A Comparison Experiment of Binary Classification for Detecting the GTP Encapsulated IoT DDoS Traffics in 5G Network

YeaSul Kim[1], YeEun Kim[1], Hwankuk Kim[2*]

[1] Department of Electronics Information and System Engineering, Sangmyung University, Republic of Korea
[2] Department of Information Security Engineering, Sangmyung University, Republic of Korea
{rlavnf10106, yeni0.0king}@gmail.com, rinyfeel@smu.ac.kr

Abstract

5G is characterized by ultra-low latency and the deployment of large-scale IoT environments. IoT devices with weak security can cause security problems such as network failures in 5G. To solve this problem, automated intrusion detection research using ML was being conducted. In previous studies, detection research using ML in the wired network environment was active, but it was relatively insufficient in the 5G network. In addition, the vast amount of traffic in IoT devices can create latency problems for intrusion detection with ML, making it difficult to achieve ultra-low latency 5G service objectives. Therefore, this study analyzed the meaning of the performance and required time of an optimized single ML model and ensemble learning experiment to detect in real time while ensuring high detection performance of large-capacity DDoS in 5G network. When the 5G GTP encapsulated traffic was collected and binary classification was performed, the optimized single ML model performed more than 99%. Especially, compared with ensemble learning, the experimental results showed similar performance and reduced detection time by at least 34 times. As a result of the experiment, it was shown that a single ML model optimized for detecting IoT DDoS in 5G with ultra-low latency is significant.

Keywords: 5G mobile network, Machine learning, Binary classification, IoT DDoS, DDoS detection

1 Introduction

As the number of internet of things (IoT) connected to the 5G network increases, the risk of 5G network DDoS attacks is increasing. IoT devices may cause security issues such as network failure (delay) in 5G core network devices due to various security issues [1-2]. This can lead to security issues due to infringement of availability [3-4]. Therefore, research is needed to analyze and detect IoT traffic in 5G mobile networks. 5G is characterized by ultra-low latency and the deployment of large-scale IoT environments. To prevent this, a security solution is needed to detect attacks in 5G [5-9]. To solve this problem, interest in automated intrusion detection using ML is increasing. However, the huge amount of data generated from 5G IoT devices can bring latency problems to intrusion detection using ML. This may give difficulty in achieving the purpose of ultra-low latency 5G network service. Therefore, in order to provide ultra-low latency 5G service, it is important to detect in real time while guaranteeing detection performance even for high-volume traffic of the 5G network. Representative methods for improving detection performance include hyperparameter tuning and ensemble learning [10-11]. Hyperparameter tuning is the process of finding the optimal combination of hyperparameters that greatly affect performance and is used to implement the optimal training model. Ensemble uses multiple learning algorithms to derive more accurate predictions but has high learning and detection costs (time required). Therefore, research is needed to improve the detection performance of large-capacity DDoS attacks in a 5G network environment and to detect them in real time.

Research to analyze attack detection performance by optimizing ML models in a wired network environment is being actively conducted [12-18]. In addition, other previous studies conducted attack detection studies using ML models in 5G networks [19-22]. However, studies that consider hyperparameter tuning to optimize ML models in 5G networks are currently lacking.

The 3rd Generation Partnership Project (3GPP) has defined the network data analytics function (NWDAF), a network function that integrates ML into the 5G network [23]. It plays a major role in collecting and analyzing data to operate the network. However, NWDAF does not deal with security data. Therefore, to strengthen the security of 5G networks, research is needed to internalize security in the network.

In this study, a single model optimization experiment was performed to detect large-capacity DDoS attacks with high performance in a 5G network environment. Then, the time required for each optimized single model and ensemble learning experiment was analyzed to study a model suitable for real-time detection. The configuration of the 5G network dataset is as follows. The dataset was constructed by collecting and processing packets through the 5G testbed directly built with the wired network DDoS attack dataset (Kitsune). The ML algorithms for binary classification of attacks were used: Decision Tree (DT), Random Forest (RF), K-Nearest Neighbors (KNN), and Support Vector Machine (SVM). We chose hyperparameters and cv fold values as factors for optimizing the models and studied the changes affecting the respective models by sequentially changing the values of that. The experimental results confirmed that the values of hyperparameters and cv fold, which play a major role in each model, are important elements for accuracy. The performance was improved by a comparative analysis of all experimental results. The following are the main contributions of the paper:

*Corresponding Author: Hwankuk Kim; E-mail: rinyfeel@smu.ac.kr
DOI: 10.53106/160792642022092305013

- First, this study established a 5G testbed environment to detect IP-based DDoS traffic in a 5G network and collected and utilized 5G GTP encapsulation traffic.
- Second, we performed optimization for each ML algorithm to improve 5G DDoS detection performance and compared/analyzed them.
- Third, we noted that it is important to fast detection of DDoS attacks in the 5G core (5GC) to meet the ultra-low latency 5G network service objective. The experiments showed the optimized single ML model has a lower required time and the similar detection performance compared to the ensemble technique.

The structure of this study is as follows. Section 2 reviews studies related to our study. Section 3 provides a detailed explanation of the concept of the 5G network and hyperparameter tuning used in this study. Section 4 describes the tuning method for improving DDoS detection performance in 5G networks. Section 5 shows the performance results and discussion of the ML model according to the tuning method described in Section 4. Section 6 presents the conclusions of this study and the direction for future research.

2 Related Works

In this section, we present studies similar to ours. In related studies, we introduce studies that performed hyperparameter tuning to improve DDoS detection performance on wired networks and studies that performed DDoS detection on 5G mobile networks.

2.1 DDoS Detection with ML Model in Wired Network

Sanchez et al. [12] used ML models (DT, KNN, Logistic Regression (LR), RF, and Multi-layer Perceptron (MLP)) that applied hyperparameters optimized by a grid search to detect DDoS attacks effectively in network traffic. The experiment was conducted with benign/malicious binary classification and showed over 98% accuracy using RF and DT for the CIC dataset. These previous work showed that if hyperparameter tuning is applied to an ML model, it can perform similar to deep learning [12].

Ma et al. [13] proposed an optimization model that adjusts the hyperparameters of Gaussian kernel SVM for anomaly detection in network traffic. The experiment was conducted with benign/malicious binary classification (for three datasets), which showed an accuracy of 99.98%. Through the experiment, these previous work confirmed that their classifier was better than other existing ML-based classifiers while saving computational costs [13].

Injadat et al. [14] proposed an effective anomaly detection framework that uses optimal parameters of SVM-RBF, RF, and RNN using the Bayesian Optimization (BO) technique. The BO method was applied to find the global minimum of the objective function and set the parameters of the classification algorithm. Overall, the BO-applied KNN showed optimal performance in terms of accuracy, precision, recall, and false alarm rate for the UNB ISCX 2012 dataset.

Misbahuddin et al. [15] clustered unlabelled traffic into normal and distributed DDoS and used optimized hyperparameters-applied KNN, SVM, and RF algorithms to verify them. The hyperparameter optimization was performed using Elbow (for KNN) and GridSearchCV (for SVM and RF), and their results were compared to the score obtained using the default hyperparameters. All three ML algorithms had improved accuracy when optimized hyperparameters were applied.

Injadat et al. [16] proposed an optimized ML-based Internet of Things (IoT) attack detection framework, which was constructed by combining a Bayesian optimization Gaussian Process (BO-GP) algorithm and a DT classification model. The BO-GP algorithm was used to maximize the detection performance by optimizing the hyperparameters of DT. As for the framework's performance, the testing accuracy was 99.99% for the Bot-IoT dataset, showing a superior detection rate compared to the default DT.

