Individual Packet Features are a Risk to Model Generalisation in ML-Based Intrusion Detection

Kahraman Kostas ⓘ, Mike Just ⓘ, and Michael A. Lones ⓘ

Abstract—**Machine learning is increasingly used for intrusion detection in IoT networks. This paper explores the effectiveness of using individual packet features (IPF), which are attributes extracted from a single network packet, such as timing, size, and source-destination information. Through literature review and experiments, we identify the limitations of IPF, showing they can produce misleadingly high detection rates. Our findings emphasize the need for approaches that consider packet interactions for robust intrusion detection. Additionally, we demonstrate that models based on IPF often fail to generalize across datasets, compromising their reliability in diverse IoT environments.**

Index Terms—**IoT security, Network Security, Intrusion detection, Machine learning, Attack Detection.**

I. INTRODUCTION

The continually growing number of connected devices, coupled with the heterogeneity of hardware and software designs across manufacturers, presents a significant challenge for securing the Internet of Things (IoT) ecosystem [1]. The inherent variety in these devices can lead to complex security vulnerabilities, potentially compromising network security. For example, weakly secured IoT devices can be exploited by malicious actors, turning them into "botnet zombies" used in large-scale cyberattacks against critical infrastructure [2].

The multifaceted nature of IoT security challenges has attracted considerable scholarly attention. Notably, intrusion detection employing machine learning (ML) techniques has emerged as a particularly active area. Although specific ML methods have evolved over time, the fundamental reliance on data remains constant, underscoring the indispensability of robust datasets in data-driven approaches.

Current intrusion detection studies with ML primarily rely on three feature types [3]: flow-based features (analyzing network traffic statistics) [4], [5], window-based features (focusing on packet variations within a timeframe) [3], [6], and individual network packet features [7], [8].

The third approach is the simplest — it uses features extracted from individual packets, and does not take into account interactions between packets — yet despite this, previous studies have reported high detection accuracy using this approach.

In this study, we critically examine the efficacy of IPF in ML-based intrusion detection systems, revealing the approach to have significant limitations, particularly in terms of the ability of models to generalise. We demonstrate, through extensive experiments on various public datasets, that reliance

on IPF can lead to vulnerabilities and reduced robustness in intrusion detection. Our findings underscore the importance of adopting more informative feature sets, such as flow-based and window-based features, to enhance the security of IoT ecosystems. Notably, this study is the first to provide empirical evidence of the risks associated with IPF.

II. WHY IPF DOES NOT TELL THE WHOLE STORY

Detecting an attack using IPF alone is challenging. In this section, we'll elaborate on the reasons behind this difficulty. In the subsequent sections, we support this notion by presenting examples from existing literature and our own experimentals. For reproducibility, our scripts are publicly available[1].

This challenge can be explained with a simple example: SYN Flood [9], which is a Denial-of-Service (DoS) attack based on the exploit of 3-way handshake in the TCP protocol. During the 3-way handshake process, the client that wants to establish a connection sends a SYN packet to the server. The server receiving this packet sends a SYN-ACK packet to the client. Finally, the client responds with the ACK packet, and the TCP connection is established. (see Figure 1). During an attack, the server is sent a larger number of SYN

Fig. 1: Legitimate and malicious use of 3-way handshake

(a) The legitimage usage.

(b) SYN Flood attack.
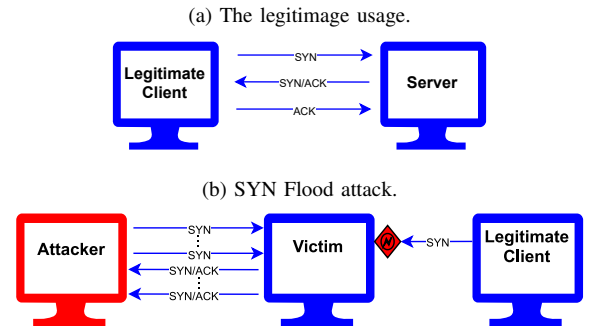
packets for which a SYN-ACK reply is sent, but the ACK reply is never received. Server resources are impacted, and subsequent legitimate connection requests could be dropped. The distinguishing individual packet features in this case are typically those such as IP or MAC addresses (or other network-specific features) which are identifying within one dataset, but which do not generalise to other datasets or network environments.

This phenomenon extends beyond SYN Flood attacks; various attack scenarios causing time, size, or protocol anomalies,

such as DoS, distributed DoS (DDos) and man-in-the-middle (MitM) attacks, display similar characteristics. To efficiently detect and mitigate an attack, it is essential to adopt an approach that considers the context and interaction of packets, rather than solely relying on individual packet characteristics.

There are some exceptions, such as for single-packet attacks that exploit malformed packets. These attacks typically involve sending malformed packets that the device cannot handle, leading to malfunctions or crashes in the receiving device [10]. However, addressing these attacks usually involves packet filtering or firewalls, not ML behavioural analysis, since they can be effectively managed using signature/rule-based approaches.

