Hermes: Boosting the Performance of Machine-Learning-Based Intrusion Detection System through Geometric Feature Learning

Chaoyu Zhang
Virginia Tech
Arlington, USA

Shanghao Shi
Virginia Tech
Arlington, USA

Ning Wang
USF
Tampa, USA

Xiangxiang Xu
MIT
Cambridge, USA

Shaoyu Li
Virginia Tech
Arlington, USA

Lizhong Zheng
MIT
Cambridge, USA

Randy Marchany
Virginia Tech
Blacksburg, USA

Mark Gardner
Virginia Tech
Blacksburg, USA

Y. Thomas Hou
Virginia Tech
Blacksburg, USA

Wenjing Lou
Virginia Tech
Arlington, USA

ABSTRACT

Anomaly-Based Intrusion Detection Systems (IDSs) have been extensively researched for their ability to detect zero-day attacks. These systems establish a baseline of normal behavior using benign traffic data and flag deviations from this norm as potential threats. They generally experience higher false alarm rates than signature-based IDSs. Unlike image data, where the observed features provide immediate utility, raw network traffic necessitates additional processing for effective detection. It is challenging to learn useful patterns directly from raw traffic data or simple traffic statistics (e.g., connection duration, package inter-arrival time) as the complex relationships are difficult to distinguish. Therefore, some feature engineering becomes imperative to extract and transform raw data into new feature representations that can directly improve the detection capability and reduce the false positive rate. We propose a geometric feature learning method to optimize the feature extraction process. We employ contrastive feature learning to learn a feature space where normal traffic instances reside in a compact cluster. We further utilize H-Score feature learning to maximize the compactness of the cluster representing the normal behavior, enhancing the subsequent anomaly detection performance. Our evaluations using the NSL-KDD and N-BaIoT datasets demonstrate that the proposed IDS powered by feature learning can consistently outperform state-of-the-art anomaly-based IDS methods by significantly lowering the false positive rate. Furthermore, we deploy the proposed IDS on a Raspberry Pi 4 and demonstrate its applicability on resource-constrained Internet of Things (IoT) devices, highlighting its versatility for diverse application scenarios.

CCS CONCEPTS

• Security and privacy → Intrusion detection systems; **• Computing methodologies → Machine learning algorithms**.

KEYWORDS

Machine Learning-Based Intrusion Detection System, Geometric Feature Learning, Contrastive Learning.

ACM Reference Format:
Chaoyu Zhang, Shanghao Shi, Ning Wang, Xiangxiang Xu, Shaoyu Li, Lizhong Zheng, Randy Marchany, Mark Gardner, Y. Thomas Hou, and Wenjing Lou. 2024. Hermes: Boosting the Performance of Machine-Learning-Based Intrusion Detection System through Geometric Feature Learning. In *International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing (MobiHoc '24), October 14–17, 2024, Athens, Greece.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3641512.3686380

1 INTRODUCTION

The development of cloud computing, the expansion of 5G networks, and the exponential growth of IoT devices have led to an unprecedented increase in network traffic and complexity. This expansion not only facilitates innovative applications but also causes significant vulnerabilities [18]. Cyber threats have evolved to be more sophisticated, and attackers are exploiting these vulnerabilities to compromise the confidentiality, integrity, and availability of critical information on the network. The 2016 Mirai botnet [2] is an example of powerful distributed denial-of-service (DDoS) attacks. In these attacks, numerous Internet-connected devices, infected with Mirai malware, flooded targeted servers with excessive Internet traffic. This attack resulted in major websites, including Twitter, Netflix, Reddit, and CNN, temporarily inaccessible to millions of users. Given these circumstances, the implementation of an intrusion detection system (IDS) has become imperative. IDS plays a crucial role in the continuous monitoring and analysis of network traffic, capable of identifying potential attacks.

In general, there are two types of IDSs to safeguard network infrastructures: signature-based and anomaly-based IDS. Signature-based IDS functions by comparing observed traffic against a database of known attack patterns or signatures [11, 13, 16, 17]. This method is highly effective in recognizing and thwarting established threats, making it an essential tool for defending against known attacks. However, its efficacy is limited to known threats, leaving a gap that anomaly-based IDS aims to fill. Anomaly-based IDS works on the principle of behavioral analysis [19, 23, 29, 32, 33, 40]. It establishes a baseline of normal network behavior and continuously monitors for deviations, flagging any unusual activity that could signify a potential intrusion. This approach enables it to detect

MobiHoc '24, October 14–17, 2024, Athens, Greece
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-0521-2/24/10
https://doi.org/10.1145/3641512.3686380

novel, a.k.a. zero-day attacks, which do not match any known signatures. While more adaptable to emerging threats, anomaly-based IDS can be prone to higher false-positive rates [34, 44].

Research Motivation: Constructing an anomaly-based IDS that not only detects intrusions but also identifies the type of intrusion (including known attack types and one superclass of zero-day attacks) is valuable. Identifying intrusion types is crucial for the intrusion response system (IRS) [3, 4]. Each type of intrusion may require a specific response. For example, a DDoS attack in [8] might necessitate rate limiting or blocking traffic from certain IPs, while a detected malware infiltration [5, 42] could require isolation of the infected system. Knowing the type of intrusion allows the IRS to activate the most effective countermeasures. Exploring zero-day attacks is also a crucial task [14, 27], since we can extract the new signatures of these emerging threats, thereby improving the robustness of existing signature-based IDS. Concurrently, recognizing zero-day attacks can provide significant and novel samples for retraining the model of the existing anomaly-based IDS, enhancing their detection capabilities. Therefore, an IDS that detects intrusion and identifies the intrusion type holds practical value and is necessary for the existing IRS and IDS. However, the all-in-one solution is particularly challenging for resource-constrained devices. These devices often have limited computational power, memory, and energy resources. Signature-based IDS typically require processing power and storage to maintain and update a database of known attack signatures, which is difficult for resource-constrained IoT devices. Keeping the signature database updated in a constantly evolving threat landscape is critical for the effectiveness of signature-based IDS. In an IoT environment, this poses a challenge due to the sheer number of devices and potentially limited network bandwidth.

Research Objective: Therefore, the goal of our research is to *1) improve the overall performance of the anomaly-based IDS by minimizing the false positive rate and reducing the incidence of detection omissions,* and *2) extend the capabilities of anomaly-based IDS by enabling them not only to detect an intrusion but also to precisely identify the specific type of the intrusion for resource-limited devices.*

In the conventional approach to machine learning-based IDS, feature engineering holds substantial importance within the pipeline as it directly impacts the ultimate detection performance. Unlike image data, where the observed features provide immediate utility, raw network traffic necessitates additional processing for effective detection [10, 15]. Learning useful patterns directly from raw data traffic or simple traffic statistics (e.g., connection duration, package inter-arrival time) is challenging as the complex relationships are difficult to distinguish. *Therefore, it becomes imperative to extract features that can directly enhance the performance of detection models rather than relying solely on raw data traffic or simple traffic statistics.*

We propose a novel IDS that optimizes the feature extraction process by learning a comprehensive and compact normal behavior. We utilize the geometric feature learning [38] as it enables an explainable distance-based feature extraction. For clarity purposes, we define the term **features** as the *feature representations extracted by the neural network.* The *space mapped by this neural network* is henceforth referred to as the **feature space**. Furthermore, we name the *space encompassing all benign feature representations* as the **baseline**.

The high-level design of Hermes includes a two-step training phase: the first step involves **contrastive feature learning**. Drawing inspiration from the application of contrastive learning in manipulating feature representations in latent space for computer vision tasks, as noted in [31, 35], our approach uses two normal traffic records to form a positive pair, and a normal and an abnormal traffic record to form a negative pair. This learning process attracts positives and increases the distance between negatives, generating a new baseline. In this space, benign instances are clustered closely while attack instances are repelled from the benign cluster, thereby enhancing the performance of subsequent anomaly detection.

The second step involves **H-Score feature learning** [36, 37] further optimizing the learned features by maximizing the compactness of the cluster representing the normal behavior. Furthermore, it can increase the distance between different types of traffic, making it easier for the IDS to distinguish between classes of attacks.