Ahmad et al. [17] proposed a framework using ML (RF, SVM, and ANN) with feature selection and hyperparameter tuning to detect malicious IoT traffic efficiently in a network-based intrusion detection system (NIDS). The hyperparameter tuning of RF and SVM was performed through GirdSearchCV. In the experimental results, the RF algorithm showed a binary classification accuracy of 98.67% and a multiclass classification accuracy of 97.37% for the Bot-IoT dataset.

Idhammad et al. [18] proposed an online sequential semi-supervised ML approach for DDoS detection. In the experiment, the hyperparameters of the ML algorithm were adjusted using a procedure based on the GridSearch CV method to achieve high performance. Next, a combination of hyperparameters that produce high performance was selected to assess the performance of the proposed ML approach. The experiment showed 98.23%, 99.88%, and 93.71% accuracy for the NSL-KDD, UNB ICSX 12, and UNSW-NB15 datasets, respectively.

Previous studies used optimized hyperparameters-applied ML for efficient DDoS attack detection in wired networks [12, 17-18]. Particularly, other studies [13-16] showed performance improvement in detection using optimized hyperparameters over the existing default models. However, hyperparameters of the single ML model were not analyzed in order to optimize to perform DDoS detection with ultra-low latency using ML in the 5G environment, so there was a limit to applying it to detection in the 5G core network.

2.2 5G Network DDoS Detection with ML Model

Yadav et al. [19] proposed a DL algorithm to detect vast amounts of IoT malicious network traffic on 5G networks. For implementation purposes, they used XGBboost, Adaboost, Extra Tree Classifier, and Random Forest Classifier, which were ensemble-based ML techniques. The proposed algorithm and four additionally provided algorithms (XGBboost, Adaboost, ExtraTree, RF) showed 99.76%, 98.4%, 98.3%, 98.3%, and 98.6% accuracy, respectively, for the UNSW NB15 dataset.

Li et al. [20] proposed an ML-applied intelligent IDS based on a software-defined 5G architecture. The proposed classification algorithm, KA, was constructed by combining k-Means++ and Adaboost. In the experiment, KA and the compared algorithms (Gradient Boosting DT, DT, and SVM) showed 94.48%, 93.09%, 92.65%, and 90.14% accuracy, respectively, for the NSL-KDD dataset.

Alamri et al. [21] analyzed an appropriate ML to protect SDN-based 5G networks from DDoS attacks. This previous

work prepared an SDN-based 5G network with datasets (CICDDoS 2019, NSL-KDD) to build and test the proposed ML model. In the experimental results, the XGBoost algorithm improved the security performance of the SDN controller over the other ML algorithms (LR and RF) compared.

Kim et al. [22] studied a feature selection-applied ML method to detect IoT DDoS attacks with low time complexity in a 5G mobile network environment. These work constructed a virtual 5G experimental environment and used the Kitsune dataset to build a 5G GTP-U dataset [22]. In the experimental results, the DT, RF, KNN, and Stacking algorithms showed 88.69%, 92.03%, 82.46%, and 97.18% accuracy, respectively.

Previous studies used ML to detect DDoS attacks effectively on 5G mobile networks [19-22]. Especially final previous work is similar to our study in that it constructed a virtual 5G environment with a Kitsune dataset, and collected a 5G dataset on its own, which was used to perform detection [22]. They used one gNB in the 5G testbed, however, we have experimented in a more realistic 5G environment by building multiple gNBs. In addition, these studies did not consider hyperparameter tuning for performance improvement [19-22]. As far as we know, hyperparameter optimization research for DDoS detection is focused on wire network environments. Therefore, we performed a single ML model optimization experiment through hyperparameter tuning to effectively perform IoT DDoS detection with ultra-low latency in a 5G mobile network environment.

3 Background

3.1 5G Core Network and SBA

5GC network is part of the 5G network system (5GS) [24]. The 5GC employs a service-based architecture (SBA), which divides the necessary functionalities into different network functions (NFs). In addition, 5G SBA consists of control and user plane separation (CUPS). The NFs on the 5GC, which are linked to the base station and send or control user data, is the Access and Mobility Function (AMF), Session Management Function (SMF), and User Plane Function (UPF). An AMF is a function that manages network access and mobility, and an SMF is a function that manages sessions between terminals and the network. AMF and SMF are responsible for the control plane where control signals such as communication connection establishment are processed. UPF is a function that provides user packet routing between the base station and data network (DN) and connections between devices. It handles the user plane where user data is transferred.

The NWDAF is one of the 5GC NFs defined by 3GPP and serves to provide network analysis information according to the request of the NF. NWDAF is composed of two types: Analytics logical function (AnLF), which serves as an interface to other NFs, and Model Training logical function (MTLF), which is an ML-trained model, for analyzing network data with AI technology [23, 25]. As such, 5G defines NFs using ML techniques to construct data collection and analysis in the 5GC. This study has the purpose of prior research to build Security NF grafted with ML technology like NWDAF in 5GC.

3.2 5GC NF and Interface N3

The UPF acts as an external PDU session point of interconnection to DN and supports N3, N4, and N6 interfaces. The N3 interface is between the RAN and the UPF [26]. And GPRS Tunnelling Protocol (GTP) is a protocol for encapsulating and tunnelling IP packets sent to and from users on the Internet/packet data network. The GTP has been used in NR (5G), LTE (4G), UMTS (3G), and GPRS (2.5G) Core Networks [27-29]. GTP is a set of three separate protocols: GTP Control (GTP-C), GTP User (GTP-U), and GTP Prime (GTP'). GTP-C is used within the GPRS core network for signaling between gateway GPRS support nodes (GGSN). GTP-U carries user data within the GPRS core network, and between the radio access network and core network. The GTP-U supports multiplexing of the traffic from different packet data unit (PDU) sessions by tunnelling user data over the N3 interface in the core network [26]. GTP' uses the same message structure as GTP-C and GTP-U [30]. We collected GTP-U packets directly from the N3 interface of the 5GC. Also, in this paper, GTP means GTP-U.

4 Experimental Methodology

4.1 Experimental Flow

The experimental order is (a) 5G GTP Dataset Configuration and (b) ML Model Performance Optimization shown in Figure 1. 5G GTP dataset configuration is the task of configuring the raw dataset collected from the wired network into the 5G network dataset. A 5G dataset was created by conducting an experiment in the testbed where the 5G environment was built. In ML model performance optimization, an ML model tuning was conducted to increase the accuracy of binary classification of 5G GTP packets as benign and malicious. Then, the results of optimizing the performance of the single ML model were summarized by comparing and analyzing the experimental results.

4.2 Experiment Environment

Configuring 5G GTP Dataset requires 5G Testbed and benign and malicious packet datasets of the wired network. In this section, how to configure the 5G Testbed, its structure map, and the selected dataset are explained.

To build a 5G testbed, UERANSIM and Open5GS were used. Figure 2 shows the 5G testbed structure map. UERANSIM is an open-source available on GitHub and supports 5G UE and RAN (gNB). UE and gNB can be viewed as 5G mobile phone and base station. UERANSIM support to be executed with Open5GS and Free5GC, which are 5GC network open sources [31]. UERANSIM can be connected to the 5GC to test the functions. We have built more than one UERANSIM. Through this, an experiment environment as similar to reality as possible was constructed by using as many UEs and gNBs as possible for the experiment. Open5GS is an open-source available on GitHub and supports the 5GC based on the 3GPP. Currently, it supports 5GC NFs including AMF, SMF, PCF, UDR, UDM, NSSF, AUSF, NRF, and UPF [32]. We have built the 5G testbed so that the UEs are connected to the 5GC via gNBs.

