Exposing Spurious Features in Traffic Anomaly Detectors for the 5G Core Network

Cristian Manca*, Christian Scano*†, Giorgio Piras*, Maura Pintor*, Fabio Brau*, Battista Biggio*
*University of Cagliari
†University of Rome Sapienza

***Abstract*—Machine learning models are increasingly used for network anomaly detection, yet their decision processes often remain difficult to interpret. In particular, models trained on high-dimensional network data may rely on features that are either spurious or easily manipulable, potentially leading to fragile detection behavior. In this work, we investigate how explainability techniques (XAI) can be used to analyze feature reliance in network anomaly detection models. Using a 5G core network traffic dataset, we train detection models under different preprocessing conditions and examine their behavior through model explanation methods. Our analysis reveals that models trained on network traffic rely heavily on environmental features, such as IP addresses and ports, rather than the actual attack behavior. Because of this, we show that an attacker can easily bypass detection by making protocol-compliant modifications to these specific features. We further show that appropriate feature preprocessing mitigates this vulnerability, forcing the models to rely on the true malicious behavior of the attacks. Overall, our results highlight the role of explainability as a diagnostic tool for understanding model behavior and identifying potential weaknesses in machine learning-based anomaly detection systems.**

***Index Terms*—anomaly detection, explainability, 5G**

I. Introduction

The 5G Core (5GC) network is the central component of the 5G architecture, responsible for critical control-plane and user-plane functions. Its cloud-native, service-based architecture generates massive and heterogeneous traffic volumes, making traditional rule-based security monitoring approaches ineffective. To address this complexity, Machine Learning (ML)-based anomaly detection systems are widely adopted to detect malicious activity and protect network infrastructure [1].

Their widespread use is often driven by evaluations in simulated environments, where models achieve near-perfect performance during the design phase but fail to reflect real-world behavior. In these settings, models tend to rely on spurious correlations (*e.g.*, specific IP addresses or ports) that are tightly coupled with the experimental setup rather than with the underlying attack dynamics. For instance, a model may associate malicious activity with a fixed source IP observed during simulation; however, in real deployments, IP addresses vary across networks or can be easily changed by adversaries. While such correlations yield high detection rates in controlled settings, they do not generalize: real networks differ in configuration, traffic patterns evolve, and attackers can deliberately manipulate these features [2]. As a result, detectors trained in simulation overfit the environment, learning to memorize setup-specific artifacts instead of capturing meaningful attack behavior, and thus become unreliable when deployed in the wild.

This reliance on environmental features also introduces a critical security vulnerability, considering that machine learning models are inherently susceptible to adversarial attacks [3]. In this context, an adversary is an attacker who deliberately manipulates the observable fields of the network traffic packets to evade detection. By applying protocol-compliant perturbations to the unprotected fields of the packets, the adversary can exploit these surface features to successfully evade detection with minimal effort, all without altering the underlying functionality of the cyberattack [4]. For this reason, deploying machine learning-based detectors in security-critical environments such as 5GC networks requires not only evaluating their performance but also understanding the features that drive their predictions.

In this work, we use explainability techniques (eXplainable AI – XAI) to analyze feature dependence in anomaly detection models for 5GC networks and systematically expose their vulnerabilities to adversarial manipulation. Using a realistic dataset of 5GC network traffic, we train an extensive set of anomaly detectors and analyze their decision-making processes using SHapley Additive exPlanations (SHAP) [5].

We demonstrate that models trained on raw data, that is, unfiltered traffic containing spurious correlations, tend to rely heavily on features that are easily manipulated by potential attackers. We further demonstrate that domain-aware preprocessing, which leverages protocol-specific knowledge to filter out these environmental artifacts, mitigates this problem and encourages models to focus on behavioral characteristics of the traffic.

Finally, we show that global explainability serves as a diagnostic tool, facilitating the finding of adversarial perturbations and guiding feature preprocessing to improve model robustness.

