Computers & Security 142 (2024) 103898

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

# NERO: NEural algorithmic reasoning for zeRO-day attack detection in the IoT: A hybrid approach

Jesús F. Cevallos M., Alessandra Rizzardi, Sabrina Sicari *, Alberto Coen Porisini

*Dipartimento di Scienze Teoriche e Applicate, Universita' degli Studi dell'Insubria, via O. Rossi 9, Varese, 21100, VA, Italy*

ARTICLE INFO

*Keywords:*
Network intrusion detection systems
Internet of things
Neural algorithmic reasoning
Meta-learning

ABSTRACT

Anomaly detection approaches for network intrusion detection learn to identify deviations from normal behavior on a data-driven basis. However, current approaches strive to infer the degree of abnormality of out-of-distribution samples when these appertain to different zero-day attacks. Inspired by the successes of the neural algorithmic reasoning paradigm to leverage the generalization of rule-based behavior, this paper presents a deep learning strategy for solving zero-day network attack detection and categorization. Moreover, focusing on the particular scenario of the Internet of Things (IoT), the privacy preservation requirement may imply a low training data regime for any learning algorithm. To this respect, the presented framework uses metric-based meta-learning to achieve few-shot learning capabilities. The presented pipeline is called **NERO**, as it imports the *encode-process-decode* architecture from the **NE**ural algorithmic reasoning blueprint to converge ze**RO**-day attack detection policies within constrained training data.

## 1. Introduction

IoT threat intelligence is a rapidly evolving field that aims to identify and mitigate security threats in IoT systems (Heidari and Jabraeil Jamali, 2023). Network intrusion detection (NID) is one of the most critical aspects of IoT security. According to the type of detection, NID systems can be classified as anomaly-based and signature-based (Jayalaxmi et al., 2022). Anomaly or *behavior-based* detection is based on the idea of identifying deviations from normal behavior (Khraisat and Alazab, 2021). It involves creating a baseline of expected behavior and monitoring network and device activity for deviations from such a baseline. As a data-driven technique involving statistical analyses, anomaly-based detection is typically implemented through Machine Learning (ML) and Deep Learning (DL) (Santhoshb Kumar et al., 2023). Instead, signature-based detection, often called *rule-based*, involves identifying known malicious patterns or signatures in traffic data. This technique compares traffic-related observations with a known attack patterns or signatures database.

The common advantages of anomaly-based over signature-based NID are associated with detecting previously unknown attacks. Moreover, the potential adaptability to changing threat behaviors allows dynamically and periodically adjusting the definition of normal behavior in anomaly-based approaches. However, anomaly-based detection can produce high false-negative rates, especially when high traffic bursts or other infrequent events occur at the inference phase (Khraisat et al., 2019). Another disadvantage of anomaly-based approaches is the lack of explainability and transparency of many black-box function approximators based on deep Artificial Neural Networks (ANN).

In this regard, the research community has recently reawakened an interest in the benefits of hybrid NID techniques that combine rule-based and anomaly-based detection to exploit the best of both approaches (Maseno et al., 2022). This manuscript stresses that a promising research direction towards these hybrid NID mechanisms may be the application of a set of recent algorithmic Inductive Biases (IB), mainly imported from the *Neural Algorithmic Reasoning* (NAR) blueprint (Veličković and Blundell, 2021) and from prototypical or metric-learning strategies for meta-learning (Snell et al., 2017).

This work focuses on IBs that combine the benefits of both neural networks and symbolic reasoning, our main argument being that the combination of these instruments may be an effective strategy that leads to data-efficient and interpretable NID models that learn to detect and *potentially categorize* previously unseen instances of attacks. Recent successes of algorithmic IBs in finding complex non-linear correlations over noisy and high-dimensional data (Derrow-Pinion et al., 2021), aligning to algorithms (Veličković et al., 2019), and giving hints to synthesize new rule-based reasoning (Pándy et al., 2022) back up and motivate our hypothesis.

***Main contribution.*** To the best of the authors' knowledge, the unsupervised categorization of Zero-day attacks (ZdA) considering the

* Corresponding author.
*E-mail address:* sabrina.sicari@uninsubria.it (S. Sicari).

https://doi.org/10.1016/j.cose.2024.103898
Received 13 November 2023; Received in revised form 9 April 2024; Accepted 9 May 2024
Available online 13 May 2024

degree of novelty has not been previously addressed by the ZdA detection research community. In this respect, this paper proposes a DL-based pipeline to tackle such a challenge, where multiple test ZdA classes are kept out of the training distribution and are given no labels. Moreover, focusing on the particular IoT scenario, our DL pipeline aims to be data-efficient, demonstrating the generalization of known attack detection and zero-shot ZdA categorization within constrained training data. Our proposed DL-based pipeline is named *NERO* because it uses *NEural algorithmic reasoning for zeRO-day attack detection in IoT environments*.

***Outline.*** This paper is organized as follows: Section 2 describes Neural Algorithmic Reasoning and the inherent algorithmic reasoning requirements of Zero-day attack detection. Section 3 gives a detailed description of our proposed framework, pointing to the potential advantages derived from its use. Section 4 offers experimental evidence about the effectiveness of the paradigm proposed by this work. Section 5 gives an overview of recent works related to symbolic inductive biases for Zero-day attack detection systems, pointing to the novelty of our proposal in this respect. Finally, Section 6 concludes the paper by pointing to the potential future research directions.

## 2. Background and motivation

### 2.1. Hybrid artificial intelligence

In Artificial Intelligence (AI), classic, symbolic, or rule-based AI is usually seen as a combination of classical programming, discrete optimization, and meta-modeling, leveraging compositionality, concept generalization, and transparency. On the other hand, sub-symbolic, perceptive, or statistical AI uses data and gradient-based optimization to extract opaque, efficient, and parametric transformations that approximate complex functions. Hybrid-AI (Raedt et al., 2020) stands for the combination of both symbolic and sub-symbolic approaches that combine the best of both worlds. Metric-based meta-learning is one applicative example of hybrid AI in which rule-based IBs like centroid computation and distance-based discrimination permit bias in the statistical learning towards the meta-knowledge hidden in data. Inspired by these IBs, the Neural Algorithmic Reasoning (NAR) blueprint (Veličković and Blundell, 2021) is used in this paper to imitate algorithmic behavior over high-dimensional manifolds, as explained in the following.

### 2.2. Neural algorithmic reasoning

The NAR blueprint helps apply algorithmic computations over information-rich environment observations (termed *natural inputs*). Such algorithmic computations have two main benefits: first, they can obtain fine-grain-precise outputs, providing applicability of algorithms to real-world scenarios, and second, algorithmically-aligned neural networks tend to generalize over out-of-distribution input dimensions.

The NAR pipeline is based on a *encode-process-decode* neural pipeline that is composed of three main ANNs:

1. An *Encoder network*, $\hat{f}$, that learns to encode high-dimensional *natural* inputs into high-dimensional *latent* representations. The encoders are devoted to learning to transform data into a high-dimensional latent space whose geometry is aligned with the abstract inputs an algorithm operates over. Encoders acquire feature importance, release representational pressure, and decouple representation learning from algorithmic reasoning learning.
2. A *Processor Network*, $P$, that receives in input the output of the encoder networks and operates over the corresponding latent space. This network is taught to imitate or *align* it is input–output mapping to the individual steps of a given algorithm $A$. Note the output representations of the processor network are still high-dimensional. Processor networks are often offline trained with synthetic input–output algorithm step-wise samples. These networks should be distribution agnostic in that they should imitate the algorithmic processing irrespective of input data distributions. For this reason, processor networks often use relational inductive biases such as message passing. In this sense, the processor network can be said to align with neuro-symbolic AI, while the encoder and decoder networks are more purely perceptive in their working principle.
3. A *Decoder network*, $\hat{g}$, whose parameters are optimized to map the high-dimensional latent representations of $P$ to human-readable outputs that align desired outcomes of algorithm $A$ over the natural inputs. The decoder (and encoder) networks are distribution-specific and are trained using a pre-trained frozen processor network. Decoder networks do not necessarily need to leverage the final outputs of an algorithm but can be used to back-propagate through single algorithm step computations, provided that step-wise supervisory signals are available.

### 2.3. Prototypical meta-learning

The term *prototypical learning* (PL) (Snell et al., 2017) is used to refer to the usage of DL for learning a representational space whose geometry is congruent with the semanticity of the use-case inferences in output. In Snell et al. (2017), a class *prototype* is defined as the *latent space centroid* of the (labeled) input samples associated with such a class. In its basic form, prototypical learning involves a DL model whose outputs are a function of the similarity distribution between the latent representations of other (unlabeled) input samples and the different class prototypes. Refinements of Snell et al. (2017) have been proposed to adapt to complexifications of the learning task, for example, optimizing the underlying metric using another neural module, refining the set of prototypes through self-attention, among others (Wang et al., 2020; Hospedales et al., 2021).

Aside from these refinements, the main benefit of prototypical learning is that it potentially incorporates algorithmic reasoning by design: focusing on the relation between inputs, PL is decoupled from class-specific distributions and thus enables learning a task in a higher abstract level, i.e., meta-learning. Such a task can then be evaluated in inference time with out-of-distribution inputs without degradation in the semantic significance of the corresponding outputs (Wang et al., 2020).