The Kitsune Network DDoS Attack Dataset, which has been open to Kaggle since 2018, was used as the benign and malicious packet dataset of the wired network [33-34]. The

Kitsune provides the Original Network Packet (.pcap) and Benign/Malicious label (.csv) for 9 types of attacks that can be seen in real network intrusions. The nine attacks are OS Scan, Fuzzing, Video Injection, ARP MitM Active Wiretap, SSDP Flood, SYN DoS, SSL Renegotiation, and Mirai [33]. In this study, all nine are not used, but seven attacks are used except for Video Injection and Mirai. When we checked the type of source IP for each attack, other attack types had at least 2 to 8 source IP types, whereas Mirai had a maximum of 21 source IPs, so it was excluded from the experiment. Also, when checked the types of protocol, only Video Injection include the Logical-Link Control (LLC) protocol of the ethernet layer, so it was excluded.

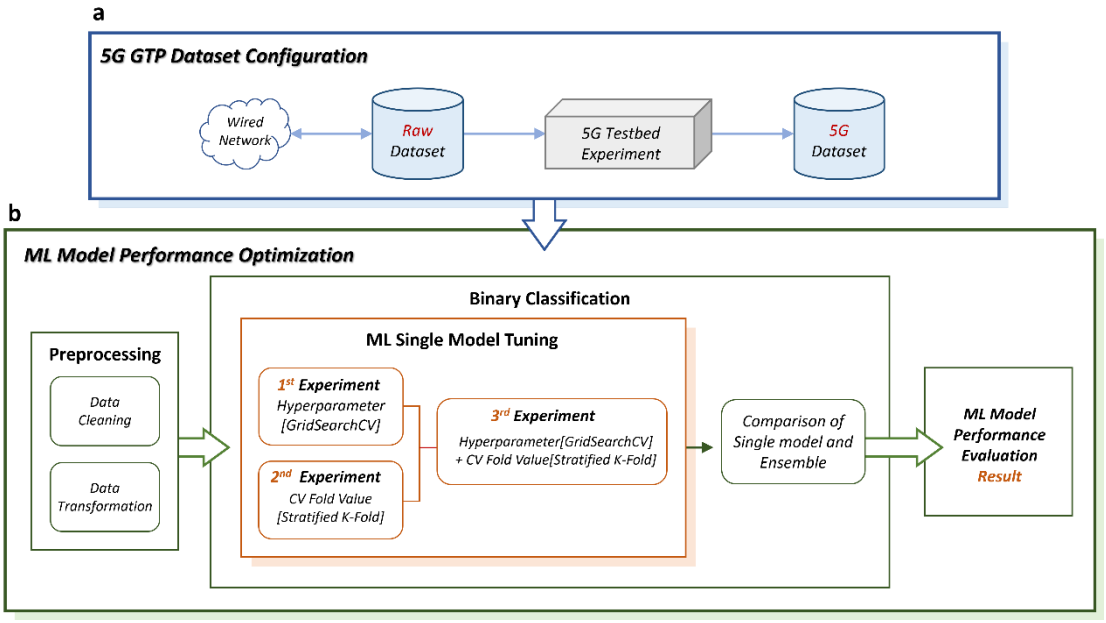

Figure 1. Experimental order

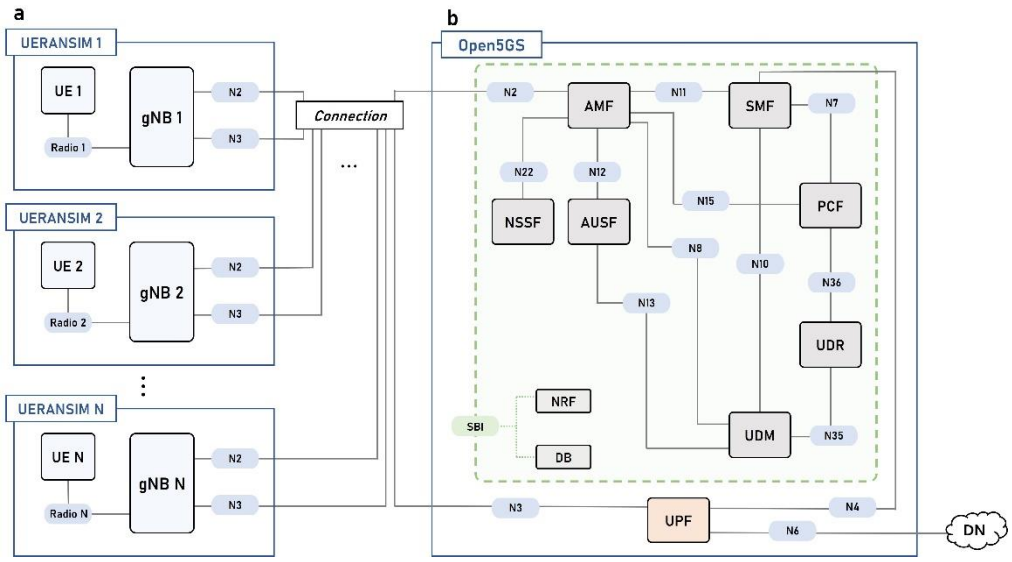

Figure 2. 5G Testbed structure map

4.3 5G Dataset: GTP encapsulated IoT DDoS Traffics

5G GTP dataset configuration is a task to configure Kitsune dataset (.pcap) as 5G GTP dataset (.csv). Figure 3 describes the 5G GTP dataset configuration procedure.

First, the original network packet (.pcap) provided by the Kitsune dataset is divided using a benign/malicious label (.csv). This operation is performed to classify benign and malicious through binary classification. When this operation is completed, the original network packet of 7 attacks is divided into 7 benign pcap files and 7 malicious pcap files.

Second, the benign/malicious pcap file divided by attacks is divided once more based on the source IP (20 types). This operation is performed to proceed with the experiment by setting the UE and gNB of the 5G testbed based on the source

IP. The network IP address is composed of the network ID and host ID. The IP address is distinguished into the network ID and host ID through various classes (e.g. A, B, C, D, and E classes). In this study, we distinguished between the network ID and host ID based on class C. As shown in Table 1, gNBs were specified based on the network ID, and UEs were specified based on the host ID. Then, experiments were conducted by dumping packets according to the UE and gNB specified for each IP. The number of gNBs and UEs used in the experiment is 7 and 20, respectively.

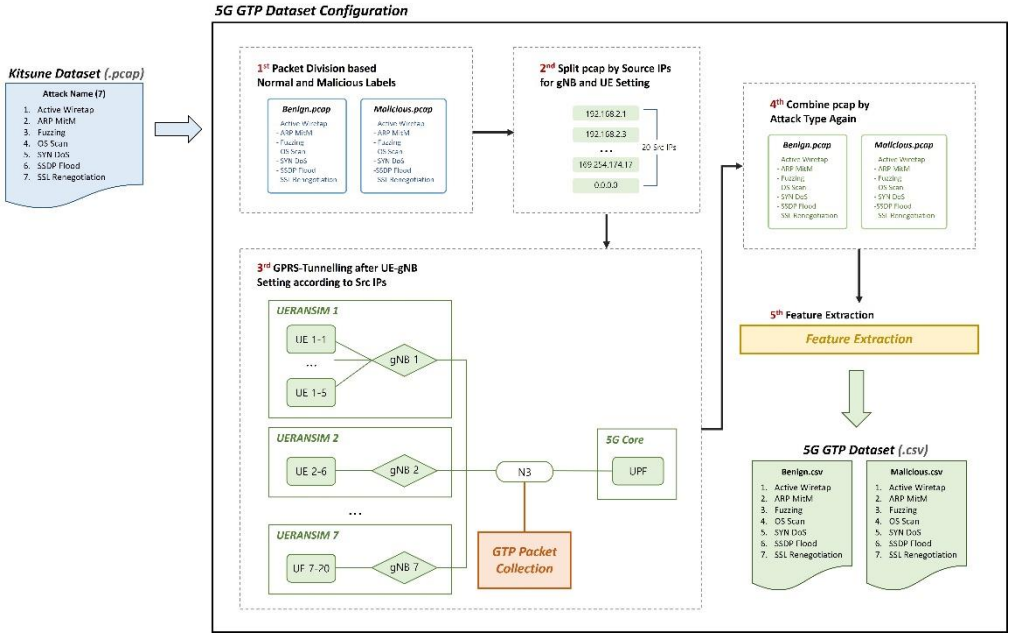

Figure 3. 5G GTP dataset configuration procedure

Table 1. gNB-UE allocation by source IP bandwidth

gNB	UE	Source IP
gNB 1	UE 1~5	192.168.2.1~15
gNB 2	UE 6	192.168.2.3
gNB 3	UE 7~8	192.168.100.5~222
gNB 4	UE 9~17	169.254.3.1~107
gNB 5	UE 18	169.254.176.87
gNB 6	UE 19	169.254.174.17
gNB 7	UE 20	0.0.0.0

