Unsupervised Network Anomaly Detection with Autoencoders and Traffic Images

Michael Neri* , Sara Baldoni†
*Faculty of Inform. Tech. and Com. Sciences, Tampere University, Tampere, Finland, michael.neri@tuni.fi
†Dept. of Information Engineering, University of Padova, Padua, Italy, sara.baldoni@unipd.it

***Abstract*—Due to the recent increase in the number of connected devices, the need to promptly detect security issues is emerging. Moreover, the high number of communication flows creates the necessity of processing huge amounts of data. Furthermore, the connected devices are heterogeneous in nature, having different computational capacities. For this reason, in this work we propose an image-based representation of network traffic which allows to realize a compact summary of the current network conditions with 1-second time windows. The proposed representation highlights the presence of anomalies thus reducing the need for complex processing architectures. Finally, we present an unsupervised learning approach which effectively detects the presence of anomalies. The code and the dataset are available at https://github.com/michaelneri/image-based-network-traffic-anomaly-detection.**

***Index Terms*—Unsupervised anomaly detection, Image-based network representation, Autoencoder.**

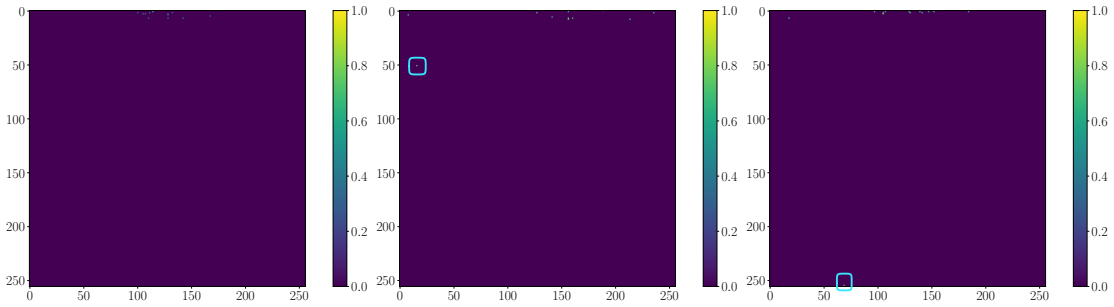
I. Introduction

The current diffusion of Internet-related technologies is leading to an unprecedented connectivity among heterogeneous devices with varied computational capabilities. The extensive use of computer networks creates security risks that can lead to misconduct and substantial harm. These threats are dynamic and prone to evolve into unknown forms [1]. To properly react to this danger, a prompt detection of anomalous network behaviors is needed. An *anomalous* event can be defined as a network pattern that diverges from the expected *normal* behavior [2]. The design of anomaly detection techniques is challenging for several reasons. First, the wide diffusion of connected devices causes a relevant increase in the number of traffic flows, making the real-time detection of anomalies a demanding task. Moreover, due to the inherent disparity between the amount of normal and anomalous data flows, the adoption of supervised learning methods is hindered. Furthermore, these techniques often fail in accurately identifying unfamiliar abnormal behaviors [3], [4]. Consequently, the exploration of unsupervised learning techniques has emerged as a prominent direction for addressing anomaly detection within telecommunication networks. In the context of unsupervised techniques, a key task consists in modeling the normal state of a telecommunication network. To this end, different types of predictors such as traffic usage, protocols, and number of flows can be employed [4], [5]. Due to the high number of predictors, dimensionality reduction techniques, such as Principal Component Analysis (PCA) [6], have been employed for analyzing network traffic [7]–[9]. Recently, deep learning in anomaly detection has represented an important shift from traditional PCA-based methods. Deep learning approaches can capture non-linear relationships and high-level abstractions, offering enhanced detection capabilities in diverse and complex scenarios [11]. However, Autoencoder (AE)-based anomaly detectors trained on normal traffic data are prone to generalization problems [11]. In fact, even unseen abnormal patterns can be correctly retrieved by reconstruction-based approaches. To mitigate this problem, One-Class Support Vector Machines (SVM), Variational Autoencoder (VAE), and Generative Adversarial Network (GAN) have been proposed in literature [12]–[15].