II. Methodology and background

We first outline the process of training a machine learning (ML) model for anomaly detection from network traffic data, which naturally leads to the question of how such models make their decisions. To address this, we leverage explainability techniques to analyze trained detectors and identify a set of features that drive their predictions despite not being associated with attack behavior. This insight, in turn, can

guide preprocessing strategies to improve the training data by removing the spurious features from training.

A. Problem Setup

The anomaly detection pipeline begins with the collection of a raw network dataset collected from a simulation testbed, which we denote by $\mathcal{P}$. This dataset consists of unfiltered packets intercepted directly from the testbed's network interfaces. Each packet $\mathbf{p}$ is processed by a feature extraction function $\mathcal{F}$, which maps it to a numerical feature vector $\mathcal{F}(\mathbf{p}) = \mathbf{x} \in \mathbb{R}^d$. During training, an anomaly detector D learns a basic representation of normal traffic using these feature vectors. Once deployed, the detector evaluates incoming packets and flags them as malicious if their anomaly score exceeds a predefined detection threshold τ, i.e., $\mathcal{D}(\mathcal{F}(\mathbf{p})) \geq \tau$.

However, models trained directly on the raw dataset $\mathcal{P}$ are highly susceptible to over-relying on strong correlations between spurious features and labels to make decisions. This occurs because standard algorithms minimize a loss function without specific constraints, failing to account for unrealistic data modeling and potential attacks. Because packet fields perfectly encapsulate the topological configuration of the simulated environment, the detector often achieves high detection rates simply by memorizing environmental artifacts (such as source IP addresses or ephemeral ports) rather than learning the structural properties of attacks. Furthermore, this phenomenon exposes the models to trivial adversarial evasion attacks, since modifying these features is often operationally simple for an attacker, posing a particular interest in understanding what drives the models' decisions.

B. Methodological Approach

To systematically investigate the phenomenon of spurious features, explainability techniques can be applied to the trained detectors. By simply querying the model and analyzing how variations in the input features affect the output score, these techniques compute the relative importance of each feature in driving the model's predictions. This allows defenders to look beyond the aggregate detection rates and verify whether the detector is genuinely identifying malicious behavior or merely exploiting spurious correlations present in the raw data $\mathcal{P}$.

To mitigate the reliance on these environmental shortcuts, we apply a domain-aware preprocessing strategy following the security-aware guidelines proposed in [6]. This approach leverages specific knowledge of the 5G Core protocols to intentionally filter out topology-specific identifiers and redundant fields from the raw dataset $\mathcal{P}$. This transformation yields a refined, environment-independent dataset, where setup-specific artifacts such as IP addresses and ports are explicitly removed, denoted as $\mathcal{P}_f$ We employ SHapley Additive exPlanations (SHAP) to extract feature importance in both scenarios. By comparing the explanations of models trained on $\mathcal{P}$ with those trained on $\mathcal{P}_f$, we formally validate the preprocessing step, ensuring that the anomaly detectors trained on $\mathcal{P}_f$ shift their focus from topological artifacts to the actual behavioral characteristics of the traffic, thus becoming more reliable.

Furthermore, to evaluate the security implications and the robustness of the detectors under both configurations, we extend the model-agnostic threat model formulated in [6] by introducing an XAI-informed attacker. We assume a strict compliance setting: given a correctly detected malicious packet $\mathbf{p}$ (where $\mathcal{D}(\mathcal{F}(\mathbf{p})) \geq \tau$), the attacker aims to craft an adversarial packet, i.e., a packet $\mathbf{p}'$ that evades detection ($\mathcal{D}(\mathcal{F}(\mathbf{p}')) < \tau$) while remaining perfectly compliant with the original protocol ($\mathbf{p}' \equiv \mathbf{p}$). This strict compliance restricts the attacker to modifying only a specific subset of feasible, unprotected features, denoted as $J \subseteq \mathbb{R}^d$. Note that, while we model an attacker with access to XAI-based feature importance, which would require querying the model many times, the pipeline is more realistically interpreted as a defender's diagnostic tool: by revealing which features drive detection, the attack guides the preprocessing strategy needed to eliminate spurious correlations and harden the model against worst-case adversaries.

