ROULETTE: A Neural Attention Multi-Output Model for Explainable Network Intrusion Detection

Giuseppina Andresini[a,*], Annalisa Appice[a,b], Francesco Paolo Caforio[c], Donato Malerba[a,b], Gennaro Vessio[a]

[a] Department of Computer Science, University of Bari Aldo Moro, Bari, Italy
[b] Consorzio Interuniversitario Nazionale per l'Informatica - CINI, Bari, Italy
[c] IISS Majorana, Martina Franca, Italy

Abstract

[1] Network Intrusion Detection (NID) systems are one of the most powerful forms of defense for protecting public and private networks. Most of the prominent methods applied to NID problems consist of Deep Learning methods that have achieved outstanding accuracy performance. However, even though they are effective, these systems are still too complex to interpret and explain. In recent years this lack of interpretability and explainability has begun to be a major drawback of deep neural models, even in NID applications. With the aim of filling this gap, we propose ROULETTE: a method based on a new neural model with attention for an accurate, explainable multi-class classification of network traffic data. In particular, attention is coupled with a multi-output Deep Learning strategy that helps to discriminate better between network intrusion categories. We report the results of extensive experimentation on two benchmark datasets, namely NSL-KDD and UNSW-NB15, which show the beneficial effects of the proposed attention mechanism and multi-output learning

*Corresponding author
Email addresses: giuseppina.andresini@uniba.it (Giuseppina Andresini), annalisa.appice@uniba.it (Annalisa Appice), francescopaolo.caforio@posta.istruzione.it (Francesco Paolo Caforio), donato.malerba@uniba.it (Donato Malerba), gennaro.vessio@uniba.it (Gennaro Vessio)

[1] This version of the contribution has been accepted for publication, after peer review but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1016/j.eswa.2022.117144. Accepted Version is subject to the publisher's Accepted Manuscript terms of use https://www.elsevier.com/about/policies-and-standards/copyright

strategy on both the accuracy and explainability of the decisions made by the method.

Keywords: network intrusion detection, multi-class classification, deep learning, attention, explainable artificial intelligence, multi-output learning

1. Introduction

Over the past decade, the predominance of Deep Learning in intrusion detection has been repeatedly assessed in the cybersecurity literature (Berman et al. 2019, Naseer et al. 2018). In particular, several methods based on Deep Learning, such as Autoencoders (Naseer et al. 2018), Recurrent Neural Networks (Folino et al. 2021), Long Short-Term Memory networks (Sovilj et al. 2020, Yin et al. 2017), Generative Adversarial Networks (Yang et al. 2019, Andresini, Appice, De Rose & Malerba 2021), Convolutional Neural Networks (Al-Turaiki & Altwaijry 2021, Andresini, Appice & Malerba 2021*b*), Siamese and Triplet networks (Bedi et al. 2020*b*, Andresini, Appice & Malerba 2021*a*), have recently contributed to introducing advanced classification capabilities into Network Intrusion Detection (NID) systems, sustaining the resilience of the security line of private and public networks.

However, Deep Learning techniques train classification models that are typically "black-boxes". Indeed, these models are implicitly represented in numerical form as synaptic weights in the network and, in general, it is difficult, if not impossible, to interpret these weights, due to the complexity of the network structure. The opacity of these black-boxes was acceptable as long as the dominant criterion for assessing the quality of NID systems was their accuracy, measured in terms of standard evaluation metrics (e.g., f-score and predictive accuracy). In some classification problems, accurate black-box models may be acceptable. However, even though the priority of modern NID systems today is still to provide accurate classifications of network traffic, easier-to-explain models are becoming increasingly desirable in NID applications. In fact, "explanations" can provide measurable factors on which characteristics influence the prediction of a cyber-attack and to what extent (Mane & Rao 2021). These factors can give the security analyst much better insight into why an alert was reported. Furthermore, if the explanation corresponds to domain knowledge, the analyst can easily and confidently approve it. Therefore, translating the model outcome into feature contribution and analyzing the impact of certain traffic characteristics can increase stakeholder confidence (Wali & Khan 2021).

The explainability of classifications refers partially to the human interpretability of the processes underlying the decisions given by a classification model. The more explainable a certain model is, the easier it will be for a human to understand or explain the underlying reasoning. Therefore, explaining intrusion classifications can help turn predictions into actions to better achieve the resilience of the network defense security line. We note that this need for explainable alerts also matches the emerging EU vision, which is extending the "right to explanation" formulated by the GDPR to solutions based on Artificial Intelligence, and especially Deep Learning (Sartor & Lagioia 2020).

Attention is an increasingly popular Deep Learning technique, oriented towards the design of explainable deep neural models. This mechanism allows the adaptive selection of the input where the network "sees" the most important information. The attention mechanism was introduced to improve the accuracy performance of the encoder-decoder model for machine translation (Bahdanau et al. 2015). It has also been exploited in Computer Vision systems to improve their performance for a variety of tasks, ranging from image captioning to visual question answering (Guo et al. 2021, Komodakis & Zagoruyko 2017). However, analyzing the results of the network attention branches could also provide insight into how the black-box model works, by contributing to the enhancement of the explainability of its decisions. Despite this relationship between the attention mechanism and the challenge of explainability, recent network security studies that have tested attention in NID applications have focused solely on the gain in accuracy achieved (Liu et al. 2020, Tang et al. 2020, Zhao et al. 2022). In addition, these studies apply the attention mechanism with feature-vector representations of network traffic flow traces. On the other hand, based on human visual attention, attention mechanisms have recently proved very effective in image classification applications (Joshi et al. 2021).

The present study is boosted by the interesting results recently presented in (Andresini, Appice, De Rose & Malerba 2021, Andresini, Appice & Malerba 2021*b*), which assessed how imagery representations of network traffic data may be adopted to accurately separate intrusions from normal traffic. Using this idea as a springboard, we propose a new NID system, called ROULETTE (neuRal attentiOn MULti-Output ModEl for explainable InTrusion DeTEction), which applies a Convolutional Neural Network (CNN) with an attention mechanism to images converted from flow characteristics of network traffic data. The main contribution of this study is the evaluation of the effectiveness of the attention mechanism in the CNN classification of network flow traces. Furthermore, contrary to (Andresini, Appice, De Rose & Malerba 2021, Andresini, Appice & Malerba 2021*b*), where the classification was binary, i.e. normal vs. intrusion, in this study the classification is multi-class, i.e. we separate intrusions from normal samples, but we also recognize the attack category. The effectiveness of the proposed neural model with attention is quantified in terms of accuracy of the classifications, as well as transparency of the decisions. In fact, thanks to the attention mechanism, we are able to produce the attention map of a network flow trace classification, and this map is expected to specify the flow characteristics that are most relevant for classification. This distinction of input traffic data allows us to identify the specific flow characteristics of intrusion categories and can provide useful information for cyber-defenders with no prior knowledge. In particular, we can consider such information as a hint to reduce the workload in manual analysis.

An additional contribution is the improvement of the performance of the classification model by using a multi-output architecture that is trained to simultaneously predict both the binary output and the multi-class output of a network flow trace. The proposed multi-output architecture has two branches that produce fully-connected heads at the end of the network: the branch with the binary head allows us to learn features that are useful to separate normal traffic from intrusions, which helps the branch with the multi-class head to recognize different categories of intrusions.

In short, this paper provides the following contributions:

- The definition of an innovative neural methodology for NID applications, which integrates the attention mechanism to achieve both accuracy and transparency in multi-class classifications.

- The formulation of a multi-output architecture to predict the intrusion category of any new network flow trace by taking advantage of features learned from the binary classification of normal traffic data vs. intrusions.

- The presentation of the results of an extensive evaluation that investigates the feasibility of the proposed learning components in the multi-class scenario, as well as the ability of our methodology to obtain accuracy comparable to competitive, Deep Learning-based approaches taken from the recent literature on NID systems.

- The exploration of the effect of several properties of explanations (i.e., compactness, robustness and separability), produced through the attention mechanism, on accuracy, and the analysis of particular information disclosed by the produced explanations on the flow characteristics of specific attack categories.

This paper is organized as follows. Related works are presented in Section 2. The proposed multi-output neural network with attention is described in Section 3. The experimental setup is described in detail in Section 4. The results of the evaluation of the proposed method are discussed in Sections 5 and 6, regarding accuracy and explainability, respectively. Finally, Section 7 refocuses on the purpose of our research, draws conclusions and illustrates possible future developments.

2. Related Work

Recent trends in cybersecurity research have classified Deep Learning as a prominent Artificial Intelligence paradigm for addressing NID problems. In this paper we renew a Deep Learning-based approach that integrates a neural attention mechanism for multi-class classification of network flow traces. Therefore, we mainly focus the literature overview on recent studies applying Deep Learning (see Section 2.1) and eXplainable Artificial Intelligence - XAI (see Section 2.2) to classify flow-based network traffic data.

2.1. Deep Learning

Several recent studies have investigated the performance of various deep neural network architectures for multi-class classification of network flow traces. Most of these studies conducted experimental studies using benchmark, network flow-based datasets such as NSL-KDD (Tavallaee et al. 2009) or UNSW-NB15 (Moustafa & Slay 2015). In particular, the empirical study illustrated in (Kasongo & Sun 2020) experimented various architectures, showing how deep neural networks can gain accuracy over various traditional classifiers (e.g., SVM, KNN and Logistic Regression). Various deep neural network architectures were tested in (Vinayakumar et al. 2019) also in combination with feature selection analysis. Following this research direction, the accuracy performance of various multi-class, fully-connected, recurrent and convolutional neural network architectures is compared in (Gao et al. 2020). Moreover, this study couples Deep Learning-based classification with association rule discovery. Specifically, the association rules with the "normal" class in the consequent are applied to match network flow traces classified as malicious, and (possibly) to filter out the misclassified normal network traffic. In (Gao et al. 2019) a multi-class NID approach is formulated by combining Incremental-Extreme Learning Machine and Adaptive Principal Component Analysis. In (Al-Turaiki & Altwaijry 2021) a 2D representation of network flow traces is adopted, in order to train a multi-class CNN. As a variant of the proposed pipeline, a so-called Deep Feature Synthesis is also introduced, in order to complete a feature engineering step.

