International Journal on Semantic Web and Information Systems
Volume 18 • Issue 1

False Alert Detection Based on Deep Learning and Machine Learning

Shudong Li, Guangzhou University, China*
Danyi Qin, Guangzhou University, China
Xiaobo Wu, Guangzhou University, China
Juan Li, Chinese Information Technology Security Evaluation Center, China
Baohui Li, Chinese Information Technology Security Evaluation Center, China
Weihong Han, Guangzhou University, China

ABSTRACT

Among the large number of network attack alerts generated every day, actual security incidents are usually overwhelmed by a large number of redundant alerts. Therefore, how to remove these redundant alerts in real time and improve the quality of alerts is an urgent problem to be solved in large-scale network security protection. This paper uses the method of combining machine learning and deep learning to improve the effect of false alarm detection and then more accurately identify real alarms, that is, in the process of training the model, the features of a hidden layer output of the DNN model are used as input to train the machine learning model. In order to verify the proposed method, the authors use the marked alert data to do classification experiments and finally use the accuracy recall rate, precision, and F1 value to evaluate the model. Good results have been obtained.

KEYWORDS

Artificial Intelligence, Cyber Security, Data Analysis, Unsupervised Learning

INTRODUCTION

The internet is developing very rapidly, making people's lives more convenient. While enjoying related services, it is also very important that information can be effectively protected. The integrity, privacy, and availability of information must be taken into consideration. As network security becomes more and more important, many security products, such as Firewalls, Intrusion Detection Systems, Vulnerability Scanning Systems, Update Service Systems, etc. continue to appear, and there are a large number of security data which can be used for auditing, such as router logs, syslog, host logs, etc. However, even if various security measures continue to be adopted, network security incidents have not decreased. Of course, this has a lot to do with the ever-expanding scale of the Internet, but there is no doubt that the situation of network security is becoming more and more serious.

The security guarantee of the information system is a defense system, including protection, detection, reaction, and recovery four levels (NURBOL, 2010). IDS (Intrusion Detection System) refers

DOI: 10.4018/IJSWIS.297035 *Corresponding Author

This article published as an Open Access Article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/) which permits unrestricted use, distribution, and production in any medium, provided the author of the original work and original publication source are properly credited.

to a system have intrusion detection function. The IDS is responsible for "supervising early warning" by collecting system programs, operating systems, network packets, applications, etc. Discover the behavior of hazard system security or violation of security strategies. The security policy of intrusion detection system requires the collection of complete data. This is different from the general information system. Sometimes we need to deal with a large number of warnings, which requires high computer performance. However, for intrusion detection technology, the quality of security data generated at home and abroad is very low, and a large number of artificial analyses was required.

At present, IDS has a variety of products, but the basic principles are the same, mainly divided into three modules: The data package sniffing, the alert detection engine and the report of the alert.

There are many problems around the intrusion detection system, we need to solve: Signature generates, attack detection performance measurements, the alert analysis, etc., In particular, the alert analysis has become a hot spot for related research since 2000. People find out for existing safety products: Any single security product is difficult to meet people's safety requirement. The firewall cannot prevent unknown security incidents, the alerts generated by the intrusion detection system have serious false positives and missed reports, and the amount of data from various security data sources is beyond the reach of human ability. The amount of security data generated in a large-scale network is huge, and a 100Mbps access network can often generate more than one hundred thousand alerts per hour (Li Dong et al., 2009). Among the large number of alerts generated every day, actual security incidents are usually overwhelmed by a large number of redundant alerts (ie, false alerts). Many techniques for analyzing the alerts generated by IDS: fuzzy theory, information theory, statistics, data mining machine learning (Shudong Li et al., 2021), pattern recognition, artificial intelligence, etc., their purpose is to discover real attacks from the large number of alerts generated by IDS. How to remove these redundant alerts in real time and improve the quality of alerts is an urgent problem to be solved in large-scale network security protection.

But how do false alerts occur? Taking the open-source intrusion detection system SNORT as an example, the corresponding signature (a set of conditions that the data packet needs) will generate a large number of alerts. Therefore, which will bring two main problems of alert analysis:

1. The alert level will be low. Packets that violate certain rules generate alerts, but multi-step attacks generate multiple alerts. Therefore, alerts need to be comprehensively analyzed. At this time, correlation analysis technology is required.
2. The formulation of Signature is too difficult to make, and its defined conditions are too general. Not only the data packets generated by security events meet these conditions, but also a large number of data packets generated by normal network activities meets. Therefore, a large number of false alerts will be generated, and the false alert rate can reach 99%. At this time, false alert detection technology is required, which can greatly improve the efficiency and timeliness of the correlation algorithm.

