On Machine Learning for Digital Forensics Investigation in Network Traffic

Andrea Tundis
Institute for the Protection of Terrestrial Infrastructures
German Aerospace Center (DLR)
Mornewegstrasse 30, 64293, Darmstadt, Germany
andrea.tundis@dlr.de

Francesco Cauteruccio
DIEM
University of Salerno
Fisciano, Italy
fcauteruccio@unisa.it

***Abstract*—Cybercrime is an ever increasing issue in the modern world. With the growing reliance of individuals, companies and countries on digital infrastructure, more people are exposed to potential attack vectors which cybercriminals can use to extort a ransom, steal data, commit fraud, or cause significant financial damage. To prevent such crimes from occurring, various security measures are being employed. One such measure is network forensics, which focuses on analyzing network traffic data to uncover evidence and information about attacks and detect intrusions. Network forensics has to deal with large, dynamic, and volatile data, which makes performing analysis a challenging task. Machine learning has been proposed to overcome some of the challenges associated with such analysis. This paper aims to give an overview of network forensics and machine learning, present some tools investigators use to perform network forensics, and introduce some results of recent research into the use of machine learning for network forensics. Finally, a brief discussion of current challenges and further research directions is provided.**

***Index Terms*—Digital Forensics Investigation, Network Traffic Analysis, Machine Learning, Artificial Intelligence, Cybersecurity.**

I. Introduction

As the digital world evolves toward the Internet of Everything (IoE), an ever-growing number of interconnected devices continuously exchange vast amounts of data. While this hyper-connectivity fosters innovation and efficiency across domains, it also expands the attack surface for malicious actors. Cyberattacks—ranging from Distributed Denial of Service (DDoS) to ransomware and botnet-based intrusions—can target individuals, corporations, and even national infrastructure. These attacks often manifest as anomalous network behavior, leaving behind digital footprints that can be leveraged for detection and investigation. Network forensics (NF) plays a critical role in this context. It involves the collection and analysis of network traffic from potentially compromised systems, aiming to identify ongoing threats, support post-incident investigations, and deter future attacks. Even when attribution is not possible, forensic activities can raise the cost of malicious operations by forcing adversaries to invest more in stealth. However, as the complexity and scale of IoE systems grow, traditional forensic techniques struggle to keep pace. The high volume, velocity, and dynamic nature of IoE data make manual or rule-based analysis increasingly infeasible. This challenge calls for modern solutions, particularly those based on Machine Learning (ML) and Artificial Intelligence (AI), to support or even automate parts of the forensic process [1].

The research activities related to this work are being conducted in the context of "urbanModel" and "Digitaler Atlas 2.0" projects that are funded by the German Aerospace Center (DLR) e.V. This activity has been carried out in cooperation with the LOEWE Zentrum emergenCITY.

In this paper, we provide a concise overview of network forensics and the tools commonly used in investigations, with a particular focus on AI-driven approaches. Particularly, Section II and III provides an overview about network forensic capabilities and tools within the IoE ecosystem. ML approaches are presented in Section IV, whereas a discussion on open research challenges and potential future directions for developing robust, scalable, and adaptive forensic solutions are discussed in Section V. Section VI concludes the work.

II. Background on Network forensics

NF is a subtopic of computer forensics that concerns with extracting digital forensic evidence from, for instance, hard drives, the operating system information of a user, or emails. This evidence can then be used to either prevent further attacks or to convict perpetrators of the cybercrimes they committed. It focuses on extracting evidence and information from network-related data, such as network traffic data, in order to (i) produce evidence, (ii) gather information of past attacks to introduce preventive measures, (iii) detect attacks while they are happening, (iv) to deploy countermeasures immediately. Generally, two types of network forensics exist [2], [3], [9]:

- *Catch-it-as-you-can*: A preemptive approach, where network traffic is continuously monitored and analyzed for anomalies or signs of a security incident.
- *Stop, look and listen*: A reactive approach, where traffic is only captured and analyzed after a potential incident has been detected.

