Received 13 May 2022, accepted 4 July 2022, date of publication 12 July 2022, date of current version 18 July 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3190008

RESEARCH ARTICLE

A Gated Recurrent Unit Deep Learning Model to Detect and Mitigate Distributed Denial of Service and Portscan Attacks

DANIEL M. BRANDÃO LENT[1], MATHEUS P. NOVAES[2], LUIZ F. CARVALHO[3], JAIME LLORET[4], (Senior Member, IEEE), JOEL J. P. C. RODRIGUES[5,6], (Fellow, IEEE), AND MARIO LEMES PROENÇA, JR.[1]

[1]Computer Science Department, State University of Londrina, Londrina 86057-970, Brazil
[2]Electrical Engineering Department, State University of Londrina, Londrina 86057-970, Brazil
[3]Computer Engineering Department, Federal Technology University of Paraná, Apucarana 86812-460, Brazil
[4]Integrated Management Coastal Research Institute, Universitat Politecnica de Valencia, 46730 Valencia, Spain
[5]College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266555, China
[6]Instituto de Telecomunicações, Covilhã 6201-001, Portugal

Corresponding author: Mario Lemes Proença, Jr. (proenca@uel.br)

This work was supported in part by the Fundação Araucária; in part by the National Council for Scientific and Technological Development (CNPq) of Brazil under Project 310668/2019-0 and Project 313036/2020-9; in part by the Superintendência Geral de Ciência, Tecnologia e Ensino Superior (SETI)/Fundação Araucária due to the Concession of Scholarships; in part by the Ministerio de Economía y Competitividad through the Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento, under Grant TIN2017-84802-C2-1-P; and in part by the Fundação para a Ciência e a Tecnologia (FCT)/Ministério da Ciência, Tecnologia e Ensino Superior (MCTES) through National Funds and when Applicable Co-Funded the European Union (EU) Funds under Project UIDB/50008/2020.

ABSTRACT Nowadays, it is common for applications to require servers to run constantly and aim as close as possible to zero downtime. The slightest failure might cause significant financial losses and sometimes even lives. For this reason, security and management measures against network threats are fundamental and have been researched for years. Software-defined networks (SDN) are an advancement in network management due to their centralization of the control plane, as it facilitates equipment setup and administration over the local network. However, this centralization makes the controller a target to denial of service attacks (DoS). In this study, we aim to develop a network anomaly detection and mitigation system that uses gated recurrent unit (GRU) neural networks combined with fuzzy logic. The neural network is trained to forecast future traffic, and anomalies are detected when the forecasting fails. The system is designed to operate in software-defined networks since they provide network flow information and tools to manage forwarding tables. We also demonstrate how the neural network's hyperparameters affect the detection module. The system was tested using two datasets: one with emulated traffic generated by the data communication and networking research group called Orion, from computer science department at state university of Londrina, and CICDDoS2019, a well-known dataset by the anomaly detection community. The results show that GRU networks combined with fuzzy logic are a viable option to detect anomalies in SDN and possibly in other anomaly detection applications. The system was compared with other deep learning techniques.

INDEX TERMS Anomaly detection, deep learning, fuzzy logic, gated recurrent unit, software-defined networks.

I. INTRODUCTION

Over the years, it has become more common for applications that require the Internet to be developed with availability as one of its main principles. It is common for those applications to require multiple servers to respond users' requests, demanding complex infrastructure to share resources and forward requests. With the constant growth of networks, their management has become increasingly complex, making them harder to upgrade and maintain due to the incompatibility

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau.

of heterogeneous hardware and software in addition to other possible issues [1], [2].

Software-defined networks (SDN) present a solution to management problems by having a centralized control plane, separated from network devices, that coordinates a data plane [2]. From the control plane, the network set of rules and policies are established to the data plane, which is only responsible for the data forwarding while it collects traffic statistics from the network [3]. Therefore, the control plane works as an interface for the applications that manage the network to control the hardware devices through a protocol (e.g., OpenFlow [4]), thus making possible more straightforward management of the network. SDN traffic is represented in the form of flows which are groups of packets exchanged between two hosts using the same port during a period of time.

With the interface created through the control plane, management applications such as resource allocation and anomaly detection can be used in the network [5], [6]. In this paper, a distributed denial of service (DDoS) and portscan attack detection and mitigation system was developed to work using SDN to collect flow information from the network and block traffic from specific malicious Internet protocol (IP) addresses. Thus, SDN is fundamental to simplify the flow collection and mitigation of attacks due to the control plane centralization mentioned above [7].

Denial of service attacks (DoS) as well as DDoS attacks can be especially harmful to SDN due to the centralization of the control plane since every new entry in a forwarding table needs to be evaluated by the network's controller, which can get overwhelmed by numerous requests [1], [8]. If the controller is not able to respond all these requests, the network switches buffers can get saturated, resulting in the switch not being able to handle legitimate flows from known sources. Thus, intrusion detection systems (IDS) have an essential role in SDN. The distributed characteristic from this type of attack makes it harder to mitigate since various hosts need to be recognized and blocked by the controller [9]. Portscan attacks are the other anomaly studied in this article since they can act as an enabler for DDoS and other threats. This type of attack is used to gather information about open ports from hosts and can result in exposure of vulnerabilities [10].

Intrusion detection techniques can be signature-based, in which the system keeps a signature of previous attack behavior and uses it to match with actual network traffic [11]. Since it needs previous information of an attack to work, these systems have a hard time detecting unknown attacks that do not match any signature. Another technique is called anomaly-based, in which the system uses a baseline to match with normal network behavior, raising an alarm when anomalous traffic is detected [12]. This type of system can be used to detect anomalies but may not be as accurate in identifying the type of the ongoing attack as the signature-based technique. Contrastingly, anomaly-based systems are more likely to detect novel or unknown attacks [13], [14]. In this work, an anomaly-based system was used.

Deep neural networks (DNN) were used to draw the baseline of the network traffic in this study due to its capabilities in pattern recognition [15]. DNN have been used in numerous pattern recognition solutions such as image classification [16], anomaly detection [17], and time series prediction [18]. Neural networks have evolved into different types of configurations as a form of improving their performance on different types of problems. Recurrent neural networks (RNN) have been used in sequential problems where previous iterations affect future ones such as time series [19].

A subtype of RNN is the gated recurrent Unit (GRU) network, which utilizes gates to influence what information to keep and discard. With reset and update gates, the network can learn what information is worth remembering while discarding useless details [20]. When compared to other RNN, such as long short-term memory networks (LSTM), GRU presents an advantage of having less trainable parameters, thus being more efficient on training [21].

In this work, a baseline is drawn from six different traffic features (we also call them flow dimensions). A decision boundary could be defined to classify if the analyzed flow is anomalous or not. However, to avoid defining a hard threshold and risk missing anomalies that are less disturbing and barely noticed, a fuzzy inference system was used to decide about the presence of an anomaly. This way, each feature has a membership function and a sum of their result compared to a boundary calculated based on the performance in each dataset. This model was tested using two datasets: the first with emulated traffic generated by the data communication and networking research group called Orion, from the Computer Science Department at State University of Londrina. The second is CICDDoS2019, a well-known dataset by the anomaly detection community that includes different types of DDoS attacks (MSSQL, SSDP, CharGen, NTP, TFTP, SYN flood, UDP flood and UDP-Lag) [22].

The main contributions of this paper are listed as follows:

- Present a combination of regression GRU networks and a Fuzzy Logic classifier for anomaly detection;
- Present a detailed description of the proposed system so it is reproducible and possible to validate;
- Compare the method with other deep learning techniques presenting better results;
- Present a mitigation algorithm to block attacks before the network controller gets overloaded and applications stop.

The rest of this work is organized as follows: section II presents related works; section III presents each module of the developed system; section IV presents the test scenarios and their results; and section V presents conclusions.

II. RELATED WORK

Network anomaly detection and intrusion detection are two important and heavily studied fields, and numerous combinations of techniques can be used to detect and mitigate attacks from a network [23]–[27]. On top of that, the software-defined networking (SDN) paradigm brings an evolution to network management and has been used as a flexible solution to manage heterogeneous networks [3]. Thus, SDN intrusion detection is an important field of study to be improved.

Novaes *et al.* [28] presented a detection and mitigation system applied in a SDN environment using deep learning and fuzzy logic techniques. In their work, long short-term memory (LSTM) networks were trained to characterize traffic by predicting 6 flow dimensions. A fuzzy membership function was used to combine the network outputs with the SDN real values and decide if the network was being attacked. Then, a mitigation policy was used to identify and drop the malicious flows. This study is an example of how effective deep learning and fuzzy logic can be in traffic characterization and how SDN can simplify the attack mitigation process. Our study uses a similar methodology, but differs in the type of neural network used, the mitigation algorithm, and hyperparameter tuning.

X. Tao *et al.* [19] applied an anomaly detection algorithm in a distributed parallel computing platform called Spark [29] to train independent models as a way to increase the data rate and prepare for 6G wireless communications. This algorithm used a gated recurrent unit network to create a long term traffic behavior and a subsample aggregation method to reduce the dimensionality of the traffic before the detection.

Jing Yu *et al.* [30] described an intrusion detection system that uses a convolutional neural network to detect intrusion behavior. The model was compared to recurrent neural networks and presents a faster convergence, with better accuracy.

Kwadwo Boahen *et al.* [31] proposed an anomaly detection method using a random forest classifier. A particle swarm optimization algorithm and a gravitational search algorithm were used to optimize the random forest classification by selecting relevant features from the datasets. In their study, well known datasets were used to present the model's efficiency: NSL-KDD and UNSW-NB15. Despite not using deep learning techniques, this work highlights how feature selection is an important topic in network anomaly detection due to the high number of features collected from traffic. In our work we use six features extracted from traffic that proved to be successful in previous works.