III. IPF Usage in the Literature

The use of individual packets in attack detection is common. Our literature review, conducted in [3], reveals that 10 out of 68 studies from 2019 to 2023 incorporate IPFs. Table I summarises key information from these studies, such as the dataset, data types, and metrics, and Table II lists the attacks detected.

This reveals numerous studies with reported high success rates in attack detection using IPF. Given that we have posited that IPFs are an ineffective basis for attack detection, it is important to understand this discrepancy.

TABLE I: Studies in the literature using IPF, their datasets and results. The dataset specified as Private* is the same in both studies.

Study	Dataset	Accuracy	Recall	Precision	F1 Score
[11]	Edge-IIoTset(CSV)	100.00	100.00	100.00	100.00
[12]	Edge-IIoTset(CSV)	100.00	89.00	95.00	87.00
[13]	MQTT-IoT-IDS2020(CSV)	99.98	99.98	99.98	99.98
[14]	MQTTset(CSV)	91.00	91.00	77.00	80.00
[8]	AB-TRAP (CSV)				100.00
[15]	AWID2 (RAW)	99.96	99.99	99.95	99.97
[16]	ISCXIDS2012 (RAW)	99.42	99.41	99.34	99.37
	CICIDS2017 (RAW)	97.87	98.16	97.59	97.83
[17]	Private	100.00	100.00	100.00	100.00
[7]	Private*		98.00	99.00	99.00
[18]	Private*		100.00	100.00	100.00

TABLE II: List of studies and the attacks they include

Study	Attacks
[11]	DoS/DDoS, Scanning, MitM, Injection, Malware
[12]	DoS/DDoS, Scanning, MitM, Injection, Malware
[13]	Brute-Force, Scanning,
[14]	DoS/DDoS, MitM, Injection, Packet Manipulation
[8]	Scanning
[15]	DoS, Injection, Authentication , Wireless Attack
[16]	DoS/DDoS, Brute-Force, Infiltration, Scanning, Botnet, Web
[17]	Wrong setup, DDoS, Probing, Scanning, MitM
[7]	DoS, MitM, Scanning, IoT-toolkit
[18]	DoS

Most studies in Table I use open access datasets, and those used in [8], [11]–[14] have a similar nature in that all contain raw data as well as individual packet specifications in pre-extracted CSV format. All of the studies used these pre-extracted versions. Another feature of these datasets is that they contain a single file for each attack, with no information about sessions. Therefore, in all these studies, it seems likely that the training and test data were created from a single file.

This promotes identifying features, such as IP addresses, across training and test sets, causing information leakage. Such leakage can inflate performance metrics by granting the model access to information during testing that it would not have during deployment. Consequently, these features serve as hidden variables, offering a shortcut to class identification. The model may then prioritize them over learning true underlying patterns, as illustrated in Figure 2. For example, in [11], [13], [14], although the reporting of features used in these studies is unclear, there is no evidence that these identifying features have been removed or censored. In this respect, it is quite possible that identifying features were used in their models.

Nevertheless, the inclusion of source and destination-based identifiers is generally considered a significant flaw, and most studies tend to remove such features from their data. For instance, in [12], IP addresses were removed. However, this study suffers from a different type of information leak, with the dataset containing features such as ports, IDs, synchronisation, and acknowledgement numbers. Despite these features being extracted from individual packets, they provide non-generalizable insights into the interrelation of packets. For instance, when a flow is established between two ports, all individual packets within that flow bear the corresponding port numbers (see Figure 2 as an example, focusing on port and ID numbers). In a train/test split that does not consider the flows, even with more robust evaluation methods like cross-validation (CV), it is highly likely that packets representing the same flow will occur in both the training data and the test data. Moreover, the fact that a single file with no distinct sessions is used to represent each attack makes it very challenging to split in a manner that prevents information leakage. So, in this study, it can be seen that the features that stand out for many attacks are sequence and acknowledgement numbers, which are session-based identifiers, and it is quite possible that they are subject to information leakage.

In none of these studies [8], [11]–[14] did we see any indication that flows were taken into account in the training and test split, or that these session-based identifiers were removed (Although session-based features are mostly removed in Bertoli et al.'s work [8], the low complexity in the structure of the dataset still leads to generalisation problems. this will be examined in Section IV-B) . Therefore information leakage between the training and test data is distinctly possible. While the results published in these studies may appear promising, models relying on identifying features due to information leakage will not generalise to other datasets, and consequently the results are likely to be misleading.

Two studies [15], [16] used raw pcap files rather than pre-extracted features. However, the use of raw data can be quite dangerous because these files also contain identifiers such as IP and MAC addresses. In one of these studies [15], there is no indication that these identifying features were deleted/discarded, so it is quite likely that they were used. In the other study [16], identifying features such as IP addresses and frame number were removed. Despite this precaution, we suspect that some identifying information might have

inadvertently leaked. For example, in their study, the internet checksum has been used, which is an identifying feature as it stores a hashed version of the source and destination IP addresses. In addition, both studies had no mechanism to prevent sessions from being split across train and test sets, providing a potential route for information leakage.

