Securing 5G Network Slices with Adaptive Machine Learning Models as-a-Service: A Novel Approach

Roumaissa Bekkouche*, Mawloud Omar*, Rami Langar*‡
* LIGM-CNRS UMR 8049, University Gustave Eiffel, F-77420 Marne-la-Vallée, France
‡ Software and IT Engineering Department, Ecole de Technologie Supérieure (ÉTS), Montréal, QC H3C 1K3, Canada
E-mails: roumaissa.bekkouche@univ-eiffel.fr ; mawloud.omar@univ-ubs.fr ; rami.langar@etsmtl.ca

***Abstract*—5G networks are highly dynamic and non-homogeneous networks, making resource management more complex and vulnerable to different network attacks like DDoS, Port Scanning, etc. In addition, Network Slicing plays an important role in these networks in enabling a multitude of 5G applications, services, and use cases. In this context, we propose in this paper, a new approach to secure 5G network slices by developing a models' orchestrator providing an adaptive Machine-Learning (ML) models as-a-Service. Specifically, the proposed models' orchestrator is a cloud server that acts as a decision-making entity to offer on-demand adaptive ML models to detect potential attacks by tuning and adapting ML parameters and algorithms according to the characteristics of the requester devices/nodes and the real-time conditions of each network slice. We demonstrate the effectiveness of our approach through a series of experiments, by training different ML algorithms with different network slices properties. Results show that our approach provides a more efficient and effective way of securing 5G networks compared to traditional methods in terms of respecting the requirements raised by each slice/node of these networks.**

***Index Terms*—Networks Security, 5G Networks, Network Slicing, Machine learning, Security as-a-service, Orchestration, Cloud server.**

I. Introduction

The transition to the fifth generation of telecommunications networks (5G) is imminent. It is a source of enthusiasm for several participants, including telecommunications operators, designers and manufacturers of telecommunications equipment, service providers, application providers, small and medium-sized enterprises, standardization organisms, and government organisms. This transition is the subject of several various research works so well in the academic community and industry fields.

5G networks are a set of significant conceptual and technological changes: Network Function Virtualisation (NFV), Software-defined Networks (SDN), and Network Slicing, that completely transform the network architecture. These conceptual, technological, and architectural transformations introduce new fundamental security requirements and show the need to consider securing 5G networks differently. Their virtualization and decentralization make the network's traditional physical and perimeter security measures less effective.

Currently, existing solutions to detect or to mitigate different attacks that may affect these networks, such as Distributed Denial of Service attacks (DDoS), are generally categorized into signature-based, anomaly-based, or hybrid defense mechanisms [1]. Recently, machine learning (ML)-based solutions have been largely advocated in network security. However, such solutions present typically general attack detection models used in all situations and by all types of devices that can be connected to the network without taking into account the characteristics of these devices that may affect the quality of the attack detection.

In addition, it can be noted that the existing proposed solutions do not take into account the characteristics and real-time conditions of the 5G network's different slices, namely, Enhanced Mobile Broadband slice (eMBB), Ultra-reliable, low-latency communications slice (URLLC) and Massive Machine-Type Communications slice (mMTC), which can also affect the process of detecting and mitigating attacks. Taking all these parameters into account can improve the quality of attack detection in such networks and create more adaptive and on-demand security models that depend on the characteristics of the requester devices/nodes and the real-time conditions of network slices.

To this end, we propose in this paper a new approach to secure 5G network slices by developing a models' orchestrator providing adaptive ML models as-a-Service. Our contributions can be summarized as follows:

- We propose a network architecture including our security models' orchestrator cloud server.
- We train multiple ML algorithms (e.g., Decision Tree, Random Forest, XGBoost, Gradient Boosting, KNN, Naive Bayes, Logistic Regression, Linear Discriminant Analysis, SVM, and CNN) with the most optimal hyper-parameters to be used by our ML-based security orchestrator.
- We develop, deploy and validate our solution using a Python-based discrete event simulator, where we model the service-request demands as a Poissonian process and implement the ML algorithms using several Python libraries (e.g., TensorFlow, Keras, Scikit-learn).