Third, the GPRS-Tunnelling experiment was conducted using the 5G testbed. First, packets are played in the UE specified based on the source IP, and then the packet transmitted to the 5GC is collected. When UEs play packets on the 5G testbed, the packet passes through the 5GC NF, the UPF, through the gNB. The network interface through which this packet passes is N3. In the N3 network interface, a GTP header is added to the existing packet through GPRS-Tunnelling. We captured GTP packets passing through the N3 interface.

Fourth, GTP packets by each source IP collected from the 5G testbed are again combined for each attack into 7 benign pcap files and 7 malicious pcap files.

Fifth, to perform ML binary classification, feature extraction is performed to convert the pcap file to a csv file. Considering the IP types (IPv4, IPv6) and protocol types (ICMP, ARP, TCP, UDP, IGMP, GTP) of the packets collected in the 5G testbed, 93 header information was extracted using Wireshark's Tshark. And labels are specified for binary classification. Labels for benign and malicious are set to 0 and 1, respectively.

When all steps are completed, the 5G GTP dataset (.csv) is finally configured Table 2. The number of benign and malicious packets is 6,500 for each attack, for a total of 45,500 benign and 45,500 malicious packets. The ratio of the number of benign and malicious packets used in the experiment is 50:50.

Table 2. Dataset for binary classification experiments

Type	Attack name (count)	Total (label)
Benign	Active Wiretap (6500) ARP MitM (6500) Fuzzing (6500) OS Scan (6500) SYN DoS (6500) SSDP Flood (6500) SSL Renegotiation (6500)	45500 (0)
Malicious	Active Wiretap (6500) ARP MitM (6500) Fuzzing (6500) OS Scan (6500) SYN DoS (6500) SSDP Flood (6500) SSL Renegotiation (6500) Active Wiretap (6500)	45500 (1)

4.4 Experiments Design

Pre-processing before ML model optimization consists of (1) Data Cleaning and (2) Data Transformation. Data cleaning

removes a feature (33) that has no value for all data among the extracted features (93). Data transformation does the following: Converts a hexadecimal number to a decimal number. The object-type variable and the float-type variable are transformed into an Integer-type variable. For IP features, if Wireshark Tshark is used, "." is extracted as it is and saved as an object type variable (e.g., 192.168.2.1). For the experiment, remove "." and convert it to an integer type variable.

In the ML model performance optimization [35-37], ML model optimization steps were conducted to increase the accuracy of binary classification of 5G GTP packets into benign and malicious ones. Accuracy was used as a performance evaluation metric for ML models. DT, RF, KNN, and SVM are used as the ML algorithm.The tuned values are hyperparameter [GridSearchCV] and cv fold value [Stratified K-Fold].

The ML model optimization was performed in three orders: First, the hyperparameters were tuned for each ML algorithm. The method used was grid search, and the cross-validation value was set to 5, which is the default. Second, the cross-validation values were tuned for each ML algorithm. The method used was Stratified K-Fold, and the hyperparameter value was set to default. Third, the tuning was performed by applying the first hyperparameter result and the second cross-validation result comprehensively.

4.4.1 Single ML Model Optimization

(1) Hyperparameter. Hyperparameter tuning was performed to optimize the ML model for each of the four algorithms performed in the step. Grid search can specify hyperparameters and values to be tuned.

Table 3. GridSearchCV hyperparameter values

Algorithm	Hyperparameter
DT	-max_depth: range (1, 20, 1) -min_samples_split: range (2, 20, 1) -min_samples_leaf: range (1, 20, 1)
RF	-max_depth: range (1, 20, 1) -min_samples_split: range (2, 20, 1) -min_samples_leaf: range (1, 20, 1)
KNN	-weights(uniform), n_neighbors: range (1, 20, 1) -weights(distance), n_neighbors: range (1, 20, 1) -metric(manhattan), n_neighbors: range (1, 20, 1) -metric(euclidean), n_neighbors: range (1, 20, 1)
SVM	-kernel(linear), C: [1, 10, 100, 300, 1000, 3000, 10000, 30000] -kernel(rbf), C: [1, 10, 100, 300, 10000], gamma: [0.00001, 0.0001,0.001, 0.01, 0.1, 1.0, 3.0]

Table 3 shows the types and values of hyperparameters for each algorithm in which the step was conducted.

Three of the same hyperparameters were tuned for DT and RF. The step was carried out one hyperparameter at a time. Thus, it was performed three times for DT and RF, respectively. The tuning was performed for one hyperparameter at a time to examine the accuracy change trend of ML affected by one hyperparameter and check its impact. First, max_depth represents the tree's maximum depth and is a parameter that splits until the class value is perfectly determined or splits until the number of data points becomes smaller than min_samples_split. The range of the value is between 1 and 20, and the value is set to increase by 1 at a time. min_samples_split represents the minimum number of samples required to split internal nodes, and as its value is set smaller, the number of split nodes increases, which may increase the possibility of overfitting. Since the minimum value that can be set is 2, the value is set to increase by 1 from 2 to 20. min_samples_leaf represents the minimum number of samples that must exist on a leaf node, and it is used to control overfitting.

The most important hyperparameter for performance in the KNN algorithm is n_neighbors, which controls the number of neighbors to be searched. The value of n_neighbors was set to increase in increments of 1 from 1 to 20, and the tuning was performed four times in total by changing the weights or metric. First, weights is used for the prediction, with uniform giving the same weight to each neighbor and distance giving greater weights to a nearby neighbor than to a distant neighbor. The metric refers to a hyperparameter that changes the distance measurement method, and two methods were used: Manhattan and Euclidean.

In SVM, different hyperparameters are selected depending on the type of kernel used. Two kernel types were used in the step: rbf and linear. In the case of the kernel-linear, data is separated linearly. However, since it is impossible to separate them perfectly, a strategy that

allows the placing of data samples in other classes is used. The hyperparameter that can control it is C. As the value of C decreases, a larger number is allowed, and as it increases, a smaller number is allowed. In the step, seven values between 1 and 30,000 were set, as shown in Table 3. Kernel-rbf maps the given data to a high-dimensional space, which is three-dimensional. This enables the classification of data, which could not be classified linearly. There are two hyperparameters that help optimize the kernel-rbf: C and gamma. C has the same role as that of the kernel (linear). Gamma determines the range of data that affects the curvature of the decision boundary. As the gamma value increases, the decision boundary becomes more curved, and as it decreases, the decision boundary becomes closer to a straight line. In the step, five values were set for C and seven values for gamma, as shown in Table 3.