Instead of realizing a complex learning-based detector, we propose to address the network anomaly detection issue from a different perspective. One of the limitations of the existing methods is that they directly process network flows and traffic features without performing a pre-processing step for highlighting hidden traffic peculiarities. This work, on the contrary, proposes an image-based representation of network traffic. The proper definition of the image representation provides a compact picture of the current condition of the monitored network, reducing the complexity of the learning architecture. This direction has been partially investigated in [16], where a 2D representation of network activity for Cyber-Physical Systems (CPSs) has been devised. However, the defined 2D representation is sparse thus failing in simplifying the processing pipeline. Indeed, in [17] a complex VAE has been implemented for detecting anomalies from the representation proposed in [16] using reconstruction-based errors for the detection. In this work, we leverage the representation used in [16], [17] reducing the image sparsity and making the network patterns more evident. This allows to reduce the complexity of the learning architecture. To demonstrate this assertion, we test two reconstruction-based anomaly detectors: a lightweight VAE and a vanilla AE.

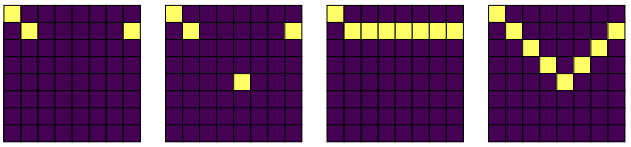
The contributions of this work can be summarized as follows: i) a new image-based representation of network traffic that highlights the presence of attacks, thus requiring a low-complexity anomaly detector; ii) the quantitative comparison between different types of 2D representations; iii) the comparison between anomaly detectors with different complexities.

The research presented in this paper was partially funded by the project "ISEEYOO: AI-based Network Anomaly Detection for CPS exploiting 2D data representation" within the University of Padova funding framework "SEED research grants."



(a) Normal (b) DoS (c) Scan

Fig. 1. Examples of $\mathcal{I}$ for normal and anomalous traffic. The light-blue square indicates the pixels corresponding to anomalous traffic patterns. The matrices are obtained for 1-second time windows from the dataset presented in [10].



(a) Normal, $\mathcal{I}$ (b) Anomaly, $\mathcal{I}$ (c) Normal, $\mathcal{I}_C$ (d) Anomaly, $\mathcal{I}_C$

Fig. 2. Examples of $\mathcal{I}$ and $\mathcal{I}_C$ for normal and anomalous traffic in a limit case scenario.

II. Anomaly Detection Method

In this section an anomaly detector which exploits an image-based traffic representation is described. In more detail, a 2D matrix containing network traffic information has been defined and an AE has been designed and trained to reconstruct normal traffic images in an unsupervised fashion. By doing so, anomalous traffic representations are poorly retrieved by the AE, yielding high reconstruction errors. This property makes the AE suitable for anomaly detection scenarios where labeled anomalous instances may be scarce or unavailable. In fact, the model learns to generalize from the normal data without explicitly receiving information about anomalies. This capability represents the reason for which we opted for an AE-based strategy.