Brajabidhu Singh *et al* [32] presented a framework based on deep transfer learning using gated recurrent unit networks. The model called WideDeep consists of a regression component (wide) and a classification component (deep). The method was also tested on well known datasets: KDDCup99 and UNSW-NB15. In this work, a feature selection process is described using principal component analysis. Long short-term memory network is also mentioned to have a similar performance as GRU, something we confirmed with our study.

Assis *et al.* [33] also used deep learning to detect and mitigate network attacks with an anomaly based approach. Convolutional neural networks were utilized to analyze IP flow dimensions an recognize a pattern on network normal traffic, pointing out anomalies as they occur. The mitigation module receives IP data and recognizes the malicious IP when an anomaly is detected. The system operates in one second intervals to detect and mitigate anomalies.

Novaes *et al.* [34] presented a generative adversarial network framework to detect and mitigate anomalies. It addresses the vulnerabilities that deep neural networks have against adversarial attacks. This way, a generator and a discriminator are trained against themselves, resulting in a discriminator that classifies traffic as normal or anomalous. As for mitigation, an algorithmic module takes actions using the detection as a basis. To validate the proposed system, two datasets were used, one of them being generated by the group, and the other being CICDDoS2019 dataset.

Andresini *et al.* [35] proposed an autoencoder-based intrusion detection system using a novel deep metric learning methodology (DML) named "RENOIR". In their work, autoencoders and triplet networks are combined to analyse flow based samples from a network. The system was tested with three datasets: KDDCUP99, AAGM177 and CICIDS17. The work did not present a mitigation module, but concluded that the applied methodology is comparable to state-of-the-art DML algorithms.

Bhuvaneswari *et al.* [36] presented an framework that detects anomalies in internet of things traffic using fog computing to decentralize the security system. Fog nodes were used to distribute the traffic load thus reducing the latency and increasing the accuracy. Convolutional deep neural networks were used to classify the traffic between normal and attack.

Ren-Hung *et al.* [37] described a system that aims to detect anomalies as fast as possible by analysing only the first few packets from each flow. To do so, they preset a framework called D-PACK that uses a convolutional neural network and an autoencoder to analyse packets. The presented results show that the detection can be made with as few as two packets from each flow.

Priyadarsini *et al.* work [3] is a survey about software defined networks. In the study, an analysis was presented about the state-of-the-art of traffic management and its details, including efficient routing, control implementation, architecture, security, etc. The study also highlighted some unexplored problems that could be discussed on future works.

At last, articles [1] and [38] are reviews about detection and mitigation of DDoS attacks in SDN. The former systematically reviewed security mechanisms and characterized them in four categories: Information theory-based, Machine learning-based, Artificial Neural Networks based and miscellaneous methods. It also presents challenges on SDN security and its research. The latter provided taxonomy based on detection strategies such as statistical and machine learning while also presented emerging approaches (e.g. honeynet, network slicing etc.).

It is noticeable that network anomaly detection is widely studied field and, despite that, there is still room for evolution. It is not common for other works to present regression approaches and mitigation solutions. This study aims to combine gated recurrent unit neural networks with fuzzy logic to detect and mitigate portscan and denial of service

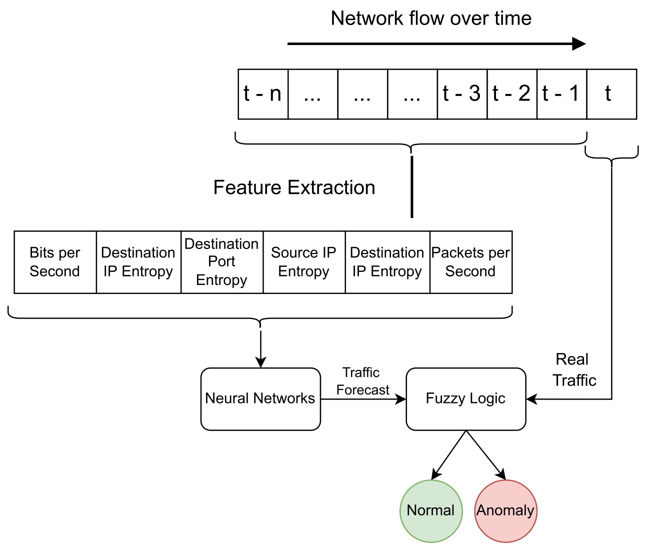


FIGURE 1. **System overview.**

attacks. We also aim to show how hyperparameters affect the algorithm's performance and why they were chosen.

III. THE PROPOSED SYSTEM

As represented in Fig. 1, the proposed system utilizes software-defined networks' capabilities of quickly gathering flow information to feed a deep learning model that detects and mitigates anomalies every second. To do so, six neural networks are trained to predict one of the following flow dimensions: bits per second, packets per second, source port entropy, source IP entropy, destination port entropy, and destination IP entropy. The reason for choosing to use six models during development is to improve performance throughout the tests, since the difference in order of magnitude between the features could harm the learning process. Each prediction is then compared to the actual traffic in a fuzzy membership function. Finally, a defuzzification threshold is defined based on attack data to classify the flow as normal or anomalous.

Shannon's entropy, represented in (1) [39], is used to transform IP and port information from a qualitative measurement to a quantitative one. This is necessary since the neural network requires numeric values to work with. In this formula, i represents each individual event, p_i the probability for it to happen, and N is the number of possible events. The resulting entropy (H) tend to be lower when the probability of a specific event is significantly higher than the others. In the proposed system, for example, a denial of service attack would notably lower the destination entropy and probably be detected for this reason.

$$H = -\sum_{i=1}^{N} p_i \log_2(p_i) \tag{1}$$

The rest of this chapter is organized as follows: section III-A explains how the gated recurrent unit (GRU) networks were utilized, section III-B describes fuzzy logic and how it was applied, and section III-C details how the system mitigates attacks.

A. GATED RECURRENT UNIT NEURAL NETWORKS

Recurrent neural networks such as LSTM and GRU are commonly used in problems in which sequence and correlation between data entries are relevant, namely in time-series forecasting or sequence generation [40]–[42]. This happens due to gates in the structure of those networks that utilize previous data entries to influence next predictions while update themselves. The influence comes in the form of decisions on rather keep an information or discard it [6], [43].

GRU neurons have two sets of weights: W, which is combined always with the neuron input (the input is represented by x), and V, which will always get combined with the neuron's hidden state. Both are adjusted during training phase, as well as the bias set (represented by letter b) which has the same role as in traditional neural networks.

GRU neurons also have two gates: "reset gate" and "update gate". The former is responsible to adjust how much information from the last output will be discarded and is represented by r_t in (2), while the latter will balance the amount of information sent to the next state and is represented by z_t in (4). The reset gate value is applied to compute $\hbar_t$ in (3) while the update gate is used to determine the current hidden state h_t in (5) [44], [45].

$$r_t = \sigma_g(W_r x_t + V_r h_{t-1} + b_r) \tag{2}$$
$$\hbar_t = tanh(W_h x_t + V_h(r_t \cdot h_{t-1}) + b_h) \tag{3}$$
$$z_t = \sigma(W_z x_t + V_z h_{t-1} + b_z) \tag{4}$$
$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \hbar_t \tag{5}$$

The functions sigmoid and tanh are defined respectively by the following equations:

$$sigmoid(x) = \frac{1}{1 - e^{-x}} \tag{6}$$
$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{7}$$

In the proposed system, as aforementioned, six GRU networks are trained to each predict one of the six flow dimensions since, during development, tests with a single network predicting all dimensions were avoided. As input, the network receives all mentioned flow dimensions of the last X seconds. The data is normalized using the standard scaler from the preprocessing module of the Python library scikit-learn [46]. The normalization is commonly used with neural networks since it helps improving the model's performance while also fixing the input and output range for activation functions [47]–[49]. As for the training, keras Python library [50] was used with Adam optimizer with default parameters. During training, only normal data is utilized as input, therefore, it is expected that anomaly data will result in significantly larger error. Hyperparameters were tuned to maximize F1-score of anomaly detection of the test dataset in the first scenario.

Neural networks require numerical input that describe the behavior of the network. For this reason, values such as IP address and protocol need at least a way to map to a numerical value. Even if the values are mapped, their numerical order does not represent a useful information, thus hindering the network's learning process. Therefore, to well represent the network's traffic to the neural network, the six dimensions used in this paper were chosen. Total of bits and packets are a quantitative dimension that measure the amount of traffic collected. Alternatively, entropies of source and destination of IP and port are quantitative measurements of the traffic extracted from qualitative ones and are most important for its characterization, since they describe how well distributed the traffic is throughout the network.

When dimensions such as bits/s and packets increase in value, for example in a flash crowd event, it is expected that entropies stay close to the same since each user uses on average the same resources. When a DDoS attack occurs, some hosts will stand out on requests thus disturbing entropy measures by lowering destination IP and port ones while increasing source port, since it is a distributed attack. A portscan attack will very likely increase destination port entropy since several ports are being targeted by packets, and possibly lower source IP entropy if a single host is the attacker. Based in this variance we are able to distinguish portscan from DDoS attacks.

B. FUZZY LOGIC

First introduced in 1965 in [51], Fuzzy Logic is a useful tool that has been used in problems that need to work with uncertainty [52]. It offers a gradient of possibilities as an alternative to the true and false from binary logic, making it closer to human logic [53]. To do so, it utilizes linguistic values, as opposed to crisp ones, defined by membership functions and allow partial truth for each of those values [54].

Membership functions define the degree of association of a variable to a set it represents. The degree varies in an interval between 0 and 1 included, where 0 means completely unrelated and 1 means completely related. Different functions can represent each set that will be used by inference rules to reach a conclusion [52]. This versatility allows for a variety of usages to help computers solve real world problems [53], [55], [56].