The rest of the paper is organized as follows. Section II discusses the related work. In Section III, we describe our proposed solution and ML algorithms used in our training phase. In Section IV, we present the experimental setup. In Section V, we present and discuss the achieved results. Finally, Section VI concludes this work with some perspectives and future research ideas.

II. Literature review

Several works have been put forth in the literature to maximize security measures in 5G networks and beyond. In

the following, we will focus only on ML-based approaches to detect cyberattacks in such networks.

Authors in [2] propose a secure deep learning framework for SDN-based 5G wireless network named SeDeN. SeDeN trains the system with a deep-learning algorithm to detect malicious traffic and mitigate its propagation. The proposed framework runs on the SDN controller and comprises three major components: monitoring, attack detection, and attack mitigation. They have trained the system with the Long Short-Term Memory and Recurrent Neural Network (LSTM-RNN) machine learning algorithm using the AWID dataset (a wireless intrusion detection dataset). To evaluate the SeDeN framework, they have used mininet-wifi emulator and three performance metrics: attack detection time, accuracy, and packet loss rate. Evaluation scenarios show that the controller could detect and mitigate attacks by setting flow rules for dropping packets from malicious nodes.

Another framework is proposed by [3] named DeepSecure. Their framework is also based on LSTM deep learning technique that detects user equipment (UE) network traffic as DDoS attacks or regular traffic and assigns an appropriate slice to a legitimate UE request. The DeepSecure framework consists of UE devices, an attack detection model, a slice prediction model, an infrastructure provider, and network slices. They evaluated the performance of the proposed framework using the CICDDoS 2019 dataset and compared their work with other works to prove the effectiveness of their solution based on several ML evaluation metrics.

For the same purpose, authors in [4, 5] propose a Deep Learning (DL)-based approach to detect DDoS attacks in 5G and beyond mobile networks, and then create a sinkhole-type slice with a small portion of physical resources to isolate and mitigate the attackers' action. They implement and evaluate their approach on a real 5G testbed based on *OpenAirInterface* and show the effectiveness of their approach in terms of detection accuracy, false positive rate, execution time, among other ML-related metrics.

In the same context, authors in [6] propose a DDoS self-protection framework that detects and mitigates autonomously the application-layer DDoS attacks. The proposed solution is based on Deep Learning techniques and SDN technology. It consists of three main modules: the "Network Flow Collector", the "Features Extractor" and the "Detector". The Network Flow Collector permanently collects network flows via port mirroring. The collected traffic is periodically exported to Features Extractor to retrieve the flow's features relevant to application-layer DDoS attack detection. Once extracted, the flow features are passed to the Detector for uncovering suspicious behavior. The framework performance is assessed in terms of the web server's response time, system load, and the effectiveness of solution even in the presence of adversarially-crafted malicious flows.

Authors in [7] developed a Neural Network-based (Secure5G) Network Slicing model to proactively detect and eliminate threats based on incoming connections before they infest the 5G core network. Their work is an extension of

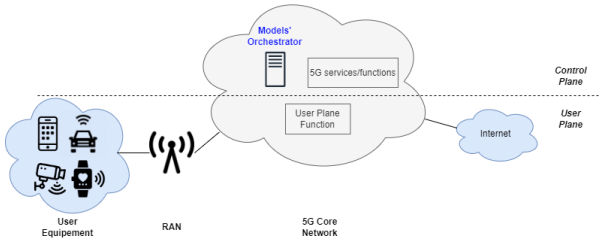

Fig. 1. Global Architecture

DeepSlice research work [8]. Their work aims to mitigate the DDoS initiation attacks by UEs. They (i) identify the incoming connection request and assign the most optimal slice based on the device type, then they (ii) verify the connection request if it is legit or a potential threat, and (iii) assign the connection to either an appropriate network slice (valid request) or transfer it to the quarantine slice (malicious request).

As a synthesis of the literature review and to the best of the authors' knowledge, the aforementioned existing works do not consider the QoS requirements of network slices and the different characteristics of network devices, when applying ML/DL algorithms to detect attacks in 5G networks. Indeed, most studies propose a general model of attack detection to be used regardless of the real-time network conditions and performance of each device requester. To overcome this limitation, we present in this paper our solution to secure 5G networks by proposing a models' orchestrator node that takes into account the requirements raised by each network slice and device. Note that this work is an extension of our article [9], which proposes a personalized model for the IoT slice only.

