Computers & Security 144 (2024) 103926

Contents lists available at ScienceDirect

Computers & Security

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

VINCENT: Cyber-threat detection through vision transformers and knowledge distillation

Luca De Rose [a], Giuseppina Andresini [a,b,*], Annalisa Appice [a,b], Donato Malerba [a,b]

[a] Dipartimento di Informatica, Università degli Studi Aldo Moro di Bari, Via Orabona 4, Bari, 70125, Italy
[b] Consorzio Interuniversitario Nazionale per l'Informatica - CINI, Via Orabona 4, Bari, 70125, Italy

ARTICLE INFO

Keywords:
Cyber-threat detection
Vision transformers
Convolution neural networks
Attention
Knowledge distillation
eXplainable Artificial Intelligence

ABSTRACT

Vision Transformers (ViTs) denote a family of attention-based deep learning techniques that have recently achieved amazing results in various problems related to the field of computer vision. In this paper, we explore the use of ViTs in problems of cyber-threat detection related to malware and network intrusion detection. In particular, we propose VINCENT, that is a novel deep neural method, which resorts to a color imagery representation of cyber-data by encoding related cyber-data features into neighboring color pixels. ViTs are trained from cyber-data images as teacher models, to extract explainable imagery signatures of cyber-data classes. This knowledge is extracted by leveraging the self-attention mechanism to give paired attention values between pairs of imagery patches. The signature knowledge, extracted through the ViT teacher, is, finally, used to train a smaller neural student model according to the knowledge distillation theory. Experiments with various benchmark cybersecurity datasets assess the accuracy of the student model VINCENT also compared to that of several state-of-the-art methods. In addition, it shows that VINCENT can obtain insights from explanations recovered through the self-attention mechanism of the ViT teacher.

1. Introduction

Cyber-threat detection is the practice of exploring security ecosystems, to detect cyber-threats that may compromise the security of systems or networks. Modern cybersecurity systems must recognize an increasing number of cyber-threats quickly and accurately to prevent attackers from having enough time to root around in sensitive data or services.

In the last decade, the predominance of deep learning in classification modeling has fostered the development of a generation of cybersecurity systems that use neural cyber-threat intelligence to gain accuracy in cyber-threat detection. Following this mainstream, several accurate deep neural networks, e.g., DNNs (Jullian et al., 2023), LSTMs (Aydın et al., 2022), CNNs (Andresini et al., 2022), GANs (Rabieinejad et al., 2023) and Autoencoders (Andresini et al., 2021a; Srikanth Yadav and Kalpana, 2022), have been recently used in various cybersecurity problems.

Although the priority of cybersecurity systems today is still to perform accurate detection and classification of cyber-threats, over the last few years explainability of deep neural models has emerged as a very active research field. In general, the explainability of classification models refers partially to how easily humans may comprehend their underlying assumptions and reasoning. Deep neural models are commonly learned as opaque models that are implicitly represented in the form of a huge number of numerical weights. However, explaining the effect of certain data features on cyber-threat predictions can contribute to allowing cybersecurity systems to benefit better from the trust of security stakeholders.

Vision Transformers (ViTs) (Dosovitskiy et al., 2021) are a recent family of deep learning techniques oriented towards the design of explainable deep neural models for imagery data. Over the last few years, ViTs have gained accuracy in several computer vision problems by taking advantage of self-attention mechanisms, to learn relationships between position-embedded patches of imagery data. In particular, the self-attention mechanism realizes an adaptive input selection that allows ViTs to focus better on the most relevant imagery information. In addition, this mechanism facilitates the achievement of the explainability of the deep neural model.

The present study is based on the results recently achieved by Andresini et al. (2022), with deep neural models trained for multi-class classification of images converted from flow features of network traffic data. Imagery representations of malware data have been recently adopted also in Seneviratne et al. (2022) and Sharma et al. (2023).

* Corresponding author at: Dipartimento di Informatica, Università degli Studi Aldo Moro di Bari, Via Orabona 4, Bari, 70125, Italy.
E-mail addresses: luca.derose@uniba.it (L. De Rose), giuseppina.andresini@uniba.it (G. Andresini), annalisa.appicei@uniba.it (A. Appice), donato.malerba@uniba.it (D. Malerba).

https://doi.org/10.1016/j.cose.2024.103926
Received 30 June 2023; Received in revised form 22 April 2024; Accepted 28 May 2024
Available online 31 May 2024

Therefore, by leveraging an imagery representation of cyber-data, we propose a new cyber-threat detection method, named VINCENT(ViT-based distillationN for CybEr-Threat detectionN), which trains a ViT on images of cyber-data, to discriminate benign data from multiple categories of cyber-threat data. Hence, one of the contributions of this study is the evaluation of the effectiveness of ViTs for detecting cyber-threats. For this purpose, we adopt the imagery representation technique described by Andresini et al. (2022), which can be used with any feature vector representation of cyber-data. However, we introduce a color encoding in place of the grayscale one adopted in the previous study. The self-attention mechanism of ViTs produces explanations that capture the effect of cyber-data features associated with imagery patches on the reasoning of classification models, disentangling benign data from multiple categories of malicious data.

Another contribution of this study is the distillation of the explanation information produced through ViTs into less complex, Convolution Neural Network (CNN) models trained to generalize in the same way. This is done according to *Knowledge Distillation* (KD) (Hinton et al., 2015), that is a learning strategy to allow a tiny, deep neural model (student network) to mimic a large, deep neural model (teacher network), in order to generalize better on data by possibly mitigating overfitting. In this study, the distilled knowledge of the teacher network is transferred to the student network using the same training set. The KD process is performed through a loss function with the optimization target of matching the class-wise probability distribution of the student network to the probability distribution calculated by the teacher. Specifically, we use KD to transfer the explanation information distilled from large ViT teachers to tiny, CNN students.

To the best of our knowledge, there are a few studies that explore KD in cybersecurity (Xia et al., 2022; Wang et al., 2022) or train ViTs for cyber-threat detection problems (Ho et al., 2022; Seneviratne et al., 2022). However, this is the first study that formulates a KD strategy to transfer explanation knowledge, distilled through the self-attention mechanisms of ViTs, by training simple, CNN students that are, notably, more accurate than teachers. Our argument is mainly supported by experiments performed in four benchmark cybersecurity datasets regarding both malware detection problems (Maldroid20 and MalMem22) and network intrusion detection problems (NSL-KDD and UNSW-NB15). The results of these experiments show that the KD deep neural model trained through VINCENT, gains accuracy compared to both the teacher model and the student model, trained without the use of the KD strategy, as well as the KD deep neural model, trained to neglect the explanation information in the distillation process. In addition, VINCENT gains accuracy compared to several state-of-the-art competitors, experimented on the same data, by obtaining insights from explanations recovered through the self-attention mechanism of the ViT teacher.

In short, the contributions provided by this paper are as follows.

- A novel cyber-threat detection method that adopts an imagery representation of cyber-data and trains ViTs for cyber-threat detection, to learn relationships between multiple patches of cyber-data features and incorporate explainability directly into the classification model.
- A KD strategy that transfers explanation information distilled from large ViTs to tiny CNNs by achieving accurate models to be deployed for the predicting stage.
- The exploration of explanations produced through the ViT self-attention mechanism, which discloses the effect of the cyber-data features on the signatures of the malicious families detected.
- An extensive experimentation that shows the effectiveness of the proposed method in the multi-class scenario of four, benchmark cybersecurity datasets.

The remainder of this paper is organized as follows. Section 2 provides an overview of recent advances in the cybersecurity literature in XAI and KD. The proposed VINCENT method is described in Section 3. The experimental setup is illustrated in Section 4. The results of the evaluation of the proposed method are discussed in Section 5 and Section 6 regarding accuracy and explainability, respectively. Finally, Section 7 recalls the purpose of our research, draws conclusions, and illustrates future research directions.

2. Related work

This study describes a cyber-treat detection method that activates the KD strategy by transferring explanation information, distilled through ViT teachers to CNN, students. The literature overview is organized on two fronts. On one hand, we focus on recent studies using eXplainable Artificial Intelligence (XAI) with deep learning in the realm of cybersecurity. On the other hand, we overview preliminary attempts to use KD strategies in cybersecurity.

Regarding the XAI literature, for a few years, several studies on cybersecurity have explored the use of existing, post-hoc, XAI techniques, to measure which cyber-data features mainly influence the prediction of malicious network traffic (Wang et al., 2020; Andresini et al., 2021d; Keshk et al., 2023) or to use explanations to improve the accuracy of opaque deep neural models (Caforio et al., 2021). Post-hoc explainers (e.g., SHAP or DALEX) are model-agnostic. They can be used independently of the model. So, they do not have any effect on the model training.

On the other hand, a few recent studies have started focusing on intrinsic explainable deep neural models, e.g., deep neural models trained with attention mechanisms. One of the seminal studies in this direction is performed by Liu et al. (2020), who describe a Bidirectional Gated Recurrent Unit network with hierarchical attention trained for the binary classification of network traffic data. Zhao et al. (2022) illustrate a Temporal CNN with the attention mechanism, while Tang et al. (2020) propose a Stacked Autoencoder with an attention mechanism. Both these studies use the attention mechanism to gain accuracy in multi-class classification of network traffic data. Andresini et al. (2022) have recently described a CNN model with attention for accurate, explainable multi-class classification of network traffic data. The study adopts an imagery representation of network traffic data and explores the beneficial effects of the attention mechanism on both the accuracy and explainability of the decisions yielded by the method.

The use of an imagery representation of cyber-data paves the way for applying recent, explainable computer vision techniques also in cybersecurity problems. Since their emergence (Dosovitskiy et al., 2021), ViTs have rapidly dominated recent years of research and practice in computer vision. A survey (Han et al., 2023) reviews ViT models by categorizing them into different computer vision tasks and analyzing their advantages and disadvantages. Pasquadibisceglie et al. (2023) investigate the use of ViTs coupled with adversarial training to gain accuracy in next activity prediction problems. Both Ho et al. (2022) and Seneviratne et al. (2022) use ViTs in problems of network intrusion detection and malware classification, respectively. However, Ho et al. (2022) force the balanced condition in data. Seneviratne et al. (2022) adopt an imagery encoding that can be used with bytecode obtained from the Android APK only.

