I. J. Computer Network and Information Security, **2024, 4, 17-30**
Published Online on August 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2024.04.02

Cascaded Machine Learning Approach with Data Augmentation for Intrusion Detection System

Argha Chandra Dhar
Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna-9203, Bangladesh
E-mail: dhar1707069@stud.kuet.ac.bd
ORCID iD: https://orcid.org/0000-0002-4650-8075

Arna Roy
Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna-9203, Bangladesh
E-mail: roy1707018@stud.kuet.ac.bd
ORCID iD: https://orcid.org/0000-0003-4859-6133

M. A. H. Akhand*
Department of Computer Science and Engineering, Khulna University of Engineering & Technology, Khulna-9203, Bangladesh
E-mail: akhand@cse.kuet.ac.bd
ORCID iD: https://orcid.org/0000-0001-5465-8519
*Corresponding author

Md. Abdus Samad Kamal
Graduate School of Science and Technology, Gunma University, Kiryu 376-8515, Japan
E-mail: maskamal@gunma-u.ac.jp
ORCID iD: https://orcid.org/0000-0003-3150-0510

Kou Yamada
Graduate School of Science and Technology, Gunma University, Kiryu 376-8515, Japan
E-mail: yamada@gunma-u.ac.jp
ORCID iD: https://orcid.org/0000-0001-5502-2264

Received: 06 February 2023; Revised: 20 April 2023; Accepted: 25 May 2023; Published: 08 August 2024

Abstract: Cybersecurity has received significant attention globally, with the ever-continuing expansion of internet usage, due to growing trends and adverse impacts of cybercrimes, which include disrupting businesses, corrupting or altering sensitive data, stealing or exposing information, and illegally accessing a computer network. As a popular way, different kinds of firewalls, antivirus systems, and Intrusion Detection Systems (IDS) have been introduced to protect a network from such attacks. Recently, Machine Learning (ML), including Deep Learning (DL) based autonomous systems, have been state-of-the-art in cyber security, along with their drastic growth and superior performance. This study aims to develop a novel IDS system that gives more attention to classifying attack cases correctly and categorizes attacks into subclass levels by proposing a two-step process with a cascaded framework. The proposed framework recognizes the attacks using one ML model and classifies them into subclass levels using the other ML model in successive operations. The most challenging part is to train both models with unbalanced cases of attacks and non-attacks in the datasets, which is overcome by proposing a data augmentation technique. Precisely, limited attack samples of the dataset are augmented in the training set to learn the attack cases properly. Finally, the proposed framework is implemented with NN, the most popular ML model, and evaluated with the NSL-KDD dataset by conducting a rigorous analysis of each subclass emphasizing the major attack class. The proficiency of the proposed cascaded approach with data augmentation is compared with the other three models: the cascaded model without data augmentation and the standard single NN model with and without the data augmentation technique. Experimental results on the NSL-KDD dataset have revealed the proposed method as an effective IDS system and outperformed existing state-of-the-art ML models.

Index Terms: Cascaded Framework, Classification, Data Augmentation, Intrusion Detection System, Neural Network.

1. Introduction

Cyber-attack is an unwanted attempt to steal, hide, expose, manipulate, and alter information through illegally accessing a computer network or the internet. It can lead to cyber-terrorism, cyber-war, or cyber threats [1]. The availability, low cost, and rapid advancement of the internet play a significant role in our lives. But this rapid development is also expanding the cyber-attack scopes due to the expanding trends of remote work, availability of sensitive data, insufficient and inefficient malicious software (malware), and intrusion detection system. Different types of cyber-attacks can happen, such as denial of service, a man-in-the-middle attack, phishing attack, cross-site scripting attack, malware, birthday attack, intruder attack, and so on [2]. Different methods are being introduced to stop and detect cyber-attacks as a remedy.

Cybersecurity has been a crucial research area over the last decade since different levels of hi-tech attacks are being employed by attackers. Cybersecurity has received significant attention globally, with the ever-continuing expansion of internet usage, due to growing trends and adverse impacts of cybercrimes. High-profile cybercrimes have revealed the ease of spreading international cyber-attacks, including disrupting businesses, corrupting sensitive data, stealing information, illegitimate access, etc. [3]. As a preventive measure against such attacks, different kinds of firewalls, antivirus systems, phishing detection [4], anomaly-based detection systems [5], and Intrusion Detection Systems (IDS) have been introduced throughout the last decade. The first IDS idea was proposed by Jim Anderson in 1980 [6]. Since then, the enormous evolution of technology has raised the need for proper IDS products.

Recently, with the drastic growth and superior performance, Machine learning (ML)-based autonomous systems have evolved state-of-the-art in cybersecurity. Diverse IDS models have been developed using several ML methods [7-17]. The IDS studies are intensely focused on achieving better accuracy with this standardized dataset, overlooking the fact of solving the discrepancies of this dataset. Most importantly, both attack and Normal cases were emphasized equally, which is not justified for a highly reliable IDS system development. It would be hazardous if any attack is treated as Normal, though the opposite (i.e., a normal case treated as an attack) is not harmful as only a few measures will apply to it. Therefore, the IDS might have more significant attention to classify attack cases correctly. Besides, the dataset holds subclass categorization of only four primary attack class samples, but none of the current studies considered the influence of attacks in the subclass levels.

This study aims to develop a novel IDS system that gives more attention to classifying attack cases correctly and categorizing attacks into subclass levels. A two-step process with a cascaded framework is proposed, where an ML model recognizes the attacks and another classifies them into subclass levels in successive operations. The most challenging part is to train both ML models with unbalanced cases of attacks and non-attacks in the datasets. Therefore, an appropriate augmentation technique is introduced for sufficiently preparing the training set to learn attack attributes using the limited attack samples of the original dataset. The proficiency of the proposed cascaded approach with data augmentation is compared with the standard NN. Furthermore, with and without the data augmentation technique, the performances of the proposed framework are compared. Evaluation results on the NSL-KDD dataset have demonstrated the proposed method as an effective IDS system. The remarkable contributions of the study are summarized as follows:

- An augmentation technique is proposed that has solved inconsistencies in the data distribution in the IDS dataset so that system learns attack issues properly;
- A cascaded ML framework model is introduced to give more considerable attention to classifying attacks into subclass levels; and
- The proposed framework is implemented with NN on the NSL-KDD dataset and rigorously analyzed, comparing it with related competitive methods and techniques.

The structure of the rest of the paper is as follows. Section 2 describes NSL-KDD and reviews IDS studies with the NSL-KDD dataset. Section 3 demonstrates the proposed Cascaded Neural Network based IDS. Section 4 presents experimental outcomes and performance comparisons with the existing studies. Finally, Section 5 concludes the paper with a few remarks.

2. Benchmark IDS Datasets and Related Studies

There are a few IDS-related datasets. The pioneer IDS dataset is KDD CUP 99, developed on the DARPA'98, and NSL-KDD is its updated version [18]. Numerous studies considered NSL-KDD as a benchmark IDS dataset, including [9,11,12,17,19] and so on. Several other studies performed on their proprietary datasets[20]. The CICIDS2017 dataset [21] comprises recent common attacks and benign traffic, and the CSE-CIC-DS2018 dataset [22] is a labeled network traffic dataset comprising a diverse range of network traffic. Over the past few years, both of these datasets have been utilized for the development of IDS. This study considered the NSL-KDD data due to its significant properties as well as uses in recent prominent studies. Existing studies indicate that the NSL-KDD dataset is well-suited for evaluating various intrusion detection models [23]. The following subsections describe the NSL-KDD dataset briefly and review central studies with the dataset.

