Irreconcilable Differences? Investigating Consensus of Post-hoc XAI for ML-NIDS via Decomposition

Katharina Dietz*, Johannes Schleicher**, Stefan Geißler*, Michael Seufert**, Tobias Hoßfeld*
*University of Würzburg, Germany, **University of Augsburg, Germany
*{katharina.dietz, stefan.geissler, tobias.hossfeld}@uni-wuerzburg.de, **{johannes.schleicher, michael.seufert}@uni-a.de

***Abstract*—Explainable Artificial Intelligence (XAI) is essential for the acceptance of machine learning (ML) models, especially in critical domains like network security. Administrators need interpretable explanations to validate decisions, yet existing XAI methods often suffer from low consensus, where different techniques yield conflicting explanations. A key factor contributing to this issue is the presence of correlated features, which allows multiple equivalent but divergent explanations. While decorrelation techniques, such as Principal Component Analysis (PCA), can mitigate this, they often reduce interpretability by abstracting original features into complex combinations. This work investigates whether feature decorrelation via decomposition techniques can improve consensus among post-hoc XAI methods in the context of ML-based network intrusion detection (ML-NIDS). Using both NIDS and synthetic data, we analyze the effect of decorrelation across different models and preprocessing. We find that decorrelation can significantly improve consensus, but its effectiveness is highly dependent on the underlying model, preprocessing, and dataset characteristics. We also explore sparsity-inducing variants of PCA to partially recover interpretability, though results vary depending on the level of sparsity enforced.**

***Index Terms*—Machine Learning, Intrusion Detection, Explainable AI, Disagreement Problem, Decorrelation, Decomposition, PCA, MCA, XAI, ML, IDS, NIDS.**

I. Introduction

The increasing reliance on machine learning (ML) models in network security, e. g., [1]–[3], has brought significant advancements in automating complex tasks like intrusion detection, threat analysis, and anomaly detection. Yet, adoption of such ML-based solutions by network administrators depends strongly on their ability to comprehend the underlying decision-making processes [4]. Explainable Artificial Intelligence (XAI) [5], [6] has thus emerged as a critical requirement in this domain. In network security, where decisions can have severe implications, administrators are understandably reluctant to rely on opaque black-box models. Beyond trust, explainability also provides insights that help administrators learn from the decisions made by these models, further enriching their expertise and improving overall security practices.

A diverse array of XAI methods has been developed to address the interpretability of ML models (e. g., [7]–[9]). These methods are often tailored to specific contexts, like the type of problem (classification, regression), model (white-box, black-box), or explanation (local, global). An important challenge arises in situations where multiple XAI techniques can be applied: the explanations generated by these methods often lack consensus [10]. High consensus is essential for generating reliable explanations that users act upon confidently.

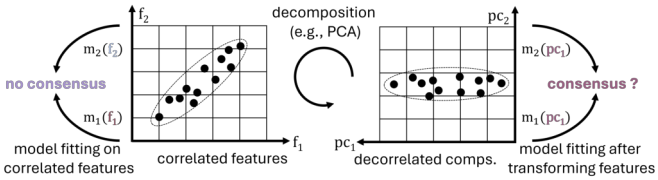


Fig. 1: Decorrelation and XAI consensus.

As shown by Krishna et al. [11], conflicting explanations undermine trust and create uncertainty about the reliability of these methods, particularly in critical domains like network security. Additionally, high disagreements enable adversaries to manipulate explanations as they desire [12], [13].

One possible reason for this lack of consensus is the presence of correlated features in the dataset. Correlated features enable multiple information-equivalent explanations, allowing models to produce different but valid explanations, depending on their internal structures and optimization strategies. While feature decorrelation via data transformations, e. g., Principal Component Analysis (PCA), or via selection of a subset of uncorrelated features (feature selection) is generally recommended in data preprocessing to improve model performance and reduce model complexity, there has been limited exploration of its impact on improving XAI consensus.

In this work, we test the hypothesis that removing correlations through decomposition techniques can improve the consensus of XAI methods. Figure 1 illustrates this idea, where two explainers m_1 and m_2 select two different, but correlated, features for explaining a decision (f_1 and f_2). After applying feature decomposition, these two correlated features are projected into the same principal component (PC), thus the two explainers can more easily agree. To this end, we apply different methods such as PCA and Multiple Correspondence Analysis (MCA), each requiring distinct preprocessing steps. Since decompositions can reduce interpretability by mixing all features into dense components, we also investigate sparsity-inducing variants. While this yields more interpretable components, it may also reintroduce correlations due to the loss of orthogonality. By doing so, we aim to strike a balance between improving consensus and maintaining interpretability.

Our key contributions are threefold: (*i*) we evaluate the impact of feature decorrelation on XAI consensus for both use-case-specific intrusion detection and configurable synthetic datasets, (*ii*) we compare different decorrelation pipelines, including PCA- and MCA-based approaches, and (*iii*) we

978-3-903176-75-1 ©2025 IFIP

investigate the trade-off between interpretability and consensus when applying sparse variants of PCA. This complements our previous work on the disagreement problem in NIDS, where (*a*) we conducted a comprehensive analysis across diverse XAI methods and models [10], and (*b*) we investigated how feature selection strategies influence consensus [14]. Our code, as well as supplementary material, is publicly available[1].

Our work is structured as follows: Section II provides information on NIDS, XAI, and feature decomposition, while Section III describes related works. The proposed methodology, including datasets, ML workflow, and consensus metrics, is detailed in Section IV. Results of our study are presented in Section V. Lastly, Section VI highlights the key findings.

II. Background

a) ML-based NIDS: Cyber intrusions refer to actions that undermine the *confidentiality*, *integrity*, or *availability* (so-called CIA triad) of systems and data [15]. To mitigate intrusions, detection systems (IDS) and prevention systems (IPS) are implemented. These systems can function either at host-level (HIDS) to inspect local activities, or at network-level (NIDS) to analyze network traffic. Network data may be logged with varying degrees of detail, e. g., per packet or per flow. Here, ML-NIDS have experienced increased attention in recent years [4]. Although such models often outperform traditional NIDS, their black-box nature limits interpretability. In critical areas (like NIDS), such black-boxes may hinder practical adoption. Thus, trust-building measures become essential, particularly in light of legal requirements like the European AI Act[2] and the GDPR's "right to explanation" [16].