Furthermore, some recent studies have explored the performance of deep metric learning architectures (e.g., Siamese networks or Triplet networks) in multi-class NID applications. For example, the multi-class problem is studied in (Bedi et al. 2020b), where a Siamese network is trained both on pairs of similar samples (belonging to the same class) and pairs of dissimilar network flow traces (belonging to opposite classes), to classify the intrusion class trace. A testing network flow trace is classified according to a distance score computed for each class: the class with the best score is predicted. An ensemble consisting of a Siamese network, a classic deep network and an XGBoost binary classifier is described in (Bedi et al. 2020a), where the goal is to separate normal network flow traces from intrusions. A multi-class XGBoost classifier is then used to classify traces detected as intrusions into different attack classes. In (Andresini, Appice & Malerba 2021a) both the one-versus-all and the one-versus-one combination strategy are applied to Triplet networks originally trained for binary classification, in order to perform multi-class classification of network flow traces.

Finally, the multi-class classification has recently attracted attention also in Adversarial Deep Learning. In this setting, the main objective is a classifier that makes mistakes by making small changes to the training data. This idea is investigated in (Caminero et al. 2019), where Variational Generative Autoencoders are experimented in an Adversarial Deep Reinforcement Learning approach, formulated for multi-class classifications of network flow traces. A specific conditional Variational Autoencoder architecture is also described in (Lopez-Martin et al. 2017) for multi-class classification. This architecture integrates intrusion labels within the decoder layers. In particular, this study classifies network flow traces according to the intuition that the autoencoder best learns how to recover the original features when it processes the correct label as input. Therefore, a testing network flow trace can be assigned to the label that yields the recovered features that are closest to those observed.

2.2. eXplainable Artificial Intelligence

eXplainable Artificial Intelligence, or XAI, is a sub-field of Artificial Intelligence that aims to enable humans to understand the decisions of artificial systems by producing more explainable models, while maintaining a good level of predictive accuracy (Lakkaraju et al. 2019). Significant interest in the XAI research community has recently been observed in the development of "post-hoc" explanations, in which an XAI technique can be applied to already trained black-box Deep Learning models. Alternatively, an XAI technique can be incorporated into the Deep Learning algorithm. Today, several XAI techniques have already been tested in many real-world applications such as business decision, process optimization, medical diagnosis and investment recommendation, in order to improve the reliability, transparency and fairness of Deep Learning-based decisions (Xu, Uszkoreit, Du, Fan, Zhao & Zhu 2019, Antwarg et al. 2021).

Even in the cybersecurity field, security practitioners have begun to complain about the black-box nature of Deep Learning-based decisions. The recent study of Warnecke et al. (2020) started the investigation into how post-hoc XAI techniques can be applied to produce explanations for the decisions of deep neural networks, trained for both malware detection and vulnerability discovery applications. Specifically, this study compares several post-hoc XAI techniques regarding the accuracy of explanations, as well as security-focused aspects, such as completeness, efficiency and robustness. Notably, this study led to very recent studies exploring the performance of various post-hoc XAI techniques also in NID applications of Deep Learning.

In (Burkart et al. 2021) a surrogate model is trained to produce explanations of binary decisions produced in NID applications. The surrogate model is a decision tree, trained from network flow traces sampled around a counterfactual-based local decision boundary. A surrogate model is also coupled to a deep neural network in (Szczepański et al. 2020), in order to explain its black-box decision. In (Sarhan et al. 2021) the transparency of a NID system is improved by using SHAP (Lundberg & Lee 2017) to identify the input features that contribute most to binary decisions produced through a neural network black-box. SHAP is also used in (Wang et al. 2020) to perform analyses of the most relevant features for detecting each category of intrusion. In (Andresini, Pendlebury, Pierazzi, Loglisci, Appice & Cavallaro 2021) the analysis of the feature relevance is conducted via DALEX (Biecek 2018). This explanation analysis aims to explain how a binary model changes over a stream of network flow traces to fit new attack categories. In (Marino et al. 2018) an adversarial learning approach is experimented in order to understand why some of the network flow traces are mis-classified by a deep neural network. The explanations are formulated in terms of the minimal modifications required to change the output of the black-box for any mis-classified network trace. Finally, in (Caforio et al. 2021) Grad-CAM (Selvaraju et al. 2020) is applied to produce gradient-based visual explanations of the CNN binary classification of imagery-represented network flow traces. The main innovation of this study is that it combines Grad-CAM explanations with the nearest-neighbour search, in order to improve the accuracy of the CNN decisions in a post-hoc way, as well as to increase the transparency of the CNN black-box decisions.

On the other hand, a few recent NID studies have also begun to investigate the effect of intrinsic XAI techniques, such as the attention mechanism incorporated into the neural architectures and used instead of post-hoc explanations. In (Liu et al. 2020) a Bidirectional Gated Recurrent Unit network with hierarchical attention is trained for the binary classification of network flow traces, while a Temporal Convolutional Network is trained with the attention mechanism in (Zhao et al. 2022) for the multi-class classification. Finally, a pipeline with a Stacked Autoencoder and an attention mechanism is experimented in (Tang et al. 2020). The use of attention in current NID studies has so far relied only on feature vectors. Furthermore, although probing into the results of attention branches of a deep neural network should provide insight into how the black-box model works, current NID studies do not explore the improvement in explainability achieved with attention, since they are focused solely on the gain achieved in accuracy. The present work aims to address these issues.

3. Proposed Method

We assume that a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of N training samples is available, where $\mathbf{x} \in \mathbb{R}^d$ is a d-dimensional vector of features (such as the number of packets transmitted and the number of failed logins) that characterize flow traces of historical network traffic, whereas $y \in \{1, \ldots, K\}$ is the target variable with K distinct classes: *normal* traffic data and various types of *intrusion*, depending on those historically detected and labeled. The proposed intrusion detection model, schematized in Fig. 1, is mainly based on three components:

- The reformulation of the network traffic classification task as an image classification problem, which makes it possible to take advantage of convolution filters to learn a new discriminating representation.
- The use of an attention mechanism followed by an *average* layer, which allows the extraction of an easy-to-explain attention map of the classifications.
- A multi-output learning strategy that allows the network to solve a binary, auxiliary task, intended to improve the performance of the main multi-class classification task.

In the pre-processing step image encoding is performed to transform the feature vector $\mathbf{x}$ of each network flow trace into a single-channel square image $\mathbf{X} \in \mathbb{R}^{m \times m}$. This transformation is done by assigning each flow feature of $\mathbf{x}$ to a pixel frame of $\mathbf{X}$. Flow features are assigned to neighbouring pixels in the image, depending on their correlation, and are not simply stacked in an arbitrary order. A detailed description of how feature-pixel assignment is performed is reported in (Andresini, Appice, De Rose & Malerba 2021). Thanks to this encoding, we are able to reformulate the classical classification as an image classification problem. Note that this reformulation has already proved useful in (Andresini, Appice, De Rose & Malerba 2021, Caforio et al. 2021), although the previous studies handled the binary classification task (to separate normal flow traces from intrusions, regardless of the specific intrusion category). A convolution

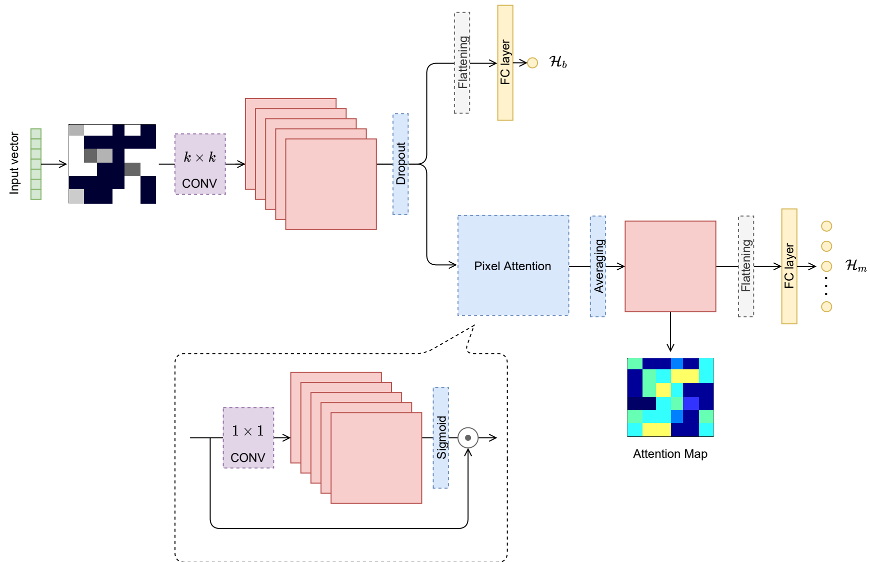


Figure 1: Schema of ROULETTE. Abbreviations: CONV = convolution; FC = fully-connected.

layer is then used to learn filters aimed at minimizing the classification error made in a backpropagated way. It is worth noting that, since the resolution of the input image is relatively low, and its content lacks the complexity of real photo-realistic images, a hierarchy of learned features is not needed, so a single convolutional layer is sufficient. For the same reason, this convolutional layer is followed by a dropout layer, which is used to mitigate early overfitting.

The *conv* + *dropout* block is then split into two branches. The first branch is responsible for the main multi-class classification task, aimed at discriminating network traffic data in multiple categories from a normal to a specific type of intrusion. Since we are interested in "explaining" the multi-class classifications made by the model, this branch includes a simple attention mechanism. This is analogous to the *pixel-attention* layer recently introduced in (Zhao et al. 2020), which is used to generate pixel-wise attention maps from the input volume.