A. Network traffic representation

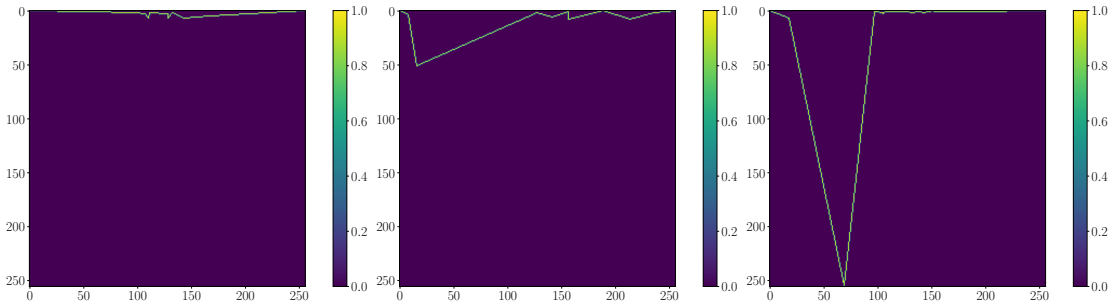
The image-based representation proposed in [16] associates every pixel to a source-destination IP pair, (i, j), and computes every pixel value, $p(i, j)$, as

$$p(i,j) = \frac{\Sigma(i,j) - \mu(i,j)}{\sigma(i,j) + 10^{-4}} \times e^{f(i,j)}, \tag{1}$$

where $\Sigma(i, j)$ is the amount of bytes exchanged between i and j in a time window, $\mu(i, j)$ and $\sigma(i, j)$ are the corresponding mean and standard deviation, and $f(i, j)$ is the number of flows. Although this representation effectively highlights the presence of an attack, it is not suitable for being processed through deep learning algorithms due to the high signal dynamics [17]. Therefore, in this work, we adopt the same mapping used in [17], [18]. Specifically, every column is associated to a node of the monitored network, and the incoming traffic is processed as in Eq. (1) and mapped to a different row through a procedure based on the incoming traffic histogram. The mapping allows to concentrate the normal traffic in the upper area of the image, while moving the attack-related pixels to lower rows (for the mapping details see [18]). In the following this representation will be referred to as $\mathcal{I}$. Fig. 1 depicts an example of $\mathcal{I}$ in normal conditions and in presence of attacks, considering a time window of 1 second. As can be noticed, $\mathcal{I}$ is a sparse matrix and the difference between normal and anomalous images is given by few and isolated points, highlighted in Fig. 1(b) and 1(c). This may result in an ineffective autoencoder training due to the high sparsity of active pixels. To highlight this issue, let us consider the limit case in which a single pixel is modified with respect to the image representing normal traffic, as shown in Fig. 2(a) and 2(b). Under these circumstances, even if the autoencoder provides as output a matrix which is very similar to the normal image (Fig. 2(a)), the reconstruction error based on the comparison with the input anomalous matrix (Fig. 2(b)) would be very small and the anomaly would not be detected. To solve this issue, we introduce a new image-based representation, $\mathcal{I}_C$, which can be obtained by connecting all the active pixels in $\mathcal{I}$. To visualize this phenomenon, we reported the $\mathcal{I}_C$ representations corresponding to Fig. 2(a) and 2(b) in Fig. 2(c) and 2(d). As can be observed, even in the limit case in which the difference between normal and anomalous traffic corresponds to a single pixel modification in $\mathcal{I}$, a relevant difference appears in $\mathcal{I}_C$, thus making the representation less sparse and the training more effective. The images $\mathcal{I}_C$ corresponding to the $\mathcal{I}$ matrices represented in Fig. 1 are reported in Fig. 3. As can be noticed, thanks to the active pixel connection, normal traffic is represented by a signal that oscillates slowly, while the anomalous traffic representation is characterized by spikes. Finally, it is useful to notice that the computational complexity for obtaining $\mathcal{I}_C$ from $\mathcal{I}$ can be deemed negligible.

B. Autoencoders for anomaly detection

The goal of an AE is to learn a compressed representation of the input and then reconstruct the original data from it.



(a) Normal (b) DoS (c) Scan

Fig. 3. Examples of $\mathcal{I}_C$ for normal and anomalous traffic. The matrices are obtained for 1-second time windows from the dataset presented in [10].

Anomalies or outliers in the data can be detected by measuring the difference between the original input and its reconstructed version. Differently, a VAE has also the objective of estimating the true posterior distribution of the latent vectors [19].

