# Recurrence Plots-Based Network Attack Classification Using CNN-Autoencoders

Gianni D'Angelo, Eslam Farsimadan(✉), and Francesco Palmieri

Department of Computer Science, Unversity of Salerno, Fisciano, (SA), Italy
{giadangelo,efarsimadan,fpalmieri}@unisa.it
https://www.di.unisa.it

**Abstract.** The advent of the Internet of Things, with the consequent changes in network architectures and communication dynamics, has affected the security market by introducing further complexity in traffic flow analysis, classification, and detection activities. Consequently, to face these emerging challenges, new empowered strategies are needed to effectively spot anomalous events within legitimate traffic and guarantee the success of early alerting facilities. However, such detection and classification strategies strongly depend on the right choice of employed features, which can be mined from individual or aggregated observations. Therefore, this work explores the theory of dynamic non-linear systems for effectively capturing and understanding the more expressive Internet traffic dynamics arranged as Recurrence Plots. To accomplish this, it leverages the abilities of Convolutional Autoencoders to derive meaningful features from the constructed plots. The achieved results, derived from a real dataset, demonstrate the effectiveness of the presented approach by also outperforming state-of-the-art classifiers.

**Keywords:** Attack Classification · Recurrence Plots · Autoencoders · Features Extraction · Stacked Networks

## 1 Introduction

The success of IoT technologies, being the origin of significant changes in communication dynamics (e.g., affecting network layout/architecture, protocols, and interaction patterns), has strongly influenced the nature of Internet traffic introducing new security challenges. In particular, due to the exponential growth of the number of IoT traffic sources and consequent traffic volumes (in terms of a superposition of individual flows on aggregation nodes and interfaces), distinguishing network anomalies became a difficult task, in which randomness and background noise effects are more and more dominant. Also, the evolution from centralized or cloud-based solutions to edge service architectures, with the consequent flattening of infrastructure topologies and the security perimeter dissolution, introduced further complexity in traffic flow collection and interpretation. Such interpretation, implying the understanding of the whole spectrum

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023
O. Gervasi et al. (Eds.): ICCSA 2023 Workshops, LNCS 14105, pp. 191–209, 2023.
https://doi.org/10.1007/978-3-031-37108-0_13

of the underlying traffic dynamics, requires careful analysis and correlation of the most characterizing traffic features with the final goal of discovering less evident (or often almost hidden) relationships between dynamics that describe the nature of a specific flow (i.e., a legitimate activity or a malicious one). Moreover, such a degree of complexity in the network traffic behavior cannot be effectively described with traditional traffic models, and it has been shown [1] that, given the well-known chaotic nature of Internet traffic, the theory of nonlinear dynamical systems can be very helpful in providing extremely deep insights into traffic flow organization properties by highlighting the existence of periodic structures and recurrence phenomena [2] that are not evident at a glance and are significantly effective in discriminating events that deviate from normality. This is also due to the capability of nonlinear analysis of exploring the system's behavior in the phase space, which is a dimensionally richer representation of the system depicting its evolution pattern simultaneously on multiple time scales. That means, in more detail, reconstructing and studying the system's attractor to appreciate the structure of multi-dimensional curves, also known as trajectories, formed in such space, corresponding to the system's evolution (or motion) over time. Such attractor is highly descriptive of the most significant, intrinsic, and discriminative system dynamics so that it can be used as an invaluable source of features for attack detection and classification.

A Recurrence Plot (RP) [3] is an extremely effective way of representing a system's behavior in phase space and hence visualizing its attractor as a two-dimensional image. Such image is characterized by large-scale or typological patterns that can be homogeneous, periodic, drift and disrupted as well as by the presence of small-scale structures, defining the specific texture, identifying single dots, diagonal, vertical, horizontal, or bowed lines [4,5]. So, the structure of an RP is an effective tool for representing the behavior of a network flow and hence for its classification as normal or anomalous. Unfortunately, while being an extremely effective way of representing the system behavior, the visual interpretation of RPs is extremely difficult and requires much experience. Accordingly, in this work, we exploit the potentialities of a Convolutional Sparse Autoencoders (CNN-SAE)-based neural network in extracting relevant spatial features from the RPs associated with traffic flows. To accomplish this, we start from some elementary statistical features (referred to as basic features) derived by sampling multiple consecutive observations over time and, thus, aggregated within the same temporal window. Then, by leveraging the aforementioned non-linear analysis framework, we arrange them as multiple RPs (one for each basic feature) and hence multi-channel images. After that, a CNN-SAE is employed to derive new spatial features capable of capturing more complex and discriminating dynamics and thus significantly improving the DNNs-based classification process. Therefore, we employ a pipelined architecture by combining the following three main aspects, namely: extracting non-linear characteristics from the basic ones, finding out relevant features from the related RPs processed through the usage of a CNN-SAE, and then performing attack classification tasks by using the classification abilities of a fully-connected softmax DNN, respectively.

In addition, a comprehensive mathematical explanation of the overall workflow is provided to clarify how the proposed approach improves and affects the feature extraction process. Moreover, its effectiveness is investigated in the presence of several unbalanced dataset partitions and finally compared with the most common and widely available state-of-the-art approaches.

Hence, the main contributions of this paper can be summarized as follows:

1. A formal description of the overall workflow is provided to investigate how the proposed approach, from the theoretical point of view, improves and affects the feature extraction process;
2. Non-linear characteristics, arranged as RPs, and the related spatial features extracted using a CNN-SAE are combined to perform attack classification tasks.

The remainder of the paper is organized as follows. Section 2 will report the related works about the most relevant DL-based approaches applied in the attack detection/classification and RP-related classification tasks, respectively. Section 3 will describe the proposed approach by providing a detailed mathematical formalization of the RPs and AEs. Finally, Sect. 4 will report the experimental phase and achieved results, while Sect. 5 will present conclusions and future work.

