Journal of Big Data

RESEARCH Open Access

Model fusion of deep neural networks for anomaly detection

Nouar AlDahoul*, Hezerul Abdul Karim and Abdulaziz Saleh Ba Wazir

*Correspondence:
nouar.aldahoul@live.iium.edu.my
Faculty of Engineering, Multimedia University, Cyberjaya, Malaysia

Abstract

Network Anomaly Detection is still an open challenging task that aims to detect anomalous network traffic for security purposes. Usually, the network traffic data are large-scale and imbalanced. Additionally, they have noisy labels. This paper addresses the previous challenges and utilizes million-scale and highly imbalanced ZYELL's dataset. We propose to train deep neural networks with class weight optimization to learn complex patterns from rare anomalies observed from the traffic data. This paper proposes a novel model fusion that combines two deep neural networks including binary normal/attack classifier and multi-attacks classifier. The proposed solution can detect various network attacks such as Distributed Denial of Service (DDOS), IP probing, PORT probing, and Network Mapper (NMAP) probing. The experiments conducted on a ZYELL's real-world dataset show promising performance. It was found that the proposed approach outperformed the baseline model in terms of average macro Fβ score and false alarm rate by 17% and 5.3%, respectively.

Keywords: Anomaly detection, Deep neural network, Highly imbalanced data, Model fusion, Class weight optimization

Introduction

In today's digital age, network security is critical as billions of computers around the world are connected over networks. The number of network attacks has been increased largely in recent years. Therefore, network threat detection aims to detect these attacks by observing traffic data over time and distinguishes anomalous behaviors from normal traffic [1].

Network anomaly detection (NAD) is a technique that facilitates network security with threat detection based on traffic exceptional patterns. NAD operates by continuously monitoring a network for unusual events or trends [1]. Therefore, NAD is usually an integral part of network behaviour analysis (NBA), in which network security is provided by anti-threat applications such as antivirus software, firewall, spyware-detection software, and intrusion detection systems [2].

Network attacks have risen drastically with Internet technology's advancements. Consequently, network intrusion detection has become an essential field to improve the systems' capability of detecting attacks for network security. Intrusion threat is a deliberate

attempt to access and manipulate information in order to render system unreliable. For example, Denial of Service (DoS) [3].

Anomaly-based intrusion detection is the process used to find nonconforming patterns in network traffic that do not match the expected normal traffic of the network. These exceptional patterns are anomalies, outliers, or exceptions [4, 5]. NAD has been extensively used for many applications such as intrusion detection for cyber security, and fraud detection of credit cards [3].

Anomaly detection systems have been developed over the years based on statistical algorithms [6], data mining approaches [7], and machine learning [8]. Most of NAD methods usually depend on developing a model for normal behaviors, and thus the developed models can detect any abnormal pattens [8]. There are many types of patterns learning in NAD systems such as supervised, semi-supervised, and unsupervised learning [9].

In recent years, deep learning methods have received much attention, since deep neural networks are able to learn complex patterns of anomalies directly from the network traffic data [10]. However, real-world traffic data are large scale, noisy labelled, and class imbalanced. In other words, the traffic data have millions of samples which are distributed unevenly with rare anomalies, and too much normal traffic data. Most of the existing network datasets do not meet the real-world conditions and not suitable for modern networks. Furthermore, Traditional datasets such as kddcup99 [11] and UNSW-NB15 [12] have been investigated largely in the literature. The methods that utilized these datasets were able to give high performance. Therefore, in this paper, we focused on the problem of large-scale (million-scale) and highly imbalanced traffic data using ZYELL's dataset [13, 14] to train, validate, and test the proposed solution.

The novel solution proposed in this paper is considered under hybrid method for NAD. The model fusion of two deep neural networks (DNNs) was utilized to detect attacks and map them to specific categories. The first end to end DNN was used to learn patterns from the traffic data for normal/attack binary classification. The second end to end DNN was utilized to learn patterns from the traffic data for classification of four types of attacks such as DDOS smurf, probing IP sweep, probing PORT sweep, and probing NMAP. The results presented in this paper show that the proposed approach outperformed the traditional single deep neural network in terms of Fβ Score and false alarm rate.

Previous work

Statistical algorithms for NAD track the network behavior using probabilistic models of anomalies. Anomalous attacks are associated with abnormal changes in the data flowing through a network. Generally, these exceptional changes are detected via hard threshold modelling techniques. The major drawback of hard threshold modelling for statistical approaches is the generation of high false alarms [15]. Consequently, Statistical approaches aimed to develop methods that can help to reduce false alarms.

Real time NAD has been developed using wavelets combined with sketches [16]. This method was a router level analysis that extracts NetFlow traces by converting traces into ASCII files. Then, the sketches used hash functions to aggregate traffic flows in the

sketch tables. Next, the produced time series were used to discover discontinuities by wavelet transform.

Correlational Paraconsistent Machine (CPM) is another method that has been developed for NAD. CPM used two methods including non-classical paraconsistent logic (PL), and unsupervised traffic characterization [17]. For example, a study has developed NAD based on both auto regressive integrated moving average (ARIMA), and ant colony optimization for digital signature (ACODS) [18] to generate two distinct network profiles that can identify normal network traffic data.

Classification methods have been widely used for NAD problem [19]. For example, Naive Bayesian classifier (NBC) has been used to detect Distributed Denial of Service (DDoS) attacks, selective forwarding, and black holes [20]. The developed NBC system was utilized to monitor packets moving between nodes to check the behavior of data for abnormality detection. The classifier calculates the probability of the samples that belong to a class based on normal distribution probability approach.

Support Vector Machine (SVM) is another known classifier that was used to find patterns and provide autonomous recognition of normal data traffic in NAD problem [21]. The main problems include training SVM classifier with imbalance in class distribution and outlier sensitivity of decision boundary. To address previous problems, two modifications of the unsupervised one-class SVM have been proposed including eta one-class SVMs, and Robust one-class SVMs [22]. Least square support vector machine (LS-SVM) was found as a modification of standard SVM classifier that has been used for intrusion detection [23]. LS-SVM is more sensitive to noise and anomalies compared to a standard SVM.