b) Explainable AI: Model explainability strengthens required trust, ultimately leading to the emergence of XAI. Khani et al. [17] categorize XAI by four characteristics: their *scope*, *stage*, *compatibility*, and *algorithm type*. The *scope* can be local or global, i. e., explain the outcome of a single prediction or the overall model. The *stage* distinguishes ante-/in-hoc XAI suitable for inherently interpretable white-box models before/during training, or post-hoc XAI which explains black-box models after training. *Compatibility* specifies if the explainer can be applied to any model or has constraints. Last, the *type* concerns how the explanation is generated. The two most common types are perturbation- and gradient-/backpropagation-based approaches. The former generates explanations by concealing or permuting input features, while the latter makes use of gradients, e. g., in Neural Networks (NNs). Thus, perturbation-based approaches also tend to be more model-agnostic compared to gradient-based ones [18].

c) LIME and SHAP: Two prominent XAI approaches are Local Interpretable Model-agnostic Explanations (LIME) [7] and SHapley Additive exPlanations (SHAP) [8]. Both are suited for local post-hoc explanations and are generally model-agnostic. Though, SHAP also provides global explanations. LIME works by constructing an interpretable linear proxy model by learning the original model's decision via input perturbations (e. g., sampling new feature values), whereas SHAP utilizes the concept of game-theoretic *Shapley values* [19], which assigns each feature its contribution to the model's prediction based on all feature "coalitions". As mentioned, SHAP (KernelSHAP) is generally model-agnostic, and, similar to LIME, makes use of a linear model. Though, more efficient model-specific variants exist, such as TreeSHAP and DeepSHAP. The former is tailored to internals of tree-based models to compute exact SHAP values, while the latter leverages the gradient-based DeepLIFT [9] to *approximate* SHAP for Deep NNs (DNNs).

d) Disagreement Problem: One challenge that arises with XAI was first coined "disagreement problem" by Krishna et al. [11] and is often mentioned alongside the *Rashomon set*, i. e., "the existence of multiple well-performing models for a given task" [20, p. 2]. This problem describes the challenge that XAI methods provide diverging explanations, making them seem unreliable. This can even happen if the same model and same explainer is used, simply due to different random seeds [21]. This phenomenon has been observed in many areas, ranging from malware analysis [22]–[24], software maintenance [25], or NIDS [10], [24], [26], to other domains that are not inherently computer-network-centric, e. g., meteorological [27], [28] or medical data [29]. One reason contributing to this disagreement are (cor)related features [10], [27]–[30], since such relationships generally complicate attribution, possibly enabling multiple explanations.

e) Feature Decomposition: To address correlated features, one common approach is feature decomposition, with PCA being the most well-known representative. PCA works by computing eigenvectors of the covariance matrix to define a new orthogonal basis. Preferably, features are standardized to avoid dominance of high-variance features. If all components are retained, this basically rotates the original features, where each component is a linear combination of input features. PCA preserves Euclidean distances, best suited for continuous data. Sparse PCA (SPCA) enforces sparsity of components to improve interpretability, though losing perfect decorrelation. MCA can be seen as PCA for categorical data. It performs decomposition on the so-called complete indicator (one-hot) matrix, i. e., only binary variables, indicating if a category is present. MCA preserves chi-square distances, capturing relationships in *continuous* space. In other words, while the MCA input is binary, the output is not. Note that decomposition and decorrelation, while technically distinct, are used interchangeably here. Methods like PCA/MCA achieve both.

III. Related Work

A. XAI-driven NIDS

NIDS research has already utilized XAI approaches, e. g., to generate explanations on local and global scopes [31], [32] or for feature selection [17], [33], potentially even in an adversarial way [34], [35]. Rather than simply explaining decisions, many works have changed their view to comparing and testing different explainers quantitatively, e. g., w.r.t. the *completeness*, *conciseness*, *consistency*, *robustness*, and/or *faithfulness* to the

[1] https://github.com/lsinfo3/cnsm2025-xai-nids-decomposition

[2] https://eur-lex.europa.eu/eli/reg/2024/1689

ground truth of an explanation [22], [36]–[40], only to name a few objective criteria of those works. Some works even include qualitative user studies with security experts to gain more subjective, practical insights [24].

Although some of the above works touch on the disagreement problem, it typically plays a secondary role. In comparison, XAI consensus is the primary concern in our work. Instead of merely identifying and/or measuring the extent of XAI disagreement, as we have done in prior work [10], [14] and now further extend, we aim for a deeper understanding of its underlying causes and potential strategies to address it.

B. XAI beyond NIDS

In addition to IDS-centric works, other domains (e. g., images, languages, finances, healthcare, criminology) have acknowledged challenges in XAI, tackling inconsistencies and disagreements, or addressing other related aspects more directly. Some works suggest joint or grouped feature importance [27], [30], [41], i. e., assigning attributions to feature groups instead of single features. Others align explanations by proposing frameworks for aggregation [42], [43] or to guide practitioners and stakeholders [20], [44], [45], redefine consensus functions [46], or adjust training objectives [47], potentially compromising, to enforce consensus.

Alternatively, some approaches constrain explanations to *regions* [48] by partitioning reference data to limit feature interactions. Other approaches reduce the number of input features directly to examine the effect on generated explanations [28]. Note that some works indeed use decomposition techniques like PCA in their XAI workflow, but merely as a preprocessing step [49], to visualize explanations [50], or to plot the aforementioned grouped feature interaction effects [41], rather than examining the effect of decorrelation.

Finally, some studies systematically vary dataset characteristics, such as number of features, samples, labels, or feature dependencies (e. g., correlations or interactions) [29], [51]. Others examine preprocessing steps like feature scaling or encoding [52], assess model-related factors, such as training duration [53], or examine hyperparameters of the actual XAI methods [13]. These investigations often leverage synthetic datasets, which offer more control over data configurations.

Similarly to Laberge et al. [48], we find aggregations or redefinitions do not grasp the problem's "roots" and only address its symptoms. Limiting features also artificially forces consensus, so the effect of decorrelation is harder to study, raising the question, how many and which features to cut. Compared to others, we focus on simply transforming our input without dropping information w.r.t. features, i. e., "rotating" our data. To the best of our knowledge, this is one of the first explorations of the disagreement problem within NIDS in conjunction with our earlier analyses.

IV. Methodology

A. NIDS Datasets

First, CICIDS2017 [54] is currently one of the most commonly used datasets in NIDS literature [4]. This dataset is on a flow basis, reporting statistical moments about inter-arrival times (IATs), packet sizes etc., totaling to 77 features[3]. The dataset depicts a week of varying attack scenarios, of which we utilize the Wednesday subset, containing (Distributed) DoS (DoS/DDoS) and almost 700k samples. To avoid overfitting on artifacts that could be spoofed, we exclude IPs and ports, which we do for the other datasets, too.