III. Proposed Approach

In this section, we present our approach starting with the proposed global architecture including the models' orchestrator. Then, we present the ML/DL models used in our analysis.

A. Global architecture

The proposed system architecture, depicted in Fig. 1, is based on the 5G Network architecture referenced in the 3GPP standards. It consists of User Equipment (UE) that regroups all the end users' devices, connected via the 5G Radio Access Network (RAN) to the 5G Core Network, and then to data networks, such as the Internet.

UEs include all types of devices that can be affected to different 5G network slices. All these different UE are continuously connected to the network and need real-time protection from threats and attacks that can affect the network and act negatively on other security principles such as privacy and availability. Taking into account the heterogeneity of these devices and the different quality of services (QoS) required by each network slice and device, we need to provide adaptive ML models to secure the set of user data exchanged on the network. This can be done by adding an ML models' orchestrator, which is represented in Fig. 1 by a server node hosted in the 5G Core Network.

B. Proposed Models' Orchestrator

The models' orchestrator is a cloud server that represents the decision-making entity of the proposed approach architecture. It gathers all the trained machine learning models and manages and orchestrates them.

To do this, the models' orchestrator uses the characteristics of each slice of the 5G network. On receipt of a request for a security device, the server checks the compatibility between the requester based on its characteristics and the machine learning model(s), and it carries out the decision process on this data to satisfy the request and meet the requirements raised by each slice and device.

The remainder of this section presents the slices' characteristics, the machine learning models used, and the models' orchestrator algorithm.

*1) **Slices characteristics**:* Slices' characteristics drive the models' orchestrator decision process. The orchestrator uses theme to decide which model suits each service request.

The characteristics mentioned in Table I can be found in the specification documents of 3GPP. 3GPP has categorized 5G network use cases into three main areas according to their requirements: eMBB, mMTC, and uRLLC. Standards define the needs and characteristics of these use cases, such as Response time, energy consumption, payload size, etc.

The automotive slice groups the modern connected vehicles that require an extremely multipurpose network that can simultaneously provide high-speed connection in the car, ultra-reliability, and low latency (URLLC) for autonomous driving, data collection, device-to-device communication, and more. In addition, these services must also be provided when vehicles are moving between different network infrastructures.

The massive IoT slice, also called the massive machine-type communications (mMTC) slice, supports an enormous number of devices, which are only periodically active and send small data payloads.

Enhanced Mobile Broadband (eMBB) is the mobile device slice offering broadband services. It is related to operations performed by common users in our daily lives. Providing mobile access to data for dense, highly mobile, and geographically distributed users.

*2) **Studied ML/DL models**:* In our study, the choice of ML/DL models is decided according to several factors, including the size of the dataset, the problem definition and its input/output, as well as the context of the study, which is attack detection in our case, defined as a multi-classification problem. Therefore, we chose the ML/DL algorithms most commonly used in this context and the most appropriate for the selected dataset, which will be described in the next section.

We select for each ML model the most advantageous hyperparameters in terms of prediction results. A description of each model used is given below.