Regarding the KD literature, a few recent studies have explored KD in cybersecurity. One of the studies in this direction is authored by Wang et al. (2022) who describe a KD model based on Triplet CNN for network intrusion detection in industrial cyber–physical systems. The Triplet CNN is chosen to reduce the distance between the teacher and the student model outputs. Xia et al. (2022) describe a KD strategy with an intermediate loss for Android malware detection via a multi-layer perceptron. Their study shows that a student network can learn more hint knowledge from the intermediate layers of the teacher network in the process of transferring knowledge.

In conclusion, over the last few years, recent advances in computer vision have also been made in cybersecurity. ViTs, which are dominating computer vision research today, have recently made their

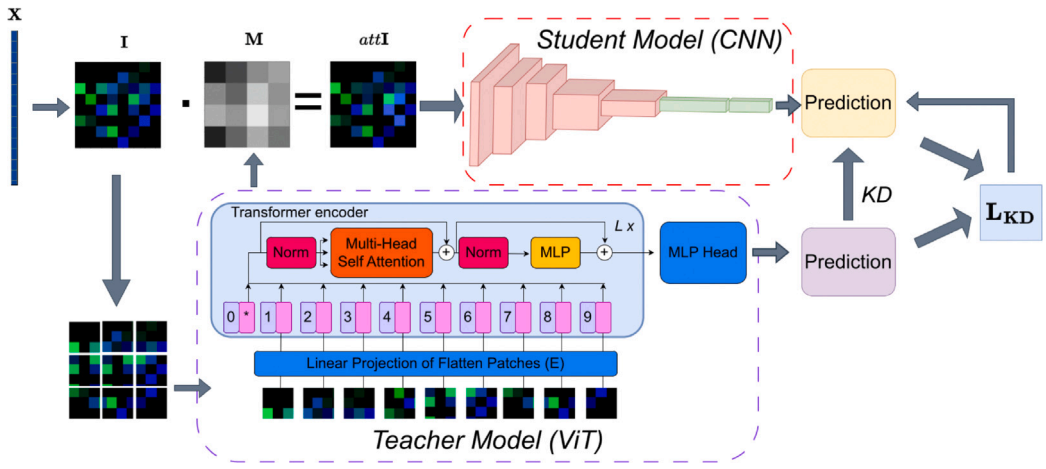


Fig. 1. Schema of VINCENT.

appearance in a few studies on cybersecurity. However, to the best of our knowledge, no studies in or outside the cybersecurity field explore the use of ViTs as a springboard for KD. On the other hand, the KD strategy, which is widely explored in vision recognition, natural language processing and speech recognition (see Gou et al. (2021) for a recent survey), is marginally explored in cyber-threat detection issues and never in connection with ViTs.

3. ViT-based distillation

Let us consider a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of N training samples. Every $\mathbf{x} \in \mathbb{R}^d$ denotes a d-dimensional numeric vector, that records the features of cyber-data traces (e.g., the number of transmitted packets in network traffic flow traces and the number of api calls in Android applications). Every $y \in \{1, \ldots, K\}$ represents the class with K distinct categories to indicate: *benign* cyber-data traces and various categories of *malicious activities*, depending on those historically detected and labeled. VINCENT resorts to a KD approach to train a deep neural model $M_\theta : \mathbb{R}^d \mapsto \{1, \ldots, K\}$ composed of a teacher model T_θ and a student model S_θ, where T_θ is a ViT, while S_θ is a CNN. Initially, VINCENT takes D as input and transforms each labeled sample $(\mathbf{x}_i, y_i) \in D$ into a labeled, color image $(\mathbf{I}_i, y_i) \in \mathcal{I}$. Subsequently, it uses $\mathcal{I}$ to pre-train T_θ. The explanation knowledge, distilled through the self-attention blocks of the pre-trained T_θ, is used to train S_θ. The KD strategy allows S_θ to mimic T_θ by minimizing the Kullback–Leibler (KL) divergence loss between the softened probability distributions of both T_θ and S_θ with the temperature scaling hyperparameter τ. VINCENT's method is schematized in Fig. 1.

3.1. Color imagery encoding

The color imagery encoding step is performed to transform the numeric vector $\mathbf{x}_i$ of each training sample $(\mathbf{x}_i, y_i) \in \mathcal{D}$ into a square color image $\mathbf{I}_i \in \mathbb{N}^{m \times m \times 3}$ so that $(\mathbf{I}_i, y_i)$ can be added to the imagery dataset $\mathcal{I}$. The imagery transformation is done by assigning each feature of $\mathbf{x}_i$ to a three-channel pixel frame of $\mathbf{I}_i$. Sample features are assigned to neighboring pixels in the image, depending on their correlation, and are not simply stacked in an arbitrary order. A description of how feature-pixel assignment is performed is given in Andresini et al. (2021c).

Each numeric feature value x is converted into a three-channel pixel by resorting to the RGB-like encoding function adopted in Pasquadibisceglie et al. (2020). First, x is scaled into the range $[0, 2^{24} - 1]$. Then the scaled value is considered as the value of a single 24-bit pixel with the first eight bits mapped into the Red band, the intermediate eight bits mapped into the Green band and the last eight bits mapped into the Blue band.

3.2. ViT architecture

A ViT architecture is considered to pre-train a teacher model T_θ. The ViT model follows the original Transformer architecture designed for NLP by Dosovitskiy et al. (2021). As the Transformer takes one-dimensional sequences of two-dimensional word embeddings as input (Vaswani et al., 2017), a flattening operator is adopted to transform a three-dimensional color image into a sequence of two-dimensional patches. Let $\phi : \mathbb{N}^{m \times m \times 3} \mapsto \mathbb{N}^{n \times (p^2 \cdot 3)}$ be the flattening operator, where $m \times m$ is the resolution of the original images and $p \times p$ is the resolution of each imagery patch, so that $n = \frac{m^2}{p^2}$ is the number of imagery patches per image. Let us consider a three-dimensional color image $\mathbf{I}_i \in \mathbb{N}^{m \times m \times 3}$, $\phi(\mathbf{I}_i) = \mathbf{p}_{i_1}, \mathbf{p}_{i_2}, \ldots, \mathbf{p}_{i_n}$ with each imagery patch $\mathbf{p}_{i_j}$ having size $p^2 \cdot 3$. The Transformer uses a constant latent vector $\mathbf{E} \in \mathbb{R}^{(p^2 \cdot 3) \times D}$, through all its layers, to map flattened patches extracted through ϕ to D dimensions with a trainable linear projection. The output of this projection corresponds to the patch embedding. A learnable class embedding, denoted as **class**, is placed before the patch embeddings. The value of the **class** token represents the classification output y. Finally, the patch embeddings are augmented with one-dimensional positional embeddings $\mathbf{E}_{pos} \in \mathbb{R}^{(n+1) \times D}$. These positional embeddings inject positional information into the input. The output of the described transformation is defined as follows:

$$\mathbf{z_0} = [\mathbf{class}; \mathbf{p}_{i_1}\mathbf{E}; \mathbf{p}_{i_2}\mathbf{E}; \ldots; \mathbf{p}_{i_n}\mathbf{E};] + \mathbf{E}_{pos}, \tag{1}$$

where $\mathbf{z}_0$ feeds the Transformer encoder architecture defined by Vaswani et al. (2017). The encoder consists of a stack of L identical blocks alternating multi-headed self-attention and MLP blocks. These blocks compute the outputs:

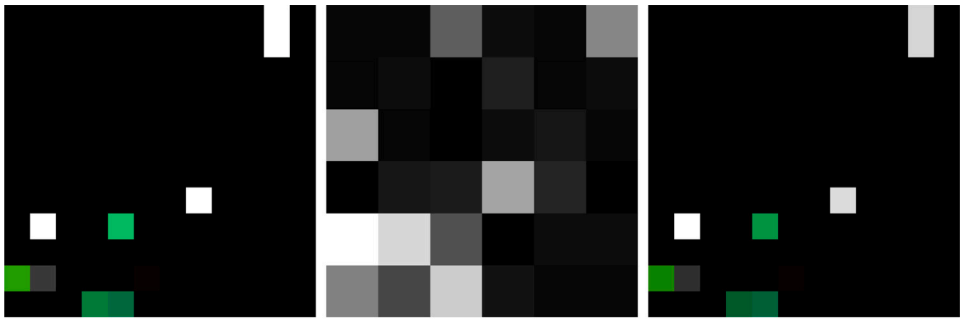
$$\mathbf{z'}_i = msa(LN(\mathbf{z}_{i-1})) + \mathbf{z}_{i-1}, \tag{2}$$

$$\mathbf{z}_i = mlp(LN(\mathbf{z'}_i)) + \mathbf{z'}_i, \tag{3}$$

where $i = 1, \ldots, L-1$, $msa(\bullet)$ is the multi-headed self-attention block, $mlp(\bullet)$ is the MLP block and $LN(\bullet)$ is the Layernorm that is applied before every block. Finally, the value **class** of the Lth block of the encoder output feeds into an MLP classification head with one hidden layer and GELU non-linearity. A self-attention block is computed as a weighted sum of all values v in the sequence $\mathbf{z}_i$ by using two linear transformations of the input, namely queries ($\mathbf{q}$) and keys ($\mathbf{k}$), respectively. Both $\mathbf{q}$ and $\mathbf{k}$ generate input transformations with the same length w. Specifically, a self-attention block is computed as:

$$sa(\mathbf{z'}_i) = softmax(\mathbf{q}_i \mathbf{k}_i^T / \sqrt{w})\mathbf{v}_i. \tag{4}$$

Eq. (2) is computed as the concatenation of multiple $sa(\bullet)$ run in parallel.



(a) $\mathbf{I}$ (b) $\mathbf{M}$ (c) $att\mathbf{I}$

Fig. 2. The imagery representation $\mathbf{I}$ of a DoS sample recorded in the dataset NSL-KDD (Fig. 2(a)), the attention map $\mathbf{M}$ of the decision of the ViT on this sample (Fig. 2(b)) and the attention-based imagery representation $att\mathbf{I}$ of the DoS sample (Fig. 2(c)).