More precisely, given an input volume X^{l-1} of size $H \times W \times C$ (height $\times$ width $\times$ channels), the *pixel-attention* layer convolves a 1×1 convolution filter, followed by a sigmoid activation. Formally:

$$\mathsf{X}^l = \sigma\left(\mathrm{CONV}\left(\mathsf{X}^{l-1}\right)\right) \odot \mathsf{X}^{l-1}, \tag{1}$$

where CONV is a 1×1 convolution, σ is the sigmoid function and X^l is the resulting output tensor at the new layer l. This mechanism basically serves to generate attention coefficients for all pixels in each feature map, thus "weighting" their contribution to the final classification. In fact, we first use C pointwise filters so that all pixels are weighted; the resulting feature maps are then squashed between 0 and 1 by the sigmoid function and multiplied element-wise with the input tensor, effectively producing a new tensor X^l of the same shape $H \times W \times C$.

Furthermore, in order to be able to produce a single attention map that explains the importance of the features in the original image, we propose using an *average* layer which basically averages corresponding pixels in the different feature maps. More precisely, given the tensor of shape $H \times W \times C$ produced by the *pixel-attention*, the *average* layer produces a single-channel matrix of size $H \times W$, for which each attention pixel α^l_{ij} is equal to:

$$\alpha^l_{ij} = \frac{1}{C} \sum_{c=1}^{C} \alpha^{l-1}_{ijc}, \tag{2}$$

where α^{l-1}_{ijc} is the attention pixel at position ij in the c-th feature map from the preceding $l-1$ layer. The *average* layer directly provides a heatmap, highlighting the input image regions the model attended on to perform the classification. In other words, a "visual explanation" is obtained. It is worth noting that since the mechanism is learnable and embedded into an end-to-end trainable model, this is in contrast to "post-hoc" visual explanation methods, such as the popular Grad-CAM (Selvaraju et al. 2017), which can only be applied after the training process has ended. A learnable attention mechanism is desirable as it helps the model focus on key parts during training. Furthermore, it does not require an expensive two-step process to derive the heatmap. In addition, this layer acts as a simple regularizer, as the feature vector to propagate would be scaled by a factor of C, and it also reduces the computational cost.

The output of the *average* layer is finally flattened and fed into a fully-connected layer, plus an output layer with a softmax activation attached and as many units as there are classes to predict. This branch aims to minimize a cross-entropy loss function:

$$\mathcal{H}_m = -\sum_{k=1}^{K} y_k \log \hat{y}_k, \tag{3}$$

where y_k is the ground truth label, while $\hat{y}_k$ is the network output for a single data sample.

Inspired by multi-output learning (Xu, Shi, Tsang, Ong, Gong & Shen 2019), we propose the use of a second branch that aims to perform a simpler binary classification between normal traffic and intrusions, regardless of the specific type of attack. The second predicted output is intended as an auxiliary classification objective, aimed at supporting the main multi-class classification task. In fact, since the two branches share the same first convolutional layer and their classification heads are optimized simultaneously by backpropagation, in this way the network is forced to learn features that are useful to separate normal traffic from attacks. This helps the main branch to focus on how to discriminate better between different types of attack that can be very similar and have overlapping characteristics. Multi-output learning is not uncommon in neural networks and has proven effective in some other domains, e.g. (Cao et al. 2018, Castellano et al. 2020). Specifically, the output of the *conv* + *dropout* block is directly flattened and given as input to a fully-connected layer followed by a single sigmoid-activated output neuron. This second output serves to minimize a binary cross-entropy:

$$\mathcal{H}_b = -\left(y \log \hat{y} + (1-y)\log\left(1-\hat{y}\right)\right). \tag{4}$$

Note that this branch does not include an attention mechanism, as the goal is to learn how to explain multi-class classification. Overall, the network learns to minimize, by backpropagation, the joint loss:

$$\mathcal{L} = \lambda \mathcal{H}_m + (1-\lambda)\mathcal{H}_b, \tag{5}$$

where $\lambda \in [0, 1]$ is a hyper-parameter that balances the contribution of the individual losses.

4. Experimental Setup

In this section we describe the datasets used for evaluating the accuracy and explainability of ROULETTE, i.e. NSL-KDD (Tavallaee et al. 2009) and UNSW-NB15 (Moustafa & Slay 2015), and implementation details.

4.1. Dataset Description

The NSL-KDD dataset (Tavallaee et al. 2009)[2] comprises normal network flow traces and four categories of attack: Denial of Service (DoS), User to Root (U2R), Remote to Local (R2L) and Probing attack. The training set is made up of 21 different attack sub-categories, while the test set is composed of 37 different attack sub-categories. This means there are 16 novel attacks in the test set. Each trace in the NSL-KDD dataset has 41 features, and detailed descriptions of these features are provided in (Tavallaee et al. 2009). We note that both U2R and R2L are rare attacks. While this dataset may not represent perfectly existing real-world networks, recent, state-of-the-art studies still use it as an effective benchmark dataset to help researchers compare different multi-class classification NID methods.

The UNSW-NB15 (Moustafa & Slay 2015),[3] on the other hand, includes realistic modern normal activities and synthetic contemporary attack behaviours extracted from network traffic monitored in 2015. Both the training set and the testing set contain normal network flow traces and nine categories of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode and Worms. Each trace in the UNSW-NB15 dataset has 49 features: detailed descriptions of these features are provided in (Moustafa & Slay 2015).

[2] `https://www.unb.ca/cic/datasets/nsl.html`

[3] `https://research.unsw.edu.au/projects/unsw-nb15-dataset`

The number of samples collected both in the training and testing set for each category per dataset is reported in Table 1. A complete description of the flow characteristics enclosed in both the NSL-KDD and UNSW-NB15 datasets is reported in (Choudhary & Kesswani 2020).

Table 1: Number of network flow traces per class type in both the training set and testing set of NSL-KDD and UNSW-NB15.

Class type	NSL-KDD		UNSW-NB15	
	Train	**Test**	**Train**	**Test**
Normal	67343	9711	56000	37000
DoS	45927	7458	12264	4089
Probe	11656	2421	-	-
R2L	995	2754	-	-
U2R	52	200	-	-
Fuzzers	-	-	18184	6062
Analysis	-	-	2000	677
Backdoors	-	-	1746	583
Exploits	-	-	33393	11132
Generic	-	-	40000	18871
Reconaissance	-	-	10491	3496
Shellcode	-	-	1133	378
Worms	-	-	130	44
Total	125973	22544	175341	82332

Table 2: Hyper-parameter search space

Hyper-parameter	Values
Mini-batch size	$\{2^5, 2^6, 2^7, 2^8, 2^9\}$
Learning rate	[0.0001, 0.001]
Dropout rate	[0, 1]
# of filters	$\{2^5, 2^6, 2^7, 2^8, 2^9\}$
Kernel size	[2, 4]
# of neurons per hidden layer	$\{2^5, 2^6, 2^7, 2^8, 2^9\}$
λ	[0.5, 1]

4.2. Implementation Details

We developed ROULETTE in Python 3, using the high-level neural network API Keras 2.4 integrated in TensorFlow (Abadi et al. 2015).[4] In the pre-processing step, the categorical input features were mapped into numerical features using the one-hot-encoder strategy, and then the numerical features were scaled using the min-max normalization.

For each dataset, we optimized the hyper-parameters of the architecture using the tree-structured Parzen estimator algorithm (Bergstra et al. 2011) as implemented in the Hyperopt library (Bergstra et al. 2013). The hyper-parameter optimization was performed using a random stratified split of 20% of the entire training as a validation set, following the Pareto principle. We selected the hyper-parameter configuration that achieved the lowest validation loss. The hyper-parameter search space is reported in Table 2. The configuration was completed by the commonly used ReLU (Glorot et al. 2011) as the activation function for each hidden layer.

We trained the network with mini-batches using back-propagation, and the gradient-based optimization was performed using the Adam update rule (Kingma

[4]https://github.com/gsndr/ROULETTE.

& Ba 2014). The weights were initialized following the Xavier scheme (Glorot & Bengio 2010). In addition, a maximum number of epochs equal to 150 was set, and an early stopping approach based on the lowest loss on the validation set (the same set used for the hyper-parameter optimization) was used, in order to retain the best classification models.

5. Accuracy Performance Analysis

In this section we show the results of an analysis aimed at evaluating the accuracy performance of ROULETTE, in order to answer the following questions:

- Q1 How does the accuracy of the proposed multi-output Deep Learning strategy change by varying a few dimensions of the neural network architecture, e.g. dropout layer, regularization penalty term and aggregation operation?
- Q2 Is the proposed multi-output Deep Learning strategy able to achieve higher accuracy than the single-output strategy?
- Q3 How does the attention mechanism help the accuracy performance of the predictive model?
- Q4 Does the defined neural attention multi-output model outperform state-of-the-art NID systems?

The multi-class accuracy metrics measured for this analysis are described in Section 5.1. The results of the sensitivity study to answer Q1 are reported in Section 5.2. The results of the ablation study to answer Q2 and Q3 are illustrated in Section 5.3. Finally, the results of the comparative analysis performed to answer Q4 are shown in Section 5.4.

5.1. Performance Metrics

We measured the predictive performance of the compared methods by computing standard multi-class classification metrics. All compared metrics were computed to evaluate performance on the classifications produced on the testing sets. Specifically, we considered the following metrics:

- Precision, which measures the precision of the classification per class type, i.e. how many network flow traces are correctly classified for a particular class type k, given all predictions of that class, i.e. $P_k = \frac{tp_k}{tp_k + fp_k}$.

- Recall, which measures the recall of the classification per class type k, i.e. how many network flow traces are correctly classified for a particular class type k given all occurrences of that class type, i.e. $R_k = \frac{tp_k}{tp_k + fn_k}$.

- F1, which measures the harmonic mean of Precision and Recall per class type k, i.e. $F1_k = 2\frac{P_k \times R_k}{P_k + R_k}$. The higher the F1 score per class type k, the better the balance between the Precision and Recall achieved by the method in predicting network flow traces of class k.