In the proposed system, fuzzy logic is used to decide whether the collected traffic can be considered normal or anomalous. Since there are six flow dimensions being analyzed, without fuzzy logic it would be necessary to choose a hard threshold to each dimension individually, creating a necessity of another decision threshold of how many anomalous dimensions to raise an alarm. Such an approach is susceptible to two main problems: if a single dimension presents an intense anomaly but others do not, a false negative would occur; additionally, if an attack affects lightly every dimension but does not surpass any of the thresholds, the attack would not be perceived. Thus, fuzzy logic is applied to determine the degree of anomalousness for each dimension and raises an alarm whenever the sum of this degree surpasses certain threshold, regardless of how many dimensions are currently affected. This solves both mentioned problems while only requiring a single threshold.

This was accomplished using the Gaussian membership function represented in (8), where x represents the real traffic, y the predicted traffic and σ the standard deviation from the last n seconds (n will be defined in the next section).

$$f(x) = 1 - e^{\frac{-(x-y)^2}{2(4.47\sigma)^2}} \tag{8}$$

The constant 4.47 was chosen based on Brett G. Amidan's usage of Chebyshev's inequality [57] represented in (9). In this equation, X_d represents the data, μ the data mean, σ the standard deviation of data, and, finally, k represents the number of standard deviations from the mean. Thus, given a k, the inequality presents the data percentage that lies between k standard deviations from the mean, which can also be interpreted as the probability of a given X_d that belongs to the data to be inside that interval. It is important to note that this inequality should be used when the data distribution is unknown [57].

$$P(|X_d - \mu| \leq k\sigma) \geq \left(1 - \frac{1}{k^2}\right) \tag{9}$$

Given a probability P of X to be an outlier, k can be determined using (10). In this work, the degree of certainty used was 95%, resulting in a $P = 0.05$ and, consequently, $k \simeq 4.47$.

$$k = \frac{1}{\sqrt{P}} \tag{10}$$

Algorithm 1 summarizes how the detection phase works. Respectively, X_t and X_{t-1} represent all network's traffic dimensions from the analyzed second and previous ones. y represents each network's prediction and is sent to the fuzzy membership function to be compared to the actual observed values (X_t). An anomaly is detected if the fuzzy membership function returns a value greater than the calculated threshold. To distinguish portscan attacks from DDoS, we check which entropies are the most affected. Portscan attacks will make the destination port entropy increase, while DDoS attacks decrease the destination port and increase source port entropy. The dimensions affected are determined by how great the prediction error is in that second. If source port entropy is the most affected and greater than expected, a DDoS attack is reported. Otherwise, if destination port entropy is greater than expected, it is considered a portscan attack.

C. ATTACK MITIGATION IN SDN

Software-defined networks have as an advantage the simplicity that the centralization of the control plane can provide while configuring its behavior [38]. From the control plane, it is possible to set up forwarding policies to network switches with instructions to block flows from specific IP addresses. Thus, it should be possible to design an autonomous system that not only detects but mitigates attacks immediately when

Algorithm 1 Anomaly Detection Algorithm

```
Require: $X_{t-1}$ ← (x1, x2, x3, x4, x5, x6); fuzzy_treshold
Ensure: Is normal or anomalous
1:  y1 ← Gru_Bits(x1, x2, x3, x4, x5, x6)
2:  y2 ← Gru_Packets(x1, x2, x3, x4, x5, x6)
3:  y3 ← Gru_IP_Src_entropy(x1, x2, x3, x4, x5, x6)
4:  y4 ← Gru_Port_Src_entropy(x1, x2, x3, x4, x5, x6)
5:  y5 ← Gru_IP_Dst_entropy(x1, x2, x3, x4, x5, x6)
6:  y6 ← Gru_Port_Dst_entropy(x1, x2, x3, x4, x5, x6)
7:  Y ← (y1, y2, y3, y4, y5, y6)
8:  fuzzy_number ← fuzzy_membership_function(Y, $X_t$)
9:  if fuzzy_number > fuzzy_threshold then
10:     return Anomalous
11: else
12:     return Normal
```

Algorithm 2 Mitigation Algorithm

```
1:  if Portscan attack then
2:      attacked IP ← IP address that receives most flows
3:      Malicious IP ← From the flows destined to attacked IP, get the one with most variety of ports
4:      if IP not in Safe List then
5:          Drop Packets from malicious IP
6:  else if DDoS attack then
7:      attacked IP ← IP that receives most packets
8:      attacked port ← from attacked IP, get the port that receives most flows
9:      suspect IP list ← all IP's that have destination to attacked IP and attacked port
10:     for IP in suspect IP list do
11:         if IP not in Safe List then
12:             Drop Packets from IP
```

the detection module raises the alarm, without the necessity to wait for human interference [1]. To do so, in addition to the detection module's warnings, the mitigation module needs the type of event detected and the flows which generated it. Then, a specific algorithm should be followed to mitigate that event by identifying and blocking the malicious flows from that origin.

When an SDN switch receives new flows, it performs a lookup in its forwarding table to determine where it should be sent. If there are no matches, the network controller is consulted for instructions. The forwarding table will be updated based on what the mitigation module decides. The process of consultation adds latency to the network and should be avoided, thus some instructions should be sent preemptively to switches if possible.

The mitigation algorithm is represented in Algorithm 2. It utilizes a safe list to avoid dropping legitimate traffic due to false positives from the detection module. To do so, while there are no attacks, the mitigation module keeps a list of source IP addresses that generated legitimate traffic in the last five minutes. When an attack is detected, flows on the safe list are not dropped since they are less likely malicious. The system was tested mainly with distributed denial of service and portscan attacks in two scenarios detailed in a further section. When an IP is considered malicious and is not in the safe list, all its flows are blocked for 20 seconds.

IV. PERFORMANCE EVALUATION AND RESULTS ANALYSIS

A. SCENARIOS

Two datasets from different sources were used to evaluate the detection and mitigation performance to showcase the development method. The datasets present different characteristics in the number of hosts, amount, and type of attacks, representing diverse networks.

All tests were made in a Windows 10 machine with a Ryzen 7 1700x, 32GB of RAM and Python version 3.8.7. During tests, we aimed to maximize the F1-score metric. Represented in (13), this metric represents the harmonic mean of precision (represented in (12)) and recall (represented in (11)) and is appropriate for measuring performance in unbalanced datasets such as the ones used in this study.

Network anomaly datasets tend to be inherently unbalanced. Network attacks are expected to happen sporadically, while the prevalent behavior tend to be normal traffic. Since datasets are designed to represent reality, it is common to see such imbalance. Additionally, datasets may have normal class as the most numerous one, but also being constituted by many attack classes that, when combined, make timestamps majoritarily malicious, thus creating imbalance for both sides at the same time. In this case, some attacks might get under or over represented, depending on how the dataset was generated, its creator's objective and available resources. In the first scenario, 90% of timestamps are benign traffic, while only 10% of the second scenario. For this reason, accuracy and precision alone will not represent well the model's performance and F1-score was chosen as main metric.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{11}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{12}$$

$$F1 - score = 2\frac{Precision \times Recall}{Precision + Recall} \tag{13}$$

For each scenario we present an overview, how it was generated, it's attacks, and how the data is organized. We used these scenarios to validate the proposed model on how well it detects anomalies and mitigates them. To do so, we need enough normal traffic so the neural networks can learn the network's behavior and labeled attack data to calibrate the fuzzy logic membership function. The following datasets attend the requirements.

1) FIRST SCENARIO

The first dataset was created by the computer networks study group from State University of Londrina and is available online [58]. To generate the data, an SDN emulator called

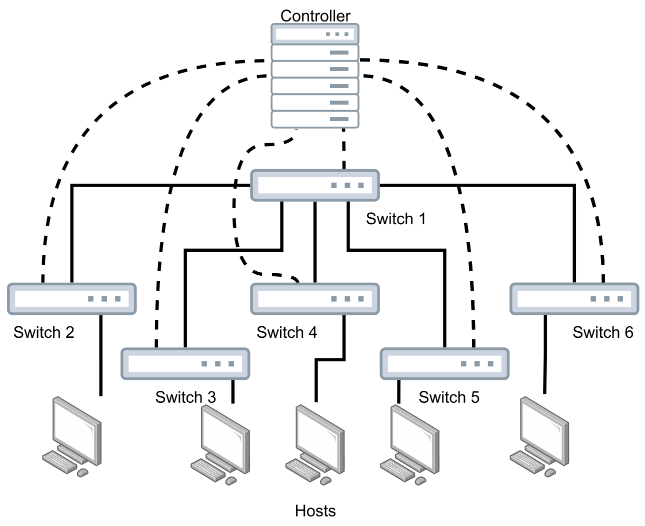


FIGURE 2. Scenario 1 emulated network topology.

mininet [59] was used to create a six switches structure in a tree topology, where the first switch is the root of the tree, and the other five are connected to it, as illustrated in Fig.2. Each of the five "leaf" switches are connected to twelve hosts. The network traffic was emulated using the *Scapy* [60] Python library, which creates and sends packets through the network. To simulate DDoS attacks, the software *hping3* [61] was used, while *Scapy* generated portscan attacks.

The dataset is divided into four days of 24 hours each, and the data is organized in flows. Thus, it was necessary to pre-process it to extract and normalize the six flow dimensions mentioned in section III. One of the days has no attack, and the other three have each a DDoS and a portscan attack interval on different timestamps. Fig.3 shows the six dimensions over the day without attacks.

2) SECOND SCENARIO

The second dataset is CIC-DDoS2019, made available by the University of New Brunswick [22]. This dataset was designed to address the most common DDoS attacks and resemble real-world data. For this reason, background traffic was generated based on a profiling system to simulate normal human behavior from 25 users. The dataset is divided in 2 days of data, each containing several DDoS attacks, and is organized as flows. The attacks are labeled based on the type of attack with the following: MSSQL, SSDP, CharGen, NTP, TFTP, SYN flood, UDP flood and UDP-Lag. This dataset has been widely used by the anomaly detection community [62]–[64].