Such distinction does not fundamentally change the process of network forensics, but only when and how data is collected

2325-2944/25/$31.00 ©2025 IEEE
DOI 10.1109/DCOSS-IoT65416.2025.00155

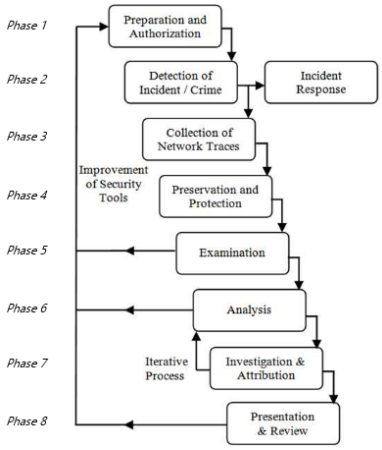

Fig. 1. Main phases of a network forensics framework [9]

and analyzed. An example of NF approach, based on the broader computer forensic methodology, is proposed [9]. Its defined phases are depicted in Figure 1. NF requires new steps to account for the distinct differences encountered when dealing with volatile network traffic data instead of data stored on hard drives. The methodology includes different phases, ranging from preparation, i.e., the necessary authorization to perform network forensics need to be obtained, and sensors or tools need to be initialized, to investigation, i.e., the obtained evidence is used to answer the central questions arising beyond the motivation of the task.

A. Machine Learning

Machine learning is a branch of artificial intelligence that enables computers to learn patterns and distributions from data to make decisions about problems without being explicitly programmed for them. Using statistical methods and modelling techniques, machine learning models can learn complex relationships and dependencies within data. One of the most frequent problems for which machine learning is used is classification, where the model has to categorize input data into predefined classes. Examples for classification problems include image recognition and text classification. A classification model is trained using supervised learning. This means that each data item in the training data is labelled with the class it belongs to. In addition to the label, each data item has so-called features, which are distinct properties of that data item. In network traffic data these may include, among others, packet size, protocol, source, and destination. By utilizing statistical and computational methods, the model then attempts to learn the relationship between the features and labels of all data items.

Researchers have identified machine learning as a useful method for network forensics, mainly for classifying traffic to detect attacks. The following algorithms are some of the most commonly used for classification:

- *K-nearest-neighbours*: Data items are grouped by the similarity of key features to generate distinct classes. For new items, the k most similar neighbours are computed, with k being a hyperparameter which can be fine-tuned to alter the performance of the model. The new data items is classified depending on the classes of its k nearest neighbours.
- *Support vector machines (SVMs)*: SVMs find the optimal hyperplane in the multidimensional feature space which divides two classes from each other by maximizing the margin between the hyperplane and the elements of a class. For more flexible classification, different kernel functions can be used, allowing for linear, polynomial, or other classifications.
- *Decision trees (DT)*: DT are binary trees where each internal node represents a decision made upon a feature value. New data items traverse the tree from the root to one of the leaves, which finally represent the class the new data item gets classified.

These algorithms can also be combined into so-called ensemble algorithms, which reduces some of their downsides. For instance, several decision trees can be combined into a random forest, which reduces overfitting and improves robustness, but also increases the computational load. Moreover, neural networks and deep neural networks can be used for solving classification problems. Neural networks consist of multiple layers of artificial neurons, with at least an input and output layer. If a neural network has several intermediate, or hidden layers, it is called a deep neural network. Neural networks are trained in an iterative process called backpropagation. The training data is put through the network, and after comparing the result of the network to the real classification, the artificial neurons are adjusted accordingly. This process is repeated until no further improvements are made.