## 2 Related Work

Recurrence Plots (RPs) have been applied in many scientific fields for studying and better understanding the behavior of complex systems, including astrophysics, earth sciences, engineering, biology, cardiology, and neuroscience, as extensively surveyed in [4]. In particular DL-based models have been widely employed for extracting non-linear features from RPs calculated on raw data samples [6–8]. For instance, L. Kirichenko et al. [9] have investigated the effectiveness of RPs derived from electroencephalograms (EEG) time series concerning various human conditions. The obtained results, carried out using the Epileptic Seizure Recognition dataset [10], have proven the effectiveness of CNNs in detecting both seizure and no-seizure situations from the related plots with an average accuracy of 98.40%. K. Tziridis et al. [11] have deeply investigated the RPs related to EEG signals by applying to them several noise levels. More precisely, they have perpetuated the input signal by adding, each time, one noise level and then classifying the related RPs through several famous state-of-the-art CNNs, such as AlexNet [12], ResNet18 [13], DenseNet121 [14], and VGG16 [15], respectively. The obtained results have proven, for each considered dataset, the validity of RPs in the presence of several noise levels by achieving an average accuracy between 95% and 97%. Furthermore, RPs have also been investigated in strictly-related Computer Science fields, like Malware classification. In this direction, S. Sartoli et al. [16] have employed a CNN to perform a malware classification task by considering the plots derived from the malware binary

images. More precisely, the obtained results carried out from the Microsoft Malware Classification Challenge dataset [17] have proven the effectiveness of the proposed approach with an average accuracy of 96.7% in the presence of nine unbalanced malware families. Internet traffic analysis solutions have also been investigated by using non-linear features arranged as RPs, which offered significant advantages in identifying more complex and unidentified traffic categories [18]. More precisely, since RPs can visualize complex systems' states by representing discriminative and dynamic characteristics [3,19], they might be employed to train a neural network or machine learning mechanism capable of providing good classification performance without being adversely affected by obfuscation techniques. The first anomaly detection experience using non-linear systems theory, and specifically Recurrence Quantification Analysis (RQA), has been described in [20]. Similar techniques have also been employed [21], relying on time-dependent Unthresholded RPs to represent traffic time series, with the goal of extracting non-linear features from them, to be passed in the following to an Extreme Learning Machine-based Auto-Encoder. Convolutional Neural Networks (CNNs), Autoencoders (AEs), and their related combinations are eligible under the umbrella of Deep Neural Networks (DNNs), which have employed to deal with the most disparate classification tasks (e.g., unbalanced dataset, encrypted traffic, attack classification, and anomaly detection) by achieving very promising results [22–25].

Therefore, based on the reported literature contributions and their related results, we combine the abilities of Autoencoders in distilling meaningful and relevant features and the effectiveness of non-linear characteristics in capturing traffic behavior, arranged as RPs, to propose a novel classifier capable of performing attack classification tasks in the presence of different conditions in terms of available training/testing datasets.

## 3 The Attack Classification Strategy

This section presents the proposed approach for implementing the proposed attack classification strategy capable of exploiting the effectiveness of RPs in capturing traffic flow information. We accomplish this through three main steps, namely: determining new non-linear features from the basic ones (that is, building the RPs), extracting the more representative features from the RPs, and then performing classification using a Stacked Neural Network. More precisely, as shown in Fig. 1, the RPs are built through the traditional delay-coordinate embedding process, whereas the intermediate feature extraction step is performed by employing a CNN-SAE trained on the derived RPs. Then, for implementing the final classification step, we use the AE-related latent space as input to feed a fully-connected softmax neural network. As depicted in Fig. 1, in this way, we combine the effectiveness of RPs in capturing traffic dynamics, the capability of AEs in finding relevant features, and the classification abilities of DNNs. Therefore, in order to better present the proposed workflow's potentialities, we give a detailed description of each step by also providing a mathematical formulation of the involved CNN-SAE to explain how it is able to interpret RPs by

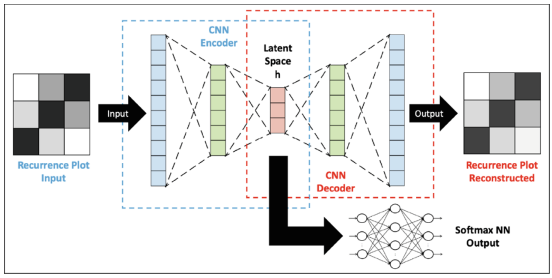


**Fig. 1.** The proposed workflow.

determining new spatial features that differ from the well-known large-scale or small-scale structures commonly used in human-driven visual inspection.

### 3.1 Determining Recurrence Plot-Based Features

Understanding the hidden criteria and laws that rule network traffic behavior in certain moments is a very complex task, essentially due to the well-known irregularities and chaotic evolution trends that characterize its evolution. Such complexity is not easily manageable with the traditional event modeling arsenal and can be much better handled by using different concepts, abstractions, and points of view coming from the nonlinear analysis and dynamical system modeling framework.

In order to better understand traffic dynamics under different conditions, we specifically search for redundancies in the observed flows, in terms of a periodic recurrence of specific end-to-end communication patterns, inside traffic data. This is relatively straightforward in typical deterministic systems (e.g., describing an oscillatory phenomenon) where a return to a previously visited state occurs with a specific periodicity. Unfortunately, Internet traffic dynamics are known to be characterizable by using a chaotic system model, which, while deterministic, is aperiodic. In this case, it is not easy to discern temporal dependencies, but considering approximate repetitions of specific events, we can build more complex rules that are able to give a deeper and better representation of the observed behavior. That is, we can still exploit recurrence phenomena to describe the system behavior if we consider the concept of recurrence within a specific threshold, and more specifically, so that we can consider a system state behaving as recurrent when it repeats its behavior after a certain quantity of time, also if not exactly, in a sufficiently close way.

