IJITCE

ISSN 2347- 3657

International Journal of
Information Technology & Computer Engineering

Email : ijitce.editor@gmail.com or editor@ijitce.com

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 12, Issue 1, 2024

ROBUST NETWORK INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING WITH EARLY CLASSIFICATION

Dr. Martin Sahayaraj, Professor, Department Of ECE SICET, Hyderabad

Sreelekha Chikka, Bhargav Dasari, Suresh Dayyala, Sai Krishna Dharamoni

UG Student, Department Of ECE, SICET, Hyderabad

Abstract

Network Intrusion Detection Systems (NIDS) that use matching patterns have a serious weakness in that they cannot detect new attacks because they only learn existing patterns and use them to detect this challenge. To solve this problem, machine learning-based NIDS (ML-NIDS) detects anomalies by ML algorithms by analyzing the behavior of the process. However, ML-NIDS learns the characteristics of the attack based on training data, so it is sensitive to attacks that have not yet been trained, as well as a comparative model of machine learning. Therefore, in this study, we analyzed the characteristics of learning using agent properties, showing that the facility has access to the external network of learning material through ML-NIDS. To avoid this, early classification of sessions before they fall outside the detection range of the ML-NIDS training data can prevent ML-NIDS skips. Many experiments confirm that the application can detect the session early (before the session is terminated) and increase the power of existing ML-NIDS. Compared with existing methods, we hope that the proposed method will be used as a solution problem to solve the weaknesses and limitations of existing ML-NIDS, as it can provide a robust distribution and is used on the same data distribution.

1. Introduction

Fast and accurate detection of network intrusions is very important for the stable operation of the network. For this purpose, a security device called Network Intrusion Detection System (NIDS) has been proposed [1], [2]. The first NIDS creates patterns from existing attacks and detects intrusions very quickly and accurately by matching the patterns with the received text [3]-

ISSN 2347–3657

Volume 12, Issue 1, 2024

[5]. However, the disadvantage of methods based on existing attack models is that previously unknown attacks cannot be detected and networks can be easily infiltrated by changing the existing resistance. Use for NIDS [6], [7]. Technology

Senior Researcher Hayder Al-Hraishawi participated in the review of this article and approved its publication. An alternative measurement method addresses the shortcomings of the comparative model NIDS (PM-NIDS). ML-NIDS uses ML to identify the characteristics of existing network intrusions and use all the attributes to detect intrusions. Therefore, PM-NIDS can be easily accessed by changing the access pattern, while ML-NIDS can detect access even with some changes as long as the behavior is all the same. Therefore, it can be seen from the results of various studies that ML-NIDS provides more stable detection access than PM-NIDS. It is based on existing attack models such as PM-NIDS, and its detection ability depends on training.

Data set. In other words, like PM-NIDS, ML-NIDS can detect intrusions that are not present in the training data with a very low probability. However, there is little research on these limitations. Instead, many methods are being developed to avoid ML-NIDS by changing features at a specific location, in addition to the power of training data from artificial neural networks (GANs) and other deep learning methods. 21]—[23] one. However, these studies do not directly analyze the dependence of ML-NIDS on study materials and therefore there are limitations in understanding the characteristics of this dependence. Way to provide stability to ML-NIDS training data without increasing its size. The scheme analyzes the features of ML-NIDS training data and uses the discovered features to improve performance attainment without requiring major changes to the system. To this end, the approach proposed in this article expands the scope of reviewing educational materials by analyzing existing literature. Make some changes to the behavior by adding some packages to check the current interference. By analyzing the ML-

NIDS dataset, it was determined that the reliability of the training data is quite high, thus it has a similar weakness to PM-NIDS. It shows that the impact of success can be very different, especially depending on ML algorithms. The NIDS method can best detect intrusions. In this way, even short or long segments that cannot be detected by existing ML-NIDS can be detected with high accuracy. Especially when compared to existing PM-NIDS, early attack detection can be made on similar hardware devices, thus helping to maintain network security. The proposed method is effective because the ML-NIDS platform is not a high-cost, high-performance platform. Previous Work

Types of ML-NIDS are packet-based methods that use packet data directly as features and interactive methods that use data. Instead of packaging the group's statistics by features, they are called discussions. Packet methods can be divided into two types: one detection method uses one packet to check bad information in each received packet, and the other method uses multiple packets, the same packet stores object and connection information, which is the data packet. . Both single packet analysis and multi-packet analysis look for malicious codes or patterns in the data payload[10]. because of pressure