For IDS inference, Hermes employs a **dual inference** mechanism: it uses a **similarity-based rule** to measure deviations of traffic features from the baseline for anomaly-based detection, and a **Maximum A Posteriori (MAP)** rule for attack type identification. This inference mechanism also evaluates if an instance is a zero-day attack using **entropy-based uncertainty** from the posterior distribution and **inconsistency** between the two predictions.

Our contributions are summarized as follows:

(1) Hermes employs contrastive feature learning to establish a well-formed baseline to distinguish benign and intrusion features, thus enabling anomaly detection.
(2) Building on this, Hermes employs H-Score feature learning to capture the dependencies between traffic and their types, enabling the inference phase to identify the attack types. This method focuses on reducing geometrical distance in baseline, narrowing distances among the same type of features, and increasing distances between different classes of features, further enhancing anomaly detection ability and enabling attack identification.
(3) The dual inference of Hermes applies a similarity-based rule to the learned features to determine if the traffic is malicious and utilizes MAP estimation to assess the preliminary type of intrusion. Furthermore, it classifies the types of known attacks and flags zero-day attacks.
(4) Extensive evaluations of Hermes across two datasets and diverse hardware platforms have consistently demonstrated its effectiveness. Hermes outperforms various state-of-the-art IDS baselines.

2 SYSTEM MODEL AND THREAT MODEL

System Model: We consider a general IDS architecture following [1, 6], which is illustrated in Fig. 1. Our system operates on the premise that a network gateway $\mathcal{G}$ interconnects various devices. This gateway manages both incoming and outgoing network traffic, making it an optimal location for monitoring and analyzing network traffic. The IDS operates in two modes: training and inference. In training mode, it employs a network traffic flow generation protocol like NetFlow[1], which aggregates packets into sequences

[1]NetFlow: Cisco's widely-used flow-based network traffic collection and monitoring protocol [7].

Table 1: Definition and Notations

Symbol	Definition
X	Network flow records variable
Y	Network traffic labels variable
x	An instance of network flow record
y	A label for corresponding flow record
P_X, P_Y	Distributions of X and Y respectively
$P_{X,Y}$	Joint distribution of X and Y
$P_{Y\|X}$	Posterior distribution given X
$\mathfrak{i}_{X;Y}$	CDK function, the statistical dependence of X, Y
$\zeta_{\leq k}(\mathfrak{i}_{X;Y})$	Top k modes of $\mathfrak{i}_{X;Y}$
$\mathcal{F}^k_{\mathcal{X}}, \mathcal{F}^k_{\mathcal{Y}}$	k-dimensional feature spaces for X and Y
$f \in \mathcal{F}^k_{\mathcal{X}}$	k-dimensional features
$g \in \mathcal{F}^k_{\mathcal{Y}}$	k-dimensional features
$f \otimes g$	Joint function
$\mathcal{F}_{\mathcal{X}\times\mathcal{Y}}$	Functional space of all joint functions $f \otimes g$
l_{ij}	Pair-wise loss function
$\mathcal{L}$	Cumulative loss function
$\mathscr{H}(f, g)$	H-Score
$\mathrm{tr}(\cdot)$	Trace function, sum of elements on the diagonal
$\mathbb{1}(\cdot)$	Indicator function
$\|\cdot\|_{\mathcal{F}}$	Induced norm in feature geometry, defined as $\|\gamma\|^2_{\mathcal{F}} = \mathbb{E}_{P_X P_Y}\left[\gamma^2(X, Y)\right]$
$f(x; \theta_0)$	Model learned by contrastive feature learning
$f(x; \theta_1)$	Model refined by H-Score feature learning
$g(y; \theta_2)$	Identification model
$\mathcal{G}$, $\mathcal{S}$	Gateway $\mathcal{G}$, training server $\mathcal{S}$
$\mathscr{Z}(P_{Y\|X})$	Uncertainty from posteriori distribution

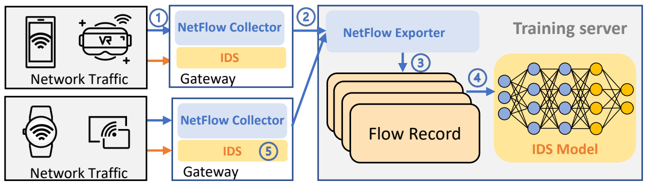

Figure 1: The system workflow of Hermes

and summarizes them for analysis. The NetFlow Exporter then compiles these sequences into flow records, encapsulating network traffic characteristics, and forwards them to a training server $\mathcal{S}$. These records are utilized for training the IDS model. In inference mode, the well-trained model is deployed at the network gateways $\mathcal{G}$ for intrusion detection.

Threat Model: In our threat model, we assume attackers will carry out various network-based attacks by manipulating network traffic. This includes maliciously generated probing traffic, data injection, denial of service attacks, and other network-based attacks that exploit system vulnerabilities, such as malware, infiltration and exfiltration, and man-in-the-middle attacks. We assume attackers may exploit zero-day attacks, implying that there are no known patterns for such attacks. We assume the traffic patterns after the malicious manipulation exhibit some deviations from the normal benign traffic. We assume that the IDS and the gateway it resides are trusted and secure.

3 DESIGN OF HERMES

3.1 Workflow

The workflow of Hermes is depicted in Fig. 1, offering an overview of our system. The pre-training phase begins with the NetFlow collector gathering packet data from network gateways (①). This data is processed and then relayed to the NetFlow exporter (②), who creates structured flow records (③). The training phase (④) involves two critical steps: first, we construct an initial anomaly detection model $f(x; \theta_0)$ using contrastive feature learning to define a baseline. We then refine this model from $f(x; \theta_0)$ to $f(x; \theta_1)$, and develop the identification model $g(y; \theta_2)$ through H-Score feature learning (as shown in Fig. 2). After the fully trained IDS model is deployed at the gateway $\mathcal{G}$ (⑤). In dual inference, Hermes utilizes a similarity-based rule for anomaly detection and a MAP rule for attack identification and analyzes if this intrusion is a zero-day attack. The system's pivotal elements, contrastive feature learning, H-Score feature learning, and dual inference, are expounded in the following sections.

3.2 Training Step One: Contrastive Feature Learning

To establish a representative baseline for anomaly detection, we employ geometrical feature extraction through contrastive feature learning. This method leverages the inherent differences and similarities in network traffic to enhance the performance of anomaly-based IDS. Specifically, each training data record in our system consists of two components: the flow record $x \in \mathbb{R}^d$, where d represents the dimensionality of the flow record, and its corresponding output label $y \in \{0, 1\}$. Here, 0 denotes benign traffic flow, while 1 indicates an intrusion. The primary aim of our contrastive learning algorithm is to extract a representative benign feature cluster as a baseline. To achieve this, we train a neural network $f(x; \theta_0)$ with weights θ_0, which takes the input record $x \in \mathbb{R}^d$ and produces a feature $f(x) \in \mathbb{R}^k$. Let x_i represent a benign input and x'_m an intrusive input. Given our dataset includes N instances of normal traffic and M intrusion cases, we compute features for each benign pair of $f(x_i)$ and $f(x_j)$. The goal is to maximize the similarity between $f(x_i)$ and $f(x_j)$, while simultaneously distinguishing between benign $f(x_i)$ and all malicious $f(x'_m)$ instances, where $f(x'_m)|_{m\in\{M\}}$. In this context, $\mathbf{u}_i = f(x_i)$ and $\mathbf{u}_j = f(x_j)$ are the features of benign inputs x_i and x_j, respectively, and $\mathbf{v}_m = f(x'_m)$ is the feature of an intrusive input x'_m. We use the same pairwise loss function as defined in [34] to achieve these goals.

$$l_{ij} = -\log\left(\frac{\exp(\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j/\tau)}{\exp(\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j/\tau) + \sum_{m=1}^{M}\exp(\mathbf{u}_i^{\mathrm{T}}\mathbf{v}_m/\tau)}\right). \tag{1}$$

The term $0 < \tau < 1$ represents the temperature coefficient. The overall loss function $\mathcal{L}$ aggregates all pairwise losses, encompassing both l_{ij} and l_{ji} terms:

$$\mathcal{L} = \frac{1}{N(N-1)}\sum_{i=1}^{N}\sum_{j=i+1}^{N}(l_{ij} + l_{ji}). \tag{2}$$