Our second dataset is CIDDS-001 [55], which comprises NetFlow, one of the most popular network monitoring protocols in practice. The contained flow information is much sparser due to this format. The dataset consists of different weeks and vantage points, of which we utilize the first week, which contains over 8 million data points, including Pingscan, Portscan, Bruteforce, and DoS. Features include standard NetFlow information, e. g., # packets, # bytes, or TCP flags. We derive two more features to enrich it further: the flow IAT and number of parallel flows, totaling to 14 features[3].

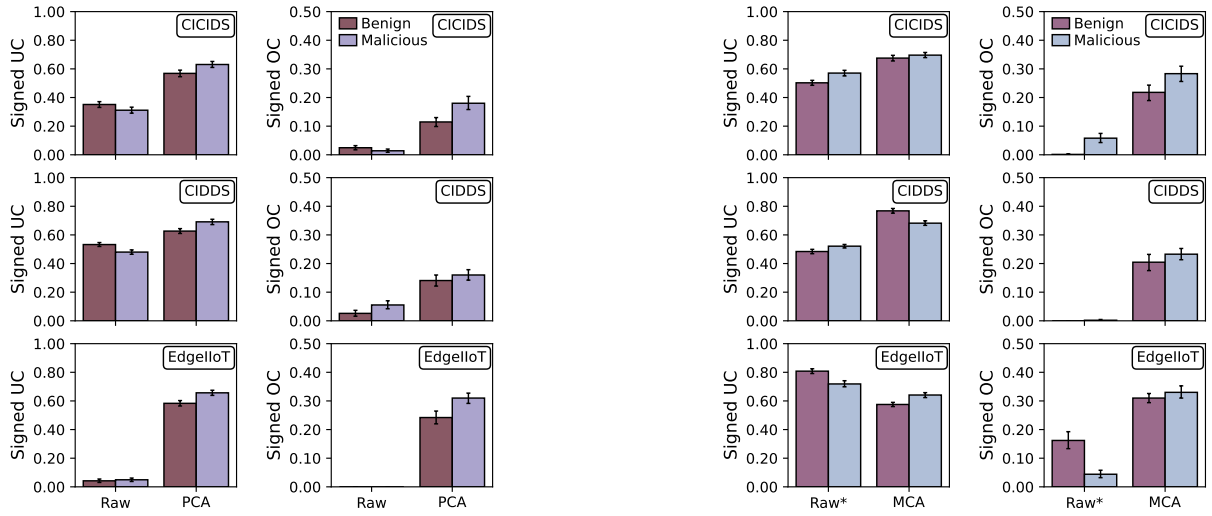
The third dataset is the Edge-IIoTset [56], which depicts the most recent data, including more fine-granular features from alerts and log data, totaling to 35 features[3] and over two million samples. Features are extracted from a variety of (Industrial) Internet of Things (IoT/IIoT) protocols (e. g., UDP, TCP, MODBUS, MQTT). It contains Portscan, DDoS, and more. Note that while we generally adhere to the authors' instructions of which features to drop, we decide to remove additional ones, which we deem of questionable generalizability, e. g., checksums, unit IDs, raw ACK numbers etc.

B. ML Workflow

We apply *StratifiedGroupKFold()* from sklearn [57], grouping samples in 30s bins based on each dataset's timestamp[4]. This avoids completely random shuffling, thus preserving some temporal structure. We utilize three folds (i. e., $K = 3$ train-test-splits) to ensure each sample is tested once. Categorical features are one-hot-encoded, and constant features are discarded. Note that we also encode low cardinality features (i. e., ≤ 5 unique values). After this initial preprocessing, we now want to compare the decorrelated data with the exact same workflow without decomposition. In our first approach (henceforth: *continuous* pipelines), we standardize the data, and compare the consensus pre- (*Raw*) and post-PCA, i. e., we treat *all* features in a more continuous/numerical way. Analogously, in our second approach (henceforth: *discrete* pipelines), we discretize all features to build the *full* indicator matrix, and then compare the consensus pre- (*Raw**) and post-MCA, i. e., we treat all features in a more discrete/categorical way. In other words, we take features like IATs and divide number ranges into categories, e. g., "short" and "long". For CICIDS and CIDDS we perform discretization based on quantiles into three bins (i. e., categories), whereas for Edge-IIoT we utilize k-means ($k = 3$), since the quantile-based approach resulted in too many empty bins. Besides sklearn for the above, we use Prince [58] for MCA and PyTorch [59]

[3]This is **before** filtering low variance/handling non-numerical features.

[4]We drop over 100k samples for the Edge-IIoTset with invalid timestamps.



(a) Continuous pipeline. (b) Discrete pipeline.

Fig. 2: Per-class-consensus pre- (raw) and post-PCA/MCA for DeepSHAP and LIME; **MLP** as underlying model.

TABLE I: F1-scores (micro and macro-avg.) for all datasets and preprocessing methods.

		CICIDS				CIDDS				Edge-IIoT			
Model	F1	Raw	PCA	Raw*	MCA	Raw	PCA	Raw*	MCA	Raw	PCA	Raw*	MCA
RF	micro	0.997	0.996	0.983	0.983	0.997	0.996	0.988	0.988	1.000	1.000	1.000	1.000
	macro	0.997	0.996	0.982	0.982	0.995	0.993	0.979	0.979	1.000	1.000	1.000	1.000
MLP	micro	0.983	0.985	0.983	0.984	0.996	0.996	0.987	0.987	> 0.999	> 0.999	1.000	> 0.999
	macro	0.981	0.984	0.982	0.982	0.993	0.993	0.978	0.978	> 0.999	> 0.999	1.000	> 0.999

for our DNN. After preprocessing, we balance training data by sampling 100k samples per class.

Our goal is not to build the perfect model, rather a lightweight but adequate one that enables parameter studies and practical applicability. To isolate the effect of decorrelation, we fix model parameters. We use a Random Forest (RF; 50 trees, max. depth 20) and a Multi-Layer-Perceptron (MLP; two fully connected layers with 64 neurons, ReLU activations, and dropout ratio of 0.5). Similar models are widely adopted in research on NIDS [4], as well as XAI [11], [47], allowing us to examine shallow ML and Deep Learning (DL) models.