### 2.4. The privacy preservation requirement in IoT and low-training data regimes.

Deep learning models learn to make inferences from trends in the statistical distributions of training data. Generally, NID systems based on DL rely on discovering correlations between attack vectors and features of traffic traces and system logs. A DL-based NID system might need high amounts of data before converging to practical detection functionalities (Dixit and Silakari, 2021). In IoT, however, sensitive information such as end-user location, health status, social interactions, and industrial asset status may be the main content of communication. Consequently, training data for IoT NID systems may be related to confidential information, and even governmental regulations may impede gathering these data to train data-driven detection systems (Chanal and Kakkasageri, 2020).

In this respect, the models that use prototypical learning reach a potential decoupling between the abstract reasoning related to classification and the specific training data distributions (Snell et al., 2017). This decoupling also has the possible advantage of efficient inference within low-data regimes (Wang et al., 2020). Such a *Few-Shot Learning* ability of prototypical meta-learning is a crucial aspect of our proposal that reduces the need for training data to converge towards ZdA detection.

### 2.5. Zero-day attacks

New attack signatures are continuously being developed, and attackers are adding new variants of well-known attacks. When the cyber-treat intelligence community discovers a new intrusion vector after the attacks are perpetuated, the attack is qualified as a Zero-day attack, in that the defensive party has *zero-days* to act countermeasures or mitigation strategies. In this sense, ZdAs are new types of cyber threats that exploit vulnerabilities in software or hardware that have not yet been patched or fixed by the vendors. Recent examples of ZdAs are the Mirai botnet (Affinito et al., 2023) and the Memcached DDoS.[1] According to a recent survey (Al-Zewairi et al., 2020), considering a two-level taxonomy of attacks where macro-categories of attacks contain other micro-categories, one can define two types of ZdAs: **type_A ZdAs** are attacks that might need the definition of a new macro-class of attack. **Type_B ZdAs**, instead, are those attacks whose features might be similar to those of a known macro-category to some extent, but variations in its signature may suggest these samples belong to a new micro-category for this attack.

### 2.6. Motivation

In ML-based cyber-attack detection scenarios, the open-world assumption asserts that the dataset used to train the model is inherently limited regarding the representativity of all the existent attacks that might be encountered during inference time. In this respect, Type_B ZdAs correspond to any evaluation sample whose macro-class was included in the training data but whose micro-class was not. Instead, the macro and micro classes of Type_A ZdAs are missing in the training data by definition.

From a probabilistic perspective, both Type_A and Type_B ZdAs may represent data sampled from different probability distributions with respect to the one that generated the training data. According to the previous observation, the ZdA detection problem can be framed as a meta-learning problem, where the base-learning level task is related to closed-set classification. Instead, the meta-learning level task is associated with the optimal definition of the set of classes upon which such a closed-set classification is performed. Thus, the supervisory signal for the meta-learning task is related to the ability of the meta-learner to discover new attack instances.

However, the generalization of algorithmic reasoning is a historically challenging requirement for pure data-driven AI instruments (Liu et al., 2021b). One example of failure in algorithmic reasoning learning for traditional ML algorithms is that encountered in the case of distribution shifts: If the statistical behavior of data in inference time diverges from the training one, the statistical assumption of querying a model with Independent and Identically Distributed observations – which is the working principle of traditional ML – is broken. This failure mode is also called an out-of-distribution (OOD) generalization failure (Liu et al., 2021b). Another exemplary failure occurs when the model converges to identifying the wrong features as discriminative of the input–output mapping.

In this respect, the NAR paradigm offers explicit guidelines for building a DL pipeline that follows an abstract reasoning-based supervisory signal. Our main hypothesis is that the meta-level reasoning might be a key enabler for the ZdA detection and categorization process, as described in Section 2.5. In other words, the abstract reasoning that rules the design of NERO may enable one to differentiate between known and unknown environment observations and to provide specialized inferences about the degree of abnormality of the latter within a low-training data regime.

## 3. Our proposed framework for zero-day attack detection and categorization in the IoT

This section describes NERO in its architectural components and operating principles. As a NAR pipeline, NERO has three main learning goals. First, the processor layers' computations should learn a distribution-agnostic algorithmic task. Second, the encoder network must learn to project the instances of natural observations to abstract representations over which the processor network performs such a task. Lastly, the decoder network must learn to extract low-dimensional indicators from the outputs of the processor network. Fig. 1 contains a schematic representation of our proposed scheme.

### 3.1. Encoder network

In any NAR pipeline, the encoder network is trained to produce high-dimensional encodings of the information sources. This network learns to transfer the *natural inputs* to a set of abstract latent representations. The processor network subsequently ingests such encodings to perform algorithmic computations in the hidden space.

In NERO, the natural inputs are assumed to be formed by the observed network traffic features. If we denote the input space with $\mathbf{I}$, the mapping from such input space to the latent space of the processor network can be denoted as:

$$\text{Enc}^{\text{traffic}} : i \rightarrow z_i \in \mathbb{R}^d, \forall i \in \mathbf{I}^{\text{traffic}}$$

The encoder networks can be implemented using any architecture that accommodates the shape of the input space observations. In our experiment, a Multi-Layer Perceptron (MLP) is used for the encoder. The main reason for such an architectural preference over Convolutional Networks is the inherent low dimensionality of input data.[2] Although convolutional filters represent a parameter-sharing mechanism, this reduction of parameters may harm low-dimensional data with skewed importance attributes, as noted in Nguyen and Le (2023). Although some literature has used recurrent modules for NID (Imrana et al., 2021), our encoder should not assume temporal correlations between successive flow-related entries. This is because the scope of this research is the classification of flows *per se* without inferring temporal correlations between the class of multiple flows. To this respect, it is worth mentioning that our encoder is a non-linear projection of the input space, with the unique goal of releasing representational pressure for the processor. The latter module exploits the inter-record relational attention, as explained in the following.

### 3.2. Processor network

The main goal of the NERO's processor network will be to manipulate the hidden representations following an algorithmic bias to produce other representations that align with the desired outcomes of the ZdA detection task. Recall that such outcomes involve inference about **(1)** known classes of traffic (either benign or malign) or unknown types of traffic, in the form of potential **(2)** type_B and **(3)** type_A zero-day Attacks. In this respect, our processor uses two hidden layers. The first layer helps differentiate between known and unknown micro-categories as defined in Section 2.5. The second layer, instead, operates on the representations of the first processor and helps to distill the association of unknown observations to classes **(2)** and **(3)**.

[1] https://www.zdnet.com/article/memcached-ddos-the-biggest-baddest-denial-of-service-attacker-yet/.

[2] See Amiri et al. (2023) for a study on the different neural architectures and their potential strengths and limitations in terms of pattern recognition on different types of data.

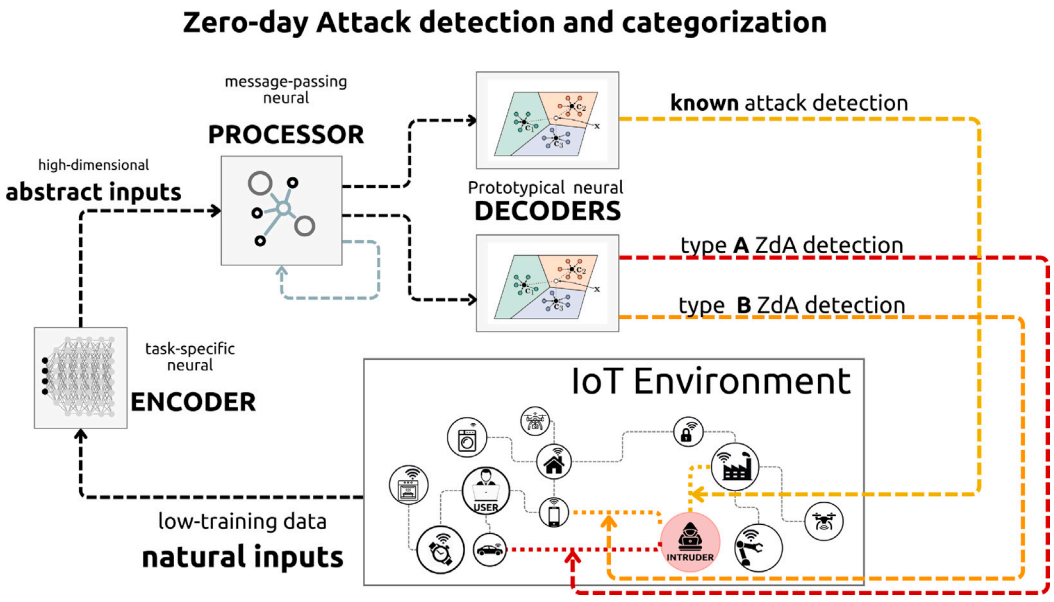


**Fig. 1.** Schematic representation of the Neural Algorithmic Reasoning blueprint applied to the task of Zero-day attack detection. The pipeline is trained iteratively with a set of specialized layer-wise supervisory signals. Two decoder modules differentiate the learning goals in our hierarchical classification task.

Our proposal is to map the abstract reasoning behind ZdA detection on a set of specialized prototypical learning operations. Our processing modules adopt geometric inductive biases to push the neural computations explicitly towards distribution-agnostic classification. In other words, the processor network uses metric-based meta-learning to align with the algorithmic rationale of ZdA detection.

Following the NAR paradigm, NERO performs intrusion detection as if it were a graph representation learning task. Thus, our pipeline encodes the input observations on a graph. Inside this design strategy, our pipeline is taught to produce node representations that can be decoded as a set of desired outputs, i.e., the artifacts **(1)**-to-**(3)** mentioned before.