III. Network Forensics Tools

In this section some common tools that are used to support network forensics analysis are presented. They largely rely on manual operation by knowledgeable users, either through direct supervision or by employing a rule-based approach, which means that an expert only analyzes a certain batch of data once it has breached predefined rules about, for instance, the number of packets within a specified time-frame.

a) Wireshark: it is a free and open-source network protocol analyzer and one of the most widespread tools for network forensics [5]. It fulfills the functionality of several NF phases. Through it, any packet being transmitted through a particular connection can be captured and subsequently stored, by default this happens in the widely used **.pcap* file format. Wireshark allows for detailed analysis of the packet data by providing a wide array of filters, such as network protocol, source or target addresses, and contents. Also, it provides the possibility for built-in statistical analysis

of the data. Nevertheless, Wireshark does not automatically guarantee preservation of the captured data by performing an automated backup or similar measures, but the file needs to be preserved manually. Moreover, even if Wireshark provides a wide array of features, it is only used to observe the network traffic, and provides no automated system for detecting intrusions.

b) Xplico: it is a free and open-source network forensics analysis tool (NFAT) which focuses on the analysis and investigation of packets [6]. Unlike Wireshark, Xplico cannot capture packets itself and is primarily used for analysis of already captured packets. From these packets, application data of the respective application protocol can be reconstructed. For instance, all captured HTTP-related packets can be used to reconstruct web pages, images etc. which were accessed during that session. Xplico is highly modular, allowing users to also develop decoders for more complex protocols. It offers the ability for a more in-depth analysis and investigation of the content of network traffic, by reconstructing user activity and interactions over the network. Encrypted traffic, however, cannot be reconstructed. With secure protocols such as HTTPS becoming the new standard, this severely impairs the usefulness of Xplico, especially since potential attackers are more likely to ensure secure communication.

c) NMap: it is a free and open-source network scanner that can be used to determine the characteristics of a network and the machines connected to it [7]. This can be useful for detecting unauthorized connections or collecting information about a host for later use. Nmap offers functionality to discover reachable hosts, scan a host for all available ports, and provide information about these ports. Among others, it includes whether they are open or not, which application is listening on a port, information about the device type, and a guess about the operating system of the device. NMap can be used to analyze the structure of a network, existing connections, and provide information about which applications may be running on the machines connected to the network. Additionally, the NMap Scripting Engine(NSE) allows for the automation of such features, simplifying the data collection and analysis.

d) Snort: it is a free and open-source intrusion detection system (IDS). It is capable of capturing packets and performing detection in real-time, using a rule-based approach [8]. Such approach combines both signature- and anomaly-based detection. The first one examines the traffic for known exploits and generates an alert when a match is found, while the second one examines the statistical properties of network traffic and raises alerts when certain deviations occur. Snort rules combine both approaches to accurately define malicious traffic and protect against unknown exploits, even if no signature exists yet. Rules can also be developed and published by users, meaning they can be tailored to very specific environments and network types. By analyzing the alerts and logs generated by Snort, investigators can take action against attacks promptly, determine the attack source, and how it was conducted to prevent further attacks of the same type.

IV. Machine Learning Approaches

Machine learning has taken on a prominent role in network forensics, with manual systems reaching their limit in the face of rapidly growing amounts of data. Initially, machine learning was impaired by the lack of high-quality public datasets which are needed for training a machine learning model. Since publishing real network traffic in a public dataset severely violates many privacy policies, artificial datasets were constructed and used for training. These datasets however lack verification that they accurately represent real data, and are thus limited in their usefulness. The choice of how to label the training data presents another challenge. In both intrusion detection and traffic analysis, the central problem is a twofold classification problem. Firstly, traffic needs to be classified to be either legitimate or malicious. Secondly, most datasets also classify malicious traffic into different types of attacks. For instance, the authors in [15] classifies attacks into nine types in total, among them DoS, Backdoors, Reconnaissance, Exploits and Worms. Most studies use certain metrics to compare their classifiers to baseline methods. To compute these metrics, the classified data is divided into four categories: If a data item is correctly classified as malicious, it is a true positive (TP), if a data item is falsely identified as malicious, it is a false positive (FP), if a data item is correctly identified as benign, it is a true negative (TN), and if a data item is falsely identified as benign, it is a false negative (FN). The most commonly defined metrics are: Accuracy, Precision, Recall and F1-score.