The other three studies [7], [17], [18] listed in Table I did not use publicly available data. Many of the features used in [17] are identifying features (such as frame number/time, eth source/destination, source/destination IP).

In two studies [7], [18], identifying features such as IP, MAC, and frame number were removed, but session-based segregation was not applied, likely causing session characteristics to leak between training and test sets. Notably, [7] claims the testing phase was isolated from training data, although the process for achieving this is not clear.

IV. Experimental Case Studies

In the previous section, we highlighted the role that identifying features may play in giving an incorrect impression that IPF-based approaches are effective. Next, we strengthen this claim using experimental case studies. We also consider another potentially misleading factor, low data complexity.

A. Identifying Features

In SectionII and III we argued that IPF-based attack detection yielded unrealistic results based on information leakage. To substantiate our claim, we conducted experiments using the IoT-NID [19] dataset, which offers multiple sessions for each attack. This dataset records each attack execution as a separate session, providing an opportunity to explore the identifiability of session-based features. We utilised examples from this dataset to demonstrate the effectiveness of session-based identifiers. Our approach involved merging the first and second sessions of the attacks to create a new dataset. We then performed a 10-times 10-fold CV on this merged dataset, employing one of the session-based identifiers at each iteration to distinguish between benign and attack samples. In the subsequent step, we executed the separation by utilising the first session as the training set and the second session as the testing set. Figure 3 illustrates the distribution of these features for four attacks.

The results of this analysis reveal a notable pattern: in the majority of cases, the identifying features demonstrate better success in the CV scenario, while in isolation, they exhibit limited effectiveness. Figure 3 visually depicts this trend. In this context, these results support the hypothesis that these features lead to information leakage.

However, upon a more comprehensive examination of the figures, we observed that certain features consistently display exceptional performance in both scenarios. Dport feature in Telnet Brute-force (Figure 3b) and HTTP Flooding (Figure 3d) attacks are prime examples of this consistent success across the two scenarios. If we examine the models associated with the features that achieve this unexpected success, as shown in Figure 4, we see that these models have an unusually simple structure. This brings us to the second explanation for

Fig. 2: An 80/20 split for a hypothetical dataset containing 40 samples (4 session, 40 network packets), showcasing the division into 80% training data and 20% testing data for model development and evaluation, respectively. This example illustrates how information leakage takes place. In the scenario involving the source IP attribute, information leakage occurs because this address uniquely identifies both the attacker and the benign source, remaining constant across both the training and test sets. If we examine other features such as source port numbers and ID numbers, they vary across different sessions but exhibit correlations within each session (port numbers remain constant, ID numbers increment sequentially). Consequently, if packets from the same session are found in both the training and test sets, it can result in information leakage. Benign: 🟢 🟢 Malicious: 🔴 🔴
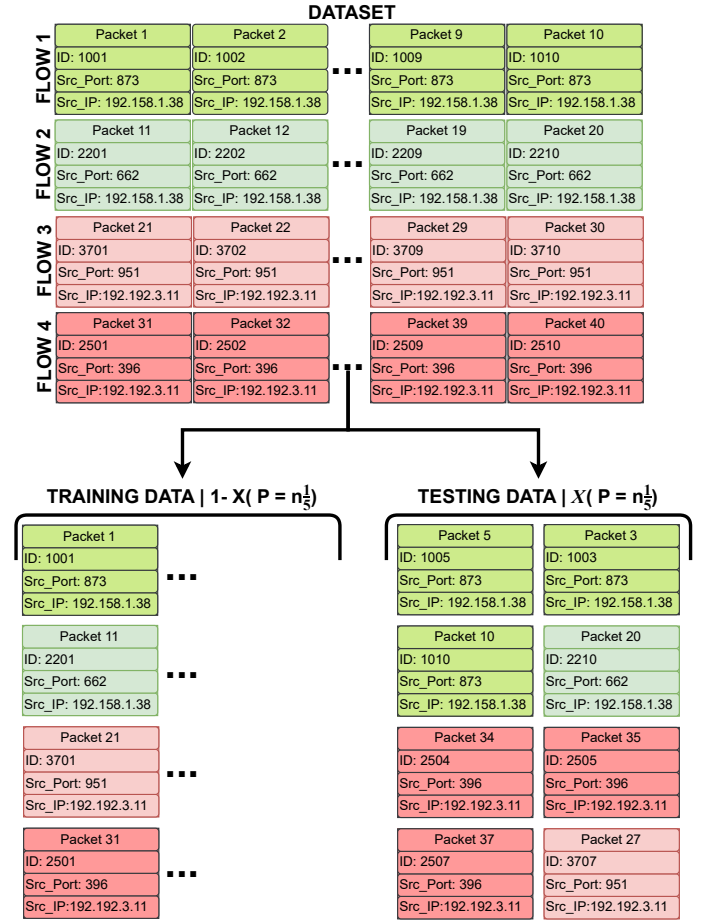
achieving high performance with individual packet features: low data complexity.