The graph representing the input space will be denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ is the set of vertices or nodes and $\mathcal{E} = \{e_{i,j}\}$, $\forall i, j \in \mathcal{V}$ is the set of edges. The nodes of the graph will correspond to the environment observations, while the edges will encode the semantic relationships between nodes. The *neighborhood* of $v$, usually denoted with $\mathcal{N}(v)$, corresponds to the set of nodes that are connected to $v$ by the edges of the graph. Our graph includes node-level features or attributes. These features are represented using a real-valued matrix $\mathbf{X} \in \mathbb{R}^{|\mathcal{V}| \times m}$ where $m$ is the dimension of the node-level feature space. The feature vector of a node $v_i \in \mathcal{V}$ coincides with a row of $\mathbf{X}$ and will be denoted as $x_i$.

An edge between the nodes $v_i$ and $v_j$ will instead be denoted by $e_{i,j} \in \mathcal{E}$. Note that the whole set of edges can be represented by an adjacency matrix $\mathbf{A}^{|\mathcal{V}| \times |\mathcal{V}|}$. The processor neural network of NERO will receive a partial graph where only a subset of observations will be labeled and thus connected between them through graph edges. The inference of potential links between the unlabeled observations will be the main task learned by the processor network.

In our experiments, the main architectural choice for a processor network in the NAR paradigm was followed: In fact, these components were implemented using Graph Attention Networks with dynamic attention, also termed GATV2, Brody et al. (2021), which are used to learn dynamic refinements of the input adjacency matrix for the message passing among neighboring nodes. The NAR blueprint proposes optimizing each layer's parameters of such a GNN using a specific supervisory signal within each layer. Such a supervisory signal follows a task-specific underlying algorithmic reasoning. In NERO, the reasoning process must effectively conduce towards ZdA detection and categorization and will be explained in Section 3.4.

From a practical viewpoint, there are two important components in the NERO processor network:

- **Episodic meta-learning.** In episodic training, the input batches have two types of samples: **support samples** are input observations accompanied by their labels. In contrast, **query samples** are unlabeled, and the model has to infer such a label. In NERO, the processor network will implicitly infer the label by learning to approximate the semantic adjacency matrix **A**. Importantly, to teach the pipeline to learn an abstract rationale over distribution-specific inputs, meta-training implies sampling data from a variety of different classes and packing these data in different training batches. In our experiments, such a *task sampling* procedure helped to avoid overfitting our representational machinery to training data distributions.
- **Manifold learning.** A key component that helps the convergence of the processor network is regularizing the latent space with manifold learning (Lei et al., 2020). Our processor network is encouraged to pull apart the hidden representations of nodes appertaining to different classes and to put the representations of same-class observations closer. Specifically, we explicitly minimize two cross-entropies (CE): first, the CE between the distributions of the baseline and predicted adjacencies, and second, the CE between the respective complementary distributions. The former CE term acts as an *attractive force* that pushes together latent space representations of the same class, while the second acts as a *repulsive force* analogously. The resultant effect is a clustering-friendly latent space, which, as shown in Section 4, helps to generalize prototypical classification out-of-distribution.

Our processor uses support labels at each forward pass to initialize the adjacency matrix. By default, each connection from a support to a query sample and between two query samples is initialized to 1. The attention mechanisms in the layers of the NERO processor, which are instances of the GATV2 architecture, then learn to refine such connections. For more information on the GATV2 layer architecture, the interested reader is referred to the original paper (Brody et al., 2021).

### 3.3. Decoder networks

The outcomes of the NERO pipeline will be obtained from the decoder networks. This paper proposes implementing two distinct decoder networks, one for each processor layer. The first decoder will be fed by the representations provided by the first processor layer, and the second decoder will ingest the representations of the processor's second layer. Each decoder will have two streams related to closed-set and open-set classification, respectively. The first decoder will help assess whether an input is potentially related to known or unknown types of traffic. The second decoder, instead, will classify its inputs, associating them to the potentially corresponding classes.

Prototypical decoding is at the core of the decoding process in NERO. In other words, the support samples will be used at each training iteration step to compute the centroids of class-specific clusters in the latent space. More specifically, the decoding will be performed as follows. First, a set of encoded inputs $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_{|\mathscr{B}|}$ from a batch $\mathscr{B}$ is given to the decoder network. Second, the decoder takes these embeddings and computes the centroids in the latent space:

$$\mathbf{c}_i = \frac{1}{N_i} \sum_{j=1}^{N_i} \mathbf{z}_j, \ \forall \mathbf{c}_i \ \in \mathscr{C} \tag{1}$$

where $N_i$ is the number of data points in cluster $i$.

Third, a similarity score $s_{ij}$ is computed between each query input $z_j \in |\mathscr{B}|$ and the latent centroid vectors $\mathbf{c}_i \ \in \mathscr{C}$. The open-set decoder neural networks judge if any significant similarity score indicates the query sample is related to a known class. Instead, the closed-set decoders associate samples to the corresponding nearest centroid in the latent space.

When a new attack is *discovered*, namely, in the form of correctly predicted type_A or type_B ZdA, new knowledge can be incorporated into the training data by simply including samples of such a new class in the support set of the following input batches. Closed-set decoding is learned by minimizing the multi-class cross-entropy loss on the query predictions whose label is among the set of known classes. Open-set classification is optimized instead by reducing the known/unknown binary-cross-entropy loss. The following section details how each prototypical step aligns with the steps of a ZdA classification and categorization algorithm by design.

### 3.4. The algorithmic reasoning behind zero-day attack detection

Our NAR-based pipeline detects and alerts about potential alignments of the environment observations to known attack fingerprints and to what potentially represents novel classes of attacks. To meet this goal, high-level supervisory signals must be defined for training. In NERO, such signals coincide with explicit evidence of examples of classes **(1)**-to-**(3)**, as defined in 3.2, among the training data. A detailed explanation of the proposed labeling strategy follows.

Our proposal is to align the layers of the processor network to a *hierarchical inference* schematized in algorithm 1. The following is one possible design specification of the step-wise tasks in our specific ZdA detection problem:

1. Using the open-set and closed-set decoders as defined in 3.3, the representations provided by the first layer of the processor network will be optimized to classify an observation $o$ between a set of $K^{micro} + 1$ classes, where $K^{micro}$ corresponds to the sum of the total number of known micro-classes. The added class slot corresponds to the *unknown_1* class in lines 1–5 of algorithm 1.
2. The representations of the second processor layer will instead be optimized through a classification task between $K^{macro} + 1$ classes. This classification task will provide the basis to associate the observations previously marked as *unknown_1* to one of the $K^{macro}$ known macro-attacks or the *unknown_2* class. (lines 6–12 of algorithm 1).

**Algorithm 1** Algorithm for Zero-day Attack detection

**Require:** A set of environment observations $O$
1: **for** each environment observation $o \in O$ **do**
2: Try to associate $o$ to one of the following: a known micro-class or the *unknown_1* class.
3: **if** $o$ is associated to a known micro-class **then**
4: **continue**
5: **else**
6: Try to associate $o$ to a known macro-class, or *unknown_2* class.
7: **if** $o$ is associated with a known macro class **then**
8: associate $o$ to a a *type_B* ZdA.
9: **continue**
10: **else**
11: associate $o$ to a a *type_A* ZdA.
12: **end if**
13: **end if**
14: **end for**

Besides known-class classification, the main assumption made for the effectiveness of ZdA detection in NERO is then the following: If an observation $o$ is classified as *unknown_1* at the first stage, and associated with an instance of a known macro-class at the second stage, then it plausibly means that $o$ is a *type_B* ZdA. Instead, if $o$ is associated with the *unknown_2* class at stage 2, then it potentially represents a type_A ZdA.

It is worth stressing that NERO minimizes multiple loss functions, each tailored to step-wise algorithmic goals. Moreover, to prove generalization over ZdA detection, evaluating our pipeline should involve assessing the classification of *new classes* of type_A and type_B ZdAs concerning those observed during the training. These two observations lead to a final question. How to train the proposed DL pipeline to (1) learn a hierarchical closed and open-set classification task and (2) generalize open-set classification out-of-distribution. In this respect, the following section details our training and evaluation framework.

### 3.5. Training and validation framework for our zero-day attack detection pipeline

To provide supervisory signals for training the processor network, a closed-set labeled dataset for NID is manipulated as described below. First of all, the train-validation-split should respect two main conditions:

1. The set of all data points corresponding to some pre-selected set of type_A ZdAs must be kept only in the validation dataset.
2. The points related to a chosen subset of type_B ZdAs must also be kept only in the validation dataset.

Without losing generality in ZdA detection, one can include data from all the known classes in the training and evaluation datasets. However, having done any train-validation-split that respects the two conditions stated above, two labeling schemes for each dataset must be created. A detailed explanation of constructing these two versions from the train and validation datasets follows. In the following, the training and validation datasets are indistinctly referred to as *datasets*.

- Each version of these datasets has the same input data points, but the variation between the first and second versions regards the labels associated with such data points.
- In the first version of each dataset, the labels regarding a subset of micro-classes are labeled as *unknown_1*. The micro-classes whose labels are manipulated must belong to at least two different macro-attack classes. Moreover, to resemble the situation in which a complete macro-class is unknown, i.e., to model the

existence of type_A ZdAs, the set of manipulated labels must *cover* at least one entire macro-class.