Once the parameters of the ViT model have been pre-trained on $\mathcal{I}$, the attention roll-out method (Abnar and Zuidema, 2020) is used to extract the maps of attention of ViT decisions on training images. For each imagery sample $\mathbf{I}_i \in \mathcal{I}$, the roll-out method computes the map of attention $\mathbf{M}_i \in \mathbb{R}^{m \times m}$ from the output token to the input space of a sample, by averaging the attention weights of the ViT model across all heads and then recursively multiplying the weight matrices of all layers. This accounts for the mixing of attention across patches through all layers. As reported by Dosovitskiy et al. (2021), a map of attention extracted through the roll-out method is a matrix of the attention weights calculated between each patch in the image and all the other patches. Weights are scaled between 0 and 1. A map of attention can be visualized as a grid of heatmaps, where each heatmap represents the attention weights between a given patch and all the other patches. The brighter the color of a pixel in the heatmap, the greater the attention weight between the corresponding patches. According to the description of Li et al. (2022), the following multiplication is finally computed:

$$att\mathbf{I}_i = \mathbf{M}_i \cdot \mathbf{I}_i, \tag{5}$$

to obtain a new square color image (i.e., $att\mathbf{I}_i \in \mathbb{N}^{m \times m \times 3}$), that explains which parts of $\mathbf{I}_i$ are the most important for the classification task at hand. Let $att\mathcal{I}$ denote the collection of attention-based representations of images collected in $\mathcal{I}$. Fig. 2(a) shows an example of the imagery representation of a sample belonging to the DoS category in dataset NSL-KDD that was used in the experimental study. Fig. 2(b) shows the attention map of the decision of the ViT that sees this sample in the DoS class. Fig. 2(c) shows the attention-based imagery representation of the DoS sample obtained by multiplying the original imagery representation and the attention map according to Eq. (5).

3.3. KD architecture

The teacher model T_θ of the KD architecture of VINCENT is the ViT pre-trained by $\mathcal{I}$. As described by Hinton et al. (2015), this teacher model is pre-trained and its weights are frozen in the KD training phase. The student model S_θ is a CNN. Let $\hat{y}_{T_\theta}$ denote the output predictions computed with T_θ using the labeled color images set $(\mathbf{I}, y) \in \mathcal{I}$, obtained as described in Section 3.1 as input. S_θ is trained with the labeled attention-based images $(att\mathbf{I}, y) \in att\mathcal{I}$, computed as defined in Section 3.2, as input. Let $\hat{y}_{S_\theta}$ denote the predictions determined with S_θ. The KD loss is learned to minimize:

$$\mathbf{L_{KD}} = \mathbf{L_{CE}}(\hat{y}_{S_\theta}, y) + \lambda \mathbf{L_{KL}}(\hat{y}_{T_\theta}, \hat{y}_{S_\theta}), \tag{6}$$

where λ is an input parameter used to balance the two terms. The term $\mathbf{L_{CE}}(\hat{y}_{S_\theta}, y) = \sum_{i=1}^{K} y_i \log \hat{y}_{S_{\theta_i}}$ refers to the standard cross-entropy loss. The term $\mathbf{L_{KL}}(\hat{y}_{T_\theta}, \hat{y}_{S_\theta})$ refers to the Kullback–Leibler (KL) divergence, computed between the outputs of both T_θ and S_θ to distill the attention-based knowledge from the ViT teacher to the CNN student. Formally, the KL divergence is computed as follows:

$$\mathbf{L_{KL}}(\hat{y}_{T_\theta}, \hat{y}_{S_\theta}) = KL(\sigma(\frac{\hat{y}_{T_\theta}}{\tau}), \sigma(\frac{\hat{y}_{S_\theta}}{\tau}))\tau^2, \tag{7}$$

where σ is the softmax activation function and τ is a user-defined parameter, called temperature, introduced to generate soft labels from both $\hat{y}_{T_\theta}$ and $\hat{y}_{S_\theta}$. The higher the value of τ, the softer the probability distribution computed over the classes (Hinton et al., 2015).

4. Experimental set-up

In this section we describe the datasets used to evaluate the performance of VINCENT and the implementation details.

4.1. Dataset description

We considered four datasets to evaluate the performance of VINCENT: CICMalDroid20 (Mahdavifar et al., 2022),[1] CICMalMem22 (Carrier et al., 2022),[2] NSL-KDD (Tavallaee et al., 2009)[3] and UNSW-NB15 (Moustafa and Slay, 2015).[4] The four datasets contain a feature-vector representation of multi-class data traces, collected in various cybersecurity applications. CICMalDroid20 contains Android apps collected from several sources s (e.g., VirusTotal service and Contagio security blog). CICMalMem22 contains memory dumps of several families of both obfuscated malware and benign software. NSL-KDD and UNSW-NB15 collect network flow traces. In addition, both NSL-KDD and UNSW-NB15 contain several rare attacks (e.g., U2R and R2L in NSL-KDD, Analysis, Backdoor, Shellcode and Worms in UNSW-NB15). Each dataset was divided into a training set and a testing set. For each dataset, we used the training set to learn a classification model, and the testing set to measure the accuracy performance of the classification model learned from the training samples. For the experiments with NSL-KDD, we adopted the data setting that includes KDDTrain+20Percent as the training set and KDDTest+ as the testing set (Tavallaee et al., 2009). For the experiments with UNSW-NB15 we used the training set and testing set described by Moustafa and Slay (2015). For the experiments with both CICMalDroid20 and CICMalMem22, we used a stratified division of the original datasets in the training sets (70%) and the testing sets (30%). Table 1 reports a description of these datasets (i.e., number of features, number of samples collected for each class in both the training and the testing set).

[1] https://www.unb.ca/cic/datasets/maldroid-2020.html.
[2] https://www.unb.ca/cic/datasets/malmem-2022.html.
[3] https://www.unb.ca/cic/datasets/nsl.html.
[4] https://research.unsw.edu.au/projects/unsw-nb15-dataset.

Table 1
Dataset description: number of features (#**A**), number of samples (#**N**) and number of samples per class in both the training set and testing set.

Dataset	#A	Split	#N	#Samples per Class
CICMalDroid20	41	Train	8118	Benign (1256), Adware (877), Banking (1470), SMS (2733), Riskware (877)
		Test	3480	Benign (539), Adware (376), Banking (630), SMS (1171), Riskware (764)
CICMalMem22	56	Train	46876	Benign (23438), Ransomware (7832), Spyware (8016), Trojan (7590)
		Test	11720	Benign (5860), Ransomware (1959), Spyware (2004), Trojan (1897)
NSL-KDD	41	Train	125973	Benign (67343), DoS (45927), Probe (11656), R2L (995), U2R (52)
		Test	22544	Benign (9711), DoS (7458), Probe (2421), R2L (2754), U2R (200)
UNSW-NB15	41	Train	175341	Benign (56000), Analysis (2000), DoS (12264), Backdoor (1746), Exploits (33393), Fuzzers (18184), Generic (40000), Reconnaissance (10491), Shellcode (1133), Worms (130)
		Test	82332	Benign (37000), Analysis (677), DoS (4089), Backdoor (583), Exploits (11132), Fuzzers (6062), Generic (18871), Reconnaissance (3496), Shellcode (378), Worms (44)

4.2. Implementation details

VINCENT was developed in Python 3, using the high-level neural network API Keras 2.4, with TensorFlow as back-end.[5]

In the pre-processing phase, categorical input features were mapped into numerical features, using the one-hot-encoder strategy, while numerical features were scaled in the range [0,1], using the min–max scaler. In the imagery encoding phase, the imagery size was automatically set equal to the smallest even integer greater than (or equal to) the square root of the number of features produced in the pre-processing phase.

For each dataset, the hyper-parameters of deep neural network architectures were optimized using the tree-structured Parzen estimator algorithm as implemented in the Hyperopt library (Bergstra et al., 2013). In particular, the hyper-parameter optimization was performed using a random stratified split of 20% of the entire training as a validation set, sampled according to the Pareto principle. The hyper-parameter configuration that achieved the lowest validation loss was automatically selected. The hyper-parameter search space adopted in this study is reported in Table 2.

The ViT encoder was defined with four blocks (that alternated multi-head self-attention and MLP). A Normalization layer was used to normalize each sample in the batch, to bring the activation mean close to zero and the activation standard deviation close to one. Finally, a Fully Connected layer was defined with softmax as the final classification.

The CNN was implemented according to description given in Andresini et al. (2021c). It was defined with three Convolutional layers, two Dropout layers, two hidden Fully-Connected layers and one Fully-Connected layer with the softmax function as the classification layer.

The *ReLu* activation function was used on all hidden layers of both ViT and CNN architectures. The deep neural models were trained with mini-batches by back-propagation. The gradient-based optimization was performed using Adam's update rule. Furthermore, the maximum number of epochs was set equal to 150, and an early stopping approach based on the lowest loss on the validation set was used, to retain the best models.

[5] https://github.com/Kyanji/VINCENT

Table 2
Hyper-parameter search space for VINCENT.

Hyper-parameter	Values
Mini-batch size	$\{2^6, 2^7, 2^8, 2^9\}$
Learning rate	[0.0001, 0.001]
Dropout rate	[0, 1]
λ	[0, 1]
τ	{2, 3, 4, 5, 6, 7, 8, 9, 10}

5. Accuracy performance analysis

In this section, we illustrate the results of the analysis performed to evaluate the accuracy performance of VINCENT. First, we present the results of the sensitivity study conducted to explore the effect of the patch size on the accuracy performance of VINCENT (Section 5.1). Then, we describe the ablation study performed to analyze the effect of the various learning components of VINCENT (Section 5.2). Finally, we compare VINCENT to other state-of-the-art methods defined for cyber-threat detection (Section 5.3).

For each method, we computed standard classification metrics to measure the accuracy performance of the classifications produced on the testing samples. In particular, we measured: the average F1 score per class type (Macro-F1), the weighted F1 score per class type (Weighted-F1) and the overall accuracy (OA).

5.1. Patch size sensitivity analysis

We performed a sensitivity analysis to explore how the accuracy performance of VINCENT can be influenced by the set-up of patch size p. Table 3 collects the Macro-F1, Weighted-F1 and OA of VINCENT

Table 3
Macro-F1, Weighted-F1 and OA of VINCENT with patch size $p = 2$ and $p = 4$, respectively. The best results are in bold.