B. Low Data Complexity

Most attacks follow a pattern and produce uniform outputs. For example, the distribution of the size of the attack packets in the HTTP flood attack on four different datasets [12], [19]–[21] is shown in Figure 5. Because of the uniform nature of the attacks, if the complexity level of the dataset is low in studies using individual packets, even very basic characteristics, such as size or time, can become identifying features.

Fig. 3: Comparison of CV and isolated data performance of features on some attacks in the IoT-NID dataset with DT.

(a) IoT-NID dataset ARP Spoofing attack.

(b) IoT-NID dataset Telnet Bruteforce attack.

(c) IoT-NID dataset HTTP flooding attack.

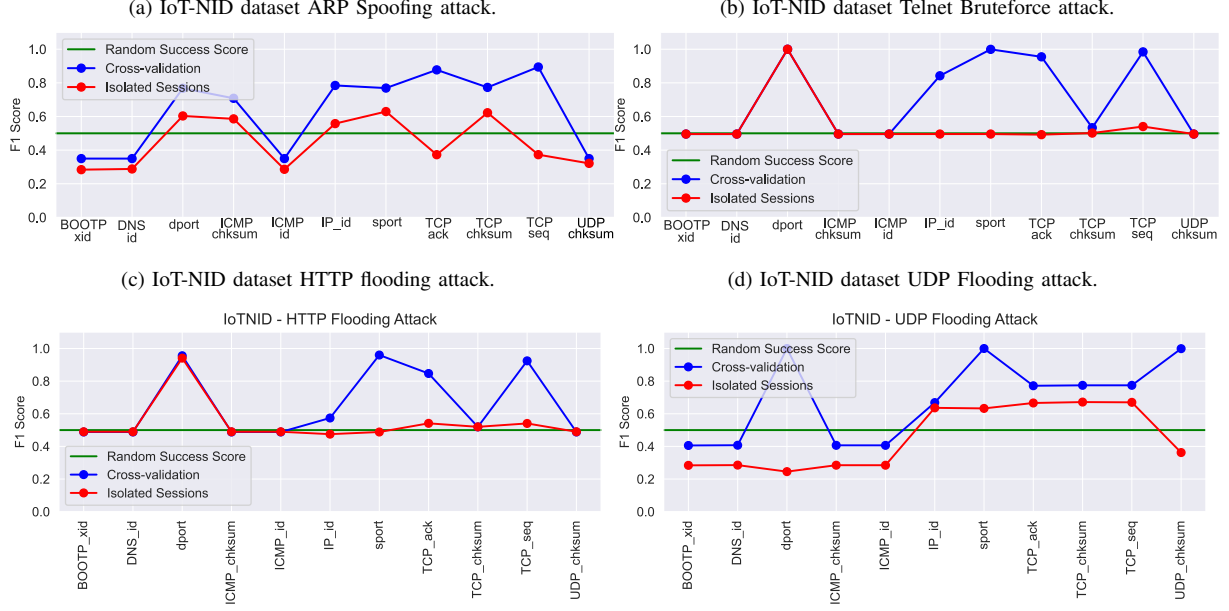(d) IoT-NID dataset UDP Flooding attack.

Fig. 4: Visualization of high-achieving decision tree models. (a) HTTP Flood model using dport feature, (b) Brute-Force model using dport feature.
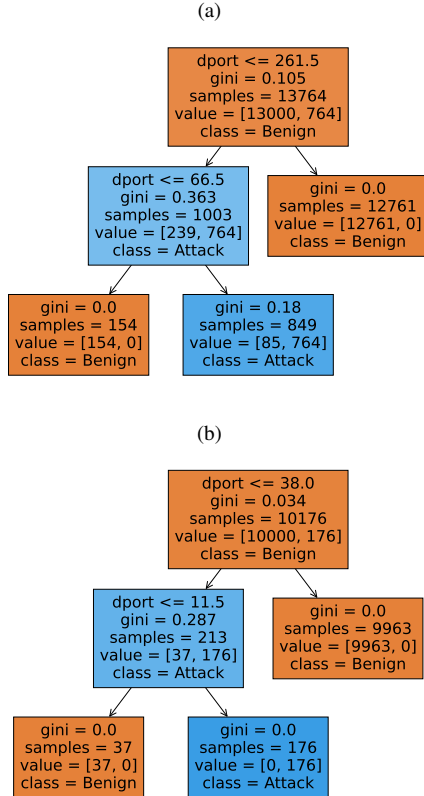