(2) CV Fold. CV fold value tuning was performed for optimized cross-validation of the four algorithms performed in this step. The cross-validation was to perform training and evaluation with the training dataset and validation dataset built as separate sets to prevent data bias. In the training process of the ML model, the fold value is set to divide the data rather than dividing it just once, and the performance of the learning model between each fold is compared to derive a mean value. The stratified K-fold used in the step is a cross-validation method used to prevent the distribution from being biased toward one side due to a very small or large value of a certain label. The tuning step was performed by increasing the fold value in increments of one from 1 to 15 for the four ML algorithms. The four ML models were all the same as the

Scikit-Learn model used in the first step, and default values were set for the hyperparameters.

(3) GridSearch. The step was performed by comprehensively applying the hyperparameter results of and the cross-validation results. The performance results for the hyperparameters are checked for each algorithm in hyperparameter step, and the value to be used in this step is determined by dividing it into three cases. The first case is for one hyperparameter that has the highest degree of influence among the values changed regardless of high accuracy. In this study, we established the following hypothesis for the degree of influence: Subtract the lowest accuracy from the highest accuracy.

The second case is for one hyperparameter that shows high accuracy, regardless of hyperparameters that have a high degree of influence. The third case is for one hyperparameter, which is the same in the first and second cases. If the first and second hyperparameters are different, both cases are dealt with in this step, and if the first and second hyperparameters are the same, only one case is treated. In cv fold value, the performance result is examined based on the optimized fold value for each algorithm, and three cases are divided to determine the value for optimizing the model: first, the cv fold value at the time of the highest accuracy; second, the cv fold value at which accuracy changes and the change is maintained; and third, if the first and the second are not the same, the lower cv fold value between the two to reduce time and calculation.

4.4.2 Comparison of Single model and Ensemble

The experimental results of single ML model optimization (excluding SVM) and stacking of the ensemble technique were compared. This was to deal with the performance and time required for the single ML model optimized through comparative experiments to detect IoT DDoS traffic in the 5G network environment. Since the accuracy of SVM is significantly lower than that of other models, this comparison did not address SVM. The base model of Stacking was DT, KNN, and SVM-rbf, and binary classification was performed using Logistic Regression. The base model excludes RF, the algorithmic ensemble technique we used. Since tuning of the stacking base model is not performed, both hyperparameter and cv fold values were set to default.

5 Results and Discussion

First, ML model optimization will be described. Finally, the results for optimizing the performance of the ML model were summarized by comparing and analyzing the three experimental results. In addition, a comparative analysis was conducted in terms of performance and required time of the optimized single ML model and the existing performance improvement method, ensemble. Through ensemble learning and comparative analysis, a single ML model will confirm its usefulness in detecting IoT DDoS traffic in a 5G environment.

5.1 Single ML Model Optimization Result

5.1.1 Hyperparameter Result

The results of hyperparameter tuning for each ML model are explained. Figure 4 to Figure 7 shows (1) "Highest/Lowest Accuracy" and (2) "The Degree of Influence" for hyperparameter cases set in Table 4.

The performance of the DT algorithm according to the change of hyperparameter values was analyzed. As a result, the hyperparameter that showed the highest accuracy (79.195%) is "min_samples_split (15)". And the hyperparameter that showed the highest degree of influence (11.678) is "max_depth". The highest accuracy of "max_depth" is 79.193% when the value is 13, and the lowest accuracy is 67.461% when the value is 5.

The performance of RF algorithms according to changes in hyperparameter values was analyzed. As a result of the step, the hyperparameter that showed the highest accuracy (87.418%) is "min_samples_split (15)". In addition, the hyperparameter showing the highest degree of influence (27.985) is "max_depth". The highest accuracy is 86.357% when the value is 18, and the lowest accuracy is 58.372% when the value is 1.

We analyzed the performance of the KNN algorithm according to the change of hyperparameter values. As a result of the step, the hyperparameter that showed the highest accuracy (79.684%) is weights: distance, n_neighbors (2). The hyperparameter showing the highest degree of influence (2.568) is also weights:distance. For this hyperparameter, the highest accuracy is 79.684% for n_neighbors (2), and the lowest accuracy is 77.116% for n_neighbors (3). In KNN, the hyperparameter showing the highest accuracy and influence is the same as weights:distance, n_neighbors (2).

Table 4. Optimization results by all algorithms hyperparameter

Algorithm	Hyperparameter	Highest acc (value)	Lowest acc (value)	The degree of. influence
DT	max_depth	79.179% (13)	67.461% (5)	11.678
	min_samples_leaf	79.194% (10)	76.790% (3)	2.374
	min_samples_split	79.195% (15)	76.796% (17)	2.399
RF	max_depth	86.357% (18)	58.372% (1)	27.985
	min_samples_leaf	86.892% (2)	73.164% (17)	13.728
	min_samples_split	87.418% (15)	76.995% (2)	10.423
KNN	weights:uniform	79.583% (2)	77.125% (3)	2.458
	weights:distance	79.684% (2)	77.116% (3)	2.568
	metric:manhattan	79.222% (2)	78.027% (19)	1.195
	metric:euclidean	79.583% (2)	77.125% (3)	2.458
SVM-linear	C	57.392% (100)	52.215% (300)	5.177
SVM-rbf	C: 1	68.679% (gamma:0.00001)	50.456% (gamma:3)	18.223
	C: 10	67.956% (gamma:0.00001)	50.028% (gamma:3)	17.928
	C: 100	67.901% (gamma:0.00001)	50.045% (gamma:3)	17.856
	C: 300	67.853% (gamma:0.00001)	50.056% (gamma:3)	17.797
	C: 1000	71.349% (gamma:0.00001)	50.050% (gamma:3)	21.299