Phase one of the training, as depicted in Fig. 2, offers an intuitive visualization of the learning process. In this phase, by minimizing

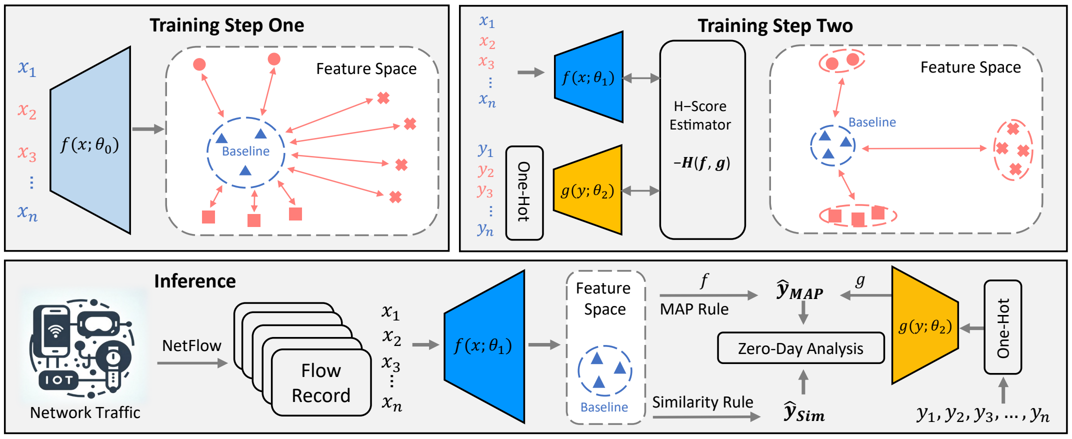

Figure 2: Hermes Design: Our approach begins with contrastive feature learning to train a model $f(x;\theta_0)$, establishing a baseline space (Training Step One). Subsequently, we refine this model to $f(x;\theta_1)$ and develop the identification model $g(y;\theta_2)$ using H-Score feature learning (Training Step Two). During the Inference phase, the established IDS model employs a similarity-based rule for anomaly detection and a MAP rule for attack identification. We further analyze each instance to ascertain whether it could be a zero-day attack, based on the results from dual inference.

the loss function, we enhance the similarity among benign data representations, effectively drawing them closer together. Concurrently, this process distances benign representations from those indicative of intrusions. Once the model converges, it results in a well-defined baseline for normal traffic, which is essential for effective anomaly detection.

3.3 Training Step Two: H-Score Feature Learning

While our current model is adept at anomaly detection, it cannot identify specific types of attack and the baseline is not optimal yet. To address this limitation, we employ H-Score feature learning [36, 37], which explores the geometric feature space beyond the established baseline. In essence, H-Score learning geometrically maps the feature to construct unique clusters for each attack type. This enables the identification of specific attack types during the inference phase. Additionally, it increases the separability between features of different classes. This approach further refines the baseline feature space, distancing it from intrusion clusters, which in turn enhances the performance of the anomaly-based IDS.

To begin, we define a variable X indicating flow records, and a variable Y as corresponding labels. In this context, each label y_i is an integer where $y_i \in \{0, 1, \ldots, n-1\}$; here, 0 represents a benign label, while any non-zero number corresponds to a specific type of attack. These variables, X and Y, follow the joint distribution $P_{X,Y}$. Given the complexity of traffic encountered by IDS, this joint distribution $P_{X,Y}$ is typically unknown, making the direct computation or estimation of the statistical dependence, as indicated by the CDK (canonical dependence kernel) function $\mathfrak{i}_{X;Y}$ (referenced in Eq. 4 in [38]), unfeasible.

$$\mathfrak{i}_{X;Y} = \frac{P_{X,Y}(x,y) - P_X(x)P_Y(y)}{P_X(x)P_Y(y)}, \tag{3}$$

Consequently, we focus on learning $\mathfrak{i}_{X;Y}$ from data samples using H-Score feature learning, specifically by considering its rank-k approximation $\zeta_{\leq k}(\mathfrak{i}_{X;Y})$, given $k \geq 1$. The approximation of $\mathfrak{i}_{X;Y}$ by the rank-k joint function is expressed as $f \otimes g = \sum_{i=1}^{k} f_i \otimes g_i$, where $f \in \mathcal{F}_{\mathcal{X}}^k$ and $g \in \mathcal{F}_{\mathcal{Y}}^k$ represent k-dimensional features. This formulation translates the computation of $\zeta_{\leq k}(\mathfrak{i}_{X;Y})$ into an optimization problem, aiming to minimize the approximation error $\|\mathfrak{i}_{X;Y} - f \otimes g\|_{\mathcal{F}}$, where $\|\cdot\|_{\mathcal{F}}$ is as defined in Table 1. The optimization variables in this context are the k-dimensional features f and g. However, the error $\|\mathfrak{i}_{X;Y} - f \otimes g\|_{\mathcal{F}}$ for a specific f and g cannot be directly computed due to the unknown nature of $\mathfrak{i}_{X;Y}$. To overcome this challenge, we utilize the H-Score $\mathscr{H}(f,g)$, defined for $k \geq 1$ and $f \in \mathcal{F}_{\mathcal{X}}^k$, $g \in \mathcal{F}_{\mathcal{Y}}^k$, as

$$\mathscr{H}(f,g) \triangleq \frac{1}{2}\left(\left\|\mathfrak{i}_{X;Y}\right\|_{\mathcal{F}}^2 - \left\|\mathfrak{i}_{X;Y} - f \otimes g\right\|_{\mathcal{F}}^2\right)$$

$$= \mathbb{E}\left[f^{\mathrm{T}}(X)g(Y)\right] - \left(\mathbb{E}\left[f(X)\right]\right)^{\mathrm{T}}\mathbb{E}\left[g(Y)\right] - \frac{1}{2}\cdot \mathrm{tr}\left(\Lambda_f \Lambda_g\right), \tag{4}$$

Where $\Lambda_f = \mathbb{E}\left[f(X)f^{\mathrm{T}}(X)\right]$ and $\Lambda_g = \mathbb{E}\left[g(Y)g^{\mathrm{T}}(Y)\right]$, with $\mathrm{tr}(\cdot)$ denoting the trace function. The H-Score serves as a measure of the quality of the approximation, where a higher H-Score value indicates a lower approximation error. Accordingly, the optimal weights refined through the H-Score are designed to maximize the correlation between the traffic flow record and its specific types. This approach not only maximizes the average distance between different classes but also minimizes the distance within the same class, thereby enhancing class separability. This detailed discussion can be found in [36, Proposition 4], with its proof provided in Appendix F therein.

In the subsequent discussion, we delve into how features are integrated to discern the specific type of traffic. Assuming we have obtained $f \in \mathcal{F}_{\mathcal{X}}^k$ and $g \in \mathcal{F}_{\mathcal{Y}}^k$ by maximizing the H-Score $\mathscr{H}(f,g)$, we determine the traffic flow where $k \geq \operatorname{rank}(\mathbf{i}_{X;Y})$, having established that $f \otimes g = \mathbf{i}_{X;Y}$. With the above assumption, we can obtain the following result, as discussed in [38, Section 3.2].

Suppose $f \otimes g = \mathbf{i}_{X;Y}$. Then, we will have

$$\|\mathbf{i}_{X;Y}\|_{\mathcal{F}}^2 = \operatorname{tr}(\Lambda_f \cdot \Lambda_g), \tag{5}$$

$$P_{Y|X}(y|x) = P_Y(y)(1 + f^{\mathrm{T}}(x)g(y)). \tag{6}$$

Using the features f and g, we are able to compute the strength of the dependence between X and Y, represented as $\mathbf{i}_{X;Y}$. The posterior distribution, as outlined in Eq. 6 is instrumental in identifying the specific type of traffic. In Hermes, where X represents the input records and Y denotes the types, the system obtains the results of the posterior distribution $P_{Y|X}$ of the label Y by Eq. 6. The traffic type is determined through a corresponding maximum a posteriori (MAP) estimation, defined as follows:

$$\hat{y}_{\mathrm{MAP}}(x) = \arg\max_{y \in \mathcal{Y}} P_{Y|X}(y|x) \tag{7}$$

$$= \arg\max_{y \in \mathcal{Y}} P_Y(y)(1 + f^{\mathrm{T}}(x)g(y)), \tag{8}$$

where P_Y can be sourced from the monitored daily traffic. This method is proposed and explained in [38] Eq. 25.