Rather than using random or genetic algorithms to select the features to modify (as in [6]), the attack leverages SHAP as a global explainability tool to strategically guide the evasion. By analyzing the SHAP values, the attack extracts a highly restricted subset $S \subseteq J$ containing only the most influential features driving the detection. Furthermore, SHAP reveals the directional impact of these features, allowing the attacker to infer a target direction $s_i \in \{-1, 1\}$ for each feature $i \in S$, indicating whether an increase or decrease in its value will push the model's prediction toward the benign class.

The adversarial evasion is therefore formulated as the following directionally-constrained minimization problem:

$$\begin{aligned} \min_{\mathbf{p}' \equiv \mathbf{p}} \quad & [\mathcal{D}(\mathcal{F}(\mathbf{p}')) - \tau]_+ \\ \text{s.t.} \quad & \mathcal{F}(\mathbf{p})_i = \mathcal{F}(\mathbf{p}')_i, \quad \forall i \notin S \\ & s_i \cdot (\mathcal{F}(\mathbf{p}')_i - \mathcal{F}(\mathbf{p})_i) \geq 0, \quad \forall i \in S \end{aligned} \tag{1}$$

By restricting the optimization space to the XAI-identified subset S and guiding the perturbations along the vulnerability directions s_i, we demonstrate how an attacker can efficiently solve the minimization problem and bypass the detector, without ever altering the underlying malicious functionality.

III. Experimental Setup

This section describes the dataset, features, models, and explainability techniques used in our analysis.

A. Dataset and Features

In our experiments, we use the 5G-Attacks dataset [7], which contains labeled network traffic with realistic attack scenarios targeting the 5G Core network.[1] The dataset includes five PFCP-based attacks exploiting the lack of authentication, integrity protection, and strict input validation in PFCP signaling, generated by manipulating protocol fields while preserving syntactic validity. The attack categories are:

[1] https://github.com/clem272001/5G-Attacks

Characteristics	Train Dataset	Validation Dataset	Test Dataset
Packets			
Normal	21341	4731	4732
PFCP Restoration-TEID	0	13	22
PFCP Flood	0	1039	1026
PFCP Deletion	0	7	13
PFCP Modification	0	16	12
UPF PDN-0 Fault	0	10	12

TABLE I: Sample Distribution of Train, Validation, and Test datasets.

Protocol	Before Preprocessing	After Preprocessing
Number of Features		
IP	16	4
UDP	9	1
TCP	37	0
PFCP	45	28

TABLE II: Number of Features before and after applying Preprocessing

PFCP Restoration-TEID. Injects forged restoration messages with invalid TEID and F-TEID values, causing desynchronization between control and user planes, leading to misrouting, session drops, or UPF crashes.
PFCP Flood. A denial-of-service attack that floods PFCP endpoints with unsolicited messages, exhausting control-plane resources and degrading session management.
PFCP Deletion. Forged deletion requests remove active session state, abruptly interrupting user-plane traffic.
PFCP Modification. Malicious modification messages alter forwarding behavior, causing packet drops, misrouting, or invalid tunneling.
UPF PDN-0 Fault. Exploits insufficient validation of PDN Type 0 contexts, leading to inconsistent PFCP states, session failures, or traffic misrouting.

Starting from the original dataset, we reorganized it into three distinct subdivisions: an unsupervised training set containing only normal traffic, a validation set, and a test set, both including both normal samples and representative attack samples to evaluate detection performance. The detailed distribution of samples across normal traffic and different attack categories for each subdivision is shown in Table I.