A. DDoS Detection using K-nearest neighbour & Naive Bayes

In [4] the K-nearest neighbour(KNN) algorithm and a naive Bayes classifier for detecting anomalous traffic encountered during a DDoS attack was evaluated. KDDCup and NSL-KDD datasets were used, which contain generated traffic data, combined with generated attacks. The datasets label traffic at any time as being either ”normal”, that is typical user activities such as file transferring, or ”attack”, with four attack types: user to root attack, root to local attack, DoS attack, and probing attack. The datasets are split into training and testing data. Initially, a pre-processing step is performed on the training data: all attacks except DoS are removed, and the correlation of the features with the label is computed. Features which do not exhibit significant correlation with the label are dropped from the dataset to reduce noise. In this case, pre-processing produces eight features that indicate if a DDoS attack is taking place. Among these features are, for instance, the total amount of connections of one host, the total amount of bytes sent from destination to source, and the connection protocol used. The KNN-algorithm uses distance measures to model similarity between different data-points. The most popular distance measure is the Euclidean distance, given by

$$dist(x, y) = \sqrt{\sum_i^n = 1(x_i - y_i)^2}$$

TABLE I
COMPARISON OF NETWORK FORENSIC TOOLS

	Wireshark	Xplico	NMap	Snort
Category	Packet Analyzer	Network Forensic Analysis	Network Scanner	Intrusion Detection System
Main Use	Traffic capture and analysis	Application data extraction	Network discovery and security auditing	Intrusion detection and countermeasures
Supported Phases	3 , 5, 6, 7	6, 7	3, 6, 7	2, 3, 5, 6, 7
Data Capture	Yes	No	Yes (no traffic data, only scanning results)	Yes
Data Inspection	Yes	Yes	Yes (no traffic data, only scanning results)	Yes
User Interface	GUI	Web-based GUI	CLI	CLIs
License	Open Source	Open Source	Open Source	Open Source

where x and y are data items with n features. Other measures exist, such as the Manhattan or Minkowski distance. The labelled training data is then placed into the feature space. When a new, unlabelled data item needs to be classified, the distance to each training data item is calculated, and the K nearest data items are identified. The new data item is then classified by a majority vote. The naive Bayes classifier is a probabilistic classifier which relies on the assumption that, given the class label, the features are independent from eachother. While this assumption is often false, it simplifies computation significantly. The classifier then assigns each data item the most likely class, given the features of the data item. The evaluation concludes that KNN performs better than naive Bayes, with an accuracy of 98.51% and 93.95%, a Recall of 97.8% and 95.54%, and a precision of 98.9% and 97.74%, respectively for KNN and naive Bayes, across the two datasets.

B. Network Intrusion Detection using Deep Learning

Ashiku and Dagli [12] implement a deep learning model to detect attacks in a network. The authors use the UNSW-NB15 dataset, which labels traffic either as normal or as one of nine different attack types. After performing brief pre-processing steps, a Convoluted neural network (CNN) is trained with the data. The authors utilize semi-dynamic hyper-parameter optimization, which includes hyperparameters such as learning rate, optimization algorithm and batch size. After defining baseline hyperparameters, new ones are chosen until a decline in performance is observed. Additionally, callback functions such as early stopping and model checkpoint are considered, which improve the training process and can help prevent overfitting by continously evaluating the model and stopping or even rolling back to a previous state if no further improvements are made after a certain time. The architecture also includes techniques such as dropout, max pooling and double stacked convolutional layers. These techniques improve the performance of the model while reducing overfitting and computational load during training. In addition to the standard UNSW-NB15 dataset, a modified variant of the same dataset is used for evaluating the architecture. In the modified dataset, the prepartitioned training and testing datasets are merged and then split with 70-30 for training and testing. The authors also note a class imbalance in the dataset. While the most common class is represented with 56 thousand instances, the least common one is only represented with 130 instances. This reduces the models robustness and accuracy. However, to provide a fair comparison with other architectures, the authors remain with the dataset. One advantage of using deep learning methods is the better performance of those models when dealing with zero-day exploits.