- *Decision Tree (DT):* The Decision Tree algorithm is a supervised machine learning algorithm for classification and regression. It consists of building a tree model where each internal node represents an input variable, each branch represents a decision rule, and each leaf represents the result or output. After several tests, the optimal depth found for the model was 20.
- *Random Forest (RF):* Random Forests algorithm is an algorithm that combines multiple decision trees to create a forest. The trees are built independently, and the final decision or prediction results from a voting process. The optimal number of estimators used was 1000.
- *Gradient Boosting (GB):* Gradient Boosting: Gradient Boosting is a machine learning algorithm that combines several predictive models (for example, decision trees) to create a robust predictive model.
- *eXtreme Gradient Boosting (XGBoost):* This algorithm relies on decision trees and uses a combination of Gradient Boosting (GB) and advanced regulation techniques to improve the accuracy of predictions. The optimal number of estimators used in this model was 100.
- *K-Nearest Neighbors (KNN):* KNN algorithm is another supervised ML method used for classification and regression. In the KNN classification, the objective is to predict the class of a data point according to the majority class of its closest neighbors K. The optimal k-value found for the model was 1.
- *Linear Discriminant Analysis (LDA):* LDA is a supervised learning method used for dimensionality reduction and classification. LDA aims to project multidimensional input data over a subspace of smaller dimensions while preserving maximum variance between classes and minimizing intra-class variance. This makes it possible to separate classes from each other, thus facilitating the classification of new data points.
- *Support Vector Machines (SVM):* SVM algorithm is a supervised classification method used to separate data into classes by finding an optimal boundary (known as hyperplane) that maximizes the margin between them.
- *Multinomial Logistic Regression (LR):* mLR algorithm is a supervised classification method to predict a discrete response variable with three or more categories. It is an extension of binary logistic regression. The algorithm aims to find regression coefficients that minimize prediction error by adjusting the regression coefficients according to learning examples and maximizing the probability of correct prediction.
- *Naive Bayes (NB):* Naive Bayes algorithm is a supervised classification method based on the Bayes theorem with the naive hypothesis of independence between dataset features. The algorithm is trained by calculating the probability of each class according to the features, and then selecting the class with the highest probability for each new observation.
- *Convolutional Neural Networks (CNN):* CNN is a class of deep learning algorithms designed for data analysis with a grid structure that can also be applied to unstructured data. In this case, the data is transformed into a matrix representation that a CNN can process. This matrix is then processed by convolution and other layers to extract relevant characteristics. The configuration used for the

Slice/Characteristics	Response Time	Energy consumption	Payload size	Data rate	Mobility	Security level
eMBB (mobile devices)	4 ms	Low to High	Small to Big	Up to 20 Gbps	Highly mobile users	Low to High
mMTC (massive IoT)	10 ms	Low	Small	Up to 1 Mbps	depends on the application	Medium to High
URLLC (automotive slice)	0.5 ms	Medium to High	Small	Up to 10 Gbps	Very high	Medium to High

TABLE I
SLICE'S CHARACTERISTICS

CNN architecture was: a convolution layer, a flatten layer, a dense layer of 100 neurons, and the output layer with nine neurons.

*3) **Proposed Algorithm**:* The proposed models' orchestrator algorithm is based on two decision levels. The first level concerns the migration and transportation of the model to the requesting device, whereas the second level is related to the deployment and the use of the selected ML/DL model by the device to start the attacks' prediction.

- Model Migration Level: At this level, we are interested in the characteristics of the network link between the cloud server and the device; that is to say, we focus the process of choosing the ML/DL model mainly on the size of the model to respect the constraints related to bandwidth, latency and minimal response time of each network slice.
- Deployment Level: Once the model migration level is terminated, we check the characteristics of the device in terms of available resources. The choice of the model, in this case, will be based on the model size, the prediction time, and the detection accuracy, in order to respect the various constraints related to the device, such as storage capacity, energy efficiency, battery life, etc.

A pseudo algorithm of the molels' orchestrator is given as follows in Algorithm 1. The algorithm's input is represented by a vector containing the requester's information and requirements, including both the device's specifications and the slice's prerequisites, as shown in Table I.

The match function thoroughly checks the compatibility between the demand requirements and model parameters. Table II briefly summarizes the correspondence between the two parameters.

Slice	Model size	Model prediction time	Model reliability
eMBB	Medium to Big	Average	Medium to High
mMTC	Small	Rapid	High
URLLC	Medium	Rapid	High

TABLE II
SLICE'S CHARACTERISTICS ACCORDING TO ML MODELS

Algorithm 1 Models' Orchestrator

Input: serviceRequest (SliceRequirs, DeviceRequirs)
Output: MLmodels

$firstLevelRequirements \leftarrow$ *getSliceRequirement (serviceRequest)*
$secondLevelRequirements \leftarrow$ *getDeviceRequirement (serviceRequest)*
for each $model$ in $modelsList$ **do**
  $modelFeatures \leftarrow$ *getModelFeatures (model)*
  if match($firstLevelRequirements$, $modelFeatures$) AND match ($secondLevelRequirements$, $modelFeatures$) **then**
    $MLmodels.append(model)$
  end if
end for
return $MLmodels$

IV. IMPLEMENTATION

In this section, we present the implementation of the proposed approach, the dataset description and pre-processing used to train our ML/DL models.