Both architectures are composed of symmetrical encoder $E(\cdot)$ and decoder $D(\cdot)$. The former lossy compresses the input representation with height H and width W $\mathcal{I}_C \in \mathbb{Z}_2^{H \times W}$ to a latent vector $\mathbf{z} \in \mathbb{R}^d$, where d is the dimension of the latent space. If the model is a VAE, two latent vectors $\mathbf{z}_\mu, \mathbf{z}_{\log \sigma} \in \mathbb{R}^d$ are estimated. In this case, $\mathbf{z} \in \mathbb{R}^d$ is composed by means of the reparametrization procedure [19]

$$\mathbf{z} = \mathbf{z}_\mu + \mathbf{z}_\sigma + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1). \tag{2}$$

Consequently, the decoder is responsible for upsampling $\mathbf{z}$ to the reconstructed image $\hat{\mathcal{I}_C}$. The proposed VAE and AE architectures are detailed in Tab. I. The encoder $E(\cdot)$ is composed of 3 convolutional blocks, denoted as $\mathrm{ConvBlock}(c_i)$ where c_i is the number of output channels, that extract spatial features from the input image. Each block performs a 2D convolution, a batch normalization, and the Exponential Linear Unit (ELU) activation function. This sequence of operations is employed twice in each block. A $\mathrm{MaxPool}(2, 2)$ function is applied after each of the first two convolutional blocks to downsample the images. A linear projection layer is applied to map the output of the last convolutional block into a feature tensor having 1 channel. This operation is carried out by a 1×1 convolutional layer. Finally, two fully connected layers, with n_{fc} and d neurons, are responsible for mapping the features into the two latent vectors for means and log variance. Symmetrically, 3 transposed convolutional blocks, indicated as $\mathrm{TranConvBlock}(c_i)$, constitute the decoder $D(\cdot)$. After the first two transposed convolutional blocks, two $\mathrm{UpSample}(2, 2)$ operations with nearest neighbour interpolation are performed to retrieve the original shape of the image. In this work, the number of channels are $\mathbf{c} = [c_1, c_2, c_3] = [16, 32, 64]$, kernel sizes are 5×5, n_{fc} and d are 128 and 64, respectively. This configuration has been tuned by means of hyperparameters optimization algorithms, i.e., grid search.

Regarding the training procedure, the objective of both architectures is to jointly minimize the error of the encoding-decoding procedure whereas the VAE additionally aims to reduce the Evidence Lower Bound (ELBO) [19]. The reconstruction error is calculated by comparing the original input and its reconstructed output using a suitable distance or similarity measure. Due to the sparse nature of the image representation, two distance losses are tested. First, the Binary Cross-Entropy (BCE) is employed as a pixel-wise statistical distance between the input image and its reconstruction. Then, as the images mostly contain few active pixels, a weighted BCE loss function is introduced to penalize wrongly reconstructed active pixels with $\mathbf{w} = [w_0, w_1]$, which is the vector that penalizes inactive and active pixels.

Regarding the VAE, thanks to the reparameterization in Eq. (2), it is possible to derive the Kullback-Lieber divergence loss as $\mathcal{L}_{KL}(\mathbf{z}, \mathcal{I}_C) = D_{KL}(q(\mathbf{z}|\mathcal{I}_C)||p(\mathbf{z}))$, where $q(\cdot)$ and $p(\cdot)$ are the learned probability distribution over the latent space and a predefined prior distribution, which is a standard normal distribution, respectively. Finally, the total ELBO loss employed for training the architecture is $\mathcal{L}_{ELBO}(\mathbf{z}, \mathcal{I}_C, \hat{\mathcal{I}_C}) = \mathcal{L}_{wBCE}(\mathcal{I}_C, \hat{\mathcal{I}_C}) - \beta \mathcal{L}_{KL}(\mathbf{z}, \mathcal{I}_C)$, where $\beta = 0.00005$ is set for balancing the two losses' magnitude.

