Preliminary Results in Using Attention for Increasing Attack Identification Efficiency

Ahmad, Tanwir; Truscan, Dragos; Vain, Jüri

Published in:
2023 IEEE International Conference on Software Testing, Verification and Validation

DOI:
10.1109/ICSTW58534.2023.00038

Published: 01/05/2023

Document Version
Accepted author manuscript

Document License
Publisher rights policy

Preliminary Results in Using Attention for Increasing Attack Identification Efficiency

Tanwir Ahmad*, Dragos Truscan*, and Jüri Vain*†
*Faculty of Science and Engineering, Åbo Akademi University, Finland
†High-assurance Software Laboratory, Tallinn University of Technology, Estonia
{tanwir.ahmad, dragos.truscan}@abo.fi, juri.vain@ttu.ee

***Abstract*—In previous work, we proposed an end-to-end intrusion early detection system to identify network attacks in real-time before they are complete and could cause more damage to the system under attack. To implement the approach, we have used a deep neural network which was trained in a supervised manner to extract relevant features from raw network traffic in order to classify network flows into different types of attacks. In this work, we discuss the initial results of the benefits that an attention mechanism brings to the classification performance and the capacity of the network to detect attacks earlier. We empirically evaluate our approach on the CICIDS2017 dataset. Preliminary results show that the attention mechanism improves both the balanced accuracy of the classifier as well as the early detection of attacks.**

***Index Terms*—Early IDS, Attention Mechanism, Deep learning, Convolution Neural Network**

I. Introduction

Early detection of attack (or intrusion) is desirable in the cyber-security domain to prevent network attacks before they could cause any more damage to the system. Based on the earlier classification results of the traffic, automatic countermeasures can be deployed in a timely manner.

In our previous work [1], we proposed an end-to-end signature-based intrusion detection system (IDS) that was able to detect network attacks as early as possible with a high degree of accuracy. A *Deep Neural Network* (DNN) [2]) was used to extract relevant features from network raw traffic. The features were used for the early classification of ongoing attacks. In order to evaluate the capability of the IDS to detect the attacks before they are completed, we have also proposed, in the same work, a new metric called *earliness*. The metric measures the ratio of information of an attack needed to classify it as a given attack type.

In this work, we extend our previous approach by incorporating an attention mechanism in order to improve classification performance and the earliness of the approach. The attention mechanism is intended to allow the DNN to focus on more relevant packets of the network traffic data and, consequently, to improve the classification performance and the earliness of the approach. We evaluate the applicability of our approach by benchmarking its performance using the CICIDS2017 dataset [3] from the web application domain.

The rest of the paper is structured as follows: Section II describes our approach. In Section III, we empirically evaluate our approach. Section IV presents an overview of the related work. Finally, Section V draws conclusions and discusses future work.

II. Approach

Our intrusion detection approach works at the network packet level, by analysing network flows. A *network flow* is a bidirectional sequence of packets exchanged between two endpoints (e.g., a web server and a client) during a certain time interval with some common flow properties [4] such as source and destination IP addresses, source and destination port numbers, and protocol type. In our work, we define a network flow as a sequence of T ordered packets, where T represents the length of a complete flow. We denote a flow as:

$$\mathbf{f} = \{p_1, p_2, ..., p_T\},\ \forall\ p_i \in \mathbb{R}^D \wedge 1 \leq i \leq T \qquad (1)$$

where D is the size (or length) of a packet.

Our approach, implemented by the EARLY tool [5], has four main components (see Figure 1):

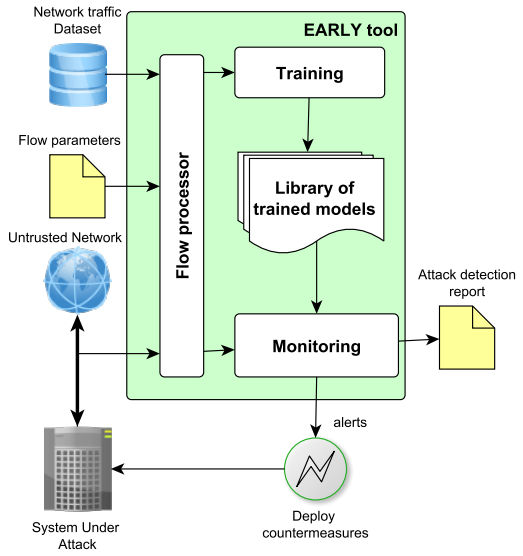

Fig. 1. Overview of EARLY tool architecture

- a **flow processing module** is used to extract flows from network traffic based on given flow parameters, and provides the flows to other modules, either for training or for monitoring;