A. Experimental setup

The experiments were conducted on a computer server under Gentoo 2.7 distribution, with 64Go of Memory, and processor Intel(R) Xeon(R) Gold 5215L CPU @ 2.50GHz. The used server to train and test the different machine-learning models. In addition, we used a Windows machine with processor Intel(R) Core(TM) i5-10310U CPU @ 1:70GHz 2:21GHz and 16Go of Memory for the simulation and orchestration part.

B. Dataset description and pre-processing

The dataset chosen for this study is a new dataset named 5G-NIDD [10]. It is built and created based on a functional 5G test network.

5G-NIDD is a fully labeled dataset with malicious and benign traffic captures using a real 5G test network and real devices. It consists of 1215890 network flows, each under a specific type of attack or benign traffic.

The composition of the dataset revolves around 52 features that include flow characteristics and label description. In Table III, we describe the labels used for the different flows.

Before using the dataset, pre-processing was performed to enhance the data quality and ML models' learning process.

Label	Description
Benign	Normal traffic.
SlowrateDoS	DDoS attack using slow rate attacks.
HTTPFlood	HTTP flood attacks that target the application layer.
ICMPFlood	ICMP flood attacks using ICMP echo requests.
UDPFlood	UDP flood attacks using UDP packets at a high rate.
SYNFlood	SYN flood attacks by transmitting SYN packets at a higher frequency.
SYNScan	SYN scan attacks used to discover open ports using TCP protocol.
TCPConnect-Scan	Similar to SYN scan, used when SYN scan is not an option.
UDPScan	UDP scan attacks used to discover open ports using UDP protocol.

TABLE III
LABELS DESCRIPTION

The pre-processing includes discarding additional and nominal attributes, discarding missing values, correlated features, and redundant features, and finally encoding some nominal features that we assumed were important for the learning process.

C. Training and Testing the ML/DL models

As indicated above (Section IV-B), the 5G-NIDD database was used to train and test machine learning models. The dataset was divided into two parts, 80% for the training and 20% for the test and validation. The algorithms were implemented using Scikit-learn, Xgboost, Keras, and TensorFlow libraries of Python.

V. Performance Results

In this section, we present and discuss the performance and the obtained results for both the ML/DL trained models and the models' orchestrator.

A. ML/DL models' performance results

ML/DL models are evaluated using a variety of performance metrics, including: i) Accuracy, ii) Precision, iii) Recall, and iv) F1 Score, in addition to v) Model size, vi) Training, and vii) Prediction Time. Table IV shows the performance details of these metrics. As a multi-class classification problem, the reported values for Precision, Recall, and F1 Score are the mean of the obtained results regarding all classes.

As you can see in Table IV, there is a difference between the models used; each model has different characteristics, whether for the accuracy score, the prediction time, or the size of the models, etc. The choice of model(s) to be sent when the models' orchestrator receives a request for a security service is made based on these results.

For example, for an IoT device, due to its minimal storage resources, the selected model should have a size that satisfies this constraint. On the other hand, if the requester specifies that the prediction time must be minimum, then the choice will be on the models with a shallow prediction time.

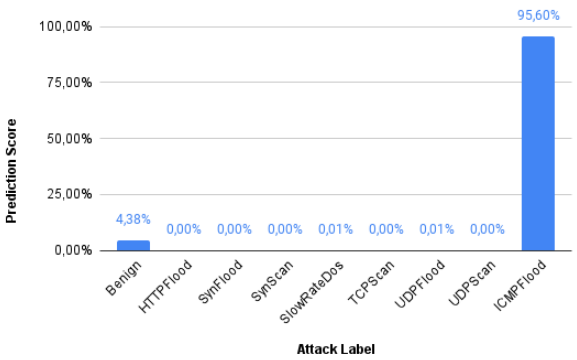

Fig. 2. Results of prediction for the ICMP Flood attack using DT model

B. Models' orchestrator performance results

To validate the proposed approach, we tested the functioning of the models' orchestrator using a Python-based discrete event simulator. Security Service requests are generated through a Poissonian process. The orchestrator receives the requests, processes them, and then sends the models that best fit the demand requirements to the requesting client.