Fig. 5: The distribution of packet size malicious data (HTTP Flood) in four different datasets [12], [19]–[21].
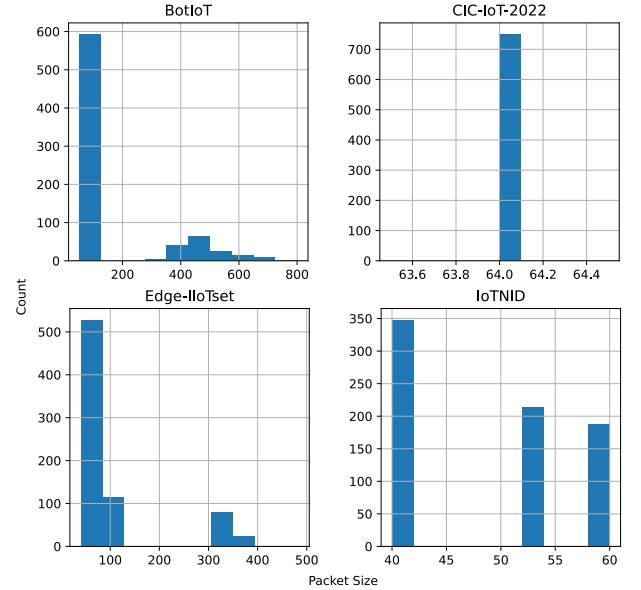

Figure 6 presents compelling empirical evidence to support this assertion. Specifically, in Figures 6a-(d), we depict the UDP attacks observed in four distinct sessions. A detailed analysis of these figures reveals notable similarities between sessions 1 (6a) and 3 (6c), as well as between sessions 2 (6b) and 4 (6d). Moreover, it is noteworthy that the size of attack packets remains constant even across these different sessions.

When incorporating only packet size as a feature in this example, we achieve perfect detection performance (see Table I). Even when testing the model trained with the first session on the third session, where the model has not encountered

this data previously, we observe a notably high success rate. The same holds true for sessions 2 and 4, indicating strong consistency in their detection capabilities. However, this superior success is only a consequence of the information leakage problem. The packet size in a UDP attack may change in another attack, and this model will fail to catch these attacks, or it will detect benign packets of this specific size as attacks.

Figure 6e provides another insightful example, showcasing the change of labels over time in the Kitsune dataset [6], Mirai attack. In this graph, packets before a specific time exhibit all benign characteristics, while those occurring after the designated time are identified as attacks.

Fig. 6: Distribution of labels in Kitsune and IoT-NID. (a), (b), (c), and (d) show the size histograms of sessions 1, 2, 3, and 4 for UDP attack in the IoT-NID dataset, respectively. (e) and (f) depict the distribution of labels over time for Mirai and Video Injection attacks, respectively, in the Kitsune.
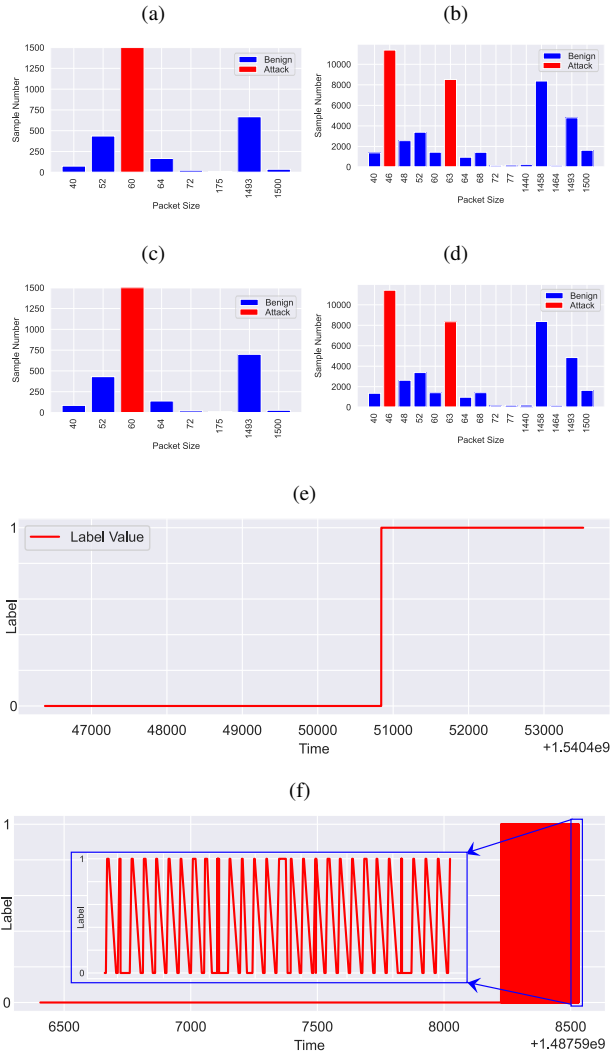
Fig. 7: Complexity analysis of UDP, Mirai, and Video injection attacks according to time and size characteristics. (a) and (b) Mirai, (c) and (d) UDP, (e) and (f) Video injection time and size complexity, respectively. The central number depicted in the figures represents the comprehensive complexity score, which is derived as the average of 22 distinct analytical methods. (The complexity score ranges from 0 to 100, where higher values indicate increasing complexity). These 22 methods stem from six different approaches to assessing complexity, each distinctly colour-coded for clarity of categorisation [22]: red for feature-based, orange for linearity-based, yellow for neighbourhood-based, light green for network-based, dark green for dimensionality-based, and blue for class imbalance-based.
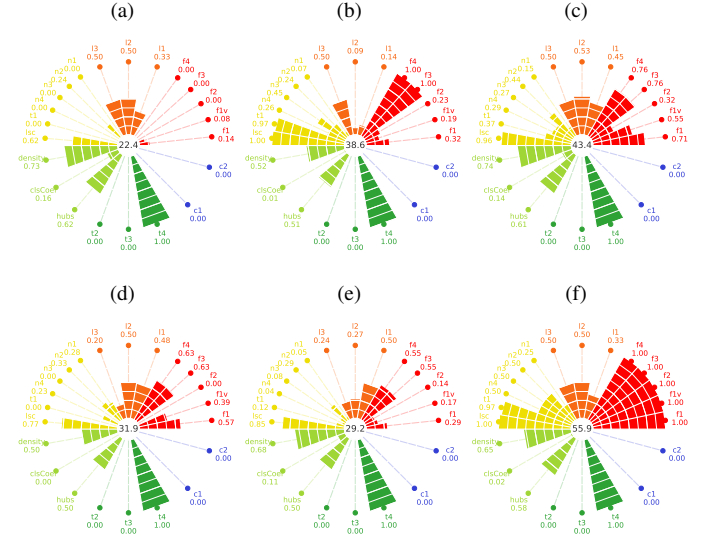
smaller portion of this mixture (the last 2000 samples) reveals a rhythmic pattern between attack and benign traffic over time. The models trained using the time feature for these two attacks perform well on these datasets, but are not capable of detecting the same attack on another dataset or in real-life.