As previously mentioned, RPs allow the visualization of complex systems states, where recurrence is one of the main concepts related to the temporal evolution of their dynamic trajectories. More precisely, as stated by the Takens theorem [5], we can construct a topologically equivalent picture of the behavior of the original multi-dimensional system starting from the time series associated with any individual observable variable. Thus, RPs are an invaluable source of

information for understanding the most representative properties of traffic flows. However, their interpretation is extremely complex and prone to errors since it implies a high degree of subjectivity and expertise. Indeed, RPs contain much interesting information about the system of interest concealed in subtle patterns that are not easily ascertained by visual inspection. To overcome this issue, Zbilut and Webber [26] introduced the concept of Recurrence Quantification Analysis (RQA), which is an efficient and deterministic way of non-stationarity features identification in traffic flows. The core of the such analysis is that it uncovers time correlations between data that are not based on linear or non-linear assumptions and cannot be distinguished through the direct study of one-dimensional series of traffic flow volumes [18,27]. However, RQA is able to quantify only an extremely limited part of the information available in an RP, and more effective ways of extracting highly discriminant spatial features from such information are needed.

Therefore, in this section, we employ RPs images to gain significant insights related to network traffic flows by skipping the threshold-related value section issue using a gray scale for distances, in which white and black colors are used to represent short and long distances, respectively. Because network anomalies usually take place within temporal regions of a certain size, they cannot be spotted starting from individual traffic samples. That is, individual observations occurring within a time series can not effectively capture anomalous events if picked one by one or isolated in a completely different context. Instead, the behavior that defines univocally an attack (and consequently the related features) becomes evident only when multiple temporal regions are considered together.

Hence, we do not scrutinize individual traffic flows as a sample at a specific time but consider aggregates of contiguous samples determined through the use of a sliding window scheme, better capturing the traffic behavior over time. This scheme is based on two distinct values: the sliding window size and the related. The offset parameter represents the time distance between two windows. The construction of RPs is performed through delay-coordinate embedding on independent windows of samples of a specific size, taken on the time series resulting from each basic feature, as shown in Fig. 2. The time delay $\tau$ and the embedding dimension $m$ used for such purpose have been determined at the training time on the entire set of flows constituting the training set, respectively as the first minimum of average mutual information function [28], and the embedding dimension $m$, by using the False Nearest Neighbors (FNN) method [29].

Also, we aggregated RPs associated with the different basic features available in sample observations as multi-channel images, in which each channel represents one of the employed features, by improving their descriptive power.

### 3.2 Using CNN Autoencoders for Extracting Spatial Features from RPs

In this configuration, we suppose that the AE used for processing aggregated RPs, with reference to Fig. 1, includes only three layers, namely: input, hidden, and output, respectively.

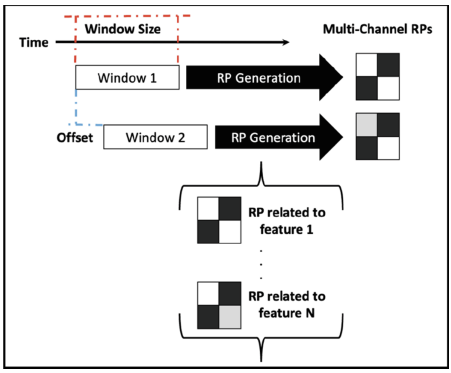


**Fig. 2.** Schema of the sliding windows for RPs aggregation.

Let $x \in \mathbb{R}^d$ be the input vector of Fig. 1, and let $x^{AE} \in \mathbb{R}^{d'}$ be the input of the considered AE, then $x \equiv x^{AE}$ and $d = d'$.

Indeed, an AE seeks to reconstruct the input by encoding it to a latent space $h$, which is then decoded to an output $\tilde{x}^{AE}$ defined as follows:

$$\tilde{x}^{AE} = y_{(W',b')}(h_{(W,b)}(x^{AE})) \equiv x^{AE} \tag{1}$$

where $(W, b)$ and $(W', b')$ represent the matrix of the weights and the bias vector of the encoder and decoder respectively, whereas $y$ is the decoder activation function.

Moreover. let $n$ be the number of hidden neurons of AE, then $W \in \mathbb{R}^{n \times d'}$, $b \in \mathbb{R}^n$, and $h_{(W,b)}$ is given by:

$$h_{(W,b)}(x^{AE}) = \sigma(Wx^{AE} + b) \tag{2}$$

where $\sigma$ is the encoder activation function.

Since an Autoencoder is trained by minimizing a loss function $\mathcal{F}$, it is possible to consider additional constraints, also referred to as regularization terms, to give the AE some specific capabilities. For instance, Sparse AEs are often employed to extract meaningful and relevant features from input data and, consequently, to improve the classification results. More precisely, Sparsity can be obtained through different strategies (e.g., L1 regularization and KL regularization) by forcing the involved AE to have only a few simultaneously active nodes (1 in theory) that, as a result, positively affects the learning process [30]. Concerning SAEs, their regularization is accomplished by adding a penalty term to the loss function, that is:

$$\mathcal{F}(x^{AE}, \tilde{x}^{AE})_{Sparse} = \mathcal{F}(x^{AE}, \tilde{x}^{AE}) + \lambda \mathcal{S}(W, b) \tag{3}$$

where $\lambda$ expresses the degree of regularization, and $\mathcal{S}(W, b)$ represents the sparsity-related term.

Once the training process is completed, the output of the $l^{th}$ hidden neuron $h_l$ can be derived by:

$$h_l = \sigma(\sum_{k=1}^{d'} w_{lk} x_k^{AE} + b_l) \tag{4}$$

Hence, since the input data of a Sparse AE is constrained by $||x^{AE}||^2 \leq 1$, each input data component $x_k^{AE}$ activating the $l^{th}$ neuron is given by:

$$x_k^{AE} = \frac{w_{lk}}{\sum_{m=1}^{d'} (w_{lm})^2}, \ \forall \ k, m = 1 \ ... \ d'. \tag{5}$$

which extracts a feature exactly corresponding to the $l^{th}$ output node. That means that a Sparse AE can learn different sets of characteristics from input data at least equal to the number of considered hidden neurons $n$.

AEs are frequently coupled with different DNNs flavors to add new functionalities and improve the ability to mine more complex features from input data. Thus, to fully exploit the effectiveness of non-linear characteristics, we use several Convolutional layers to derive relevant relations from the related aggregated RPs that can be considered as spatial features [23].