Dataset	Patch size	Macro-F1	Weighted-F1	OA
CICMalDroid20	2	0.841	0.871	0.872
	4	**0.848**	**0.874**	**0.875**
CICMalMem22	2	**0.739**	**0.825**	**0.826**
	4	0.706	0.803	0.803
NSL-KDD	2	**0.669**	**0.818**	**0.829**
	4	0.639	0.810	0.821
UNSW-NB15	2	**0.474**	**0.781**	**0.784**
	4	0.439	0.775	0.774

run with both $p = 2$ and $p = 4$. The results show that training the ViT teacher by accounting for finer-grained imagery patches (i.e., $p = 2$) allows VINCENT to gain accuracy. The only exception occurs in CICMalDroid20, although, in this dataset, the difference between the accuracy performance obtained by VINCENTwith $p = 2$ and the accuracy obtained with $p = 4$ is negligible (0.841 vs. 0.848 for Macro-F1, 0.871 vs. 0.874 for Weighted-F1 and 0.872 vs. 0.875 for OA). In any case, as CICMalDroid20 is the more balanced dataset of this study,[6] we delved deeper into the analysis of a possible relationship between the patch size set-up and the balanced condition of the problem. To this purpose, we considered a balanced version of the CICMalMem22, named CICMalMem22-Balanced, that was obtained by down-sampling samples in the majority class (i.e., Benign) to the size of the runner-up class (i.e., Spyware). Specifically, the training set of CICMalMem22-Balanced includes 8016 samples in class "Benign", 7832 samples in class "Ransomware", 8016 samples in class "Spyware" and 7590 samples in class "Trojan". The testing set of CICMalMem22-Balanced includes 2004 samples in class "Benign", 1959 samples in class "Ransomware", 2004 samples in class "Spyware" and 1897 samples in class "Trojan". We decided to consider this dataset for this balancing operation since it does not contain minority classes present with very few examples in the original dataset. So, it allows us to build a balanced dataset without creating synthetic artificial samples but keeping enough samples for both the training and testing stages.

The accuracy performance results achieved by running VINCENT, with both $p = 2$ and $p = 4$, on CICMalMem22-Balanced are reported in Fig. 3. These results support the intuition that VINCENT run with $p = 4$ can gain accuracy compared to VINCENT run with $p = 2$ in the presence of balanced data. Although data imbalance is a common issue in cybersecurity problems (Pawlicki et al., 2020), these additional results support the thesis that larger patch sizes may be preferable to smaller ones in the presence of balanced data. Our intuition is that finer-grained patches can somehow facilitate the disentanglement of minority classes from majority classes by taking advantage of the better separation of groups of uncorrelated pixels in different patches. On the other hand, finer-grained patches can somehow lead the decision model to lose its generalization ability in the presence of approximately balanced data. This preliminary achievement paves the way for further investigations in this direction, suggesting that more sophisticated mechanisms should be incorporated into the approach to allow the automatic selection of p according to the characteristics of the source dataset.

Finally, according to the results reported in Table 3, we can observe that VINCENT run with $p = 2$ achieves a remarkable gain in accuracy compared to VINCENT run with $p = 4$ in CICMalMem22, NSL-KDD and UNSW-NB15. On the other hand, VINCENT, run with $p = 2$, shows a negligible loss of accuracy in CICMaldroid22. For example, Macro-F1 is up a full 3 percentage of points with $p = 2$ in CICMalMem22, NSL-KDD and UNSW-NB15, and down just 0.7 percentage of points in CICMalDroid20. Based on this analysis, we decided to consider VINCENT run with $p = 2$ in the rest of this evaluation study.

[6] We would thank one of the anonymous reviewers of this paper for this intuition.

Table 4
Macro-F1, Weighted-F1 and OA of both VINCENT and its baseline configurations: CNN, ViT, ViT+KL and CNN+KD. The best results are in bold.

Dataset	Architecture	Macro-F1	Weighted-F1	OA
CICMalDroid20	CNN	0.696	0.738	0.740
	ViT	0.701	0.748	0.757
	ViT+KD	0.733	0.773	0.780
	CNN+KD	0.741	0.777	0.783
	VINCENT	**0.841**	**0.871**	**0.872**
CICMalMem22	CNN	0.531	0.685	0.702
	ViT	0.501	0.664	0.685
	ViT+KD	0.712	0.808	0.808
	CNN+KD	0.696	0.797	0.797
	VINCENT	**0.739**	**0.825**	**0.826**
NSL-KDD	CNN	0.488	0.716	0.767
	ViT	0.495	0.723	0.760
	ViT+KD	0.542	0.743	0.771
	CNN+KD	0.549	0.744	0.768
	VINCENT	**0.669**	**0.818**	**0.829**
UNSW-NB15	CNN	0.325	0.677	0.650
	ViT	0.331	0.686	0.677
	ViT+KD	**0.478**	0.770	0.765
	CNN+KD	0.387	0.746	0.741
	VINCENT	0.474	**0.781**	**0.784**

5.2. Ablation study

The ablation study of VINCENT was performed by evaluating the accuracy performance of three deep neural architectures identified as baselines of VINCENT. These baselines were defined as follows:

- CNN, that was obtained by discarding the KD strategy and using a CNN architecture, trained on the original imagery representation of the cyber-data, as the classification model.
- ViT, that was obtained by discarding the KD strategy and using the ViT architecture, trained on the original imagery representation of the cyber-data, as the classification model.
- ViT+KD, that was obtained by keeping the KD strategy but completing the training of the CNN student without accounting for the attention information that can be extracted through the ViT teacher. Notably, this configuration trained the ViT teacher and the CNN student as defined in VINCENT. However, in VINCENT, the ViT teacher was trained on the original images of the training samples, while the student was trained on the original images multiplied by the attention values extracted for the same images through the ViT teacher. Instead, in ViT+KD, both the ViT teacher and the CNN student were trained on the original imagery representation of the training samples.
- CNN+KD, that was obtained by keeping the KD strategy but replacing the ViT architecture of the teacher branch of VINCENTwith a simpler CNN architecture. Specifically, the teacher branch of CNN+KD used a CNN architecture with the same layer structure of the CNN used in the student branch. However, as the configuration CNN+KD realized the same distillation pipeline formulated for VINCENT, it also trained the student on the original images appropriately multiplied by the attention values produced by the teacher. To obtain these attention values through a CNN teacher, we integrated a *pixel-attention* layer followed by an *average layer* after the last convolutional layer of the CNN teacher. This was done according to the implementation illustrated by Andresini et al. (2022). The output of the *average layer* was then used as the map of attention multiplied by the original imagery representation to obtain the attention-based imagery representation used as input of the student model in the distillation strategy.

Table 4 collects the results of Macro-F1, Weighted-F1 and OA of CNN, ViT, CNN+KD, ViT+KD and VINCENT. The results show that VINCENT takes advantage of all the learning components of the

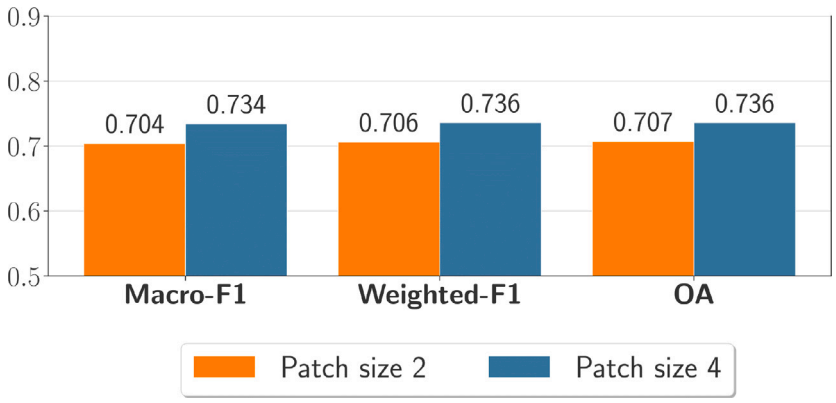


Fig. 3. Macro-F1, Weighted-F1 and OA achieved by VINCENT run with patch size $p = 2$ and $p = 4$ in CICMalMem22-Balanced.

proposed KD architecture producing more accurate classifications than all its baselines. In fact, VINCENT achieves a significant gain in all the accuracy metrics, compared to the baselines CNN and ViT, that discard the KD strategy. In addition, VINCENT also outperforms both ViT+KD and CNN+KD that integrate a KD strategy in their pipeline. In particular, the ViT+KD configuration is the runner-up approach of this experimental study for both CICMalMem22, NSL-KDD and UNSW-NB15 datasets. The only exception is achieved by CICMaldroid20, where the runner-up approach is the CNN+KD configuration. These results show that the KD architecture proposed by VINCENT can take advantage of the use of the ViT architecture to train a teacher model, as well as the consideration of the attention information that can be produced through the ViT teacher, to obtain a better representation of the input images for the distillation of the CNN student. On the other hand, VINCENT suffers from a negligible loss of Macro-F1 only in UNSW-NB15, although it achieves higher Weighted-F1 and OA than KD+ViT also in UNSW-NB15.

Fig. 4 reports the F1 score computed for each class in the four datasets of this study. These detailed results show that VINCENT generally performs better than CNN, ViT, ViT+KD and CNN+KD in disentangling the samples of different class types in all the multi-class problems of this study. In particular, VINCENT outperforms its baselines also in predicting rare classes of both NSL-KDD (i.e., R2L and U2R) and UNSW-NB15 (i.e., Backdoor and Shellcode). However, a few exceptions are observed with DoS and Worms classes in UNSW-NB15, that is the dataset with the anomaly in the Macro-F1 score measured for VINCENT. The class of DoS intrusions is one of the most difficult classes to recognize in UNSW-NB15. In fact, no method of this ablation study can recognize DoS intrusions by achieving a score of F1 greater than 0.2. VINCENT performs worse than both CNN and ViT+KD in this class. Similarly, VINCENT performs worse than ViT+KD in the rare class of Worms, although it outperforms both CNN, ViT and CNN+KD in the correct classification of Worms. Finally, we note that VINCENT does not gain any accuracy in the recognition of the rare class Analysis of UNSW-NB15. However, in this case, no method of the ablation study can correctly recognize any Analysis intrusion. Finally, based on this analysis, we can explain the fact that the macro-F1 score of VINCENT is slightly lower than the macro-F1 score of ViT+KD in UNSW-NB15. Although VINCENT outperforms (or performs equally to) ViT+KD in several majority classes (i.e., Benign, Exploits, Fuzzers and Generic) and rare classes (i.e., Reconnaissance and Shellcode) of UNSW-NB15, it performs worse than ViT+KD in a few rare classes (i.e., DoS, Analysis and Worms). This causes a decrease of the Macro-F1 score since all the classes equally contribute to the measurement of Macro-F1. However, there is an increase of both Weighted-F1 and OA since the accuracy performance of rare classes contributes less than the accuracy performance of majority classes to the measurement of both Weighted-F1 and OA.