- Macro-F1, which measures the average F1 score per class type k, i.e. Macro-$F1 = \frac{1}{K}\sum_{k=1}^{K} F1_k$.

- Weighted-F1, which measures the weighted mean of the F1 score per class type k, i.e. Weighted-$F1 = \sum_{k=1}^{K} w_k F1_k$, where each weight w_k is equal to the probability of class k in the testing set, i.e. $w_k = \frac{n_k}{N}$.

- A, which measures the overall accuracy, i.e. the proportion of correctly classified network flow traces in all the classified traces, i.e. $A = \frac{1}{N}\sum_{k=1}^{K} tp_k$.

In the above formulation: tp_k is the number of network flow traces of class k that are correctly predicted as belonging to class k; fp_k is the number of network flow traces that are incorrectly classified as belonging to k; tn_k is the number of network flow traces that are correctly classified as not belonging to class k; fn_k is the number of network flow traces that are incorrectly classified as not belonging to k; n_k is the ground truth size of class k (i.e., the number of traces labeled with class k); $N = \sum_k N_k$ is the total size of the classifications; K is the number of distinct class types. In calculating the macro-metric reported above, we gave equal weights (i.e., $\frac{1}{k}$) to each class type. In this way, we avoided our evaluation offsetting the possible impact of unbalanced data learning. On

Table 3: Macro-F1, Weighted-F1 and A obtained using L1, L2 or DROPOUT. The best results are in bold.

Dataset	Architecture	Macro-F1	Weighted-F1	A
NSL-KDD	L1	0.566	0.753	0.781
	L2	0.562	0.747	0.777
	DROPOUT	**0.613**	**0.790**	**0.815**
UNSW-NB15	L1	0.374	0.731	0.727
	L2	0.414	0.760	0.751
	DROPOUT	**0.424**	**0.767**	**0.764**

the contrary, when calculating the weighted metric, the highly populated classes had a higher weight compared to the smaller ones.

5.2. Sensitivity Study

This sensitivity analysis was performed on the specific neural network architecture trained by ROULETTE, in order to explore how the accuracy of the trained neural network can be influenced by:

- The use of the dropout layer in place of the L1 or L2 regularization to prevent overfitting.
- The addition of a regularization penalty term to the objective function.
- The aggregation operation (i.e., AVG or MAX) adopted after the pixel attention layer; in other words, if it is better to take the maximum attention pixel per channel than to average all of the pixels.

Table 3 collects the Macro-F1, Weighted-F1 and A achieved by replacing the dropout layer of ROULETTE with either the L1 regularization or the L2 regularization. The results show that the dropout layer is able to gain accuracy

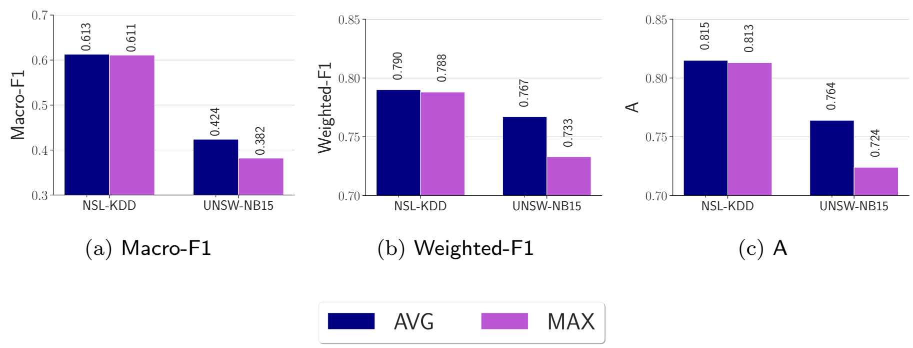
Table 4: Macro-F1, Weighted-F1 and A of ROULETTE with L1 Penalty, L2 Penalty and No Penalty in the objective function. The best results are in bold.

Dataset	Architecture	Macro-F1	Weighted-F1	A
NSL-KDD	L1 Penalty	0.551	0.731	0.760
	L2 Penalty	0.541	0.723	0.754
	No Penalty	**0.613**	**0.790**	**0.815**
UNSW-NB15	L1 Penalty	0.353	0.721	0.711
	L2 Penalty	0.390	0.735	0.717
	No Penalty	**0.424**	**0.767**	**0.764**

when compared to its counterparts that use the L1 and L2 regularization, respectively. These results were expected as dropout has become the standard choice as a regularization technique in today's architectures (Guo et al. 2019, Phaisangittisagul 2016). Studies have shown the effectiveness of dropout training also in combination with convolutional layers (Alex Kendall & Cipolla 2017, Phaisangittisagul 2016, Srivastava et al. 2014).

Table 4 collects the Macro-F1, Weighted-F1 and A achieved by adding the L1 penalty, L2 penalty and No penalty to the objective functions of ROULETTE described in Equations 3 and 4. The results show that the introduction of a regularizer to apply a penalty on the layer outputs does not lead to any improvement in the accuracy in either NSL-KDD or in UNSW-NB15. This can be explained by considering that the aforementioned dropout is already effective in countering overfitting.

Finally, Fig. 2 reports the Macro-F1, Weighted-F1 and A achieved by ROULETTE using attention with either an average layer (AVG) or a max layer (MAX). The AVG configuration, which is the baseline described in Equation 2, averages the pixels across all channels of the attention layer to produce a single attention map. The MAX configuration determines the maximum pixel on all channels.



(a) Macro-F1 (b) Weighted-F1 (c) A

Figure 2: Macro-F1, Weighted-F1 and A of ROULETTE using attention with either an average layer (AVG) or a max layer (MAX).

The experimental results show that AVG outperforms MAX in both NSL-KDD and UNSW-NB15, although the difference between the performance is higher in UNSW-NB15 than in NSL-KDD. Our main intuition is that by averaging we can retain more pixel information across all the channels rather than simply selecting the maximum.

5.3. Ablation Study

The ablation study of ROULETTE was performed by evaluating the predictive performance of three architecture configurations identified as baselines of ROULETTE. These were in turn defined by discarding the branch with the binary head or the attention mechanism from the training stage of ROULETTE. In particular, we considered the following baseline architectures:

- SO, which discards both the branch with the binary head and the attention mechanism. The deep neural network produces a single, multi-class output by minimizing the cross-entropy loss function.
- SO+A, which discards the branch with the binary head. The neural network produces a single, multi-class output with attention by minimizing the cross-entropy loss function.

Table 5: Macro-F1, Weighted-F1 and A both of ROULETTE and its baseline configurations SO, SO+A and MO. The best results are in bold.

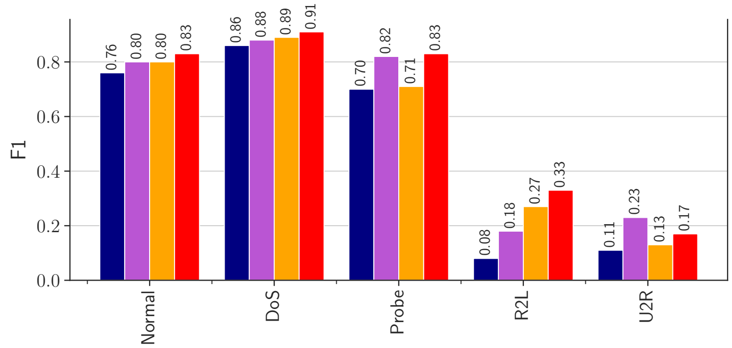
Dataset	Architecture	Macro-F1	Weighted-F1	A
NSL-KDD	SO	0.503	0.701	0.741
	SO+A	0.579	0.745	0.779
	MO	0.558	0.749	0.778
	ROULETTE	**0.613**	**0.790**	**0.815**
UNSW-NB15	SO	0.393	0.747	0.748
	SO+A	0.391	0.753	0.754
	MO	0.391	0.751	0.753
	ROULETTE	**0.424**	**0.767**	**0.764**

- MO, which discards the attention mechanism. The deep neural network produces both a multi-class and a binary output by minimizing the joint loss function.

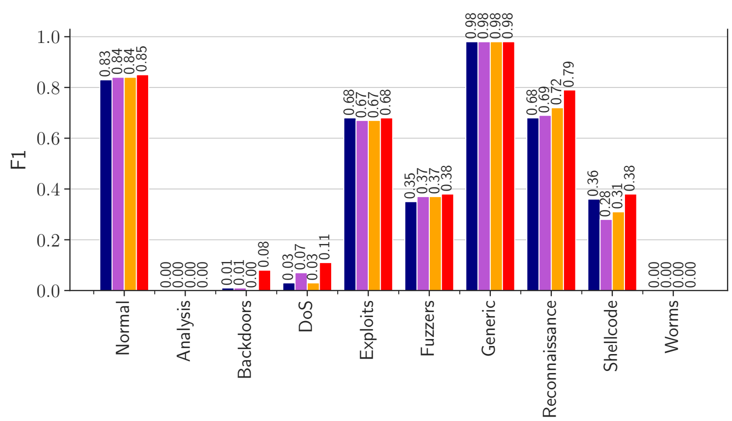
The results of Macro-F1, Weighted-F1 and A of SO, SO+A, MO and ROULETTE are reported in Table 5. They show that ROULETTE can take advantage of both the multi-output strategy and the attention mechanism achieving more accurate decisions than all its baselines.

Figure 3 reports the F1 scores computed per each class. These detailed results show that ROULETTE performs better than (or equal to) its baseline architectures in predicting all class categories of the multi-class problems considered. Notably, ROULETTE is capable of achieving a good level of accuracy on various rare classes, such as R2L and U2R of NSL-KDD, and Shellcode of UNSW-NB15.