A visual representation of H-Score feature learning is illustrated in Fig. 2. In Training Step Two, the primary loss function used is $-\mathscr{H}(f, g)$. This function is used for calculating the optimal parameters θ_1 and θ_2 during the second step of the training phase. It is specifically designed for the joint training of the parameters θ_1 and θ_2 in a two-network architecture. In this architecture, the neural network $f(x; \theta_1)$ is responsible for generating k-dimensional features f in the feature space $\mathcal{F}_{\mathcal{X}}^k$. The identification model $g(y; \theta_2)$ is a fully connected network, given the underlying distribution of Y. This network takes the one-hot encoded vector $[\mathbb{1}\{y = 1\}, \ldots, \mathbb{1}\{y = |n|\}]$ of y as input, and produces the k-dimensional features g from labels in the feature space $\mathcal{F}_{\mathcal{Y}}^k$ as output. Thus, the H-Score estimator evaluates the H-Score between f and g to capture the dependence between traffic and the corresponding type. The main goal during the second step of the training phase is to maximize this H-Score for geometrically optimal representation of features, as depicted in the training step two of Figure 2.

3.4 Dual Inference Phase

The inference phase, as shown in Figure 2, is designed to detect the intrusion, identify the type, and explore zero-day attacks. In this phase, two rules are employed, both utilizing the same features extracted from the model $f(x; \theta_1)$. The first rule, a similarity-based detection rule, measures the deviation of traffic features from the learned baseline, thereby enabling anomaly detection. The second rule, a Maximum A Posteriori rule, identifies the type of attack. It compares the posteriori distribution for all types of traffic and outputs the type with the maximum value as the classification result. The dual inference mechanism also evaluates if an instance is a zero-day attack using **entropy-based uncertainty** from the posterior distribution and **inconsistency** between the two predictions.

3.4.1 Rule 1: Similarity-Based Detection. The anomaly-based detection in our system utilizes a threshold-based mechanism based on similarity measures. This process starts with the aggregation of features from the baseline space. We then compute the mean of these normalized features to establish an anchor baseline template, denoted as $\bar{z}$:

$$\bar{z} = \frac{1}{\sum_i \mathbb{1}(y_i = 0)} \sum_i \left(\mathbb{1}(y_i = 0) \frac{f(x_i)}{\|f(x_i)\|_2} \right), \tag{9}$$

Where $\mathbb{1}(\cdot)$ denotes the indicator function, with $y_i = 0$ signify the benign traffic, and $\|\cdot\|_2$ represents the L^2-norm.

To assess the similarity $S(x_i)$ between a new traffic flow x_i and the anchor baseline template $\bar{z}$, we employ the similarity estimator:

$$S(x_i) = \frac{\bar{z}^{\mathrm{T}} f(x_i)}{\|\bar{z}\| \times \|f(x_i)\|}. \tag{10}$$

This similarity score ranges between 0 and 1. In order to classify a new traffic flow x_i as normal or anomalous, we set a threshold score ρ within the range of $[0, 1]$. This threshold is derived from the distribution of similarity scores of the benign training data. We start by sorting these scores in ascending order, resulting in a sequence $\mathrm{S} = [S_1, S_2, \ldots, S_N]$. The threshold ρ is then determined by selecting the $p - th$ percentile of this sequence:

$$r = \left\lfloor \frac{p}{100} \cdot N \right\rfloor, \tag{11}$$

where $\lfloor \cdot \rfloor$ denotes the floor function, and p represents a percentage. As a result, the threshold ρ is set to S_r, which corresponds to the $r - th$ value in the sorted sequence S. To minimize the False Positive Rate, a low percentile p is recommended. An observation x is predicted as an intrusion if its similarity score falls below this threshold ρ:

$$\hat{y}_{Sim}(x) = \mathbb{1}(S(x) < \rho). \tag{12}$$

3.4.2 Rule 2: Maximum A Posteriori Rule for Attack Identification. The second rule, a MAP rule, can help to identify the type of attack. It does this by comparing the correlation between the features extracted from the traffic and the geometric features corresponding to specific attack types. The neural network $f(x; \theta_1)$, processes these records to extract corresponding features, represented as $f(x)$. The identification model $g(y; \theta_2)$ generates the feature for each class, denoted as $g(y)$. For a new record x, we predict the corresponding attack type using the MAP rule as:

$$\hat{y}_{\mathrm{MAP}}(x) = \arg\max_{y \in \mathcal{Y}} \left\{ P_Y(y)[1 + f^{\mathrm{T}}(x)g(y)] \right\}, \tag{13}$$

where $P_Y(y)$ is the true traffic type distribution that can be monitored from the daily network flow traffic.

3.5 Zero-Day Attack Analysis

The dual inference of Hermes is adept at discerning whether an attack could be a zero-day attack.

Suppose an instance is a zero-day attack; it will exhibit new characteristics and previously unseen signatures, leading to increased uncertainty in the model's predictions. This uncertainty is evaluated based on the entropy of the posterior distribution (see Equation 6): $P_{Y|X}(y|x) = P_Y(y) \cdot (1 + f^{\mathrm{T}}(x)g(y))$. The features of a zero-day attack may manifest in a feature space distinct from any known attack's space, resulting in no dominant probability for any known attack in the posterior distribution and consequently, a higher entropy. This metric can help to determine if it is a zero-day attack.

Algorithm 1 Training Phase

```
1:  Training Step One: Contrastive Feature Learning
2:  Input: Flow Records with benign inputs $\{(x_i)\}_{i=1}^N$ and malicious inputs $\{(x'_i)\}_{i=1}^M$
3:  Output: Optimized parameters $\theta_0$ for $f(x;\theta_0)$
4:  for each epoch in num_epochs do
5:      for each benign input pair $(x_i, x_j)$ do
    /* Compute the features */
6:          $\mathbf{u}_i \leftarrow f(x_i)$, $\mathbf{u}_j \leftarrow f(x_j)$
7:          $\mathbf{v}_m \leftarrow f,(x'_m)$
    /* Compute the pairwise loss $l_{ij}$ */
8:          $l_{ij} = -\log(\frac{\exp(\mathbf{u}_i^T\mathbf{u}_j/\tau)}{\exp(\mathbf{u}_i^T\mathbf{u}_j/\tau)+\sum_{m=1}^M \exp(\mathbf{u}_i^T\mathbf{v}_m/\tau)})$
9:      end for
    /* Compute the cumulative loss function $\mathcal{L}$ */
10:     $\mathcal{L} = \frac{1}{N(N-1)}\sum_{i=1}^N\sum_{j=i+1}^N l_{ij} + l_{ji}$
    /* Update the parameters */
11:     $\theta_0 = \theta_0 - \alpha\nabla\mathcal{L}$,
12: end for

13: Training Step Two: H-Score Feature Learning
14: Input: Flow Records with labels $\{(x_i, y_i)\}_{i=1}^N$
15: Output: Optimized parameters $\theta_1$ for $f(x;\theta_1)$, and $\theta_2$ for $g(y;\theta_2)$
16: for each epoch in num_epochs do
17:     for each $(x_i, y_i)$ in Training data $\{(x_i, y_i)\}_{i=1}^N$ do
    /* Convert labels $y$ to one-hot encoding $y_{\text{one-hot}}$*/
18:         $y_{\text{one-hot}}[i] = \begin{cases} 1 & \text{if } i = y \\ 0 & \text{otherwise} \end{cases}$
    /* Get $f$ and $g$ values using $x$ and $y_{\text{one-hot}}$*/
19:         $f = \{f(x_i)\}_{i=1}^N$, $g = \{g(y_{\text{one-hot i}}))\}_{i=1}^N$
    /* Compute the negative H-Score as loss, $\mathcal{L}$ */
20:         $\mathcal{L} = -\mathscr{H}(f, g)$
    /* Update the models' parameters */
21:         $\theta_1 = \theta_1 - \alpha\nabla\mathcal{L}$, $\theta_2 = \theta_2 - \alpha\nabla\mathcal{L}$
22:     end for
23: end for
```

We define this uncertainty as follows:

$$\mathscr{X}(P_{Y|X}(y|x)) = -\sum_i P_{Y|X}(y_i|x)\log P_{Y|X}(y_i|x). \quad (14)$$

Consistent results from similarity-based and MAP rules are typically observed in detection scenarios. However, inconsistencies may arise, such as when the similarity rule indicates an instance is benign, while the MAP rule identifies it as an attack, or vice versa. These inconsistencies are more likely to occur when the features of a zero-day attack are geometrically situated in the mid-area between the feature spaces of different types of traffic. Therefore, if an instance is a zero-day attack, it is more likely to provoke inconsistent predictions between the similarity and MAP rules, necessitating further investigation to confirm if it is a zero-day attack. We have validated these findings through observations and experimental results.