Let $X \in \mathbb{R}^{N_x \times N_y}$ be the CNN-related input, $RP \in \mathbb{R}^{N \times N}$ be an aggregated RP of Fig. 2, and $C^f \in \mathbb{R}^{a \times b}$ be the $f^{th}$ filter, respectively. Then, $X \equiv RP$ and $N_x \times N_y = d = N \times N$.

On these assumptions, the convolution operation, applied on the CNN-input $RP$ with $N_f$ filters, is defined by:

$$F_{i,j} = \sum_{f=1}^{N_f} \sum_{p=1}^{a} \sum_{q=1}^{b} C_{p,q}^f RP_{i+p-1,j+q-1} \tag{6}$$

with $F_{i,j}$ the components of the filtered input $F$.

The size of $F$ is defined through its row $F_x$ and column $F_y$ dimensions, by:

$$\begin{aligned} F_x &= \frac{N_x - a + 2P}{S_x} + 1 \\ F_y &= \frac{N_y - b + 2P}{S_y} + 1 \end{aligned} \tag{7}$$

where $P$ is the Padding referring to the number of zeros around the border of $X$, while $S_x$ and $S_y$ are the Strides related to the row and column, which control the shifting of the filter on the input matrix.

Since $d = N_x \times N_y$, it is possible to map any $x_k$ to a point $RP_{i,j}$ into a two-dimensional array. Thus, with abuse of notation, we can assert that:

$$x_k \equiv RP_{i,j} \tag{8}$$

with $k = 1, ..., d$, $i = 1, ..., N_x$, and $j = 1, ..., N_y$.

By substituting $F_{i,j}$ of Eq. 6 into $x_k^{AE}$ of Eq. 4, it yields:

$$h_l = \sigma(\sum_{k=1}^{d'} w'_{lk} x_{\phi(k)} + b_l) \tag{9}$$

such that $d' = F_x \times F_y$, while

$$w'_{lk} = \sum_{f=1}^{N_f} \sum_{p=1}^{a} \sum_{q=1}^{b} C_{p,q}^{f} w_{lk} \tag{10}$$

and

$$x_{\phi(k)} \equiv RP_{i+p-1,j+q-1} \tag{11}$$

Analogously to Eq. 4 and Eq. 5, Eq. 9 and Eq. 10 indicate that $w'_{lk}$ (for each $l, k$) represent the new extracted features that express a more complex knowledge because they are defined as the linear combination of the original $w_{lk}$. Therefore, the employed CNN-SAE configuration can process non-linear characteristics, arranged as RPs, by providing meaningful features to feed the fully-connected softmax neural network.

## 4 Performance Evaluation and Results Analysis

The presented experiments and their results are devoted to demonstrating the effectiveness of the proposed approach leveraging RPs for representing network traffic anomalies as multi-channel images and automatically interpreting them through the extraction of highly discriminative spatial classification features. To accomplish this, we employ the learning abilities of CNN Autoencoders trained on a famous dataset, which includes packets generated by different workstations, protocols, and applications.

### 4.1 Dataset and Basic Features

For the following experiments, we used real-world traffic features available in the Intrusion Detection Evaluation Dataset (CIC-IDS2017)[1] [31], which are composed of several .csv files related to Benign and Attack activities. More precisely, we considered all the Normal traffic captured during Monday 03/07/2017 and Malicious traffic captured from Tuesday 04/07/2017 to Friday 07/07/2017. Table 1 summarizes the traffic nature and related activities for each day and each employed .csv file, respectively.

Since the used dataset provides more than 80 network-related features, we pre-processed it in order to represent each bidirectional flow (summarized as a single dataset entry) as a low-dimensional vector characterized by only the following nine basic features:

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

**Table 1.** Overview on the employed .csv files.

| Day/CSV Name | Typology | Description |
|---|---|---|
| Monday | Normal | Normal human activities |
| Tuesday | Attack | Brute Force (FTP/SSH-Patator) |
| Wednesday | Attack | DoS/DDoS |
| Thursday | Attack | Brute Force (FTP/SSH-Patator) |
| Friday1 | Attack | Port Scan |
| Friday2 | Attack | DDoS |

- *Total Fwd Packets*: number of sent packets from Sender;
- *Total Bwd Packets*: number of sent packets from Receiver;
- *Flow Bytes/s*: flow of bytes for second;
- *Flow Packets/s*: flow of exchanged packets for second;
- *Average Packet Size*: average number of sent packets;
- *Packet Length Mean*: mean length of the exchanged packet;
- *Down/Up Ration*: the Down/Up ratio estimated for the considered flow;
- *SYN Flag Count*: number of set SYN Flags;
- *ACK Flag Count*: number of set ACK Flags.

Then, we used the related ground-truth to associate each flow with its corresponding typology (normal or each type of attack), thus adding the supervisory signal. Finally, following the achieved results, we rescaled each .csv file employing a standard normal distribution, with zero mean and standard deviation equal to 1 ($\mu = 0$, $\sigma = 1$).

### 4.2 Experimental Setting

Once the basic features have been selected and scaled, we merged all related .csv files as a whole dataset in order to estimate the embedding dimension $m$ and the time_delay $\tau$. More precisely, for each feature-related column, we performed the False Nearest Neighbors (FNN) Algorithm [29] and derived the Average Mutual Information (AMI) using the Non-Linear Time Series Analysis (NoLiTSA) Python framework [32]. Therefore, we set the minimum value $m = 3$ and $\tau = 1$ because they were the common values capable of minimizing the false neighbors and the AMI, respectively. Then, we split the employed dataset into two mutually-exclusive sets using, for each .csv file, the 70% of vectors for training and the remaining 30% for testing. The resulting .csv files were aggregated to form the overall training and testing datasets, respectively. Note that the temporal order was conserved during such a splitting process. Next, by following the schema proposed in Fig. 2, several sliding windows characterized by having different dimensions and offsets have been considered, as reported below:

- winDim: dimension for each window (8, 16, 32, 64, 128);
- offset: distance between two windows (1 - Consecutive, winDim/2 - semi-overlapped, winDim - non-overlapped).