- Macro classes are the labels of each data point in the second version of the dataset. Importantly, the elements previously labeled as *unknown_1* and belonging to a known macro-class will now be labeled with the corresponding macro-class label. In contrast, the rest of the *unknown_1* records will be instead labeled as *unknown_2*. This dataset constitutes a supervisory signal for learning to differentiate among type_A and type_B ZdAs.

By manipulating the labels of the datasets in such a way, a step-wise supervisory signal for the processor network is created. Specifically, the first version of the dataset encodes a supervisory signal for optimizing/evaluating the classification task of the first NAR phase (both open and closed-set). Instead, the second version of the dataset guides the learning and evaluation of the second processor layer and its corresponding decoders. The following section provides a case-study evaluation based on synthetic and real data that offers evidence of the potential effectiveness of our proposal.

## 4. Experimental case study

Evidence on the effectiveness of the NAR blueprint for ZdA detection in IoT is given in this section as a realistic use case. Our belief is that these preliminary results on ZdA generalization may encourage and shed light on the construction of a delivery-ready instrument.

### 4.1. Implementation of the NERO pipeline

All our code is implemented in Python 3.10.10, and our neural modules are implemented using the PyTorch deep learning library, version 2.0.1. The NERO neural modules were trained on an NVIDIA A100-SXM4 GPU with 80 GB of RAM. The working principles and training data for our encoder, processor, and decoder networks are those explained in Section 3.

***Encoder***. Our encoder consists of two sequential linear layers preceded by a Batch Normalization and with a Dropout intermediate layer. After each linear transformation, a ReLU activation function is applied. The first layer expands the input dimension to half the dimension of the latent space.

***Processor***. Our processor consists of a GATV2 Layer (Brody et al., 2021) with different learnable weight matrices for the linear transformations of senders and receiver nodes, respectively. The activation function is the Leaky ReLU with a slope of 0.2. A Dropout Layer is applied to the attention matrix before the final message passing operation. A second Leaky ReLU with the same slope is applied to the output of the GATV2 Layer.

***Decoder***. Our closed-set decoder is implemented as specified in Section 3.3. The open-set decoder is implemented through a linear transformation matrix with an output dimension of one followed by a Sigmoid activation function.

Additional parameters and hyperparameters used for our experiments are detailed in Table 1.

In our implementation, samples for four distinct macro-classes (and, consequently, micro-classes) are taken at each batch. One of the four sampled classes corresponds to a type_A ZdA and the other to a type_B ZdA. By doing so, our pipeline can be assessed in the differentiation capability between the two types of unknown attacks, avoiding a macro-class collapse in the last phase. In future works, a more flexible decoding architecture should permit inference over batches that contain class collapses.

Concerning the open-set decoding, the ZdA samples were given more weight in the loss computation to deal with the inherent imbalance of known/unknown input samples. Additionally, the closed-set gradient computation was restricted to the query samples of known classes. By doing so, our encoder and processor are prevented from *pushing* the latent representations of query samples that correspond to unknown attacks to known class signatures, which would be incorrect. Finally, the gradients related to the open-set loss were detached from the rest of the pipeline to avoid overfitting the latent representations to the ZdAs in the training data.

**Table 1**
Hyperparameter specifications.

| Hyperparameter | Specification |
|---|---|
| Learning rate (for all modules) | 0.001 |
| Support samples per class (K-SHOT) | 15 |
| Query samples per class | 5 |
| Classes per batch (N-WAY) | 4 |
| Total macro classes | 10 |
| Total micro classes | 40 |
| Dropout rate (processor and encoders) | 0.1 |
| Processors hidden space dimension | 1024 |
| Processor attention-heads | 8 |
| Batch size | 80 |
| Training batches | 500 |
| Test batches | 50 |
| Few-shot sampler worker threads | 4 |
| Loss function (Closed set classification) | Multiclass cross-entropy |
| Loss function (Open set classification) | Binary Cross-Entropy (BCE) |
| Positive weight for phase-1 BCE | 1 |
| Positive weight for phase-2 BCE | 3 |
| Optimizer | Adam (without weight decay) |
| Max. number of attacks per class in the whole dataset | 10 000 |
| Train-evaluation split (for known attacks) | 80% 20% |
| Training epochs | 80 |

### 4.2. Dataset details

The Edge-IIoT dataset in Ferrag et al. (2022) is used in this work to assess the ZdA detection accuracy of our methodology. Being released in 2022, this is the most recent IoT-focused NID dataset. Concerning other IoT-oriented datasets for NID, the Edge-IIoTset aims to include traffic from both IoT and I-IoT. Another advantage of this dataset is the labeled two-level taxonomy of attacks, which permits assessing type_A and type_B Zda detection. Fourteen types of attacks are categorized into five macro-categories: DoS/DDoS attacks, Information Gathering, Man-In-The-Middle attacks, Injection attacks, and Malware attacks.

The preprocessed Edge-to-IIoTset dataset in Ferrag et al. (2022) contains sixty-one features and is publicly accessible in the IEEE dataport.[3] Having downloaded this dataset, our preprocessing follows further author's guidelines available in Kaggle[4] and involves taking out various features like IP addresses and ports that could be erroneously identified as discriminant. The final data contains forty-two numerical and seven categorical features. The latter were encoded using integers and further treated as numerical.

Our training data is limited to $4 \times 10^4$ samples for the most-represented classes to resemble low-training data regimes. In the original dataset, the Man-In-The-Middle class was not divided into micro-clusters. Consequently, *HDBScan* clustering[5] was performed on the feature space to divide such a class into four synthetic microattack classes. This class was particularly underrepresented in the original dataset. After removing the scattered points according to the soft-clustering, we obtained a few samples to test the type_A ZdA detection.

[3] https://ieee-dataport.org/documents/edge-iiotset-new-comprehensive-realistic-cyber-security-dataset-iot-and-iiot-applications.

[4] https://www.kaggle.com/code/mohamedamineferrag/edge-iiotset-preprocessing.

[5] https://hdbscan.readthedocs.io/en/latest/index.html.

**Table 2**
Two-level Taxonomy of attack in the pre-processed Edge-to-IIoT Dataset in Ferrag et al. (2022). According to our proposed labeling strategy, two macro-classes are labeled as type_A ZdAs, and multiple micro-classes are labeled as type_B ZdAs. Importantly, these classes may differ between the train and test split to demonstrate the out-of-distribution generalization of ZdA detection.

| Level 1 | Level 2 | ZdA | Type | Split | Training samples (per class) |
|---|---|---|---|---|---|
| DDoS | DDoS_HTTP | Y | B | Test | $4 \times 10^4$ |
| | DDoS_ICMP | Y | B | Train | $4 \times 10^4$ |
| | DDoS_TCP | ZdA3 | Known | Both | $4 \times 10^4$ |
| | DDoS_UDP | ZdA3 | Known | Both | $1.2 \times 10^4$ |
| Information Gathering | Fingerprinting | Y | A | Train | $0.08 \times 10^4$ |
| | Port-scanning | Y | A | Train | $2 \times 10^4$ |
| | Vulnerability-scanning | Y | A | Train | $4 \times 10^4$ |
| Injection | SQL injection | Y | B | Test | $4 \times 10^4$ |
| | Uploading | Y | B | Train | $3.6 \times 10^4$ |
| | Cross-site scripting | Both | Known | Split2 | $1.5 \times 10^4$ |
| Man-in-the-middle | MITM-0 | Y | A | Test | 35 |
| | MITM-1 | Y | A | Test | 123 |
| | MITM-2 | Y | A | Test | 44 |
| | MITM-3 | Y | A | Test | 35 |
| Malware | Backdoor | Y | B | Train | $2.4 \times 10^4$ |
| | Password | Y | B | Test | $4 \times 10^4$ |
| | Ransomware | Both | Known | Split2 | $0.9 \times 10^4$ |
| Normal | Normal | Both | Known | Split1 | $1.3 \times 10^4$ |

The Normal class is considered a macro-attack containing a unique micro-attack for our experiments. Table 2 shows our training and testing data details. To facilitate data exploration, the preprocessing code has been open-sourced as a .ipynb notebook in a public GitHub Gist.[6]

### 4.3. Baselines

The following experiments compare our pipeline to another state-of-the-art ZdA detection pipeline in Bovenzi et al. (2020), a hierarchical approach for disentangling known attack detection from ZdA detection. The first phase of the pipeline in Bovenzi et al. (2020) performs a binary classification in which normal traffic is separated from potential attacks. The second phase is a thresholding filter that signals potential ZdAs when the classification confidence of attacks is low. The baselines assume a perfect anomaly detector in the first phase, separating normal vs. attack-related traffic with 100% accuracy.

Four variants for the baseline were tested: **(1)** Fixing the confidence threshold $\tau_u$ to 0.75. **(2)** Fixing the confidence threshold $\tau_u$ to 0.95. **(3)** Online learning of the threshold and the known-attack classifier. **(4)** Offline learning the threshold with a pre-trained classifier. The following experiments implement two versions of each baseline to compare each time with NERO. One is used for ZdA detection over micro-attacks, and the other for macro-attacks.

Following the authors' guidelines in Bovenzi et al. (2020), the implementation of the anomaly detection module was equipped with the same Dropout ratio as in our modules to preserve fairness in the comparison. Two linear layers with a ReLU activation compose the MLP, and we found the best-hidden layer dimension was the same as the NERO pipeline. The multiclass classifier is naturally equipped with a softmax operation after the output layer (the Sigmoid activation was also tested, being the anomaly threshold unaware of relative entropy among outputs, but the best results were found with the Softmax). The chosen values for the threshold parameters were representative of the authors' experiments. It is also worth noting that avoiding implementing the first phase does not prejudice the baseline, as the assumption is to have a perfect anomaly detector in this baseline.