The raw dataset consists of packet-level features extracted via `tshark`, across multiple protocol layers, including IP, UDP, TCP, and PFCP. These features capture both generic network characteristics (e.g., header lengths, flags, timestamps) and protocol-specific information, such as PFCP identifiers and TEID-related fields, which are particularly relevant for modeling control-plane behavior in the 5G Core.

To focus our analysis on relevant control-plane traffic, and in line with the focus on PFCP-based attacks, we filter out all TCP and ICMP packets, retaining only UDP flows. Table II summarizes the feature distribution across protocols (excluding TCP, since PFCP communication in the 5GC relies on UDP) before and after applying our preprocessing pipeline, reducing the feature space and preserving only the relevant and discriminative information for anomaly detection.

B. Models

We consider a diverse set of state-of-the-art anomaly detectors, imported from the `PyOD` library [8], as well as custom pipelines tailored for the 5G network data. Specifically, we evaluate: `COPOD` [9], `ECOD` [10], `GMM` [11], `HBOS` [12], `Isolation Forest` [13], `LOF` [14], and `PCA` [15]. For each detector, we tune the hyperparameters via grid search on the validation set. For each model, the grid search spans detector-specific parameters such as neighborhood size for density-based methods (e.g., k in `LOF`), number of bins or splits for histogram/statistical models (e.g., `HBOS`), and contamination (expected outlier proportion). Given the restricted number of pages, full results are reported only for some illustrative models in the following sections, considering that the shown behavior is reflected overall.

C. Explainability Method

To understand what drives the models' decisions and to identify the most relevant features to detect the malicious behavior, we rely on the SHAP Explainability method [2]. Specifically, we employ the `KernelExplainer`, which is perfectly aligned with our proposed threat model as it operates in a strictly *model-agnostic* manner.

Unlike other explainability algorithms (such as `TreeExplainer`, `LinearExplainer`, or `DeepExplainer`) that require white-box access to the model's internal architecture, weights, or gradients, the `KernelExplainer` treats the anomaly detector as a pure black box and only requires access to the prediction function (i.e., the anomaly score output). By systematically generating perturbations on the input features and observing the corresponding changes in the detector's output score, it accurately infers the SHAP values through feature observation alone.

In SHAP, the `KernelExplainer` requires a background dataset to establish a baseline (i.e., the expected model output). This background acts as a reference distribution representing typical feature values. SHAP values are then computed by measuring how much each feature of a given sample contributes to pushing the model's output away from this baseline expectation. In our setup, global explainability is first performed on 500 attack samples from the test set, utilizing a background of 100 benign instances randomly sampled from the training set to compute the expected base score. By defining normal traffic as the baseline, SHAP accurately evaluates the deviations that cause a sample to be flagged as an anomaly. This global analysis reveals which features generally drive the decisions of the models, highlighting the attributes with the highest contributions in malicious samples and their reference values. Then, after using these results to manipulate attack samples, we conduct an in-depth local explainability analysis over both the original attack and its manipulated