More precisely, for each set of flows (each characterized by the 9 basic features mentioned above) falling within a specific sliding window, the related RPs as $(winDim-(m-1)\tau)\times(winDim-(m-1)\tau)$ matrices for each considered basic feature were first generated. Then, the resulting RPs (one for each basic feature) were aggregated to form a single multi-channel (9-channels) RP image, which characterizes the set of traffic flows falling within the considered time window.

Although many experiments were carried out by using different combinations of $winDim/offset$, $winDim = 16$ showed to get best results. Therefore, in all experiments, we used this dimension for the time window. Table 2, Table 3, and Table 4 show the details of the training and testing sets derived by using a time window of size 16 and offset values 1, 8, and 16, respectively.

**Table 2.** Dataset division with winDim = 16 and offset = 1.

| Category | Training | Testing | Total |
|---|---|---|---|
| DDoS | 89601 | 38385 | 127986 |
| DoS | 172275 | 73795 | 246070 |
| FTP-Patator | 5537 | 2353 | 7890 |
| Port Scan | 111137 | 47617 | 158754 |
| SSH-Patator | 4097 | 1745 | 5842 |
| Normal | 370609 | 158817 | 529426 |
| Total | 753256 | 322712 | 1075968 |

**Table 3.** Dataset division with winDim = 16 and offset = 8.

| Category | Training | Testing | Total |
|---|---|---|---|
| DDoS | 11201 | 4799 | 16000 |
| DoS | 21537 | 9227 | 30764 |
| FTP-Patator | 693 | 295 | 988 |
| Port Scan | 13893 | 5953 | 19846 |
| SSH-Patator | 513 | 219 | 732 |
| Normal | 46327 | 19853 | 66180 |
| Total | 94164 | 40346 | 134510 |

**Table 4.** Dataset division with winDim = 16 and offset = 16.

| Category | Training | Testing | Total |
|---|---|---|---|
| DDoS | 5601 | 2400 | 8001 |
| DoS | 10770 | 4615 | 15385 |
| FTP-Patator | 347 | 148 | 495 |
| Port Scan | 6947 | 2977 | 9924 |
| SSH-Patator | 257 | 110 | 367 |
| Normal | 23164 | 9927 | 33091 |
| Total | 47086 | 20177 | 67263 |

Finally, the dataset pre-processing phase, the related RPs extraction, and all the experiments have been conducted with an iMac Desktop equipped with an Intel 6-core $i7$ CPU @ 3.2 GHz and 16 GB RAM.

### 4.3 Evaluation Metrics

To assess the classification effectiveness of the presented scheme, we considered the traditional evaluation metrics that can be extracted from the confusion matrix: Accuracy (Acc.), Sensitivity (Sens.), Specificity (Spec.), Precision (Prec.), F-Measure (F-Mea.), and Area Under the ROC Curve (AUC), as defined below.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{13}$$

$$Specificity = \frac{TN}{TN + FP} \tag{14}$$

$$Precision = \frac{TP}{TP + FP} \tag{15}$$

$$F - Measure = \frac{2 * Sens * Prec}{Sens + Prec} \tag{16}$$

$$AUC = \frac{Sens + Spec}{2} \tag{17}$$

where for each category, TPs (True Positives) are the flows correctly classified, FPs (False Positives) are the flows incorrectly classified, FNs (False Negatives) are the flows incorrectly rejected, and TNs (True Negatives) are the flows correctly rejected.

### 4.4 Model Description and Results

As mentioned earlier, we used a CNN-SAE for interpreting RPs. Accordingly, in this Section, the implementation details are first reported, and then the experimental results are shown.

Figure 3 shows the high-level organization of the CNN layers employed for the encoder side. As depicted, it includes a sequence of three Conv2D layers with kernel_size=$(2, 2)$, activation=relu, and padding=same, characterized by having 18, 8, and 4 filters, respectively. After that, a Flatten layer is employed to map the extracted features as one-dimensional latent vectors. Hence, after having built the decoder side using the inverse sequence of the encoding layers, we trained the CNN-SAE configuration by using Adam optimizer as well as the Mean Absolute Error (MAE) as loss function for 50 epochs and batch_size = 64.

Next, we fed a fully-connected softmax neural network comprising two Dense layers with 512 neurons for each layer, activation=relu for both layers and dropout=$0.5, 0.4$, respectively. Also, to obtain the classification results as probability distributions, we used a third dense layer with 6 neurons and activation=softmax as the output layer. Then, we trained the whole network with Adam optimizer and SparseCategoricalFocalLoss loss function for 20 epochs and batch_size = 256.

Many experiments were carried out by varying the following hyper-parameters:

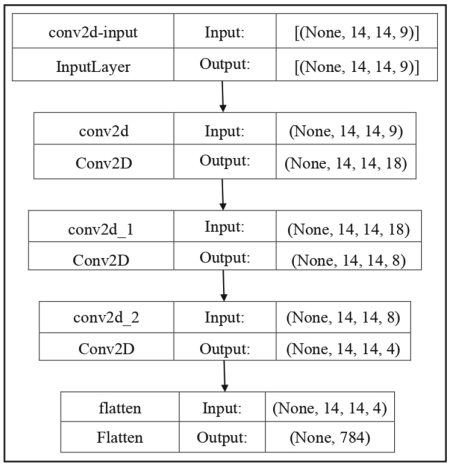


**Fig. 3.** CNN layers high-level organization.

- numConvLayers: the number of Conv2D layers considered (1, 2, 3);
- numFilters: the number of filters considered for each Conv2D layer (1, 2, 4, 8, 12, 14, 16, 18, 28, 36);
- strides: the stride length and combinations for each Conv2D layer ((1, 1), (1, 2), (2, 1), (2, 2));
- numDenseLayers: the number of Dense layers considered (1, 2, 3, 4);
- numNeurons: the number of neurons considered for each Dense layer (16, 24, 28, 32, 64, 96, 128, 256, 512, 1024);
- dropout: dropout values for each Dense layer (0.05, 0.1, 0.2, 0.3, 0.4, 0.5);
- activation: activation functions employed (relu, softmax, sigmoid);
- batch_size: considered batch_size values (32, 64, 128, 256, 512, 1024, 2048);
- loss: loss functions used (Mean Squared Error (MSE), Mean Absolute Error (MAE), SparseCategoricalCrossentropy, SparseCategoricalFocalLoss).