2.1. NSL-KDD Dataset and Its Features

The NSL-KDD dataset is the most popular [18], containing different types of cyber-attacks with various attributes. There are 41 features in the NSL-KDD dataset. Among those features, four features are string type: Protocol_type, Class, Service, and Flag. The dataset is divided into training and test sets. Jointly, there is a total of 125973 samples, where 67343 samples are for Normal (i.e., non-attack) and the remaining 58,630 samples are for four different attacks: Denial of Service (DoS), Probing, Root to Local (R2L), and User to Root (U2R). DoS is an attack intended to make a cyber service or website unavailable to users. Probing is one kind of invasive attack that tries to find the weakness of the network. In R2L, the attacker gains local access by sending packets from a remote machine, which is mainly illegal [24]. U2R attacker illegally accesses the root's privilege and where at the same time, legally accesses the local machine. Samples of each class have also been categorized into subclasses.

The dataset is highly imbalanced in samples for the Normal and four different attack types; the imbalance issue is clearly observed from the distribution of samples according to classes in the Pie Chart of Fig. 1. ML-based IDS using the NSL-KDD dataset is a classification task having five categories. The Normal class alone holds around 52% of samples, and U2R holds only 119 samples (0.04%). In the case of the subclass level, the imbalance issue is more severe. Several subclasses contain only two samples. In the case of the IDS dataset, the imbalance matter is acceptable because Normal is an ordinary/default case having more presence, attack issues are noted as a regular matter, and some attack types may happen rarely. The imbalance matter has a significant concern in the case of ML. It is easy for the ML model to show acceptable performance, truly classifying large proportionate class samples, while the failure for classes having very few samples, such a system is not perfect for IDS as it fails to tackle the rare sample classes.

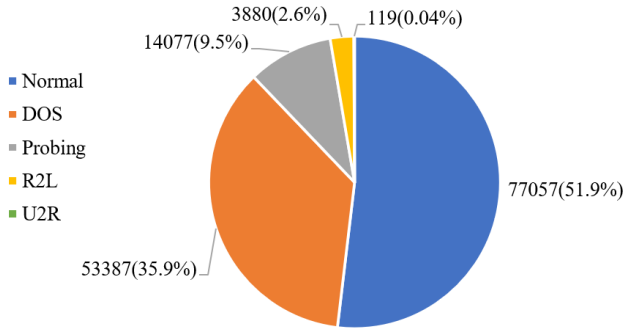

Fig.1. Distribution of samples according to classes in pie chart; samples are highly imbalanced

2.2. IDS Studies with NSL-KDD Dataset

Diverse IDS models have been developed in recent years using the NSL-KDD dataset using several ML and deep learning methods [11,17,19]. Extensive studies on the NSL-KDD dataset and evaluation using typical ML algorithms have been shown in [25]. The study mainly measured the effectiveness of the various classification algorithms in detecting anomalies in network traffic patterns. The automated data mining tool WEKA was used to conduct experiments using J48, SVM, and Naïve Bayes algorithms for classification. In [17], a self-organized map was used to develop IDS, where the K-means clustering algorithm is used to measure the accuracy. Effective classifier approach by combining different types of decision tree has been proposed in [26] to overcome the limitations of traditional IDS, such as high false positive rates and low detection rates.

Neural Network (NN) was considered in several IDS studies. The study of [11] considered NN, the Bayesian Net-GR (Net with Gain Ratio) technique, and the Bayesian feature selection technique. In [27], an ensemble of NNs was used to classify the different types of attacks in the dataset. The base classifiers of the ensemble are an autoencoder, a deep belief NN, a deep NN, and an extreme learning machine. In particular, the detection and false alarm rates of the implemented ensemble were evaluated. The performance of this dataset is also evaluated using ANN in [28]; the performance is measured for both binary class and five class classification (type of attacks).

Deep learning (DL) methods have been considered in recent studies. Convolution Neural Network (CNN) has been used for IDS development and found to be better than traditional ML methods (i.e., Random Forest and SVM) and other DL methods (i.e., Deep Belief Network and Long Short-Term Memory) [19]. The study in [12] proposed a hybrid IDS that combines the advantages of K-means clustering, RF classification, and DL techniques where the proposed system aims to address the limitations of traditional IDS, such as high false positive rates, low detection accuracy, and high computational complexity. A DL method with filter-based feature engineering for wireless IDS has been proposed in [10] where the authors used feed forward Deep Neural Network (DNN) with multiple hidden layers to learn the complex patterns and relationships between the selected features. The authors in [29] proposed a Deep Q-Learning based Reinforcement Learning (RL) approach by using RL to train a deep Q-network (DQN) to detect network intrusions.

3. Cascaded Neural Network based Intrusion Detection System

To achieve the goal of developing a novel IDS employing more attention to classifying attack cases, the challenges addressed in this study include properly handling highly imbalanced IDS datasets, increasing attention to classifying attack cases correctly, and extending the classification of attack cases up to the subclass level. The following subsection first proposes a proposed cascaded framework and then briefly describes individual components.

3.1. Cascaded Framework Overview

Figure 2 depicts the proposed cascaded framework as a whole. At first, available samples are divided into training and test sets. A training set was used to develop the model, and a test set was reserved to evaluate the performance of the developed system. Augmentation is performed on the training set samples, and the augmented data is used to develop ML models in the cascaded framework. The cascaded framework consists of two different stages: Stage 1 for the classification of samples into major classes and Stage 2 for the classification into the subclass levels. A single ML model is trained with the whole augmented dataset for Stage 1. For subclass classification in Stage 2, the augmented dataset was divided into five individual classes (i.e., Normal and four attack classes), and then four distinct ML models were trained with individual attack case samples for subclass classification. In the figure, ML_{Main} in Stage 1 is the classifier for the major class; and ML_{DoS}, ML_{Prob}, ML_{R2L}, and ML_{U2R} in Stage 2 are the classifiers for subclass classification of Denial of Service (DoS), Probing, Root to Local (R2L), and User to Root (U2R) attacks, respectively. The ML_{Main} classifier might have five output classes, and the number of classes for other classifiers depends on the number of subclasses in the individual attack cases. According to attack and subclass cases described in Table 1, the number of subclasses for DoS, Proving, R2L, and U2R are 11, 6, 15, and 7, respectively.

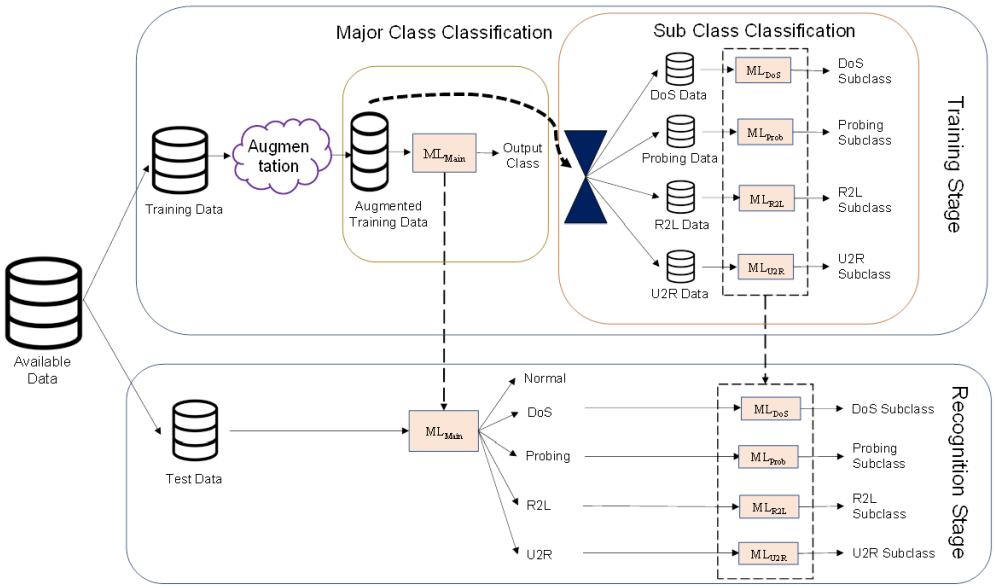

Fig.2. Proposed cascaded framework architecture

In the case of the testing phase, a test sample is passed through the ML_{Main} to classify into a major class, which might be Normal or an attack among four different categories. The attack-categorized sample is then used to measure its subclass level using the classier trained for the corresponding class. For example, if a test sample is classified as U2R, then ML_{U2R} is used to measure its class level. Notably, the misclassification in the proposed cascaded model may occur in two different forms: misclassification in the major class level in Stage 1 and misclassification in the subclass levels in Stage 2.