[6] https://gist.github.com/QwertyJacob/ab87264bc5e6fba9bbd17bd0d7faa168#file-iiotset_graphviz-ipynb.

### 4.4. Metrics

The performance of the pipeline was measured using the following metrics:

1. For the closed-set classification tasks, the accuracy measure defined in Eq. (2) is used, where the number of *total predictions* corresponds to the known attacks in each batch/epoch:

$$\text{Acc} = \frac{\text{Correct predictions}}{\text{Total predictions}} \tag{2}$$

2. For the open-set classification tasks, instead, the *balanced accuracy* is used as defined in Eq. (3):

$$\hat{\text{Acc}} = \frac{\text{TNP} + \text{TPP}}{2} \tag{3}$$

where TNP is the true negative *proportion* and is defined as the ratio of predicted negatives and the total number of negatives in the episode:

$$\text{TNP} = \frac{\text{Predicted Negatives}}{\text{Total Negatives}} \tag{4}$$

Conversely, TPP is the true positive proportion and is defined analogously:

$$\text{TPP} = \frac{\text{Predicted Positives}}{\text{Total Positives}} \tag{5}$$

The reason for using (3) is that ZdAs – our positive samples in the open-set classification – represent an unbalanced class, and using (2) would result in a biased comparison: a deterministic negative predictor would achieve better performance than any other model, compressed a random baseline.

***Training and testing configuration.*** The training epochs were made of 500 batches, while the evaluation epochs contained 50 batches. A 4-way 5-shot classification is performed at each batch: four micro-classes are sampled from different macro-classes. One of such four classes is a type_B ZdA, and one is a type_A ZdA; the other two are known attacks. Five support samples and fifteen query samples are given for each sampled class, and NERO classifies the query samples in the open and closed set task of each phase.

### 4.5. Results

Fig. 2 contains the closed-set classification task evaluation results. The mean accuracy over one hundred evaluation batches was computed

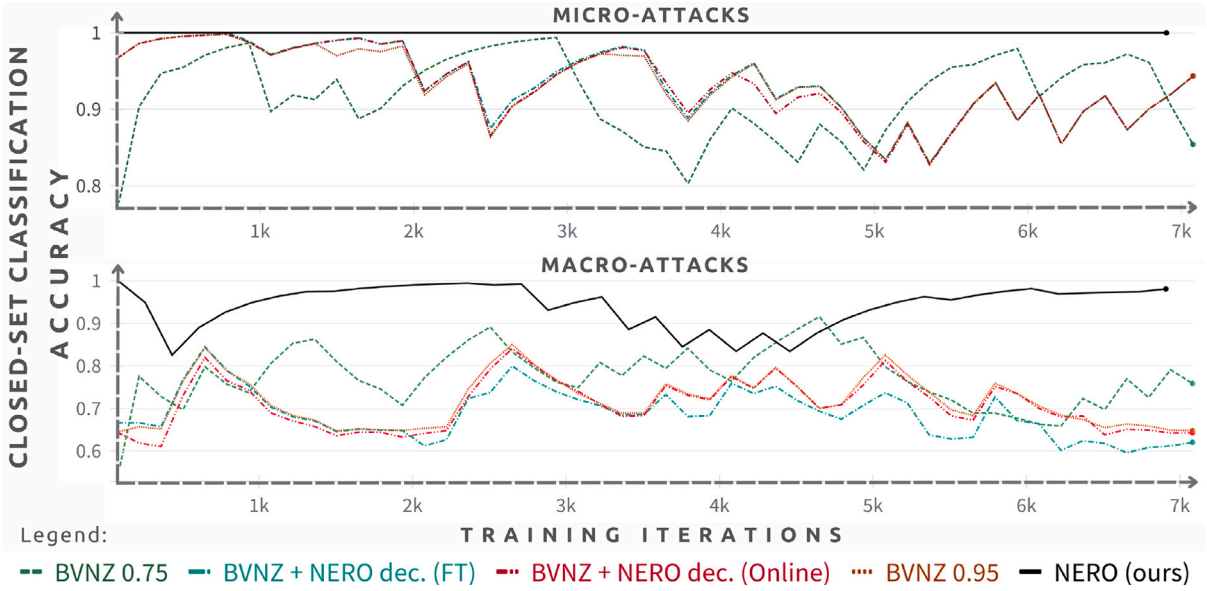


**Fig. 2.** Closed-set classification evaluation of our algorithm (NERO) and multiple baselines derived from the work in Bovenzi et al. (2020). Our algorithm is more data efficient and generalizes well in distribution concerning the attacks in the Edge-to-IIotSet in Ferrag et al. (2022).

for all the baselines and NERO. The accuracy in each batch was computed using (2). Finally, the exponential moving average with a smoothing factor of 0.5 was computed and reported in the line plots of Fig. 2. Such a figure shows a neat superiority of our method concerning the rest of the baselines in the micro-attack (upper plot) and macro-attack (lower plot) domain. Even if the baselines struggle to classify attacks in the evaluation data set, their accuracy converges in the training set from the first epochs. The accuracies with respect to the training distributions are omitted for brevity.

As seen from Fig. 2, the micro-attack closed-set performance of NERO is at 100% accuracy from the initial evaluation epochs. Such a result does not contrast with recent literature focused on supervised NID, nor is it difficult to understand the effectiveness of prototypical learning when the latent space is regularized. To give a clear picture of this effect, another experiment using synthetic data was made, in which the hidden representations of the processor's first layer were linearly reduced to two dimensions using principal component analysis (PCA). A scatter plot of these decompositions at the first training iterations is shown in Fig. 3. A clear prototypical polarization is observed in such a figure within a few training iterations.

Fig. 4 presents instead the results of the open-set classification task, which is a binary classification task. In the first phase (upper plot), the traffic is differentiated between known and unknown attacks, while in the second phase (lower plot), the unknown traffic is differentiated between type_A and type_B ZdAs. The plots contain an analogous averaging as those of Fig. 2, but using the balanced accuracy in (3). As can be seen from the figure, the NERO pipeline improves the performance in both ZdA detection and type_A and type_B differentiation concerning the baselines. From the figure, one can also notice that a great part of the improvement in the A/B ZdA classification task is achieved by using our open-set decoding strategy, which is a learnable linear projection with a Sigmoid activation rather than a fixed threshold as proposed in Bovenzi et al. (2020). Additional improvement apart from our decoding strategy may be the merit of combining the NAR paradigm and prototypical learning, as explained in the following.

The open-set classification tasks are challenging because new classes of ZdAs are given at test time, and the absolute position of the corresponding representations might be at any region of the latent space. Moreover, prototypical classification is impossible with such attacks because no support labels are given for new attacks. Hence, the NERO pipeline must recognize that these representations belong to unidentified clusters. For making this inference, the rationale followed by the open set decoder should consist of two steps: first, associating new attacks to a unique class, and second, realizing that such a class is not any of the support classes by looking at the similarity scores of the unknown points and the prototypes of known classes. To visually confirm that the representations ingested by the open-set decoders are congruent with this rationale, another experiment was done with synthetic data, keeping track of these representations. In Fig. 5, a two-dimensional linear PCA decomposition of the inputs to the open-set decoder in phase 1 is plotted at different training steps. One can easily observe how the unknown classes, corresponding to type_A ZdAs (black dots) and type_B ZdAs (gray dots), are pushed to a unique region despite the particular ZdA classes being potentially different at each batch.

## 5. Related works

### 5.1. Algorithmic-oriented inductive biases in deep learning

Neural Algorithmic Reasoning is a recently proposed paradigm. However, empirical evidence of the effectiveness of such a paradigm in other network-related scenarios exists. A successful and direct application of the NAR blueprint is offered in Deac et al. (2020), where Deac et al. showed improved data efficiency with respect to state-of-the-art deep reinforcement learning models in the ATARI benchmark. Another NAR pipeline is offered in Beurer-Kellner et al. (2022), where authors teach an ANN to output candidate values for missing parameters on network configuration specifications. Beurer-Kellner L. et al. used a graph modelization of network configuration instances specified as lists of facts where some parameters were masked. Then, the problem of masked parameter completion was modeled as a graph-representation learning task.

On the side of standardization, the work in Veličković et al. (2022) proposed a benchmark dataset to assess the algorithmic reasoning capabilities of neural networks. A successive work in Ibarz et al. (2022), identified many inductive biases that lead to the convergence of a unique GNN-based processor network that can solve many potentially different dynamic programs. Instead, many other neural algorithmic reasoning inductive biases are evidenced in Cappart et al. (2023), where the focus goes beyond dynamic programming and concentrates on the ability of GNNs to solve combinatorial problems.