Pre-processing is also necessary to extract and normalize the six flow dimensions mentioned in section III. The first day without its attacks was used to train the network, and the results were drawn using the second day. Fig.4 displays the six dimensions calculated from the dataset's training day. Not all of the attacks available were used in this work. The attacks used were: MSSQL, NetBios, SSDP, and UDP. In this dataset, all four have similar number of flows, so none of them should be prominent over the others during the tests.

B. THE NEURAL NETWORK HYPERPARAMETERS

This section presents the tests and results from the hyperparameter tuning using the first dataset. After the tuning, the model is trained individually for each dataset to make it possible to see the model's performance with a different, unrelated network. The hyperparameters were tuned to find the best neural network configuration. A brief explanation of each:

1) Number of input seconds: it represents the amount of previous traffic data used as input to the network. If the number is X, the network will receive the X previous seconds of data to predict the next one. More seconds mean more context to the model.
2) Number of neurons in dense and GRU layers: More neurons means more ways of processing information and can result in a better performance on extracting relevant knowledge from data. Too many neurons will result in overfitting since the model will not have enough data to train them, losing the ability to generalize the solution to the task.
3) Number of dense layers in the network: more layers can result in the model developing the ability to extract more complex information from the data presented. Excess layers will result in a slower training time and more data required to achieve the same result of fewer layers.
4) Training batch size: the number of training samples used between each weights update. Too few samples will result in a poor gradient estimation since it only uses a subset of the data to update the weights. Too many will require more epochs to train the model since the weights will be updated fewer times each epoch.
5) Fuzzy window: the network output is compared with the actual traffic using a fuzzy membership function to evaluate if an anomaly is present. This function also receives the standard deviation from the last n seconds, where n is the fuzzy window. Thus, the membership tolerance to errors from the network will change depending on how the traffic behaves. If the value is too high, the fuzzy function will consider a traffic profile that does not represent the current instant. A low value can result in wrong classifications due to brief traffic variations.

The first hyperparameter tuned was the number of seconds of flow used as an entry to the network. As its possible to see in Fig.5, models were trained with 5, 10, 15, 30, 40, 50 and 60 seconds, with 5 seconds settling the best performance in F1-score metric (the closest to 1 the better). This demonstrates that adding to the network more context in the form of previous seconds does not improve its effectiveness in detecting anomalies. Tests with lower amount of seconds did not resulted in significant improvement thus being discarded.

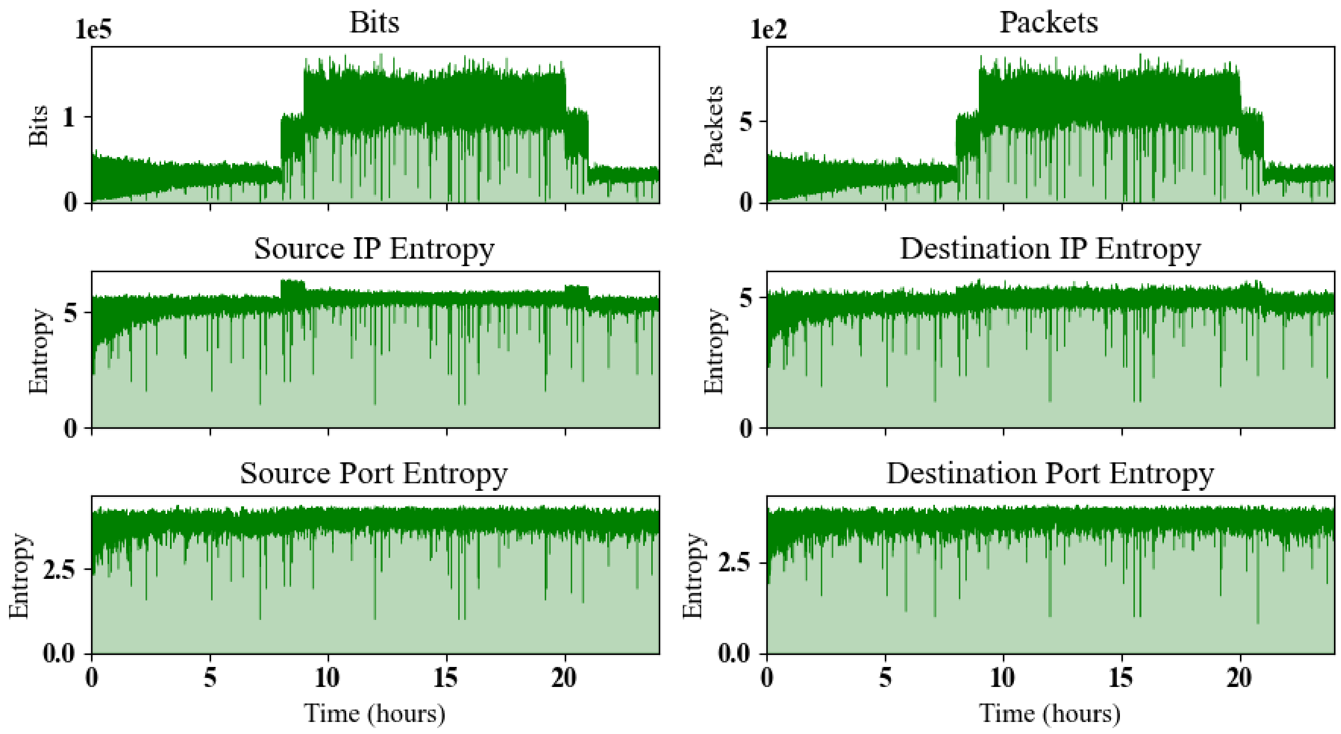


FIGURE 3. Scenario 1 data without attacks.

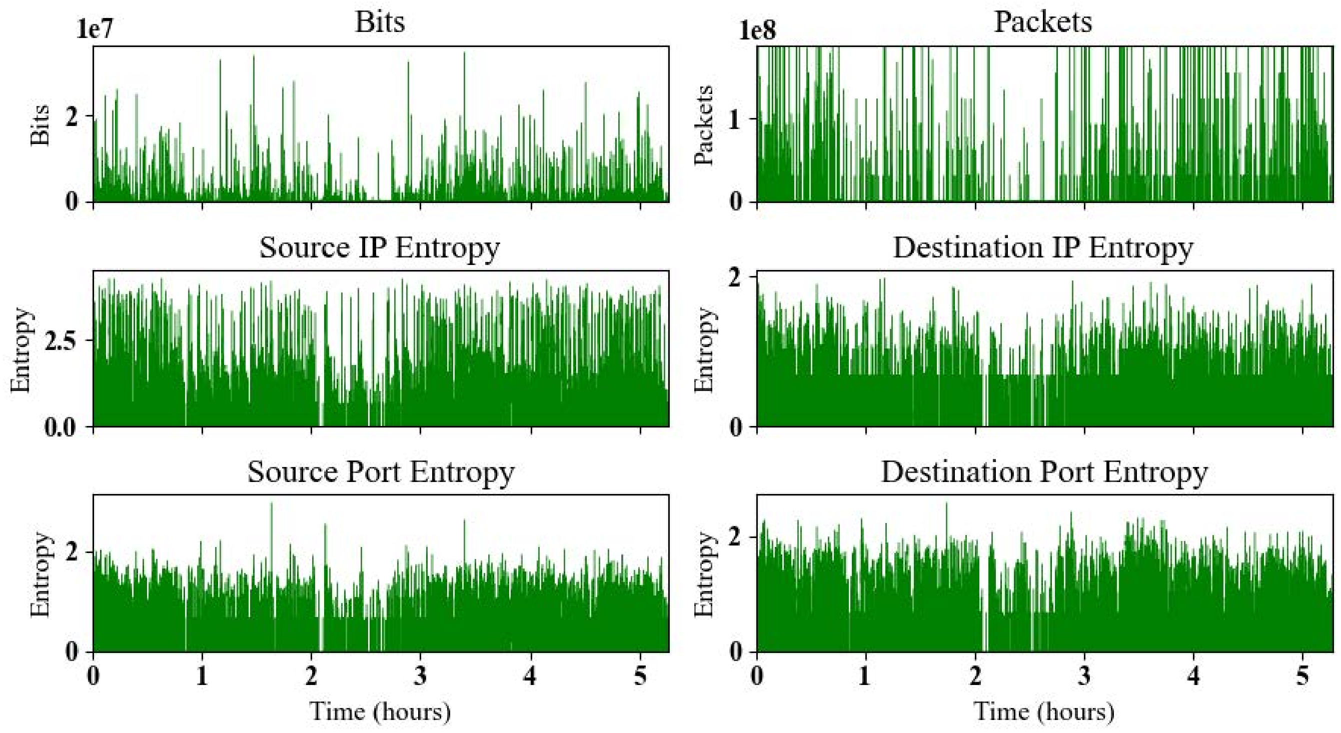


FIGURE 4. Scenario 2 training day.

Next, tests varying the number of neurons on the GRU layer were made using 16, 32, and 64 neurons, and Fig.6 shows that the difference between the three is negligible. After that, the number of dense layers in the network was tested. By training networks with one, two, and three layers with one and two layers leading the best performance as shown in Fig.7.

To decide which layer disposition to choose, tests were made with different amounts of GRU and dense neurons. Fig.8 exhibits the best combinations between all the tests

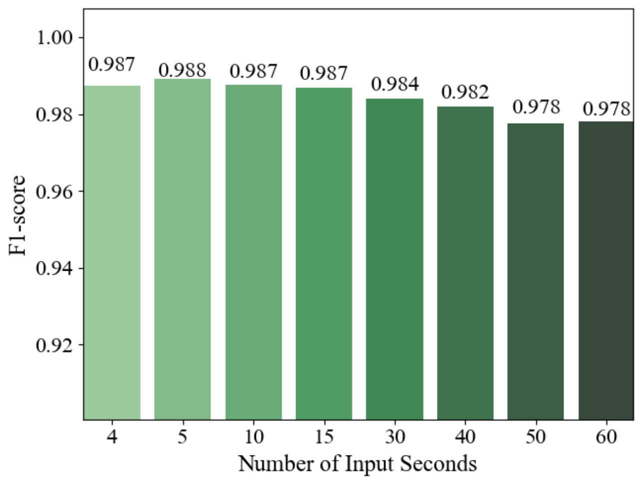


FIGURE 5. F1-score by number of input seconds.