3.2. Data Preprocessing and Augmentation

The NSL-KDD dataset holds 41 features or attributes; among those three attributes, protocol_type, service, and flag are non-numeric (i.e., string) types. These three attributes are processed by using One Hot Encoding. For example, the feature protocol_type has three types of attributes, which are TCP, UDP, and ICMP; and its one-hot encoder values are binary vectors [1,0,0], [0,1,0], and [0,0,1]. Similarly, the feature service has 70 types of attributes, and the feature flag has 11 kinds of attributes. This way, 41-dimensional features are mapped into 122-dimensional features to use in the ML model. Table 1 presents the individual attributes' brief descriptions, types, and inputs after one-hot encoding.

After taking samples reserved for the test set, the training set is augmented, which is essential in the proposed model. Note that the augmentation aims to handle the biasing issues towards highly populated data and improves the overall accuracy throughout all the classes. Table 2 shows the separated samples for training and test sets and the augmented training samples for individual subclass levels. It is uncovered from the table that the number of examples in different attack classes varies widely. Firstly, the available data samples are divided into training and test sets to obtain the balanced

ones. For classes with large samples, around 20% are reserved for the test set. The remaining about 80% of the samples are used as the training set. For example, the Neptune subclass of DoS has a total of 45871 samples; the samples are divided into test and training sets 9175 and 36696, respectively. For such subclasses cases having large samples, augmentation is not necessary and is not performed.

Table 1. Attributes (i.e., Input Features) and one-hot encoding of NSL-KDD dataset

Att. No.	Attribute Name	Description	Att. Type	Number of Inputs after One-Hot Encoding
1	Flag	Status	String	11
2	Protocol_type	Protocol		3
3	Service	Destination network		70
4	Duration	Basic Feature	Numerical	1
5	Src_bytes	Transferred from source to destination		1
6	Dst_bytes	Transferred from destination to source		1
7	Land	Check source and destination port numbers		1
8	Wrong_fragment	Number of wrong fragments		1
9	Urgent	Number of urgent packets		1
10	Hot	Indicate contents		1
11	Num_failed_logins	Count of failed login attempts		1
12	Logged in	Login Status		1
13	Num_comp romised	Number of conditions		1
14	Root_shell	Check if root shell is obtained or not		1
15	Su_attempted	Check if Su_root shell is obtained or not		1
16	Num_root	Number of root accesses		1
17	Num_file_creations	Number of file creation operations		1
18	Num_shells	Number of shell prompts		1
19	Num_access_files	Number of operations on access control		1
20	Num_outbound_cmds	Number of out of bound commands		1
21	Is_hot_login	Check login		1
22	Is_guest_login	Check guest		1
23	Count	Number of connections to the same destination host		1
24	Srv_count	Number of connections to the same service		1
25	Serror_rate	Flag error rate		1
26	Srv_serror_rate	Flag error rate		1
27	Rerror_rate	Flag error rate		1
28	Srv_rerror_rate	Flag error rate		1
29	Same_srv_rate	Rate of connections		1
30	Diff_srv_rate	Rate of connections to a different server		1
31	Srv_diff_host_rate	Rate of connections to different destination		1
32	Dst_host_count	Number of connections of same destination IP		1
33	Dst_host_srv_count	Number of connections of same destination port		1
34	Dst_host_same_srv_rate	Rate of connections to same service		1
35	Dst_host_diff_srv_rate	Rate of connections to different service		1
36	Dst_host_same_src_port_rate	Rate of connections to same source port		1
37	Dst_host_srv_diff_host_rate	Rate of connections to different machines		1
38	Dst_host_serror_rate	Rate of connections having flags		1
39	Dst_host_srv_serror_rate	Rate of connections having flags		1
40	Dst_host_rerror_rate	Rate of connections having flags		1
41	Dst_host_srv_rerror_rate	Rate of connections having flags		1
Total Processed Inputs				**122**

On the other hand, classes having a few samples are divided training and test sets fairly. Several cases among 23 different types have a minimal number of examples; a few cases (e.g., spy) have only two samples. Such case samples are divided fairly into training and test sets. As an example, two spy samples are divided into training and testing. This uneven distribution results in bias and poor accuracy, which is handled by augmentation of the training set. After

augmentation, the minimum number of samples for a subclass becomes larger than 1000. The augmentation algorithm followed in this study is as follows:

train = no. of training example in subclass x
  if train<1000:
    ratio = $ceil(\frac{1500}{train})$
    for 1 to ratio:
      x += x

Table 2. Summary of data augmentation on NSL-KDD dataset

Major Class (Total Samples)	Sub Class	Available Samples	Reserved as Test Samples	Considered as Training Samples	Augmented Training Set	Comments on Augmentation
Normal (77057)	Normal	77057	15411	61646	61646	-
DoS (53387)	1. neptune	45871	9175	36696	36696	-
	2. teardrop	904	181	723	2169	Applied
	3. back	1315	263	1052	1052	-
	4. smurf	3311	663	2648	2648	-
	5. pod	242	49	193	1544	Applied
	6. land	25	5	20	1500	Applied
	7. mailbomb	293	59	234	1638	Applied
	8. processtable	685	137	548	1644	Applied
	9. udpstorm	2	1	1	1500	Applied
	10. apache2	737	148	589	1767	Applied
	11. worm	2	1	1	1500	Applied
Probing (14077)	1. portsweep	3088	618	2470	2470	-
	2. ipsweep	3740	748	2992	2992	-
	3. nmap	1566	314	1252	1252	-
	4. satan	4368	874	3494	3494	-
	5. mscan	996	200	796	1592	Applied
	6. saint	319	64	255	1530	Applied
R2L (3880)	1. spy	2	1	1	1500	Applied
	2. phf	6	2	4	1500	Applied
	3. warezclient	890	178	712	2136	Applied
	4. guess_passwd	1284	257	1027	2054	Applied
	5. multihop	25	5	20	1500	Applied
	6. ftp_write	11	3	8	1504	Applied
	7. imap	12	3	9	1503	Applied
	8. warezmaster	964	193	771	1542	Applied
	9. xlock	9	2	7	1505	Applied
	10. xsnoop	4	1	3	1500	Applied
	11. snmpguess	331	67	264	1584	Applied
	12. Snmpgetattack	178	36	142	1562	Applied
	13. httptunnel	133	27	106	1590	Applied
	14. sendmail	14	3	11	1507	Applied
	15. named	17	4	13	1508	Applied
U2R (119)	1. rootkit	23	5	18	1512	Applied
	2. perl	5	1	4	1500	Applied
	3. buffer_overflow	50	10	40	1520	Applied
	4. loadmodule	11	3	8	1504	Applied
	5. sqlattack	2	1	1	1500	Applied
	6. xterm	13	3	10	1500	Applied
	7. ps	15	3	12	1500	Applied
	Total:	**125973**	**29719**	**118798**	**123757**	

3.3. Neural Network (NN) Architecture in the Cascaded Framework

ML classifier is the basic computational unit in the proposed cascaded framework, as shown in Fig. 2. NN is well-known for the classification task, and NNs with several hidden layers are considered a classifier in the proposed cascaded framework. The NN architecture is different in Stage 1 and Stage 2 due to other class numbers to classify.