C. Metrics for XAI Consensus

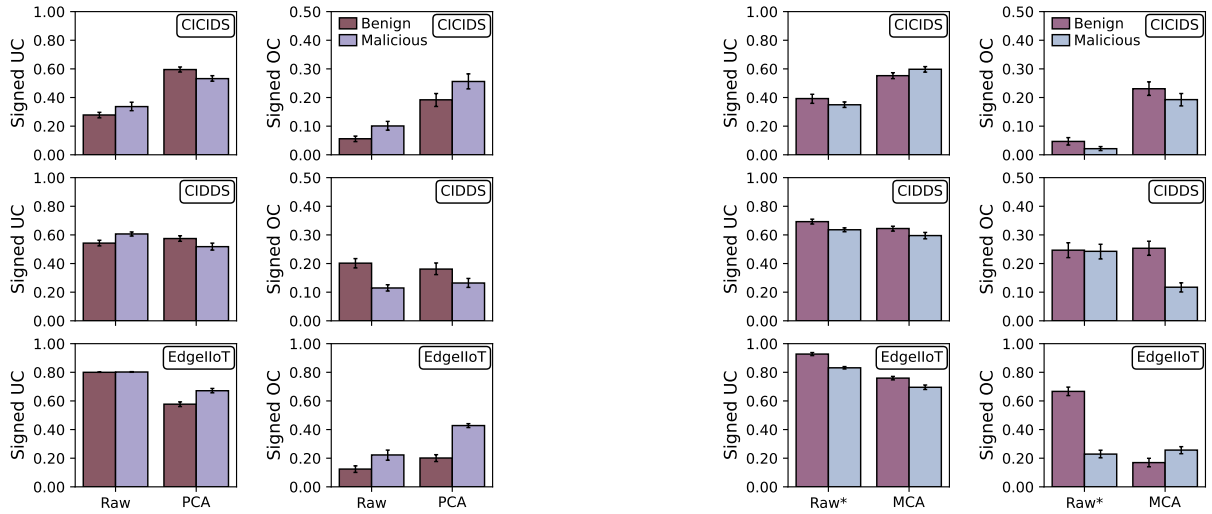
For the post-hoc explainers, we use SHAP and LIME, as previously discussed. For SHAP, we opt for the more efficient, model-specific implementations TreeSHAP and DeepSHAP. To compare explanations generated by SHAP and LIME, we adopt consensus metrics similar to those proposed by Krishna et al. [11]. We use two forms of agreement: unordered (UC) and ordered consensus (OC), based on the top-5 features. For the UC, we simply compute the feature overlap, ignoring their order of importance. For the OC, however, we care about how many of these top features match in sequence until the first mismatch. That is, we do not care if, e. g., only the third feature matches if the first and/or second feature mismatch. We argue that this "early agreement" is more intuitive than simply comparing the ranks, since the topmost feature is not only the most important for the model's decision but also for the human interpreting the explanations. In other words, if there are disagreements on the first feature, the explanation may already be deemed diverging, and agreements on later features are less important. Last, for both UC and OC we consider features as matching only if their sign also matches, because a negative sign means the feature argues against the predicted class, while a positive one supports it, raising confusion and mistrust. As a concrete example, take explanations $\{(f_1,+),(f_2,-),(f_3,+),(f_4,+),(f_5,+),...\}$ and $\{(f_1,+),(f_3,-),(f_2,-),(f_4,+),(f_6,+),...\}$, with signs indicating whether features $(f_1, f_2, ...)$ contribute positively or negatively. The UC is then $3/5$, since three features overlap with matching sign. For the OC, both start with $(f_1,+)$, but then $(f_2,-) \neq (f_3,-)$, so the match stops, giving $1/5$.

V. Evaluation

A. Impact of Decomposition on NIDS Data

To ensure that potential disagreements are not caused by underperforming models, Table I shows F1-scores (micro and macro) for all datasets, pipelines, and models. The RF tends to perform slightly better for CICIDS in the continuous pipelines. The discretized pipelines slightly decrease the F1-scores in a few cases, as it might remove some details. Since Edge-IIoT already consists of many categorical and binary features, it remains unaffected. Nonetheless, the models perform adequately across the board and ensure meaningful explanations.

Figure 2 illustrates the UC and OC between SHAP- and LIME-based explanations as a barplot for the MLP for all pipelines. The x-axis shows the preprocessing pipeline, while the y-axis holds the normalized consensus (i. e., from 0 to 1), averaged over 100 random samples from the test set for



(a) Continuous pipeline. (b) Discrete pipeline.

Fig. 3: Per-class-consensus pre- (raw) and post-PCA/MCA for TreeSHAP and LIME; **RF** as underlying model.

all three splits, i. e., each bar is made up by $3 \cdot 100$ values (for both classes to account for imbalances). The errorbars depict the 95% confidence intervals. We start with objective descriptions, before diving into a more subjective discussion.

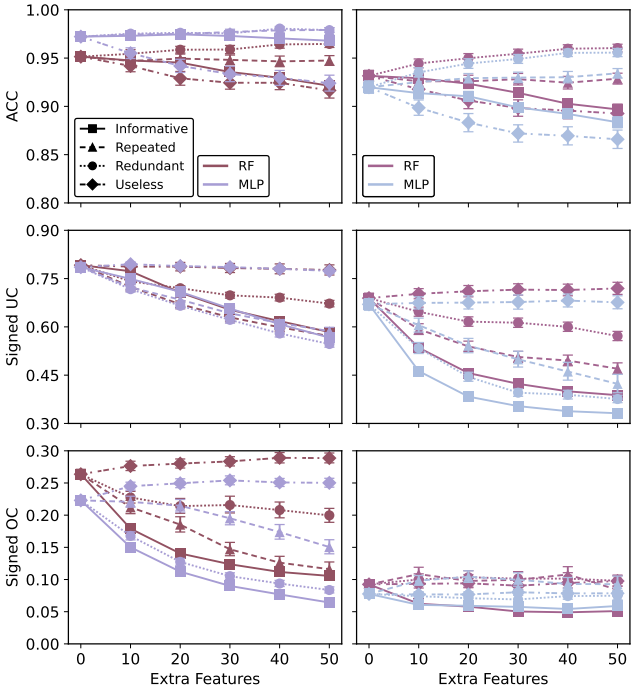
For the continuous pipeline in Figure 2a, we see that decorrelation improves the UC significantly for all datasets to roughly 60%, whereas the consensus on the raw features is heavily dependent on the dataset. For the OC, PCA is also able to establish some consensus, compared to almost zero consensus for the raw approach. Naturally, since the OC is much stricter, the consensus is generally lower. Interestingly, for the discrete pipeline in Figure 2b, the UC for Edge-IIoT jumps from almost zero consensus seen in the continuous pipeline to around 80% for the raw features and applying MCA actually decreases the consensus. For CICIDS and CIDDS, however, the UC is improved by applying MCA. For the OC, we see a trend similar to that for the PCA-based pipeline, where MCA creates at least some agreement.

Figures 3a and 3b depict the UC and OC of SHAP and LIME for the RF as underlying model analogously to the MLP. Here, the results are mixed. For both PCA and MCA, we only observe a meaningful impact for the CICIDS dataset. This holds true for UC and OC, as well as PCA and MCA. For CIDDS, the pre- and post-decorrelation consensus is similar but slightly decreased, whereas for Edge-IIoT the UC on raw features is near perfection and both consensus types do not show major benefits for applying decorrelation.