[2]https://github.com/shap/shap

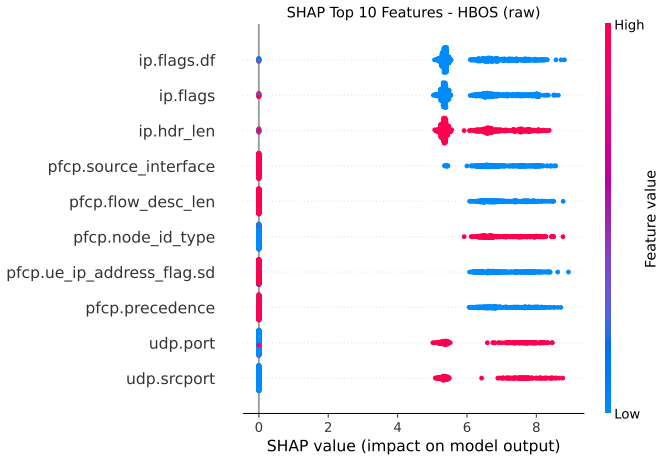

Fig. 1: SHAP Global Explainability's Top-10 HBOS Raw Features

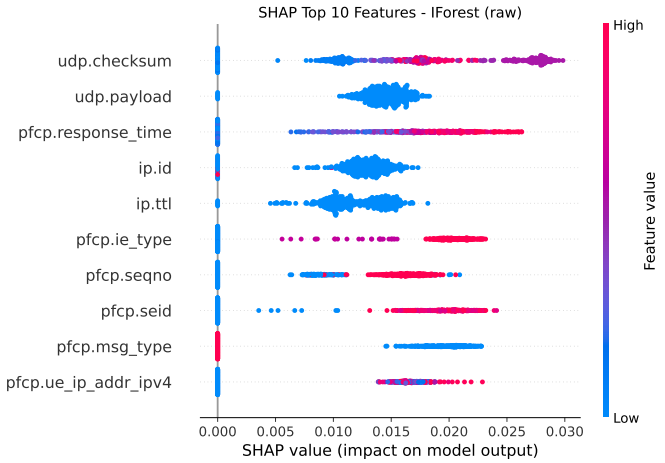

Fig. 2: SHAP Global Explainability's Top-10 IForest Raw Features

counterpart to formally evaluate the effect of the adversarial perturbation.

IV. Explainability Analysis

This section reports and describes the conducted explainability analysis.

A. Feature Importance in Raw Models

The results obtained from the SHAP Global Explanation over attack samples on raw models are illustrated in Fig. 1 for `HBOS` and Fig. 2 for `IForest`. Observing the top-10 most relevant features presented in these figures, we note a clear manifestation of shortcut learning. The detectors assign high importance to characteristics that are absolutely uncorrelated with the actual attack behavior. For instance, `HBOS` relies heavily on topological and environmental artifacts such as `ip.flags`, `ip.flags.df`, and `udp.srcport`. Similarly, `IForest` strongly depends on `udp.payload`. These features reflect the specific testbed configuration and network topology rather than the intrinsic structural anomalies of the cyberattacks. Because these superficial identifiers act as perfect discriminators in the raw dataset, the models naively overfit the simulated environment to achieve high performance.

Furthermore, most of these fields are trivially controllable by an attacker in a real-world scenario. For instance, features like `ip.flags` (e.g., the Don't Fragment bit) or `udp.srcport` belongs to the unprotected IP and UDP headers encapsulating the malicious payload. An attacker crafting a malicious packet can freely forge these values using standard network tools without violating protocol specifications. Since these modifications occur at the network or transport layers, they do not interfere with the underlying PFCP exploits. Consequently, an adversary can easily assign these fields the specific values expected by the detector for benign traffic, successfully bypassing the security system while fully preserving the attack's semantics and functionality.

B. Feature Manipulation Effects

To systematically analyze the effects of feature manipulation, we must first define the attacker's capabilities, identifying which fields can be arbitrarily modified without invalidating the network packet or its malicious behavior. Once these constraints are established, we apply targeted perturbations to the modifiable features that SHAP global explainability identified as highly relevant. We optimize the direction of these modifications to maximize the impact on the model's decision, following the minimization problem introduced in Equation 1.

Subsequently, we perform SHAP local explainability on both the original, correctly detected attack samples and their evaded adversarial counterparts. As illustrated in Fig. 3 and Fig. 4, the attacker can successfully alter the model's prediction by manipulating irrelevant but easily modifiable features, such as `ip.id`, `udp.checksum`, `pfcp.ie_type`, `pfcp.response_time`, `pfcp.ue_ip_addr_ipv4`, fooling the detector and fully preserving the attack semantics.