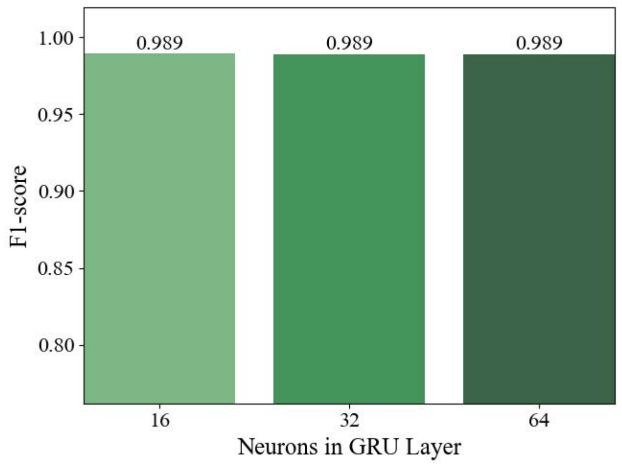


FIGURE 6. F1-score over number of neurons in GRU.

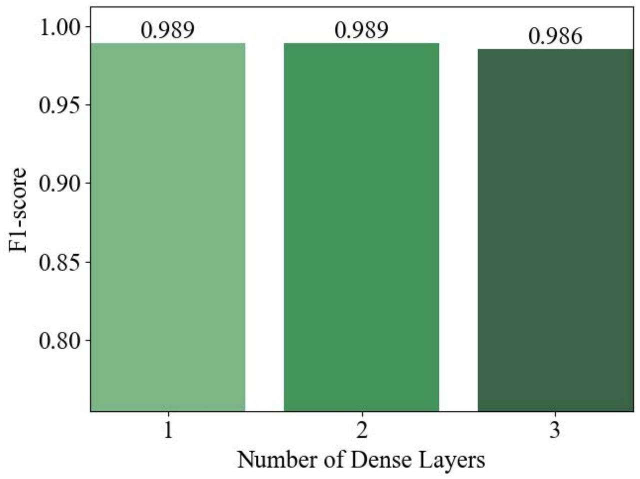


FIGURE 7. F1-score over number of dense layers.

made. The arrangements always use a single GRU layer followed by one or two fully connected layers. For example, the best combination with a result of 0.9888: [32, 16, 1] represents 32 GRU neurons, a dense layer with 16 neurons followed by a single fully-connected neuron.

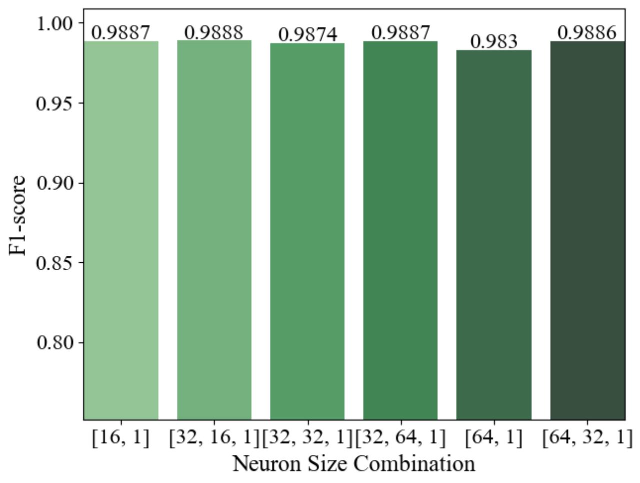


FIGURE 8. F1-score over the number of neurons on each layer.

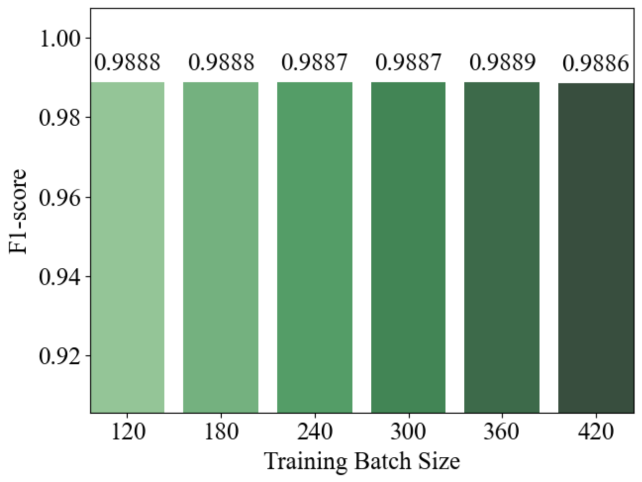


FIGURE 9. F1-score over training batch size.

Subsequently, training batch size was evaluated from 120 to 420 inclusive with increments of 60. Results in Fig.9 show that 360 was the most effective batch by a slight difference.

Next, different fuzzy window values were tested. As explained before, the usage of fuzzy logic in this work consists of a membership function that decides if the neural network's traffic prediction could represent a real value. To do so, an amount of previous traffic seconds is used to calculate a standard deviation. We call this amount of seconds the fuzzy window. Fig.10 shows that at least 20 seconds should be necessary to detect anomalies better.

C. SCENARIOS RESULTS

1) FIRST SCENARIO

After tuning the model, anomaly detection results were measured in the last day of the dataset. Only deep learning techniques were tested because they already outperformed most shallow learning techniques and are dominant in publications [28], [36]. The results can be represented in a confusion matrix as in Table 1 and Fig.11. Fig.12 shows values for

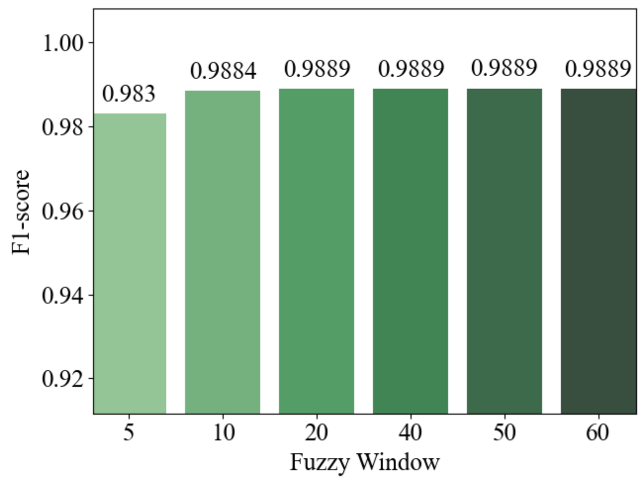


FIGURE 10. F1-score over fuzzy window.

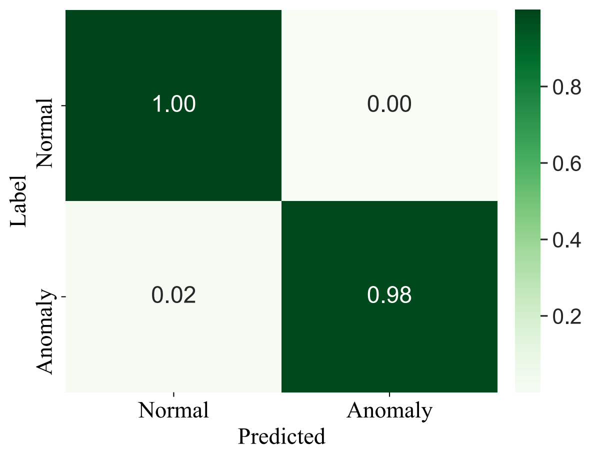


FIGURE 11. Scenario 1 normalized confusion matrix.

TABLE 1. Scenario 1 detection confusion matrix.

	No Attack Detected	Attack Detected
Label: no attack	76,236	20
Label: with attack	191	9,949

precision, recall and F1-score (the closer to 1.0 the better). The tests were also made with other deep learning techniques using the same architecture as GRU and the results are displayed in Table 2. The results show that CNN and LSTM were slightly under GRU's performance but DNN did significantly worse.

The precision value decreases its score when false positives occur. In other words, when the neural network labels normal data as anomalous, its precision decreases. We can conclude that the higher the precision, the better the model could learn and characterize the network traffic. The GRU model had a better precision score than the others due to its proficiency with time series. As expected, LSTM had results similar to GRU since both are recurrent neural networks.

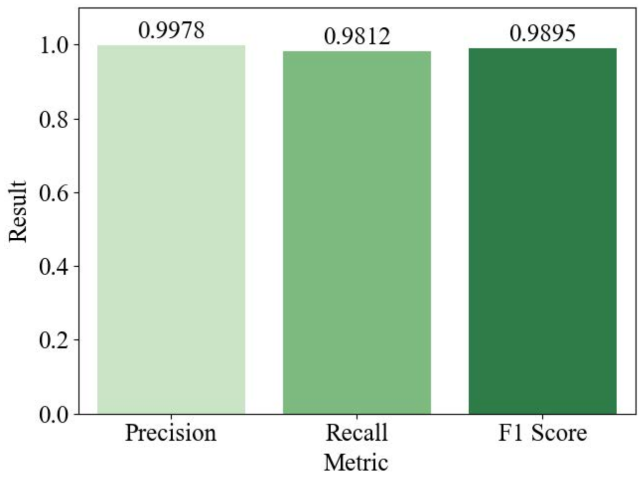


FIGURE 12. Scenario 1 detection metrics.

TABLE 2. Scenario 1 result comparison.

	Precision	Recall	F1-score
DNN	0.9507	0.8616	0.9040
CNN	0.9745	**0.9940**	0.9843
LSTM	0.9829	0.9829	0.9829
GRU	**0.9978**	0.9812	**0.9895**

TABLE 3. Scenario 1 mitigation results.