Discussion: The only dataset that shows a consistent consensus improvement under decorrelation is CICIDS, as it generally contains more correlated features. In detail, some features represent different statistical moments, which tend to be naturally correlated (e. g., higher mean might come along with higher maximum), as well as features split up by traffic direction. Thus, it makes sense that decorrelation is the most fruitful here. In addition, especially visible for the MLP, the discrete pipelines generally are often preferable compared to their continuous counterparts for Edge-IIoT (partially for CIDDS, too). We hypothesize that this is due to the fact that particularly Edge-IIoT contains many categorical variables. Also, tree-based models generally do not care if features are scaled or normalized, since they simply generate rules. However, for MLP having features being strictly binary, thus eliminating any disparity in feature ranges which might occur with standardization and potential outliers, might make gradients more stable. Lastly, the RF responds mixed to PCA/MCA. Our hypothesis is that the RF is better at filtering useless features before any XAI is applied. In other words, the RF will simply not use noisy features and opt for more meaningful ones, while the MLP takes all features into consideration due to its architecture. Especially since we use the model-specific SHAP versions, this directly impacts the consensus. Summarizing, while decorrelation can have a positive impact, this is highly dependent on the dataset and underlying ML model. Aligning encoding with feature types may also benefit consensus. Overall, this highlights that consensus is highly sensitive to seemingly small details.

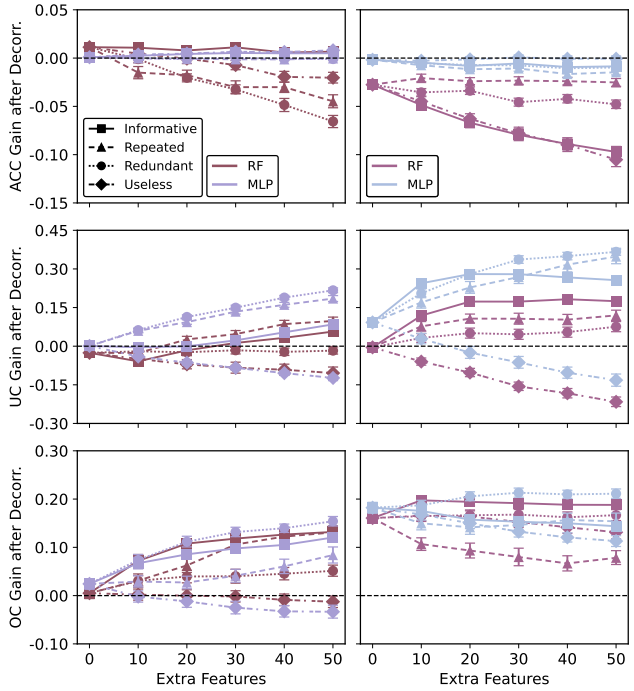
B. Impact of Decomposition on Synthetic Data

To get a clearer idea when decorrelation is beneficial, we shift our view to adjustable synthetic data. We utilize sklearn's *make_classification()*, which has four configurable feature types (initially, synthesized features are continuous): *informative*, *repeated*, *redundant*, and *useless*. Informative features are directly relevant to the target variable (i. e., class label). Repeated features are exact duplications, redundant features are linear combinations of others, and useless features consist of pure noise. We additionally implement three custom features: *squared*, *cubed*, and *exponential*, to examine non-linear redundancies. Each run begins with ten informative features. We incrementally add up to 50 features (in steps of 10) for each type and measure consensus before and after decorrelation. For every combination of feature type and quantity, we synthesize 100 balanced datasets consisting of 1k samples to ensure robust results, with train-test-splits of 80:20.



(a) pre-PCA. (b) pre-MCA.

Fig. 4: Performance and consensus pre decorrelation.



(a) post-PCA. (b) post-MCA.

Fig. 5: Performance and consensus **gain** post decorrelation.

For this, Figure 4a depicts the results for the continuous workflow for both RF and MLP for the pre-PCA features. The x-axis represents the number of injected features (i. e., excluding the ten initial ones), while the y-axis represents the accuracy and our two consensus types. Note that accuracy is appropriate here, since the synthesized datasets are balanced. The purple lines depict the MLP, while the red lines show the RF. The different linestyles depict the four default features of *make_classification()*. Again, we start by providing neutral figure descriptions before discussing them subjectively. Note that we discuss the three custom feature types in the end, too.

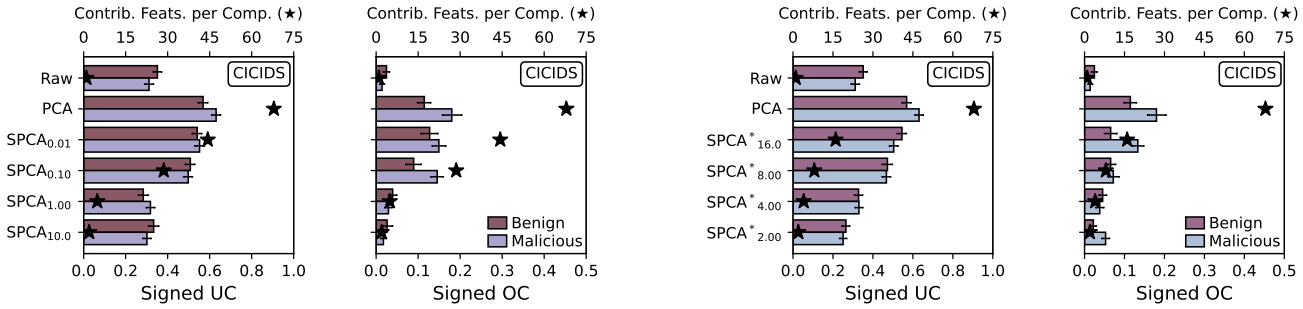
For the accuracy, we see a slight drop in performance for both models when extra informative or useless features are added, since they either make the task more complex by distributing information over more features or mask important features via noise. The impact of extra features on the UC is almost identical for both models with exception of redundant features, where the RF appears to be more robust than the MLP. Generally, both models are only robust for noisy features, however. For the other three features, the UC declines gradually. For the OC, we see similar trends, though the RF generally tends to have a slight increased consensus.

Figure 4b shows the same analyses for the discrete workflow for the pre-MCA features analogously. For the accuracy, we see a similar trend like for the pre-PCA features, where extra informative and useless features worsen both models' performance. Though, for both models the overall accuracy decreases, similarly to the NIDS data, due to potential information loss. The UC also depicts similar findings to before, with useless features not impacting the consensus, and the RF appearing significantly more robust to redundant features. The UC is generally lower, and extra features have a greater impact compared to the continuous pipeline. Similarly, the OC is also much lower, but trends stay mostly comparable.