TABLE I
DESCRIPTION OF BOTH AE AND VAE STRUCTURES.

Input: input image $\mathcal{I}_C \in \mathbb{Z}_2^{W \times W}$	**Input**: latent vector $\mathbf{z} \in \mathbb{R}^d$
Encoder $E(\cdot)$	**Decoder** $D(\cdot)$
$\mathrm{ConvBlock}(c_1)$	$\mathrm{FC}(n_{fc})$
$\mathrm{MaxPool}(2, 2)$	ELU
$\mathrm{ConvBlock}(c_2)$	$\mathrm{FC}(n_{fc})$
$\mathrm{MaxPool}(2, 2)$	Unflatten to 3D tensor
$\mathrm{ConvBlock}(c_3)$	Projection to C_3 channels
Projection to 1 channel	$\mathrm{TranConvBlock}(c_3)$
Flatten to 1D vector	$\mathrm{UpSample}(2, 2)$
$\mathrm{FC}(n_{fc})$	$\mathrm{TranConvBlock}(c_2)$
ELU	$\mathrm{UpSample}(2, 2)$
$\mathrm{FC}(d)$	$\mathrm{TranConvBlock}(c_1)$
if VAE **then** Reparameterization (Eq. (2))	Projection to 1 channel
Output: latent vector $\mathbf{z} \in \mathbb{R}^d$	**Output**: reconstructed image $\hat{\mathcal{I}_C} \in \mathbb{R}^{H \times W}$

III. RESULTS

The proposed image representation and anomaly detector are tested on the UGR'16 dataset [10]. The performance of the learning architectures trained on the proposed 2D representation is assessed on the test set by means of Information Retrieval (IR) metrics such as precision, recall, accuracy,

and F1 score. Moreover, a quantitative analysis is performed, varying the deep learning architecture, the 2D representation, and the loss function during the training phase.

A. Dataset

The UGR'16 dataset consists of two subsets: calibration and test. The calibration capture consists of real background network traffic and can be employed for training normality models. Instead, the test capture includes both clean traffic and anomalous flows obtained as the combination of background traffic and controlled attack traffic generated using advanced hacking tools. Three classes of attack have been considered: Denial-of-Service (DoS), scan, and botnet. Since the effect of botnet over the normal traffic has not been taken into account in [10], we selected only the first two attack categories. These attacks are denoted as DoS53, DoS11, Scan44, and Scan11. The first number in the attack names refers to the number of attackers whereas the second identifies the number of victims [10]. In this work, $320,000$ images have been generated from the calibration set with a 1-second time window. The validation set is obtained by sampling 10% of the training set. For testing, $70,462$ images have been generated from the test set. $60,000$ images are normal, $1,676$ represent DoS11, $4,596$ are DoS53 attack samples, 992 are Scan11 attacks, and $3,198$ images are Scan44 representations. Based on the network structure in the dataset, images $\mathcal{I}_C$ are of shape $H = 256$ and $W = 256$.

B. Results on all attacks

Tab. II depicts the performance of the proposed approach on all the considered attacks with different options for the loss function and the learning architecture. It is worth noticing the superiority in performance of $\mathcal{I}_C$ with respect to $\mathcal{I}$. More specifically, the VAE, both with unweighted and weighted BCE, suffered from mode collapse using $\mathcal{I}$, i.e., all the input images have been mapped to the same latent vector, providing poor generalization capabilities. Concerning the AE, to achieve a recall of approximately 50% when using $\mathcal{I}$, the p_{fa}. i.e., probability of false alarm, threshold has to be set to 20%, thus proving that $\mathcal{I}$ is not a suitable input for low-complexity deep learning algorithms. In addition, the results obtained using $\mathcal{I}_C$ with unweighted BCE clearly show the importance of the introduced weighting procedure to penalize wrongly reconstructed active pixels. To fairly compare the performances of the AE and the VAE when using $\mathcal{I}_C$, we selected as target p_{fa} a value of 0.15%. Tab. II clearly indicates that, thanks to the definition of $\mathcal{I}_C$, the AE and the VAE achieve comparable performances, although the former has a higher precision and the latter has a higher recall.