	Total Mitigated	Total Accepted
Malicious Flows	2,959,021 (99.99%)	18 (0.01%)
Normal Flows	264 (0.01%)	3,957,094 (99.99%)

An analysis of all false negatives of the models show that, for both GRU and LSTM, 99% of them were from portscan attacks. CNN had 53% of all its false negatives from portscan attacks. Since convolutional neurons have no memory or recurrent structure, CNN has to rely on spacial relations between the values to represent the temporal factor from the traffic. This is probably the reason that it has different results from the recurrent models.

For mitigation tests, the algorithm was applied in the dataset using the detection module's output. There were 3, 957, 358 normal traffic flows, 87, 284 portscan flows, and 2, 871, 755 DDoS flows. The results are shown in Fig.13 and Table 3, both display the number of flows of each type before and after mitigation. It is worth pointing out that even if the detection module presented false negatives, the mitigation module could drop all DDoS flows from the test. This happens thanks to the blocklist that keeps dropping flows from previously detected malicious IPs for a while. However, there is a cost since if a legitimate user is flagged as malicious, it will briefly be unable to access the server.

It is noteworthy how portscan attacks are harder to detect than DDoS. This happens because our system uses the neural network's prediction error to indicate attacks. During a portscan attack, destination port entropy increases since there

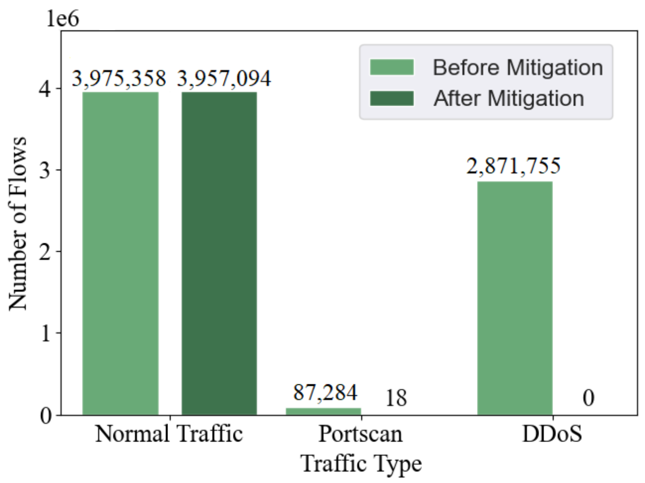


FIGURE 13. Scenario 1: Number of flows before and after mitigation.

TABLE 4. Scenario 2 detection results.

	No Attack Detected	Attack Detected
Label: no attack	181	116
Label: with attack	50	1,376

are more ports being targeted and more uncertainty is added to the system, source and destination IP entropy decreases since a single IP sends more packages than normal average. How impactful the attack is to our features is relative to how many packets are malicious compared to the average normal traffic. Portscan is a reconnaissance attack, its purpose is to gather information rather than overload a system. Therefore, with less flows and packets, a smaller and harder to perceive variation will happen. DDoS attacks by nature have a great amount of flows and packets as shown in Fig.13 thus being easily detected.

This highlights the importance of fuzzy logic. Since it combines the errors of all features, even when a single feature presents an anomaly, the system is still able to recognize an attack. Without a fuzzy membership function, the system would rely on multiple hard thresholds to detect anomalies. Those multiple thresholds would either be susceptible to false negatives when not enough features point anomalies, false positives when a single anomaly presents an outlier, or even both.

2) SECOND SCENARIO

For the second scenario, the model's anomaly detection results are represented in Table 4 and Fig.14. Values for precision, recall and F1-score are displayed in Fig.15. Results with other deep learning methods are also displayed in Table 5 and GRU, CNN and LSTM are still close to each other with DNN behind by a bigger margin. Again, GRU's precision is higher than the other networks, but it is less sensible for attacks than CNN and LSTM. The f1-score is still higher for GRU as it has a better balance between precision and recall.

Based on the output of the detection module, mitigation tests were made. The same algorithm was applied and the results are displayed in Fig.16 and Table 6. It is possible to see that most malicious flows were dropped but there is an increase in false positives, even with the safe list avoiding mistaken drops. Different types networks might require adjusts in how many seconds of safe list should be implemented since depending on average user behavior. Probably some normal traffic could be saved with different

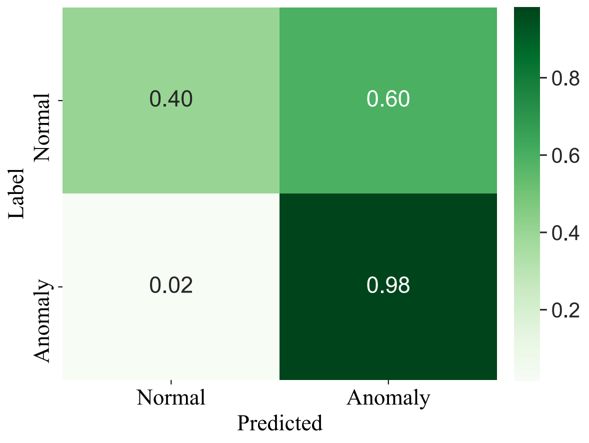


FIGURE 14. Scenario 2 normalized confusion matrix.

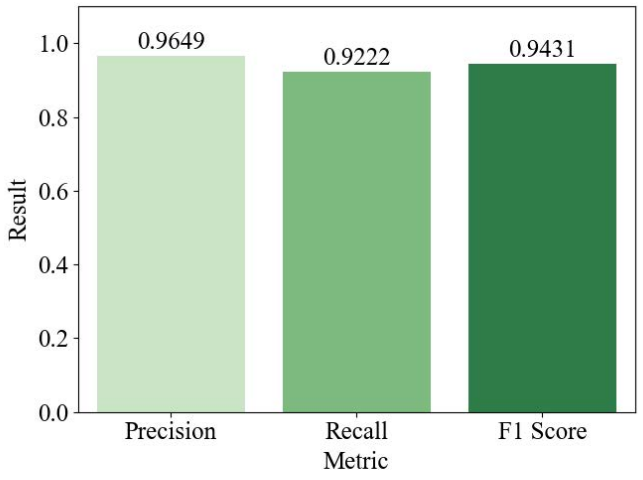


FIGURE 15. Scenario 2 detection metrics.

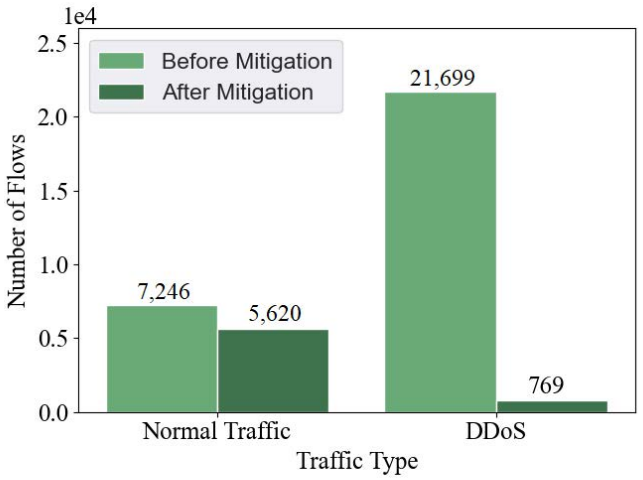


FIGURE 16. Scenario 2: Number of flows before and after mitigation.

TABLE 5. Scenario 2 result comparison.

	Precision	Recall	F1-score
DNN	0.8534	0.9739	0.9097
CNN	0.8818	**0.9894**	0.9324
LSTM	0.8859	0.9859	0.9332
GRU	**0.9222**	0.9649	**0.9431**

TABLE 6. Scenario 2 mitigation results.

	Total Mitigated	Total Accepted
Malicious Flows	20,930 (96.45%)	769 (3.54%)
Normal Flows	1626 (22.43%)	5,620 (77.56%)

configurations. It could be subject of a future work how a safe list with variable seconds would work based on real time user behavior.

V. CONCLUSION

In this paper, we presented an anomaly-based network anomaly detection system using six GRU neural networks to predict network traffic and apply fuzzy logic to compare it with real traffic. The neural network is trained with regular traffic only. For this reason, it is expected that anomalous traffic results in significantly wrong predictions that will be detected by the fuzzy membership function. A fuzzy threshold is defined using training data with attacks. This model was validated using two different scenarios. During tests, the detection module achieved an F1-score of 98.9% in the first scenario and 93.2% in the second.

The mitigation module relies on the detection module to drop malicious flows. It contains a safe list and a block list that gets updated automatically every second with the information received. With this algorithm, we were able to mitigate DDoS and portscan attacks.

In the first scenario results, it is noticeable how DDoS attacks are more straightforward to detect due to their more disturbing nature. Portscan attacks can be harder to identify due to their low impact on network flow dimensions, thus sometimes not reaching the defined fuzzy threshold. Despite this, the proposed system still mitigated most of the attack flows.

The second scenario original dataset has a variety of denial of service attacks, but not all of them were used in this work. The attacks used were: MSSQL, NetBios, SSDP, and UDP. In this dataset, all four have similar number of flows, so none of them should be over or underrepresented in relation to the others. This scenario results show that the model could block most of the malicious flows but had a higher false-positive rate than the first scenario. The mitigation module's safe list was projected to help in this type of scenario by preventing normal flows from being dropped. We believe that it is necessary more hours of normal data to improve the neural network's learning.

In this work, we compared the performance of GRU only with deep learning methods. A deep neural network (DNN), convolutional neural network (CNN), and long short-term memory network (LSTM) were evaluated with the same methodology. CNN and LSTM had similar results to GRU, whereas DNN was noticeably worse.

For future work, we plan on improving the model's detection by testing with other fuzzy logic membership functions that may be more effective. Trapezoidal or beta-shaped functions might help ignore minor deviations while highlighting more significant ones. Although it still requires testing, this might help minimize false positives. Another possible experiment is to give less importance to bits/s and packets/s since they have a less predictable pattern as shown in figures 3 and 4.