To further quantify the model's vulnerability to such adversarial modifications, we evaluate its robustness using an L_0 norm constraint, which defines the maximum number of perturbed features allowed k. As illustrated in Fig. 5, we evaluate the detection rate across four different strategies:

- *Random Baseline (All Features):* A naive, single-shot approach in which feature values are randomly changed across the entire available feature space.
- *Genetic (All Features):* A standard genetic algorithm that optimizes perturbations over a restricted computational budget of 100 iterations, without using any a priori explainability knowledge.
- *Genetic + SHAP Directions:* Operating under the same 100-iteration budget, the genetic algorithm is restricted to features with non-zero SHAP relevance, optimizing perturbations strictly along the specific direction indicated

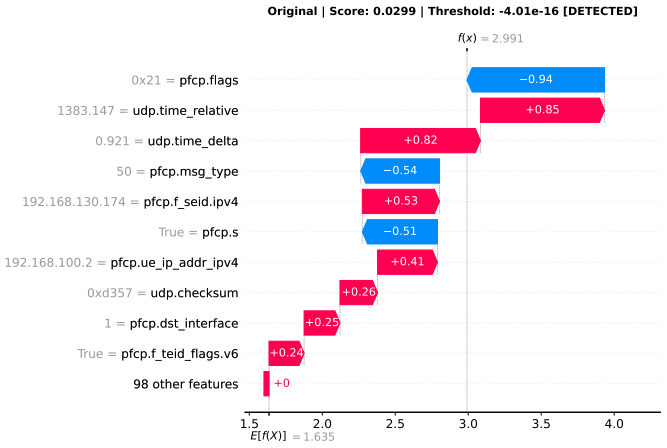

Fig. 3: Top 10 Isolation Forest Raw Features According to SHAP Local Explainability on the Original Attack Sample

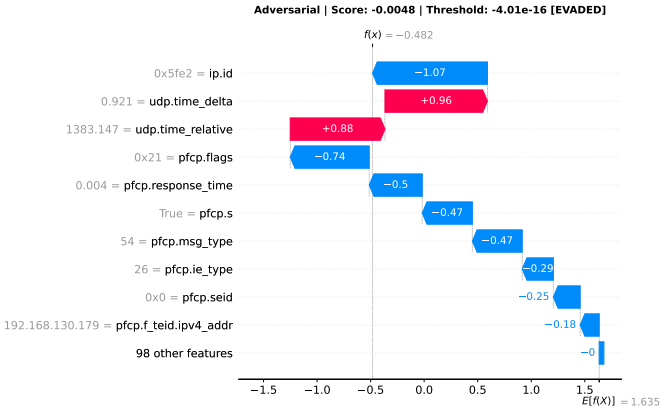

Fig. 4: Top 10 Isolation Forest Raw Features According to SHAP Local Explainability on the Adversarial Attack Sample

by their SHAP impact.

- *Genetic + SHAP Directions (Top-10 Features):* A highly constrained attack, also restricted to 100 iterations, in which SHAP-directed genetic optimization is rigorously applied only to the 10 most relevant features.

The robustness curves clearly demonstrate that blind perturbation strategies are highly ineffective. The random baseline model fails to evade the model, while the standard genetic algorithm encounters significant difficulty; it fails to find an effective evasion path during 100 iterations, resulting in a negligible reduction in detection rate.

In contrast, incorporating SHAP directions into genetic optimization drastically compromises the model's robustness, achieving high evasion rates despite the limited computational budget. By targeting only the top-10 features, the attacker can only manipulate up to $k = 7$ features to bring the detection rate close to 0. Similarly, applying SHAP directions to all features with SHAP values results in a near-zero detection rate within $k = 20$ changes. This highlights that an attacker with interpretability data can effectively evade detectors by