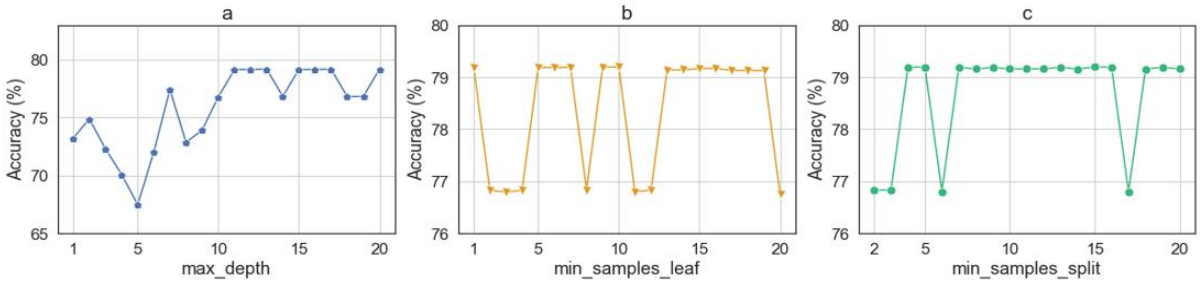

Figure 4. Trend of tuning results by DT hyperparameter

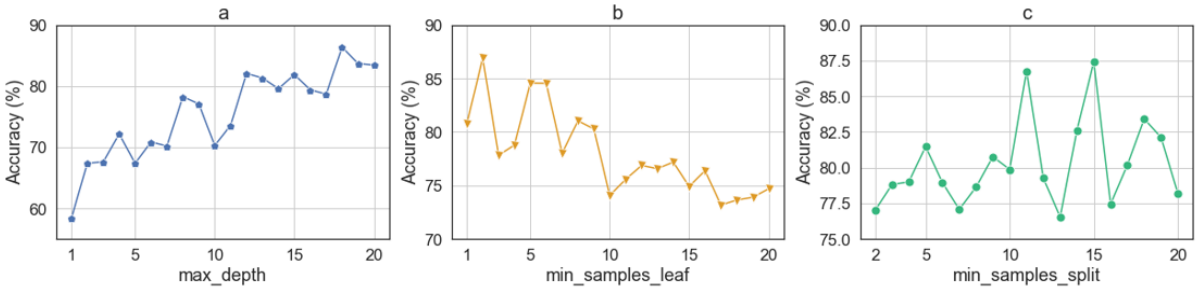

Figure 5. Trend of tuning results by RF hyperparameter

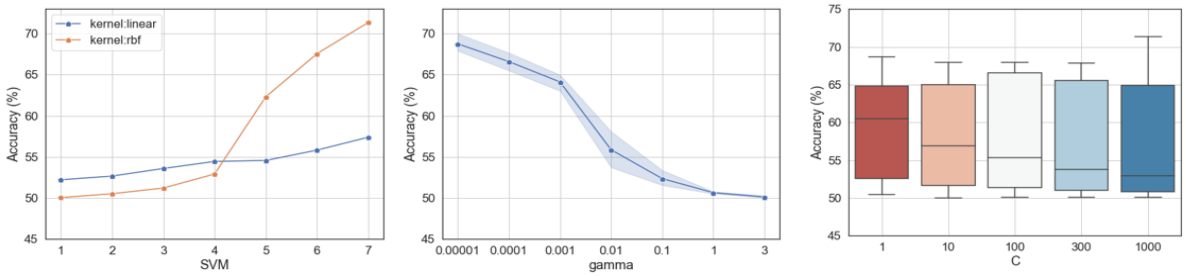

Figure 6. Trend of tuning results by SVM hyperparameter

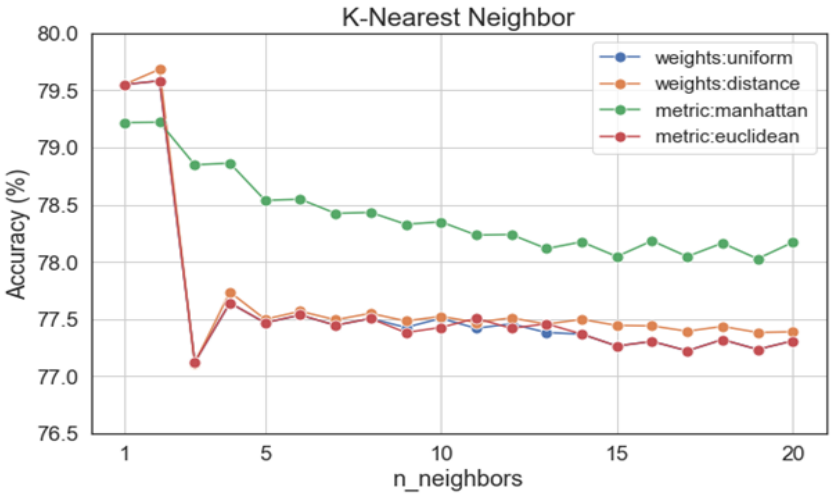

Figure 7. Trend of tuning results by KNN hyperparameter

We analyzed the performance according to the change in the hyperparameter value of the SVM algorithm. The hyperparameters set for SVM are different depending on the kernel type. First, the experimental results of SVM-linear will be described. As a result of the step, the result value of the hyperparameter that showed the highest accuracy (57.342%) is C (100). And the result value of the hyperparameter showing the highest degree of influence (5.177) is also C (100). The following describes the experimental results of SVM-rbf. In the trend of accuracy, high accuracy is obtained when the value of C is high and the value of gamma is low. Conversely, low accuracy is shown when the value of C is low and the value of gamma is high. In the experimental results, the values of the hyperparameters that show the highest accuracy (71.349%) are 1,000 for C and 0.00001 for gamma. Furthermore, the values of the hyperparameters that show the highest degree of influence (21.299) are 1,000 for C and 0.00001 for gamma. In the case of SVM-rbf, the same hyperparameter values show the highest accuracy and the highest degree of accuracy. Comparing the kernel of SVM, SVM-rbf (71.349%) shows higher accuracy than SVM-linear (57.392%). Among the kernels of SVM, rbf shows good performance.

5.1.2 CV Fold Result

Figure 8 shows the trend of accuracy change according to cv fold value tuning for each ML model. Table 5 shows the results in terms of the highest, lowest accuracy and the difference between the highest/lowest accuracy for each ML model. In the results, three algorithms (DT, RF, KNN) show the same trend. In the trend, the accuracy rises when the fold value is below 7. It decreases at a value of 7 but increases again afterward and is maintained. For DT, RF, and KNN, the value of folds for optimizing performance is determined to be 9, a point where the accuracy increases and is maintained, not 8 where the accuracy starts to rise.

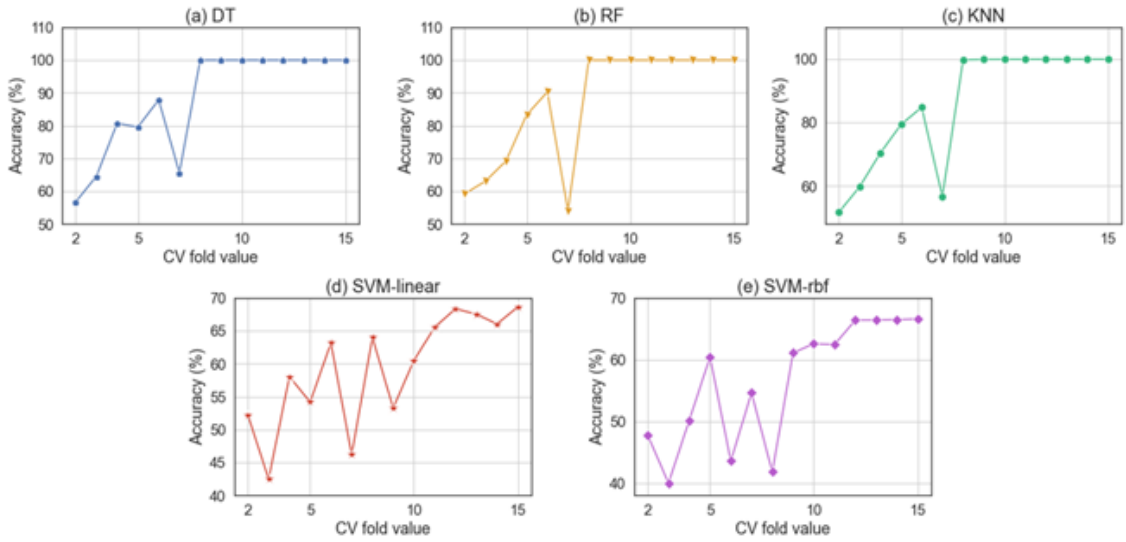

Figure 8. CV fold value results trend by algorithms

In the case of SVM (linear), the accuracy rises and falls repeatedly without having a point where the accuracy is maintained. For SVM (linear), the value of the number of folds for optimizing the algorithm's performance is determined to be 15, which shows the highest accuracy (68.617%) among the results. In the case of SVM (linear), the accuracy rises and falls repeatedly without having a point where the accuracy is maintained. For SVM (linear), the value of folds for optimizing the algorithm's performance is determined to be 15, which shows the highest accuracy (68.617%) among the results. In the case of SVM-rbf, the accuracy fluctuates, increasing and decreasing repeatedly, when the cv fold value is below 9, and then increases afterward. Furthermore, it shows the trend that the accuracy is maintained starting from a point where the value is 12. For SVM-rbf, the value of folds for optimizing the algorithm's performance is determined to be 13, a point where the accuracy increases and is maintained, not 12 where the accuracy starts to rise.