An ensemble approach for NAD has been demonstrated using AdaBoost algorithm, that combines multiple classifiers including decision tree, K-nearest neighbor (k-NN), naive Bayes, SVM, and multilayer perceptron (MLP) [24]. The AdaBoost algorithm was used to initialize data distribution, classifiers training, error evaluation, and weights assignment to each of the classifiers. Then, weighted voting approach has been used to combine the classifiers prediction of outliers.

Delayed Long Short-Term Memory (dLSTM) was a recent deep learning method that has been used for NAD problem with time-series data [10]. A predictive model was developed based on normal training data, then anomalies were detected from observed data using prediction error. The study proposed to develop multiple LSTM predictive models and produce multiple prediction values. Then, the model with the predictive value that is closest to the measured value was selected. Their developed model can delay the timing of prediction until the associated measured value is acquired.

NAD problem is usually associated with extreme class imbalance issue. However, recent studies, that have considered the data imbalance problem associated with NAD solutions are still limited [25]. Several techniques such as algorithmic-level and data-level approaches have been proposed in the imbalanced class domain and found to improve the performance of models. Algorithmic-level methods have been used to handle the issue of data imbalance by modifying existing algorithms [26] such as hyperparameters optimization for imbalanced data classification [27]. On the other hand, date-level or resampling methods were standard approaches that can handle class imbalance issue by adding more data (creating synthetic samples) to the original

dataset and generating new balanced dataset [28]. However, learning from imbalanced data is still an open challenge.

Recent study has been proposed to introduce additional attributes to seven different imbalanced datasets for NAD [25]. The attributes include an outlier score and four types of samples including borderline, safe, rare, and outlier to gain additional information, enrich imbalanced data characteristics, and improve the classification performance.

Cascade two-stage deep learning model based on a deep stacked auto-encoder was demonstrated for network intrusion detection [29]. It includes two stages with two hidden layers in each. The deep learning model was trained in unsupervised manner on unlabelled traffic and was fine-tuned using labelled traffic data. The first stage was used to classify the normal and abnormal traffic. On the other hand, the second stage detects normal with other types of attacks.

A convolutional neural network-based payload classification (CNN) and a recurrent neural network-based payload classification (RNN) were used separately for attack detection [30]. Additionally, ID hybrid convolutional recurrent neural network (CRNN) was used to predict malicious attacks in the network [31]. The CNN and the RNN capture local and temporal features, respectively. Convolution Neural Network (CNN) was utilized to extract the accurate representation of data that were classified by Long Short-Term Memory (LSTM) Model [32].

In summary, deep learning methods for attack detection are divided into several categories [33]:

(1) Supervised methods [deep neural network (DNN), convolutional neural network (CNN), recurrent neural network (RNN)].
(2) Unsupervised methods [deep belief network (DBN), autoencoder (AE), generative adversarial network (GAN)].
(3) Hybrid methods (ensemble learning, multimodal learning).

This paper is organized as follows: "Previous work" Section describes datasets, data splitting, and data pre-processing. The proposed approach of model fusion was demonstrated in "Materials and methods" Section. In "Results and discussion" Section, the experimental setup, performance metrics, and results evaluation are discussed. Finally, "Conclusion and future work" Section summarizes the outcome and significance of this work and the open doors for future enhancement.

Materials and methods

Dataset overview

The dataset used in this work was a million-scale dataset of real-world network traffic. It was released by ZYELL group and National Chiao Tung University for network anomaly detection challenge [13, 14]. The data is a time series of network traffic records captured by ZYELL's firewall. Each network traffic record is a network connection session and is labelled as either normal or a specific type of attack. The proportion of anomalies is about 1% during the network connection session [13].

This dataset was stored as a collection of csv files with 981 MB (3 csv files) of training and 1.28 GB (4 csv files) of testing. The training dataset contains 3-date traffic logs with total of 9,241,463 samples given with labels. The three files are [13]:

A. The first file, '1210_firewall.csv' contains 3,265,630 traffic logs.
B. The second file, '1203_filewall.csv' contains 2,809,865 traffic logs.
C. The third file, '1216_filewall.csv' contains 3,165,968 traffic logs.

On the other hand, the separated testing dataset contains 4-date traffic logs with total of 13,290,530 samples that were given in the challenge without labels. The four files are [13]:

A. The first file, '0123_firewall.csv' contains 3,601,186 traffic logs.
B. The second file, '0124_filewall.csv' contains 2,050,710 traffic logs.
C. The third file, '0125_filewall.csv' contains 2,120,819 traffic logs.
D. The fourth file, '0126_filewall.csv' contains 5,517,815 traffic logs.

In Table 1, examples of the ZYELL data were shown. The traffic record has one label column and 22 features including connection duration (seconds), inbound/outbound traffic count (bytes), protocol ID, application name, number of unique source and destination IP addresses in the last T seconds, and others [13].

The raw traffic log table has the following 23 columns [13]:

['time', 'src', 'dst', 'spt', 'dpt', 'duration', 'out (bytes)','in (bytes)', 'proto', 'app', 'cnt_dst', 'cnt_src', 'cnt_serv_src', 'cnt_serv_dst', 'cnt_dst_slow', 'cnt_src_slow', 'cnt_serv_src_slow', 'cnt_serv_dst_slow', 'cnt_dst_conn', 'cnt_src_conn', 'cnt_serv_src_conn', 'cnt_serv_dst_conn', 'label']. Table 2 shows features and their descriptions in ZYELL dataset.

The other features are:

(1) Four features 'cnt_dst', 'cnt_src', 'cnt_serv_src', 'cnt_serv_dst' with '_slow' suffix for the last T′ seconds.
(2) Four features 'cnt_dst', 'cnt_src', 'cnt_serv_src', 'cnt_serv_dst' with '_conn' suffix for the last N connections.

Where T, T′, N are the selected secret numbers determined by challenge organizers [13].

This dataset targets two main categories of attacks such as denial of service (DOS) and probing. The training dataset is distributed unevenly into five categories as shown in Table 3. These categories include normal traffic, and four types of attacks including DDOS smurf. Probing Nmap, probing port sweep, and probing IP sweep [13].