More datasets are also being researched and will be used in future work. Hyperparameters tuning can be made with the other scenarios to evaluate the performance. Finally, we also plan on experimenting different deep learning methods such as temporal-convolutional networks to characterize traffic.

REFERENCES

[1] J. Singh and S. Behal, "Detection and mitigation of DDoS attacks in SDN: A comprehensive review, research challenges and future directions," *Comput. Sci. Rev.*, vol. 37, Aug. 2020, Art. no. 100279.

[2] B. Alhijawi, S. Almajali, H. Elgala, H. B. Salameh, and M. Ayyash, "A survey on DoS/DDoS mitigation techniques in SDNs: Classification, comparison, solutions, testing tools and datasets," *Comput. Electr. Eng.*, vol. 99, Apr. 2022, Art. no. 107706.

[3] M. Priyadarsini and P. Bera, "Software defined networking architecture, traffic management, security, and placement: A survey," *Comput. Netw.*, vol. 192, Jun. 2021, Art. no. 108047.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Mar. 2008.

[5] A. Mohamed, M. Hamdan, S. Khan, A. Abdelaziz, S. F. Babiker, M. Imran, and M. N. Marsono, "Software-defined networks for resource allocation in cloud computing: A survey," *Comput. Netw.*, vol. 195, Aug. 2021, Art. no. 108151.

[6] M. V. O. Assis, L. F. Carvalho, J. Lloret, and M. L. Proença, "A GRU deep learning system against attacks in software defined networks," *J. Netw. Comput. Appl.*, vol. 177, Mar. 2021, Art. no. 102942.

[7] R. Deb and S. Roy, "A comprehensive survey of vulnerability and information security in SDN," *Comput. Netw.*, vol. 206, Apr. 2022, Art. no. 108802.

[8] R. Sanjeetha, P. Benoor, and A. Kanavalli, "Mitigation of DDoS attacks in software defined networks at application level," in *Proc. PhD Colloq. Ethically Driven Innov. Technol. Soc. (PhD EDITS)*, Aug. 2019, pp. 1–3.

[9] M. Dimolianis, A. Pavlidis, and V. Maglaris, "Signature-based traffic classification and mitigation for DDoS attacks using programmable network data planes," *IEEE Access*, vol. 9, pp. 113061–113076, 2021.

[10] M. Di Mauro, G. Galatro, G. Fortino, and A. Liotta, "Supervised feature selection techniques in network intrusion detection: A critical review," *Eng. Appl. Artif. Intell.*, vol. 101, May 2021, Art. no. 104216.

[11] Z. S. Malek, B. Trivedi, and A. Shah, "User behavior pattern-signature based intrusion detection," in *Proc. 4th World Conf. Smart Trends Syst., Secur. Sustainability (WorldS)*, Jul. 2020, pp. 549–552.

[12] A. Shah, S. Clachar, M. Minimair, and D. Cook, "Building multiclass classification baselines for anomaly-based network intrusion detection systems," in *Proc. IEEE 7th Int. Conf. Data Sci. Adv. Analytics (DSAA)*, Oct. 2020, pp. 759–760.

[13] M. L. Proença, C. Coppelmans, M. Bottoli, A. Alberti, and L. S. Mendes, "The hurst parameter for digital signature of network segment," in *Telecommunications and Networking*, J. N. de Souza, P. Dini, and P. Lorenz, Eds. Berlin, Germany: Springer, 2004, pp. 772–781.

[14] M. F. Umer, M. Sher, and Y. Bi, "Flow-based intrusion detection: Techniques and challenges," *Comput. Secur.*, vol. 70, pp. 238–254, Sep. 2017.

[15] M. Ozdag, "Adversarial attacks and defenses against deep neural networks: A survey," *Proc. Comput. Sci.*, vol. 140, pp. 152–161, Jan. 2018.

[16] E. C. Leek, A. Leonardis, and D. Heinke, "Deep neural networks and image classification in biological vision," *Vis. Res.*, vol. 197, Aug. 2022, Art. no. 108058.

[17] P. Khaire and P. Kumar, "A semi-supervised deep learning based video anomaly detection framework using RGB-D for surveillance of real-world critical environments," *Forensic Sci. Int., Digit. Invest.*, vol. 40, Mar. 2022, Art. no. 301346.

[18] K. H. Poon, P. K.-Y. Wong, and J. C. P. Cheng, "Long-time gap crowd prediction using time series deep learning models with two-dimensional single attribute inputs," *Adv. Eng. Informat.*, vol. 51, Jan. 2022, Art. no. 101482.

[19] X. Tao, Y. Peng, F. Zhao, C. Yang, B. Qiang, Y. Wang, and Z. Xiong, "Gated recurrent unit-based parallel network traffic anomaly detection using subagging ensembles," *Ad Hoc Netw.*, vol. 116, May 2021, Art. no. 102465.

[20] C. Hu, T. Ou, H. Chang, Y. Zhu, and L. Zhu, "Deep GRU neural network prediction and feedforward compensation for precision multiaxis motion control systems," *IEEE/ASME Trans. Mechatronics*, vol. 25, no. 3, pp. 1377–1388, Jun. 2020.

[21] J. Zhang, X. Mu, J. Fang, and Y. Yang, "Time series imputation via integration of revealed information based on the residual shortcut connection," *IEEE Access*, vol. 7, pp. 102397–102405, 2019.

[22] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. Int. Carnahan Conf. Secur. Technol. (ICCST)*, Oct. 2019, pp. 1–8.

[23] W. Wang, S. Jian, Y. Tan, Q. Wu, and C. Huang, "Representation learning-based network intrusion detection system by capturing explicit and implicit feature interactions," *Comput. Secur.*, vol. 112, Jan. 2022, Art. no. 102537.

[24] X. Zhang, F. Yang, Y. Hu, Z. Tian, W. Liu, Y. Li, and W. She, "RANet: Network intrusion detection with group-gating convolutional neural network," *J. Netw. Comput. Appl.*, vol. 198, Feb. 2022, Art. no. 103266.

[25] H. Li, C. Zhao, Y. Liu, and X. Zhang, "Anomaly detection by discovering bipartite structure on complex networks," *Comput. Netw.*, vol. 190, May 2021, Art. no. 107899.

[26] W. Xu, J. Jang-Jaccard, A. Singh, Y. Wei, and F. Sabrina, "Improving performance of autoencoder-based network anomaly detection on NSL-KDD dataset," *IEEE Access*, vol. 9, pp. 140136–140146, 2021.

[27] C. Wang, H. Zhou, Z. Hao, S. Hu, J. Li, X. Zhang, B. Jiang, and X. Chen, "Network traffic analysis over clustering-based collective anomaly detection," *Comput. Netw.*, vol. 205, Mar. 2022, Art. no. 108760.

[28] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proenca, "Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment," *IEEE Access*, vol. 8, pp. 83765–83781, 2020.

[29] *Apache Spark—Unified Engine for Large-Scale Data Analytics*. Accessed: May 4, 2022. [Online]. Available: https://spark.apache.org/

[30] J. Yu, X. Ye, and H. Li, "A high precision intrusion detection system for network security communication based on multi-scale convolutional neural network," *Future Gener. Comput. Syst.*, vol. 129, pp. 399–406, Apr. 2022.

[31] E. K. Boahen, B. E. Bouya-Moko, and C. Wang, "Network anomaly detection in a controlled environment based on an enhanced PSOGSARFC," *Comput. Secur.*, vol. 104, May 2021, Art. no. 102225.

[32] N. B. Singh, M. M. Singh, A. Sarkar, and A. K. Mandal, "A novel wide & deep transfer learning stacked GRU framework for network intrusion detection," *J. Inf. Secur. Appl.*, vol. 61, Sep. 2021, Art. no. 102899.

[33] M. V. O. de Assis, L. F. Carvalho, J. J. P. C. Rodrigues, J. Lloret, and M. L. Proença, "Near real-time security system applied to SDN environments in IoT networks using convolutional neural network," *Comput. Electr. Eng.*, vol. 86, Sep. 2020, Art. no. 106738.

[34] M. P. Novaes, L. F. Carvalho, J. Lloret, and M. L. Proença, "Adversarial deep learning approach detection and defense against DDoS attacks in SDN environments," *Future Gener. Comput. Syst.*, vol. 125, pp. 156–167, Dec. 2021.

[35] G. Andresini, A. Appice, and D. Malerba, "Autoencoder-based deep metric learning for network intrusion detection," *Inf. Sci.*, vol. 569, pp. 706–727, Aug. 2021.

[36] N. G. B. Amma and S. Selvakumar, "Anomaly detection framework for Internet of Things traffic using vector convolutional deep learning approach in fog environment," *Future Gener. Comput. Syst.*, vol. 113, pp. 255–265, Dec. 2020.

[37] R.-H. Hwang, M.-C. Peng, C.-W. Huang, P.-C. Lin, and V.-L. Nguyen, "An unsupervised deep learning model for early network traffic anomaly detection," *IEEE Access*, vol. 8, pp. 30387–30399, 2020.

[38] I. A. Valdovinos, J. A. Pérez-Díaz, K.-K.-R. Choo, and J. F. Botero, "Emerging DDoS attack detection and mitigation strategies in software-defined networks: Taxonomy, challenges and future directions," *J. Netw. Comput. Appl.*, vol. 187, Aug. 2021, Art. no. 103093.

[39] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[40] R. Al-Shabandar, A. Jaddoa, P. Liatsis, and A. J. Hussain, "A deep gated recurrent neural network for petroleum production forecasting," *Mach. Learn. Appl.*, vol. 3, Mar. 2021, Art. no. 100013.

[41] X. Li, X. Ma, F. Xiao, C. Xiao, F. Wang, and S. Zhang, "Time-series production forecasting method based on the integration of bidirectional gated recurrent unit (Bi-GRU) network and sparrow search algorithm (SSA)," *J. Petroleum Sci. Eng.*, vol. 208, Jan. 2022, Art. no. 109309.