In solving these problems, machine learning algorithms play an important role, Especially recently, for problems that were considered unsolvable in the past, deep learning has shown incredible performance, making people to see its potential, increasing the reliability of its application in artificial intelligence and unsupervised learning(Vinayakumar et al., 2018; Amrita Dahiya & Brij B. Gupta, 2021; Sedik et al., 2021; Mishra et al., 2021; Vinayakuar et al., 2017; M. Shafiq et al., 2021; M. Shafiq et al., 2020; Adat & Gupta, B. B., 2018). Deep learning imitates the function of the human brain and is part of machine learning. In recent years, it has been applied in many security use cases, such as (Vinayakumar et al., 2017; M. Li et al., 2019; Z. Tian et al., 2020; Shudong Li et al., 2020; C. Luo et al., 2021; Esposito et al., 2021; Hangfeng Yang et al., 2020). This paper uses real alert records, and uses one method of combining deep learning and machine learning to conduct alert false alert detection. Using the features extracted from the hidden

layers of deep learning, as training data, input into the traditional machine learning model for training, the authors found that compared with direct training, the classification detection effect has been significantly improved, among the choices of traditional machine learning methods, integrated algorithms perform particularly well. So, this article recommends using the features of DNN hidden layer output to train traditional machine learning ensemble model random forest, AdaBoost and other methods in false alert detection. In the process of solving such problems in the future, trying a variety of combinations of multiple models is an effective way of solving problems or improving efficiency.

Section 1 explains the background, concept, and significance of the research; Next, Section 2 gives relevant research in the past; Section 3 briefly shows the proposed method in this article; Section 4 elaborates on the experimental process and results; Section 5 summarizes the results of the experiment and prospects for future research.

RELATED WORKS

Intrusion detection system, has been developed for more than forty years: In 1980, P. Anderson proposed the concept of intrusion detection in literature (Anderson J P et al., 1980). In 1987, Denning proposed a standard IDS model, In the 1990s, a large number of mature commercial and scientific research products appeared, and alert analysis has become a hot spot among related research since then.

T. Pietraszek analyzed some of the reasons for false alerts in the literature (T. Pietraszek & A. Tanner, 2005): Time limit; Signature limitations; dependence on the environment; false alerts generated by normal activities; false alerts generated by non-threatening attacks. For the two problems of alert analysis mentioned in section 1, the IDS in the actual large-scale network, because of the very large background traffic, the number of false alerts generated by the first problem is also very large, so a real-time and efficient false alert detection method is required; In a small network, the second type of problem causes more false alerts, relatively Less quantity, needs more accurate false alert detection methods. Introduce some existing methods of false alert analysis.

Data Mining and Machine Learning Methods

The analysis of data mining can determine which alert is a false alert, is a relatively accurate false alert detection method (Jidong Long & Daniel Schwartz, 2002). The relatively mature analysis is the root case and ALAC method from IBM Zurich laboratory (J. L. Hellerstein et al., 2002).

CLARAty is an offline analysis method. It analyzes existing alert data, and discovers rules that generate false alerts through aggregation and combination of personal analysis and network environment. The aggregation of alerts is based on a directed acyclic graph (Pang-Ning Tan et al., 2006). The main steps of aggregation are: 1) Select the attributes to be aggregated; 2) All alerts are aggregated according to the parent node; 3) The aggregated alerts are uniformly displayed, and record the number of aggregated alerts; 4) Repeat the above steps until the number in 3) is greater than a set Threshold. Next, reset the false alert detection rules and rewrite the IDS rules to reduce the false alert rate.

ALAC analyzes the alerts in a dynamic way: First, artificially classify the generated alerts according to their own knowledge and experience (make labels), and use it as training samples. Then use the RIPPER algorithm to train the two-classification model (Cohen William W, 1995), according to the evaluation results to adapt to the actual network security environment, set the alert classifier according to the extracted rules, evaluate the results, and then modify the corresponding rules.

CLARAty discovers that the toot causes that generates the alert causes a relatively high cost, and the effect of aggregation is also questionable; ALAC is based on manual correct classification and requires a lot of manual intervention. The RIPPER algorithm is used to dynamically learn the rules of fuzzy sets, as the amount of sample data increases, its complexity increases exponentially.

Statistics-Based Method

Compared with machine learning and data mining, the most prominent advantage of statistical methods is high efficiency, and the most prominent disadvantage is false negatives and false positives. Statistics are faced with data. It cannot accurately know which alert is a false alert, On the other hand, attacks can be discovered through statistical anomalies, once the pattern is discovered, a large number of false alerts can be removed quickly.

In literature (Hideki Koike & Kazuhiro Ohno, 2004), introduced an alert visualization analysis method. It displays the alerts generated by snort in the form of two-dimensional graphics. The horizontal axis represents time, the vertical axis categorizes the generated alerts according to the source IP. Moreover, the protocol type and threat degree of the alert are also marked differently in the figure.

In addition, in literature (NURBOL, 2010), mentioned the false alert filtering algorithm based on hidden Markov and conditional random field.

Time Series Method

The time series method discovers the law according to the time characteristics of alert generation, that is, the time series method, and analyzes the alerts generated by IDS.