To get a better understanding of the complexity of the data, we calculated the complexity score [22] of the attack files. Figure 7 shows the computation of the complexity scores of 3 different data using time and size characteristics. When these figures are analysed, it can be seen that the UDP flood attack has low size complexity and high time complexity. On the other hand, Mirai and video injection attacks show low size complexity and high time complexity.

Table III shows the results of the attack detection process using only one feature for each of the three attacks with shared complexity values. Upon examination of the CV (10-times, 10-fold) results in Table III, we observe that the UDP attack, characterised by very low complexity in terms of size, achieves flawless detection. Similarly, in the Mirai attack, very low time complexity leads to nearly perfect detection rates. Conversely, the video injection attack, which possesses a relatively higher size complexity but lower time complexity, did not achieve substantial success in terms of size-based detection

Additionally, we present yet another example, Figure 6f, in which all packets up to a certain time demonstrate benign behaviour, while thereafter, a mixture of attack and benign packets is evident. The graph illustrates a dense concentration of data over a brief temporal window. However, focusing on a

TABLE III: Near-perfect classification of 2 datasets afflicted by low data complexity. In Kitsune Mirai and Video injection data, the labels follow a certain time pattern. In the IoT-NID UDP data, attack packets have a specific size. In the Cross-Validation part of the table, the results depict the mean and standard deviation based on 10 repetitions of 10-fold cross-validation (using DT). In the isolated part, the mean and standard deviation are presented from 100 repetitions. Temporal information is not used in order to keep the features at the simplest level. The "time" feature here refers to a timestamp, not a temporal feature.

	Dataset	Subdataset	Feature	ML	Accuracy	Precsion	Recall	F1 Score	Kappa
Cross-validated	Kitsune	Video Inj	Size	DT	0.959±0.000	0.479±0.000	0.500±0.000	0.489±0.000	0.000±0.000
	Kitsune	Mirai	Size	DT	0.899±0.001	0.804±0.002	0.918±0.001	0.842±0.002	0.688±0.003
	IoT-NID	UDP-S2	Size	DT	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000	0.999±0.001
	Kitsune	Video Inj	Time	DT	0.988±0.000	0.933±0.002	0.919±0.002	0.926±0.001	0.852±0.003
	Kitsune	Mirai	Time	DT	0.997±0.000	0.998±0.000	0.990±0.001	0.994±0.000	0.988±0.001
	IoT-NID	UDP-S2	Time	DT	0.841±0.007	0.833±0.008	0.839±0.008	0.835±0.008	0.671±0.016
Isolated	IoT-NID	UDP-S2vsS3	Size	DT	0.032±0.000	0.016±0.000	0.500±0.000	0.031±0.000	0.000±0.000
	IoT-NID	UDP-S2vsS4	Size	DT	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000	0.999±0.000
	IoT-NID	UDP-S2vsS3	Time	DT	0.526±0.000	0.508±0.000	0.566±0.000	0.379±0.000	0.017±0.000
	IoT-NID	UDP-S2vsS4	Time	DT	0.866±0.000	0.858±0.000	0.865±0.000	0.861±0.000	0.722±0.000

but demonstrated significant success when considering time-based detection.