C. Intrusion Detection System in Imbalanced Network Traffic

Ullah et al. [13] introduce a novel intrusion detection system (IDS) called IDS-INT. IDS-INT is designed to detect attacks in Imbalanced Network Traffic (INT), which is achieved by utilizing a transformer-based transfer learning approach to extract complex feature interactions that indicate different attack types. The system is designed as a network-based intrusion detection system, which receives captured packets in the human-readable *.pcap file format. After preprocessing, BERT is used to learn deep and complex feature dependencies. BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained model that can comprehend text based on the surrounding context, as such it is called a contextual model. Since *.pcap is a textual representation of packets, the authors use BERT to extract the contextual meaning from the captured data. To avoid reduced accuracy due to class imbalance, SMOTE (Synthetic Minority Over-sampling Technique) is employed. SMOTE generates synthetic instances of rare classes based on the real instances in the data. Afterwards, the data is passed through a hybrid model combining a (CNN) and a long short-term memory (LSTM) network. The model utilizes techniques such as max-pooling to reduce computational overhead as well as softmax and dropout layers to reduce overfitting. The combination of a CNN, which is effective at understanding local, subtle patterns in features, and LSTM, which can correlate both long- and short-term dependencies, improves overall accuracy and makes the system more robust. The authors evaluate IDS-INT on three datasets, including UNSW-NB15. IDS-INT performs with an accuracy of 99.21% on the UNSW-NB15 dataset, better than the comparative baseline models.

D. Ransomware Detection using Entropy

Williams et al. [11] propose using entropy to detect ransomware. Ransomware describes malicious software that

encrypts a users data against their will and only provides the decryption key after the transfer of a certain amount of money to the attacker, often in the form of cryptocurrencies such as Bitcoin. Present detection methods such as signature-based detection suffer from vulnerability against unknown ransomware or a high false-positive rate. Due to the high financial incentive for attackers, ransomware is continuously evolving and developed to evade known defences. This increases the necessity for an adaptive approach that can detect novel or polymorphic ransomware attacks.

The authors use entropy to detect ransomware attacks. Entropy describes the randomness or variability of a certain variable. For instance, the entropy values of packet size or source-destination pairs can be computed. Entropy would be high if packets of many different sizes were encountered, and low if most packets were similar in size. More specifically, Shannon Entropy is used. For each feature, it is defined as:

$$H(X) = -\sum_{i=1}^{n} p(x_i) log p(x_i)$$

where $p(x_i)$ described the probability of observing X_I when drawing from the random variable X, which represents a single feature.

Initially, packets are captured from a network using a tool like Wireshark. The packets are then preprocessed in real-time to facilitate proper calculation of the entropy values. To properly detect sharp fluctuations in the entropy values, the first derivative of the entropy values over time is calculated. This derivative is used to detect certain phases in a ransomware attack, such as data exfiltration. Additionally, the second derivative is calculated to detect the acceleration of entropy change. This derivative is used to detect phases such as the onset of encryption of the victims data. Before classificaion, the entropy values for different variables are aggregated. One one hand, they are aggregated by different time windows to detect changing traffic behavior in the entire network. Additionally, they are also aggregated by the different traffic flows they belong to, for instance all entropy values of packets originating from the same source are aggregated. These aggregations improve the overall detection rate of different types of ransomware attacks. The aggregated entropy values are then classified using machine learning models such as SVM, Random Forest, or neural networks. The use of several models, each with their own advantages and disadvantages, allows for a hybrid model which improves overall detection and can allow for real-time traffic classification. If the hybrid model classifies a certain entropy value as being the product of a ransomware attack, a real-time alert is produced. The authors evaluate their entropy-based system across several different datasets, and against other methods for detecting ransomware attacks. The entropy-based system performs best among those methods, achieving a detection rate of 99.2% with a false-positive rate of 1.1%. Additionally, it is the fastest method, an important factor in real-time detection. In conclusion, the entropy-based ransomware detection system is proven to be adaptable and fast compared to alternative methods. Entropy analysis can adapt to changing traffic flows, and can still detect ransomware attacks, even new and unknown ones.