Distributed Denial of Service (DDoS) is the most common type of attack that tries to stop traffic flow from and to the target system. This attack comes from different sources to target the network by flooding it with an abnormal amount of traffic, and thus the target system shutdowns to protect itself [13]. As a result, normal traffic cannot flow to network. An example of DDoS attack is when attackers send a huge number of requests to a target network as online orders for a predefined time interval which prevents customers from paying to purchase online. Smurf DDoS is a type of DDoS that occurs at

Table 1 examples of the ZYELL data with 22 features

Time	src	dst	spt	dpt	Duration	Out (bytes)	In (bytes)	Proto	App	cnt_ dst	cnt_ src	cnt_ serv_ src	cnt_ serv_ dst	cnt_ dst_ slow	cnt_ src_ slow	cnt_ serv_ src_ slow	cnt_ serv_ dst_ slow	cnt_ dst_ conn	cnt_ src_ conn	cnt_ serv_ src_ conn	cnt_ serv_ dst_ conn
12/3/2020 3:29	172.24.78.64	10.1.1.9	56,927	161	616	0	0	17	snmp	1	1	0	0	1	1	0	0	1	1	0	0
12/3/2020 3:29	172.24.78.64	10.1.1.9	56,927	161	593	0	35,391	17	snmp	1	1	1	1	1	1	1	1	1	1	1	1
12/3/2020 3:29	172.24.78.64	10.1.1.9	53,221	161	387	0	0	17	snmp	1	1	2	0	1	1	2	0	1	1	2	0
12/3/2020 3:29	172.24.78.64	10.1.1.9	53,221	161	277	0	0	17	snmp	1	1	3	1	1	1	3	1	1	1	3	1
12/3/2020 3:29	172.24.78.64	10.1.1.9	47,455	161	0	0	49,383	17	snmp	1	1	4	0	1	1	4	0	1	1	4	0
12/3/2020 3:29	172.24.78.64	10.1.1.9	47,455	161	650	0	0	17	snmp	1	1	5	1	1	1	5	1	1	1	5	1
12/3/2020 3:29	172.24.78.64	10.1.1.9	60,626	161	666	2,82,726	65,788	17	snmp	1	1	6	0	1	1	6	0	1	1	6	0
12/3/2020 3:29	172.24.78.64	10.1.1.9	60,626	161	1278	0	1,03,142	17	snmp	1	1	7	1	1	1	7	1	1	1	7	1
12/3/2020 3:29	172.24.65.130	8.8.8.8	0	0	0	0	27,609	1	icmp	1	1	0	0	1	1	0	0	1	1	0	0
12/3/2020 3:29	172.24.65.130	8.8.8.8	0	0	0	8,17,644	0	1	icmp	1	1	1	1	1	1	1	1	1	1	1	7

Table 2 The features and descriptions in ZYELL dataset [13]

Feature	Description
Time	The time when the traffic is detected by the firewall
src	Source IP address
dst	Destination IP address
spt	Source port
dpt	Destination port
Duration	Connection duration (seconds)
Out (bytes)	Outbound traffic count (bytes)
In (bytes)	Inbound traffic count (bytes)
Proto	Protocol ID
App	Application name
cnt_dst	For the same source IP address, the number of unique destination IP addresses inside the network in the last T seconds
cnt_src	For the same destination IP address, the number of unique source IP addresses inside the network in the last T seconds
cnt_serv_src	Number of connections from the source IP to the same destination port in the last T seconds
cnt_serv_dst	Number of connections from the destination IP to the same source port in the last T seconds

Table 3 Statistics about ZYELL's training set [13]

Class	ZYELL's data distribution
Normal	96.53%
DDOS	0.03%
P-IP	2.60%
P-Port	0.84%
P-NMAP	0.01%
Total	9,241,463 samples

the network layer and floods network with Internet Control Message Protocol (ICMP) packets. On the other hand, probe is another type of attack that tries to steal important information such as personal and banking information [13]. There are three types of probe attacks:

(1) IP sweep probing is ICMP echo requests (pings) sent by an attacker to several destination addresses. When the target replies, the attacker would be able to see the target's IP address from the reply [34].
(2) A port probing or scanning is a series of messages sent by an attacker to break into a computer with a port number to gain unauthorized access to sensitive information [34].
(3) Network Mapper (Nmap) is an open-source Linux command-line tool used to scan IP addresses and ports in a network [13, 34].

Dataset splitting protocol

Usually, in machine learning experiments, the data should be divided into three separated and unique sets without replicating any sample in any set. The three sets are

named: training, validation, and testing. The large part of data should be used to train the model. In this study, we divided the ZYELL's training dataset that was already labelled into three sets: training, validation, and testing as shown in Table 4. The splitting of training and testing followed the common rule (80/20) which is summarized by taking 80% of the dataset for training and 20% for testing. After that, the new training set was divided again into two sets including training and validation by the same 80/20 rule. The division was done as follows:

(1) Divide the data randomly without replicating any sample in any set.
(2) Be sure that all five categories are available in each set (training, validation, and testing).

The training set was used to train DNN to find model's weights. Furthermore, validation set was used to optimize network's architecture and finetuning hyperparameters. Finally, the testing set was used to evaluate the model and calculate the evaluation metrics.

Feature preprocessing stage

Remove irrelevant features

The 22 features have few features that are not important such as time, source IP, destination IP, source port, and destination port. These features were removed to have 17 features in each record.

Normalization techniques

The feature vector x was scaled using standard scaler by removing the mean and scaling to unit variance as follows:

$$z = (x - u)/s$$

where u is the mean, s is the standard deviation.

After scaling, clipping was done to clip the feature values between − 50 and 50 to avoid extreme outliers.

Converting categorical data to numerical representation

The column of application name is categorical and has 45 unique string values as follows [13]:

['others', 'domain', 'https', 'snmp', 'icmp', 'http', 'microsoft-ds', 'ssdp', 'netbios-ssn', 'netbios-dgm', 'ssh', 'netbios-ns', 'ftp', 'syslog', 'igmp', 'h323', 'real-audio', 'pop3', 'telnet', 'smtp', 'rtsp',

Table 4 Number of samples for training, validation, and testing set [13]

Samples type	Number of samples
Training samples	5,914,536
Validation samples	1,478,634
Testing samples	1,848,293

'pptp', 'auth', 'roadrunner', 'bgp', 'isakmp', 'rexec', 'rcmd', 'finger', 'bootps', 'sql-net', 'vdolive', 'irc', 'nntp', 'aim', 'rlogin', 'msn', 'news', 'bootpc', 'snmp-trap', 'tftp', 'nfs', 'tacacs', 'icq', 'sftp'].