- a **training module** is used to train neural network models using various datasets for different application domains (e.g, web, IoT, etc.);
- a **library of attack models** contains trained models for different attack types and application domains;
- a **monitoring module** is used to monitor network traffic using a trained model from the library. Whenever attacks are detected, this module will trigger alerts based on predefined triggers in order to deploy automatic countermeasures.

The focus of this paper is the training module, while we defer further details about the other modules to the previous publication [1]. The training module uses a Convolution Neural Network (CNN) [6], a type of DNN, to extract a good internal representation of network flows. The CNNs are commonly used to extract relevant features from grid-shaped input data such as images and sequences. They are well capable of modeling the spatial and temporal dependencies in the data by learning relevant convolution filters (i.e., a set of grid-shaped weights or trainable parameters). The CNNs have a smaller number of trainable parameters than other types of artificial neural networks such as fully-connected networks [6]. Therefore, they train faster and are less likely to overfit the training data than the fully-connected networks resulting in a better generalization.

A convolution layer is composed of several convolution filters where each filter is used to extract a certain feature from the input data. In the previous work, we used a 1D-CNN layer with kernel size B to extract a set of relevant feature vectors from the network raw traffic. We define the set of feature vectors: $m = \{\mathbf{m}_1, ..., \mathbf{m}_T\}, \mathbf{m} \in \mathbb{R}^B$. In the current work, we extend our approach by incorporating an attention mechanism (AM). The attention mechanism is a technique used in deep learning to selectively focus on certain parts of an input, rather than processing the entire input uniformly [7]. It allows the model to dynamically weigh the importance of different parts of the input, and to emphasize the most relevant information. It has often been used in natural language processing tasks, such as machine translation and text summarization, where the model needs to focus on specific words in a sentence to accurately understand the meaning.

Our attention mechanism is based on the mechanism proposed by Bahdanau et al. [8]. We compute the context vector $c \in \mathbb{R}^B$ that combines the feature vectors (corresponding to the packets in a given flow) with respect to their importance for the class prediction. The vector is calculated by taking a weighted sum of the feature vectors, where the weights are determined by the attention scores α_t:

$$c = \sum_{t=1}^{T} \alpha_t \mathbf{m}_t \tag{2}$$

For every feature vector $\mathbf{m}_t$, the attention mechanism generates a positive weight α_t which can be considered either as the probability that the feature vector of packet t is the right place to focus on for predicting the class of the given flow or as the relative importance to give to the feature vector in combing with the others. The weight α_t of each feature vector $\mathbf{m}_t$ is calculated as:

$$\alpha_t = \frac{\exp(s_t)}{\sum_{t=1}^{T} \exp(s_t)} \tag{3}$$

$$s_t = att(\mathbf{m}_t) \tag{4}$$

where att is an attention sub-model for which we use a fully connected layer. The latter is jointly trained with the rest of the model. A graphical representation of the neural network architecture will be provided in a concrete example in the following section.

III. Evaluation

We evaluate the performance of our approach by answering the following research questions:

- RQ1: How does the attention mechanism affect the classification performance of our approach in terms of classifying the complete flows (i.e., flows with all the packets)?
- RQ2: Does the attention mechanism make the approach more effective in identifying the class of a given flow in real-time by inspecting only the first few packets of the flow?

A. Dataset

We use CICIDS2017 [3] dataset to evaluate the effectiveness of our approach. The dataset is composed of normal and seven types of attack flows (e.g., Hearbleed, Botnet, Web) along with the network packets corresponding to the flows. We use a specific part of the dataset that was captured on Thursday, July 6, 2017 and contains network flows related to the following web attacks: (1) *SQL Injection:* an attacker provides a string of SQL commands to be injected in the database; (2) *Cross-Site Scripting (XSS):* an attacker injects a script into the web application code; (3) *Brute Force:* an attacker tries a list of passwords to find the administrator's password. Table I lists the number of flows and the average flow length (i.e., number of packets) per class in the Thursday dataset.