Accuracy of the pattern matching algorithm that can detect bad vehicles while maintaining a very low rate (FPR). However, attacks that use packet matching, such as Distributed Denial of Service (DDoS), are difficult to detect using packet-based methods, and incoming matching algorithms are easily bypassed by adding random data to the payload. Therefore, the modeling method cannot be used alone. When using the session function it is not possible to bypass NIDS by simply adding some dummy data. Moreover, the size of all features is always the same regardless of the packet size or the length of the session, so the session-based approach is more efficient as a packet-based method of managing traffic. Distribute received traffic. Various ML-NIDS have been developed so far and are expected to overcome the weaknesses of PM-NIDS. Inevitably, malicious actors develop different ways to bypass ML-NIDS (usually divided into white boxes, gray boxes, and black boxes) depending on what information is available. The white box technique is a way to bypass NIDS in cases where the attacker knows all the information about NIDS [19] – [23] . This is ideal but not accurate because information about the data, machine learning models, and methods used for learning must be available. The gray box method, like the extraction method, is a method in which the malicious user knows the minimum information [24], [25]. However, the black box approach finds a way to bypass NIDS without prior knowledge [26]. Therefore, although it is the most accurate, it is quite difficult to use due to the frequent collection of necessary data and the characteristics of NIDS are not directly determined. It has been shown that the accuracy of the classification model is mostly affected by the small number [19], [25]. Therefore, in the white box approach, common features can be easily found and NIDS can be easily bypassed by creating attacks by examining these features. Of course, there is research to reduce these weaknesses. Various methods have been proposed, such as removing some of the best effects and classification methods. However, removing some features is not a solution as it will affect the performance of the ML model. Finally, there is an urgent need to develop good machine learning models by reducing the dependence and sensitivity of features affecting the learning model while maintaining the accuracy of classification [26], [27]. NIDS does not need to build a model with training data, which can be learned before each session is received by the real network. Actually educational materials The size is limited, so the implications of some details in the discussion may be beyond the scope of work in the training materials. ML-NIDS cannot classify classes correctly if the corresponding features have a significant impact on the performance of the above learning model. Therefore, this is a problem that needs to be solved in order to create a system that will prevent attacks. However, research on this has not been done yet. Table 1 describes the advantages and disadvantages of each ML-NIDS, including recommendations

ISSN 2347–3657

Volume 12, Issue 1, 2024

TABLE 1. Strengths and weaknesses of each type of ML-NIDS.

Type	Strength	Weakness
Packet-based	- Fast detection. - Low false positive rate for existing intrusion	Vulnerable to attacks exploiting normal packets or adding random data to the payload.
Session-based	- High detection rate for attacks based on normal packets or randomly increased payload. - High scalability in terms of traffic rate and volume	Vulnerable to adversarial attack targeting some specific features
Proposed	Robustness against adversarial attack to any specific features.	Real-time monitoring overhead for each session.

THE PROPOSED APPROACH

We propose a new method to improve ML-
NIDS to control for interactions with positive extrapolations. Therefore, the ML model com bined with our proposed method can detect intrusions that exceed the distribution of traini ng data at a certain location with high probability, and therefore want to avoid, not only str engthening ML-NIDS, but also preventing the current gender. . Motivation
Since training data determines the performance of MLNIDS, it is very important to use trai ning data that is rich in network access and as nonduplicated as possible. However, since the size of the training data is limited, the area of the specific study area where the trainin g data is located is inevitably limited. To clarify this in more detail, it is necessary to analy ze the results when the training data range and the test data range do not overlap, especi ally in space. To explain in more detail, let's define some symbols as follows:
A session S is defined by S = {P1,P2, . . . ,Pk }, where it consists of k packets. Let us define src (Pi), size (Pi),rtime (Pi) by source IP of Pi, size of Pi, reception time ofPi, respectively. Then the forward packet count and the total data rate are defined by |S | and Σk size(Pi) , whereSforward = {Ph | Ph ∈ S, src (Ph) = src (P1)} .
In previous studies, the forward packet count and the totaldata rate are known to be very important features in the
ML-
NWS [19], [25]. Based on the amount of packets counted forward, this test created a traini ng dataset containing sessions with values

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 12, Issue 1, 2024

smaller than the threshold and a test dataset consisting of segments larger than the thres hold and ran experiments using these to evaluate the distribution. yield. Since the effect o f the number of letters sent may be different for each class, the following tests were condu cted for analysis. In the entire data set i. For the next data, only sessions with the value of the number of transmitted packets equal to or less than the threshold are selected (maxi mum number of transmitted packets, the value of the i-
th category, αi) are used to generate training data, and only forward data with values greater than Δi are used to generate testing data. Here αi sets the data distribution ratio o f the group to 7:3. For other classes, training and testing data are set regardless of the val ue of αi. αi is set to a value close to the 7:3 ratio because sufficient training and testing da ta sizes are required to obtain an accurate classification. That is, if αi is too large, the test data will be too small to evaluate the effectiveness of the classification. On the contrary, if αi is too small, the training data becomes too small compared to the training ML model, re sulting in reduced classification. Figure 1 shows the f1 scores of some selected classes b ased on comparison data. From the picture, we can see that the ratio should not be too s mall or too large depending on the category.

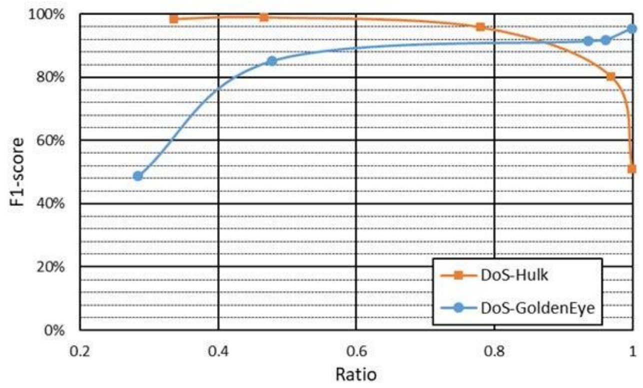


FIGURE 1. F1-scores of some selected classes according to the ratio of training dataset size to test dataset size when CIC-IDS 2017 dataset is used.