Each of these unique values is repeated in the traffic record in different manners. For example, https is repeated 1,577,502 times, while sftp is only repeated 8 times. Figure 1 shows the 45 unique values in application name column with the frequency of each. The string values in application name column were converted to numerical values for further processing. One-hot encoding of column with 45 unique values result in sparse matrix with large number of zeros. Therefore, to avoid memory problem, the column was not encoded. However, it was rescaled and clipped.

The column of label is also categorical and has highly imbalanced five categories. Figure 2 illustrates the five unique values in label column with the frequency of each. The values in label column were encoded and converted to a binary form using one hot encoder.

Few numeric columns in this dataset such as cnt_src have discrete number with few tens of unique values. Figure 3 shows the values in cnt_src column with the frequency of each. The value 1 was repeated in this column more than 6 million times, while the values between 2 and 10 were repeated between 100 and 800 hundred thousand. On the contrary, other values such as ones between 20 and 40 have the frequency less than 200.

Correlation

In this section, the correlation and degree of correlation were calculated between the features of traffic samples. The matrix of correlation between each pair of features is graphically represented as a heatmap with color-coding as shown in Fig. 4. The correlation coefficients measure the strength of the relationship between the variables with the values range between − 1.0 and 1.0. In other words, a correlation of − 1.0 shows a

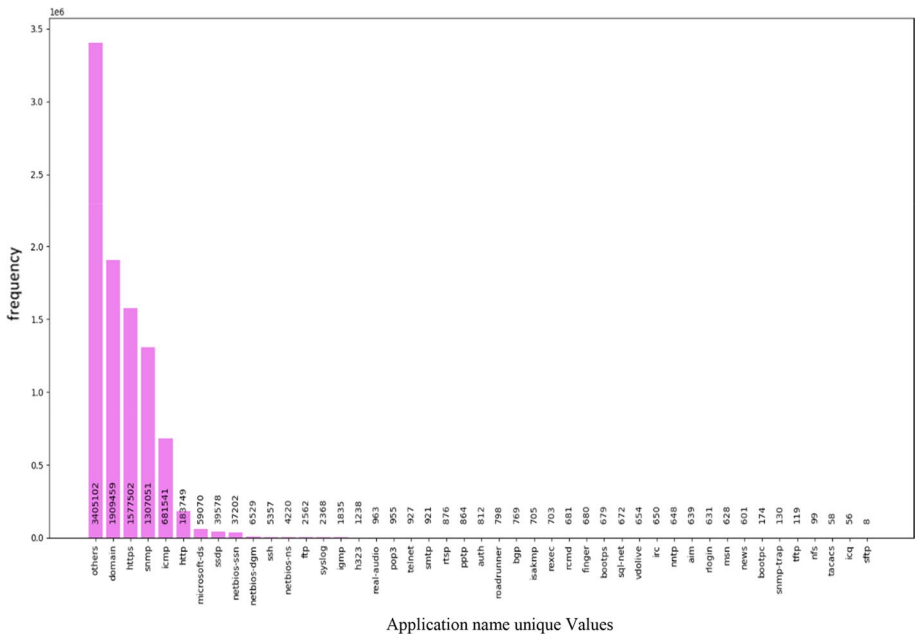

Fig. 1 Bar plot to illustrate frequency of unique values in application name column

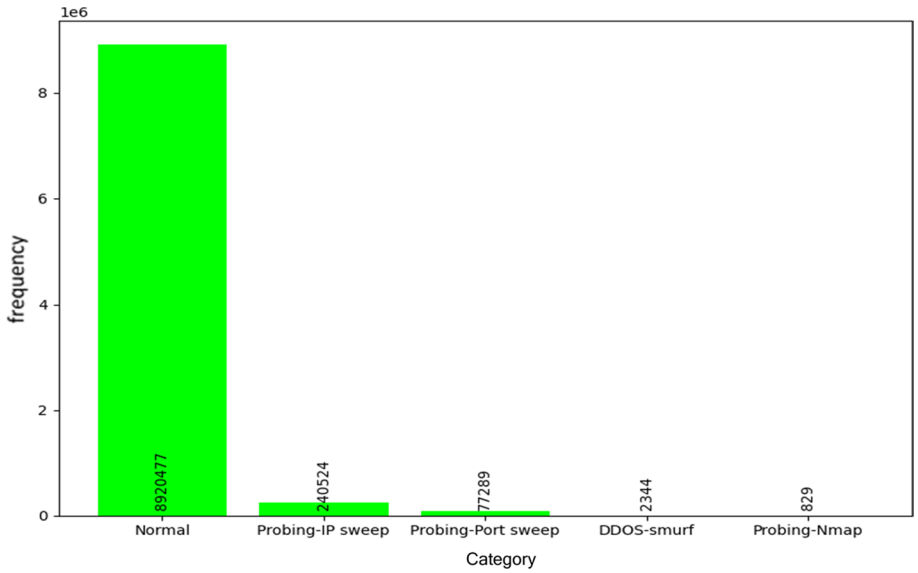

Fig. 2 Bar plot to illustrate frequency of label categories

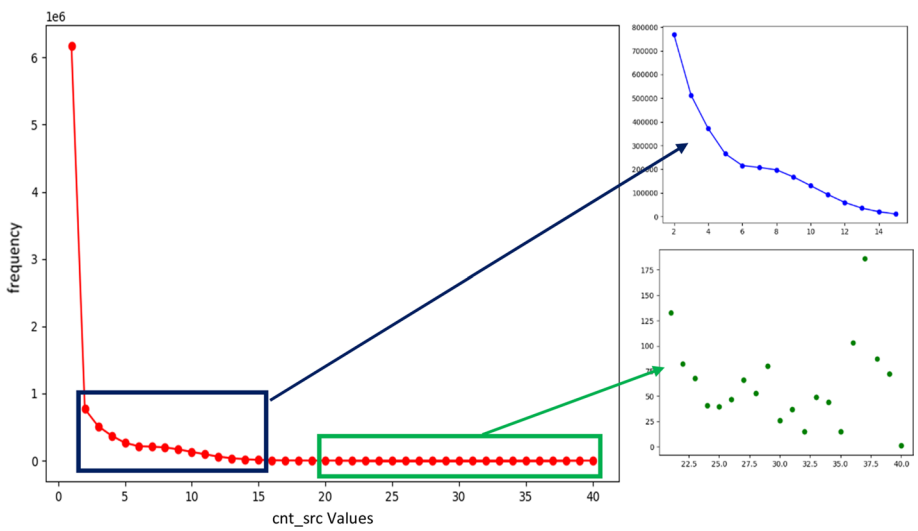

Fig. 3 The frequency of cnt_src column values

perfect negative correlation, while a correlation of 1.0 shows a perfect positive correlation (the pair of features are highly correlated). On the other hand, zero or near zero values of correlation demonstrate that this pair of features is weakly correlated. The significance of correlation matrix is related to feature selection which is the main stage before classification. When two features are highly correlated, one of the two features can be dropped.