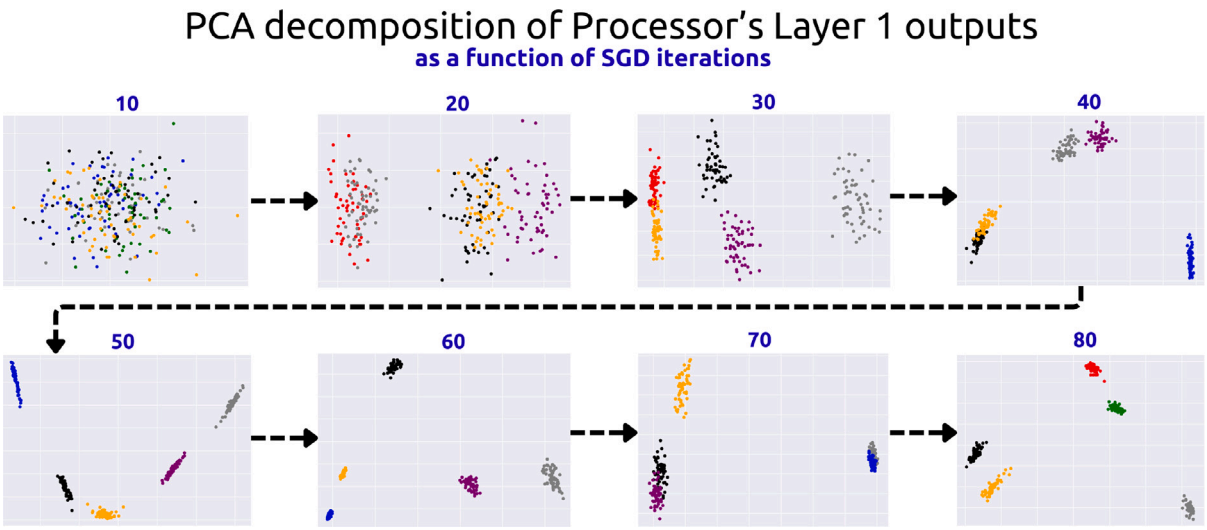


**Fig. 3.** Two-dimensional Principal Component Analysis decomposition of the hidden representations of the first layer of the NERO processor network. Different colors indicate the sampling of different classes at each batch. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

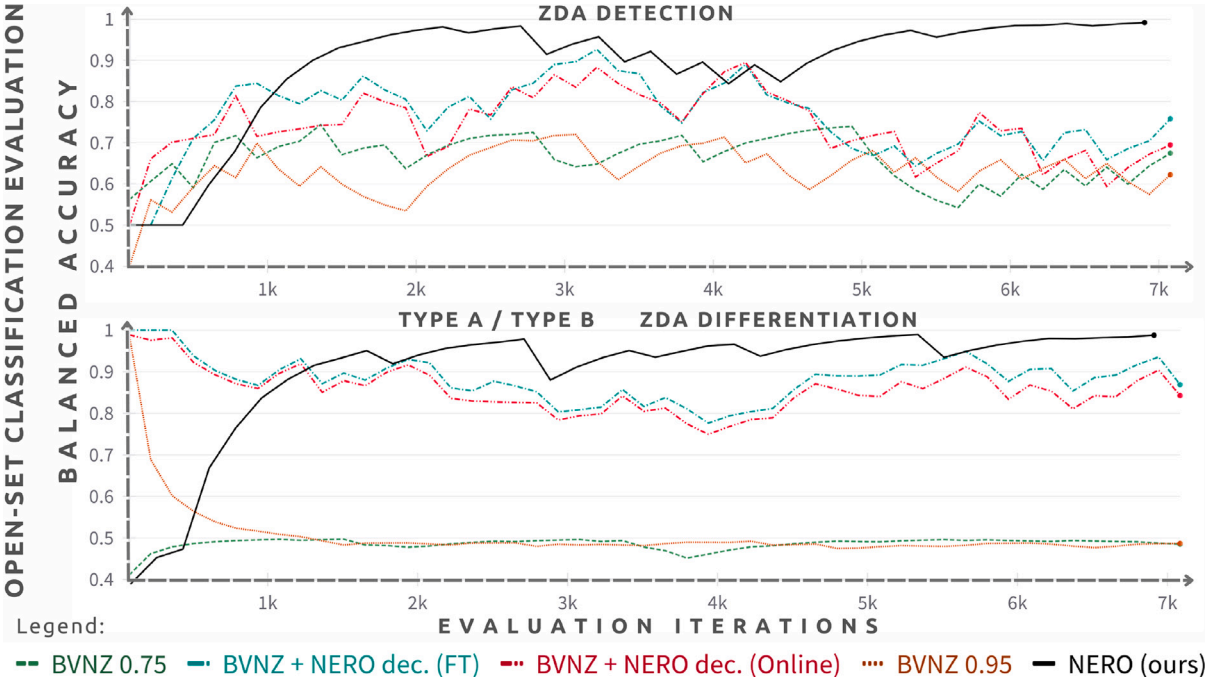


**Fig. 4.** Results of the open-set classification evaluation for all the baselines and NERO with the proposed dataset.

### 5.2. Deep learning for network intrusion detection

Supervised (Tsimenidis et al., 2022), and unsupervised (Yang et al., 2022) DL has been used for NID purposes for decades. Convolutional Neural Networks (Ashiku and Dagli, 2021), Recurrent Modules (Gamage and Samarabandu, 2020), and Graph Neural networks (Dong et al., 2023) are used in NID tmulticlassand perform inferences over network-related data. Detection efficiency (Heidari et al., 2023b), data re-balancing strategies (Liu et al., 2021a), and adversarial training techniques (He et al., 2023), among others, have been modeled alongside NID accuracy with the help of Deep Learning in the past years. However, being a statistical-driven paradigm, DL and ML generally strive to generalize classification out of the training-data distribution (Liu et al., 2021b). For this reason, ZdA detection might not be as easy as closed-set multiclass classification in DL scenarios and could require a meta-modeling strategy to learn a proto-distribution that generates the attack distributions. The works reviewed in the following offer some candidate strategies in this sense.

### 5.3. Zero-day attack detection

The authors of Zhang et al. (2020) relied on zero-shot learning to achieve zero-day attack detection. They relied on descriptions of seen and unseen classes and a multi-view learning paradigm in which an autoencoder is trained to translate between the feature and semantic domains. More specifically, the authors of this work used semantic prior knowledge of attacks whose samples were not included in the training dataset to compare its feature-domain translation with the testing data at inference time. Our method instead avoids the assumption of having second-domain information to perform inferences about unknown attacks.

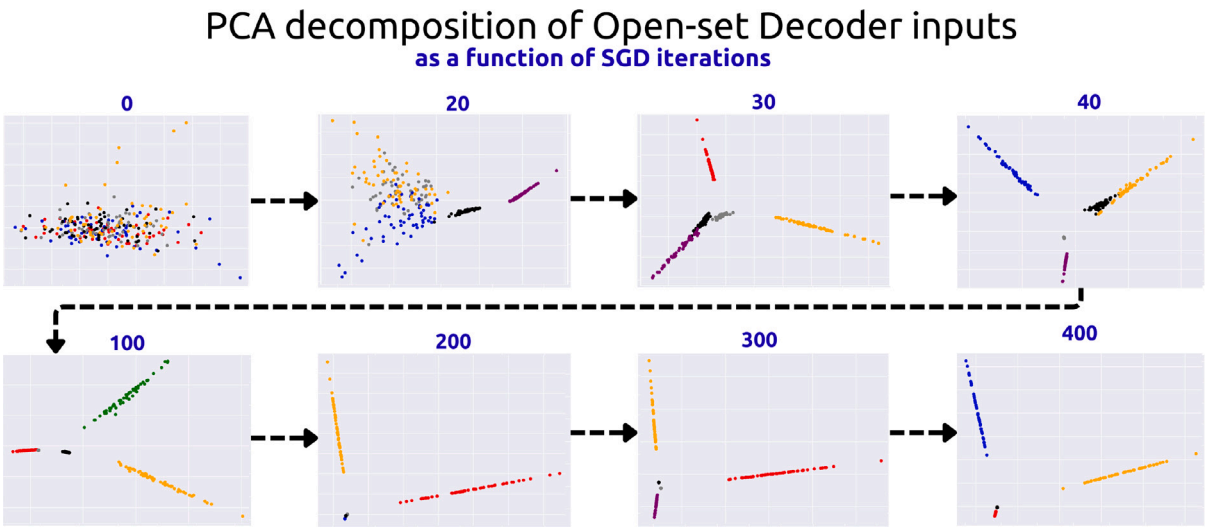


**Fig. 5.** PCA decomposition of the hidden representations ingested by the open set phase-1 decoder. Despite type_A ZdAs (black dots) and type_B ZdAs (gray dots) potentially appertain to different classes at each batch, focusing on similarities to known-class prototypes permits to map them to the same region of this latent space.

The authors in Bovenzi et al. (2020) introduced a hierarchical model in which two anomaly-detection modules are considered. The first is a lightweight anomaly detection component that permits the benign traffic to bypass the second module, which operates open-set intrusion classification (thus, related to zero-day attack detection). Bovenzi et al. designed two threshold parameters to control both the anomaly criterion and the minimum confidence to associate an observation to a known attack class rather than an unknown type. NERO also differentiates between two types of unknown attacks, namely, type_A and type_B attacks.

In Vu et al. (2020), an encoder–decoder paradigm is used to learn effective representations of known attack vectors into a regularized latent space. The authors in this work evaluated the convergence of the latent space to representations that permit shallow-ML classifiers to differentiate between regular traffic and unknown attack samples. The data-distribution decoupling strategy in NERO permits the detection of unknown attacks of more than one macro-category instead.

The work in Yang et al. (2021) offers a two-step ZdA attack detection based on conditional and variational encodings. In the first phase, a closed-set classification task is optimized in the latent space. In the second stage, the conditioned reconstructions of the learned latent variables are used to assess if an observation is an instance of an unknown attack. Extreme Value Theory concepts are used to determine the presence of unknown attacks as a function of the reconstruction errors of the known-class latent samples. Concerning this work, NERO uses episodic learning and thus incorporates a balancing technique for training. At the same time, our proposal uses neural machinery to learn to extract the novelty degree automatically through a specialized labeling strategy.