E. Ransomware Detection using Network Traffic Analysis and Generative Adversarial Networks

A different approach for detecting ransomware using Generative Adversarial Networks (GANs) is proposed in [14]. GANs consist of two neural networks, a generator and a discriminator. While the generator generates artificial data based on real data, the discriminator receives both the real and artificial data and is then tasked with deciding if the data is artificial or real. Both networks work against each other, with the generator attempting to deceive the discriminator, and the discriminator attempting to classifying the data as accurately as possible. The goal is to train the discriminator to be very sensitive to deviations from the distribution of nominal traffic, to allow for the detection of anomalies later on.

Before feeding the real data into the GAN, feature extraction is performed. The authors selected, among others, packet size, flow duration and packets per flow. As discussed in section *D*, these attributes can hold indicators for a potential ransomware attack, for instance a high flow duration, which describes the time between the first and last package sent within a flow, may indicate a connection between an affected machine and a command-and-control server. Similarly, a high number of packets per flow may indicate encryption or data transfer.

The two components of the GAN, the generator and the discriminator, are implemented as deep neural networks to capture the complex dependencies of network traffic data. During training of the GAN, both the generator and discriminator are iteratively updated through backpropagation. For each iteration, the generator generates artificial data, which is then combined with the real training data and fed to the discriminator. The discriminator is then tasked with classifying the input data into artificial and real data. Training is performed in such a way that the generator aims to deceive the discriminator by generating data as similar as possible to the real data, while the discriminator aims to classify the data as accurately as possible. This adversarial approach results in the discriminator becoming very sensitive to differences between real and anomalous data.

After training, the discriminator can be used to detect anomalous behavior in network traffic data. While the authors focus on detecting ransomware by selecting the respective features during feature extraction, the adaptation of GANs to other types of attacks is possible. The GAN-based detection method performs better than both signature- and statistics-based detection methods, with an accuracy of 94.2% and a precision of 91.5%. However, the GAN-based approach produced a significant amount of false-positives due to the inherent variability of network traffic, such as network congestion or legitimate encrypted traffic.

TABLE II
COMPARISON OF MACHINE LEARNING APPROACHES

	KNN	Naive Bayes	Deep Learning	IDS-INT	Entropy	GANs
Year	2021	2021	2021	2023	2024 (preprint)	2024 (preprint)
Used techniques	K-Nearest Neighbours	Naive Bayes	Deep Learning	BERT, SMOTE, LSTM, CNN	Entropy analysis, SVM, RF, NN, ensemble methods	GAN, Deep Learning
Datasets	KDDCup & NSL-KDD	KDDCup & NSL-KDD	UNSW-NB15	UNSW-NB15 & NSL-KDD	multiple, not disclosed	multiple, not disclosed
Accuracy	98.51%	93.95%	94.4% (95.6% on modified dataset)	99.21% and 98.1%	96.84%	94.2%
Precision	98.9%	97.74%	-	99% and 99%	97.5%	91.5%
Recall	97.8%	95.54%	-	100% and 98%	98.5%%	89.1%
F1-score	omitted	omitted	-	99% and 98%	98% (own calculation)	90.3%

V. DISCUSSION

A. Challenges

Despite significant progress of the utilization of ML in the domain of network forensics, some challenges remain.

a) Datasets: The lack of sufficiently large, labelled, real-world data presents a fundamental challenge in utilizing machine learning. Without representative data, the performance of models in real-world applications may be significantly worse than the initial evaluation results. For instance, deviations may cause a significant amount of false positive alerts, which would result in a lower precision. Simply capturing real network traffic data raises significant privacy concerns. Therefore, research has been focused on creating artificial datasets, relying on statistical methods and resampling to create data which closely resembles real network traffic [12].

More recently, hybrid datasets have risen in prominence, for instance the widely used USNW-NB15 dataset [15], which includes real traffic collected at the University of New South Wales in Australia. However, in both synthetic and hybrid datasets, attacks are inserted artificially based on know attack profiles. For instance, one of the first widely used datasets, KDDCup, consists of captured generated network traffic, amended with artificial attacks to simulate the environment of a US air force base. An improved version of this dataset, NSL-KDD, remains one of the most used datasets despite being criticized for not representing real-world data. Recently however, some datasets including real world-data have been published [2].