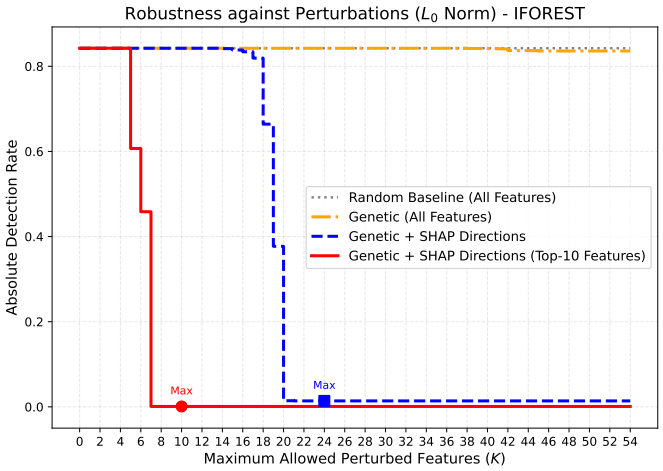

Fig. 5: Isolation Forest robustness under L_0 constraints. Detection Rate vs. maximum perturbed features (k).

perturbing a small subset of carefully selected features.

C. Effect of Feature Preprocessing

Raw packet features often reflect network topology rather than actual malicious behavior. To prevent models from relying on these spurious patterns, applying domain-aware feature preprocessing is a critical step. As introduced in Sect. II-B, our preprocessing strategy and the adversarial threat model build upon the guidelines proposed in [6]. However, while that study demonstrated the vulnerability of anomaly detectors using black-box techniques, it treated the models as opaque systems without investigating the underlying decision-making mechanisms that led to such vulnerabilities. In this work, we bridge this gap through an explainability-driven methodology. We employ SHAP not only to deterministically guide adversarial evasion, but, crucially, to formally validate the necessity and effectiveness of the preprocessing step.

To demonstrate this, we perform SHAP global explainability on models trained with the preprocessed dataset. As shown in Fig. 6 and Fig. 7, the most relevant features change dramatically compared to the raw models. The highlighted features are better related to the actual packet behavior and protocol semantics rather than the network topology, visually and formally confirming that the filtered detectors have successfully shifted their focus toward more relevant patterns.

V. Conclusion

In this work, we investigate how the nature of training data impacts the security and reliability of ML-based anomaly detectors in the 5G Core network. Specifically, we analyze the critical differences between models trained on raw traffic generated in simulated test environments and those trained on the same data after applying appropriate domain-specific feature preprocessing. Using a model-independent explainability technique, we show how models trained on raw network traffic achieve high detection by relying primarily on spurious environmental artifacts, rather than identifying behavioral patterns