Note that only the configurations that achieved the best results were reported in this Sect. Table 5, Table 6, and Table 7 report the derived Confusion Matrices, while Table 8, Table 9, and Table 10 summarize the corresponding classification metrics derived by fine-tuning the whole CNN-SAE-NN stacked network for 20 additional epochs.

**Table 5.** Confusion Matrix (offset = 1).

| | DDoS | DoS | FTP-Patator | Port Scan | SSH-Patator | Normal |
|---|---|---|---|---|---|---|
| DDoS | 38149 | 215 | 0 | 14 | 0 | 7 |
| DoS | 1992 | 70734 | 0 | 661 | 0 | 408 |
| FTP-Patator | 0 | 0 | 2331 | 0 | 4 | 18 |
| Port Scan | 90 | 122 | 0 | 47303 | 0 | 102 |
| SSH-Patator | 0 | 0 | 0 | 0 | 1745 | 0 |
| Normal | 6 | 52 | 0 | 0 | 0 | 158759 |

**Table 6.** Confusion Matrix (offset = 8).

| | DDoS | DoS | FTP-Patator | Port Scan | SSH-Patator | Normal |
|---|---|---|---|---|---|---|
| DDoS | 4761 | 30 | 0 | 3 | 0 | 5 |
| DoS | 351 | 8712 | 0 | 97 | 0 | 67 |
| FTP-Patator | 0 | 0 | 291 | 0 | 0 | 4 |
| Port Scan | 2 | 13 | 0 | 5925 | 0 | 13 |
| SSH-Patator | 0 | 0 | 0 | 0 | 219 | 0 |
| Normal | 0 | 10 | 0 | 0 | 0 | 19843 |

**Table 7.** Confusion Matrix (offset = 16).

| | DDoS | DoS | FTP-Patator | Port Scan | SSH-Patator | Normal |
|---|---|---|---|---|---|---|
| DDoS | 2365 | 21 | 0 | 5 | 0 | 9 |
| DoS | 118 | 4251 | 0 | 103 | 1 | 142 |
| FTP-Patator | 0 | 0 | 145 | 0 | 0 | 3 |
| Port Scan | 3 | 5 | 0 | 2962 | 0 | 7 |
| SSH-Patator | 0 | 0 | 0 | 0 | 109 | 1 |
| Normal | 0 | 2 | 0 | 0 | 0 | 9925 |

**Table 8.** Statistic Metrics related to dataset with offset = 1.

| Category | Acc. | Spec. | Prec. | Sens. | F-Mea. | AUC |
|---|---|---|---|---|---|---|
| DDoS | 0.9928 | 0.9927 | 0.9481 | 0.9939 | 0.9704 | 0.9933 |
| DoS | 0.9893 | 0.9984 | 0.9945 | 0.9585 | 0.9762 | 0.9785 |
| FTP-Patator | 0.9999 | 1.0000 | 1.0000 | 0.9907 | 0.9953 | 0.9953 |
| Port Scan | 0.9969 | 0.9975 | 0.9859 | 0.9934 | 0.9897 | 0.9955 |
| SSH-Patator | 1.0000 | 1.0000 | 0.9977 | 1.0000 | 0.9989 | 1.0000 |
| Normal | 0.9982 | 0.9967 | 0.9966 | 0.9996 | 0.9981 | 0.9982 |
| Avg. | 0.9962 | 0.9976 | 0.9872 | 0.9893 | 0.9881 | 0.9935 |

### 4.5 Comparison and Discussion

Table 11 summarizes the averages of the performance metrics related to the employed CNN-SAE-NN configuration for each considered offset.

To better demonstrate the effectiveness of the proposed classification framework, we compare its obtained results with those derived by using the most famous state-of-the-art ML-based classifiers, such as Naive Bayes-based classifier (NB), Logistic classifier (LOG), Support Vector Machine classifier (SVM), J48 decision-tree classifier (J48), and Multilayer Perceptron classifier (MLP), specifically arranged for attack detection and classification, where all the experiments were carried out using WEKA [33]. Moreover, in order to compare the proposed method with another model based on DL, we examined several CNN and convolutional sparse autoencoder architectures that are very effective in image processing. In this section, we report just the results achieved from the best architecture (in terms of performance metrics) [34]. More precisely, we have evaluated their classification capabilities in comparison with our method by considering non-overlapped time windows with offset = 16, which is the worstscenario of

our approach (see table Table 10). Table 12 reports this comparison between the average results of our proposed neural network configuration and the average results of these classifiers.

**Table 9.** Statistic Metrics related to dataset with offset = 8.

| Category | Acc. | Spec. | Prec. | Sens. | F-Mea. | AUC |
|---|---|---|---|---|---|---|
| DDoS | 0.9903 | 0.9901 | 0.9310 | 0.9921 | 0.9606 | 0.9911 |
| DoS | 0.9859 | 0.9983 | 0.9940 | 0.9442 | 0.9684 | 0.9712 |
| FTP-Patator | 0.9999 | 1.0000 | 1.0000 | 0.9864 | 0.9932 | 0.9932 |
| Port Scan | 0.9968 | 0.9971 | 0.9834 | 0.9953 | 0.9893 | 0.9962 |
| SSH-Patator | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 | 1.0000 |
| Normal | 0.9975 | 0.9957 | 0.9955 | 0.9995 | 0.9975 | 0.9976 |
| Avg. | 0.9951 | 0.9969 | 0.9840 | 0.9863 | 0.9848 | 0.9916 |