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 12, Issue 1, 2024

Tables 2 to 5 show experimental results using the ISCX2012 dataset to set training and te sting data for Brute Force-SSH, DDoS, DoS-HTTP, and intrusion. For Brute Force-SSH, when the ML model is trained using only forward packet count values less than αi (as shown in Table 2), the classifier cannot detect anything using the model. On the other hand, more than 98.5% is detected when training and testing data are set to Brute Force-SSH, as shown in Table 3 to Table 5. As for the value analysis of other groups (DDoS, DoS-HTTP and access), they are shown in Table 3 and Table 4:

TABLE 2. Confusion matrix where the training dataset and test dataset, respectively, are composed of sessions with small forward packet count values and sessions with large forward packet count values for Brute Force-SSH, whereas training and test datasets for other classes are randomly composed. Columns and rows of the matrix represent instances of actual and predicted classes, respectively.

	Brute Force-SSH	DDoS	DoS-HTTP	Infiltration	Normal
Brute Force-SSH	0	0	0	0	3
DDoS	0	41,290	2	0	1,890
DoS-HTTP	0	0	1,097	33	4
Infiltration	0	0	160	2,313	5
Normal	3,010	3,514	49	93	185,219

TABLE 3. Confusion matrix where the training dataset and test dataset, respectively, are composed of sessions with small forward packet count values and sessions with large forward packet count values for DDoS, whereas training and test datasets for other classes are randomly composed. Columns and rows of the matrix represent instances of actual and predicted classes, respectively.

	Brute Force-SSH	DDoS	DoS-HTTP	Infiltration	Normal
Brute Force-SSH	2,966	0	0	0	1
DDoS	0	6	3	0	2,094
DoS-HTTP	0	0	1,104	30	7
Infiltration	0	0	164	2,316	5
Normal	44	44,798	37	93	185,014

TABLE 4. Confusion matrix where the training dataset and test dataset, respectively, are composed of sessions with small forward packet count values and sessions with large forward packet count values for DoS-HTTP, whereas training and test datasets for other classes are randomly composed. Columns and rows of the matrix represent instances of actual and predicted classes, respectively.

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 12, Issue 1, 2024

	Brute Force-SSH	DDoS	DoS-HTTP	Infiltration	Normal
Brute Force-SSH	2,971	0	0	0	2
DDoS	0	41,039	6	0	1,616
DoS-HTTP	0	0	68	48	7
Infiltration	0	0	34	2,308	4
Normal	39	3,765	1,200	83	185,492

TABLE 5. Confusion matrix where the training dataset and test dataset, respectively, are composed of sessions with small forward packet count values and sessions with large forward packet count values for the Infiltration class, whereas training and test datasets for other classes are randomly composed. Columns and rows of the matrix represent instances of actual and predicted classes, respectively.

	Brute Force-SSH	DDoS	DoS-HTTP	Infiltration	Normal
Brute Force-SSH	2,972	0	0	22	1
DDoS	0	40,844	4	0	1,652
DoS-HTTP	0	0	1,087	22	8
Infiltration	0	0	166	74	9
Normal	38	3,960	51	2,321	185,451

Only 0.01%, 5.2%, and 3% were detected for Grade 5 and Grade 5, respectively, and all groups showed similar results. Finally, it was confirmed that the classification was greatly affecte d when the range of packet count values

set in the training data differed from the packet count forward values

set in the test scenario. The number of sessions exceeding the number of packets sent in the tra ining materials is almost invisible, regardless of the course type. In a real network, the number of referrals will only increase as the attack continues. That is, it is easy (compared to other feature s) to make the training data pass the next number of packets, but the impact of the current ML-NIDS is very high. Write a variety of messages, from small packets to large messages. However, this method not only makes the training dataset very large, but also makes it difficult to obtain s ufficient training data without significant incompleteness since many of the references are signifi cant. In addition, when the size of the training data increases, the training time will increase due t o the size of the data, and as the complexity of the model increases, the detection speed will dec rease. Therefore, increasing the forward packet size by increasing the size of the training data is not a solution. Finally, malicious users can neutralize existing ML-NIDS through attacks that increase the number of redirects and easily bypass detection regardle ss of the dataset.