TABLE I
Labeled Flow Dataset

Class	Number of Flows	Average Flow length
Normal	27 129	124.39
Brute Force	1 507	18.43
XSS	652	11.48
SQL Injection	21	5.71

We have observed that the header and payload length of 99% of packets in the dataset are less or equal to 40 and 356 bytes, respectively. To handle the packets with different header and payload lengths, we crop or pad them with zeros at the end to 50 and 400 bytes, respectively. We scale all the packet bytes between 0 and 1 by dividing them by 255. In practice, scaling the input data helps machine learning algorithms converge faster [9].

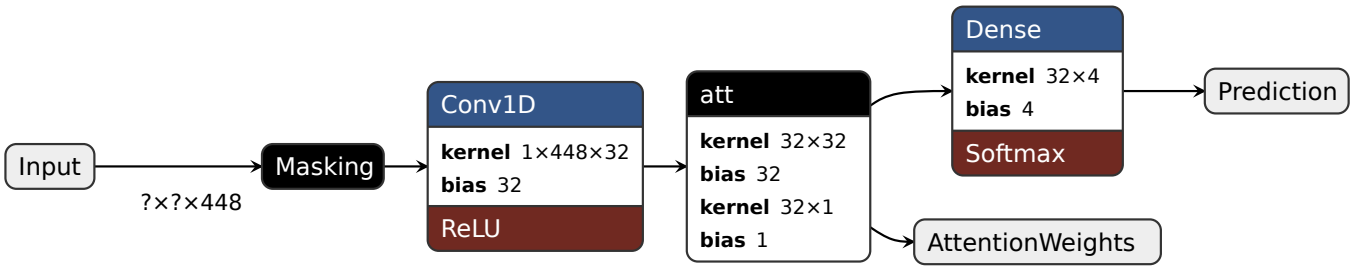

Fig. 2. *EAtt* model architecture

B. Model Architecture

Our neural network model (shown in Figure 2) has a 1D CNN layer, *Conv1D*, with kernel size 32, valid padding, *ReLU* activation, and bias. The attention model *att* has a fully connected layer of size 32. This sub-model takes the set of feature vectors as input from the convolution layer and returns the context vector of size 32 and the attention weights. The context vector is then provided as input to a *Dense* layer with *Softmax* activation to obtain a probability distribution for each class. Based on the probability distribution and the classification threshold, we make the final predictions. The total number of trainable parameters of the model is 15 589.

For evaluation purposes, we denote the model from our previous work [1] as *EGlo* and the model in Figure 2 as *EAtt*. The difference is that the former uses global average pooling to combine the feature vectors instead of the attention mechanism. We use the *EGlo* model for baseline comparison.

C. Evaluation Metrics

Evaluating the classification performance of a machine learning-based approach on an imbalanced dataset is a challenging task [10]. The majority of the existing intrusion detection tools using machine learning have reported the performance of their approaches using traditional metrics such as accuracy and F1-score [11]. These metrics are designed to evaluate the performance of a classifier on balanced datasets. They do not work well when there is a large imbalance in the distribution of the classes in the dataset [10]. Therefore, we evaluate our tool using the following metrics:

- *Precision* calculates the percentage of instances identified as positive that were correctly classified.
- *Recall* (i.e., also known as detection rate) computes the percentage of actual positive instances that were correctly classified.
- *False positive rate* (FPR) (i.e., also known as false alarm rate) estimates the proportion of negative observations wrongly predicted as positive over the total number of negative observations.
- *Balanced Accuracy* (BA) is the arithmetic mean of recall obtained on each class.
- *Bookmaker Informedness* (BM) is defined as the probability that the classifier will make a correct decision as opposed to random guessing.
- *Prediction time* indicates the average time needed by the tool to detect a security attack.
- *Earliness* metric measures the earliness of our approach. It specifies how early the correct class of a flow can be predicted ahead of the end of the flow. We define the *earliness* as:

$$Earliness = \begin{cases} \dfrac{T-t}{T-1} & \text{if } T > 1 \\ 1 & \text{if } T = 1 \end{cases} \quad (5)$$

  where t is the minimum number of packets required to correctly predict the class of a given flow, and T is the total number of packets in the flow. Since this metric aims to evaluate the earliness of the prediction instead of its quality, this metric is only applied to those flows that are correctly classified.