[42] T. Shi, S. Huang, L. Chen, Y. Heng, Z. Kuang, L. Xu, and H. Mei, "A molecular generative model of ADAM10 inhibitors by using GRU-based deep neural network and transfer learning," *Chemometric Intell. Lab. Syst.*, vol. 205, Oct. 2020, Art. no. 104122.

[43] S. M. Kasongo and Y. Sun, "A deep gated recurrent unit based model for wireless intrusion detection system," *ICT Exp.*, vol. 7, no. 1, pp. 81–87, Mar. 2021.

[44] Y. Qin, D. Chen, S. Xiang, and C. Zhu, "Gated dual attention unit neural networks for remaining useful life prediction of rolling bearings," *IEEE Trans. Ind. Informat.*, vol. 17, no. 9, pp. 6438–6447, Sep. 2021.

[45] S. Mirzaei, J.-L. Kang, and K.-Y. Chu, "A comparative study on long short-term memory and gated recurrent unit neural networks in fault diagnosis for chemical processes using visualization," *J. Taiwan Inst. Chem. Eng.*, vol. 130, Jan. 2022, Art. no. 104028.

[46] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Dec. 2011.

[47] S. Bhanja and A. Das, "Impact of data normalization on deep neural network for time series forecasting," 2018, *arXiv:1812.05519*.

[48] B. Lu, D. Xu, and B. Huang, "Deep-learning-based anomaly detection for lace defect inspection employing videos in production line," *Adv. Eng. Informat.*, vol. 51, Jan. 2022, Art. no. 101471.

[49] T. Sun, X. Wang, J. Wang, X. Yang, T. Meng, Y. Shuai, and Y. Chen, "Magnetic anomaly detection of adjacent parallel pipelines using deep learning neural networks," *Comput. Geosci.*, vol. 159, Feb. 2022, Art. no. 104987.

[50] F. Chollet *et al.* (2015). *Keras*. Accessed: May 11, 2022. [Online]. Available: https://keras.io

[51] L. A. Zadeh, "Fuzzy sets," *Inf. Control*, vol. 8, no. 3, pp. 338–353, Jun. 1965.

[52] P. Hilletofth, M. Sequeira, and A. Adlemo, "Three novel fuzzy logic concepts applied to reshoring decision-making," *Expert Syst. Appl.*, vol. 126, pp. 133–143, Jul. 2019.

[53] G. T. Coşkun and A. Y. Yalçıner, "Determining the best price with linear performance pricing and checking with fuzzy logic," *Comput. Ind. Eng.*, vol. 154, Apr. 2021, Art. no. 107150.

[54] E. van Krieken, E. Acar, and F. van Harmelen, "Analyzing differentiable fuzzy logic operators," *Artif. Intell.*, vol. 302, Jan. 2022, Art. no. 103602.

[55] V. E. Mirzakhanov, "Value of fuzzy logic for data mining and machine learning: A case study," *Expert Syst. Appl.*, vol. 162, Dec. 2020, Art. no. 113781.

[56] J. Serrano-Guerrero, F. P. Romero, and J. A. Olivas, "Fuzzy logic applied to opinion mining: A review," *Knowl.-Based Syst.*, vol. 222, Jun. 2021, Art. no. 107018.

[57] B. G. Amidan, T. A. Ferryman, and S. K. Cooley, "Data outlier detection using the Chebyshev theorem," in *Proc. IEEE Aerosp. Conf.*, Mar. 2005, pp. 3814–3819.

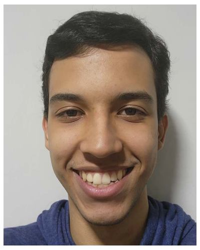
[58] *Datasets Used in Publications—Orion Research Group*. Accessed: May 4, 2022. [Online]. Available: http://www.uel.br/grupos/orion/datasets.html

[59] *MiniNet: An Instant Virtual Network on Your Laptop (or Other PC)*. Accessed: May 4, 2022. [Online]. Available: http://mininet.org/

[60] *Scapy—Packet Crafting for Python2 and Python3*. Accessed: May 4, 2022. [Online]. Available: https://scapy.net

[61] *Hping3—Command-Line Oriented TCP/IP Packet Assembler/Analyzer*. Accessed: May 4, 2022. [Online]. Available: http://hping.org

[62] M. A. Lawal, R. A. Shaikh, and S. R. Hassan, "A DDoS attack mitigation framework for IoT networks using fog computing," *Proc. Comput. Sci.*, vol. 182, pp. 13–20, Jan. 2021.
[63] R. O. Ogundokun, J. B. Awotunde, P. Sadiku, E. A. Adeniyi, M. Abiodun, and O. I. Dauda, "An enhanced intrusion detection system using particle swarm optimization feature extraction technique," *Proc. Comput. Sci.*, vol. 193, pp. 504–512, Jan. 2021.
[64] G. C. Amaizu, C. I. Nwakanma, S. Bhardwaj, J. M. Lee, and D. S. Kim, "Composite and efficient DDoS attack detection framework for B5G networks," *Comput. Netw.*, vol. 188, Apr. 2021, Art. no. 107871.


DANIEL M. BRANDÃO LENT received the bachelor's degree from the State University of Londrina (UEL), in 2020. He is currently pursuing the master's degree in computer science studying computer networks security and artificial intelligence. He is also a member of the Orion Research Group that studies computer networks and data communication at the Computer Science Department, UEL.

MATHEUS P. NOVAES received the master's degree in computer science from the State University of Londrina (UEL), Brazil, where he is currently pursuing the Ph.D. degree with the Electrical Engineering Department. He has been a member of the Research Group Computer Networks and Data Communication, Computer Science Department, UEL. His research interests include management and security of computer networks.

LUIZ F. CARVALHO received the master's degree in computer science from the State University of Londrina, in 2014, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas, in 2018. He has experience in computer science with emphasis in computer networks and is part of the Research Group Computer Networks and Data Communication. His research interests include management and security of computer networks and software-defined.

JAIME LLORET (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in physics, in 1997, the B.Sc. and M.Sc. degrees in electronic engineering, in 2003, and the Ph.D. (Dr.Ing.) degree in telecommunication engineering, in 2006. He has been the Chair of the Integrated Management Coastal Research Institute (IGIC), since January 2017. He is currently a Full Professor with the Polytechnic University of Valencia. He is also the Head of the "Active and collaborative techniques and use of technologic resources in the education (EITACURTE)" Innovation Group. He has been the Internet Technical Committee Chair (IEEE Communications Society and Internet Society), from 2013 to 2015. He has authored 14 books and has more than 650 research papers published in national and international conferences, international journals (more than 375 with with Clarivate Analytics JCR). He has been the co-editor of 54 conference proceedings and a guest editor of several international books and journals. He is the Editor-In-Chief of the *Ad-Hoc and Sensor Wireless Networks* (with Clarivate Analytics JCR) and the *International Journal of Networks Protocols and Algorithms*. He has led many local, regional, national, and European projects. He was the Chair of the Working Group of the Standard IEEE 1907.1, from 2013 to 2018. Since 2016, he has been the Spanish Researcher with highest H-index in the *Telecommunications* journal list according to Clarivate Analytics Ranking. Moreover, he is included in the world's top 2% scientists according to the Stanford University List, since 2020. He has been the General Chair (or Co-Chair) of 75 international workshops and conferences. He is a Senior Member of ACM and an IARIA Fellow.

JOEL J. P. C. RODRIGUES (Fellow, IEEE) is currently with the College of Computer Science and Technology, China University of Petroleum, Qingdao, China; the Senac Faculty of Ceará, Brazil, the Head of Research, Development, and Innovation; and a Senior Researcher at the Instituto de Telecomunicações, Portugal. He is also a Highly Cited Researcher, the Leader of the Next Generation Networks and Applications (NetGNA) Research Group (CNPq), an IEEE Distinguished Lecturer, a Member Representative of the IEEE Communications Society on the IEEE Biometrics Council, and the President of the Scientific Council at ParkUrbis—Covilhã Science and Technology Park. He was the Director of Conference Development—IEEE ComSoc Board of Governors, the Technical Activities Committee Chair of the IEEE ComSoc Latin America Region Board, the Past-Chair of the IEEE ComSoc Technical Committee (TC) on e-Health and the TC on Communications Software, a Steering Committee Member of the IEEE Life Sciences Technical Community, and the Publications Co-Chair. He is the Editor-In-Chief of the *International Journal of E-Health and Medical Communications* and the editorial board member of several high-reputed journals (mainly, from IEEE). He has been the General Chair and the TPC Chair of many international conferences, including IEEE ICC, IEEE GLOBECOM, IEEE HEALTHCOM, and IEEE LatinCom. He has authored or coauthored about 1000 papers in refereed international journals and conferences, three books, two patents, and one ITU-T recommendation. He had been awarded several outstanding leadership and outstanding service awards by IEEE Communications Society and several best papers awards. He is a member of the Internet Society, a Senior Member of ACM, and a Fellow of AAIA.

MARIO LEMES PROENÇA, JR. received the M.Sc. degree in computer science from the Informatics Institute of Federal University of Rio Grande do Sul (UFRGS), in 1998, and the Ph.D. degree in electrical engineering and telecommunications from the State University of Campinas (UNICAMP), in 2005. He is currently an Associate Professor and the Leader of the Research Group that studies computer networks with the Computer Science Department, State University of Londrina (UEL), Brazil. He has authored or coauthored over 110 papers in refereed international journals and conferences, books chapters, and one software register patent. He has supervised more than 140 B.Sc., M.Sc., and Ph.D. students. He has been a master's Supervisor in computer science with UEL, where he is a Ph.D. Supervisor with the Department of Electrical Engineering. His research interests include computer networks, network operations, management and security, and IT governance.