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 12, Issue 1, 2024

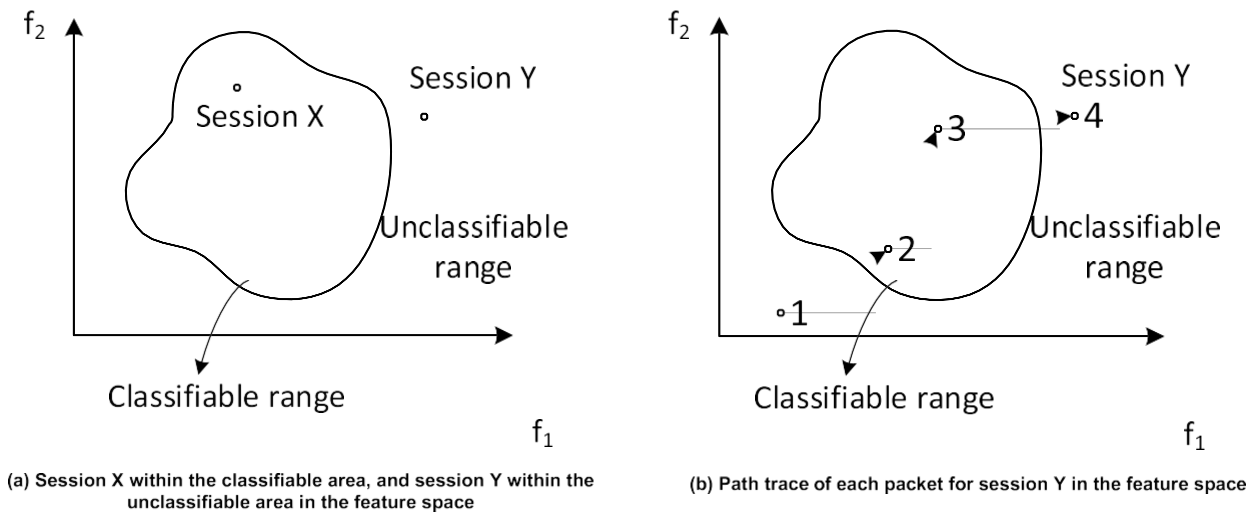


FIGURE 2. Classifiable and unclassifiable regions of sessions in the feature space.

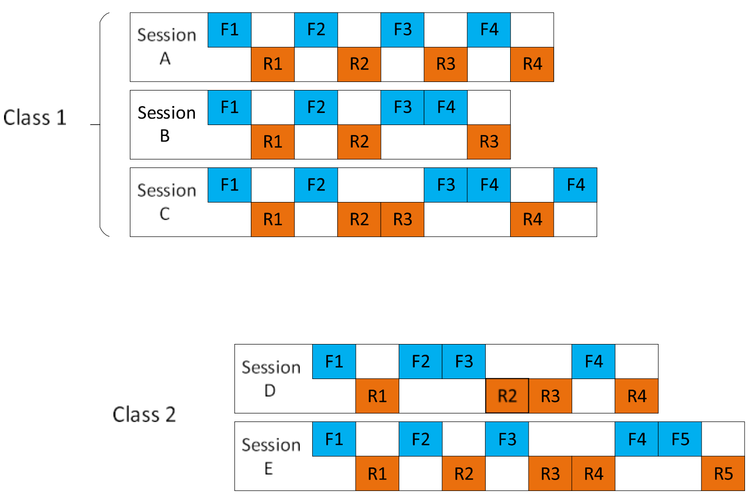
size, but there is no good way to prevent this. Figure 2 is a diagram showing two ways an ML model can and cannot classify sessions when the model is trained on data containing two elements. According to Figure 2(a), session X can be classified, so that the ML classifier can determine whether it is an interference or a positive session. On the other hand, since session Y is located in the unclassified area, it is not possible to classify it using the ML classifier. We also assume that every time a packet is received, NIDS will use the current received packet to establish a link between the original packet and organize it at a specific location, as shown in Figure 2(b). The path numbers shown in Figure 2(b) represent the packages used to create the features. For example, 2 in Figure 2(b) represents session features created using the first and second words. in the feature space, while features take the second and third packages in the classifiable range. Therefore, if we find the right time to detect the right session, we can divide the session correctly before the session ends, instead of dividing it during the session. Now let's discuss in detail how to use this idea. Proposed Algorithm

The algorithm should classify the session even before the session is terminated when the middle session features are in the classifiable area. However,

it is difficult to determine whether a session is running in a field where the name of the current se

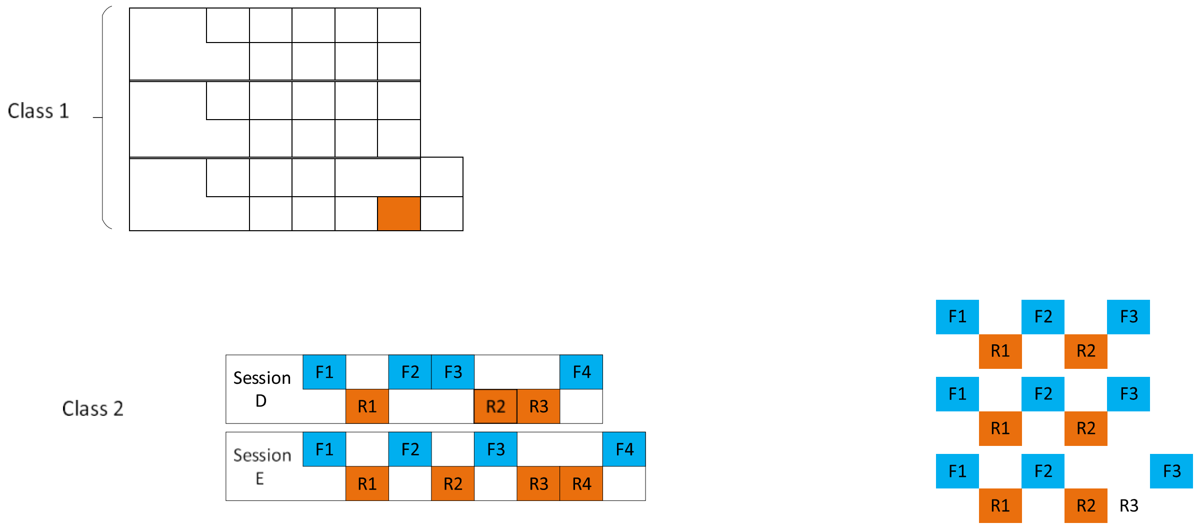
International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 12, Issue 1, 2024