To actually analyze the impact of decorrelation, Figure 5a shows the gains post-PCA. The general visualization remains the same as in the previous figure, but the y-axis now illustrates the *gain* after applying PCA, i. e., the differences in accuracy and consensus. Additionally, the black dashed line at $y = 0$ visualizes the cutoff where decorrelation has a positive effect. For the accuracy, we see that the MLP performs just like on the raw features, even slightly better in some cases. The RF, however, only keeps (relative) performance up on the informative features, whereas for the rest performance is gradually decreasing. For the UC, we see mostly negative effects for the useless features, whereas the other feature types are positively impacted by this. Note that the informative features do indeed contain correlations as well. The MLP also appears to benefit more from PCA than the RF, with redundant features having a neutral impact for the RF and most benefits for the MLP. The OC trend is similar but clearer, where decorrelation impacts all feature types except useless ones positively.

Figure 5b shows the gains post-MCA in the same fashion. Here, too, MLP mostly maintains its performance compared to the raw features, and the RF generally drops in accuracy, though more pronounced compared to the continuous pipeline. The UC is also impacted similarly, but also more pronounced, with MLP benefiting even more and both models being more prone to noise. For the OC, this impact is even greater, where consensus is already significantly improved even before adding any features, and even noisy features are positively impacted.

Lastly, for our three custom feature types, their impact



(a) XAI consensus after applying sklearn's SPCA.

(b) XAI consensus after applying customized SPCA*.

Fig. 6: Per-class-consensus pre- (raw) and post-(S)PCA for DeepSHAP and LIME; **MLP** as underlying model.

can be summarized quite easily. In general, the cubed and exponential features have a similar impact as the repeated features, while the squared features behave noisier. This is attributable to the fact that by squaring we lose the feature's sign, thus any relation gets blurrier, while cubing and exponentiation are both still monotonic transformations. Though, in the continuous pipeline decorrelation has less of an impact on the OC, potentially since it does not capture the non-linear relations, which is negated in the discrete pipeline by binning.

Discussion: The analyses shed some light on why the decomposition has a positive or negative effect on the consensus for actual NIDS data. Noisy features obfuscate decomposition, as they may randomly correlate and get intermixed into components with actually relevant features. Decomposition is less beneficial for the RF, which handles noise, non-linearity, and multicollinearity well. Thus, the feature rotation or transformation may hinder the RF to form effective splits. This is supported by the improved consensus for MLP and the slight drop in classification performance for the RF when decorrelation is applied. We hypothesize that we may not observe this effect as clearly in the NIDS data, since for the synthetic data noise and correlations are artificially "perfect" and thus potentially induce stronger effects. The MCA-based pipeline benefits more, especially for the OC, as the consensus on raw discretized features is lower due to inflated feature space. Since we discretize each feature into three bins, the number of features is roughly multiplied by three, thus choosing the "correct" features might become even harder, since these binned features may also exhibit high correlations (i. e., if a feature falls into one category, it cannot fall into the other categories). MCA counteracts this by decorrelation. Again, we hypothesize that this effect is less visible in NIDS data, because it naturally contains some categorical features and has different feature ranges or distributions, making it more suitable for this pipeline. Overall, this confirms that decorrelation has a positive impact when applied properly, but also reinforces dependence on both model and preprocessing.

C. Interpretability of Components

After quantifying when decomposition is useful, what is worth discussing next is the interpretability of the transformed features, i. e., the components. After transformation, every input feature is now a linear combination of the original features. Thus, if a human expert is tasked to interpret the resulting explanations, this might be more complex than in the original feature space. There is, however, the option to algorithmically limit the number of contributing features per component with SPCA. SPCA modifies the traditional PCA by enforcing sparsity constraints onto the component's loadings, i. e., each component depends only on a subset of features instead of potentially all features, by controlling the regularization parameter α. Higher values of α enforce higher sparsity. So, normal PCA is essentially SPCA with $\alpha = 0$. Unlike PCA, SPCA does not guarantee uncorrelated components.

While SPCA is able to drastically increase sparsity and thus interpretability, the number of non-zero loadings per component is only indirectly controllable. To address this, we also consider a custom SPCA approach (SPCA*), where we manually sparsify the PCA components by retaining only the top n loadings and setting the rest to zero. This allows precise control over the number of contributing features per component, but also induces feature correlations again.

We substitute PCA with SPCA with $\alpha \in \{0.01, 0.1, 1, 10\}$ and SPCA* with $n \in \{2, 4, 8, 16\}$ and execute our XAI pipeline again. Note that in terms of classification performance, both approaches perform almost exactly like PCA. For the consensus, Figure 6 shows our two consensus metrics for both SPCA-based pipelines. We focus on the MLP and CICIDS as a use case, since here PCA is the most fruitful. Figures for the other datasets and the RF can be found in the accompanying repository for the sake of transparency. Generally, the trends in these cases are somewhat similar, but less distinct (e. g., when PCA has no impact in the first place). The bottom x-axis shows the consensus, while the y-axis shows the applied transformation. Errorbars and colors are identical to the previous analyses. The top x-axis shows the average number of features contributing to a component, depicted with a star. For the raw, untransformed features this is one, since they only consist of themselves. For the PCA, this is equal to the number of features. Again, we start with a neutral description before discussing the results.

For both sparse transformations and both UC and OC, we see a tradeoff between sparsity/regularization and consensus. For the regular SPCA, we see that applying only a small regularization still improves the consensus significantly. While even this small regularization sparsifies the components drasti-

cally compared to PCA, this still results in almost 30 features per component for an alpha of 0.1, and around 45 for an alpha of 0.01. We argue that explanations based on components that are consisting of 30 features are still hard to interpret for a human user. When increasing the alpha, the benefits disappear and are only slightly visible. The custom SPCA* shows similar trends, where the more loadings we set to zero, the more the consensus decreases. Though, the consensus when setting the number of non-zero loadings to 16 (and partially 8), shows a similar consensus to regular SPCA with low regularization.

Discussion: The results are generally in accordance to our previous analyses, since the sparser components are, the more similar they become to the original features again, depicting a tradeoff between interpretability and consensus. However, even though we are able to sparsify the components drastically with SPCA, this might still not be enough to make it truly interpretable. The presented values are only average values, meaning that specific components can be even denser, making matters worse. As an alternative, the custom SPCA* is able to achieve similar consensus increases, it is also able to limit the maximum number of features, too. So, in general, sparsifying the components can be a remedy to making PCA more interpretable by design. However, this is only applicable if PCA has a benefit in the first place. That is, as mentioned previously, the effectiveness may vary with dataset and model.