NN for Classification in Major Class: As per Fig. 2, after augmentation, the training data is used to train the major class classification model ML_{Main}, for using them in categorizing five major classes. Therefore, the NN for ML_{Main} will have five neurons in the output layer, where individual neurons will represent a particular class. The neurons in the input layer will be 122, where each processed feature will map to a specific input neuron. The NN has three hidden layers, with 50, 20, and 10 neurons in three layers, respectively.

NNs for Classification into Subclass Level: In the case of the subclass classification model, each augmented subclass training dataset is used to train four individual ML classifiers. Each one will ultimately categorize their respective attacks into subclass levels. The NNs for subclass classification are similar to the major class model ML_{Main} except for the output layer. The size of the output layer is based on the number of subclasses for a particular major class. For example, the ML_{DoS} subclass model will have 11 neurons to classify a DoS attack into one of 11 subclass attacks. Again, the number of output layer neurons of NNs for ML_{Prob}, ML_{R2L}, and ML_{U2R} are 6, 15, and 7, respectively.

3.4. Significance of the Proposed Model

The proposed cascaded model of this study is significantly different from the existing IDS studies in several directions, including management of inconsistencies of IDS data, emphasis on learning attack samples, and classifying attacks into subclass levels. The IDS dataset is highly imbalanced as attack samples are very few concerning the non-attack cases, as demonstrated in the NSL-KDD dataset. Any ML model with this highly imbalanced data might suffer from learning the attack classes properly due to the lack of samples. An innovative augmentation is employed in this study to manage such inconsistency, which is explained in Section 3.2. It is observed from Table 2 that a subclass having very few training samples (even one or two) is augmented in number in such a way that it is more than 1000 in the actual training of the ML model. Such augmentation gives intensive attention to attack issues and helps to properly learn individual attack classes and subclasses. At the same time, it will reduce the risk of attack misclassification as Normal, which is very important. Such attentions are not available in the existing studies; most of the studies only focused on the accuracy of available samples, which is not adequate for a highly imbalanced dataset like the NSL-KDD dataset.

Attack categorization into subclass levels through a cascaded framework is a significant contribution of this study. The NSL-KDD dataset holds subclass categorization of four major attack class samples, but no existing studies considered the influence of attacks in the subclass levels. The problem turns into a 40-class problem to classify a sample into subclass levels with a single ML model, as depicted in Table 2, where one for the Normal class and 39 (=11+6+15+7) for subclasses of four major categories. Such a 40-class problem is relatively complex for any ML model. The whole task is subdivided in the cascaded framework: classify into major classes first and then classify into subclass levels with ML/NN models specialized for individual classes. Such sub-division simplifies the task for ML models to perform better in classification. The proposed model's performance has been evaluated through rigorous experiments for both the cascaded and single ML models, which will be presented in the following section.

4. Experimental Studies

The proposed IDS is a cascaded framework with data augmentation and five NNs. To evaluate the performance of the proposed model, single NN with and without augmentation are also investigated. Finally, four different models are considered for proper evaluation of the techniques employed in the proposed model.

Model 1: A single NN is trained with the dataset without augmentation.

Model 2: A single NN is trained with augmented training data.

Model 3: The cascaded NN model without augmentation where the ML_{Main} model is trained for major class, and four others are trained for subclass attacks without augmentation.

Model 4: The cascaded NN model with augmentation where ML_{Main} is trained for five major classes and four others are trained for subclass attacks with augmented data. This is the proposed cascaded NN model.

The architecture of NNs for Model 4 is explained as the proposed model; the NN architectures of Model 3 are the same as Model 4. The NN architecture for Model 1 and Model 2 is the same; the neurons in the output layer are 40 to classify a sample into Normal as well as subclass levels with the single NN. All the models were trained with Adam optimizer with a learning rate of 0.001 for 500 epochs.

Python Programming Language and Anaconda IDE have been preferred to conduct the experiment. The PC in which the experiments were conducted had the following configuration: processor of 7th Generation Intel® Core™ i5-7400 CPU @ 3.50GHz, GPU of NVIDIA Ge-Force GTX 1070Ti, 8 GB.

4.1. Experimental Results and Analysis

This section presents experimental outcomes on the NSL-KDD dataset with four models. At first, Loss curves for

different models are shown. Insightful evaluation of model-wise misclassification is the main/crucial points of the study to realize the proposed approach. Finally, the overall performances of the models are analyzed and then the outcome of the proposed method are compared with existing methods.

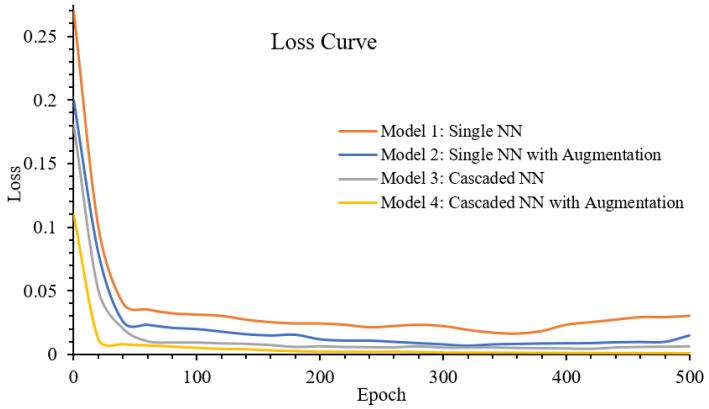

Fig.3. Loss curves of four different models with NNs considering data augmentation and cascaded manner

Figure 3 shows the loss curves of the implemented four different models with NNs considering data augmentation and cascaded manner. For simplicity, cascaded models' (i.e., Model 3 and Model 4) curves are for NNs of ML_{Main} in Stage 1. Therefore, the curves for cascaded cases are for major class classification while the curves for Model 1 and Model 2 are for subclass classification. It is visible from the figure that Model 1 hold highest amount of loss, then Model 2. On the other hand, it is clearly seen that Model 4 (i.e., the cascaded model with augmentation) has the lowest amount of loss and Model 3 also close to it. Lowest loss amount in cascaded cases is logical as it classifies only major five classes. At a glance, Model 4 is much efficient than others and its proficiency is explained in detail with the help of bar charts.

Figures 4-7 illustrate model-wise misclassification of test set attack samples as Normal, other major classes, and subclass level in their own class. Fig. 4 demonstrates misclassification for DoS class samples. It is clear from the figure that Model 1 generates the greatest number of misclassifications where it gets decreased in the subsequent three models, and Model 4 minimizes the misclassifications to the lowest. To clearly demonstrate the role of cascaded network and data augmentation in Model 4, let's take an example of the smurf subclass. The numbers of misclassification are 3, 2, and 1 for Model 1, Model 2, and Model 3, respectively. Model 1 makes some outer class misclassification, including Normal class, which is removed by using augmentation methodology in Model 2. Model 3 removes misclassified as Normal errors, which cannot be done by Model 1 and Model 2. But still, it makes some misclassification in a subclass. Finally, Model 4, with cascading and data augmentation, makes no error in classification, which manifests the effectiveness of the proposed model. For another example, the processtable subclass has a total of 18 misclassifications, which are solely incurred by misclassified as Normal. The most notable thing is that when Model 1, Model 2, and Model 3 cannot make any improvement, Model 4 decreases the errors from 18 to 13. Though Model 4 cannot lessen it to zero, it manages to decrease misclassified as Normal, which is one of the main objectives of our study. The effectiveness of the data augmentation is clearly revealed from the result presented in Fig. 4.