ssion is not available. Of course, if you create intermediate sessions and use them for classification of all received packets, you can split the session even before the session is terminated, when the intermediate session is available in a separate area in a special location. choose. However, this would require very high cost and memory; This means it requires a very expensive, high-end platform that far exceeds the performance of current NIDS. Therefore, it is not possible to specify the distribution cost of each package received. In a particular case, the range of outcomes that can be effectively distributed to each group is determined in advance, and the distribution is attempted only if the middle part of the discussion is now included in the range. Here, the amount of features of each group is determined because the amount of information showing its relationship with the features of each group will be different. In this case, it is useful to choose certain types that are easy to calculate and have a greater impact on the distribution of the right people. In this article, we choose the package that will be considered as the decision because it meets all the conditions. Similarly, when the session ends, create a discussion session for deployment. However, this classification system differs from existing classifications in the following aspects. So when there are N total groups, we will have at most N - 1 values due to replication. If the forward count of the currently received packet matches one of these values, the session intermediate signature of the session to which the packet belongs is created and allocated. This time, if the classification is a class, here



(a) Original dataset

International Journal of Information Technology & Computer Engineering
ISSN 2347–3657
Volume 12, Issue 1, 2024



(b) Modified dataset for early classification when Θ_1 and Θ_2 are set to 3 and 4.

FIGURE 3. The comparison original and modified datasets with five sessions belonging to two classes where F_k and R_k stand for the k-th forward and backward packets.

If the maximum number of transmits is the same as the current number of receives, the p acket will be processed according to classification. For example, the package can be canceled a nd the results can be recorded or reported to the manager. Figure 3 shows how to obtain trainin g data from original data using pre-

computed αi. For the training data, each session of class i is normalized by αi. This type of static data goes a long way in avoiding dispersion in inequality between groups. Optimize each catego ry to be shared. Here, since the input group will be classified as benign, if the result of each clas sification is negative, the result is not taken into account. The packet is treated as benign only if t he session completes and is classified as benign. Details of this method are included in Algorith m 1. N + 1) = O(N), where N is the sequence number. In general

the number of classes is less than the length of the session. This means that the algorithm is les s complex than the packet inspection process. As shown in the figure, allocation is only possible when α(Ci) and the forwarding address are the same, thus reducing the total allocation overhead and increasing the probability of an attempt to complete the allocation before the outgoing pack et becomes too large. idea by doing this

Algorithm 1 Proposed NIDS Classification

Input: $C = \{C_1, C_2, \ldots, C_N\}$ where N is the total number of classes, and C_1 denotes a benign class.

$\vartheta(C_i)$: the maximum packet count value for C_i in the training dataset. i.e., ϑ_i. ① $= \{\vartheta(C_2), \vartheta(C_3), \ldots, \vartheta(C_N)\}$.

P: the current received packet.

n(P): the maximum forward packet count of P in the session.

F(P): the intermediate session features created from the first to P packets.

Output: Class ID if found

None, otherwise

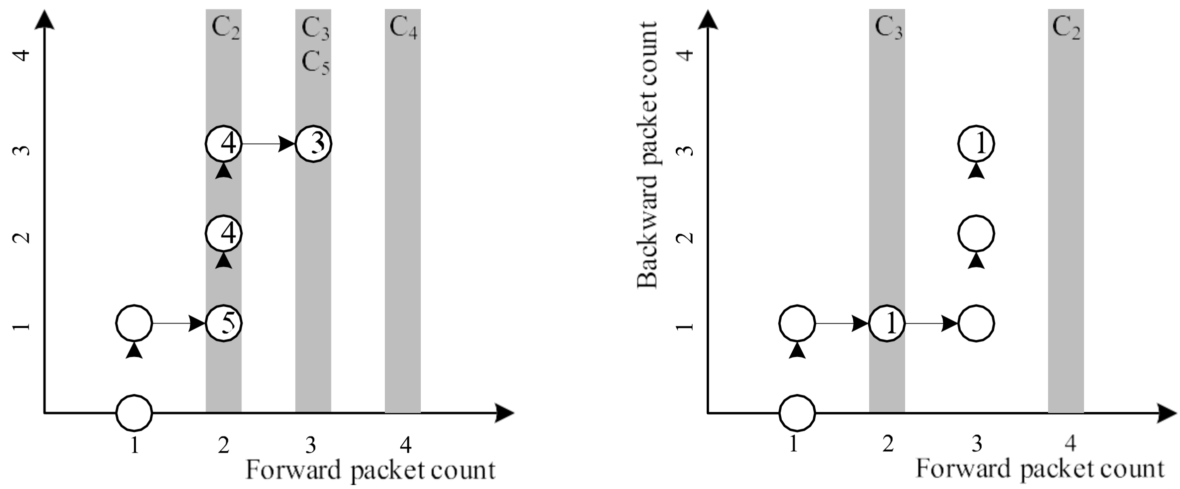
IF P is the last packet of the session THEN
    C_{est} = classifier(F(P))
    Return C_{est}
ELSE
    IF n(P) ∈ ① THEN
        C_{est} = classifier(F(P))
        IF $\vartheta(C_{est}) == n(P)$ THEN
            Return C_{est}
        ELSE
            Postpone the decision until the next packet is received.
        ENDIF
    ENDIF