It is obvious in Fig. 4 that label (output) which should be predicted is not highly correlated with any input feature of traffic. Additionally, there is only medium correlation (0.5–0.8) between the feature with its versions that have suffix of '_slow' and '_conn'. For example, cnt_src has medium correlation with cnt_src_slow and cnt_src_conn.

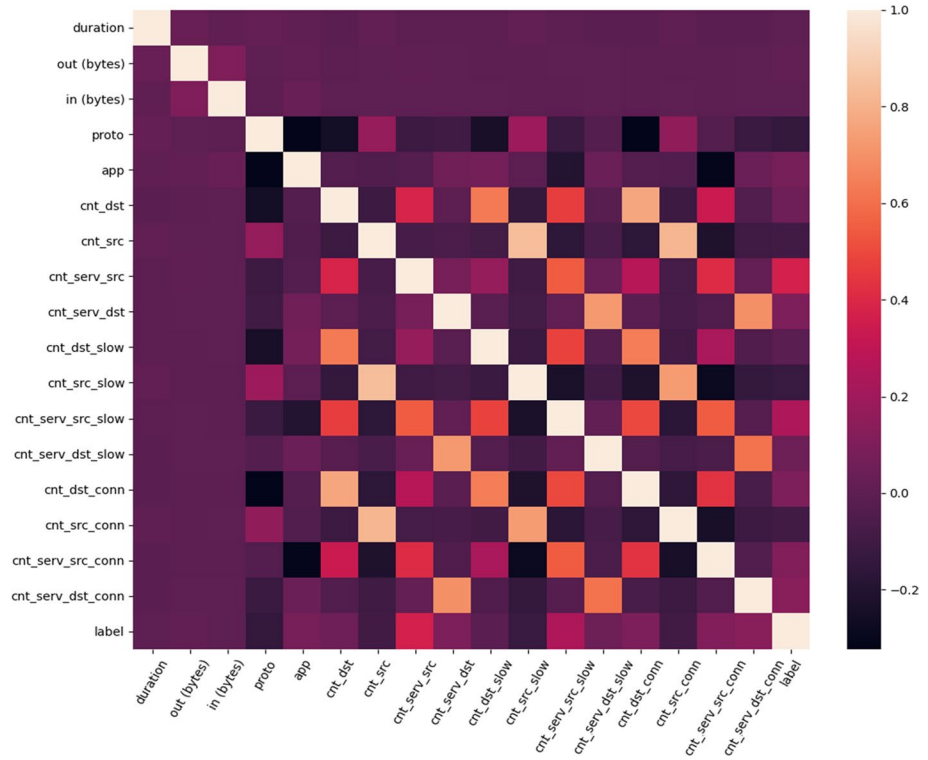

Fig. 4 Correlation heatmap between traffic features

Therefore, no feature in traffic was dropped because no one has high correlation with others.

The proposed model fusion

In this section, the proposed approach of model fusion is described. The model fusion contains two deep neural networks. The binary model 1 includes feature pre-processing and DNN. The DNN was used as a binary classifier to detect any attack by classifying traffic data into two categories: normal and attack. To compose new attack traffic set, four types of attacks including DDOS smurf, IP probing, PORT probing, and NMAP probing were combined into one set as shown in Fig. 5. The two sets of new attack traffic and normal traffic were fed to the binary DNN.

The multi-class model 2 includes feature pre-processing and DNN. The DNN was utilized as a multi-class classifier to categorize attacks into four classes after removing normal traffic data as shown in Fig. 5. The multi-class model 2 is run only if the model 1 produces an attack category. Otherwise, when normal traffic was produced at output of model 1, the model 2 is not run. The last dense layer has 2 classes in normal/attack DNN, 4 classes in multi-attacks DNN. On the other hand, the proposed approach of model fusion was compared with the baseline model. The baseline is a single deep neural network that has trained on data with five categories including normal traffic and four types of attack traffic to categorize the traffic data into 5 classes. The DNN in the baseline method has the same architecture of each of two DNNs utilized in the proposed approach as shown in Table 5.

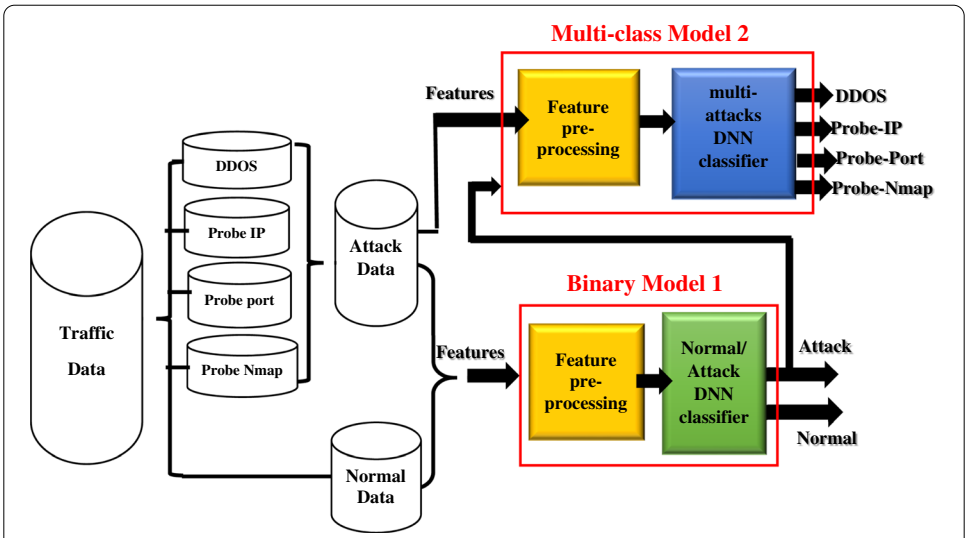

Fig. 5 Block diagram of the proposed model

Table 5 DNN Architecture

Operation layer	Size and activation
Dense	128 (Relu)
Dropout	0.5
Dense	64 (Relu)
Dropout	0.2
Dense	32 (Relu)
Dropout	0.2
Dense (classes)	2 or 4 or 5 (Softmax)