The authors of Sarhan et al. (2023) realized many experiments that resembled a zero-day attack detection problem by excluding one class at a time from the training set of two ML-based NID systems. The first was based on a random forest, and the second was based on a multi-layer perceptron. Authors used the NetFlow v9 format of the UNSW-NB15 dataset.[7] They used a nearest-neighbor assignation to classify out-of-distribution samples as attacks. Our work not only detects anomalies but also makes inferences about their degree of novelty.

The work in Hindy et al. (2020) compares the performance of ZdA detectors based on Support Vector Machines and Autoencoders. The authors of this work used the CIC-IDS2017[8] and NSL-KDD[9] datasets and trained both models using only normal data. During evaluation, all the attack-related samples were fed to the models. The position of the samples' latent representations concerning the support hyperplane and the reconstruction error were used to discriminate between attacks and benign traffic.

The work in Sameera and Shashi (2020) uses manifold alignment to create cluster correspondences between different slices of the same datasets and between different datasets. The alignment between a full-labeled source dataset and a partially labeled target dataset performs zero-day attack inferences on the second. To assess their method, the authors performed experiments using different chunks of the NSL-KDD dataset. Unlike this work, in NERO, the eigenvalue decomposition of big feature matrices is avoided using only deep learning-based latent spaces.

The authors of Thein et al. (2023) presented a Few-Shot classification pipeline based on a fusion of the prototypical and graph convolutional networks. Remarkably, they first transformed the flows' traffic captures to expressive bidimensional images using open-source tools. Then, they performed Few-Shot learning on the image embeddings. They validated their algorithm with a fraction of the IoT-23 dataset.[10] In contrast to the Few-Shot setting of this work, our method infers unknown attacks on an open-set basis, i.e., without using labels.

The authors of Nguyen and Le (2023) presented a pipeline for NID that inverts the approach of Bovenzi et al. (2020). First, a high-dimensional latent space is constructed, and a CNN divides inputs between benign and known attacks. Then, two anomaly-based classifiers analyze the traffic deemed as benign to identify oversights and novel attacks. The authors differentiate between novelty (type_B) and out-of-distribution (type_A) attacks and use their approach to detect only the former. NERO instead combines meta-learning and NAR to detect out-of-distribution attacks. Recent surveys on zero-day attack detection are available in Ahmad et al. (2023), Mearaj and Arif Wani (2023) and Guo (2023).

Most of the state-of-the-art related to zero-shot detection of novel attack types relies on multidomain meta-learning, where parallel feature domain(s) that *describe* the new kinds of attacks are assumed to be given at inference time. Other works focus on anomaly detection techniques but tend not to address the assessment of the degree of novelty of

[7] https://www.kaggle.com/datasets/dhoogla/nfunswnb15v2.

[8] https://www.unb.ca/cic/datasets/ids-2017.html.

[9] https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[10] https://www.stratosphereips.org/datasets-iot23.

anomalies. In this respect, the main contribution of our work relies on a framework proposal that uses a cascading *encode-process-decode* meta-learning strategy for learning to detect and *potentially* categorize out-of-distribution attacks without auxiliary-domain labels and within a low-training-data environment.

## 6. Conclusive remarks

This paper proposed NERO, a Deep Learning pipeline focused on learning to detect Zero-day Attacks with a particular focus on the IoT environment, i.e., with special care in training data efficiency. The Neural Algorithmic Reasoning and Prototypical meta-learning blueprint are followed to augment data efficiency and to model high-level abstract reasoning.

The presented algorithmic inductive biases represent a candidate strategy to equip a DL pipeline with a meta-level abstraction mechanism, decoupling its behavior from use-case-specific data distributions. Our proposal has demonstrated to potentially generalize the concept of type_A and type_B attacks regarding a two-level balanced binary classification accuracy. In other words, the NERO processor network is taught to distill the unknown hierarchy of distributions that generate anomalous data at inference time through the NAR inductive biases. The envisioned setting can be seen as an implicit or architectural mapping for the conditional discriminative criteria modeled in other recent works and extended to type_B and type_A ZdA distillation.

Other paradigms exist that seek to enable differentiable machine learning over symbolic reasoning to achieve out-of-distribution generalization. The interested reader can refer to some instances of *neuro-symbolic AI* (Hitzler and Kamruzzaman Sarker, 2022) as exemplary techniques. We mainly opt for the *NAR prototypical blueprint* over neuro-symbolic AI because the IoT scenario may require low computational and network overhead (Heidari et al., 2023a). In contrast, neuro-symbolic approaches may assume to work with a distributed modular system compressing knowledge bases and other processing subsystems that may be online queried.

The main future research direction consists of deploying the NERO pipeline in real IoT scenarios to assess its ability to online learning to detect type_A and type_B ZdAs. Additionally, research efforts should be devoted to enhancing the evaluation of open-set classification accuracies of the current version of our pipeline.

## CRediT authorship contribution statement

**Jesús F. Cevallos M.:** Writing – review & editing, Writing – original draft, Methodology, Formal analysis, Conceptualization. **Alessandra Rizzardi:** Writing – review & editing, Writing – original draft, Funding acquisition, Conceptualization. **Sabrina Sicari:** Writing – review & editing, Investigation, Funding acquisition, Conceptualization. **Alberto Coen Porisini:** Writing – review & editing, Funding acquisition, Conceptualization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

No data was used for the research described in the article.

## Acknowledgments

This work was supported in part by the SERENA-IIoT project, which has been funded by MUR (Ministero dell'Università e della Ricerca), Italy under the PRIN 2022 program (project code 2022CN4EBH), and in part by project SERICS (project code PE00000014), under the NRRP MUR program funded by the EU - NGEU, Italy.

## References

Affinito, Antonia, Zinno, Stefania, Stanco, Giovanni, Botta, Alessio, Ventre, Giorgio, 2023. The evolution of mirai botnet scans over a six-year period. J. Inf. Secur. Appl. 79, 103629.

Ahmad, Rasheed, Alsmadi, Izzat, Alhamdani, Wasim, Tawalbeh, Lo'ai, 2023. Zero-day attack detection: a systematic literature review. Artif. Intell. Rev. 1–79.

Al-Zewairi, Malek, Almajali, Sufyan, Ayyash, Moussa, 2020. Unknown security attack detection using shallow and deep ann classifiers. Electronics 9 (12), 2006.

Amiri, Zahra, Heidari, Arash, Navimipour, Nima Jafari, Unal, Mehmet, Mousavi, Ali, 2023. Adventures in data analysis: A systematic review of deep learning techniques for pattern recognition in cyber–physical-social systems. Multimedia Tools Appl. 1–65.

Ashiku, Lirim, Dagli, Cihan, 2021. Network intrusion detection system using deep learning. Procedia Comput. Sci. 185, 239–247.

Beurer-Kellner, Luca, Vechev, Martin, Vanbever, Laurent, Veličković, Petar, 2022. Learning to configure computer networks with neural algorithmic reasoning. arXiv preprint arXiv:2211.01980.

Bovenzi, Giampaolo, Aceto, Giuseppe, Ciuonzo, Domenico, Persico, Valerio, Pescapé, Antonio, 2020. A hierarchical hybrid intrusion detection approach in iot scenarios. In: GLOBECOM 2020-2020 IEEE Global Communications Conference. IEEE, pp. 1–7.

Brody, Shaked, Alon, Uri, Yahav, Eran, 2021. How attentive are graph attention networks?. arXiv preprint arXiv:2105.14491.

Cappart, Quentin, Chételat, Didier, Khalil, Elias.B, Lodi, Andrea, Morris, Christopher, Velickovic, Petar, 2023. Combinatorial optimization and reasoning with graph neural networks. J. Mach. Learn. Res. 24, 130–131.

Chanal, Poornima M., Kakkasageri, Mahabaleshwar S., 2020. Security and privacy in iot: a survey. Wirel. Pers. Commun. 115, 1667–1693.

Deac, Andreea, Veličković, Petar, Milinković, Ognjen, Bacon, Pierre-Luc, Tang, Jian, Nikolić, Mladen, 2020. Xlvin: executed latent value iteration nets. arXiv preprint arXiv:2010.13146.

Derrow-Pinion, Austin, She, Jennifer, Wong, David, Lange, Oliver, Hester, Todd, Perez, Luis, Nunkesser, Marc, Lee, Seongjae, Guo, Xueying, Wiltshire, Brett, et al., 2021. Eta prediction with graph neural networks in google maps. In: Proceedings of the 30th ACM International Conference on Information & Knowledge Management. pp. 3767–3776.

Dixit, Priyanka, Silakari, Sanjay, 2021. Deep learning algorithms for cybersecurity applications: A technological and status review. Comp. Sci. Rev. 39, 100317.

Dong, Guimin, Tang, Mingyue, Wang, Zhiyuan, Gao, Jiechao, Guo, Sikun, Cai, Lihua, Gutierrez, Robert, Campbel, Bradford, Barnes, Laura E., Boukhechba, Mehdi, 2023. Graph neural networks in iot: A survey. ACM Trans. Sensor Netw. 19 (2), 1–50.

Ferrag, Mohamed Amine, Friha, Othmane, Hamouda, Djallel, Maglaras, Leandros, Janicke, Helge, 2022. Edge-iiotset: A new comprehensive realistic cyber security dataset of iot and iiot applications for centralized and federated learning. IEEE Access 10, 40281–40306.