Jouni Viinikka and others set up three sensors to record alerts: one collects alerts on the Internet, and two collects alerts on the protected network (Jouni Viinikka & Aristides Vagelatos, 2006). After 42 days, 68% of the alerts were generated, then the alerts were clustered according to the Signature and analysis rules. Take the number of alerts generated per hour as a variable. Suppose the number of alerts generated at time t is X_t, according to time series analysis, it is broken down into three parts: Trend items T_t, periodic items T_t, and random items T_t, that is:

$$X_t = T_t + P_t + R_t \tag{2.1}$$

As discussed by Smith Chen Xingshu (Chen Xingshu et al., 2019), which is based on the method proposed by Li Dong et al. (Li Dong et al., 2009), to process periodic false alert data. For the preprocessed alert log, it is similar to the previous method, Count the number of alerts generated by each triplet of $\langle srcIP, dstIP, attactType \rangle$ in the unit of hour, thereby obtaining the time series of the triplet. Use FFT to convert the sequence diagram to frequency diagram, calculate the alert period, and use the Pearson correlation coefficient to verify the correctness of the period value calculated after FFT. The above two statistical methods are suitable for fast and efficient alert analysis, but although the method is high-speed, the quality of false alert detection is not high enough. Rules can both generate real alerts and false alerts, a large number of false alerts triggered by certain rules will bring a sharp increase in security costs. Therefore, this paper chooses to use deep learning and traditional machine learning methods for false alert detection.

OUR METHOD

Data Set

The data set the authors used is the data provided by the West Lake Lunjian AI Big Data Security Analysis Competition, Alert False Alert Detection in October 2020.

Background of the competition: Due to the complexity of the network environment and the massive amount of data, most security products are faced with a high false alert rate. These inevitable false alerts will not only consume a certain number of resources and time for processing, but also reduce the sensitivity of security analysts to alerts and distract them from dealing with real security threats.

Challenge task: The log data provided in this competition is a traffic log that generates an alert in a security product, and some of the data is marked according to whether it is a false alert. Please perform analysis and model training based on the provided marked training samples, so that the model can correctly distinguish between normal alerts and false alerts from the test samples, and submit prediction results, related codes and documentation as required.

The training data has a total of 10396 alert records, each of which has 36 features and a list of label values, and the test data has a total of 15,978 alert records.

Basic Model Selection

Before introducing our own method, I will briefly introduce the basic model for improvement and the selection of activation function.

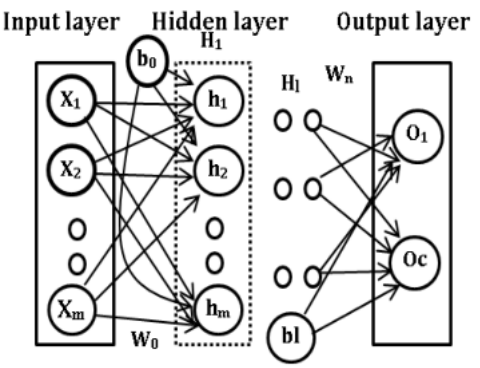
First understand the various components of DNN including:

- **Fully connected layer:** The units in this layer are connected to every unit in the subsequent layers, so it is called a fully connected layer. Generally, it maps data into higher dimensions. When the data have more dimensions, the output will be more accurate. About non-linear activation function the authors use ReLU.
- **Batch normalization and regularization:** In order to accelerate the training of the DNN model, for avoiding overfitting, Dropout: 0.01, and Batch normalization are used between fully connected layers (Shudong Li et al., 2021). Dropout will randomly remove neurons with connections.
- **Classification:** Fully connected layer is the last layer is, which uses the sigmoid for binary classification and multi-classification.

The method (DNNs) of stacking hidden layers, the depth and width of the neural network respectively refer to the number of neurons defined by the number of hidden layers. Figure 1 shows a DNN with a hidden layer, which input is $x = x_1$, x_2, $\cdots, x_{m-1}$, x_m, and output: $o = o_1$, o_2, $\cdots, o_{c-1}$, o_c.

The above figure only shows the units and connections of hidden layers partially. Compared with the classic FFN, DNNs is a more advanced model, because each hidden layer uses the nonlinear activation function ReLU, ReLU is the main breakthrough in the history of neural networks to reduce the disappearance and explosion gradient problems. When the gradient of the lower layer of the DNN is almost zero, because the unit of the upper layer is close to saturation at the asymptotic line of the tangent function, the gradient disappears. ReLU provides a non-linear alternative to sigmoid and tangent function. It is considered to be the most effective method for training large amounts of data in terms of time and cost. Its mathematical definition is as follows:

Figure 1. Architecture of a DNN



$$f(x) = max(0, x) \tag{3.1}$$

The characteristics of this method are reflected in the modeling of ReLU function to effectively optimize the deep learning model.