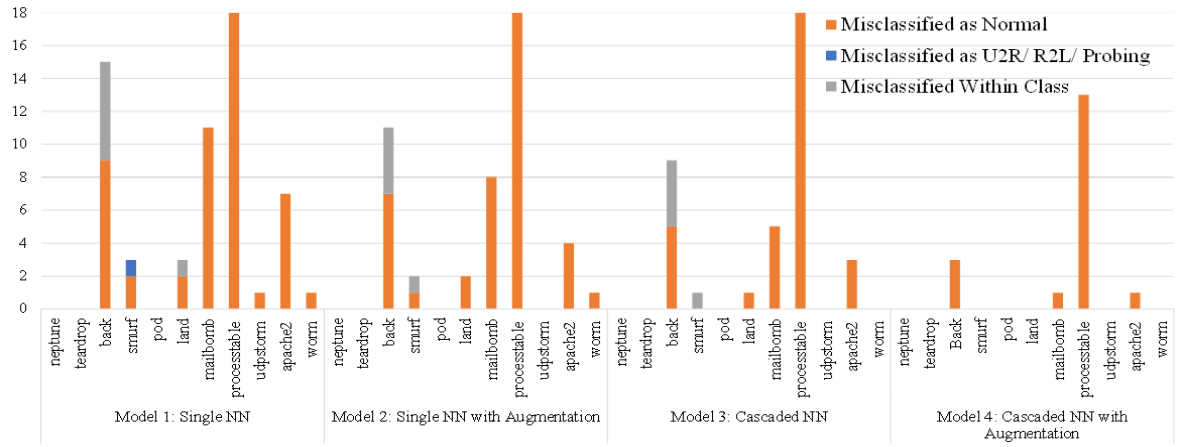

Fig.4. Misclassification of DoS class samples as normal, other classes and subclass level in own class

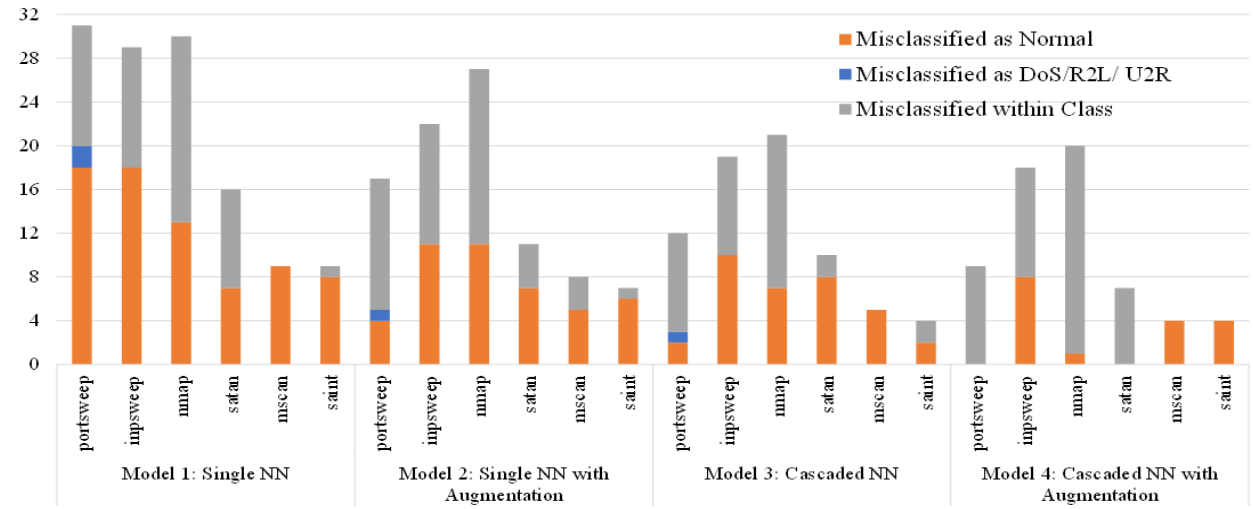

Fig.5. Misclassification of probing class samples as normal, other classes and subclass level in own class

Figure 5 demonstrates misclassification for Probing class samples for four different models into as Normal, other major classes, and subclass level within its own major class. The figure shows the outperformance of Model 4, although it has some misclassifications. Moreover, the figure shows some important findings. As an example, the portsweep subclass has all three different types of misclassifications in Model 1. Model 2 decreases the misclassification for Normal classes, but increases misclassified subclasses within its own Probing class. Augmentation in Model 2 tries to make data distribution even, but it may augment some of the samples, which is confusing to predict for the ML model. This might be the reason behind the remarkable improvement failure of augmentation alone. The cascaded technique in Model 3 decreases the total misclassification for the portsweep subclass further, but three different errors are still present there. Model 4 then removes misclassified as Normal and misclassified in other classes, which proves the effectiveness of the combination of cascading and augmentation once again. At a glance, Model 4 is the best, and Model 1 is the worst, considering the misclassification of all the test set samples for the Probing class.

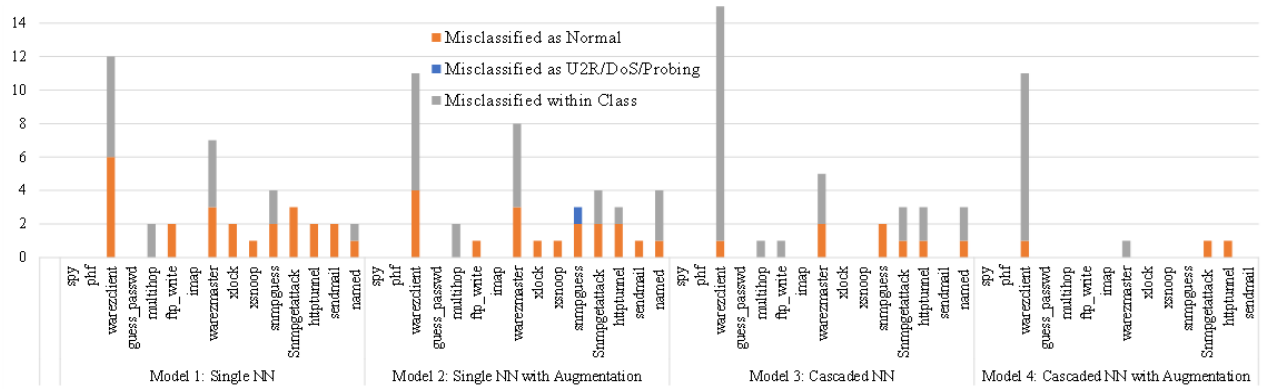

Fig.6. Misclassification of R2L class samples as normal, other classes and subclass level in own class

The proficiency of the proposed model is more visible from the test set misclassification analysis for the R2L class shown in Fig. 6. According to Fig. 6, a large number of misclassifications is observed for the warezclient subclass and has remarkable findings. Model 2 increases misclassified within the major class from Model 1 in the warezclient subclass from 6 to 7. But Model 2 manages to decrease the number of misclassifications from 12 to 11. The other point to be noted here is that Model 3 not only increases misclassified in subclass (from 7 to 14) but also increases the total misclassification (11 to 15). This represents only cascaded techniques that might not be good enough for some of the specific subclasses. Finally, Model 4 decreases the total misclassification (i.e., 15 to 11) as well as misclassified in the subclass level (i.e., 14 to 10). Though Model 4 cannot vanish the misclassification, it shows good performance by keeping misclassified as Normal, only 2, which is the major concern.

U2R attack class has the least number of samples among all five classes, and Fig. 7 demonstrates misclassification like other classes explained above. Apart from the buffer_overflow, Model 4 can remove all the misclassifications. There are two misclassifications by Model 4 for Buffer_overflow misclassified as Normal is only one. On the other hand, the misclassifications as Normal class were 4, 4, and 2 by Model 1, Model 2, and Model 3, respectively. With the lowest misclassification, Model 4 yet again manifests its effectiveness even with low data.