This method increases both classification speed and classification accuracy. It counts the maximum number of submissions for each category and uses this number to create a training se t. It then attempts to analyze the received packet to determine the session class to which the pac ket belongs.

International Journal of Information Technology & Computer Engineering

ISSN 2347–3657

Volume 12, Issue 1, 2024



(a) Case 1: When C_i matching the classification result for the forward packet count: lines 7-8 of Algorithm 1, where $\theta(C_2)=2$, $\theta(C_3)=3$, $\theta(C_4)=4$, and $\theta(C_5)=3$.

(b) Case 2: When the session finished before find C_i matching the classification result for the forward packet count: lines 1-3 of Algorithm 1, where $\theta(C_2)=4$ and $\theta(C_3)=2$.

FIGURE 4. Two cases that the proposed algorithm classifies an incoming. Each circle represents each packet of the session. The empty and numbered circles denote no classification and classification result, i.e., class ID, respectivel

PERFORMANCE EVALUATION

To analyze and analyze the proposed system, a lot of data and many classification algorithms are used to analyze its performance in different areas. Six algorithms were select ed for evaluation: Random Forest [28], Adaboost Decision Tree [29], XGBoost [30], Extreme Learning Machine (ELM) [31], Deep Neural Network (DNN) [32], and Convolutional Neural N etwork (CNN). [17]. By comparing from deep learning to decision trees, we compare how the proposed method affects performance when applied to various algorithms. Environmental as sessment

It is important to use a lot of data because features in the same category may differ dependin g on the network environment in which the data was obtained in writing. Three databases we re used in this experiment: ISCX2012, CIC-IDS2017 and CSE-CIC-IDS2018 [33], [34]. Small classes are not included here. Additionally, since there is no need t o use the plan, only one group value is removed from the group count in the class. For exam ple, in PortScan in CIC-IDS2017, there is no need to use the recommended method, since the session with a single packet sent constitutes 99.5% of all data. For the same reason, FTP-Brute Force and DoS-SlowHTTPTest, which only have the sending value of the number of packets, are not include d in CSE-CIC-IDS2018. The total scores of ISCX2012, CIC-IDS2017 and CSE-CIC-IDS2018 are 6, 9 and 8 respectively. A test case involving many calculations. For this purpos e the section depends on:

TABLE 6. The ISCX2012 dataset.

Average detection length using random forest with the CSE-CIC-IDS2018 dataset.

	Average session length	Average detection length (packets)	Average detection length (sessions)	Average total detection length
Benign	6.9	5.5	6.9	6.9
Bot	53.0	21.0	73.0	26.5
Brute Force-SSH	22.4	19.0	22.1	19.2
Brute Force-WEB	151.2	38.0	0.0	38.0
Brute Force-XSS	202.7	78.5	0.0	78.5
DoS-GoldenEye	5.7	4.0	6.2	5.1
DoS-Hulk	3.2	2.0	3.7	2.5
DoS-Slowloris	14.4	4.0	14.9	4.5
Total average	9.3	11.2	6.9	8.3

TABLE 14. The average number of classifications for each class from using the ISCX2012 dataset.

Class	Benign	Brute Force-SSH	DDoS	HTTPDoS	Infiltration	Total
C/F number	1.33	2.65	3	1.1	2.24	1.58

TABLE 15. The average number of classifications for each class from using the CIC-IDS2017 dataset.

Class	Benign	Bot	DDoS	DoS GoldenEye	DoS Hulk	DoS Slowhttptest	DoS Slowloris	FTP-Patator	SSH-Patator	Total
C/F number	1.47	5	1.04	2.62	1.03	3.97	3	2.32	4.01	1.29

TABLE 16. The average number of classifications for each class from using the CSE-CIC-IDS2018 dataset.