In our second experiment, capitalising on the multi-session nature of the UDP attack, We conducted tests involving the second session, pairing it with the fourth session, which has very similar characteristics. Additionally, we tested the second session with the third sessions, which are notably different from it. The similarities and differences between these sessions can be better understood by examining their distributions in Fig 6. The results, shown under "isolated" in Table III, indicate exceptional performance in the fourth sessions that were similar, while not achieving notable results in the third sessions exhibited substantial dissimilarities from the second session.

What all of these examples have in common is that high levels of success can be achieved by using individual features that are not suitable for attack detection. Although identifying features are not used here, the uniformity of attack data enables even basic individual characteristics to serve as identifying features.

To give more insight into the relationship between complexity and overfitting, we have expanded this sample a little more. Figure 8 shows the results of another experiment on complexity. In this experiment, a basic feature (packet size) was used to separate all attack types in the IoT-NID dataset. From this graph, we can observe the inverse correlation between complexity and success. Especially, if the complexity is below the critical level (45), highly successful discrimination is achieved. On the other hand, as complexity increases, success seems to decrease.

V. CONCLUSION

In summary, this study highlights the limitations of relying solely on individual packet features (IPF) for intrusion detection within IoT environments. While conducted within the IoT domain to emphasise security considerations in IoT devices, the implications extend beyond this realm. Our findings stress the necessity of holistic approaches considering contextual fea-

Fig. 8: Reflection of the change in the complexity value due to a single feature on the model success (with DT).
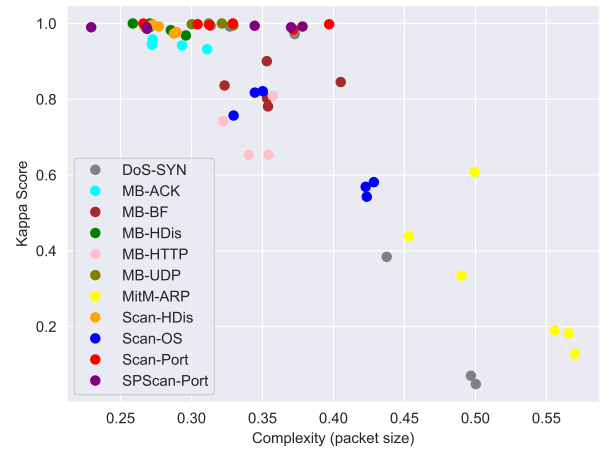
tures and packet interactions for effective intrusion detection in various network security contexts.

Through literature review and experimental analysis, we've shown that although IPF may show high detection rates, they suffer from inherent flaws like information leakage and low data complexity. This underscores the importance of prioritising robust ML-based intrusion detection systems that address the multifaceted nature of security challenges in networked environments.

Moving forward, researchers and practitioners should prioritise comprehensive approaches incorporating contextual features and packet interactions to enhance IoT ecosystem resilience and networked environment security against evolving cyber-threats. By addressing these limitations and embracing sophisticated detection methodologies, we can strengthen defences and mitigate risks posed by malicious actors.

REFERENCES

[1] Y. Harbi, Z. Aliouat, A. Refoufi, and S. Harous, "Recent security trends in internet of things: A comprehensive survey," *IEEE Access*,

vol. 9, pp. 113 292–113 314, 2021, https://doi.org/10.1109/ACCESS.2021.3103725.

[2] G. Kambourakis, C. Kolias, and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," in *MILCOM 2017-2017 IEEE military communications conference (MILCOM)*. IEEE, 2017, pp. 267–272, https://doi.org/10.1109/MILCOM.2017.8170867.

[3] K. Kostas, M. Just, and M. A. Lones, "IoTGeM: Generalizable Models for Behaviour-Based IoT Attack Detection," *arXiv preprint arXiv:2401.01343*, 2023, https://doi.org/10.48550/arXiv.2401.01343.

[4] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSp*, vol. 1, pp. 108–116, 2018, https://doi.org/10.5220/0006639801080116.

[5] D. Zhao, I. Traore, B. Sayed, W. Lu, S. Saad, A. Ghorbani, and D. Garant, "Botnet detection based on traffic behavior analysis and flow intervals," *computers & security*, vol. 39, pp. 2–16, 2013.

[6] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: an ensemble of autoencoders for online network intrusion detection," in *Network and Distributed Systems Security (NDSS) Symposium 2018*, 2018, http://dx.doi.org/10.14722/ndss.2018.23204.