Our Specific Methods

Inspired by many machine learning models based on combinatorial optimization (Prabhat Kumar et al., 2021; John Sarivougioukas et al., 2020; Somya Ranjan Sahoo & Brij B. Gupta, 2021; Ivan Letteri et al., 2019). The authors put forward our own method: deep learning and machine learning combination optimization method, the specific explanation is as follows.

Generally speaking, for multiple hidden layers, the MLP formula is as follows:

$$H(x) = H_I\left(H_{I-1}\left(H_{I-2}\left(\cdots\left(H_1\left(x\right)\right)\right)\right)\right) \tag{3.2}$$

We all know that after the training of each layer of the deep neural network, the data input to the next layer obviously has its practical meaning, but this meaning is usually difficult to be easily understood by humans,That is, each H(x) in formula 3 . The output of each intermediate layer is a deeper expression of the input features of the previous layer. We can design some methods to output the features extracted by each hidden layer, and then use it in traditional machine learning to see if it will improve.

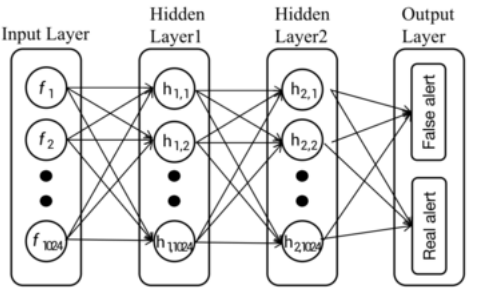
In the following experiment, the feature dimension after coding is 567. (The next part will explain in detail)

Determine network parameters: over-tune the parameters to find the optimal parameter set for achieving the best result (there is a lot of room for future research) In this paper, when other parameters are optimized, the learning is kept at 0.01. Experiment by changing the number of neurons in a layer from 2 to 1024. Then, the count increased to 1280, but the accuracy did not increase. Therefore, the number of neurons is adjusted to 1024.

Build the network structure: Generally, increasing the number of layers is better than increasing the number of neurons in a layer. Therefore, the authors used the network topology introduced below to check and summarize the best network structure. DNN sets 1-5 layers respectively. Figure 2 shows the DNNs architecture used, including hidden layer and output layer.

The input layer has 567 neurons. The back propagation mechanism of the training process is used to train the DNN network. The input layer consists of 567 neurons. Then enter the hidden layer. The hidden layer uses ReLU as a nonlinear activation function. Then add the weight and forward it to the

Figure 2. Network structure



next hidden layer. In order to make the results more accurate and lower cost, from the input layer to the output layer, the number of neurons in each layer gradually decreases steadily. Regularization: The time cost of the entire process is reduced and reduced: 0.01. The outer layer contains only two neurons: true alert and false alert. Because it is a two-class classification, 1024 neurons in the previous layer must be converted into 2, so it only returns two outputs. Therefore, the sigmoid function is used for the last two classifications.

In the process of training the model, the method of combining deep learning and machine learning models is used, that is, the features of a hidden layer output of the DNN model are used as input to train the machine learning model. In order to show the specific improvement details of the method, Figure 3 shows the process of using this data to train a common machine learning model and Figure 4 shows the our method process.

The figure shows that the original features are input to the DNN model, and the authors select a layer from the 1-5 hidden layers to output the features obtained by its training, and n represents the dimension of the feature (the feature dimension of each layer is different). Input the obtained features as training data into the machine learning model for training, and finally get the classification result.

Figure 3. Normal machine learning model flow

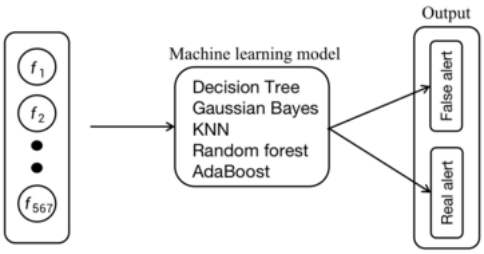
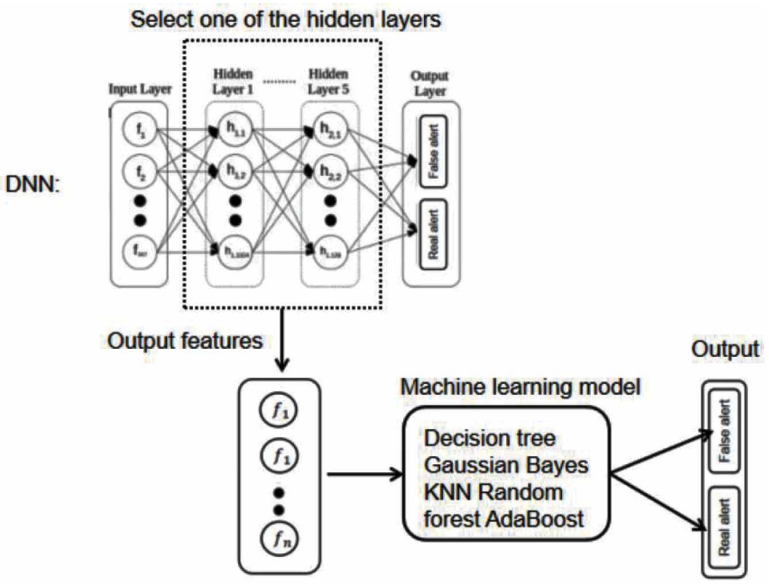


Figure 4. Our method flow



The authors use all the hidden layers to do the above experiment. The figure above is a flow chart of one of the experiments. Repeat the experiment many times to find the best solution.