**Table 10.** Statistic Metrics related to dataset with offset = 16.

| Category | Acc. | Spec. | Prec. | Sens. | F-Mea. | AUC |
|---|---|---|---|---|---|---|
| DDoS | 0.9923 | 0.9932 | 0.9513 | 0.9854 | 0.9606 | 0.9893 |
| DoS | 0.9806 | 0.9982 | 0.9935 | 0.9211 | 0.9684 | 0.9597 |
| FTP-Patator | 0.9999 | 1.0000 | 1.0000 | 0.9898 | 0.9932 | 0.9899 |
| Port Scan | 0.9939 | 0.9937 | 0.9648 | 0.9797 | 0.9893 | 0.9943 |
| SSH-Patator | 0.9999 | 1.0000 | 0.9909 | 0.9909 | 1.0000 | 0.9954 |
| Normal | 0.9919 | 0.9842 | 0.9839 | 0.9998 | 0.9918 | 0.9920 |
| Avg. | 0.9931 | 0.9949 | 0.9807 | 0.9787 | 0.9794 | 0.9868 |

**Table 11.** Avg. Statistic Metrics comparison related to each considered offset value.

| Offset | Acc. | Spec. | Prec. | Sens. | F-Mea. | AUC |
|---|---|---|---|---|---|---|
| offset = 1 | 0.9962 | 0.9976 | 0.9872 | 0.9893 | 0.9881 | 0.9935 |
| offset = 8 | 0.9951 | 0.9969 | 0.9840 | 0.9863 | 0.9848 | 0.9916 |
| offset = 16 | 0.9931 | 0.9949 | 0.9807 | 0.9787 | 0.9794 | 0.9868 |

As shown in Table 12, the proposed CNN-SAE network drastically outperforms other ML-based methods by significantly improving the average of all evaluation metrics, specifically F-Measure. It proves, once again, the effectiveness of the proposed approach and the abilities of Autoencoders that, in any application scenario (e.g., dataset partitions), are capable of deriving meaningful features from traffic dynamics arranged as RPs and thus achieve excellent classification results.

**Table 12.** Comparison between the proposed configuration and ML-based approaches.

| Method | Acc. | Spec. | Prec. | Sens. | F-Mea. | AUC |
|---|---|---|---|---|---|---|
| NB | 0.7604 | 0.8716 | 0.4506 | 0.5448 | 0.2657 | 0.7082 |
| LOG | 0.9657 | 0.9731 | 0.6175 | 0.6005 | 0.6020 | 0.7868 |
| SVM | 0.9369 | 0.9446 | 0.6295 | 0.4873 | 0.4999 | 0.7160 |
| J48 | 0.9926 | 0.9937 | 0.9868 | 0.8929 | 0.9243 | 0.9434 |
| MLP | 0.9499 | 0.9533 | 0.7759 | 0.6034 | 0.6546 | 0.7783 |
| CNN | 0.9919 | 0.9942 | 0.9733 | 0.9376 | 0.9539 | 0.9659 |
| Proposed | 0.9931 | 0.9949 | 0.9807 | 0.9787 | 0.9794 | 0.9868 |

## 5 Conclusion and Future Work

In this paper, we proposed a novel IoT-related network attacks classifier by combining the capabilities of AEs in automatically finding relevant features and the effectiveness of characteristics coming from non-linear analysis theory arranged as RPs. More precisely, we considered statistical information related to benign and malicious traffic-related activities available in the CIC-IDS2017 dataset as raw input data. Then, we employed the corresponding multi-channel image representation of RPs and a Convolutional Sparse Autoencoder (CNN-SAE) to perform the automatic interpretation of RP images, aimed at using the reliable discriminating power of such non-linear structures in attack classification tasks. The achieved results, derived as statistic metrics from the multi-class confusion matrix, have proven the effectiveness of the proposed classifier in the presence of several unbalanced datasets partitions by obtaining an average F-Measure of more than 98%. Finally, we have evaluated the proposed approach by comparing it with the most famous state-of-the-art ML-based attack classification techniques. The proposed approach outperformed all considered approaches with significant improvements in all evaluation metrics.

Therefore, based on the derived outcomes, we would like to propose two possible future works. The first one can be investigating the effectiveness of RPs and CNN-SAEs for multiple hostile traffic flow detection and classification in federated organizations. A possible starting point can be analyzing some of the existing effective federated approaches in this direction, such as [35]. The second one is exploring the abilities of Autoencoders in deriving relevant traffic features by considering the combination of several non-linear extraction methods. More precisely, the following studies might improve the effectiveness of real-time attack detectors and classifiers by reducing the impact of Distributed Denial of Services (DDoS) attacks and the required time for their detection, classification, and mitigation, respectively.

**Acknowledgements.** This work was partially supported by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.

## References


1. Palmieri, F.: Network anomaly detection based on logistic regression of nonlinear chaotic invariants. J. Netw. Comput. Appl. **148**, 102460 (2019). ISSN 1084–8045. https://doi.org/10.1016/j.jnca.2019.102460
2. Poincaré, H.: Sur le problème des trois corps et les équations de la dynamique. Acta mathematica **13**(1), A3–A270 (1890)
3. Eckmann, J.-P., Kamphorst, S.O., Ruelle, D.: Europhysics Letters (EPL). https://doi.org/10.1209/0295-5075/4/9/004 (1987)
4. Marwan, N., Carmen Romano, M., Thiel, M., Kurths, J.: Recurrence plots for the analysis of complex systems. Phys. Rep. **438**(5), 237–329 (2007). ISSN 0370–1573. https://doi.org/10.1016/j.physrep.2006.11.001
5. Takens, F.: Detecting strange attractors in turbulence. In: David, R., Young, L.S., (eds.) Dynamical Systems and Turbulence, Warwick 1980, pp. 366–381, Berlin, Heidelberg, 1981. Springer, Berlin Heidelberg (1980). ISBN 978-3-540-38945-3
6. Garcia-Ceja, E., Uddin, M.Z., Torresen, J.: Classification of recurrence pots distance matrices with a convolutional neural network for activity recognition. Procedia Comput. Sci. **130**, 157–163 (2018)
7. Menini, N., et al.: A soft computing framework for image classification based on recurrence plots. IEEE Geosci. Rem. Sens. Lett. **16**(2), 320–324 (2019). https://doi.org/10.1109/LGRS.2018.2872132
8. Patil, M.S., Loka, R., Parimi, A.M.: Application of ARIMA and 2d-CNNNS using recurrence plots for medium-term load forecasting. In: 2021 IEEE 2nd China International Youth Conference on Electrical Engineering (CIYCEE), pp. 1–5 (2021). https://doi.org/10.1109/CIYCEE53554.2021.9676838
9. Kirichenko, L., Radivilova, T., Bulakh, V., Zinchenko, P., Saif Alghawli, A.: Two approaches to machine learning classification of time series based on recurrence plots. In: 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP), pp. 84–89 (2020). https://doi.org/10.1109/DSMP47368.2020.9204021
10. Andrzejak, R.G., et al.: Indications of nonlinear deterministic and finite-dimensional structures in time series of brain electrical activity: dependence on recording region and brain state. Phys. Rev. E **64**, 061907 (2001)
11. Tziridis, K., Kalampokas, T., Papakostas, G.A.: EEG signal analysis for seizure detection using recurrence plots and tchebichef moments. In: 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), pp. 0184–0190 (2021). https://doi.org/10.1109/CCWC51732.2021.9376134
12. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J., Bottou, L., Weinberger, K.Q., (eds.), Advances in Neural Information Processing Systems. Curran Associates Inc
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(2015). https://arxiv.org/abs/1512.03385
14. Huang, G., Liu, Z., van der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700–4708 (2016). https://arxiv.org/abs/1608.06993
15. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014). https://arxiv.org/abs/1409.1556