We summarize the whole training phase and dual inference procedures in Algorithms 1 and 2, and compose the notations in

Algorithm 2 Dual Inference

```
1:  Input: Flow Records $\{(x_i)\}_{i=1}^N$
2:  Output: Anomaly detection results: $\{\hat{y}_{Sim}(x_i)\}_{i=1}^N$ , and the traffic types $\{\hat{y}_{MAP}(x_i)\}_{i=1}^N$.
    /* Compute normalized features for benign traffic */
3:  for all benign records $x_i$ do
4:      $z_{sum} \leftarrow z_{sum} + \frac{f(x_i)}{\|f(x_i)\|_2}$
5:  end for
6:  $\bar{z} \leftarrow \frac{z_{sum}}{\text{number of benign records}}$
7:  for each $x_i$ in Flow Records $\{(x_i)\}_{i=1}^N$ do
    /*Compute the similarity $S$ for new traffic flow $x_i$ */
8:      $S(x_i) \leftarrow \frac{\bar{z}^T f(x_i)}{\|\bar{z}\|\times\|f(x_i)\|}$
    /*Sort similarity scores of benign traffic and compute $\rho$*/
9:      S ← sorted similarity scores of benign traffic
10:     $r \leftarrow \lfloor \frac{p}{100} \cdot N \rfloor$
11:     $\rho \leftarrow S_r$
    /* Predict new observation based on similarity score and threshold */
12:     $\hat{y}_{Sim}(x_i) = \mathbb{1}(S(x_i) < \rho)$
    /* Predict the traffic by MAP rule $\hat{y}(x)$ */
13:     Estimate the prior distribution $P_y(y)$
14:     $\hat{y}_{MAP}(x_i) = \arg\max_{y\in\mathcal{Y}} P_Y(y)\cdot(1 + f^T(x_i)g(y))$
    /* Zero-Day Attack Analysis */
15:     $\mathscr{X}(P_{Y|X}) = -\sum_j P_{Y|X}(y_j|x_i)\log P_{Y|X}(y_j|x_i)$
16:     if $\mathscr{X}(P_{Y|X})$ is an outlier then
17:         return zero-day attack
18:     else if $\hat{y}_{MAP}(x)$ consist with $\hat{y}_{Sim}(x)$ then
19:         return not zero-day attack
20:     else if $\hat{y}_{MAP}(x)$ not consist with $\hat{y}_{Sim}(x)$ then
21:         return zero-day attack
22:     end if
23: end for
```

Tab. 1. The whole illustration for H-Score feature learning can be found in [38].

4 EXPERIMENT

To evaluate the effectiveness of Hermes, we assess its capabilities in three key areas: anomaly detection, attack identification, and exploration of zero-day attack performance. Concurrently, we benchmark the performance of Hermes against a range of baseline methods and state-of-the-art anomaly-based IDS.

4.1 Datasets and Experiment Settings

We prototype Hermes using the PyTorch framework [25] and evaluate on two prominent network traffic datasets: NSL-KDD [30] and N-BaIoT [20]. The NSL-KDD dataset, a well-established benchmark in IDS performance evaluation [6, 24, 28], includes benign traffic along with four categories of intrusions, in Table 3. We highlight in bold text the 16 intrusion sub-classes that are only present in the test set. This presents a significant challenge to Hermes in detecting emerging zero-day threats. Each record in the dataset features 41 attributes extracted from network traffic. These evaluations were conducted on a server equipped with an Intel Core i9-11900K CPU

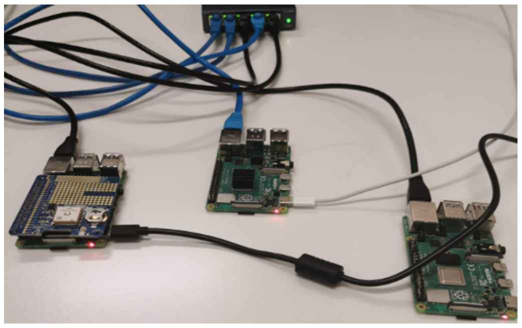

Figure 3: Evaluating Hermes on Raspberry Pi 4, an IoT device in a resource-constrained environment.

@ 3.50GHz×16, a GeForce RTX 3080 GPU, and running Ubuntu 20.04.5 LTS.

The N-BaIoT dataset focuses on IoT traffic, encompassing data from nine commercial IoT devices, including those compromised by malicious software, such as Mirai and BASHLITE. This dataset records benign traffic and five types of attacks for both Mirai ('scan', 'ack', 'syn', 'udp', and 'udpplain') and BASHLITE ('scan', 'junk', 'udp', 'tcp', and 'combo') [20]. A principal goal to use this dataset is to verify the applicability of Hermes in resource-constrained IoT environments. Accordingly, our experiments with the N-BaIoT dataset were conducted on a Raspberry Pi 4 as depicted in Fig. 3, equipped with a 1.5 GHz quad-core A72 64-bit ARMV8 CPU, 4GB RAM, 32GB SIM card storage, and running the Raspbian OS, to evaluate Hermes's efficiency in such settings.

The default configurations of Hermes are outlined as follows. For the NSL-KDD dataset, the network $f(x;\theta_0)/f(x;\theta_1)$ is composed of an input layer with $d = 112$ dimensions and an output layer with $k = 256$ dimensions. Additionally, it includes two hidden layers with sizes of 128 and 256. The $g(y;\theta_2)$ network, designed for attack identification, utilizes a three-layer structure with 256, 128, and 5 neurons. For the N-BaIoT dataset, the architecture largely mirrors that of the NSL-KDD. The key difference is in the input and output layers of the $f(x;\theta_0)/f(x;\theta_1)$ network, which are configured to accommodate 115 feature inputs and 11 class outputs. The ReLU (Rectified Linear Unit) activation function is implemented after each hidden layer in both models to introduce non-linearity. Hermes employs the Adam optimizer with parameters that include a learning rate of $\alpha = 1e-4$ and a batch size of $b = 128$. The training regimen varies between datasets: for the first phase, $e = 20$ epochs are used for NSL-KDD and $e = 15$ for the N-BaIoT dataset. In the second training phase, an additional 20 epochs are applied to both datasets. For comprehensive analysis, Hermes's performance is benchmarked against machine learning models from the renowned scikit-learn library [26].

4.2 Evaluation Metric and Baselines

For intrusion detection system performance evaluation, we compute key metrics: *Accuracy*, *Recall*, *Precision*, *F1 Score*, and *False Positive Rate (FPR)*, following the same definition in [23, 24, 28]. We also provide the receiver operating characteristic curve (ROC), plotting recall against FPR, and the area under the curve (AUC) score representing the area under this curve. Our experimental configuration is as follows: **Hermes_Sim** employs $f(x;\theta_1)$ with the similarity rule for anomaly detection. **Hermes_MAP** uses $f(x;\theta_1)$ and $g(y;\theta_2)$ with the MAP rule for attack identification. **CL** utilizes the architecture of $f(x;\theta_1)$, It adheres to the similarity rule. The multi-layer perceptron (**MLP**) model integrates the architectures of $f(x;\theta_1)$ and a classification head, optimized using cross-entropy loss. Comparative performance evaluations include models such as support vector machine (SVM), variational autoencoder (VAE), isolation forest (IsoForest), logistic regression (LGR), Bernoulli naive Bayes (BNB), K-nearest neighbors (KNN), and decision tree classifier (DTC), Along with IDS methods, Two-Tier [28], TDTC [23], ESFCM [24], and contrastive-learning-based variants, AOC-IDS [45], FeCo [34], and CIDS [43].