Table 5
Macro-F1, Weighted-F1 and OA of VINCENT and several competitors reported in the recent literature. "–" denotes that the code was not publicly available to reproduce results and no value is reported in the reference paper for the corresponding metric. The best results are in bold.

Dataset	Algorithm	Macro-F1	Weighted-F1	OA
CICMalDroid20	VINCENT	**0.841**	**0.871**	**0.872**
	ROULETTE	0.536	0.576	0.606
	RENOIR	0.736	0.739	0.747
CICMalMem22	VINCENT	**0.739**	**0.825**	**0.826**
	ROULETTE	0.480	0.647	0.682
	RENOIR	0.483	0.637	0.631
NSL-KDD	VINCENT	**0.669**	**0.818**	**0.829**
	ROULETTE	0.613	0.790	0.815
	RENOIR	0.584	0.733	0.778
	SAE-DNN	–	–	0.821
	MCNN-DFS	–	–	0.814
	DNN+SHAP	–	–	0.803
	ID-GAN	–	–	0.828
UNSW-NB15	VINCENT	**0.474**	**0.781**	**0.784**
	ROULETTE	0.423	0.767	0.763
	RENOIR	0.201	0.590	0.537
	MCNN-DFS	–	–	0.685
	TCN-Attention	–	–	0.724

5.3. Competitor analysis

We compare the accuracy performance of VINCENT to that of several, multi-class, cyber-threat detection methods reported in the recent literature. The selected competitors differ in the deep neural network architecture tested. We considered:

- ROULETTE (Andresini et al., 2022), which combined CNN and Attention.
- RENOIR (Andresini et al., 2021b), which combined Autoencoder and Triplet Network.
- SAE-DNN (Tang et al., 2020), which combined Autoencoder and MLP.
- MCNN-DFS (Al-Turaiki and Altwaijry, 2021), which combined Feature Engineering and CNN.
- DNN+SHAP (Wang et al., 2020), which combined DNN and SHAP Explainer.
- ID-GAN (Yin et al., 2020), which combined GAN and MLP.
- TCN-Attention (Zhao et al., 2022), which combined CNN and Attention.

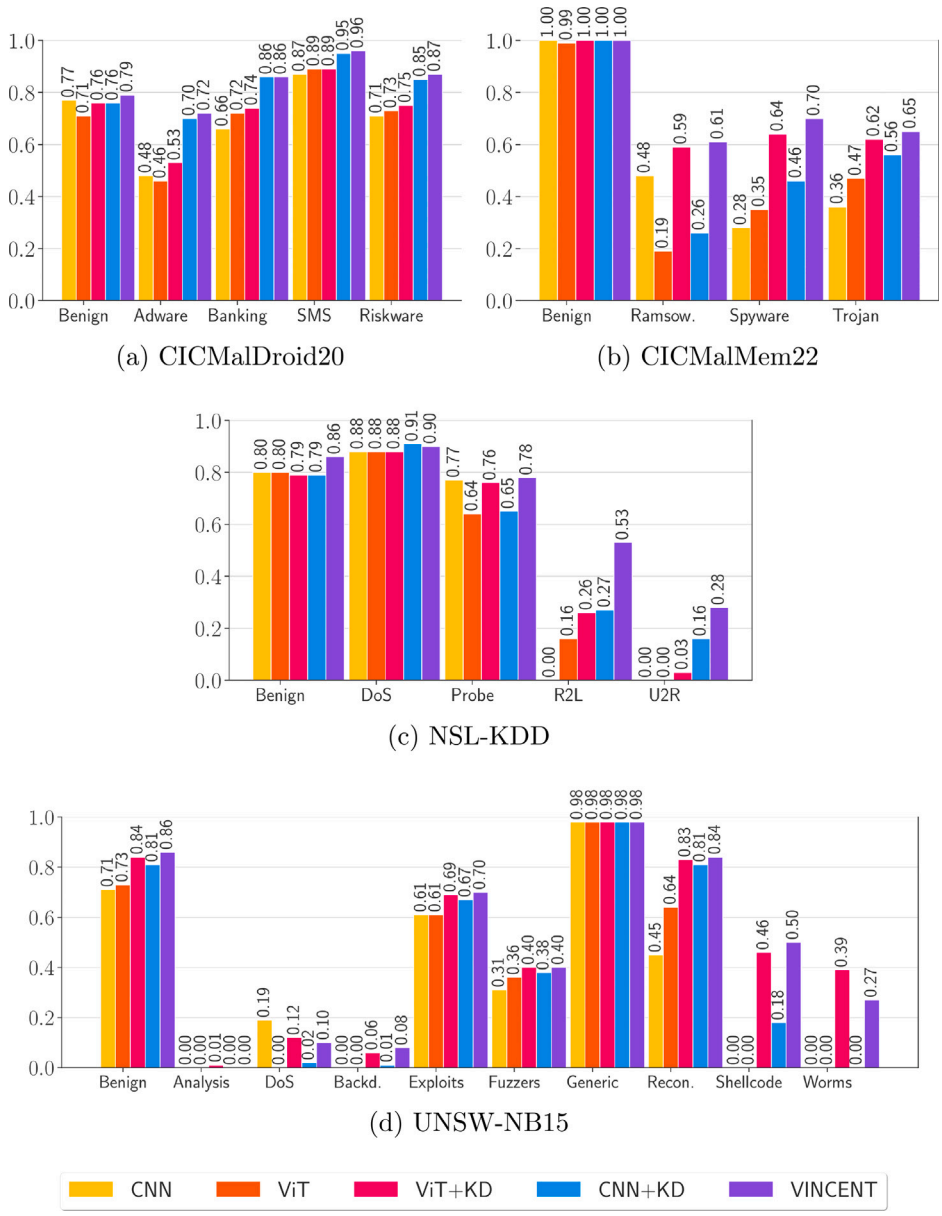


Fig. 4. F1 score, computed for each class, of CNN, ViT, ViT+KD, CNN+KD and VINCENT.

The code of ROULETTE[7] and RENOIR[8] are publicly available. So both these methods were run on all datasets to collect Macro-F1, Weighted-F1 and OA metrics. The code of the remaining competitors is not publicly available for performing the experiments. Therefore, the accuracy performance results of these competitors are taken from the reference papers for the datasets where the methods are evaluated by their authors. However, the comparison is safe as all methods were tested on the multi-class problem of the same training and testing sets described in Section 4.1. The methods that integrate the attention mechanism (Tang et al., 2020; Zhao et al., 2022; Andresini et al., 2022) are the closest to VINCENT. The method proposed by Wang et al. (2020) also adopts an explanation mechanism. However, it uses a post-hoc explainer that explains decisions with no effect on the overall accuracy of decisions.

Table 5 reports the Macro-F1, Weighted-F1 and OA of the compared methods. The results show that VINCENT outperforms its competitors. Notably, the closest OA is achieved with ID-GAN in NSL-KDD. This competitor resorts to generative adversarial learning, which may aid in gaining accuracy in unseen subcategories of intrusions which appear in the testing set of NSL-KDD. This outcome suggests that a future direction of this research may be the exploration of the possible achievements of framing the KD strategy in the realm of adversarial learning.

6. Visual signature analysis

This analysis was performed to explore the properties of visual signatures of training samples belonging to different class types, which are constructed by taking advantage of the explanation information synthesized through the ViT in the proposed KD architecture. For each dataset, and for each class, we constructed the visual signature of the class type, by computing the pixel-wise average of the attention-based images of the training samples belonging to the class considered. The

[7] https://github.com/gsndr/ROULETTE.
[8] https://github.com/gsndr/RENOIR.

Table 6
Average inertia of the visual signatures extracted by considering the imagery configurations: ORIGINAL and ATTENTION. The visual signatures are extracted from the training set. The inertia of the visual signatures is computed on the samples of both the training set and the testing set. The best results are in bold.

Dataset	Training set		Testing set	
	ORIGINAL	ATTENTION	ORIGINAL	ATTENTION
CICMalDroid20	1695.75	**1373.34**	1683.77	**1364.14**
CICMalMem22	577.88	**332.35**	578.37	**332.64**
NSL-KDD	819.14	**667.62**	2954.40	**2387.91**
UNSW-NB15	396.83	**48.33**	393.18	**45.15**

attention-based images are the ones obtained by multiplying the maps of attention extracted through the roll-out method after the pre-training of the ViT architecture and the original images used to pre-train the ViT. These visual signatures, denoted as ATTENTION, were computed by leveraging the explanation information. On the other hand, for each class type, we also computed the pixel-wise average of all the original images associated with the training samples belonging to the class considered. These visual signatures are denoted as ORIGINAL.

In this study, we measured the inertia of the visual signatures extracted by using both ORIGINAL and ATTENTION for the distinct class types. In particular, the inertia was measured as the squared Euclidean distance computed for each imagery representation of a sample to the visual signature of its ground truth class. The lower the inertia, the greater the ability of the visual signature to reveal the features of the signature that may foster decisions correctly produced for either the benign behavior or the malicious behavior of the same class type.

We measured the inertia of the visual signatures of the distinct classes on both the training set and the testing set of each dataset. Table 6 collects the average inertia measured on the visual signatures computed with both ORIGINAL and ATTENTION for all class types of the problems considered. The results show that the visual signatures constructed with ATTENTION always achieve lower average inertia than their counterparts constructed with ORIGINAL. This suggests that the explanation information distilled in the KD architecture of VINCENT actually contributes to synthesizing a better visual signature of each class type, by facilitating the same decision process for samples in the same class. This is an explanation of the higher accuracy achieved by distilling the explanation information of the ViT teacher to the CNN student in the KD architecture of VINCENT. Notably, similar values of average inertia are computed on both the training set and the testing set, except for NSL-KDD. This is due to the fact that the testing set of NSL-KDD contains sub-categories of intrusions that are unseen in the training set. This condition simulates the zero-day attack scenario in NSL-KDD. In any case, the testing samples, recording intrusions that were unseen during the training stage, are better described by intrusion visual signatures, computed by accounting for the explanation information than by intrusion visual signatures, computed by neglecting the explanation information distilled in the training stage. These considerations are also supported by the analysis of the inertia scores, computed for each class, in the testing sets of the four datasets of this study. These detailed results, reported in Fig. 5, show that the visual signatures computed with ATTENTION achieve lower inertia than their counterparts computed with ORIGINAL in all the classes.