[7] E. Anthi, L. Williams, M. Słowińska, G. Theodorakopoulos, and P. Burnap, "A supervised intrusion detection system for smart home IoT devices," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 9042–9053, 2019, https://doi.org/10.1109/JIOT.2019.2926365.

[8] G. De Carvalho Bertoli, L. A. Pereira Junior, O. Saotome, A. L. Dos Santos, F. A. N. Verri, C. A. C. Marcondes, S. Barbieri, M. S. Rodrigues, and J. M. Parente De Oliveira, "An End-to-End Framework for Machine Learning-Based Network Intrusion Detection System," *IEEE Access*, vol. 9, pp. 106 790–106 805, 2021, https://doi.org/10.1109/ACCESS.2021.3101188.

[9] P. Gulihar and B. B. Gupta, "Cooperative mechanisms for defending distributed denial of service (DDoS) attacks," in *Handbook of Computer Networks and Cyber Security*. Springer, 2020, pp. 421–443, https://doi.org/10.1007/978-3-030-22277-2_16.

[10] Hewlett-Packard-Enterprise, "Single-packet attacks," in *HPE FlexNetwork 7500 Switch Series Security Configuration Guide*. Hewlett Packard Enterprise Development LP, 2017, accessed: 2023-09-17, Available at https://techhub.hpe.com/eginfolib/networking/docs/switches/7500/5200-1952a_security_cg/content/495505992.htm.

[11] A. Ghourabi, "A security model based on lightgbm and transformer to protect healthcare systems from cyberattacks," *IEEE Access*, vol. 10, pp. 48 890–48 903, 2022, https://doi.org/10.1109/ACCESS.2022.3172432.

[12] M. A. Ferrag, O. Friha, D. Hamouda, L. Maglaras, and H. Janicke, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, vol. 10, pp. 40 281–40 306, 2022, https://doi.org/10.1109/ACCESS.2022.3165809.

[13] N. Saran and N. Kesswani, "A comparative study of supervised machine learning classifiers for intrusion detection in internet of things," *Procedia Computer Science*, vol. 218, pp. 2049–2057, 2023, https://doi.org/10.1016/j.procs.2023.01.181.

[14] O. Friha, M. A. Ferrag, L. Shu, L. Maglaras, K.-K. R. Choo, and M. Nafaa, "Felids: Federated learning-based intrusion detection system for agricultural internet of things," *Journal of Parallel and Distributed Computing*, vol. 165, pp. 17–31, 2022, https://doi.org/10.1016/j.jpdc.2022.03.003.

[15] Y. Chen, Q. Lin, W. Wei, J. Ji, K.-C. Wong, and C. A. C. Coello, "Intrusion detection using multi-objective evolutionary convolutional neural network for internet of things in fog computing," *Knowledge-Based Systems*, vol. 244, p. 108505, 2022, https://doi.org/10.1016/j.knosys.2022.108505.

[16] X. Han, S. Cui, S. Liu, C. Zhang, B. Jiang, and Z. Lu, "Network intrusion detection based on n-gram frequency and time-aware transformer," *Computers & Security*, vol. 128, p. 103171, 2023, https://doi.org/10.1016/j.cose.2023.103171.

[17] B. Mondal and S. K. Singh, "A comparative analysis of network intrusion detection system for IoT using machine learning," in *Internet of Things and Its Applications: Select Proceedings of ICIA 2020*. Springer, 2022, pp. 211–221, https://doi.org/10.1007/978-981-16-7637-6_19.

[18] E. Anthi, L. Williams, A. Javed, and P. Burnap, "Hardening machine learning denial of service (DoS) defences against adversarial attacks in IoT smart home networks," *computers & security*, vol. 108, p. 102352, 2021, https://doi.org/10.1016/j.cose.2021.102352.

[19] K. Hyunjae, A. D. Hyun, L. G. Min, Y. J. Do, P. K. Ho, and K. H. Kang, "IoT network intrusion dataset," accessed: 2023-09-15, https://dx.doi.org/10.21227/q70p-q449. [Online]. Available: https://dx.doi.org/10.21227/q70p-q449

[20] N. Koroniotis, N. Moustafa, E. Sitnikova, and B. Turnbull, "Towards the development of realistic botnet dataset in the internet of things for network forensic analytics: Bot-IoT dataset," *Future Generation Computer Systems*, vol. 100, pp. 779–796, 2019, https://doi.org/10.1016/j.future.2019.05.041.

[21] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multidimensional IoT profiling dataset," in *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*, 2022, pp. 1–11, https://doi.org/10.1109/PST55820.2022.9851966.

[22] A. C. Lorena, L. P. Garcia, J. Lehmann, M. C. Souto, and T. K. Ho, "How complex is your classification problem? a survey on measuring classification complexity," *ACM Computing Surveys (CSUR)*, vol. 52, no. 5, pp. 1–34, 2019, https://doi.org/10.1145/3347711.