EXPERIMENT RESULTS

Data Preprocessing

Data description: Table 1 shows the field description of the data provided in this question.

The specific data is stored in train.csv as the training data, with a label, and test_1.csv is the test data distributed to the contestants. The format is the same as the training data except that there is no label field. The amount of data comparison of black and white samples of training data is shown in figure 5.

Perform feature engineering before training the model. The first step of feature engineering is to manually view the specific data content and distribution of the data. This data belongs to structured data. The structured data can be regarded as a table of a relational database. Each column has a clear definition and contains two basic types: numeric and categorical; each row of data represents information about a sample.

Table 2 shows the number of missing samples for each feature.

Viewing the proportion of the number of types of each characteristic value and the data value of each category, observing the fluctuation trend of the data range, etc., helps to determine which model to use in the future. Figure 6 gives an example of part of the data describing the numerical distribution.

After analyzing each feature of the data in detail, analyze it in conjunction with the detailed description of the data field, remove meaningless features, such as eventId, name, etc; remove features with too many missing values, such as srcGeoCity, srcGeoAddress, srcGeoLatitude, srcGeoLongitude, etc.

Statistical Measures

The quality of the model in the experiment needs to be tested and evaluated. This section introduces some evaluation standards used in this experiment. Use the following definitions to evaluate the model:

True Positive (TP): The amount of all alerts that are classified correctly into the correct alert category.
True Negative (TN): The amount of all alerts that are classified correctly into the false alert category.
False Positive (FP): The amount of all correct alerts that are classified correctly as false alerts.
False Negative (FN): The amount of false alerts that are classified incorrectly as correct alerts.

Based on the above terminology, consider the following most commonly used evaluation indicators:

- **Accuracy:** It calculates the percentage of correctly identified data in the entire test set. The higher the better, the calculation method is as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

- **Recall rate:** It calculates how many positive examples in the sample are predicted correctly. The higher the recall rate, the better the machine learning model. The calculation method is as follows:

$$Recall = \frac{TP}{TP + FP} \tag{4.2}$$

Table 1 Detailed description of data fields

Standard field	Meaning	Sample
eventId	eventID	5234650076632393473
startTime	start time	***11:11:10
transProtocol	Transfer Protocol	TCP
appProtocol	Application protocol	http
name	Event name	HTTP request access
srcAddress	Source address	*.11.40.*
srcPort	Source port	54478
destAddress	Destination address	*.172.47.*
destPort	Destination port	3478
srcGeoCity	Source city	Hangzhou
srcGeoAddress	Source address	111 Tianmushan Road, Hangzhou City, Zhejiang Province
srcGeoLatitude	Source latitude	120.144061
srcGeoLongitude	Source longitude	30.289935
destGeoCountry	destination country	China
destGeoRegion	Destination area	Zhejiang
destGeoCity	Destination city	Hangzhou
destGeoAddress	Destination address	111 Tianmushan Road, Hangzhou City, Zhejiang Province
destGeoLatitude	Destination latitude	120.144061
destGeoLongitude	Longitude of destination	30.289935
catOutcome	Event result classification	Attempt
txId	Protocol communication ID	0
bytesOut	Number of bytes sent to the client	0
bytesIn	The number of bytes sent to the server	34
destHostName	HTTP request domain name	*.163.21.*
requestMethod	HTTP request method	GET
httpVersion	HTTP protocol version	HTTP/1.1
requestUrlQuery	HTTP request URI	/fwlink/?LinkID=252669&clcid=0x409
requestUrl	HTTP request URL	/fwlink/
httpReferer	HTTPRefer	https://www.xcar.com.cn/
accessAgent	HTTPUserAgent	MicroMessenger Client
responseCode	HTTP response code	200
requestBody	HTTP request body	<?php%20echo%20"Anonymous_".".woopra_test";%20?>'
requestHeader	Request header	Connection: Keep-Alive Accept-Language: zh-CN User-Agent: 4176 Host: 10.20.120.136
responseHeader	Response header	Server: nginx/1.14.0 Date: Thu, 09 Jul 2020 07:36:04 GMT Content-Type: application/json;charset=UTF-8 Transfer-Encoding: chunked Connection: keep-alive Pragma: no-cache Cache-Control: no-cache Expires: Thu, 01 Jan 1970 00:00:00 GMT
requestContentType	HTTP request ContentType	application/octet-stream
responseContentType	HTTP response ContentType	application/jason
label	Whether it is false positive	0 means normal alert, 1 means false alert