At the end of this analysis, Figs. 6 and 7 show the visual signatures extracted in the configuration ATTENTION on the five classes recorded in CICMalDroid20 and NSL-KDD, respectively. In addition, Figs. 8 and 9 show the maps of the average Euclidean distance, computed pixel-wise, between the visual signature of each class type and the visual signatures of the remaining class types in both CICMalDroid20 and NSL-KDD. In CICMalDroid20, the distance results show that the system call *statfs64*, which is used to obtain information on the mounted file system, takes on distant values in all five classes. This system call assumes, in fact, a different value in the visual signatures of the five classes. The system call *pwrite64*, which writes bytes from buf to a file at an offset, takes on distant values on the classes: Benign, Adware, SMS and Riskware, while the *fdatasync* call, that is used for selective synchronization, has a discriminating role in the classes: Adware, SMS and Riskware. This information is coherent with the study of Ham et al. (2013) that already identified the joint presence of system calls like *statfs64*, *pwrite* and *fdatasync* as a sign of malicious Android apps. More recently, the study of Chew et al. (2020) suggested monitoring *pwrite64*, to identify signs of malicious behaviors.

In NSL-KDD, distance results show that the feature *dst_host_count*, which counts "the number of connections having the same destination host", takes on distant values in all the classes. In particular, the highest distance value in this feature (i.e., 225.157) is achieved for the DoS class. We note that *dst_host_count* is recognized as important to identify DoS attacks also in Andresini et al. (2022), since DoS attacks flood a server by sending numerous service requests to make the server unavailable. In fact, *dst_host_count* assumes a higher value in the visual signature of the DoS class than in the visual signatures of the remaining classes (Fig. 7). By continuing this analysis on the visual signature of the DoS class, we note that the *serror_rate* feature takes a high distance value in the DoS class. This feature, which indicates "the percentage of connections that have SYN errors", is recognized as a relevant feature to detect DoS attacks also in Wang et al. (2020). In particular, this study uses the post-hoc SHAP explainer to show that high values of *serror_rate* help to discriminate DoS attacks. The visual signature of the DoS class, reported in Fig. 7(b), highlights the same pattern by showing a high value of *serror_rate*. In fact, in SYN flood attacks (a subcategory of DoS), targets are flooded with several SYN requests without ACK response, with the aim of consuming the target resource and making a server unavailable to legitimate traffic. By considering the visual signature of the U2R class, we note that the feature *root_shell* takes a high distance value in this class. This feature, which explores if "the root shell is obtained", is also identified as an important feature for recognizing U2R attacks in Andresini et al. (2022). This is expected as a U2R attacker commonly tries to gain full access to the system by obtaining access to a shell with administrator privilege (i.e., root shell). Final considerations concern the *service_ftp* feature, that takes on a distant value in the R2L class. This feature is also identified in Wang et al. (2020) as one of the top-twenty important features, calculated with SHAP, to detect *R2L* attacks. An R2L attack occurs when the attacker is able to send packets to a machine without gaining access to the local machine. The NSL-KDD dataset contains two subcategories of R2L attacks, called Warez Client and Warez Master. Dey et al. (2017) highlight that both Warez Client and Warez Master attacks exploit vulnerabilities of the "anonymous" login FTP. The conclusions drawn in Dey et al. (2017) are supported by the visual signature of R2L produced in our study. In fact, *is_guest_login* also takes on a high distance value in the R2L class, and both *service_ftp* and *is_guest_login* assume higher values in the visual signature of R2L than in the visual signatures of the remaining classes (Fig. 7).

7. Conclusion

In this paper, we present VINCENT, that is a deep neural model for multi-class, cyber-threat detection. The proposed method leverages a KD strategy that trains a teacher–student architecture with the knowledge distilled from a complex teacher to guide the knowledge of a tiny student. In VINCENT the teacher model consists of a ViT that takes an imagery representation of cyber-data features as input. The student model consists of a tiny CNN that takes an attention-based representation of cyber-data images as input. These attention-based images are produced from the original cyber-data images by accounting for the attention maps learned with the ViT.

Extensive experiments were performed using four benchmark datasets of malware applications and network flow attacks. The results

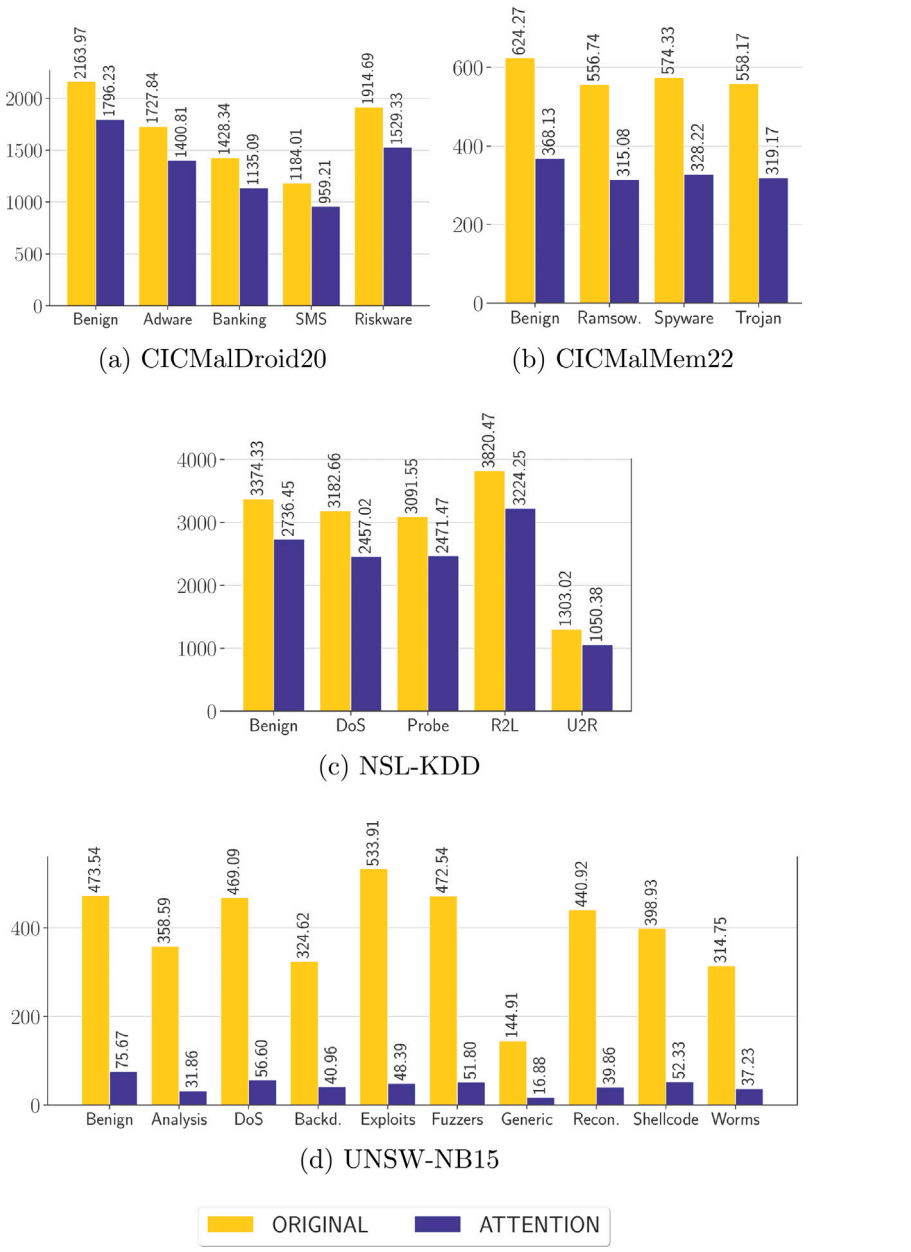


Fig. 5. inertia score, computed for each class, for the visual signatures of all the classes. The inertia is computed for both ORIGINAL and ATTENTION on the testing set.

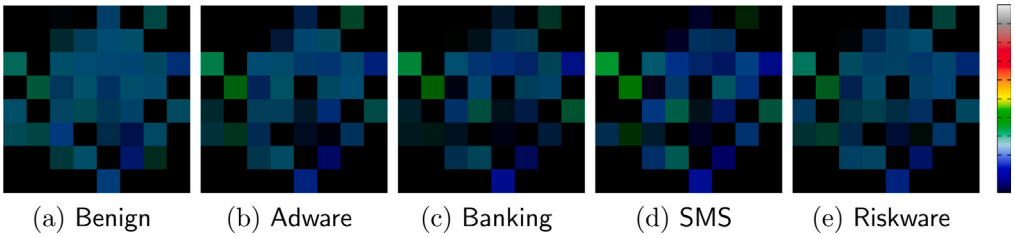


Fig. 6. Attention-based visual signatures extracted from the training set of CICMalDroid20.

provide evidence of the effectiveness of our idea of performing the KD strategy by accounting for the attention maps produced through the self-attention mechanism of the ViT teacher, to generate the input images to train the CNN student. Specifically, the results show that the KD model trained with VINCENT gains accuracy compared to the single models trained with either the ViT architecture or the CNN architecture, separately. In addition, the model trained with VINCENT outperforms the model trained with the traditional KD strategy using the original imagery representation of cyber-data to train both the teacher and the student. Finally, VINCENT outperforms several state-of-the-art methods recently defined for cyber-threat detection problems, comprising methods using attention mechanisms. On

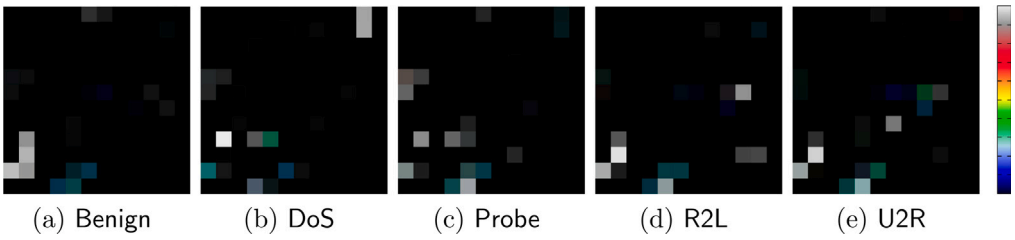


Fig. 7. Attention-based visual signatures extracted from the training set of NSL-KDD.