Simulated Attacks' Scenarios: Different attacks scenarios have been implemented to validate the proposed solution. Attacks are generated using tools: $hping3$ for flooding attacks and $nmap$ for scanning port attacks.

In order to respect real-time network conditions, we keep capturing traffic in real-time by the victim device, The captured traffic is then passed through a feature extraction phase, and finally, the ML model sent by the orchestrator will be used to predict the attack.

Commands used to generate different attacks are:

- ICMP Flooding attack :
  sudo hping3 –rand-source –flood -1 -p "Range of ports" "IP address of the victim device"
- UDP Flooding attack :
  sudo hping3 –rand-source –flood –udp -p "Range of ports" "IP address of the victim device"
- SYN Scan attack :
  sudo nmap -sS "IP address of the victim device" -p "Range of ports"

Obtained Results: For each attack scenario, the models' orchestrator chose the best ML model to send to the client according to the requester's specifications and slice's requirements. Figures 2, 3, and 4 show the prediction score of the selected models by the models' orchestrator for the previous attacks scenarios.

For each attack scenario, the models' orchestrator have selected, respectively, DT model, both KNN and LDA models, LDA model for ICMP Flood attack, UDP Flood attack and Syn Scan attack.

Discussion: As can be seen in the results graphs, the selected models have a prediction rate of more than 80% on the actual traffic. An 80% of prediction rate indicates that the ML models can correctly classify up to 80% of the incoming traffic as either normal or as an attack type. This demonstrates the effectiveness of the trained models used by the models'

Metric	Decision Tree	Random Forest	XGBoost	Gradient Boost-ing	KNN	LDA	mLR	Naive Bayes	SVM	CNN
Accuracy	0.9614	0.9550	0.9621	0.9559	0.8108	0.8893	0.3935	0.3951	0.5361	0.7618
Precision	0.9877	0.9875	0.9884	0.9860	0.7244	0.8024	0.1548	0.2897	0.3534	0.9450
Recall	0.9873	0.9856	0.9883	0.9859	0.7158	0.9030	0.1113	0.2467	0.1560	0.9228
F1 Score	0.9874	0.9856	0.9883	0.9859	0.7164	0.8153	0.0630	0.1574	0.1382	0.9150
Model Size	750 K	6135M	1564K	3140M	274M	10K	4K	7K	290M	49K
Training Time (s)	125.18	3755.18	792.61	33697.42	3226.02	124.50	3266.98	13.28	163140	39322.11
Prediction Time (s)	0.096	71.41	0.3	38.01	98.51	0.10	0.07	1.07	14147	16.5

TABLE IV
RESULTS OF THE DIFFERENT MODELS

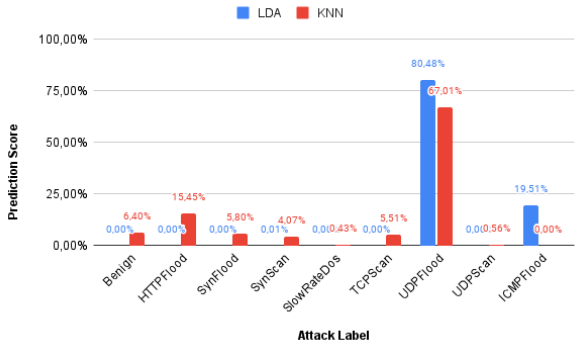

Fig. 3. Results of prediction for the UDP Flood attack using LDA and KNN mo

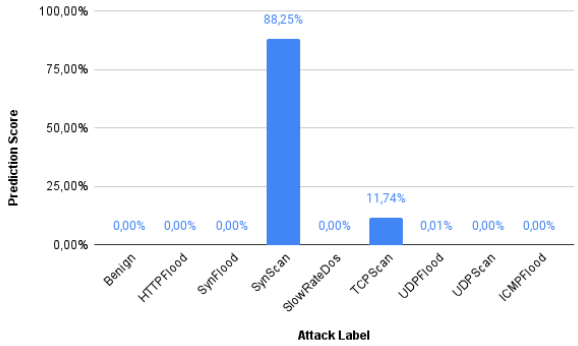

Fig. 4. Results of prediction for the Syn Scan attack using LDA model

orchestrator, considering the complex and dynamic nature of network traffic and the diversity of attack features.