Figure 5. Comparison of black and white samples

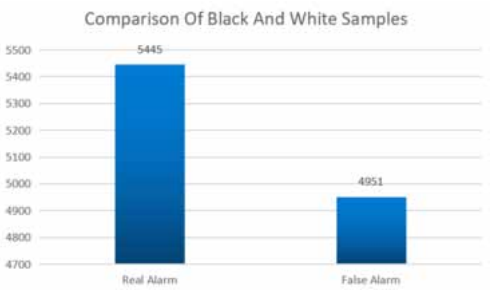


Table 2. Missing number for all features

Feature Missing number	Feature Missing number
eventId 0	destGeoLongitude 3224
startTime 0	catOutcome 122
transProtocol 0	txId 0
appProtocol 0	bytesIn 0
name 0	destHostName 12
srcAddress 0	requestMethod 0
srcPort 0	httpVersion 9
destAddress 0	requestUrlQuery 0
destPort 0	requestUrl 0
srcGeoCity 10396	httpReferer 5505
srcGeoAddress 10396	accessAgent 378
srcGeoLatitude 10396	responseCode 122
srcGeoLongitude 10396	requestBody 3722
destGeoCountry 6	requestHeader 8
destGeoRegion 6	responseHeader 140
destGeoCity 3260	requestContentType 5709
destGeoAddress 10396	responseContentType 712
destGeoLatitude 3224	label 0

Figure 6. Partial data value distribution description

	eventId	srcPort	destPort	destGeoLatitude	destGeoLongitude	txId	bytesOut	bytesIn	responseCode	label
count	1.039600e+04	10396.000000	10396.000000	7172.000000	7172.000000	10396.000000	1.039600e+04	1.039600e+04	10274.000000	10396.000000
mean	2.917163e+17	46243.746345	2773.468449	118.124423	32.873783	9.713351	1.504173e+04	6.229609e+04	270.348647	0.476241
std	1.728837e+15	17950.697864	6647.386896	11.286796	4.527958	80.852490	2.345500e+05	5.237128e+06	102.572317	0.499459
min	2.885701e+17	1028.000000	80.000000	-122.329437	1.352083	0.000000	0.000000e+00	0.000000e+00	101.000000	0.000000
25%	2.919396e+17	40202.250000	80.000000	116.405285	30.287459	0.000000	2.590000e+02	4.790000e+02	200.000000	0.000000
50%	2.926673e+17	52954.000000	80.000000	120.153576	31.231706	0.000000	3.365000e+02	7.570000e+02	200.000000	0.000000
75%	2.926788e+17	59120.750000	80.000000	120.153576	36.094406	3.000000	7.140000e+02	1.379000e+03	404.000000	1.000000
max	2.937332e+17	65532.000000	52323.000000	139.731993	53.349805	4704.000000	1.048642e+07	5.311278e+08	503.000000	1.000000

- **Precision:** It calculates the proportion of the number of correctly identified alarms to the total number of alarms, the higher the better. The calculation method is as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.3}$$

- **F1-Score:** Also called F1-Measure. The higher the F1 score, the better the model. The calculation method is as follows:

$$F1 - score = 2 * \left(\frac{Precision * Recall}{Precision + Recall} \right) \tag{4.4}$$

Experiment Procedure

The first step is data preprocessing. According to the analysis, some features are selected for dummy coding. The basic idea of one-hot: Treat each value of discrete elements as a state. Suppose the elements have N different values, then the authors can abstract the feature into N types for different states, one-hot encoding ensures that only one of the N states has a value of 1, and the other state bits are all 0. One-of-K is used to convert all classification features into binary features. Requirements for a hot encoding: "The input to this converter should be an integer matrix, representing the value obtained by the classification (discrete) feature. The output will be a sparse matrix, where each column corresponds to a possible value of a feature. The intuitive explanation of variable coding is to remove a status bit arbitrarily. The feature dimension after coding is 567.

Divide the encoded data into training and test data sets, the test sample accounts for 30%, and the random number seed is 0, which is the number of the group of random numbers. When the experiment needs to be repeated, the random number obtained is different each time.

The traditional machine learning methods used in this experiment include decision trees, Gaussian Bayes, K-Nearest neighbors, random forests, and AdaBoost. Then the authors compare our method with these traditional machine learning methods.