Table 2: Performance (%) of Hermes on NSL-KDD Dataset

Method	Acc	Recall	Precis	F1	FPR
Hermes_Sim	90.17	85.55	96.80	90.83	3.74
CL	87.40	81.89	95.31	88.09	5.32
MLP	85.22	77.92	95.29	85.68	5.10
IsoForest	79.54	69.35	92.90	79.42	7.00
SVM	77.16	72.80	84.92	78.39	17.08
VAE	78.33	85.50	77.88	81.96	31.46
LGR	77.41	72.24	85.83	78.45	15.76
BNB	78.43	65.66	94.88	77.61	4.69
KNN	78.50	64.37	96.79	77.32	2.82
DTC	79.09	74.60	86.81	80.24	14.98
AOC-IDS	89.51	96.59	86.54	91.29	19.84
FeCo	89.55	86.80	94.39	90.44	6.82
CIDS	82.29	77.18	93.54	80.65	-
ESFCM	80.69	80.72	80.85	80.45	-
Two-Tier	-	82.00	-	-	5.43
TDTC	-	84.86	-	-	4.86

Hermes_MAP	Recall	Precis	F1	FPR	Ratio
Normal	88.96	92.66	90.77	7.18	50.48
DoS	84.12	99.79	91.29	0.09	35.86
Probe	92.42	91.62	92.02	0.89	9.56
R2L	94.98	56.04	70.49	3.06	3.96
L2R	66.67	1.74	3.38	4.85	0.13
Weighted Avg	87.77	93.55	90.16	3.87	-

Note: Acc: 87.77 is a multi-class accuracy.

Table 3: Intrusion classes (sub-classes) of the NSL-KDD Dataset.

Category	Sub-Classes(**bold zero-day attacks**)
DoS(10)	back, land, Neptune, pod, smurf, teardrop, **apache2**, **mailbomb**, **processtable**, udpstorm
Probe(6)	ipsweep, nmap, portsweep, satan, **mscan**, **saint**
R2L(16)	ftp_write, guesspasswd, imap, multihop, phf, warezmaster, **httptunnel**, **named**, **sendmail**, **snmpgetattack**, **snmpguess**, **xlock**, **xsnoop**, **worm**, spy, warezclient
U2R(7)	bufferoverflow, loadmodule, perl, rootkit, **ps**, **sqlattack**, **xterm**

4.3 Experimental Results

4.3.1 Feature Space Insights. We utilize t-SNE—a dimensionality reduction technique to project the features space, $\mathcal{F}_{\mathcal{X}}^{k}$, into a two-dimensional figure as shown in Fig. 4. In the presented visualizations

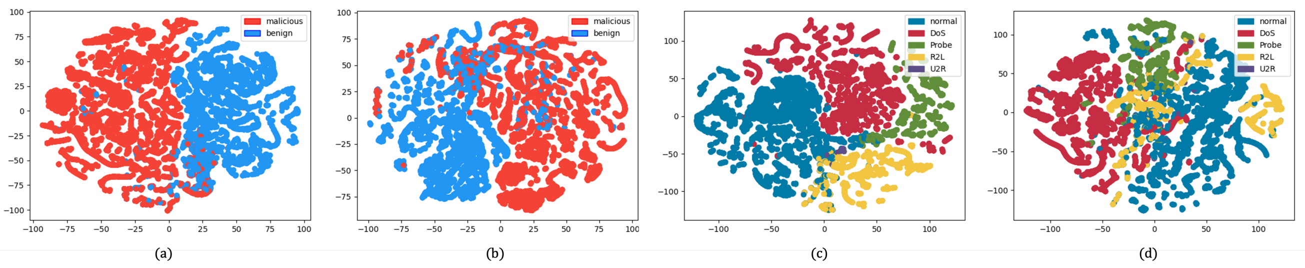

Figure 4: Visualization of $\mathcal{F}_{\mathcal{X}}^{k}$ and MLP Output on a Randomly Selected 5% Subset of the NSL-KDD Test Dataset for: (a) Hermes Anomaly-Detection, (b) MLP Anomaly-Detection, (c) Hermes Attack Identification, and (d) MLP Attack Identification.

Table 4: Devices Specification, and Performance(%) of Hermes_Sim and Hermes_MAP on the N-BaIoT Dataset.

Devices Specification			Hermes_Sim						Hermes_MAP				
Index	Device Make and Model	Device Type	Acc1	Recall1	Precis1	F1_1	FPR1	FPR(CL)	Acc2	Recall2	Precis2	F1_2	FPR2
0	Danmini	Doorbell	99.98	99.99	99.99	99.99	0.03	0.28	97.30	97.31	97.30	97.30	0.82
1	Ennio	Doorbell	99.98	99.99	99.99	99.99	0.08	0.31	96.66	96.63	96.66	96.63	0.98
2	Ecobee	Thermostat	99.86	99.97	99.77	99.87	0.28	0.31	96.19	96.19	96.24	96.09	1.21
3	Phillips B120N/10	Baby monitor	99.97	99.97	99.96	99.97	0.03	0.06	96.83	96.82	96.83	96.82	0.93
4	Provision PT-737E	Security camera	99.94	99.99	99.92	99.95	0.15	0.26	96.00	96.02	96.00	95.97	1.22
5	Provision PT-838	Security camera	99.95	99.99	99.92	99.95	0.09	0.13	97.15	97.15	97.15	97.13	0.82
6	SimpleHome XCS7-1002-WHT	Security camera	99.98	99.97	99.99	99.98	0.01	0.52	96.69	96.70	96.69	96.68	0.99
7	SimpleHome XCS7-1003-WHT	Security camera	99.98	99.97	99.99	99.98	0.02	0.39	95.53	95.71	95.53	95.39	1.43
8	Samsung SNH 1011 N	Webcam	99.97	99.98	99.99	99.99	0.07	0.68	95.56	95.75	95.56	95.43	1.46

Note: Acc2 is a multi-class accuracy, while Recall2, Precis2, F1_2, and FPR2 are weighted class-wise averages.

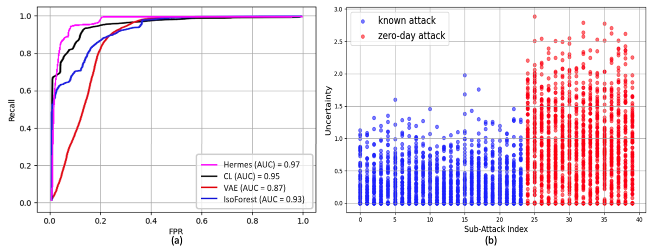

Figure 5: (a) The ROC curve of Hermes_Sim. (b) Uncertainty analysis for subclass traffics in exploring zero-day attacks.

of $\mathcal{F}_{\mathcal{X}}^{k}$ with a randomly selected 5% subset of the NSL-KDD test dataset, distinct patterns emerge when comparing Hermes and the MLP method across both binary anomaly detection and multi-class attack identification. For anomaly detection, the Hermes approach in (a) exhibits a more discernible separation between the "malicious" and "benign" clusters, whereas the MLP model in (b) shows a considerable overlap between the two categories. (c) Hermes and (d) MLP demonstrate multi-cluster separations. Hermes in (c) maintains better-delineated boundaries between the classes, especially when differentiating subtle distinctions like between "DoS", "Probe", "U2R", and "R2L". In contrast, the MLP's features in (d) show more intermingling among classes.

4.3.2 Hermes Performance on NSL-KDD. In Tab. 2, Hermes_Sim achieves high accuracy, recall, precision, and F1 score among all tested methods while maintaining an FPR as low as 3.74%.

The **ablation study** demonstrates that adding contrastive feature learning to the basic MLP method results in improvements across all metrics, including a 3.97% in recall, while keeping a similar FPR. Introducing H-Score feature learning further enhances performance, with overall improvements of 7.63% in recall, while reducing the FPR by 1.36% compared to the baseline MLP. **Detection Speed:** The average inference time (over 100 records) is 3.58 ms for MLP, 3.20 ms for CL, and 4.25 ms for Hermes_Sim.