16. Sartoli, S., Wei, Y., Hampton, S.: Malware classification using recurrence plots and deep neural network. In: 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 901–906 (2020).https://doi.org/10.1109/ICMLA51294.2020.00147
17. Ronen, R., Radu, M., Feuerstein, C., Yom-Tov, E., Ahmadi, M.: Microsoft malware classification challenge (2018). https://arxiv.org/abs/1802.10135
18. Palmieri, F., Fiore, U.: A nonlinear, recurrence-based approach to traffic classification. Comput. Netw. **53**(6), 761–773 (2009). https://doi.org/10.1016/j.comnet.2008.12.015
19. Hirata, Y.: Recurrence plots for characterizing random dynamical systems. Commun. Nonlinear Sci. Numer. Simul. **94**, 105552 (2021). https://doi.org/10.1016/j.cnsns.2020.105552. ISSN 1007–5704
20. Palmieri, F., Fiore, U.: Network anomaly detection through nonlinear analysis. Comp. Secur. **29**(7), 737–755 (2010)
21. Hu, Min, Feng, Xiaowei, Zhou, Shengchen, Xu, Wei: Anomaly detection in time series data based on unthresholded recurrence plots. In: Abawajy, Jemal, Choo, Kim-Kwang Raymond., Islam, Rafiqul, Xu, Zheng, Atiquzzaman, Mohammed (eds.) ATCI 2018. AISC, vol. 842, pp. 477–484. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-98776-7_52
22. Wang, P., Ye, F., Chen, X., Qian, Y.: Datanet: deep learning based encrypted network traffic classification in SDN home gateway. IEEE Access **6**, 55380–55391 (2018). https://doi.org/10.1109/ACCESS.2018.2872430
23. D'Angelo, G., Palmieri, F.: Network traffic classification using deep convolutional recurrent autoencoder neural networks for spatial–temporal features extraction. J. Netw. Comput. Appl. **173**, 102890 (2021). https://doi.org/10.1016/j.jnca.2020.102890. ISSN 1084–8045
24. Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A., Lloret, J.: Network traffic classifier with convolutional and recurrent neural networks for internet of things. IEEE Access **5**, 18042–18050 (2017)
25. Lotfollahi, M., Jafari Siavoshani, M., Shirali Hossein Zade, R., Saberian, M.: Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput. **24**(3), 1999–2012 (2020). https://doi.org/10.48550/arXiv.1709.02656
26. Zbilut, J.P., Webber, C.L.: Embeddings and delays as derived from quantification of recurrence plots. Phys. Lett. A **3**, 199–203 (1992). https://doi.org/10.1016/0375-9601(92)90426-M. ISSN 0375–9601
27. Almeida-Ñauñay, A.F., Benito, R.M., Quemada, M., Losada, J.C., Tarquis, A.M.: Recurrence plots for quantifying the vegetation indices dynamics in a semi-arid grassland. Geoderma **406**, 115488 (2022). https://doi.org/10.1016/j.geoderma.2021.115488. ISSN 0016–7061
28. Hegger, R., Kantz, H., Schreiber, T.: Practical implementation of nonlinear time series methods: the tisean package. Chaos Interdisc. J. Nonlinear Sci. **9**(2), 413–435 (1999)
29. Kennel, M.B., Brown, R., Abarbanel, H.D.I.: Determining embedding dimension for phase-space reconstruction using a geometrical construction. Phys. Rev. A **45**, 3403–3411 (1992)
30. Makhzani, A., Frey, B.: k-sparse autoencoders (2014). arXiv: 1312.5663 [cs.LG]
31. Sharafaldin, I., Lashkari, A.H., Ghorbani, A.A.: Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: ICISSP, vol. 1, pp. 108–116 (2018)
32. Mannattil, M.: NoLiTSA - NonLinear Time Series Analysis (2022). https://github.com/manu-mannattil/nolitsa. Accessed 26 July 2022

33. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques 4th. isbn: 0128042915 (Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2016)
34. Luo, W., Li, J., Yang, J., Xu, W., Zhang, J.: Convolutional sparse autoencoders for image classification. IEEE Trans. Neural Netw. Learn. Syst. **29**(7), 3289–3294 (2018). https://doi.org/10.1109/TNNLS.2017.2712793
35. Marulli, F., Balzanella, A., Campanile, L., Iacono, M., Mastroianni, M.: Exploring a federated learning approach to enhance authorship attribution of misleading information from heterogeneous sources. In: 2021 International Joint Conference on Neural Networks (IJCNN), pp. 1–8 (2021). https://doi.org/10.1109/IJCNN52387.2021.9534377