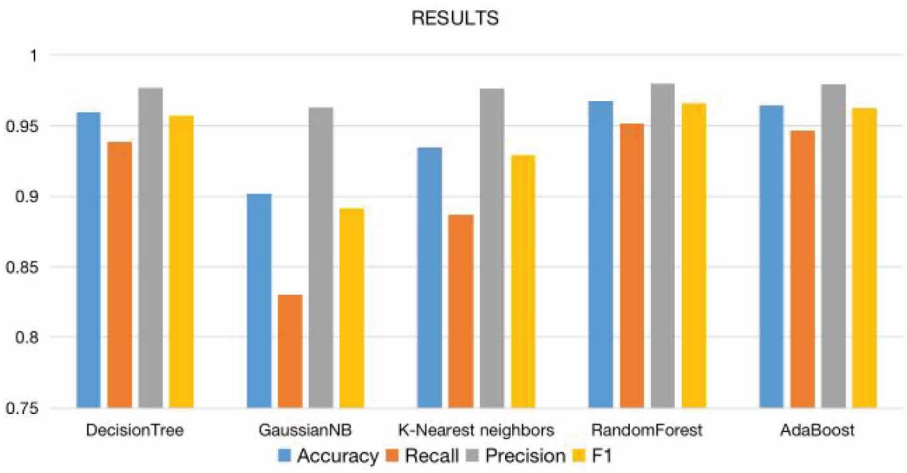
Train Traditional Machine Learning Models

Use the divided training set to train the traditional machine learning model decision tree, Gaussian Bayes, KNN, random forest, AdaBoost, and then use the test set to evaluate the model effect. The evaluation results are shown in Table 3 and Figure 7.

Table 3. Traditional machine learning model evaluation results

Model	Accuracy	Recall	Precision	F1
DecisionTree	0.959281821	0.938451357	0.976584022	0.957138036
GaussianNB	0.901891632	0.829913964	0.962394474	0.891257996
K-Nearest neighbors	0.934594421	0.886829914	0.97596504	0.92926491
RandomForest	0.966976595	0.951687624	0.979564033	0.965424639
AdaBoost	0.964091055	0.946393117	0.978781656	0.962314939

Figure 7. Traditional machine learning model evaluation results



Training DNN Model

The authors use Keras (F. Chollet, 2017; Abhinaya Nagisetty & Govind P. Gupta, 2019), as a wrapper and Tensorflow as the software framework (M. Abadi et al., 2016). In order to exponentially increase the agility of data processing in the deep learning architecture.

The divided training set is used for the training of dnn model with 1-5 hidden layers, and the test set is used for evaluation. The final evaluation results are shown in Table 4 and Figure 8. The results show that the accuracy of the DNN5 layer is the highest, which is better than other layers. DNN4s has the lowest loss rate.

Train our Method Model

From the above two evaluation results, we can see that the dnn model is not completely better than the ordinary machine learning model. There are some traditional machine learning models that perform better. We all know that after the training of each layer of the dnn, the data input to the next layer obviously has its practical meaning, but this meaning is usually difficult to be easily understood by humans. The output of each intermediate layer is a deeper expression of the input features of the previous layer. We can design some methods to output the features extracted by each hidden layer, and then use it in traditional machine learning to see if it will improve. The features extracted from each layer are used as the training set to train the traditional machine learning model and evaluate the results. The results are shown in Figure 9, Figure10, Figure 11 and Figure12.

Table 4. Deep learning model evaluation results

Model	Accuracy	Loss
DNN-1	0.9239	0.2323
DNN-2	0.9647	0.1163
DNN-3	0.9612	0.1163
DNN-4	0.9667	0.1091
DNN-5	0.9670	0.1293

Figure 8. Deep learning model evaluation results

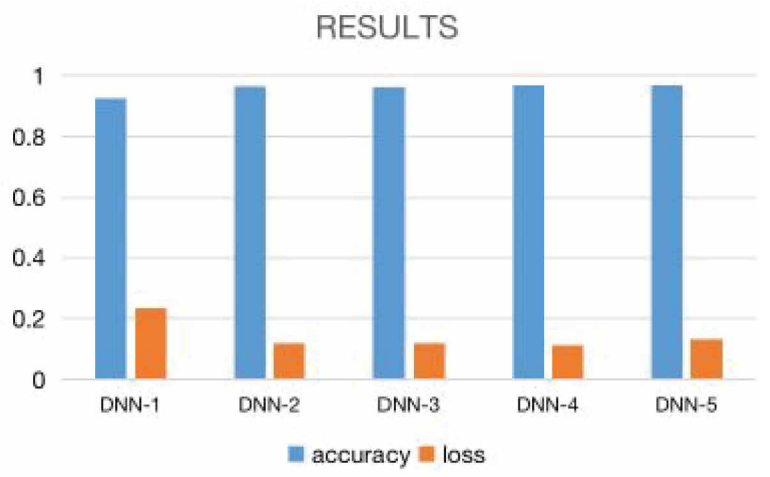
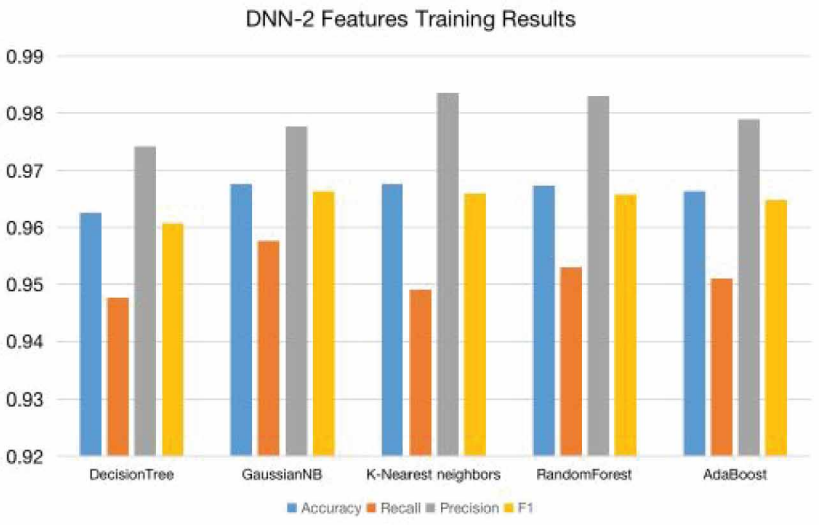