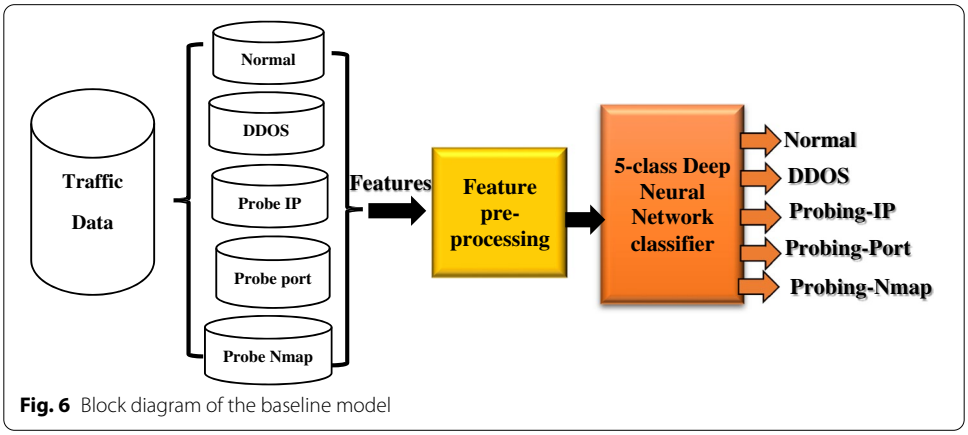

Fig. 6 Block diagram of the baseline model

Figure 6 illustrates the block diagram of the baseline model. The hyperparameter were tuned as follows:

1. Learning rate: 0.0001.

2. Batch size: 4096.
3. Epochs: (50 for binary DNN and single 5-class DNN) and (200 for 4-class DNN).
4. Optimizer: Adam.
5. Loss function:

- Categorical cross-entropy for multi-attacks DNN and 5-class DNN.
- Binary cross-entropy for normal/attack binary DNN.

The class weight optimization approach was used for model's training. The loss function which measures the performance of a classification model aims to minimize the cross entropy between the predicted probability of a sample and the actual probability. In the proposed model, the loss function used is a weighted average, where the weight of each sample is specified by a class weight. This method gives different weights to both the majority and minority classes. It aims to penalize the misclassification of traffic into normal or attack from one side and into various attack types in the second side. The penalization was done by giving a higher weight to the minority class (attack traffic samples) and a lower weight to the majority class (normal traffic samples). The weight of each class is calculated as follows:

$$W^i = \text{total}/(s \times n)$$

where total is the sum of all samples, s is the number of samples for class i, n is the number of classes.

Results and discussion

The experiments were conducted in this work to compare the proposed model fusion with the baseline method. Two scenarios were carried out. The first one is to validate the proposed solution of model fusion by using validation and testing sets obtained from training set as shown in Table 4. On the other hand, the second scenario evaluated the proposed model fusion with an external testing set including new 4-date traffic logs.

Evaluation metrics

In this section, we discuss specific performance metrics, that are ideal to evaluate methods trained on imbalanced data. The evaluation metrics used in this paper are as follows:

a) Recall (Sensitivity) is a measure that calculates the number of traffic samples predicted correctly as attacks over the number of all attack traffic samples.

$$\frac{TP}{TP + FN} \tag{1}$$

b) Precision is a measure that calculates the number of traffic samples predicted correctly as attacks over the number of all traffic predicted to exhibit attacks, both correctly or incorrectly.

$$\frac{TP}{TP + FP} \tag{2}$$

c) F_β score is the weighted harmonic mean of precision and recall. The beta parameter determines the weight of recall in the combined score. If beta is smaller than 1, more weight is given to precision. If beta is greater than 1, the recall is given more focus.

$$F\beta \text{ score} = \frac{\left(1+\beta^2\right) \times \text{precision} \times \text{recall}}{\left(\beta^2 \text{ precision} + \text{recall}\right)} \tag{3}$$

d) Evaluation Criteria of ZYELL's challenge [13] =

$$\alpha\left(1 - \frac{\log\left(\text{total cost}\right)}{\log\left(\max \cos t\right)}\right) + (1-\alpha)(\text{macro } F\beta \text{ score}) \tag{4}$$

max cost = max cost value × number of total entities.

In ZYELL NAD challenge, the value of beta and alpha were determined by challenge organizers as follows:

$\boldsymbol{\beta} = 2, \alpha = 0.3$

The total cost is the cost value calculated by a given cost matrix shown in Table 6.

e) False Alarm Rate (FAR)

The ratio between the number of normal traffic wrongly predicted to exhibit an attack and the total number of actual normal traffic. The optimal NAD system should produce low value of FAR.

$$\text{FAR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{5}$$

Evaluation results

In this paper, we presented the evaluation metrics of the baseline model which includes single multi-class (5-class) DNN trained on ZYELL's dataset. Additionally, the performance of the proposed solution, including two separated DNNs trained and evaluated on the ZYELL's dataset [13], was also evaluated. The comparison was done in terms of average micro of precision, recall, F1 score, and Fβ score. It was found that our proposed model fusion outperformed the baseline in terms of average macro precision by more than 11% as shown in Table 7. Although the baseline produced 5% better recall than our method, the F1 score and Fβ score of the proposed method were largely better than ones of the baseline by 14, and 17% respectively. The evaluation in Table 7 was done utilizing testing set that we mentioned in "Dataset splitting protocol" Section. Figures 7, 8 illustrate the confusion matrices of the baseline and the proposed, respectively.