To better understand which attacks are easier to detect exploiting the two approaches, Tab. III shows the performance of the AE and VAE with $p_{fa} = 0.15\%$ on each attack scenario. These results have been obtained considering as test set the combination of background test data and the single attack samples. Overall, the detection performance on DoS are better with respect to scan attacks. A possible interpretation is that, on average, the $\mathcal{I}_C$ representation is less effective in highlighting the presence of scan attacks, thus resulting in images that resemble normal traffic which are not easily detected by lightweight architectures. It is worth highlighting how the AE is more precise to detect DoS53 than the VAE while the latter outperforms the former in identifying Scan11 attacks in terms of recall, albeit with a higher rate of false positives. To further compare the two architectures, we computed the number of parameters to learn as well as the number of Giga Multiply-Accumulate Operations (GMACs). Both models have 1.2 millions of parameters, and 1.84 GMACs, thus being equally lightweight [20]. Therefore, it is possible to conclude that the proposed pre-processing of the incoming traffic allows to effectively and promptly detect the presence of attacks with low-complexity learning architectures. Depending on the specific application for which the anomaly detector is employed, the learning model can be selected based on the performance metric that needs to be prioritized.

TABLE II
RESULTS ON ALL ATTACKS. DASH SYMBOL MEANS RANDOM PREDICTIONS, I.E., THE MODEL SUFFERED FROM MODE COLLAPSE.

Model	wBCE	Image	p_{fa} (%)	F1	Recall	Precision	Accuracy
AE	1	$\mathcal{I}$	20	0.3662	0.4551	0.3072	0.7678
AE	15	$\mathcal{I}$	20	0.3897	0.4730	0.3302	0.7795
AE	1	$\mathcal{I}_C$	14	0.8058	0.8835	0.7407	0.9367
AE	15	$\mathcal{I}_C$	0.15	0.9405	0.9093	0.9739	0.9829
VAE	1	$\mathcal{I}$	-	-	-	-	-
VAE	15	$\mathcal{I}$	-	-	-	-	-
VAE	1	$\mathcal{I}_C$	0.15	**0.9428**	0.9037	**0.9852**	**0.9837**
VAE	15	$\mathcal{I}_C$	0.15	0.9248	**0.9233**	0.9264	0.9772

TABLE III
ATTACK-WISE RESULTS OF THE TWO ANOMALY DETECTORS IN GRAY FROM TAB. II. ALL METHODS EMPLOY $\mathcal{I}_C$ AS INPUT.

Attack	Model	wBCE	F1	Recall	Precision	Accuracy
DoS11	AE	15	**0.9243**	0.9899	**0.8668**	**0.9956**
	VAE	15	0.8033	**0.9905**	0.6756	0.9870
DoS53	AE	15	**0.9697**	**0.9934**	**0.9471**	**0.9956**
	VAE	15	0.9165	0.9928	0.8510	0.9872
Scan11	AE	15	**0.7647**	0.7782	**0.7517**	**0.9922**
	VAE	15	0.6368	**0.8387**	0.5132	0.9844
Scan44	AE	15	**0.8432**	0.7871	**0.9080**	**0.9852**
	VAE	15	0.7893	**0.8183**	0.7623	0.9779

TABLE IV
COMPARISON IN TERMS OF RECALL WITH STATE-OF-THE-ART ON UGR'16. DASH SYMBOL − MEANS NOT AVAILABLE.