Gamage, Sunanda, Samarabandu, Jagath, 2020. Deep learning methods in network intrusion detection: A survey and an objective comparison. J. Netw. Comput. Appl. 169, 102767.

Guo, Yang, 2023. A review of machine learning-based zero-day attack detection: Challenges and future directions. Comput. Commun. 198, 175–185.

He, Ke, Kim, Dan Dongseong, Asghar, Muhammad Rizwan, 2023. Adversarial machine learning for network intrusion detection systems: a comprehensive survey. IEEE Commun. Surv. Tutor..

Heidari, Arash, Jabraeil Jamali, Mohammad Ali, 2023. Internet of things intrusion detection systems: A comprehensive review and future directions. Cluster Comput. 26 (6), 3753–3780.

Heidari, Arash, Navimipour, Nima Jafari, Jabraeil Jamali, Mohammad Ali, Akbarpour, Shahin, 2023a. A green, secure, and deep intelligent method for dynamic iot-edge-cloud offloading scenarios. Sustain. Comput.: Inform. Syst. 38, 100859.

Heidari, Arash, Navimipour, Nima Jafari, Unal, Mehmet, 2023b. A secure intrusion detection platform using blockchain and radial basis function neural networks for internet of drones. IEEE Internet Things J..

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

Hitzler, Pascal, Kamruzzaman Sarker, Md., 2022. Neuro-Symbolic Artificial Intelligence: The State of the Art. IOS Press.

Hospedales, Timothy, Antoniou, Antreas, Micaelli, Paul, Storkey, Amos, 2021. Meta-learning in neural networks: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 44 (9), 5149–5169.

Ibarz, Borja, Kurin, Vitaly, Papamakarios, George, Nikiforou, Kyriacos, Bennani, Mehdi, Csordás, Róbert, Dudzik, Andrew Joseph, Bošnjak, Matko, Vitvitskyi, Alex, Rubanova, Yulia, et al., 2022. A generalist neural algorithmic learner. In: Learning on Graphs Conference. PMLR, pp. 1–2.

Imrana, Yakubu, Xiang, Yanping, Ali, Liaqat, Abdul-Rauf, Zaharawu, 2021. A bidirectional lstm deep learning approach for intrusion detection. Expert Syst. Appl. 185, 115524.

Jayalaxmi, Pls, Saha, Rahul, Kumar, Gulshan, Conti, Mauro, Kim, Tai-Hoon, 2022. Machine and deep learning solutions for intrusion detection and prevention in iots: A survey. IEEE Access.

Khraisat, Ansam, Alazab, Ammar, 2021. A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, public datasets and challenges. Cybersecurity 4, 1–27.

Khraisat, Ansam, Gondal, Iqbal, Vamplew, Peter, Kamruzzaman, Joarder, 2019. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2 (1), 1–22.

Lei, Na, An, Dongsheng, Guo, Yang, Su, Kehua, Liu, Shixia, Luo, Zhongxuan, Yau, Shing-Tung, Gu, Xianfeng, 2020. A geometric understanding of deep learning. Engineering 6 (3), 361–374.

Liu, Xiaodong, Li, Tong, Zhang, Runzi, Wu, Di, Liu, Yongheng, Yang, Zhen, 2021a. A gan and feature selection-based oversampling technique for intrusion detection. Secur. Commun. Netw. 2021, 1–15.

Liu, Jiashuo, Shen, Zheyan, He, Yue, Zhang, Xingxuan, Xu, Renzhe, Yu, Han, Cui, Peng, 2021b. Towards out-of-distribution generalization: A survey. arXiv preprint arXiv:2108.13624.

Maseno, Elijah M., Wang, Zenghui, Xing, Hongyan, et al., 2022. A systematic review on hybrid intrusion detection system. Secur. Commun. Netw. 2022.

Mearaj, Nowsheen, Arif Wani, M., 2023. Zero-day attack detection with machine learning and deep learning. In: 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, pp. 719–725.

Nguyen, Xuan-Ha, Le, Kim-Hung, 2023. Robust detection of unknown dos/ddos attacks in iot networks using a hybrid learning model. Internet Things 23, 100851.

Pándy, Michal, Qiu, Weikang, Corso, Gabriele, Veličković, Petar, Ying, Zhitao, Leskovec, Jure, Liò, Pietro, 2022. Learning graph search heuristics. In: Learning on Graphs Conference. PMLR, pp. 10–11.

Raedt, Luc De, Dumančić, Sebastijan, Manhaeve, Robin, Marra, Giuseppe, 2020. From statistical relational to neuro-symbolic artificial intelligence. arXiv preprint arXiv:2003.08316.

Sameera, Nerella, Shashi, M., 2020. Deep transductive transfer learning framework for zero-day attack detection. ICT Express 6 (4), 361–367.

Santhoshb Kumar, SVN., Selvi, M, Kannan, A, et al., 2023. A comprehensive survey on machine learning-based intrusion detection systems for secure communication in internet of things. Comput. Intell. Neurosci. 2023.

Sarhan, Mohanad, Layeghy, Siamak, Gallagher, Marcus, Portmann, Marius, 2023. From zero-shot machine learning to zero-day attack detection. Int. J. Inf. Secur. 1–13.

Snell, Jake, Swersky, Kevin, Zemel, Richard, 2017. Prototypical networks for few-shot learning. Adv. Neural Inf. Process. Syst. 30.

Thein, Thin Tharaphe, Shiraishi, Yoshiaki, Morii, Masakatu, 2023. Few-shot learning-based malicious iot traffic detection with prototypical graph neural networks. IEICE Trans. Inf. Syst. 106 (9), 1480–1489.

Tsimenidis, Stefanos, Lagkas, Thomas, Rantos, Konstantinos, 2022. Deep learning in iot intrusion detection. J. Netw. Syst. Manage. 30, 1–40.

Veličković, Petar, Badia, Adrià Puigdomènech, Budden, David, Pascanu, Razvan, Banino, Andrea, Dashevskiy, Misha, Hadsell, Raia, Blundell, Charles, 2022. The clrs algorithmic reasoning benchmark. In: International Conference on Machine Learning. PMLR, pp. 22084–22102.

Veličković, Petar, Blundell, Charles, 2021. Neural algorithmic reasoning. Patterns 2 (7), 100273.

Veličković, Petar, Ying, Rex, Padovano, Matilde, Hadsell, Raia, Blundell, Charles, 2019. Neural execution of graph algorithms. arXiv preprint arXiv:1910.10593.

Vu, Ly, Nguyen, Quang Uy, Nguyen, Diep N., Hoang, Dinh Thai, Dutkiewicz, Eryk, et al., 2020. Learning latent representation for iot anomaly detection. IEEE Trans. Cybern. 52 (5), 3769–3782.

Wang, Yaqing, Yao, Quanming, Kwok, James T., Ni, Lionel M., 2020. Generalizing from a few examples: A survey on few-shot learning. ACM Comput. Surv. (CSUR) 53 (3), 1–34.

Yang, Jian, Chen, Xiang, Chen, Shuangwu, Jiang, Xiaofeng, Tan, Xiaobin, 2021. Conditional variational auto-encoder and extreme value theory aided two-stage learning approach for intelligent fine-grained known/unknown intrusion detection. IEEE Trans. Inf. Forensics Secur. 16, 3538–3553.

Yang, Zhen, Liu, Xiaodong, Li, Tong, Wu, Di, Wang, Jinjiang, Zhao, Yunwei, Han, Han, 2022. A systematic literature review of methods and datasets for anomaly-based network intrusion detection. Comput. Secur. 116, 102675.

Zhang, Zhun, Liu, Qihe, Qiu, Shilin, Zhou, Shijie, Zhang, Cheng, 2020. Unknown attack detection based on zero-shot learning. IEEE Access 8, 193981–193991.

**Jesús F. Cevallos M.** received a Ph.D. in Computer Science Engineering from Sapienza University (Rome) in 2022. He now covers a post-doc researcher position at University of Insubria (Varese). His main research interests are industrial applications of Deep Learning over heterogeneous networks, with a special focus on Deep Reinforcement Learning and Graph Representation Learning.

**Alessandra Rizzardi** is Assistant Professor at University of Insubria (Varese), where she received BS/MS degree in Computer Science 110/110 cum laude in 2011 and 2013, respectively. In 2016 she got Ph.D. in Computer Science and Computational Mathematics at the same university, under the guidance of Prof. Sabrina Sicari. Her research activity is on WSN and IoT security issues. She is member of ETT, ITL, and Sensors editorial board. She is IEEE member.

**Sabrina Sicari** is Associate Professor at University of Insubria (Varese). She received degree in Electronical Engineering, 110/110 cum laude, from University of Catania, in 2002, where in 2006 she got Ph.D. in Computer and Telecommunications Engineering, followed by Prof. Aurelio La Corte. She is member of COMNET, IEEE IoT, ETT, ITL editorial board. Her research activity security, privacy and trust in WSN, WMSN, IoT, and distributed systems. She is IEEE senior member.

**Alberto Coen Porisini** received Dr. Eng. degree and Ph.D. in Computer Engineering from Politecnico di Milano in 1987 and 1992. He is Full Professor of Software Engineering at Università degli Studi dell'Insubria since 2001, Dean of the School of Science from 2006 and Dean from 2012 to 2018. His research regards specification/design of real-time systems, privacy models and WSN.