Number 5200.28-STD in Rainbow Series. Dept. of Defense. doi: 10.1007/978-1-349-12020-8_1

Wielemaker, J., and Anjewierden, A. (2002). "An architecture for making object-oriented systems available from prolog," in *Workshop on Logic Programming Environments*.

Wielemaker, J., Schrijvers, T., Triska, M., and Lager, T. (2010). Swi-prolog. *Theory Pract Logic Progr.* 12, 67–96. doi: 10.1017/S1471068411000494

Wolfram, S. (2023). *What Is ChatGPT Doing ... and Why Does It Work?* Champaign, IL: Wolfram Media.

Zeek (2021). *The Zeek Network Security Monitor*. Available online at: https://zeek.org/ (accessed May 03, 2024).