Table 5. CV fold value results

Algorithm	Highest acc (value)	Lowest acc (value)	Diff in. acc
DT	99.995% (10)	56.705% (2)	43.29%
RF	99.986% (15)	53.954% (7)	46.032%
KNN	99.926% (15)	51.724% (2)	48.202%
SVM-linear	68.617% (15)	42.492% (3)	26.125%
SVM-rbf	66.571% (15)	39.997% (3)	26.574%

5.1.3 GridSearch Result

Hyperparameter tuning and cv fold tuning were performed and the results were analyzed, and values for optimizing the model were selected. In this section, (1) this tuning is performed by synthesizing the results of two steps, and (2) comparative analysis is performed to determine which tuning shows the highest performance for each algorithm. Table 7

shows the experimental results of all stage for each algorithm optimization.

Hyperparameter tuning, which performed only hyperparameter tuning without considering cross-validation, showed the lowest accuracy in all algorithms except SVM-rbf among the three experiments. KNN showed the highest accuracy of cv fold tuning, which performed only cross-validation tuning without considering hyperparameter tuning. RF and SVM (linear, rbf) showed the highest accuracy of gridsearch result considering both hyperparameters and cross-validation. It can be seen that DT has the same accuracy as the highest accuracy in cv fold tuning and gridsearch.

5.2 Comparison Result of Single Model and Ensemble

Table 6 shows the highest accuracy and time required in the DT, RF, and KNN experiment results, and the stacking accuracy and time required without tuning. As a result, the untuned stacking accuracy showed the highest performance. In our experiments, DT, RF, and KNN are lower than Stacking, but they perform similarly. Therefore, the tuned DT, RF, and KNN have relatively lower performance than stacking but require significantly less time to train the model.

Table 6. Comparison result of single model and ensemble

Algorithm	Model tuning	Acc	Time required
DT	O	99.989%	0.00215h
RF	O	99.964%	0.00328h
KNN	O	99.968%	0.03575h
Stacking	X	99.996%	1.24882h

5.3 Comparison with Previous Works

Most of the previous studies were introduced in Section 2. Among the works introduced in Section 2, we would like to compare our study with the previous study of 5G Mobile Network DDoS detection to examine the differences. We compared in terms of (1) Dataset, (2) ML/DL Model, (3) Key Approach, and (4) Results. Previous studies commonly performed research for anomaly detection using ML in a 5G network environment. However, as shown in Table 8, the results of all studies are different. Especially, the research results of this paper confirmed that a single ML model optimized through ensemble learning and comparative experiments in a 5G network environment is useful for IoT DDoS detection with ultra-low latency.

Table 7. Final single ML model tuning results

Algorithm	Hyperparameter		CV fold		GridSearch		
	value	Accuracy	Fold	Accuracy	value	Fold	Accuracy
DT	max_depth(13)	79.193%	9	99.989%	max_depth(13)	9	99.989%
	min_samples_split(15)	79.195%			min_samples_split(15)		99.968%
RF	max_depth(18)	86.357%	9	99.867%	max_depth(18)	9	99.964%
	min_samples_split(15)	87.418%			min_samples_split(15)		99.989%
KNN	weights:distance(2)	79.684%	9	99.968%	weights:distance(2)	9	99.950%
SVM	kernel(linear), C(100)	57.392%	15	68.617%	kernel(linear), C(100)	15	69.145%
	kernel(rbf), C(1000), gamma(0.00001)	71.349%	13	66.404%	kernel(rbf), C(1000), gamma(0.00001)	13	84.629%

Table 8. Comparison with previous works

Authors	Dataset	ML/DL Model	Key Approach	Results
Yadav et al. [19]	- UNSW-NB15	DL (Softmax)	- Binary Classification - AutoEncoder	Improved accuracy (99.76%) compared to existing IDS
Li et al. [20]	- KDDCup99 - NSL-KDD	ML (KA that combined Kmean++ and Adaboost)	- Binary Classification - Feature Selection	Improved accuracy (99.97%) compared to existing IDS
Alamri et al. [21]	- CICDDoS2019 - NSL-KDD	ML (XGBoost)	- Binary/Multi Classification - Using Ensemble	Improved accuracy (99.9%) of the proposed SDN
Kim et al. [22]	- Kitsune	ML (DT, RF, KNN, Stacking)	- Multi Classification - Feature Selection	Reduced time complexity (88.89%) and improved accuracy (97.18%)
Our research	- Kitsune	ML (DT, RF, KNN, SVM, Stacking)	- Binary Classification, - Single Model Optimizing	Improved required time (at least 34 times) and accuracy (99.98%)

6 Conclusion

This study conducted a comparative experiment to detect large-capacity IoT DDoS traffic in 5G with high performance and ultra-low latency. We optimized a single ML model and compared it with ensemble learning, which is a conventional performance improvement method. In addition, in order to consider IoT DDoS traffic of 5G network, a 5G testbed was established and GTP encapsulated traffic was collected and used.

Among the optimized single ML algorithms (DT, RF, KNN, SVM) used in the experiment, DT, RF, and KNN showed more than 99% accuracy. In particular, we confirmed that the optimized single ML model has a similar accuracy of more than 99% and at least 34 times faster the required time compared to ensemble learning, in experiment that comparison of optimized single model and ensemble. Experimental results shown that the optimized single ML model was meaningful in terms of performance and time required for real-time IoT DDoS detection in 5G.

In addition, we presented the performance results of optimizing the training model. This was an experiment conducted assuming that the performance results for the testset would also be good. In the future, research is needed to configure the 5G testset and present performance results. In addition, in order to detect IoT DDoS with ultra-low latency in 5G, we plan to conduct ML-based native security framework research on 5GC in the future.

Acknowledgement

The research was funded by 2020 research Grant from Sangmyung University.

References

[1] M. Alizadeh, K. Andersson, O. Schelen, A survey of secure internet of things in relation to blockchain, *Journal of Internet Services and Information Security (JISIS)*, Vol. 10, No. 3, pp. 47-75, August, 2020.

[2] H. Kim, 5G core network security issues and attack classification from network protocol perspective, *Journal of Internet Services and Information Security (JISIS)*, Vol. 10, No. 2, pp. 1-15, May, 2020.

[3] A. Abhishta, W. van Heeswijk, M. Junger, L. J. M. Nieuwenhuis, R. Joosten, Why would we get attacked? An analysis of attacker's aims behind DDoS attacks, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, Vol. 11, No. 2, pp. 3-22, June, 2020.

[4] M. Kolomeets, A. Chechulin, I. Kotenko, Bot detection by friends graph in social networks, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, Vol. 12, No. 2, pp. 141-159, June, 2021.

[5] S. Nowaczewski, W. Mazurczyk, Securing future internet and 5g using customer edge switching using DNSCrypt and DNSSEC, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, Vol. 11, No. 3, pp. 87-106, September, 2020.

[6] G. Choudhary, J. Kim, V. Sharma, Security of 5G-mobile backhaul networks: A survey, *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, Vol. 9, No. 4, pp. 41-70, December, 2018.

[7] H. Moudoud, L. Khoukhi, S. Cherkaoui, Prediction and Detection of FDIA and DDoS Attacks in 5G Enabled IoT, *IEEE Network*, Vol. 35, No. 2, pp. 194-201, March/April, 2021.

[8] C. Lai, R. Lu, D. Zheng, X. S. Shen, Security and privacy challenges in 5g-enabled vehicular networks, *IEEE Network*, Vol. 34, No. 2, pp. 37-45, March/April, 2020.

[9] A. Abouaomar, M. Elmachkour, A. Kobbane, H. Tembine, M. Ayaida, Users-Fogs association within a cache context in 5G networks: Coalition game model, *Proceedings of the 2018 IEEE Symposium on Computers and Communications (ISCC)*, Natal, Brazil, 2018, pp. 14-19.