Class	Benign	Bot	Brute Force-SSH	Brute Force-WEB	Brute Force-XSS	DoS-GoldenEye	DoS-Hulk	DoS-Slowloris	Total
C/F number	2.76	4.13	3.15	5	5.9	2.51	1.34	2.04	2.57

Performance increase for DDoS, 33% performance increase for SSH-Patator. For example, detecting Brute Force-WEB and Brute Force-XSS without using the scheme requires 151.2 and 202.7 packets respectively, when using the method only retrieving 38 and 78.5 packets increases the performance increase. 75% and 61% respectively. In particular, greater improvement in detection speed can be achieved with longer session durations. Long sessions often consume a lot of NIDS memory because NIDS needs all the data in each packet to create a unique packet after the session ends. Thus, the scheme can allocate such long segments before the session is terminated, thereby reducing the memory size of the session while improving detection. Complete

CONCLUSION

the total number of classifications to be identified separately. Now, before dividing the discussion, let's determine the number of groups according to the number of groups to be made. platform. Finally, the closer the distribution number is to 1, the higher the efficiency of using the proposed system on existing session-based NIDS hardware platforms. As shown in Tables 14 to 16, the average number of distributions for each data is very different for each category. This is because the total length of each class and the size of each class 'I¸i' are different. However, in Tables 14 to 16, the group mean does not exceed 3 for all data sets. This is not an increase based on the distribution. Therefore, even if the plan is implemented on existing hardware, it will not have a huge impact. Conclusion
The most important part in ML-NIDS is the training data used to build the classification model. However, it is not possible to obtain training data that includes all network interactions that occur in nature. Instead, the key is to find ways to take advantage.

Data available even if it has insufficient input data. This article offers a new approach to this problem. Using various datasets, the proposed method is shown to improve the weaknesses of existing ML-NIDS. Of course, the plan still has a lot of room for improvement. For example, simply using the next packet count may not be sufficient to determine whether the study has been overextended. However, if more tasks are considered, the number of sessions that can be completed per secon d decreases. Moreover, for some groups, the improvement in price determination is not very larg e. Although there is no such disadvantage, the ability to expand the distribution in a particular ar ea using data with limited information is better. Additionally, the deployment speed can also be i mproved, so the scheme is expected to be useful in maintaining the security of large networks w hen installed on Truly NIDS equipment. In our future work, we will focus on how we can sustain current results to support more employment. If the solution is found, ML-

REFERENCES

NIDS can complete the classification to find the value without reducing the classification speed. Analysis of Intrusion Detection Systems (IDS) and Internal Intrusion Detection and Protection Sy stems (IDPS), Proc. internationalization. Meeting. Creative calculation. to show. (ICICI), Novemb er 2017, p. 949~953, doi: 10.1109/ICICI.2017.8365277. internationalization. Meeting. network. n umber. Sociology Association, May 2010, p. 194-196, doi: 10.1109/ICNDS.2010. 5479341.

Second International. Meeting. to count. resources. development. 196-
199, doi: 10.1109/ICCRD.2010.25.
[4] H.Zhang, ——
Intrusion detection system design based on new pattern matching algorithm "Proc". internationali zation. Meeting. to count. engineer. Technology, January 2009,
p. 545–
548, doi: 10.1109/ICCET.2009.244. Singh and V. K. Bhalla, “Pattern matching algorithms for intr usion detection and prevention systems: A comparative analysis,” Proc. internationalization. Mee ting. envelope. Computers, communications. He taught. (ICACCI), September. Year 2014, p.17. 50–
54, doi: 10.1109/ICACCI.2014.6968595. Peb International. Meeting. Trend Electronics. to show. (ICOEI), April 2019, p. 916–920, doi: 10.1109/ICOEI.2019.8862784.
[7] A. Phadke, M. Kulkarni, P. Bhawalkar ve R . Bhattad, “A review of machine learning techniqu es for network intrusion detection,” in Proc. We are International. Meeting. to count. path. comm unication. (ICCMC), March 2019,

ISSN 2347–3657

Volume 12, Issue 1, 2024

p. 272-275, doi: 10.1109/ICCMC.2019.8819748. A. Marotta, M. Kist, L.R. Faganello, C. B. Ob, J. Rochol and L. Z. Granville, “Kitsune: a radio-
based radio control system based on spectrum sensing,” Proc. IEEE Network. Management.Sy mptoms. (NOMS), May 2014, p. 1–
9, doi: 10.1109/NOMS.2014.6838316. Gaddam and M. Nandhini "Analysis of various Snort-
based application code that refactors Snort tools in the Kali Linux environment to detect and pre vent network intrusions", Proc. internationalization. Meeting. Creative communication. to count. machine. (ICICCT), March 2017, p. 10-
15, doi: 10.1109/ICICCT.2017.7975177. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang an d M. Zhu, “HAST-
IDS: Learning hierarchical spatiotemporal features using deep neural networks to improve acces s detection,” IEEE Access, vol. 6,
Page. 1792-
1806, 2018. Marteau and N. Bechet, “Input detection of system coupling between supervised an d unsupervised systems: A study on the ISCX dataset,” Proc. First International. Meeting. Intellig ence data. Safety. (ICDIS), April 2018, p. 219-226. Huo and D. Hogrefe

layer multiclass detection method for network access,” Proc. IEEE symptoms. to count. communi cation. (ISCC), July. 2017, s. 101-1 767-
772. J. Iqbal and A. Raheem, “Comparison of support vector machine, random forest, and machi ne learning climate for access control,” IEEE Access, vol. 6, p. 33789–
33795, 2018. Cheong, K. Park, H. Kim, J. Kim, and S. Hyun, “Machine learning-
based intrusion detection system for class imbalanced datasets,” J. South Korean Research Insti tute Information.Security. Cryptozoology, vol. 27. No. 6, p. 1385-
1395, December 2017. Fei and X. He, “A deep learning approach for intrusion detection using n eural networks,” IEEE Access, vol. 5, one. 21954–21961, 2017. Song and Y.