All the metrics mentioned above can have values between 0 and 1. Higher values of precision, recall, BA, and BM and lower values of FPR indicate better classification performance of a classifier. The earliness value lies between 0 and 1, with extreme values 1 and 0 reached in case a classifier can accurately classify a given flow by analyzing only the first packet and all the packets of the flow, respectively. In addition to the above metrics, we also report the *macro* and *weighted average* values of precision, recall, and FPR metrics. The macro average value (i.e., arithmetic mean) provides an unbiased estimate of the classification performance of a model across all classes, regardless of their frequency in the dataset. On the other hand, the weighted average value takes into account the imbalance in the dataset by giving more weight to the metrics of the classes with more instances.

D. RQ1: Classification Performance

The objective of this research question is to investigate how the classification performance of our approach is affected by the attention mechanism. To answer this question, *EAtt* and *EGlo* models are trained and evaluated against the independent test set that is extracted from the dataset. We split the dataset into two subsets using the ratio 0.7:0.3: training and test set (see Table II). As one can notice, there is a large imbalance in the distribution of the flow classes; for example, roughly 1291.86 times more flows belong to the normal class than the SQL Injection class. We have used stratified sampling [12] to split the original dataset. Unlike random sampling, stratified

sampling creates the splits by maintaining the same percentage for each class as in the complete dataset.

TABLE II
TRAINING AND TEST DATASETS

Class	Training	Test
Normal	18 990	8 139
Brute Force	1 055	452
XSS	456	196
SQL Injection	15	6

We use sample weighting during the training process to improve the classification accuracy of our classifier. The weight assigned to each sample or flow is determined based on its class and is computed as the inverse of the total number of samples belonging to that class in the training data. The goal is to get the classifier to focus more on samples from the classes that are underrepresented. The cost-sensitive learning method has been demonstrated to be more effective than the sampling-based method in many different areas for dealing with imbalanced datasets [13].

We used 10-fold cross-validation on the training dataset to fine-tune the hyper-parameter values and model selection. For statistical reasons, the evaluation procedure is repeated 20 times, and every time, we randomly shuffle the dataset to remove any ordering bias before splitting it into training and test set. The total number of training epochs is set to 50. We have augmented the training dataset using the segmentation rate $s_r = 0.1$. We set the *classification threshold* to 0, which means that the final class of a flow is a class that has a higher probability than other classes.

An ideal IDS would have a 1 recall and a 0 FPR for every type of network flow. This means that it would be able to accurately identify all flows without missing any potential attacks. However, it is challenging to achieve such a perfect IDS in real-time environment due to the high complexity and large volume of network traffic. Therefore, building a flawless IDS in practice is often not feasible. Table III compares the achieved classification performance of our approach using two different neural network models on the test set. The bold values in the table show the best result for each metric between the models. One can notice that, on average, the *EAtt* model provides better recall or high detection rate at lower FPR than the *EGlo* model which is highly desirable for an IDS. Further, the *EAtt* model has achieved higher balanced accuracy (i.e., 0.853) than the *EGlo* model (i.e., 0.803). Overall, the results indicate that the attention mechanism has improved the classification performance.

E. RQ2: Earliness Performance

This research question aims to study the earliness performance of our approach with the attention mechanism. In our opinion, a real-time IDS should satisfy the following two requirements. First, the IDS should be able to process the data (i.e., network packets) as fast as it is being produced under normal circumstances. Second, the minimum number of packets required to accurately predict the class of a given flow (MNP) should be less than the total number of packets in the flow.

To answer this question, we ran a replay session where we reproduced the network traffic captured in the dataset against our approach, using two different model architectures: *EAtt* and *EGlo*, to emulate a real-time environment. Our approach and the software that replayed the traffic ran on different machines. Each machine featured an Intel Core i9-10900X CPU, 64 GB of memory, RTX 3090 graphics card, and Ubuntu 20.04 Operating System. The machines were connected via a 1Gb Ethernet connection in an isolated environment to reduce network latency.