To provide an overall practical guideline of all of our previous analyses, decorrelation can increase agreement and thus trust in downstream decisions, but is not one-size-fits-all. Feature engineering remains necessary to remove useless features that diffuse components. If few correlations are present, decompositions may even hinder interpretation, while sparsity-inducing variants can further aid decision-making by limiting the number of contributing features.

D. Further Considerations and Limitations

Besides the in-depth analyses presented, we also briefly explored additional aspects of the disagreement problem on NIDS data that we find noteworthy. All figures corresponding to these additional analyses can be found in the accompanying repository as well, and are discussed briefly in the following.

So far, we have focused on intra-model consensus, i. e., comparing explainers for the same model. However, especially when ensembles are used, it is valuable to examine agreement between different models, i. e., inter-model consensus [44]. Our analyses on the NIDS data suggest that inter-model consensus exhibits similar trends w.r.t. consensus improvement after applying decomposition techniques. That is, decorrelation proves to be most fruitful for CICIDS, while response to the other two datasets is more mixed and inconsistent.

Additionally, while we have focused on binary classification, we can naturally extend this to a multiclass problem, especially relevant for downstream tasks like intrusion prevention and containment. Again, results in multiclass settings on the NIDS data reveal somewhat comparable overall trends w.r.t. consensus, but with nuanced, class-specific variations. This is potentially attributable to mixed detection accuracies and complexity levels of the various attacks.

Besides PCA and MCA, we also experimented with Independent Component Analysis (ICA), which enforces statistical independence rather than mere decorrelation. ICA sometimes reduced classification accuracy due to its stricter assumptions on data, while mostly having less impact than PCA on consensus. Additionally, Kernel PCA, capable of capturing non-linear correlations via custom kernels, proved computationally infeasible for the NIDS data, requiring terabytes of RAM.

Limitations: Besides the aspects above, there are additional factors, which we did not explore further. For example, we used fixed model configurations, but consensus may vary with different hyperparameters (e. g., tree-depth of RF, training duration and depth of MLP). For example, disagreements could be greater for unpruned trees, or maybe "settle" if we increase training durations. Moreover, we limited our analysis to SHAP and LIME, which are both perturbation-based explainers. Including a broader range of models and explanation methods, such as gradient-based approaches, could paint a more comprehensive picture. Likewise, we focused on tabular data. Exploring more complex models and input domains would further strengthen the analysis.

VI. Conclusion

This work investigated the impact of feature decorrelation on the consensus of XAI methods in ML-based network intrusion detection. We showed that decomposition techniques such as PCA and MCA can significantly improve consensus between explainers like SHAP and LIME, especially for models and datasets prone to feature correlations. However, the benefits vary depending on model type, preprocessing, and dataset characteristics. Notably, decorrelation has a more stable positive effect on MLPs, while RFs are more robust to noise and correlation by design. To address the interpretability loss, we explored SPCA and a custom sparsification approach. While these methods improved component sparsity, the results were mixed in terms of maintaining consensus, highlighting a tradeoff between interpretability and agreement.

Our results underline that improving XAI consensus is not one-size-fits-all. Decomposition can be a useful tool, but must be aligned with model architecture and data characteristics. A natural extension of this work is using more ML models and XAI methods. Future work could explore whether inherently interpretable models offer more stable and trustworthy explanations, potentially avoiding the sensitivity issues observed with post-hoc methods. In other words, if we already have to be very careful when designing our ML pipeline anyway in order to make post-hoc XAI robust, we may prefer to put this energy more towards designing white-boxes.

Acknowledgment

This work has been funded by the Bavarian Ministry of Economics, Regional Development and Energy (StMWI) as part of the project VIPNANO (DIK-2307-0006) and by Deutsche Forschungsgemeinschaft (DFG) under grant SE 3163/3-1, project nr.: 500105691 (UserNet). The authors alone are responsible for the content.

REFERENCES

[1] D. Ucci, L. Aniello, and R. Baldoni, "Survey of machine learning techniques for malware analysis," *Comput. Secur.*, 2019.

[2] G. Apruzzese, P. Laskov, and J. Schneider, "SoK: Pragmatic assessment of machine learning for network intrusion detection," in *IEEE EuroS&P*, 2023.

[3] M. Seufert *et al.*, "Marina: Realizing ML-driven real-time network traffic monitoring at terabit scale," *IEEE Trans. Netw. Serv. Manag.*, 2024.

[4] K. Dietz *et al.*, "The missing link in network intrusion detection: Taking AI/ML research efforts to users," *IEEE Access*, 2024.

[5] A. Adadi and M. Berrada, "Peeking inside the black-box: a survey on explainable artificial intelligence (XAI)," *IEEE Access*, 2018.

[6] D. Minh, H. X. Wang, Y. F. Li, and T. N. Nguyen, "Explainable artificial intelligence: a comprehensive review," *Artif. Intell. Rev.*, 2022.

[7] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?": Explaining the predictions of any classifier," in *ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2016.

[8] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *NeurIPS*, 2017.

[9] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *ICML*, 2017.

[10] K. Dietz *et al.*, "Agree to disagree: Exploring consensus of XAI methods for ML-based NIDS," in *NeSecOr*, 2024.

[11] S. Krishna *et al.*, "The disagreement problem in explainable machine learning: A practitioner's perspective," *Trans. Mach. Learn. Res.*, 2024.

[12] D. Brughmans, L. Melis, and D. Martens, "Disagreement amongst counterfactual explanations: How transparency can be deceptive," *Trans. Oper. Res. (TOP)*, 2024.

[13] K. Wickstrøm, M. Höhne, and A. Hedström, "From flexibility to manipulation: The slippery slope of XAI evaluation," in *Eur. Conf. Comput. Vis. (ECCV) Workshops*, 2024.

[14] K. Dietz *et al.*, "I choose you: Evaluating the impact of feature selection on XAI consensus for ML-NIDS," in *MaLeNe*, 2025.

[15] T. Grance *et al.*, "Guide to information technology security services," *NIST Special Publication 800-35*, 2003.

[16] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI Mag.*, 2017.

[17] P. Khani, E. Moeinaddini, N. D. Abnavi, and A. Shahraki, "Explainable artificial intelligence for feature selection in network traffic classification: A comparative study," *Trans. Emerg. Telecommun. Technol.*, 2024.

[18] J. Tritscher, A. Krause, and A. Hotho, "Feature relevance XAI in anomaly detection: Reviewing approaches and challenges," *Front. Artif. Intell.*, 2023.