G. Cheong, “Distributed attack types of intrusion detection systems using machine learning algor ithms,” Proc. IEEE 4th International Conference Proceedings. Big data counting. Application Ser vices (BigDataService), March 2018, pages 282–286. Lin, H.-C. Li, P. Wang, B.-H. Wu, J.-

Y. Tsai, “Network intrusion detection of cyber and civ convolutional neural networks,” hauv Proc. IEEE International. Meeting. application form. system. Invention (ICASI), April 2018, p. 1107–1110.

Y. Otoum, D. Liu, and A. Nayak, “DL-IDS: Deep learning-based intrusion detection framework for securing the Internet of Things,” ¤ Trans. tshwm. communication. Technology, p. e3803, 2019, doi: 10.1002/ett.3803, 2019, doi: 10.1002/ett.3803. Page 6. 38367–38384, 2018. Lao, “Combining deep learning attacks for cybersecurity,” 2019, arXiv: 1903.11688. Exploring the anti-revolutionary nature of deep learning”, Proc. Meeting. (GLOBECOM), December 2019, p. 1-6.

A. Piplai, S. S. L. Chukkapalli and A. Joshi, “Attack! In Proc, adversarial attack bypassing GAN-based classifiers trained to detect network interference. IEEE 6th International Conference Meeting. Big data security. Cloud (Big Data Security) International. Meeting. high performance. Intelligent Computing, (HPSC) IEEE Int. Meeting. Intel. Information security. (IDS), May 2020, p. 49-54: İ.

Han, Z.W., Zhong, W.W. Yang, S. Lu, X. Shi, and X. Yin, “Evaluation and improvement of machine learning-based network access anti-malware system,” IEEE J. Sel. Regional House of Commons, Vol. 39, no. 8 p.m. 2632–2647, August 2021. Lin, Y. Shi and Z. Xu, IDSGAN: Generative Adversarial Networks for Attack Generation for Intrusion Detection, 2018, arXiv:1809.02077. Shi, “Strategy against DoS intrusion detection: An improved boundary-based approach,” Proc. IEEE 31st International Conference Proceedings. Artif. Intel. (BTK), Kasım 2019, p. 1288-1295 lb. Marchetti, “Measuring the effectiveness of attacks against botnet detectors,” Proc. IEEE 18th International Symp Conference. network. to count. application form. (NCA), September 2019,

p. 1~8.
M.J. Hashemi and E. Keller, "Development of a robust anti-
aliasing algorithm in network access," Proc. IEEE Conference Network. running virtualization software Identify the network. (NFV-SDN), November 2020, page 17. 37€ 43, doi: 10.1109/NFV-SDN50289.2020.9289869. Liaw, C. Tong, J. Culberson, R. Sheridan, and B. Feuston, "Random forests: A classification and regression tool for composite classification and QSAR modeling," J. Medicine. Calculation. Science, vol. 43, one. 1947-
1958, November 2003. EPJ Web Conference, vol. We 55, 2013, p. 02004.

T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," Proc 02004. 22nd ACM SIGKDD International. Kev sib ntsib. To know. Discovery Data Mining, San Francisco, CA, USA, August 2016, p. 785-
794. Huang, Q. Y. Zhu, and C. K. Siew, "Advanced machine learning: Theory and applications," Neural Computing, vol. 70, p. 489–501, December 2006. Trend Mach. learn 2, p. 1-
127, January 2007. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Work on the development of a method for generating data entry analysis in databases," Comput. Security, vol. 31.No. Sayfa 3. 357âǨ 374, May 2012, doi: 10.1016/j.cose.2011.12.012. [34] İ. Sharafaldin, A. Habibi Lashkari and A. A. Ghorbani, "New access information and development of signature access," Proc. Fourth International. Meeting. information. system. Safety. Secret, 2018, p. 108–
116, doi: 10.5220/0006639801080116. Aceto, D. Ciuonzo, V. Persico, and A. Pescape, "Hierarchical hybrid intrusion detection approach in IoT scenarios," Proc. IEEE Global Communications. Conference (GLOBECOM), December 2020, p. 1â7.
[36] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Using deep learning for mobile traffic classification: Experimental results, lessons learned, and challenges," IEEE Trans. network. Service Management, Vol. 16. No. 2,
Page. 445–
458 February 2019. and M.S. degree in electrical engineering and a Ph.D. degrees in electrical engineering and computer science from Seoul National University in 1999, 2001, and 2009, respectively. In 2010, Jangwee joined the National Defense Research Institute as a researcher, and in 2013, he joined Keimyung University in Daegu, South Korea. He is currently a professor at

Yeongnam University in Gyeongsan, South Korea. His current research interests include collabo ration

n and security, blockchain, machine learning-based user analytics, and network security for high-speed networks. He received his BSc degree in Information and Communication Engineering fro m Lingnan University in 2018 and is currently pursuing his master's degree at the same universit y. degree. His current research interests include machine learning-based intrusion detection and prevention in high-speed networks.