One of the main concerns of the study is to emphasize not classifying attack samples as Normal rather than vice-versa. To observe the system's performance in this regard, hacking classes misclassification to Normal class and vise-

versa are summarized in Fig. 8. and Fig. 9, respectively. Fig. 8 shows misclassifications of the Normal class into DoS, Probing, R2L, or U2R attack classes by four models. The number of misclassified as U2R equals five for all four models. The same goes for Probing, except Model 4 has a slight increase in misclassifying Normal as Probing class. For DoS and R2L, Model 2 has a smaller number of misclassifications than Model 1, but the misclassifications increase in Model 3 and Model 4. Although the number of misclassification samples (less than 100 in total) is negligible concerning the total Normal test samples 15,411, Model 4 is not better than other models for Normal class samples classification. But the most significant issue to minimizing the misclassification of attack class samples as Normal class is observed in Fig. 9. It is observed from the figure that Model 1 is the worst among the four models having the highest number of misclassifications (in total 160) of four different hacking class samples as Normal. The number of misclassifications decreased for Model 2 (i.e., 114) and Model 3 (i.e., 78) gradually, and Model 4 performed the best with a total number of misclassifications of 39. At a glance, the proposed Model 4 efficiently decreases hacking attacks misclassified as Normal, which is crucial in defending a system from suspicious attacks and a major concern of the study.

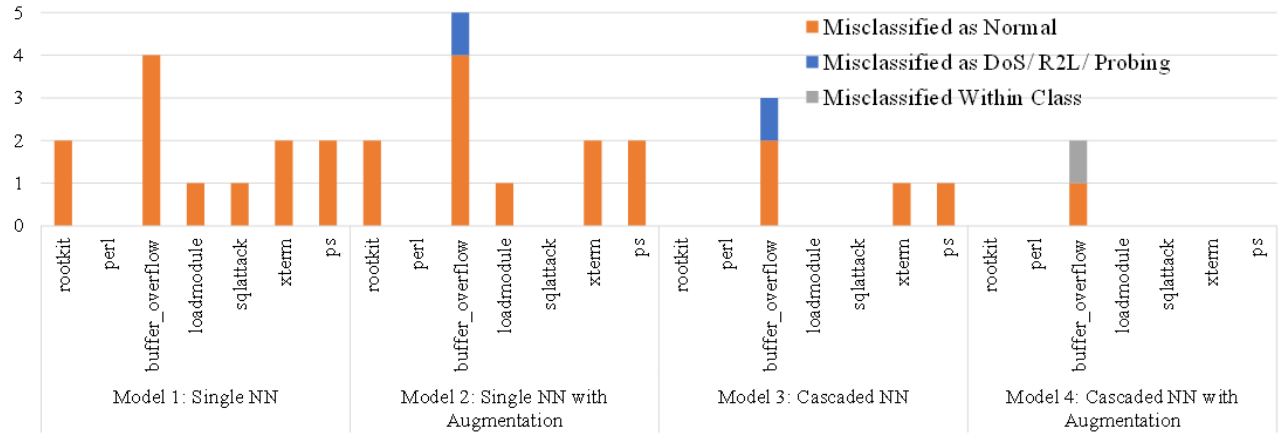

Fig.7. Misclassification of U2R class samples as normal, other classes and subclass level in own class

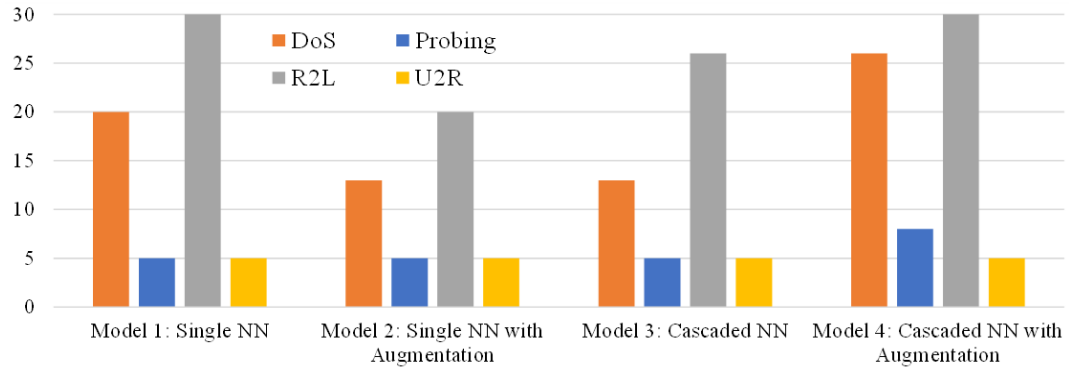

Fig.8. Misclassification of normal class samples as hacking classes (i.e., DoS, Probing, R2L and U2R)

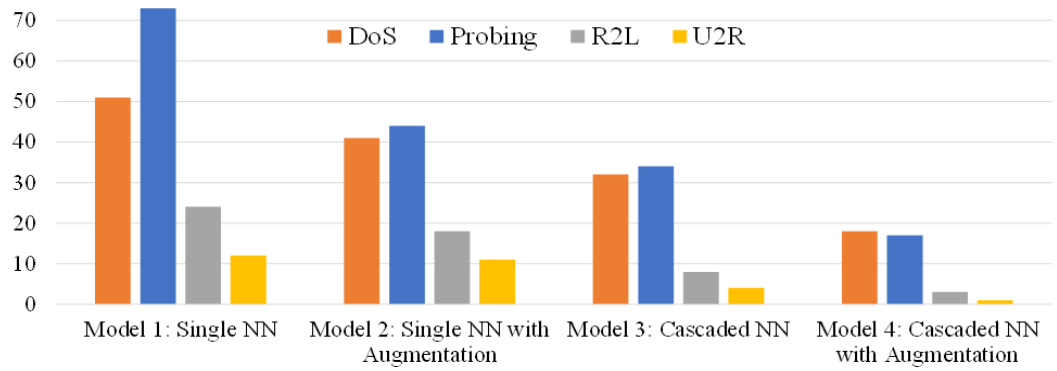

Fig.9. Misclassification of hacking classes (i.e., DoS, Probing, R2L and U2R) sample as normal class

To evaluate the overall performance of the four implemented models, the Precision, Recall, F1 Score, and Accuracy of the models are measured on the test samples. Eq. (1) – Eq. (4) are the well-known measures of the scores. Table 3 shows the achieved Precision, Recall, F1 Score, and Accuracy scores of individual models. The presented results are the

summary of the experiment analyzed above. According to the results presented in the table, Model 1 (i.e., single NN without data augmentation) is the worst, and Model 4 (proposed cascaded model with data augmentation) is the best.

$$Precision = TP/(TP + FP) \quad (1)$$

$$Recall = TP/(TP + FN) \quad (2)$$

$$F1\,Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

$$Accuracy = (TP + TN)/(TP + TN + FP + FN) \quad (4)$$

Table 3. Overall performance comparison among the four implemented models

Model	Precision (%)	Recall (%)	F1 Score (%)	Accuracy (%)
Mode l	99.07	99.17	99.12	99.16
Mode 2	99.19	99.22	99.20	99.22
Mode 3	99.57	99.57	99.57	99.34
Mode 4	**99.63**	**99.72**	**99.67**	**99.42**

4.2. Performance Comparison with Existing Models

Table 4 compares the performance of the proposed IDS with other prominent works with the NSL-KDD dataset. Computational methods used by the individual studies and other important remarkable issues are mentioned in the table for better comparison among the methods. As stated earlier already, the previous IDS work on the NSL-KDD dataset was focused on increasing the accuracy of an ML model. But the concentration of this study was not only increasing the accuracy of the model but also giving more attention to handling attack samples not to be classified as Normal. The efforts eventually helped to reach a satisfactory accuracy of the proposed model. Finally, the proposed cascaded model with data augmentation (i.e., Model 4) achieved remarkable accuracy of 99.42%, which is the best among the methods mentioned in the table. Moreover, the proposed models' evaluation is on the subclass level, which is a 40-class classification problem; on the contrary, the existing methods were concerned only major class level, which is a 5-class problem. In the case of ML, problems with a large number of classes are usually more complicated than problems with fewer classes. In such circumstances, the proposed method is much more effective than any other existing method. It is also notable that existing methods considered only accuracy issues without emphasizing attack matter with classification major class levels. Therefore, attack emphasizing and classification up to subclass levels are the two folds significance of the proposed study over the existing studies with the NSL-KDD dataset.