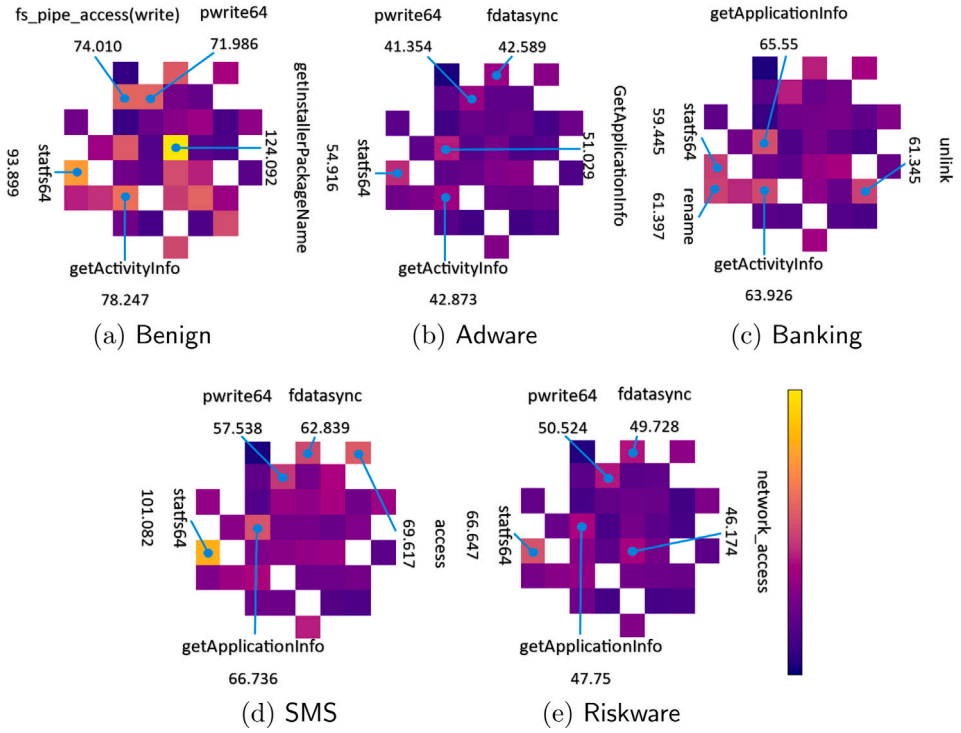


Fig. 8. Average Euclidean distance computed pixel-wise between the attention-based visual signature of a class type and the attention-based visual signatures of the remaining class types in CICMalDroid20. For each class type, the pixels associated with the top-five features that measure the greatest distance are highlighted.

the other hand, the attention maps, disclosed by the ViT, generate cyber-threat signature knowledge that discloses useful information on features that mainly help to recognize specific threat categories.

One limitation of the proposed method is the absence of an explicit strategy to deal with rare classes. In any case, the experimental results proved that the idea of leveraging the attention-based representation of imagery cyber-data helps the final model to disentangle better samples belonging to the different threat categories and benign samples. In particular, the F1 score computed for each class shows that VINCENT is more accurate than its baselines in recognizing rare cyber-threats. Another limitation of VINCENT is that it performs the learning stage in a batch fashion, without considering that characteristics of cyber-threats may evolve over time. For this reason, as future work, we plan to integrate the proposed approach with a concept drift detection mechanism to fit VINCENT into an evolving strategy that is able to continuously update the classification model.

CRediT authorship contribution statement

Luca De Rose: Conceptualization, Data curation, Investigation, Software, Validation, Writing – original draft, Writing – review & editing. **Giuseppina Andresini:** Conceptualization, Data curation, Investigation, Methodology, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. **Annalisa Appice:** Conceptualization, Investigation, Methodology, Project administration, Validation, Writing – original draft, Writing – review & editing. **Donato Malerba:** Conceptualization, Methodology, Writing – review & editing.

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

Data are benchmark datasets available online.

Acknowledgments

Giuseppina Andresini is supported by the project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI, under the NRRP MUR program funded by the European Union - NextGenerationEU. Annalisa Appice and Donato Malerba are partially supported by project SERICS (PE00000014) under the NRRP MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU. The research objectives of this paper are in partial fulfillment of the project AI-CREED (CUP H93C23000880005). The authors wish to thank Lynn Rudd for her help in reading the manuscript and Vincenzo Pasquadibisceglie for his helpful discussion on Vision Transformers. The authors wish to thank anonymous reviewers for the helpful suggestions they provided to improve the paper.

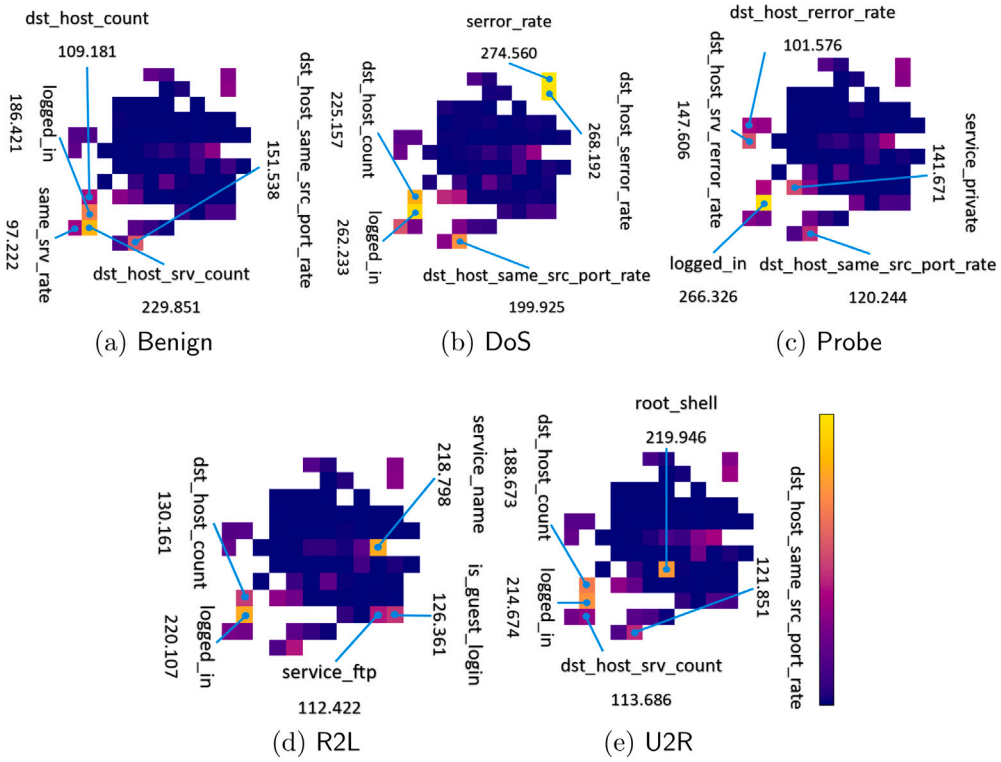


Fig. 9. Average Euclidean distance computed pixel-wise between the attention-based visual signature of a class type and the attention-based visual signatures of the remaining class types in NSL-KDD. For each class type, the pixels associated with the top-five features that measure the greatest distance are highlighted.

References

Abnar, S., Zuidema, W.H., 2020. Quantifying attention flow in transformers. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. ACL 2020, Association for Computational Linguistics, pp. 4190–4197. http://dx.doi.org/10.18653/v1/2020.acl-main.385.

Al-Turaiki, I., Altwaijry, N., 2021. A convolutional neural network for improved anomaly-based network intrusion detection. Big Data 9, 233–252. http://dx.doi.org/10.1089/big.2020.0263.

Andresini, G., Appice, A., Caforio, P.F., Malerba, D., 2021a. Improving Cyber-Threat Detection by Moving the Boundary Around the Normal Samples. Springer International Publishing, Cham, pp. 105–127. http://dx.doi.org/10.1007/978-3-030-57024-8_5.

Andresini, G., Appice, A., Caforio, F.P., Malerba, D., Vessio, G., 2022. ROULETTE: A neural attention multi-output model for explainable network intrusion detection. Expert Syst. Appl. 201, 117144. http://dx.doi.org/10.1016/j.eswa.2022.117144.

Andresini, G., Appice, A., Malerba, D., 2021b. Autoencoder-based deep metric learning for network intrusion detection. Inf. Sci. 569, 706–727. http://dx.doi.org/10.1016/j.ins.2021.05.016.

Andresini, G., Appice, A., Rose, L.D., Malerba, D., 2021c. GAN augmentation to deal with imbalance in imaging-based intrusion detection. Future Gener. Comput. Syst. 123, 108–127. http://dx.doi.org/10.1016/j.future.2021.04.017.

Andresini, G., Pendlebury, F., Pierazzi, F., Loglisci, C., Appice, A., Cavallaro, L., 2021d. INSOMNIA: towards concept-drift robustness in network intrusion detection. In: 14th ACM Workshop on Artificial Intelligence and Security. ACM, pp. 111–122. http://dx.doi.org/10.1145/3474369.3486864.

Aydın, H., Orman, Z., Aydın, M.A., 2022. A long short-term memory (lstm)-based distributed denial of service (ddos) detection and defense system design in public cloud network environment. Comput. Secur. 118, 102725. http://dx.doi.org/10.1016/j.cose.2022.102725.

Bergstra, J., Yamins, D., Cox, D.D., 2013. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In: Icml. pp. I–115–I–123. http://dx.doi.org/10.5555/3042817.3042832.

Caforio, F.P., Andresini, G., Vessio, G., Appice, A., Malerba, D., 2021. Leveraging grad-cam to improve the accuracy of network intrusion detection systems. In: 24th Conference on Discovery Science. DS 2021, Springer, pp. 385–400. http://dx.doi.org/10.1007/978-3-030-88942-5_30.