The ROC curves of Hermes and three other baselines are plotted in Fig. 5 (a), with our method exhibiting a higher AUC score compared to the baselines. For intrusion identification, Tab. 2 shows that Hermes_MAP attains an 87.77% accuracy in identifying four classes of intrusions amid normal traffic. The table's lower section details the identification performance for each traffic class, alongside their ratio in the training data. With adequate samples, Hermes consistently maintains high accuracy. Remarkably, it effectively identifies the L2R attack, which represents a mere 0.13% of the training dataset, with a low FPR of 4.85%.

To illustrate Hermes's ability to identify zero-day attacks, we display in Fig. 5 (b) the uncertainty for each intrusion subclass and normal traffic, calculated using Eq. 14. The x-axis in the figure represents the intrusion subclasses enumerated in Table 3, with indices 0 to 23 denoting benign and known attacks, and 24 to 39 indicating zero-day attacks. The y-axis is the uncertainty associated with the corresponding traffic. Notably, known traffic types exhibit lower mean and variance in uncertainty, whereas zero-day samples display significantly higher variance and mean. This pattern suggests that high-uncertainty outliers could be potential zero-day attack samples warranting further investigation. Furthermore, by analyzing the discrepancies between outputs from $\hat{y}_{Sim}(x)$ and $\hat{y}_{MAP}(x)$, we find that 77.24% of the intrusions are zero-day attacks.

This insight offers an additional method for identifying potential zero-day attack samples.

4.3.3 Hermes performance on N-BaIoT dataset. We expanded our evaluation of Hermes to include real IoT botnet traffic datasets, originating from nine commercially available IoT devices infected by the Mirai and BASHLITE botnets. A comprehensive breakdown of these devices is provided in Tab. 4. For detailed analysis, we created a specific model for each IoT device and evaluated their performance individually. Tab. 4 details the performance of our methods. All devices showed high rates in all metrics, exceeding 99.6%, with Hermes_Sim consistently achieving a lower FPR than the CL method. Hermes_MAP excelled in attack identification, consistently delivering high performance with a lower FPR among all the devices. These experimental results validate the performance enhancement of Hermes in anomaly detection, attack identification, and exploration of the zero-day attack. They also demonstrate the practicality of Hermes in resource-constrained environments, highlighting its versatility in multi-application scenarios.

5 RELATED WORK

A critical aspect influencing the performance of anomaly-based IDS is the construction of the baseline space. Yan et al. [39] utilized a non-parametric density estimation method to learn legitimate access patterns. K-Nearest Neighbors, as detailed in [23], enhances baseline space representativeness by evaluating the similarity of new data points to established benign instances. Du et al. [9] employs Long Short-Term Memory networks to model system logs as natural language sequences. It learns log features from normal executions and detects anomalies when log features deviate from the learned patterns. Mirsky et al. [21] proposed an ensemble of autoencoders to distinguish between normal and abnormal traffic features. The Variational Autoencoder, explored in [19, 41], excels in feature extraction and dimensionality reduction, crucial for distilling vital features from intricate network data. Nguyen et al. [22] modeled network packets as symbols in a language, enabling the use of Gated Recurrent Units for anomaly detection. Isolation Forest, as discussed in [12, 46], takes a unique approach by not modeling normalcy but instead focusing on the property of anomalies being more 'isolatable' than normal points. State-of-the-art methods, such as FeCo [34] and CIDS [43], employ contrastive learning to enhance the performance of IDSs. AOC-IDS [45] introduced a contrastive loss tailored for Autoencoder to obtain a better baseline representation, thereby facilitating anomaly detection.

To specifically identify the type of intrusion, signature-based IDS [11, 13, 16, 17] can achieve this goal by directly comparing intrusions with known attack database. Wang et al. [32] proposed a graphical model that stores known traffic features as a relational graph between features and their labels to detect DDoS attacks. Yan et al. [40] employed SVM to detect botnets using high-level features extracted from command and control channels. Pajouh et al. [24] introduced TDTC, which utilizes a two-tier classification system to classify traffic. Rathore et al. [28] proposed ESFCM, integrating Fuzzy C-Means with the Extreme Learning Machine classifier for attack detection.

Anomaly-based IDS systems face significant challenges with performance issues and fail to identify the type of intrusion. On the other hand, while signature-based IDS is mature and effective in detecting attacks recorded in databases, it fails to address zero-day attacks. Thus, Hermes represents a novel anomaly-based IDS designed to unify the advantages of these two categories of IDS with high performance.

6 DISCUSSION

The rationale behind our design is as follows: traditionally, signature-based IDS focuses on malicious traffic and extracting their signatures, while the focus in the anomaly-based IDS is normal traffic and establishing the baseline representing the normal behaviors. The tasks of identifying attacks and detecting anomalies, including zero-day, are therefore two separate processes, with each type of IDS doing one. In Hermes, we take a different approach to integrating the two types of IDS using machine learning. Instead of looking at the original traffic representation and building a network traffic classifier, we focus on feature engineering and propose a framework to construct new feature representations of network traffic. While our goal is to build a robust baseline, the core component of anomaly-based IDS, we leverage the knowledge that already exists in reality (i.e., known attacks). This knowledge helps us to redefine the feature space so we can have a more robust baseline traffic representation which leads to more effective detection in an anomaly-based IDS. Further, since our learning process used the known attack traffic, as a side effect, the trained model can recognize the known attacks. The proposed feature learning framework seamlessly integrates the functions of the two types of IDS into a single system.

7 CONCLUSION

In this paper, we propose Hermes, an anomaly-based IDS that utilizes a geometric feature learning to enhance detection performance and reduce false positives. This framework aims to extract optimal features from network traffic, enabling the IDS to detect attacks, identify the type of attack, and evaluate if it is a previously unseen attack. Hermes is a novel anomaly-based IDS that harnesses the advantages of both signature-based and anomaly-based approaches. Extensive evaluations of Hermes across two datasets and diverse hardware platforms have consistently demonstrated its effectiveness, high performance, and practical viability in resource-constrained application scenarios.

8 ACKNOWLEDGMENT

This work was supported in part by the Office of Naval Research under grant N00014-19-1-2621, the US National Science Foundation under grants 1916902, 2154929, 2247560, 2235232, 2312447, and the Virginia Commonwealth Cyber Initiative (CCI).

REFERENCES

[1] Mohammed Ali Al-Garadi, Amr Mohamed, Abdulla Khalid Al-Ali, Xiaojiang Du, Ihsan Ali, and Mohsen Guizani. 2020. A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Communications Surveys & Tutorials* 22, 3 (2020), 1646–1685.
[2] Manos Antonakakis, Tim April, Michael Bailey, et al. 2017. Understanding the mirai botnet. In *26th {USENIX} security symposium ({USENIX} Security 17)*. 1093–1110.

[3] Shahid Anwar, Jasni Mohamad Zain, Mohamad Fadli Zolkipli, Zakira Inayat, Suleman Khan, Bokolo Anthony, and Victor Chang. 2017. From intrusion detection to an intrusion response system: fundamentals, requirements, and future directions. *Algorithms* 10, 2 (2017), 39.

[4] May Bashendy, Ashraf Tantawy, and Abdelkarim Erradi. 2023. Intrusion response systems for cyber-physical systems: A comprehensive survey. *Computers & Security* 124 (2023), 102984.

[5] Ray Canzanese. 2023. Cloud as a malware infiltration channel. *Computer Fraud & Security* 2023, 4 (2023).

[6] Nadia Chaabouni, Mohamed Mosbah, Akka Zemmari, Cyrille Sauvignac, and Parvez Faruki. 2019. Network intrusion detection for IoT security based on learning techniques. *IEEE Communications Surveys & Tutorials* 21, 3 (2019), 2671–2701.

[7] Benoit Claise. 2004. *Cisco systems netflow services export version 9*. Technical Report.

[8] G Dayanandam, TV Rao, D Bujji Babu, and S Nalini Durga. 2019. DDoS attacks—analysis and prevention. In *Innovations in Computer Science and Engineering: Proceedings of the Fifth ICICSE 2017*. Springer, 1–10.

[9] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 1285–1298.