Besides inaccurately depicting real-world data, some datasets display a heavy class imbalance, which can cause bad performance in real-world applications. For instance, the UNSW-NB15 dataset contains ten classes: nine attack types and normal traffic. However, the most frequent attack has 56 thousand instances, while the least frequent only contains 130; among a total of 2.54 million entries. This class imbalance can cause disproportional detection accuracy of certain attack types, where the model essentially disregards learning the rare attack type patterns. In [13], this problem was mitigated by resampling the rare attack cases, artificially increasing their frequency, while in [12] the problem was identified, but disregard it to maintain comparability to other studies.

The concern that user behavior may differ between countries, cultures, and other socioeconomic factors was raised in [11]. If network traffic is indeed significantly different, creating suitable datasets would be a near-impossible task for many applications: e.g. global companies with employees from several countries may have to create tailor-made datasets for their network analysis system.

b) Encryption: With the growing popularity and awareness for privacy, an increasing amount of internet and network traffic is being encrypted. The Google transparency report indicates that nearly 95% of traffic generated by users browsing the internet with chrome is using the HTTPS protocol. This poses a challenge for tools such as Xplico, which aim to reconstruct application data. Additionally, encrypted data may exhibit different properties than unencrypted data, which adds to the already existing challenges surrounding suitable datasets. For instance, in their preprint, Williams et al. [11] measure a reduction in detection of attacks from 99.4% for unencrypted traffic to 96.1% for partially encrypted and 91.3% for fully encrypted traffic. However, they only perform this evaluation one one of their several datasets. Despite all the challenges encryption poses, increasing the ratio of encrypted traffic should also be in the interest of those actors who persecute criminals or need to protect their system against attacks, since encryption can prevent many crimes, intrusions and information breaches or leaks.

c) Feature Selection (IDS-INT): Feature selection or extraction describes the process of preemptively selecting meaningful features, while discarding unimportant or noisy ones. Through feature selection, accuracy of the model can be improved and the computation cost of the model can be reduced. However, as Ullah et al. [13] noted, current feature selection techniques are limited in understanding complex dependencies between features, which may impair

performance. Alternatively, they can be selected manually, but this requires expert knowledge about the relevant types of attacks.

B. Outlook

Based on the research done for this report, a few opinions can be voiced about the future of machine learning in network forensics. Despite the remaining challenges, machine learning has shown its potential to improve cybersecurity infrastructure through its ability to reliably detect anomalies and classify attacks. While some of the presented papers are preprints, published studies confirm the capacity of ML techniques.

Research in this area is largely focused on intrusion detection and traffic analysis, and primarily on the detection, analysis and investigation phases of network forensics. This seems reasonable, since the other relevant phases, namely collection, preservation and examination, are sufficiently covered by other tools, and there is little room for improving, for instance, the capture of packets by using machine learning, unless computational power increases to the point where stream-based anomaly detection becomes feasible.

Regarding the collection of criminal evidence, using machine learning techniques such as facial recognition in law enforcement is a controversial topic. Widely employing such techniques would likely warrant the need for large numbers of trained officials to supervise machine learning algorithms and react to alerts. Additionally, producing legally permissible evidence that can be used in court requires strict guidelines. It is entirely unclear if evidence produced by machine learning techniques would be sufficient. Unless artificial intelligence becomes more transparent and accountable, it will likely be limited to a supportive role. To this end, explainable AI would be a reasonable solution.

As discussed above, the scarcity of sufficient datasets poses a challenge for supervised learning algorithms. To combat this issue, a hybrid system could be used. For instance, instead of using supervised learning for detecting attacks, unsupervised learning methods, which do not require labelled data, could be used to learn the nominal behavior of a network. To prevent the model from learning attacks that occur during operation as nominal data, such attacks would have to be removed from the data as soon as they are discovered elsewhere. When the unsupervised model detects an anomaly, a supervised model could then classify the anomaly again. Such a system could potentially reduce the occurrence of false positive alerts due to the adaptive and dynamic nature of unsupervised learning methods. Regarding the challenge of encrypted data, researchers will have to focus on analyzing metadata instead of contents. Privacy is an important right, and if state actors or companies were able to extract contents or infer user activity from network traffic, so could malicious actors. Therefore, furthering privacy-protecting encryption schemes is desirable.