In general, this analysis shows the viability of our idea of exploiting the multi-output strategy, in order to obtain accuracy in the multi-class branch thanks to the knowledge learned in the auxiliary binary branch. In addition,



(a) NSL-KDD



(b) UNSW-NB15

Figure 3: F1 score per class of both ROULETTE and its baseline configurations SO, SO+A and MO.

the attention mechanism, which we have introduced in the neural network to see which input information is relevant for decisions, also allows ROULETTE to achieve higher levels of accuracy.

5.4. Competitor Analysis

The comparative analysis is performed to assess the significance of accuracy and novelty of ROULETTE compared to several competitors, selected from the

Table 6: A of ROULETTE vs. state-of-the-art, Deep Learning algorithms that perform multi-class classification.

Dataset	Approach	Description	A
NSL-KDD	Al-Turaiki & Altwaijry (2021)	CNN, Deep Feature Synthesis	0.814
	Andresini, Appice & Malerba (2021*a*)	Autoencoder, Triplet network, OVA	0.778
	Andresini, Appice & Malerba (2021*a*)	Autoencoder, Triplet network, OVO	0.766
	Bedi et al. (2020*a*)	Siamese network, Ensemble, XGBoost	0.79.9
	Bedi et al. (2020*b*)	Siamese network	0.769
	Caminero et al. (2019)	Variational Generative Autoencoder	0.801
	Gao et al. (2019)	I-ELM, PCA	0.812
	Gao et al. (2020)	DNN	0.773
	Gao et al. (2020)	RNN	0.713
	Gao et al. (2020)	CNN	0.735
	Lopez-Martin et al. (2017)	Conditional Variation Autoencoder	0.801
	Tang et al. (2020)	Autoencoder, Attention, DNN	0.821
	Vinayakumar et al. (2019)	DNN	0.778
	Wang et al. (2020)	DNN, SHAP	0.803
	ROULETTE	CNN, Attention, Multi-output	0.815
UNSW-NB15	Al-Turaiki & Altwaijry (2021)	CNN, Deep Feature Synthesis	0.685
	Gao et al. (2019)	I-ELM, PCA	0.707
	Kasongo & Sun (2020)	DNN, Feature selection	0.756
	Vinayakumar et al. (2019)	DNN	0.660
	Zhao et al. (2022)	Temporal CNN, Attention	0.729
	ROULETTE	CNN, Attention, Multi-output	0.764

state of the art in NID literature. Note that the selected competitors differ in the deep neural network architecture tested. The results of the competitors are taken from the reference papers, as their code is not publicly available for repeating the experiments. However, the comparison is safe as all methods have been tested on the multi-class problem of the same training and testing sets described in Section 4.1.

We point out that the competitors that integrate the attention mechanism (Tang et al. 2020, Zhao et al. 2022) are closest to ROULETTE. Specifically, the method described in (Tang et al. 2020), which has been tested on NSL-KDD, integrates the attention layer into a deep neural network, trained with the flow-based characteristics of the dataset encoded at the encoder level of an autoencoder. On the other hand, the method described in (Zhao et al. 2022), which was tested on UNSW-NB15, integrates the attention layer into a Temporal CNN trained with the original flow-based characteristics of the dataset. Finally, we note that the method described in (Wang et al. 2020) also experiments an explanation mechanism. However, it uses post-hoc explanations based on SHAP, which allow the author to achieve transparency of the Deep Learning decisions, but it has no effect on the overall accuracy of these decisions.

For all methods in this comparative study we collect the A as this metric is provided in all reference studies. The A results, reported in Table 6, show that ROULETTE outperforms its competitors, including the attention-based competitor evaluated on UNSW-NB15 (Zhao et al. 2022). The only exception is the attention-based competitor defined in (Tang et al. 2020) which outperforms ROULETTE on NSL-KDD. However, we note that the attention mechanism of this competitor is trained on the encoded features of the dataset, instead of the original input features. The encoded features commonly allow us to gain accuracy in classification. This positive effect of autoencoders is also proved in (Andresini, Appice & Malerba 2021*b*, Andresini et al. 2020) for binary formulations of NID problems. However, training attention on encoded features excludes the opportunity to explain the effect of input traffic features on decisions.

6. Explanation Property Analysis

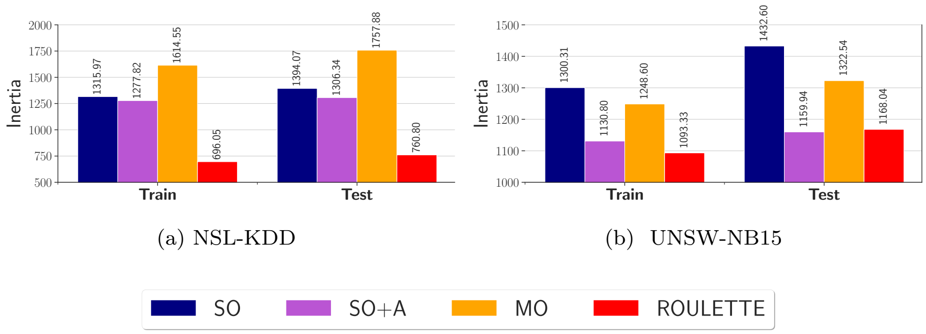
This analysis aimed to explore which properties of the intrinsic explanations produced through the attention mechanism may be connected to their ability to outperform post-hoc constructed counterparts. As post-hoc explanations we considered the visual explanation maps produced through the popular Grad-CAM technique (Selvaraju et al. 2017), after the neural training process has been completed without attention. Moreover, we explored the relationship between the observed properties of attention explanations and the accuracy performance already investigated in Section 5. Specifically, this study explores properties referred to as *compactness* (Section 6.2), *robustness* (Section 6.3) and *separability* (Section 6.4). The additional metrics considered to explore these explanation properties are introduced in Section 6.1.

6.1. Explanation Metrics

Two new metrics, namely Inertia and Average link distance, were measured for the analysis of compactness and robustness, respectively. The aforementioned Macro-F1, Weighted-F1 and A were measured for the analysis of separability.

Inertia measures the similarities of visual explanation maps clustered by class type. The lower the Inertia, the higher the ability to make transparent the common, intrinsic factors of the signature that is beyond the decisions produced for the network flow traces of the same class type. We measured Inertia as the averaged squared Euclidean distance computed for each visual explanation map to the visual explanation map centroid of its ground truth class, i.e. $Inertia = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{|C_k|}\sum_{\mathbf{H}_j \in C_k} d\left(\mathbf{H}_j, \hat{\mathbf{H}}_k\right)$, where K is the number of classes, $\mathbf{H}_j$ is a visual explanation map in class C_k and $\hat{\mathbf{H}}_k$ is the centroid of all visual explanation maps with class k.

Average link distance measures the average distance between the visual explanation maps produced for the same class type on both the training set and the testing set, respectively. For each class type k, we determined the mean of the Euclidean distances between each pair made up of the visual explanation map of a network flow trace of class k from the training set and the testing set, respectively, i.e. $d(k) = \frac{1}{|N_k||N'_k|}\sum_{\mathbf{H}_j \in N_k, \mathbf{H}'_j \in N'_k} d\left(\mathbf{H}_j, \mathbf{H}'_j\right)$, where $\mathbf{H}_j$ and $\mathbf{H}'_j$ denote two network flow traces of class k belonging to the training set and testing set, while N_k and N'_k indicate the number of network flow traces of class k recorded in both the training set and the testing set, respectively. The lower the distance $d(k)$, the more robust the explanation of the learned classification model on the predictions produced for unseen network flow traces of class k.



(a) NSL-KDD (b) UNSW-NB15

Figure 4: Inertia of the visual explanation maps produced with decisions yielded on the network flow traces of both the training set and the testing set, respectively.

6.2. Compactness

In principle, we expect a good multi-class classification model to be able to learn the distinctive signature of each class type. This signature should allow us to trigger the same decision process on network flow traces of the same class type. In this study we explored this ability related to the "compactness" of the decision explanations.

Moreover, we analyzed the Inertia of the visual explanation maps that were produced with attention in ROULETTE and SO+A, as well as with Grad-CAM in MO and SO. The results of Inertia, computed separately on the training and testing set of the performed experiments, are reported in Fig. 4. These results show that the visual explanation maps produced through attention always achieve lower Inertia than their respective counterparts produced through Grad-CAM (i.e., the Inertia of ROULETTE is always lower than the Inertia of MO, just as the Inertia of SO+A is always lower than the Inertia of SO). This behaviour, which is observed equally in both the training set and the testing set, assesses that the explanations per class, produced with the intrinsic attention mechanism, are more compact than the explanations per class eventually produced post-hoc. In addition, we note that the attention mechanism, coupled with the multi-output Deep Learning strategy, commonly achieves the lowest Inertia in our study. The only exception is observed with the UNSW-NB15 testing set.

However, the difference between the best Inertia of SO+A and the runner-up Inertia of ROULETTE is small (1159.94 vs. 1168.04). Therefore, this empirical study suggests that a relationship may exist between the higher compactness of the decision explanations produced by ROULETTE and the higher accuracy (assessed in Section 5) of these decisions.

Table 7: Average link distance between pairs of visual explanation maps consisting of a network flow trace of the same class from the training set and the testing set, respectively. The best results are in bold.