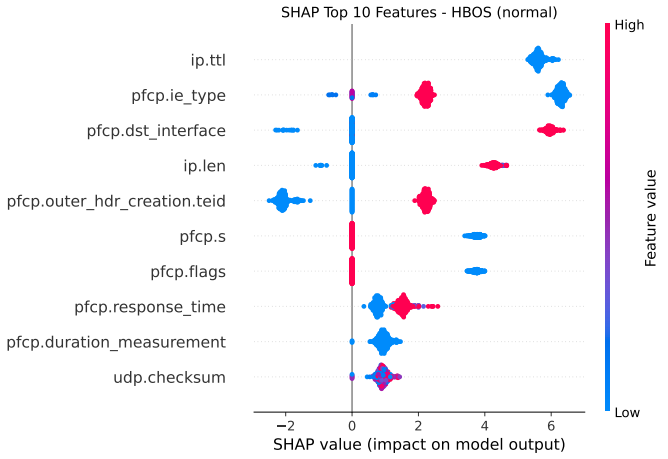

Fig. 6: SHAP Global Explainability's Top-10 HBOS Features

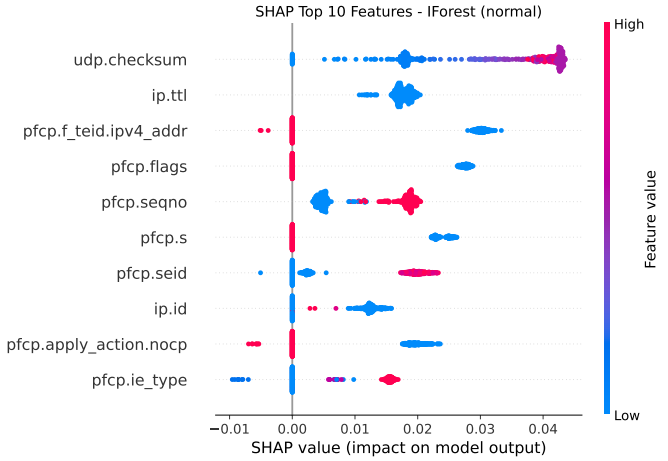

Fig. 7: SHAP Global Explainability's Top-10 IForest Features

of cyberattacks. We also show that explainability analysis can be used to automatically identify a restricted subset of spurious features driving the model's decision, providing defenders with a practical debugging tool to guide targeted preprocessing and improve the system's robustness. Finally, we provide empirical evidence that these vulnerabilities can be effectively mitigated through rigorous, domain-aware feature preprocessing. By enforcing environment independence and removing topology-specific identifiers during the training phase, we force the anomaly detectors to base their decisions on protocol-specific semantics, improving their robustness against adversarial evasion.

Based on these findings, future research could explore the integration of interpretability-based insights to actively improve the robustness of the model.

Acknowledgments

This work was partially supported by the EU-funded project Sec4AI4Sec (grant no. 101120393). This work was carried out while C. Scano was enrolled in the Italian National Doctorate on AI run by the Sapienza University of Rome in collaboration with the University of Cagliari.

References

[1] J. Lam and R. Abbas, "Machine learning based anomaly detection for 5g networks," *arXiv preprint arXiv:2003.03474*, 2020.

[2] D. Arp, E. Quiring, F. Pendlebury, A. Warnecke, F. Pierazzi, C. Wressnegger, L. Cavallaro, and K. Rieck, "Dos and don'ts of machine learning in computer security," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 3971–3988.

[3] B. Biggio and F. Roli, "Wild patterns: Ten years after the rise of adversarial machine learning," *Pattern Recognition*, vol. 84, pp. 317–331, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320318302565

[4] G. Apruzzese, R. Vladimirov, A. Tastemirova, and P. Laskov, "Wild networks: Exposure of 5g network infrastructures to adversarial examples," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 5312–5332, 2022.

[5] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions." Curran Associates, Inc., 2017. [Online]. Available: http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[6] C. Manca, C. Scano, G. Piras, F. Brau, M. Pintor, and B. Biggio, "Sage-5gc: Security-aware guidelines for evaluating anomaly detection in the 5g core network," *arXiv preprint arXiv:2602.03596*, 2026.

[7] C. Duchesne, A. Blaise, P. Velloso, and V. Conan, "5g-attacks: A new dataset from realistic 5g-core attacks," 12 2025, pp. 92–96.

[8] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019. [Online]. Available: http://jmlr.org/papers/v20/19-011.html

[9] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu, "Copod: copula-based outlier detection," in *2020 IEEE international conference on data mining (ICDM)*. IEEE, 2020, pp. 1118–1123.

[10] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. H. Chen, "Ecod: Unsupervised outlier detection using empirical cumulative distribution functions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 12, pp. 12 181–12 193, 2022.

[11] C. C. Aggarwal, "Probabilistic and statistical models for outlier detection," in *Outlier analysis*. Springer, 2016, pp. 35–64.

[12] M. Goldstein and A. Dengel, "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm," *KI-2012: poster and demo track*, vol. 1, pp. 59–63, 2012.

[13] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 eighth ieee international conference on data mining*. IEEE, 2008, pp. 413–422.

[14] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 93–104.

[15] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," in *IEEE Foundations and New Directions of Data Mining Workshop, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03)*, 2003.