Table 4. Performance comparison of proposed model with state-of-arts on NSL-KDD dataset

Work Ref.	Method	Accuracy	Test Data for Performance Evaluation	Level of Classification	Attack Emphasized	Remarks and Significance
Dhanabal & Shantharajah, 2015[25]	J48	98.88%	20% of NSL-KDD Dataset	Major Class Only	No	Emphasis on the accuracy of the ML models
	SVM	95.20%				
	Naïve Bayes	73.32%				
Ingre & Yadav, 2015[28]	ANN	79.90%	NSL-KDD Test Set	Major Class Only	No	Emphasis on the accuracy of the ML model
Almi'Ani et al., 2018[17]	Self-Organized Map and K-means clustering	83.46%		Major Class Only	No	Emphasis on the accuracy of that model
Ludwig, 2019[27]	Autoencoder, Deep Belief NN	92.50%		Major Class Only	No	Ensemble model and emphasis on the accuracy of that model
Ding & Zhai, 2018[19]	CNN	80.13%		Major Class Only	No	Emphasis on the accuracy of the DL model
Proposed Model	Cascaded NN with Data Augmentation (Model 4)	**99.42%**	20% of Total NSL-KDD Dataset	Classification in Subclass Level	Yes	Cascaded framework, classification in subclass level, and emphasis not to classify attack as Normal

5. Conclusions

This paper has presented the development of a machine Learning (ML)-based effective intrusion detection system (IDS) by addressing the three crucial issues with innovative techniques. Specifically, the proposed ML-based IDS in this study copes with a high imbalance in the dataset, i.e., having fewer attack samples, emphasizing that attacks should not go into the Normal class while classifying them further in subclass levels. For a compelling training scheme, data is

augmented reasonably so that samples of individual classes and subclasses hold an appropriate ratio for proper training of the ML model. In augmentation, a limited number of instances of the attack class/subclass in the dataset are multiplied to make them sufficiently large in number, emphasizing the consistent distributions of samples per class/subclass in the training set. The cascaded ML model is developed for classifying attack samples into subclass levels, i.e., once the attack is detected, the attack instance is further fed to the next ML model to classify its subclass. The proposed cascaded framework with data augmentation is implemented with NN (the most popular ML model for classification) and evaluated with the NSL-KDD dataset. The same task has been performed with three other models: the cascaded model without data augmentation and a single NN model with and without data augmentation, to observe the proficiency of the employed techniques properly. Experimental results on the NSL-KDD dataset have revealed that the proposed cascaded model is the best, showing 99.42% accuracy on the test set. In contrast, the single NN model without data augmentation is the worst (with 99.16% accuracy) among the four models implemented and analyzed for comparison. Remarkably, the proposed cascaded model efficiently reduced the hacking attack misclassification as Normal, which is crucial in defending a system from suspicious attacks and a major concern of the study. Finally, based on superior accuracy, the proposed method is identified as an effective IDS system with distinct properties to tackle attack issues concerning the existing state-of-the-art ML-based IDS models. At a glance, attack emphasizing and classification up to subclass levels are the two notable significances of the proposed study over the existing ones.

Several future potential research directions have been explored from this study. Firstly, instead of NN in the proposed cascaded framework, other classifiers, especially DL models (e.g., CNN, DBN), might perform differently. Adaptation of data augmentation and appropriate DL model selection might give better performance for ML-based IDS. Another future work could be to adopt this IDS for a real-time scenario in the underlying security system. More interestingly, the idea of data augmentation and cascaded framework will be applicable for similar scenario cases, such as security issue handling in vehicular Ad hoc networks (VANET), since it is also challenging due to the lack of intruder instances.

References

[1] Y. Li and Q. Liu, "A comprehensive review study of cyber-attacks and cyber security; Emerging trends and recent developments," *Energy Reports*, 2021, doi: 10.1016/j.egyr.2021.08.126.

[2] M. Z. Gunduz and R. Das, "Cyber-security on smart grid: Threats and potential solutions," *Comput. Networks*, vol. 169, p. 107094, Mar. 2020, doi: 10.1016/j.comnet.2019.107094.

[3] M. Yıldırım and I. Mackie, "Encouraging users to improve password security and memorability," *Int. J. Inf. Secur.*, vol. 18, no. 6, pp. 741–759, Dec. 2019, doi: 10.1007/s10207-019-00429-y.

[4] A. Awasthi and N. Goel, "Phishing website prediction using base and ensemble classifier techniques with cross-validation," *Cybersecurity*, vol. 5, no. 1, p. 22, Nov. 2022, doi: 10.1186/s42400-022-00126-9.

[5] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *J. Comput. Sci.*, vol. 25, pp. 152–160, Mar. 2018, doi: 10.1016/j.jocs.2017.03.006.

[6] Z. Ahmad, A. Shahid Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Trans. Emerg. Telecommun. Technol.*, vol. 32, no. 1, Jan. 2021, doi: 10.1002/ett.4150.

[7] N. Awadallah Awad, "Enhancing Network Intrusion Detection Model Using Machine Learning Algorithms," *Comput. Mater. Contin.*, vol. 67, no. 1, pp. 979–990, 2021, doi: 10.32604/cmc.2021.014307.

[8] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, "Benchmarking datasets for Anomaly-based Network Intrusion Detection: KDD CUP 99 alternatives," in *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, 2018, pp. 1–8, doi: 10.1109/CCCS.2018.8586840.

[9] B. Ingre, A. Yadav, and A. K. Soni, "Decision Tree Based Intrusion Detection System for NSL-KDD Dataset," in *Smart Innovation, Systems and Technologies*, 2018, pp. 207–218.

[10] S. M. Kasongo and Y. Sun, "A Deep Learning Method With Filter Based Feature Engineering for Wireless Intrusion Detection System," *IEEE Access*, vol. 7, pp. 38597–38607, 2019, doi: 10.1109/ACCESS.2019.2905633.

[11] A. KumarShrivas and A. Kumar Dewangan, "An Ensemble Model for Classification of Attacks with Feature Selection based on KDD99 and NSL-KDD Data Set," *Int. J. Comput. Appl.*, vol. 99, no. 15, pp. 8–13, Aug. 2014, doi: 10.5120/17447-5392.

[12] C. Liu, Z. Gu, and J. Wang, "A Hybrid Intrusion Detection System Based on Scalable K-Means+ Random Forest and Deep Learning," *IEEE Access*, vol. 9, pp. 75729–75740, 2021, doi: 10.1109/ACCESS.2021.3082147.

[13] P. Sangkatsanee, N. Wattanapongsakorn, and C. Charnsripinyo, "Practical real-time intrusion detection using machine learning approaches," *Comput. Commun.*, vol. 34, no. 18, pp. 2227–2235, Dec. 2011, doi: 10.1016/j.comcom.2011.07.001.

[14] M. Sarnovsky and J. Paralic, "Hierarchical Intrusion Detection Using Machine Learning and Knowledge Model," *Symmetry (Basel).*, vol. 12, no. 2, p. 203, Feb. 2020, doi: 10.3390/sym12020203.

[15] N. Satheesh *et al.*, "Flow-based anomaly intrusion detection using machine learning model with software defined networking for OpenFlow network," *Microprocess. Microsyst.*, vol. 79, p. 103285, Nov. 2020, doi: 10.1016/j.micpro.2020.103285.

[16] S. P. Thirimanne, L. Jayawardana, L. Yasakethu, P. Liyanaarachchi, and C. Hewage, "Deep Neural Network Based Real-Time Intrusion Detection System," *SN Comput. Sci.*, vol. 3, no. 2, p. 145, Mar. 2022, doi: 10.1007/s42979-022-01031-1.