Carrier, T., Victor, P., Tekeoglu, A., Lashkari, A.H., 2022. Detecting obfuscated malware using memory feature engineering. In: Proceedings of the 8th International Conference on Information Systems Security and Privacy. ICISSP 2022, SCITEPRESS, pp. 177–188. http://dx.doi.org/10.5220/0010908200003120.

Chew, C., Kumar, V., Patros, P., Malik, R., 2020. Escapade: Encryption-Type-Ransomeware: System Call Based Pattern Detection. pp. 388–407. http://dx.doi.org/10.1007/978-3-030-65745-1_23.

Dey, D., Dinda, A., Kundapur, P., Smitha, R., 2017. Warezmaster and warezclient: An implementation of ftp based r2l attacks. pp. 1–6. http://dx.doi.org/10.1109/ICCCNT.2017.8203964.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In: 9th International Conference on Learning Representations. ICLR 2021, http://dx.doi.org/10.5220/0010908200003120.

Gou, J., Yu, B., Maybank, S.J., Tao, D., 2021. Knowledge distillation: A survey. Int. J. Comput. Vis. 129, 1789–1819. http://dx.doi.org/10.1007/s11263-021-01453-z.

Ham, Y.J., Moon, D., Lee, H.W., Lim, J.D., Kim, J.N., 2013. Activation pattern analysis on malicious android mobile applications. In: 1st International Conference on Artificial Intelligence, Modelling and Simulation. AIMS 2013, IEEE Computer Society, http://dx.doi.org/10.5555/2605697.2605748.

Han, K., Wang, Y., Chen, H., Chen, X., Guo, J., Liu, Z., Tang, Y., Xiao, A., Xu, C., Xu, Y., Yang, Z., Zhang, Y., Tao, D., 2023. A survey on vision transformer. IEEE Trans. Pattern Anal. Mach. Intell. 45, 87–110. http://dx.doi.org/10.1109/TPAMI.2022.3152247.

Hinton, G.E., Vinyals, O., Dean, J., 2015. Distilling the knowledge in a neural network. CoRR abs/1503.02531.

Ho, C.M.K., Yow, K.C., Zhu, Z., Aravamuthan, S., 2022. Network intrusion detection via flow-to-image conversion and vision transformer classification. IEEE Access 10, 97780–97793. http://dx.doi.org/10.1109/ACCESS.2022.3200034.

Jullian, O., Otero, B., Rodríguez, E., Gutiérrez, N., Antona, H., Canal, R., 2023. Deep-learning based detection for cyber-attacks in iot networks: A distributed attack detection framework. J. Netw. Syst. Manag. 31, 33. http://dx.doi.org/10.1007/s10922-023-09722-7.

Keshk, M., Koroniotis, N., Pham, N., Moustafa, N., Turnbull, B., Zomaya, A.Y., 2023. An explainable deep learning-enabled intrusion detection framework in iot networks. Inform. Sci. 639, 119000. http://dx.doi.org/10.1016/j.ins.2023.119000.

Li, K., Yu, R., Wang, Z., Yuan, L., Song, G., Chen, J., 2022. Locality guidance for improving vision transformers on tiny datasets. In: Computer Vision – ECCV 2022. Springer Nature Switzerland, Cham, pp. 110–127. http://dx.doi.org/10.1007/978-3-031-20053-3_7.

Liu, C., Liu, Y., Yan, Y., Wang, J., 2020. An intrusion detection model with hierarchical attention mechanism. IEEE Access 8, 67542–67554. http://dx.doi.org/10.1109/ACCESS.2020.2983568.

Mahdavifar, S., Alhadidi, D., Ghorbani, A.A., 2022. Effective and efficient hybrid android malware classification using pseudo-label stacked auto-encoder. J. Netw. Syst. Manag. 30, 22. http://dx.doi.org/10.1007/s10922-021-09634-4.

Moustafa, N., Slay, J., 2015. Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: Military Communications and Information Systems Conference. MilCIS 2015, pp. 1–6. http://dx.doi.org/10.1109/MilCIS.2015.7348942.

Pasquadibisceglie, V., Appice, A., Castellano, G., Malerba, D., 2020. Predictive process mining meets computer vision. In: Business Process Management Forum. BPM 2020, Springer International Publishing, Cham, pp. 176–192. http://dx.doi.org/10.1007/978-3-030-58638-6_11.

Pasquadibisceglie, V., Appice, A., Castellano, G., Malerba, D., 2023. JARVIS: Joining adversarial training with vision transformers in next-activity prediction. IEEE Trans. Serv. Comput. 1–14. http://dx.doi.org/10.1109/TSC.2023.3331020.

Pawlicki, M., Choraś, M., Kozik, R., Hołubowicz, W., 2020. On the impact of network data balancing in cybersecurity applications. In: Computational Science – ICCS 2020. pp. 196–210. http://dx.doi.org/10.1007/978-3-030-50423-6_15.

Rabieinejad, E., Yazdinejad, A., Parizi, R.M., Dehghantanha, A., 2023. Generative adversarial networks for cyber threat hunting in ethereum blockchain. Distrib. Ledger Technol. http://dx.doi.org/10.1145/3584666.

Seneviratne, S., Shariffdeen, R., Rasnayaka, S., Kasthuriarachchi, N., 2022. Self-supervised vision transformers for malware detection. IEEE Access 10, 103121–103135. http://dx.doi.org/10.1109/ACCESS.2022.3206445.

Sharma, O., Sharma, A., Kalia, A., 2023. Windows and iot malware visualization and classification with deep CNN and xception CNN using markov images. J. Intell. Inf. Syst. 60, 349–375. http://dx.doi.org/10.1007/s10844-022-00734-4.

Srikanth Yadav, M., Kalpana, R., 2022. Recurrent nonsymmetric deep auto encoder approach for network intrusion detection system. Meas.: Sens. 24, 100527. http://dx.doi.org/10.1016/j.measen.2022.100527.

Tang, C., Luktarhan, N., Zhao, Y., 2020. SAAE-DNN: Deep learning method on intrusion detection. Symmetry 12, 1–20. http://dx.doi.org/10.3390/sym12101695.

Tavallaee, M., Bagheri, E., Lu, W., Ghorbani, A.A., 2009. A detailed analysis of the KDD CUP 99 data set. In: IEEE Symposium on Computational Intelligence for Security and Defense Applications. CISDA 2009, IEEE, pp. 1–6. http://dx.doi.org/10.1109/CISDA.2009.5356528.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. Curran Associates Inc., pp. 6000–6010. http://dx.doi.org/10.5555/3295222.3295349.

Wang, Z., Li, Z., He, D., Chan, S., 2022. A lightweight approach for network intrusion detection in industrial cyber–physical systems based on knowledge distillation and deep metric learning. Expert Syst. Appl. 206, 117671. http://dx.doi.org/10.1016/j.eswa.2022.117671.

Wang, M., Zheng, K., Yang, Y., Wang, X., 2020. An explainable machine learning framework for intrusion detection systems. IEEE Access 8, 73127–73141. http://dx.doi.org/10.1109/ACCESS.2020.2988359.

Xia, M., Xu, Z., Zhu, H., 2022. A novel knowledge distillation framework with intermediate loss for android malware detection. In: 2022 IEEE Asia-Pacific Conference on Computer Science and Data Engineering. CSDE, pp. 1–6. http://dx.doi.org/10.1109/CSDE56538.2022.10089266.

Yin, C., Zhu, Y., Liu, S., Fei, J., Zhang, H., 2020. Enhancing network intrusion detection classifiers using supervised adversarial training. J. Supercomput. 76, 6690–6719. http://dx.doi.org/10.1007/s11227-019-03092-1.

Zhao, P., Fan, Z., Cao, Z., Li, X., 2022. Intrusion detection model using temporal convolutional network blend into attention mechanism. Int. J. Inf. Secur. Priv. 16, 1–20. http://dx.doi.org/10.4018/IJISP.290832.

Luca De Rose graduated in "Cybersecurity" from the University of Bari Aldo Moro, Italy, in July 2020, discussing a Laurea thesis on Intrusion Detection Systems. He is a Ph.D. student in the Department of Computer Science at the University of Bari Aldo Moro. He participated at the CyberChallenge.IT 2019 in the Department of Computer Science of the University of Bari Aldo Moro. He has been part of the local team of the University of Bari Aldo Moro that participated in the National competition of CyberChallenge.IT in June 2019.

Giuseppina Andresini is an Assistant Professor (non-tenure) at the Department of Computer Science, the University of Bari Aldo Moro. She received a Ph.D. in Computer Science, University of Bari Aldo Moro, Italy, in March 2022. Her current research interests mainly concern deep learning, XAI and adversarial learning with applications in cybersecurity and remote sensing. On these topics, she has published papers in international journals and conferences. She is on the program committee of top conferences (e.g., ECML-PKDD, ICDE, IJCAI, Discover Science, ECAI) and a member of the editorial board of "Machine Learning" Journal.

Annalisa Appice is a Full Professor at the Department of Computer Science, University of Bari Aldo Moro, Italy. She received a Ph.D. in Computer Science in the University of Bari Aldo Moro. Her current research interests include data mining with spatio-temporal data, data streams and event logs with applications to remote sensing, process mining and cybersecurity. She has published more than 190 papers in international journals and conferences. She has been Program co-Chair of ECML-PKDD 2015, ISMIS 2017 and DS 2020. She was Journal Track co-Chair of ECML-PKDD 2021. She has participated in the organization of several workshops in machine learning, process mining and cybersecurity. She is a member of the editorial board of MACH, DAMI, EAAI and JIIS. She is responsible for the local research unit of University of Bari Aldo Moro in the EU project SWIFTT.

Donato Malerba received the M.Sc. degree in Computer Science from the University of Bari, Italy, in 1987. He is a Full Professor in the Department of Computer Science, University of Bari Aldo Moro, Italy. He is a Senior IEEE member. He has authored and co-authored more than 330 papers in international journals and conferences. His research interests include machine learning, data mining, big data analytics and their applications. He has been responsible for the local research unit of several European and national projects, and received an IBM Faculty Award in 2004. He was Program (co-)Chair of IEA-AIE 2005, ISMIS 2006, SEBD 2007, ECMLPKDD 2011 and the General Chair of ALT/DS 2016. He is the scientific responsible of Spoke 6- Symbiotic AI of the Future AI Research (FAIR) project. He is on the editorial board of several international journals.