[19] L. Shapley, "A value for n-person games," *Contrib. Theory Games*, 1953.

[20] S. Li, Q. Deng, and A. S. Barnard, "EXAGREE: Towards explanation agreement in explainable machine learning," *arXiv:2411.01956*, 2024.

[21] M. Watson, B. A. S. Hasan, and N. Al Moubayed, "Agree to disagree: When deep learning models with identical architectures produce distinct explanations," in *IEEE/CVF WACV*, 2022.

[22] A. Warnecke, D. Arp, C. Wressnegger, and K. Rieck, "Evaluating explanation methods for deep learning in security," in *IEEE EuroS&P*, 2020.

[23] L. Rui and O. Gadyatskaya, "Position: The explainability paradox-challenges for XAI in malware detection and analysis," in *IEEE EuroS&PW*, 2024.

[24] D. Bhusal *et al.*, "SoK: Modeling explainability in security analytics for interpretability, trustworthiness, and usability," in *ACM ARES*, 2023.

[25] S. Roy *et al.*, "Why don't XAI techniques agree? characterizing the disagreements between post-hoc explanations of defect predictions," in *IEEE ICSME*, 2022.

[26] A. Nadeem *et al.*, "SoK: Explainable machine learning for computer security applications," in *IEEE EuroS&P*, 2023.

[27] M. Flora, C. Potvin, A. McGovern, and S. Handler, "Comparing explanation methods for traditional machine learning models part 1: an overview of current methods and quantifying their disagreement," *arXiv:2211.08943*, 2022.

[28] ——, "Comparing explanation methods for traditional machine learning models part 2: Quantifying model explainability faithfulness and improvements with dimensionality reduction," *arXiv:2211.10378*, 2022.

[29] A. F. Markus *et al.*, "Understanding the size of the feature importance disagreement problem in real-world data," in *ICML Workshop IMLH*, 2023.

[30] F. Fumagalli *et al.*, "Unifying feature-based explanations with functional ANOVA and cooperative game theory," in *AISTATS*, 2025.

[31] S. Hariharan *et al.*, "XAI for intrusion detection system: comparing explanations based on global and local scope," *J. Comput. Virol. Hacking Tech.*, 2023.

[32] M. Wang, K. Zheng, Y. Yang, and X. Wang, "An explainable machine learning framework for intrusion detection systems," *IEEE Access*, 2020.

[33] O. Arreche, T. Guntur, and M. Abdallah, "XAI-based feature selection for improved network intrusion detection systems," *arXiv:2410.10050*, 2024.

[34] H. Qiu *et al.*, "Adversarial attacks against network intrusion detection in IoT systems," *IEEE Internet Things J.*, 2021.

[35] S. Okada *et al.*, "XAI-driven adversarial attacks on network intrusion detectors," in *EICC*, 2024.

[36] O. Arreche, T. R. Guntur, J. W. Roberts, and M. Abdallah, "E-XAI: Evaluating black-box explainable AI frameworks for network intrusion detection," *IEEE Access*, 2024.

[37] A. N. Gummadi, O. Arreche, and M. Abdallah, "A systematic evaluation of white-box explainable AI methods for anomaly detection in IoT systems," *Internet of Things*, 2025.

[38] J. Tritscher *et al.*, "Evaluation of post-hoc XAI approaches through synthetic tabular data," in *ISMIS*, 2020.

[39] J. Tritscher, M. Wolf, A. Hotho, and D. Schlör, "Evaluating feature relevance XAI in network intrusion detection," in *World Conf. Explain. Artif. Intell.*, 2023.

[40] O. Lukás and S. García, "Bridging the explanation gap in AI security: A task-driven approach to XAI methods evaluation." in *ICAART*, 2024.

[41] Q. Au *et al.*, "Grouped feature importance and combined features effect plot," *Data Min. Knowl. Discov.*, 2022.

[42] O. Mitruț *et al.*, "Clarity in complexity: how aggregating explanations resolves the disagreement problem," *Artif. Intell. Rev.*, 2024.

[43] C. Pirie *et al.*, "AGREE: A feature attribution aggregation framework to address explainer disagreements with alignment metrics." in *CEUR Workshop Proc.*, 2023.

[44] C. Poiret, A. Grigis, J. Thomas, and M. Noulhiane, "Can we agree? on the Rashômon effect and the reliability of post-hoc explainable AI," *arXiv:2308.07247*, 2023.

[45] T. Han, S. Srinivas, and H. Lakkaraju, "Which explanation should I choose? A function approximation perspective to characterizing post hoc explanations," *NeurIPS*, 2022.

[46] A. J. Banegas-Luna, C. Martınez-Cortes, and H. Perez-Sanchez, "Fighting the disagreement in explainable machine learning with consensus," *arXiv:2307.01288*, 2023.

[47] A. Schwarzschild *et al.*, "Reckoning with the disagreement problem: Explanation consensus as a training objective," in *AAAI/ACM Conf. AI Ethics Soc. (AIES)*, 2023.

[48] G. Laberge, Y. B. Pequignot, M. Marchand, and F. Khomh, "Tackling the XAI disagreement problem with regional explanations," in *AISTATS*, 2024.

[49] P. Alves *et al.*, "Comparing LIME and SHAP global explanations for human activity recognition," in *Braz. Conf. Intell. Syst. (BRACIS)*, 2024.

[50] V. Swamy *et al.*, "Evaluating the explainers: black-box explainable machine learning for student success prediction in MOOCs," in *Int. Conf. Educ. Data Min. (EDM)*, 2022.

[51] Z. Carmichael and W. Scheirer, "How well do feature-additive explainers explain feature-additive predictors?" in *NeurIPS Workshop XAIA*, 2023.

[52] N. Koenen and M. N. Wright, "Toward understanding the disagreement problem in neural network feature attribution," in *World Conf. Explain. Artif. Intell.*, 2024.

[53] P. Silva, C. T. Silva, and L. G. Nonato, "Exploring the relationship between feature attribution methods and model performance," in *AAAI Workshop AI4ED*, 2024.

[54] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization." *ICISSP*, 2018.

[55] M. Ring *et al.*, "Flow-based benchmark data sets for intrusion detection," in *Eur. Conf. Cyber Warf. Secur. (ECCWS)*, 2017.

[56] M. A. Ferrag *et al.*, "Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning," *IEEE Access*, 2022.

[57] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res. (JMLR)*, 2011.

[58] M. Halford, "Prince: Python factor analysis library," https://github.com/MaxHalford/prince, MIT License.

[59] A. Paszke, *et al.*, "PyTorch: An imperative style, high-performance deep learning library," in *NeurIPS*, 2019.