[10] Gianni D'Angelo and Francesco Palmieri. 2021. Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial–temporal features extraction. *Journal of Network and Computer Applications* 173 (2021), 102890.

[11] Mojtaba Eskandari, Zaffar Haider Janjua, Massimo Vecchio, and Fabio Antonelli. 2020. Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices. *IEEE Internet of Things Journal* 7, 8 (2020), 6882–6897.

[12] Yulin Fan, Yang Li, Mengqi Zhan, Huajun Cui, and Yan Zhang. 2020. Iotdefender: A federated transfer learning intrusion detection framework for 5g iot. In *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)*. IEEE, 88–95.

[13] Pedro Garcia-Teodoro, Jesus Diaz-Verdejo, Gabriel Maciá-Fernández, and Enrique Vázquez. 2009. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security* 28, 1-2 (2009), 18–28.

[14] Hanan Hindy, Robert Atkinson, Christos Tachtatzis, Jean-Noël Colin, Ethan Bayne, and Xavier Bellekens. 2020. Utilising deep learning techniques for effective zero-day attack detection. *Electronics* 9, 10 (2020), 1684.

[15] Jordan Holland, Paul Schmitt, Nick Feamster, and Prateek Mittal. 2021. New directions in automated traffic analysis. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 3366–3383.

[16] VVRPV Jyothsna, Rama Prasad, and K Munivara Prasad. 2011. A review of anomaly based intrusion detection systems. *International Journal of Computer Applications* 28, 7 (2011), 26–35.

[17] Okan Kopuklu, Jiapeng Zheng, Hang Xu, and Gerhard Rigoll. 2021. Driver anomaly detection: A dataset and contrastive learning approach. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 91–100.

[18] Shaoyu Li, Shanghao Shi, Yang Xiao, Chaoyu Zhang, Y Thomas Hou, and Wenjing Lou. 2023. Bijack: Breaking bitcoin network with tcp vulnerabilities. In *European Symposium on Research in Computer Security*. Springer, 306–326.

[19] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, and Jaime Lloret. 2017. Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in iot. *Sensors* 17, 9 (2017), 1967.

[20] Yair Meidan, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Asaf Shabtai, Dominik Breitenbacher, and Yuval Elovici. 2018. N-baiot—network-based detection of iot botnet attacks using deep autoencoders. *IEEE Pervasive Computing* 17, 3 (2018), 12–22.

[21] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. Kitsune: an ensemble of autoencoders for online network intrusion detection. *arXiv preprint arXiv:1802.09089* (2018).

[22] Thien Duc Nguyen, Samuel Marchal, Markus Miettinen, et al. 2019. DÏoT: A federated self-learning anomaly detection system for IoT. In *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 756–767.

[23] Hamed Haddad Pajouh, GholamHossein Dastghaibyfard, and Sattar Hashemi. 2017. Two-tier network anomaly detection model: a machine learning approach. *Journal of Intelligent Information Systems* 48, 1 (2017), 61–74.

[24] Hamed Haddad Pajouh, Reza Javidan, Raouf Khayami, Ali Dehghantanha, and Kim-Kwang Raymond Choo. 2019. A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks. *IEEE Transactions on Emerging Topics in Computing* 7, 02 (2019), 314–323.

[25] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019), 8026–8037.

[26] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[27] Segun I Popoola, Ruth Ande, Bamidele Adebisi, Guan Gui, Mohammad Hammoudeh, and Olamide Jogunola. 2021. Federated deep learning for zero-day botnet attack detection in IoT-edge devices. *IEEE Internet of Things Journal* 9, 5 (2021), 3930–3944.

[28] Shailendra Rathore and Jong Hyuk Park. 2018. Semi-supervised learning based distributed attack detection framework for IoT. *Applied Soft Computing* 72 (2018), 79–89.

[29] Shahadate Rezvy, Yuan Luo, Miltos Petridis, Aboubaker Lasebae, and Tahmina Zebin. 2019. An efficient deep learning model for intrusion classification and prediction in 5G and IoT networks. In *2019 53rd Annual Conference on information sciences and systems (CISS)*. IEEE, 1–6.

[30] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A Ghorbani. 2009. A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE, 1–6.

[31] Yonglong Tian, Dilip Krishnan, and Phillip Isola. 2020. Contrastive multiview coding. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer, 776–794.

[32] Bing Wang, Yao Zheng, Wenjing Lou, and Y Thomas Hou. 2015. DDoS attack protection in the era of cloud computing and software-defined networking. *Computer Networks* 81 (2015), 308–319.

[33] Ning Wang, Yimin Chen, Yang Hu, Wenjing Lou, and Y Thomas Hou. 2021. MANDA: On Adversarial Example Detection for Network Intrusion Detection System. In *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*. IEEE, 1–10.

[34] Ning Wang, Yimin Chen, Yang Hu, Wenjing Lou, and Y. Thomas Hou. 2022. FeCo: Boosting Intrusion Detection Capability in IoT Networks via Contrastive Learning. In *IEEE INFOCOM 2022 - IEEE Conference on Computer Communications*. 1409–1418. https://doi.org/10.1109/INFOCOM48880.2022.9796926

[35] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3733–3742.

[36] Xiangxiang Xu and Shao-Lun Huang. 2020. Maximal correlation regression. *IEEE Access* 8 (2020), 26591–26601.

[37] Xiangxiang Xu and Lizhong Zheng. 2022. Multivariate Feature Extraction. In *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 1–8. https://doi.org/10.1109/Allerton49937.2022.9929401

[38] Xiangxiang Xu and Lizhong Zheng. 2024. Neural Feature Learning in Function Space. *Journal of Machine Learning Research* 25, 142 (2024), 1–76. http://jmlr.org/papers/v25/23-1202.html

[39] Qiben Yan, Ming Li, Feng Chen, Tingting Jiang, Wenjing Lou, Y Thomas Hou, and Chang-Tien Lu. 2014. SpecMonitor: Toward efficient passive traffic monitoring for cognitive radio networks. *IEEE Transactions on Wireless Communications* 13, 10 (2014), 5893–5905.

[40] Qiben Yan, Yao Zheng, Tingting Jiang, Wenjing Lou, and Y Thomas Hou. 2015. Peerclean: Unveiling peer-to-peer botnets through dynamic group behavior analysis. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 316–324.

[41] Yanqing Yang, Kangfeng Zheng, Chunhua Wu, and Yixian Yang. 2019. Improving the classification effectiveness of intrusion detection by using improved conditional variational autoencoder and deep neural network. *Sensors* 19, 11 (2019), 2528.

[42] Mingxuan Yao, Jonathan Fuller, Ranjita Pai Kasturi, Saumya Agarwal, Amit Kumar Sikder, and Brendan Saltaformaggio. 2023. Hiding in plain sight: an empirical study of web application abuse in malware. In *32nd USENIX Security Symposium (USENIX Security 23)*. 6115–6132.

[43] Yawei Yue, Xingshu Chen, Zhenhui Han, Xuemei Zeng, and Yi Zhu. 2022. Contrastive Learning Enhanced Intrusion Detection. *IEEE Transactions on Network and Service Management* 19, 4 (2022), 4232–4247. https://doi.org/10.1109/TNSM.2022.3218843

[44] Chaoyu Zhang, Ning Wang, Shanghao Shi, Changlai Du, Wenjing Lou, and Y Thomas Hou. 2023. Mindfl: Mitigating the impact of imbalanced and noisy-labeled data in federated learning with quality and fairness-aware client selection. In *MILCOM 2023-2023 IEEE Military Communications Conference (MILCOM)*. IEEE, 331–338.

[45] Xinchen Zhang, Running Zhao, Zhihan Jiang, Zhicong Sun, Yulong Ding, Edith CH Ngai, and Shuang-Hua Yang. 2024. AOC-IDS: Autonomous Online Framework with Contrastive Learning for Intrusion Detection. *IEEE INFOCOM 2024* (2024).

[46] Yuntong Zhang, Jingye Xu, Zhiwei Wang, Rong Geng, Kim-Kwang Raymond Choo, Jesús Arturo Pérez-Díaz, and Dakai Zhu. 2020. Efficient and intelligent attack detection in software defined iot networks. In *2020 IEEE International Conference on Embedded Software and Systems (ICESS)*. IEEE, 1–9.