In contrast to existing solutions, our proposition introduces a personalized model for each requester of the security service. This approach aims to meet the requirements of all users across different slices in a 5G network.

VI. CONCLUSION

This paper proposes a new approach to secure 5G mobile networks and beyond using adaptive ML models As-A-Service. The solution is designed, deployed, and validated using a Python-based discrete event simulator, where we modeled the service-request demands as a Poissonian process, and implemented the ML/DL algorithms using several Python libraries (e.g., TensorFlow, Keras, Scikit-learn). The Obtained results show the efficiency of our proposed approach in using a model orchestrator to manage the security service in the network. The trained models have different characteristics, making each of them appropriate for a particular situation. The final models are tested and proved their effectiveness under real-time traffic and with different attacks scenarios.

In future work, we intend to integrate the proposed models' orchestrator in a real 5G testbed and work on mitigating the detected attacks.

REFERENCES

[1] P. Kaur, M. Kumar, and A. Bhandari, "A review of detection approaches for distributed denial of service attacks," *Systems Science & Control Engineering*, no. 1, pp. 301–320, 2017.

[2] N. Ravi, P. V. Rani, and S. M. Shalinie, "Secure deep neural (seden) framework for 5g wireless networks," in *2019 10th International Conference on computing, communication and networking technologies (ICCCNT)*. IEEE, 2019, pp. 1–6.

[3] N. A. E. Kuadey, G. T. Maale, T. Kwantwi, G. Sun, and G. Liu, "Deepsecure: Detection of distributed denial of service attacks on 5g network slicing—deep learning approach," *IEEE Wireless Communications Letters*, vol. 11, no. 3, pp. 488–492, 2021.

[4] B. Bousalem, V. F. Silva, R. Langar, and S. Cherrier, "Deep Learning-based Approach for DDoS Attacks Detection and Mitigation in 5G and Beyond Mobile Networks," in *IEEE 8th International Conference on Network Softwarization (NetSoft)*, 2022, pp. 228–230.

[5] ——, "DDoS Attacks Detection and Mitigation in 5G and Beyond Networks: A Deep Learning-based Approach," in *IEEE Global Communications Conference (GLOBECOM)*, 2022, pp. 1259–1264.

[6] C. Benzaïd, M. Boukhalfa, and T. Taleb, "Robust self-protection against application-layer (d) dos attacks in sdn environment," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2020, pp. 1–6.

[7] A. Thantharate, R. Paropkari, V. Walunj, C. Beard, and P. Kankariya, "Secure5g: A deep learning framework towards a secure network slicing in 5g and beyond," in *2020 10th annual computing and communication workshop and conference (CCWC)*. IEEE, 2020, p. 852–857.

[8] A. Thantharate, R. Paropkari, V. Walunj, and C. Beard, "Deepslice: A deep learning approach towards an efficient and reliable network slicing in 5g networks," in *2019 IEEE 10th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*. IEEE, 2019, pp. 0762–0767.

[9] R. Bekkouche, M. Omar, R. Langar, and B. Hamdaoui, "Ultra-lightweight and secure intrusion detection system for massive-iot networks," in *IEEE International Conference on Communications (ICC)*. IEEE, 2022, p. 5719–5724.

[10] S. Samarakoon, Y. Siriwardhana, P. Porambage, M. Liyanage, S.-Y. Chang, J. Kim, J. Kim, and M. Ylianttila, "5g-nidd: A comprehensive network intrusion detection dataset generated over 5g wireless network," *arXiv preprint arXiv:2212.01298*, 2022.