[17] M. Almi'ani, A. A. Ghazleh, A. Al-Rahayfeh, and A. Razaque, "Intelligent intrusion detection system using clustered self organized map," in *2018 Fifth International Conference on Software Defined Systems (SDS)*, 2018, pp. 138–144, doi: 10.1109/SDS.2018.8370435.

[18] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, 2009, pp. 1–6, doi: 10.1109/CISDA.2009.5356528.

[19] Y. Ding and Y. Zhai, "Intrusion Detection System for NSL-KDD Dataset Using Convolutional Neural Networks," in *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence - CSAI '18*, 2018, pp. 81–85, doi: 10.1145/3297156.3297230.
[20] H. Zhou, L. Kang, H. Pan, G. Wei, and Y. Feng, "An intrusion detection approach based on incremental long short-term memory," *Int. J. Inf. Secur.*, vol. 22, no. 2, pp. 433–446, Apr. 2023, doi: 10.1007/s10207-022-00632-4.
[21] Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset," *IEEE Access*, vol. 9, pp. 22351–22370, 2021, doi: 10.1109/ACCESS.2021.3056614.
[22] M. A. Khan, "HCRNNIDS: Hybrid Convolutional Recurrent Neural Network-Based Network Intrusion Detection System," *Processes*, vol. 9, no. 5, p. 834, May 2021, doi: 10.3390/pr9050834.
[23] A. M. Mahfouz, D. Venugopal, and S. G. Shiva, "Comparative Analysis of ML Classifiers for Network Intrusion Detection," in *Advances in Intelligent Systems and Computing*, 2020, pp. 193–207.
[24] A. Lamba, S. Singh, S. Bhardwaj, N. Dutta, and S. S. R. Muni, "Uses of Artificial Intelligent Techniques to Build Accurate Models for Intrusion Detection System," *SSRN Electron. J.*, vol. 2, no. 12, pp. 5826–5830, 2015, doi: 10.2139/ssrn.3492675.
[25] L. Dhanabal and S. P. Shantharajah, "A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms," *Int. J. Adv. Res. Comput. Commun. Eng.*, vol. 4, no. 6, pp. 446–452, 2015.
[26] J. Kevric, S. Jukic, and A. Subasi, "An effective combining classifier approach using tree algorithms for network intrusion detection," *Neural Comput. Appl.*, vol. 28, no. S1, pp. 1051–1058, Dec. 2017, doi: 10.1007/s00521-016-2418-1.
[27] S. A. Ludwig, "Applying a Neural Network Ensemble to Intrusion Detection," *J. Artif. Intell. Soft Comput. Res.*, vol. 9, no. 3, pp. 177–188, Jul. 2019, doi: 10.2478/jaiscr-2019-0002.
[28] B. Ingre and A. Yadav, "Performance analysis of NSL-KDD dataset using ANN," in *2015 International Conference on Signal Processing and Communication Engineering Systems*, 2015, pp. 92–96, doi: 10.1109/SPACES.2015.7058223.
[29] H. Alavizadeh, H. Alavizadeh, and J. Jang-Jaccard, "Deep Q-Learning Based Reinforcement Learning Approach for Network Intrusion Detection," *Computers*, vol. 11, no. 3, p. 41, Mar. 2022, doi: 10.3390/computers11030041.

Authors' Profiles

Argha Chandra Dhar received his B.Sc. degree in Computer Science and Engineering from Khulna University of Engineering & Technology (KUET), Bangladesh in February, 2023. He has several research publications in international journals and conferences from the researches during his undergraduate study period. His research interest includes machine learning, deep learning, cybersecurity and game development. Now he is preparing for higher studies abroad to conduct research on those fields.

Arna Roy received her B.Sc. degree in Computer Science and Engineering from Khulna University of Engineering & Technology (KUET), Bangladesh in February, 2023. She has several research publications from thesis and project works in her undergraduate study period. Her research interest includes machine learning, deep learning and cybersecurity. Now she is preparing for higher studies abroad to conduct research on those fields.

M. A. H. Akhand received his B.Sc. degree in Electrical and Electronic Engineering from Khulna University of Engineering and Technology (KUET), Bangladesh, in 1999, the M.E. degree in Human and Artificial Intelligent Systems, in 2006, and the Ph.D. degree in System Design Engineering, in 2009 from University of Fukui, Japan. He joined as a lecturer at the Department of Computer Science and Engineering at KUET in 2001, and is now Professor since 2014. He is also head of Computational Intelligence Research Group of this department. He is a life member of Institution of Engineers, Bangladesh (IEB), a life time member Bangladesh Computer Society (BCS) and a senior member of IEEE. He has more than 150 research publications. His research interest includes artificial neural networks, evolutionary computation, bioinformatics, swarm intelligence and other bio-inspired computing techniques. Dr. Akhand received several best paper Prizes in international conferences.

Md. Abdus Samad Kamal received the B.Sc. Engineering degree from Khulna University of Engineering and Technology (KUET), Bangladesh, in 1997 and the Master and Ph.D. degrees from Kyushu University, Japan, in 2003 and 2006, respectively. He was a Lecturer with KUET from 1997 to 2000, a Researcher with Kyushu University in 2006, and 2008 to 2011, an Assistant Professor with International Islamic University Malaysia, Malaysia from 2006 to 2008, a Researcher with The University of Tokyo, Japan from 2011 to 2014. He worked as a Visiting Researcher with Toyota Central R \& D Labs., Inc., Japan, from 2014 to 2016. He was a Senior Lecturer in the School of Engineering, Monash University Malaysia from 2016 to 2019. Currently, he is an Associate Professor in the Graduate School of Science and Technology, Gunma University, Japan. His research interests include intelligent transportation systems, connected and automated vehicles, and the applications of machine learning and model predictive control. Dr.

Kamal is a member of the Society of Instrument and Control Engineers (SICE), a Senior Member of Institute of Electrical and Electronics Engineers (IEEE), and a Chartered Engineer of Institution of Engineering and Technology (IET).

Kou Yamada was born in Akita, Japan, in 1964. He received B.S. and M.S. degrees from Yamagata University, Yamagata, Japan, in 1987 and 1989, respectively, and the Dr. Eng. degree from Osaka University, Osaka, Japan in 1997. From 1991 to 2000, he was with the Department of Electrical and Information Engineering, Yamagata University, Yamagata, Japan, as a research associate. From 2000 to 2008, he was an associate professor in the Department of Mechanical System Engineering, Gunma University, Gunma, Japan. Since 2008, he has been a professor in the Department of Mechanical System Engineering, Gunma University, Gunma, Japan. His research interests include robust control, repetitive control, process control and control theory for inverse systems and infinite-dimensional systems. Dr. Yamada received the 2005 Yokoyama Award in Science and Technology, the 2005 Electrical Engineering/Electronics, Computer, Telecommunication, and Information Technology International Conference (ECTI-CON2005) Best Paper Award, the Japanese Ergonomics Society Encouragement Award for Academic Paper in 2007, the 2008 Electrical Engineering/Electronics, Computer, Telecommunication, and Information Technology International Conference (ECTI-CON2008) Best Paper Award, Fourth International Conference on Innovative Computing, Information and Control Best Paper Award in 2009 and 14th International Conference on Innovative Computing, Information and Control Best Paper Award in 2019, and Outstanding Achievement Award from Kanto Branch of Japanese Society for Engineering Education in 2022. He is a member of IEEE and SICE.

How to cite this paper: Argha Chandra Dhar, Arna Roy, M. A. H. Akhand, Md. Abdus Samad Kamal, Kou Yamada, "Cascaded Machine Learning Approach with Data Augmentation for Intrusion Detection System", International Journal of Computer Network and Information Security(IJCNIS), Vol.16, No.4, pp.17-30, 2024. DOI:10.5815/ijcnis.2024.04.02