Dataset	Class Type	SO	SO+A	MA	ROULETTE
NSL-KDD	**Normal**	1696.76	1775.18	2056.15	**967.72**
	DoS	2684.77	2512.99	2287.34	**1263.12**
	Probe	3157.55	2193.46	2075.19	**1416.23**
	U2R	2206.23	2211.46	3248.76	**936.60**
	R2L	2014.20	2433.27	2901.02	**1038.04**
	Average	2351.90	2225.28	2513.69	**1124.34**
UNSW-NB15	**Normal**	2645.61	2119.41	2660.66	**2051.93**
	Analysis	2361.67	2147.85	2306.63	**2101.82**
	Backdoors	1600.91	1442.60	1822.27	**1343.21**
	DoS	2500.13	2262.99	2421.57	**2150.49**
	Exploits	2259.04	2155.58	2496.73	**1955.91**
	Fuzzers	2492.79	**1536.99**	2140.86	1642.90
	Generic	2691.97	2492.47	2253.04	**1997.52**
	Reconnaissance	1608.62	1558.29	1807.41	**1608.31**
	Shellcode	1979.01	1704.12	2004.33	**1571.44**
	Worms	2320.29	**1921.21**	2571.25	2069.88
	Average	2246.00	1934.15	2248.48	**1849.34**

6.3. Robustness

As a further property, we explored the robustness of decision explanations in relation to their ability to learn a classification model whose decisions are still accurate on new attacks (e.g., variants of existing attacks). The robustness of a classification model is commonly evaluated in terms of the accuracy of classifications produced on unseen (test) data. In particular, our expectation regarding the robustness of the decision explanations is that the explanations of the decisions learned on the training network flow traces are roughly similar to the explanations produced on unseen traces, which may be zero-day attacks. We feel that this property of explanation robustness, if verified, helps predict the correct class of unseen data.

Table 7 reports the results of the Average link distance computed for each single class, as well as the mean of the Average link distance computed on all the classes. The mean results highlight that the overall robustness of the explanations produced with the trainable attention mechanism (ROULETTE and SO+A) is better than the robustness of the counterpart explanations produced with the post-hoc Grad-CAM technique (MO and SO). The lowest Average link distance is measured again with ROULETTE, which also achieves the highest accuracy performance in Section 5. In these results we can see empirical evidence that the increased robustness of decision explanations may be responsible for the increased accuracy of the decisions.

The same conclusions can also be drawn by analyzing the results of Average link distance computed per class type. Indeed, ROULETTE produces the most robust decision explanations in almost any class. The only exceptions are observed for the Fuzzer and Worm classes of UNSW-NB15. However, the differences between the best Average link distance of SO+A and the runner-up Average link distance of ROULETTE are small in both of these classes (1536.99 vs. 1642.90 in Fuzzers and 1921.21 vs. 2069.88 in Worms). In addition, this difference has negligible impact on accuracy. ROULETTE slightly outperforms SO+A on Fuzzers (F1 = 0.38 in ROULETTE vs. F1 = 0.37 in SO+A in Fig. 3), while both ROULETTE and SO+A (as well as SO and MO) fail to recognize all

Worm testing traces (F1 = 0.0 in Fig. 3).

6.4. Separability

Table 8: Macro-F1, Weighted-F1 and A of cluster-based classifications produced on both the training set and testing set. Clusters are computed with k-means on the training set populated with the original images of network flow traffic traces, as well as on the training set populated with the images of the visual explanation maps determined with the attention mechanism.

		Image	k	Macro-F1	Weighted-F1	A
NSL-KDD	Train	Attention	18	0.55	0.94	0.95
		Original	17	0.52	0.92	0.93
	Test	Attention	18	0.41	0.64	0.68
		Original	17	0.35	0.60	0.66
UNSW-NB15	Train	Attention	29	0.35	0.70	0.73
		Original	27	0.30	0.67	0.71
	Test	Attention	29	0.31	0.69	0.68
		Original	27	0.25	0.64	0.63

Finally, we explored how the explanation information synthesized through the attention mechanism can actually help achieve accuracy in separating the network flow traces of different class types. To this end, we compared the effect of information enclosed in:

- The set of visual explanation maps that are intrinsically produced by the neural model with attention for the training network flow traces.
- The original images of the training network flow traces.

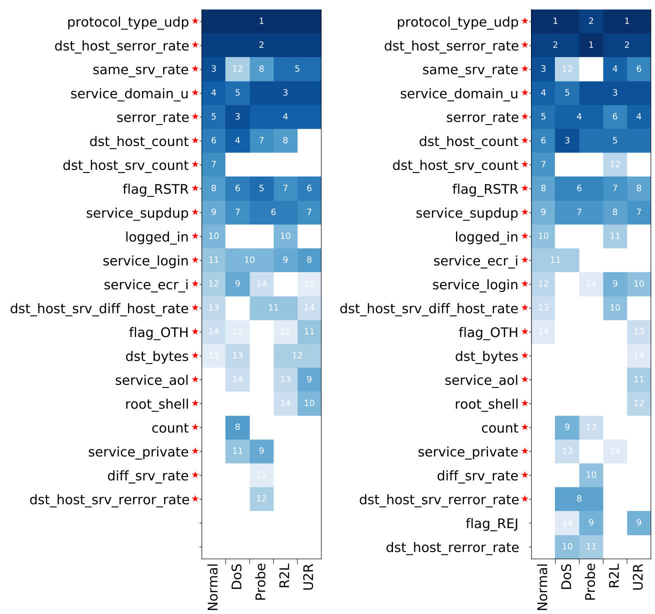
Since the previous analyses had already assessed the performance of the multi-output Deep Learning strategy, we considered here the visual explanation maps produced through the attention mechanism of ROULETTE. We first performed a clustering to isolate distinct prototypes hidden in both the original images and the visual explanations that populate the training set. We used the Elbow method with Inertia to determine the optimal value of k for k-means (Ketchen & Shook 1996). Then, we assigned each cluster centroid to the majority class of the training network flow traces grouped in the cluster. Finally, we measured the accuracy of cluster centroid-based decisions, in order to evaluate how the information contained in both the original images and the visual explanation maps is actually useful for properly separating network flow traces belonging to multiple classes. This evaluation was done on both the training set and the testing set by classifying each network flow trace in the class type associated with the nearest centroid.

The results of Macro-F1, Weighted-F1 and A computed on the decisions based on the cluster centroids are collected in Table 8. These results show empirical evidence that attention can actually gain accuracy as it is able to capture information that separate network flow traces better than the original data.

6.5. Insights into the Classification Explanation

We completed this study by exploring how the attention mechanism of ROULETTE can help reveal important relationships between the flow characteristics of the processed network traffic and the observed categories of the observed intrusions. To this end, we analyzed the information contained in the visual explanation maps produced for both the NSL-KDD and UNSW-NB15 datasets.

The heatmaps in Figs. 5 and 6 depict the ranked average feature relevance of the flow characteristics that "drew" the attention of the neural models learned from both NSL-KDD and UNSW-NB15, respectively. For each dataset, two heatmaps report the top 15 flow characteristics ranked by class on both the training set and the testing set, respectively. Notably, the set of the top-ranked flow characteristics that the neural model mostly attended on in the training set largely overlaps the set of the top-ranked flow characteristics attended on in the testing set. In particular, 21 flow characteristics are enclosed in the top 15 characteristics highlighted both in the training set and the testing set



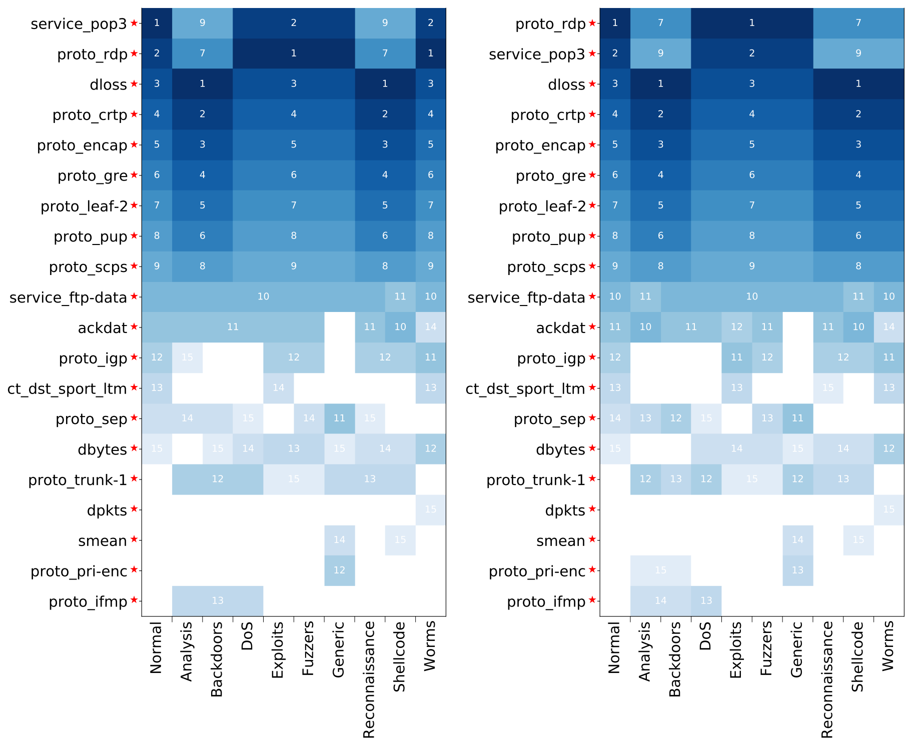
(a) Train (b) Test

Figure 5: NSL-KDD: feature ranking map of the classification model learned with ROULETTE. We plot the ranking (1–15) of the flow characteristics (Y axis), which are ranked in the top 15 positions of the feature ranking determined with the attention mechanism over the various network flow traces grouped by class type (X axis) of both the training set (Fig. 5a) and the testing set (Fig. 5b). A star denotes the features that appear in the top 15 ranking of both the training set and testing set.

of NSL-KDD, while 20 flow characteristics are enclosed in the top 15 characteristics highlighted both in the training set and testing set of UNSW-NB15. Further considerations can be made by analyzing the meaning of the specific flow characteristics mostly attended on in both datasets.

6.5.1. NSL-KDD

Figure 5 shows that both *protocol_type_udp* and *dst_host_serror_rate* are relevant for classifying every class type in NSL-KDD, while the remaining flow characteristics are considered more or less relevant, depending on the specific type of intrusion to be classified.