The replay session ran for 29 146 seconds and re-transmitted 9 322 025 packets. We configure the packet filtering module to forward only those packets whose destination or source port is 80 since we are interested in detecting web attacks. However, our approach allows any other port or even combinations of ports to be selected.

During the replay session, we monitor packet inter-arrival times, processing times required by our approach to make predictions, and the minimum number of packets required (MNP) to predict a flow class accurately. The average packet inter-arrival time of all the packets and the filtered packets were 3.13 and 15.96 milliseconds, respectively. On average, our approach with the *EAtt* model was able to make a prediction in 0.07 milliseconds per packet, which is 0.03 milliseconds more than the *EGlo* model. Nevertheless, the prediction time per packet is still less than the average packet inter-arrival time. This shows that approach is able to process the network packets faster than they are being produced and satisfies the first requirement. Table IV shows the earliness, MNP, and the average flow length per class per model. The results show that *EAtt* model improves the earliness performance.

IV. RELATED WORK

A couple of related works which employed attention-based mechanisms reported improvements in both training from imbalance data sets and detection accuracy.

For instance, the authors of [14] use a long short term memory (LSTM) combined with an attention mechanism to implement a dynamic network anomaly detection system. Their experimental results show that by adding the attention mechanism, the classification accuracy was improved compared to other machine learning algorithms. Also, they report some benefits with respect to training from imbalanced data.

The authors of [15] suggest an approach in which a hierarchical attention based gated recurrent unit model is used to process the network flows. The approach relies on the flow-based statistical features extracted by analyzing all the packets in a given flow. In contrast, our approach extracts relevant features from raw network traffic data for early detection of attacks by analyzing the partial information already available in the flows.

Finally, the work reported in [16] combines a multi-head attention network and a gated convolution network, together

TABLE III
CLASSIFICATION PERFORMANCE COMPARED BETWEEN EGLO (MODEL WITH GLOBALAVGPOOLING) AND EATT (MODEL WITH ATTENTION)

Class	Precision		Recall		FPR		BM	
	EGlo	EAtt	EGlo	EAtt	EGlo	EAtt	EGlo	EAtt
Normal	0.996	**0.997**	0.944	**0.964**	0.054	**0.033**	0.891	**0.931**
Brute force	**0.720**	0.641	0.828	**0.881**	0.051	**0.028**	0.778	**0.853**
XSS	0.754	**0.820**	**0.911**	0.886	0.008	**0.005**	**0.904**	0.881
SQL Injection	**0.343**	0.094	0.528	**0.692**	**0.003**	0.008	0.525	**0.692**
Macro average	**0.703**	0.638	0.803	**0.856**	0.029	**0.019**	0.774	**0.837**
Weighted average	**0.976**	0.974	0.937	**0.958**	0.052	**0.032**	0.885	**0.925**

TABLE IV
EARLINESS METRIC AND THE AVERAGE MINIMUM NUMBER OF PACKETS REQUIRED (MNP) TO PREDICT THE FLOW CLASS

Class	Avg. flow length	Earliness		MNP	
		EGlo	EAtt	EGlo	EAtt
Normal	124.39	**0.994**	0.992	**1.74**	1.99
Brute force	18.43	0.937	**0.964**	2.10	**1.63**
XSS	11.48	0.888	**0.923**	2.17	**1.81**
SQL Injection	5.71	0.72	**0.796**	2.32	**1.96**

with a Structure-Level Segmentation (SLS) method for structuring the input data, with the goal of detecting malicious requests in web applications. The approach shows a performance improvement over two other approaches, but from the experiments, it is not clear if the improvement is due to the use of the attention mechanism or of the SLS method.

Laghrissi et al., [17] proposed an IDS based on an LSTM neural network and attention mechanism focusing on the most relevant features of network traffic. The approach relies on the flow-based statistical features extracted by analyzing all the packets in a given flow. In contrast, our approach extracts relevant features from raw network traffic data that can be used to reliably detect attack flows by analyzing the partial information already available of the flows. Further, in our opinion, the author can utilize a standard feed-forward neural network to process a fixed-size one-dimensional vector as an input instead of LSTM. Because an LSTM network and the attention mechanism are mostly used to process long sequential data. For instance, in our case, we use 1D-CNN to process variable-length network flows (i.e., a set of raw network packets) and the attention mechanism to dynamically focus on the most relevant packets of the given flow.