	Approach	DoS11	DoS53	DoS	Scan11	Scan44	Scan
S	LR [21]	-	-	0.9150	-	-	0.9160
	RF [21]	-	-	0.8840	-	-	**0.9250**
	Radial SVM [21]	-	-	0.8310	-	-	0.5360
	Linear SVM [21]	-	-	0.8980	-	-	0.9100
U	Kitsune [22]	-	-	0.6400	0.0100	0.7200	-
	Tensor-based [23]	-	-	**0.9966**	**0.9999**	**0.9999**	-
Ours	AE	0.9900	0.9935	0.9927	0.7782	0.7871	0.7849
	VAE	0.9905	0.9928	0.9925	0.8387	0.8183	0.8230

C. Comparison with state-of-the-art

To prove the effectiveness of the proposed representation, we compared our approach with both supervised (S) and

unsupervised (U) methods presented in the literature. We report the results in terms of recall in Tab. IV. The proposed method outperforms supervised approaches, i.e., Logistic Regression (LR), Random Forest (RF), and SVM, for DoS, while it achieves lower performance on scan. Differently from supervised methods, the proposed detector has no previous knowledge about the attack types, thus making the anomaly detection more challenging. As for the unsupervised methods, our representation achieved significantly higher performance on both attacks with respect to the method presented in [22]. In contrast, it reached lower recall values than the technique introduced in [23]. The performance gap is negligible for DoS, while it is considerable for scan. This behavior can be attributed to the difference in complexity of the two approaches. Although in [23] the number of parameters and GMACs are not provided, they perform 3D convolutions with $64 \times 64 \times 64$ input tensors, while we employ 2D convolutions with 256×256 images. Moreover, while in [23] data was processed in 1-minute chunks, the proposed approach has a significantly higher promptness by using 1-second time windows. Additionally, we compare our results with the method proposed in [17]. To perform a fair comparison, however, the same data cleaning procedure introduced in [17] has been applied. Specifically, the authors retained only the traffic windows for which the mapping was successful (i.e., the attack pixel p was mapped to a raw index larger than 35), resulting in $9,541$ anomalous data samples. In [17], a binary classification between normal and anomalous instances has been performed achieving a recall of approximately 1. By using the proposed representation $\mathcal{I}_{\mathcal{C}}$, we obtained a recall of 0.9951 and 0.9947 for the AE and VAE, respectively. This demonstrates the effectiveness of the proposed representation which, thanks to its ability of highlighting the presence of attacks, leads to similar performance with respect to [17] despite the smaller complexity. In fact, while the proposed vanilla architectures require the training of 1.2M parameters, the VAE proposed in [17] involves 268M parameters. Our architectures, instead, are noticeably more lightweight than the one proposed in [17] (1.84 GMACs versus 2.41 GMACs). Therefore, the usage of the $\mathcal{I}_C$ representation allows to decrease the computational complexity of both training and inference processes while achieving similar performance.

IV. Conclusions

In this work, an image-based representation of network traffic has been presented, and its efficacy for anomaly detection has been demonstrated. The comparison between two lightweight learning architectures and state-of-the art approaches highlighted that, thanks to the definition of a proper image-based representation, it is possible to reduce the computational complexity of the processing algorithm. A possible drawback of the proposed approach is that 1-second chunks of traffic data could be insufficient to capture long-term attacks in a complex network, e.g., scan attacks. As a future work, the introduction of temporal information will be explored, thus analyzing the evolution of network status.

References

[1] P. Lin, K. Ye, and C. Xu, "Dynamic network anomaly detection system by using deep learning techniques," in *12th CLOUD*. Springer, 2019.

[2] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, pp. 1–58, 2009.

[3] A. A. Cook, G. Mısırlı, and Z. Fan, "Anomaly Detection for IoT Time-Series Data: A Survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, 2020.

[4] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, pp. e4150, 2021.

[5] T. Ahmad, D. Truscan, J. Vain, and I. Porres, "Early Detection of Network Attacks Using Deep Learning," in *IEEE ICSTW*, 2022.