(a) Train

(b) Test

Figure 6: UNSW-NB15: feature ranking map of the classification model learned with ROULETTE. We plot the rank (1–15) of the flow characteristics (Y axis), which are ranked in the top 15 positions of the feature ranking determined with the attention mechanism over the various network flow traces grouped by class type (X axis) of both the training set (Fig. 6a) and the testing set (Fig. 6b). A star denotes the features that appear in the top 15 ranking of both the training set and testing set.

For example, *count* (i.e., "the number of connections to the same host as the current connection in the past two seconds") is seen as one of the most relevant flow characteristics that the neural model takes into account to detect DoS attacks, while it is less important when detecting other types of attacks. This prominent role of *count* for DoS intrusion detection is consistent with the target of a DoS attack which is to make a computer or network resource unavailable (temporarily or indefinitely) to users by flooding the targeted machine with various connection requests. Similar considerations can be made on *dst_host_count*. This flow characteristic, which counts "the number of connections having the same destination host", also conveys relevant information to detect DoS intrusions. In fact, a DoS attack can be in progress when a server is flooded by sending numerous service requests to the target host.

Further considerations concern the attention paid to *service_ecr_i* to detect DoS intrusions. In (Wang et al. 2020), *service_ecr_i* is recognized as a relevant flow characteristic for the detection of Smurf attacks (a subcategory of DoS intrusions) since, in this type of DoS, the targets are flooded with ECHO REPLAY packets from each host on the broadcast address. Our study reveals that the neural model attends on *service_ecr_i* to detect DoS intrusions. In addition, *root_shell* (which equals 1 if the root shell is obtained; 0 otherwise) contains relevant information for detecting U2R intrusions. U2R is a type of attack in which the attacker tries to access network resources as a normal user, in order to gain full access to the system. A U2R strategy might attempt to gain access to a shell with administrator privilege (root shell). Finally, *dst_host_srv_rerror_rate* and *diff_srv_rate* were considered relevant by the attention mechanism to detect Probe attacks. The relationship between these two flow characteristics and Probe has recently been discussed in (Wang et al. 2020).

6.5.2. UNSW-NB15

Figure 6 shows that various *proto* and *service*-based flow characteristics "attracted" the attention of the neural model in UNSW-NB15. These flow characteristics correspond to the transaction protocol (e.g., RDP, CRTP) used in the network flow trace and the type of connection service (e.g., FTP, HTTP), respectively. While these flow characteristics appear relevant for the detection of several class types, their relevance changes with the class type. For example, *service_pop3*, which is the most relevant flow characteristic for detecting network flow traces in the categories Normal, DoS, Exploits, Fuzzers, Generic and Worms, is slightly relevant for detecting traces in the categories Analysis, Backdoors, Reconnaissance and Shellcode.

In addition, *ackdat*, which refers to the TCP connection setup time (i.e., "the time between the SYN ACK and the ACK response") is relevant for detecting shellcode intrusions, while it becomes less important when detecting other types of attacks. Shellcode, in fact, is an exploiting attack in which the attacker penetrates a piece of code from a shell to control a target machine using the standard TCP/IP socket connections.

Finally, *dpkts*, i.e., "the count of the number of packets from source to destination", is relevant for recognizing worm attacks. We recall that worm attacks are self-replicating computer programs that spread automatically and can flood the Internet in a very short time (Chen et al. 2003).

7. Conclusions

In this paper, we have presented ROULETTE: a system for multi-class classification of network traffic data. The proposed method learns a neural classification model through a multi-output Deep Learning strategy that encompasses both convolution and attention. Extensive experimentation was performed to show the effectiveness of the proposed neural model with attention, quantified in terms of accuracy of classifications, as well as transparency of decisions. In particular, the results obtained indicate that ROULETTE is able to produce decisions that are comparable to (or even more accurate than) decisions produced with competitive, Deep Learning-based approaches. Furthermore, the results of the experimentation highlighted how the good accuracy performance of ROULETTE can also be attributed to specific properties, such as the compactness, robustness, and separability of the produced attention-based explanations. Finally, the attention mechanism helps us to see particular characteristics of network traffic that mainly help to recognize specific intrusion categories. This may support the dissemination of useful information to cyber-defenders, thus reducing the workload in manual analysis.

One limitation of the proposed method is the absence of a specific mechanism for dealing with rare classes. A research direction is to explore data augmentation techniques, in order to reach a balancing condition in the learning stage. For example, GANs have recently helped to increase accuracy in the binary classification of images of network traffic (Andresini, Appice, De Rose & Malerba 2021).

Another limitation is that the proposed method performs the learning stage in a batch fashion, without integrating any concept drift detection mechanism to properly fit the learned model to an evolving streaming environment. This is an issue for all adversary-facing security systems. Recent studies have begun to investigate this problem both in NID applications (Andresini, Pendlebury, Pierazzi, Loglisci, Appice & Cavallaro 2021) and malware detection problems (Pendlebury et al. 2019). These studies propose incremental, semi-supervised security systems to process cyber-data streams by reducing labeling overhead and continuously updating the underlying model as the data characteristics are affected by concept drift.

Finally, recent studies in Computer Vision have achieved amazing results with transformer-based architectures, such as the increasingly popular ViT (Dosovitskiy et al. 2020). The idea behind transformers is to define the layers of the neural network entirely on the attention mechanism. At each layer a new hidden representation is generated for each position in the input data by using multiple attention heads that calculate attention weights for all pairs of positions in the input. Although the image encoding adopted in this study is already robust to arbitrary permutations of features, it is still constrained by the inductive "locality" bias of standard convolutions. The multi-headed self-attention strategy of vision transformers could further improve the generalizability of the model.

CRediT Authorship Contribution Statement

Giuseppina Andresini: Conceptualization, Methodology, Software, Data curation, Investigation, Validation, Visualization, Writing - original draft, Writing - review & editing. **Annalisa Appice**: Conceptualization, Methodology, Investigation, Validation, Supervision, Writing - original draft, Writing - review & editing. **Francesco Paolo Caforio**: Conceptualization, Software, Investigation. **Donato Malerba**: Conceptualization, Project administration, Writing - review & editing. **Gennaro Vessio**: Conceptualization, Methodology, Validation, Writing - original draft, Writing - review & editing, Supervision.

Acknowledgments

We acknowledge the support of the Italian Ministry of University and Research through the project "TALIsMan - Tecnologie di Assistenza personALizzata per il Miglioramento della quAlità della vitA" (Grant ID: ARS01_01116), funding scheme PON RI 2014-2020, as well as the project "Modelli e tecniche di data science per l'analisi di dati strutturati" funded by the University of Bari "Aldo Moro". The authors wish to thank Jerzy Stefanowski for his helpful comments which inspired the evaluation of the separability property and Lynn Rudd for her help in reading the manuscript.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., ... & Zheng, X. (2015), 'TensorFlow: Large-scale machine learning on heterogeneous systems'.
URL: *https://www.tensorflow.org/*

Al-Turaiki, I. & Altwaijry, N. (2021), 'A convolutional neural network for improved anomaly-based network intrusion detection', *Big Data* **9**(3), 233–252.

Alex Kendall, V. B. & Cipolla, R. (2017), Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding, *in* 'Proceedings of the British Machine Vision Conference (BMVC)', BMVA Press, pp. 57.1–57.12.

Andresini, G., Appice, A., De Rose, L. & Malerba, D. (2021), 'Gan augmentation to deal with imbalance in imaging-based intrusion detection', *Future Generation Computer Systems* **123**, 108–127.

Andresini, G., Appice, A. & Malerba, D. (2021*a*), 'Autoencoder-based deep metric learning for network intrusion detection', *Inf. Sci.* **569**, 706–727.

Andresini, G., Appice, A. & Malerba, D. (2021*b*), 'Nearest cluster-based intrusion detection through convolutional neural networks', *Knowledge-Based Systems* **216**, 106798.

Andresini, G., Appice, A., Mauro, N. D., Loglisci, C. & Malerba, D. (2020), 'Multi-channel deep feature learning for intrusion detection', *IEEE Access* **8**, 53346–53359.

Andresini, G., Pendlebury, F., Pierazzi, F., Loglisci, C., Appice, A. & Cavallaro, L. (2021), Insomnia: Towards concept-drift robustness in network intrusion detection, *in* 'Proceedings of the 14th ACM Workshop on Artificial Intelligence and Security', AISec'21, Association for Computing Machinery, New York, USA, p. 111–122.

Antwarg, L., Miller, R. M., Shapira, B. & Rokach, L. (2021), 'Explaining anomalies detected by autoencoders using shapley additive explanations', *Expert Systems with Applications* **186**, 115736.

Bahdanau, D., Cho, K. & Bengio, Y. (2015), Neural machine translation by jointly learning to align and translate, *in* Y. Bengio & Y. LeCun, eds, '3rd International Conference on Learning Representations, ICLR 2015, Conference Track Proceedings'.

Bedi, P., Gupta, N. & Jindal, V. (2020*a*), 'I-SiamIDS: an improved siam-ids for handling class imbalance in network-based intrusion detection systems', *Applied Intelligence* **abs/2009.10940**.

Bedi, P., Gupta, N. & Jindal, V. (2020*b*), ‘Siam-IDS: Handling class imbalance problem in intrusion detection systems using siamese neural network’, *Procedia Computer Science* **171**, 780 – 789.

Bergstra, J., Bardenet, R., Bengio, Y. & Kégl, B. (2011), Algorithms for hyper-parameter optimization, *in* ‘Advances in Neural Information Processing Systems (NeurIPS)’.

Bergstra, J., Yamins, D. & Cox, D. D. (2013), Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, *in* ‘Proc. of the International Conference on Machine Learning (ICML)’.

Berman, D. S., Buczak, A. L., Chavis, J. S. & Corbett, C. L. (2019), ‘A survey of deep learning methods for cyber security’, *Information)* **10**(4), 1–35.

Biecek, P. (2018), ‘Dalex: Explainers for complex predictive models in r’, *Journal of Machine Learning Research* **19**(84), 1–5.