VI. Conclusion

The paper has focused on Machine Learning, as a part of the rapidly growing field of AI in the context of Internet of Everything (IoE), especially considering the large role to play in the future digital forensics investigation in network traffic. Some of the most pupular forensics tools have been introduced. Furthermore, machine learning techniques, which are being utilized, ranging from relatively simple ones such as KNN and SVMs to deep learning and CNNs, have been presented. However, as highlighted in the discussion section, challenges remain, such as those related to the quality of the available data, which is being one of the most urgent one.

Acknowledgment

We would like to thank Mr. Jan-Philipp Kolb for his invaluable work conducted in the context of the seminar Protection in Infrastructures and Networks (PIN) held at Dept. of Computer Science at Technical University of Darmstadt (TU Darmstadt), contributing substantially to the realization of this research.

References

[1] G. Fedele and L. D'Alfonso, "A model for swarm formation with reference tracking," 2017 IEEE 56th Annual Conference on Decision and Control (CDC), Melbourne, VIC, Australia, 2017, pp. 381-386, doi: 10.1109/CDC.2017.8263694

[2] S. Rizvi, M. Scanlon, J. Mcgibney and J. Sheppard, "Application of Artificial Intelligence to Network Forensics: Survey, Challenges and Future Directions," in IEEE Access, vol. 10, pp. 110362-110384, 2022.

[3] A. R. Javed, W. Ahmed, M. Alazab, Z. Jalil, K. Kifayat and T. R. Gadekallu, "A Comprehensive Survey on Computer Forensics: State-of-the-Art, Tools, Techniques, Challenges, and Future Directions," in IEEE Access, vol. 10, pp. 11065-11089, 2022.

[4] A. V. Kachavimath, S. V. Nazare and S. S. Akki, "Distributed Denial of Service Attack Detection using Naïve Bayes and K-Nearest Neighbor for Network Forensics," 2nd Int. Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India, 2020, pp. 711-717.

[5] J. Beale, A. Orebaugh and G. Ramirez, "Wireshark & Ethereal network protocol analyzer toolkit", Elsevier, 2006.

[6] https://www.xplico.org. Visited 11.01.2025

[7] https://nmap.org. Visited 11.01.2025

[8] https://www.snort.org. Visited 12.01.2025

[9] E. S. Pilli, R. C. Joshi and R. Niyogi, "A Generic Framework for Network Forensics", International Journal of Computer Applications, 2010, 1(11), 1-6.

[10] E. van De Wiel, M. Scanlon and N. A. Le-Khac, "Enabling non-expert analysis of large volumes of intercepted network traffic", Prof of the 14th IFIP Int. Conf. on Digital Forensics (DigitalForensics), Jan 2018, New Delhi, India. pp.183-197.

[11] Williams, M., Morales, R., Johnson, K., Martinez, G., Bennett, J. (2024). Entropy-based network traffic analysis for efficient ransomware detection. TechRxiv. October 08, 2024.

[12] Ashiku, L., Dagli, C. (2021). Network intrusion detection system using deep learning. Procedia Computer Science, 185, pp. 239-247.

[13] Ullah, F., Ullah, S., Srivastava, G., Lin, J. C. W. (2024). IDS-INT: Intrusion detection system using transformer-based transfer learning for imbalanced network traffic. Digital Communications and Networks, 10(1), 190-204.

[14] Wiles, A., Colombo, F., Mascorro, R. (2024). Ransomware detection using network traffic analysis and generative adversarial networks. Authorea, September 17, 2024. DOI: 10.22541/au.172659907.77469627/v1

[15] Moustafa, N, and Slay, J. "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)." Military Communications and Information Systems Conference (MilCIS), IEEE, 2015.

[16] https://transparencyreport.google.com/https/overview Visited:29.01.2025