Table 6 Cost matrix given by challenge organizers [13]

	Normal	DOS-smurf	Probing-Ip sweep	Probing-Nmap	Probing-port sweep
Normal	0	2	1	1	1
DOS-smurf	2	0	1	1	1
Probing-Ip sweep	1	2	0	1	1
Probing-Nmap	1	2	1	0	1
Probing-port sweep	1	2	1	1	0

Table 7 Classification results for the proposed and the baseline models calculated with macro averaging

Method	Macro recall	Macro precision	Macro F1 score	Macro Fβ score	False alarm rate
5-class DNN (baseline) (%)	**95**	36.67	41	48.20	7.66
Model fusion (proposed) (%)	90.33	**48**	**55.33**	**65.18**	**2.30**

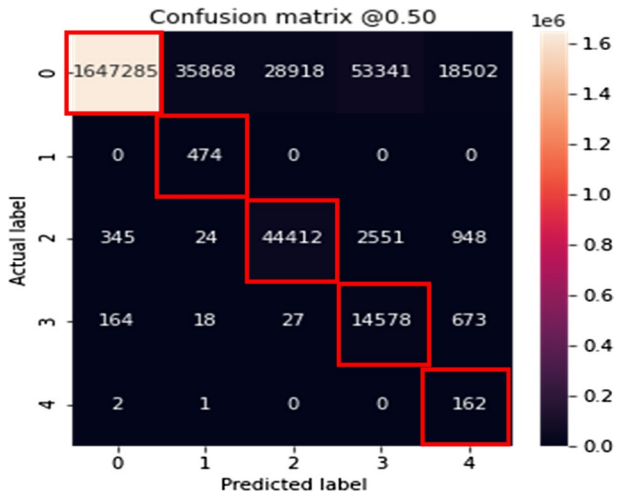

Fig. 7 Confusion Matrix of the 5-class DNN classifier (baseline model)

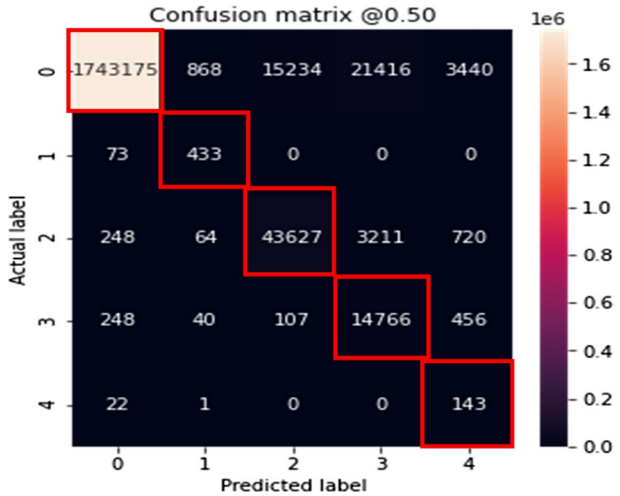

Fig. 8 Confusion Matrix of the proposed model fusion

Additionally, the comparison between the proposed and the baseline methods was done in terms of false alarm rate which is a significant performance indicator in NAD systems. The results in Table 7 shows a remarkable improvement by 5.3% in reducing the false alarm rate in the proposed approach compared to the baseline. This improvement was achieved without degrading the performance of detecting real attacks.

As shown in Figs. 7, 8, the proposed approach was able to detect 1,743,175 normal traffic compared to 1,647,285 normal traffic detected by the baseline. In other words, about 1,00,000 traffic records were misclassified by the baseline and lead to false alarm. The reason behinds the big difference in false alarm rate is that even both models were trained with the same data but the binary DNN in the proposed approach learned the patterns to distinguish between normal and attack traffic only. On the other hand, the patterns learned with the baseline target to classify the traffic into 5 classes. This confirms the significance of patterns learning in NAD system to improve the detection performance.

The evaluation in Table 8 was done using external separated testing set mentioned in "Dataset overview" Section. The score of the proposed model fusion calculated by evaluation criteria given by Eq. (4) was 30.24% which outperformed the score of the baseline solution (19.18%) by 11%.

This big difference in evaluation criteria score was mainly caused by the value 2 in cost matrix shown in Table 6. This cost results from misclassification of normal traffic records as DDoS traffic. By comparing between the two confusion matrices, the proposed method misclassified only 868 records compared to 35,868 traffic records misclassified by the baseline.

Conclusion and future work

In this paper, a novel strategy of anomaly detection and classification was proposed for network security purposes. A model fusion, that combines binary normal/attack DNN to detect the availability of any attack and multi-attacks DNN to categorize the attacks, was demonstrated. Furthermore, this paper addressed the problem of million-scale and highly imbalanced traffic data. The proposed solution was trained, validated, and tested with real world ZYELL's dataset and the results were promising. It was found that our solution outperformed the baseline solution in terms of Fβ Score by 17%. Additionally, the proposed solution played a significant role to reduce the false alarm rate that most of NAD systems are suffering from by 5.3%. Usually, the false alarm reduces the reliability of NAD system. Therefore, reducing the false alarm rate can make NAD system more robust and reliable. However, low false alarm rate in the proposed solution did not degrade the ability to detect real attacks.

For future work, we aim to enhance the performance by using other types of deep learning models such as 1D convolutional neural network (CNN) to learn spatial features and long short-term memory (LSTM) to learn temporal features. In addition, unsupervised learning of LSTM autoencoder is also a promising candidate solution for this million-scale dataset.

Table 8 The score of evaluation criteria for the proposed and baseline models using external testing dataset

Method	5-class DNN (baseline)	Model fusion (proposed)
Evaluation criteria score (%)	19.18	**30.24**

Acknowledgements
This research work was fully funded by Multimedia University, Malaysia. Additionally, Thanks to (©Zyell Solutions Corporation who has the copyright of the Syslog traffics dataset) for sharing their dataset in ICASSP2021 Challenge.

Authors' contributions
Formal analysis, NA; methodology, NA, and HAK; writing—original draft, NA and ASBW; writing—review and editing, NA, HAK and ASBW. All the authors read and approved the final manuscript.

Funding
This research project was funded by Multimedia University, Malaysia.

Availability of data and materials
©Zyell Solutions Corporation who has the copyright of the Syslog traffics dataset.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare no competing interests.