[6] A. Maćkiewicz and W. Ratajczak, "Principal components analysis (PCA)," *Computers & Geosciences*, vol. 19, no. 3, pp. 303–342, 1993.

[7] C. Callegari, L. Gazzarrini, S. Giordano, M. Pagano, and T. Pepe, "A Novel PCA-Based Network Anomaly Detection," in *IEEE ICC*, 2011.

[8] J. Camacho, G. Maciá-Fernández, J. Díaz-Verdejo, and P. García-Teodoro, "Tackling the Big Data 4 Vs for anomaly detection," in *IEEE INFOCOMW*, 2014.

[9] T Nguyen, J. He, L. T. Le, W. Bao, and N. H. Tran, "Federated PCA on Grassmann Manifold for Anomaly Detection in IoT Networks," in *IEEE CCC*, 2023.

[10] G. Maciá-Fernández, J. Camacho, R. Magán-Carrión, P. García-Teodoro, and R. Therón, "UGR '16: A new dataset for the evaluation of cyclostationarity-based network IDSs," *Computers & Security*, vol. 73, pp. 411–424, 2018.

[11] G. Bovenzi, G. Aceto, D. Ciuonzo, A. Montieri, V. Persico, and A. Pescapé, "Network anomaly detection methods in IoT environments via deep learning: A Fair comparison of performance and robustness," *Computers & Security*, vol. 128, pp. 103167, 2023.

[12] W. Chen, Z. Wang, L. Chang, K. Wang, Y. Zhong, D. Han, C. Duan, X. Yin, J. Yang, and X. Shi, "Network anomaly detection via similarity-aware ensemble learning with ADSim," *Computer Networks*, vol. 247, pp. 110423, 2024.

[13] J. Fu, L. Wang, J. Ke, K. Yang, and R. Yu, "GANAD: A GAN-based method for network anomaly detection," *World Wide Web*, pp. 1–22, 2023.

[14] P. Zhang, F. He, H. Zhang, J. Hu, X. Huang, J. Wang, X. Yin, H. Zhu, and Y. Li, "Real-Time Malicious Traffic Detection With Online Isolation Forest Over SD-WAN," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 2076–2090, 2023.

[15] H. Geng, Q. Ma, H. Chi, Z. Zhang, J. Yang, and X. Yin, "DUdetector: A dual-granularity unsupervised model for network anomaly detection," *Computer Networks*, vol. 257, pp. 110937, 2025.

[16] S. Baldoni, M. Carli, and F. Battisti, "Analysis of a 2D Representation for CPS Anomaly Detection in a Context-Based Security Framework," *Frontiers in Signal Processing*, vol. 1, 2022.

[17] S. Casarin, S. Baldoni, M. Carli, P. Zanuttigh, and F. Battisti, "Unsupervised Network Anomaly Detection by Learning on 2D Data Representations," in *2022 9th Swiss Conference on Data Science (SDS)*, 2022, pp. 53–58.

[18] S. Baldoni and F. Battisti, "Histogram-based network traffic representation for anomaly detection through PCA," *Computer Networks*, vol. 265, pp. 111276, 2025.

[19] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[20] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Communications of the ACM*, vol. 64, no. 3, pp. 107–115, 2021.

[21] R. Magán-Carrión, D. Urda, I. Díaz-Cano, and B. Dorronsoro, "Towards a Reliable Comparison and Evaluation of Network Intrusion Detection Systems Based on Machine Learning Approaches," *Applied Sciences*, vol. 10, no. 5, 2020.

[22] J. G. Medina-Arco, R. Magán-Carrión, and R. A. Rodríguez-Gómez, "Exploring Hidden Anomalies in UGR'16 Network Dataset with Kitsune," in *FQAS*, 2023.

[23] M. Shajari, H. Geng, K. Hu, and A. Leon-Garcia, "Tensor-Based Online Network Anomaly Detection and Diagnosis," *IEEE Access*, vol. 10, pp. 85792–85817, 2022.