[10] W. Zhou, X. Han, Xu. Y, R. Chen, Z. Zhang, Embryo Evaluation Based on ResNet with AdaptiveGA-optimized Hyperparameters, *Journal of Internet Technology (JIT)*, Vol. 23, No. 3, pp. 527-538, May, 2022.

[11] B. A. Tama, K. H. Rhee, Performance evaluation of intrusion detection system using classifier ensembles, *International Journal of Internet Protocol Technology (IJIPT)*, Vol. 10, No. 1, pp. 22-29, 2017.

[12] O. R. Sanchez, M. Repello, A. Carrega, R. Bolla, Evaluating ML-based DDoS Detection with Grid Search Hyperparameter Optimization, *Proceedings of the 2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*, Tokyo, Japan, 2021, pp. 402-408.

[13] Q. Ma, C. Sun, B. Cui, A Novel Model for Anomaly Detection in Network Traffic Based on Support Vector Machine and Clustering, *Security and Communication Networks*, Vol. 2021, pp. 1-12, November, 2021.

[14] M. Injadat, F. Salo, A. B. Nassif, A. Essex, A. Shami, Bayesian Optimization with Machine Learning Algorithms Towards Anomaly Detection, *Proceedings of the 2018 IEEE global communications conference (GLOBECOM)*, Abu Dhabi, United Arab Emirates, 2018, pp. 1-6.

[15] M. Aamir S. M. Ali Zaidi, Clustering based semi-supervised machine learning for DDoS attack classification, *Journal of King Saud University-Computer and Information Sciences*, Vol. 33, No. 4, pp. 436-446, May, 2021.

[16] M. N. Injadat, A. Moubayed, A. Shami, Detecting Botnet Attacks in IoT Environments: An Optimized Machine Learning Approach, *Proceedings of the 2020 32nd International Conference on Microelectronics (ICM)*, Aqaba, Jordan, 2020, pp. 2020-2023.

[17] M. Ahmad, Q. Riaz, M. Zeeshan, H. Tahir, S. A. Haider, M. S. Khan, Intrusion detection in internet of things using supervised machine learning based on application and transport layer features using UNSW-NB15 data-set, *EURASIP Journal on Wireless Communications and Networking*, Vol. 2021, No. 1, pp. 1-23, January, 2021.

[18] M. Idhammad, K. Afdel, M. Belouch, Semi-supervised machine learning approach for DDoS detection, *Applied*

Intelligence, Vol. 48, No. 10, pp. 3193-3208, October, 2018.
[19] N. Yadav, S. Pande, A. Khamparia, D. Gupta, Intrusion Detection System on IoT with 5G Network Using Deep Learning, *Wireless Communications and Mobile Computing*, Vol. 2022, pp. 1-13, March, 2022.
[20] J. Li, Z. Zhao, R. Li, Machine learning-based IDS for software-defined 5G network, *IET Networks*, Vol. 7, No. 2, pp. 53-60, March, 2018.
[21] H. A. Alamri, V. Thayananthan, J. Yazdani, Machine Learning for Securing SDN based 5G Network, *International Journal of Computer Applications*, Vol. 174, No. 14, pp. 9-16, January, 2021.
[22] Y.-E. Kim, Y.-S. Kim, H. Kim, Effective Feature Selection Methods to Detect IoT DDoS Attack in 5G Core Network, *Sensors*, Vol. 22, No. 10, Article No. 3819, May, 2022.
[23] 3GPP TS 23.288, *Architecture enhancements for 5G System (5GS) to support network data analytics services*, V. 17.0.0, March, 2021. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.288/23288-h00.zip [Online; accessed on July 17, 2022]
[24] 3GPP TS 23.501, *System architecture for the 5G System (5GS)*, V. 17.0.0, March, 2021. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-h00.zip [Online; accessed on July 17, 2022]
[25] S. Sevgican, M. Turan, K. Gokarslan, H. B. Yilmaz, T. Tugcu, Intelligent Network Data Analytics Function in 5G Cellular Networks using Machine Learning, *Journal of Communications and Networks*, Vol. 22, No. 3, pp. 269-280, June, 2020.
[26] Cisco, *Ultra Cloud Core 5G User Plane Function, Release 2020.03 – Configuration and Administration Guide: 5G-UPF Overview*, pp. 1-10, March, 2020. Available: https://www.cisco.com/c/en/us/td/docs/wireless/ucc/upf/2020-03/b_ucc-5g-upf-config-and-admin-guide_2020-03/b_UPF_chapter_011011.html [Online; accessed on July 17, 2022]
[27] 3GPP TS 23.060, *General Packet Radio Service (GPRS)*, V. 16.0.0, March, 2019. Available: https://www.3gpp.org/ftp/Specs/archive/23_series/23.060/23060-g00.zip [Online; accessed on July 17, 2022]
[28] A. K. Salkintzis, C. Fors; R. Pazhyannur, WLAN-GPRS integration for next-generation mobile data networks, *IEEE Wireless communications*, Vol. 9, No. 5, pp. 112-124, October, 2002.
[29] S. B. H. Said, M. R. Sama, K. Guillouard, L. Suciu, G. Simon, X. Lagrange, J. M. Bonnin, New control plane in 3GPP LTE/EPC architecture for on-demand connectivity service, *Proceedings of the 2013 IEEE 2nd International Conference on Cloud Networking (CloudNet)*, San Francisco, CA, USA, 2013, pp. 205-209.
[30] F5 Networks, *F5 and the Growing Role of GTP for Traffic Shaping, Network Slicing, IoT, and Security*, August, 2018. Available: https://www.f5.com/pdf/solution-guides/f5-and-the-growing-role-of-gtp-for-traffic-shaping-network-slicing-iot-and-security.pdf [Online; accessed on July 17, 2022]
[31] Ali Güngör, *UERANSIM*, V. 3.2.6, January, 2022. Available: https://github.com/aligungr/UERANSIM [Online; accessed on July 17, 2022]
[32] S. Kim, *Open5GS*, July, 2022. Available: https://github.com/open5gs/open5gs [Online; accessed on July 17, 2022]
[33] Y. Mirsky, *Kitsune Network Attack Dataset*, May, 2018. Available: https://github.com/open5gs/open5gs [Online; accessed on July 17, 2022]
[34] Y. Mirsky, T. Doitshman, Y. Elovici, A. Shabtai, *Kitsune: an ensemble of autoencoders for online network intrusion detection*, May, 2018, Available: https://arxiv.org/abs/1802.09089 [Online; accessed on July 17, 2022]
[35] J. Bergstra, Y. Bengio, Random search for hyper-parameter optimization, *Journal of machine learning research*, Vol. 13, pp. 281-305, February, 2012.
[36] L. Zahedi, F. G. Mohammadi, S. Rezapour, M. W. Ohland, M. H, Amini, *Search algorithms for automated hyper-parameter tuning*, April, 2021. Available: https://arxiv.org/abs/2104.14677 [Online; accessed on July 17, 2022]
[37] B. H. Shekar, G. Dagnew, Grid search-based hyperparameter tuning and classification of microarray cancer data, *Proceedings of the 2019 Second International Conference on Advanced Computational and Communication Paradigms (ICACCP)*, Gangtok, India, February, 2019, pp. 1-8.

Biographies

YeaSul Kim received the B.S. degree in Information Security Engineering from Sangmyung University in 2021. Her research interests include network security, 5G core, IoT security and security data analysis.

YeEun Kim received the B.S. degree in Information Security Engineering from Sangmyung University in 2021. Her research interests include 5G network security, information security, IoT security and security data analysis.

Hwankuk Kim received the B.S. and M.S. degrees in Computer Engineering from Korea Aerospace University in 1998 and 2000, and Ph.D degrees in Korea University in 2017. He is currently an assistant professor in the Sangmyung University. His research interests include 5G network security, software vulnerability analysis, IoT security.