These studies are in line with our findings and show that further investigating the use of attention mechanisms is relevant.

V. CONCLUSION

In this work, we presented the initial results of using an attention-based CNN model for the early detection of network attacks. The attention mechanism allows the model to dynamically focus on the most relevant packets of the flow and this results in improved classification performance and earliness.

We empirically evaluated our approach on the CICIDS2017 dataset. The results show that the attention mechanism improves both the classification performance of the classifier as well as the early detection of attacks. For our future work, we aim to investigate how the attention mechanism can be used to provide insights to the network administrator into the model's reasoning behind its prediction.

ACKNOWLEDGMENT

This work was made possible with funding from the European Union's Horizon 2020 research and innovation programme, under grant agreement No. 957212 (VeriDevOps). The opinions expressed and arguments employed herein do not necessarily reflect the official views of the funding body.

REFERENCES

[1] T. Ahmad, D. Truscan, J. Vain, and I. Porres, "Early detection of network attacks using deep learning," in *15th IEEE International Conference on Software Testing, Verification and Validation Workshops ICST Workshops 2022*. IEEE, 2022.

[2] Y. Bengio, "Deep learning of representations for unsupervised and transfer learning," in *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, ser. Proceedings of Machine Learning Research, vol. 27. PMLR, 02 Jul 2012, pp. 17–36. [Online]. Available: http://proceedings.mlr.press/v27/bengio12a.html

[3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proceedings of the 4th International Conference on Information Systems Security and Privacy, ICISSP 2018*. SciTePress, 2018, pp. 108–116.

[4] B. Claise, B. Trammell, and P. Aitken, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information," *RFC*, vol. 7011, pp. 1–76, 2013.

[5] T. Ahmad and D. Truscan, "Early tool," https://github.com/VeriDevOps/Earlytool, 2022.

[6] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 1998.

[7] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. [Online]. Available: http://arxiv.org/abs/1706.03762

[8] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *3rd International Conference on Learning Representations, ICLR 2015*, Y. Bengio and Y. LeCun, Eds., 2015. [Online]. Available: http://arxiv.org/abs/1409.0473

[9] J. Sola and J. Sevilla, "Importance of input data normalization for the application of neural networks to complex industrial problems," vol. 44, pp. 1464–1468, 1997.

[10] Q. Zhu, "On the performance of matthews correlation coefficient (MCC) for imbalanced dataset," *Pattern Recognit. Lett.*, vol. 136, pp. 71–80, 2020.

[11] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, p. e4150, 2021. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.4150

[12] W. G. Cochran, *Sampling Techniques, 3rd Edition*. John Wiley, 1977.

[13] N. Thai-Nghe, Z. Gantner, and L. Schmidt-Thieme, "Cost-sensitive learning methods for imbalanced data," in *International Joint Conference on Neural Networks, IJCNN 2010, Barcelona, Spain, 18-23 July, 2010*. IEEE, 2010, pp. 1–8.

[14] P. Lin, K. Ye, and C.-Z. Xu, "Dynamic network anomaly detection system by using deep learning techniques," in *Cloud Computing – CLOUD 2019*. Cham: Springer International Publishing, 2019, pp. 161–176.

[15] X. Liu and J. Liu, "Malicious traffic detection combined deep neural network with hierarchical attention mechanism," *Scientific Reports*, vol. 11, no. 1, p. 12363, Jun 2021. [Online]. Available: https://doi.org/10.1038/s41598-021-91805-z

[16] J. Li, Y. Fu, J. Xu, C. Ren, X. Xiang, and J. Guo, "Web application attack detection based on attention and gated convolution networks," *IEEE Access*, vol. 8, pp. 20 717–20 724, 2020.

[17] F. Laghrissi, S. Douzi, K. Douzi, and B. Hssina, "IDS-attention: an efficient algorithm for intrusion detection systems using attention mechanism," *J. Big Data*, vol. 8, no. 1, p. 149, 2021.