Received: 4 May 2021 Accepted: 25 July 2021
Published online: 05 August 2021

References

1. Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput Surv. 2009;41(3):1–58.
2. Patcha A, Park JM. An overview of anomaly detection techniques: existing solutions and latest technological trends. Comput Netw. 2007;51(12):3448–70.
3. Bhuyan MH, Bhattacharyya DK, Kalita JK. Network anomaly detection: methods, systems and tools. IEEE Commun Surv Tutor. 2014;16(1):303–36. https://doi.org/10.1109/SURV.2013.052213.00046.
4. Ahmed M, Mahmood AN, Hu J. A survey of network anomaly detection techniques. J Netw Comput Appl. 2016;60:19–31.
5. Kwon D, Kim H, Kim J, Suh SC, Kim I, Kim KJ. A survey of deep learning-based network anomaly detection. Clust Comput. 2017;22(1):949–61.
6. Manikopoulos C, Papavassiliou S. Network intrusion and fault detection: a statistical anomaly approach. IEEE Commun Mag. 2002;40(10):76–82. https://doi.org/10.1109/MCOM.2002.1039860.
7. Idhammad M, Afdel K, Belouch M. Distributed intrusion detection system for cloud environments based on data mining techniques. Procedia Comput Sci. 2018;127:35–41.
8. Shon T, Moon J. A hybrid machine learning approach to network anomaly detection. Inf Sci. 2007;177(18):3799–821.
9. Omar S, Ngadi A, Jebur HH. Machine learning techniques for anomaly detection: an overview. Int J Comput Appl. 2013;79(2):33-41.
10. Maya S, Ueno K, Nishikawa T. dLSTM: a new approach for anomaly detection using deep learning with delayed prediction. Int J Data Sci Anal. 2019;8(2):137–64.
11. KDD Cup 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html. 2007. Accessed 30 May 2021.
12. The UNSW-NB15 Dataset. https://research.unsw.edu.au/projects/unsw-nb15-dataset. 2015. Accessed 30 May 2021.
13. ZYELL's Dataset. https://nad2021.nctu.edu.tw/Dataset.html. Accessed 27 Apr 2021.
14. Chen L, Weng S-E, Peng C-J, Shuai H-H, and Cheng W-H. Zyell-nctu nettraffic1.0: A large-scale dataset for real-world network anomaly detection.https://arxiv.org/abs/2103.05767, 2021.
15. Thottan M, Liu G, Ji C. Anomaly detection approaches for communication networks. In: Algorithms for next generation networks. Berlin: Springer; 2010. p. 239–61.
16. Callegari C, Giordano S, Pagano M, Pepe T. Combining sketches and wavelet analysis for multi time-scale network anomaly detection. Comput Secur. 2011;30:692–704.
17. Pena EHM, Carvalho LF, Barbon SJ, Rodrigues JJPC, Proença MLJ. Anomaly detection using the correlational paraconsistent machine with digital signatures of network segment. Inf Sci. 2017;420:313–28.
18. EHM Pena, LF Carvalho, SJ Barbon, JJPC Rodrigues and MLJ Proença. Correlational paraconsistent machine for anomaly detection. In: 2014 IEEE global communications conference, pp. 551–6, 2014.
19. Duda RO, Hart PE, Stork DG. Pattern classification. New York: Wiley; 2012.
20. M Klassen and Y Ning. Anomaly based intrusion detection in wireless networks using Bayesian classifier. In: 2012 IEEE fifth international conference on advanced computational intelligence (ICACI), pp. 257–64, 2012.
21. Catania CA, Bromberg F, Garino CG. An autonomous labeling approach to support vector machines algorithms for network traffic anomaly detection. Expert Syst Appl. 2012;39:1822–9.
22. M Amer, M Goldstein and S Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In: Proceedings of the ACM SIGKDD workshop on outlier detection and description, pp. 8–15, 2013.
23. Kabir E, Hu J, Wang H, Zhuo G. A novel statistical technique for intrusion detection systems. Futur Gener Comput Syst. 2017;79:303.

24. P Sornsuwit and S Jaiyen. Intrusion detection model based on ensemble learning for U2R and R2L attacks. In: 2015 7th international conference on information technology and electrical engineering (ICITEE), pp. 354–9, 2015.
25. J Kong, W Kowalczyk, S Menzel, T Bäck. Improving Imbalanced Classification by Anomaly Detection. In: International Conference on Parallel Problem Solving from Nature, pp. 512–23, 2020.
26. Ganganwar V. An overview of classification algorithms for imbalanced datasets. Int J Emerg Technol Adv Eng. 2012;2(4):42–7.
27. Kong J, Kowalczyk W, Nguyen DA, Bäck T, Menzel S. Hyperparameter optimisation for improving classification under class imbalance. 2019 IEEE Symposium Series on Computational Intelligence (SSCI): Xiamen; 2019. p. 3072–8.
28. Fernández A, García S, Galar M, Prati RC, Krawczyk B, Herrera F. Learning from imbalanced data sets. Berlin: Springer; 2018.
29. Khan FA, Gumaei A, Derhab A, Hussain A. A novel two-stage deep learning model for efficient network intrusion detection. IEEE Access. 2019;7:30373–85. https://doi.org/10.1109/ACCESS.2019.2899721.
30. Liu H, Lang B, Liu M, Yan H. CNN and RNN based payload classification methods for attack detection. Knowl Based Syst. 2019;163:332–41. https://doi.org/10.1016/j.knosys.2018.08.036.
31. Khan MA. HCRNNIDS: hybrid convolutional recurrent neural network-based network intrusion detection system. Processes. 2021;9(5):834. https://doi.org/10.3390/pr9050834.
32. Kumar-Sahu A, Sharma S, Tanveer M, Raja R. Internet of things attack detection using hybrid deep learning model. Comput Commun. 2021;176:146–54. https://doi.org/10.1016/j.comcom.2021.05.024.
33. Wu Y, Wei D, Feng J. Network attacks detection methods based on deep learning techniques: a survey. Secur Commun Netw. 2020. https://doi.org/10.1155/2020/8872923.
34. IP Address Sweep and Port Scan. https://www.juniper.net/documentation/us/en/software/junos/denial-of-service/topics/topic-map/security-ip-sweep-and-port-option.html. Accessed 30 May 2021.

Publisher's Note