Burkart, N., Franz, M. & Huber, M. F. (2021), Explanation framework for intrusion detection, *in* J. Beyerer, A. Maier & O. Niggemann, eds, ‘Machine Learning for Cyber Physical Systems’, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 83–91.

Caforio, F., Andresini, G., Vessio, G., Appice, A. & Malerba, D. (2021), Leveraging grad-cam to improve the accuracy of network intrusion detection systems, *in* ‘Discovery Science’, Springer International Publishing, pp. 385–400.

Caminero, G., Lopez-Martin, M. & Carro, B. (2019), ‘Adversarial environment reinforcement learning algorithm for intrusion detection’, *Computer Networks* **159**, 96 – 109.

Cao, L., Li, L., Zheng, J., Fan, X., Yin, F., Shen, H. & Zhang, J. (2018), ‘Multi-task neural networks for joint hippocampus segmentation and clinical score regression’, *Multimedia Tools and Applications* **77**(22), 29669–29686.

Castellano, G., Castiello, C., Mencar, C. & Vessio, G. (2020), 'Crowd detection in aerial images using spatial graphs and fully-convolutional neural networks', *IEEE Access* **8**, 64534–64544.

Chen, Z., Gao, L. & Kwiat, K. (2003), Modeling the spread of active worms, *in* 'IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies (IEEE Cat. No.03CH37428)', Vol. 3, pp. 1890–1900 vol.3.

Choudhary, S. & Kesswani, N. (2020), 'Analysis of kdd-cup'99, nsl-kdd and unsw-nb15 datasets using deep learning in iot', *Procedia Computer Science* **167**, 1561–1573. International Conference on Computational Intelligence and Data Science.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S. et al. (2020), 'An image is worth 16x16 words: Transformers for image recognition at scale', *arXiv preprint arXiv:2010.11929* .

Folino, F., Folino, G., Guarascio, M., Pisani, F. & Pontieri, L. (2021), 'On learning effective ensembles of deep neural networks for intrusion detection', *Information Fusion* **72**, 48–69.

Gao, J., Chai, S., Zhang, B. & Xia, Y. (2019), 'Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis', *Energies* **12**(7).

Gao, M., Ma, L., Liu, H., Zhang, Z., Ning, Z. & Xu, J. (2020), 'Malicious network traffic detection based on deep neural networks and association analysis', *Sensors* **20**(5), 1452.

Glorot, X. & Bengio, Y. (2010), 'Understanding the difficulty of training deep feedforward neural networks', *Journal of Machine Learning Research - Proceedings Track* **9**, 249–256.

Glorot, X., Bordes, A. & Bengio, Y. (2011), Deep sparse rectifier neural networks., *in* 'AISTATS', JMLR.org, pp. 315–323.

Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., Zhang, S.-H., Martin, R. R., Cheng, M.-M. & Hu, S.-M. (2021), 'Attention mechanisms in computer vision: A survey', *arXiv preprint arXiv:2111.07624* .

Guo, Y., Cao, H., Bai, J. & Bai, Y. (2019), 'High efficient deep feature extraction and classification of spectral-spatial hyperspectral image using cross domain convolutional neural networks', *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **12**(1), 345–356.

Joshi, G., Walambe, R. & Kotecha, K. (2021), 'A review on explainability in multimodal deep neural nets', *IEEE Access* **PP**, 1–1.

Kasongo, S. & Sun, Y. (2020), 'Performance analysis of intrusion detection systems using a feature selection method on the unsw-nb15 dataset', *J Big Data* **7**(105), 1–20.

Ketchen, D. J. & Shook, C. L. (1996), 'The application of cluster analysis in strategic management research: An analysis and critique', *Strategic Management Journal* **17**(6), 441–458.

Kingma, D. P. & Ba, J. (2014), Adam: A method for stochastic optimization, *in* 'ICLR'.

Komodakis, N. & Zagoruyko, S. (2017), Paying more attention to attention: improving the performance of convolutional neural networks via attention transfer, *in* 'ICLR'.

Lakkaraju, H., Kamar, E., Caruana, R. & Leskovec, J. (2019), Faithful and customizable explanations of black box models, *in* 'Proc. of the 2019 AAAI/ACM Conference on AI, Ethics, and Society', pp. 131–138.

Liu, C., Liu, Y., Yan, Y. & Wang, J. (2020), 'An intrusion detection model with hierarchical attention mechanism', *IEEE Access* **8**, 67542–67554.

Lopez-Martin, M., Carro, B., Sanchez-Esguevillas, A. & Lloret, J. (2017), 'Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IoT', *Sensors* **17**(9), 1–17.

Lundberg, S. M. & Lee, S.-I. (2017), A unified approach to interpreting model predictions, *in* 'Proceedings of the 31st International Conference on Neural Information Processing Systems', NIPS'17, Curran Associates Inc., pp. 4768–4777.

Mane, S. & Rao, D. (2021), 'Explaining network intrusion detection system using explainable AI framework', *arXiv preprint arXiv:2103.07110* .

Marino, D. L., Wickramasinghe, C. S. & Manic, M. (2018), An adversarial approach for explainable ai in intrusion detection systems, *in* 'IECON 2018 - 44th Annual Conference of the IEEE Industrial Electronics Society', pp. 3237–3243.

Moustafa, N. & Slay, J. (2015), Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set), *in* '2015 Military Communications and Information Systems Conference (MilCIS)', pp. 1–6.

Naseer, S., Saleem, Y., Khalid, S., Bashir, M. K., Han, J., Iqbal, M. M. & Han, K. (2018), 'Enhanced network anomaly detection based on deep neural networks', *IEEE Access* **6**, 48231–48246.

Pendlebury, F., Pierazzi, F., Jordaney, R., Kinder, J. & Cavallaro, L. (2019), TESSERACT: Eliminating Experimental Bias in Malware Classification across Space and Time, *in* '28th USENIX Security Symposium', USENIX Association. USENIX Sec.

Phaisangittisagul, E. (2016), An analysis of the regularization between l2 and dropout in single hidden layer neural network, *in* '2016 7th International Conference on Intelligent Systems, Modelling and Simulation (ISMS)', pp. 174–179.

Sarhan, M., Layeghy, S. & Portmann, M. (2021), 'An explainable machine learning-based network intrusion detection system for enabling generalisability in securing iot networks', *CoRR* **abs/2104.07183**.

Sartor, G. & Lagioia, F. (2020), *The impact of the General Data Protection Regulation (GDPR) on artificial intelligence.*

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. & Batra, D. (2017), Grad-CAM: Visual explanations from deep networks via gradient-based localization, *in* 'Proceedings of the IEEE international conference on computer vision', pp. 618–626.

Selvaraju, R. R., Cogswell, M., Das, A., Vedantam, R., Parikh, D. & Batra, D. (2020), 'Grad-cam: Visual explanations from deep networks via gradient-based localization', *Int. J. Comput. Vis.* **128**(2), 336–359.

Sovilj, D., Budnarain, P., Sanner, S., Salmon, G. & Rao, M. (2020), 'A comparative evaluation of unsupervised deep architectures for intrusion detection in sequential data streams', *Expert Systems with Applications* **159**, 113577.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. (2014), 'Dropout: A simple way to prevent neural networks from overfitting', *Journal of Machine Learning Research* **15**(56), 1929–1958.

Szczepański, M., Choraś, M., Pawlicki, M. & Kozik, R. (2020), Achieving explainability of intrusion detection system by hybrid oracle-explainer approach, *in* '2020 International Joint Conference on Neural Networks (IJCNN)', pp. 1–8.

Tang, C., Luktarhan, N. & Zhao, Y. (2020), 'SAAE-DNN: Deep learning method on intrusion detection', *Symmetry* **12**(10), 1–20.

Tavallaee, M., Bagheri, E., Lu, W. & Ghorbani, A. A. (2009), A detailed analysis of the KDD CUP 99 data set, *in* 'CISDA', IEEE, pp. 1–6.

Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A. & Venkatraman, S. (2019), 'Deep learning approach for intelligent intrusion detection system', *IEEE Access* **7**, 41525–41550.

Wali, S. & Khan, I. (2021), 'Explainable AI and random forest based reliable intrusion detection system'.

Wang, M., Zheng, K., Yang, Y. & Wang, X. (2020), 'An explainable machine learning framework for intrusion detection systems', *IEEE Access* **8**, 73127–73141.

Warnecke, A., Arp, D., Wressnegger, C. & Rieck, K. (2020), Evaluating explanation methods for deep learning in security, *in* '2020 IEEE European Symposium on Security and Privacy (EuroS&P)', IEEE, pp. 158–174.

Xu, D., Shi, Y., Tsang, I. W., Ong, Y.-S., Gong, C. & Shen, X. (2019), 'Survey on multi-output learning', *IEEE Transactions on Neural Networks and Learning Systems* **31**(7), 2409–2429.

Xu, F., Uszkoreit, H., Du, Y., Fan, W., Zhao, D. & Zhu, J. (2019), Explainable ai: A brief survey on history, research areas, approaches and challenges, *in* J. Tang, M.-Y. Kan, D. Zhao, S. Li & H. Zan, eds, 'Natural Language Processing and Chinese Computing', Springer International Publishing, Cham, pp. 563–574.

Yang, J., Li, T., Liang, G., He, W. & Zhao, Y. (2019), 'A simple recurrent unit model based intrusion detection system with DCGAN', *IEEE Access* **7**, 83286–83296.

Yin, C., Zhu, Y., Fei, J. & He, X. (2017), 'A deep learning approach for intrusion detection using recurrent neural networks', *IEEE Access* **5**, 21954–21961.

Zhao, H., Kong, X., He, J., Qiao, Y. & Dong, C. (2020), Efficient image super-resolution using pixel attention, *in* 'European Conference on Computer Vision', Springer, pp. 56–72.

Zhao, P., Fan, Z., Cao, Z. & Li, X. (2022), 'Intrusion detection model using temporal convolutional network blend into attention mechanism', *International Journal of Information Security and Privacy* **16**(1), 1–20.