Figure 9. Model fusion evaluation results (DNN-2)



According to the results, it is found that the effects of all machine learning models have been improved overall. In order to be more intuitive, draw a line chart as shown in Figure 13, Figure14, Figure15, Figure16 and Figure 17. The light blue line represents the effect of the original training feature training, and the other lines represent the effect of the combination of deep learning and machine learning.

Improve Feature Engineering

At the end of the experiment, I tried to improve the feature engineering and try not to discard any feature that might be meaningful for training. Combined with the detailed description of the data fields, the following processing is done for each feature:

Figure 10. Model fusion evaluation results (DNN-3)

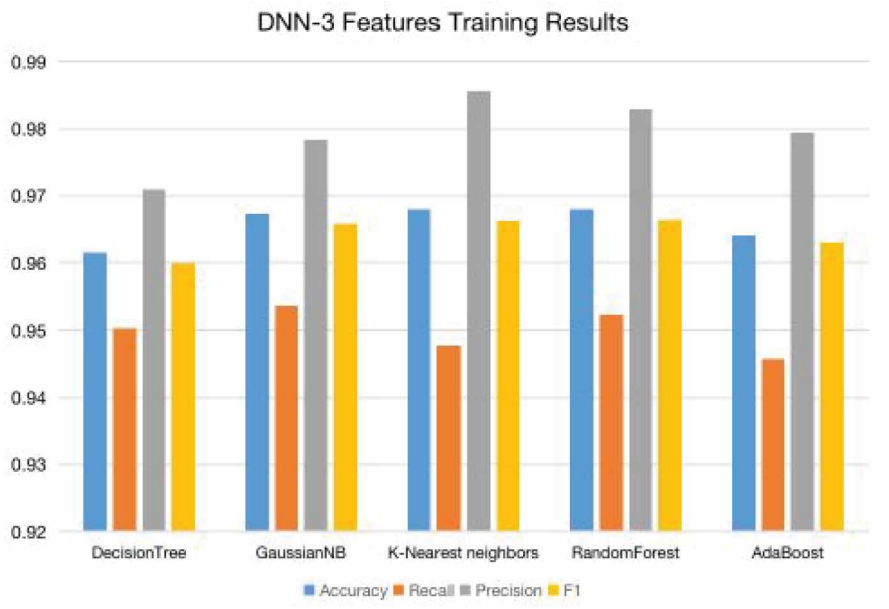
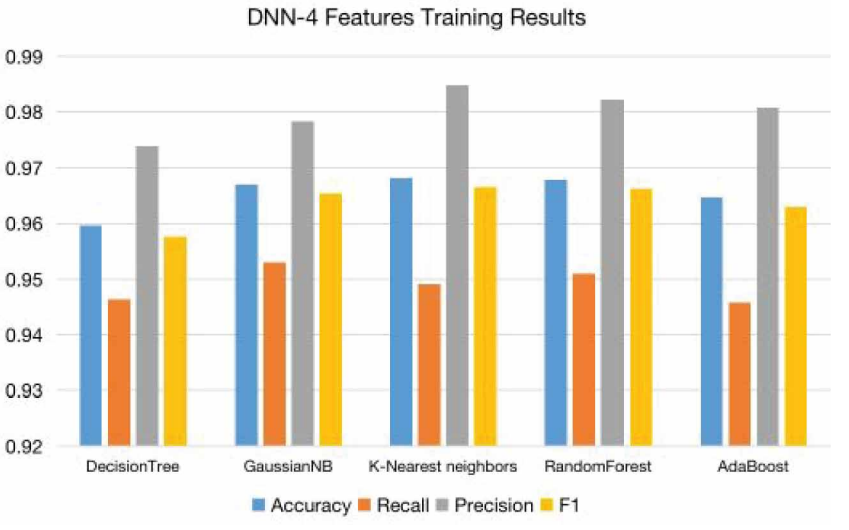


Figure 11. Model fusion evaluation results (DNN-4)



"srcAddress", "estAddress",Extract the pure digital port number; "requestHeader", "responseHeader", "requestUrl","httpReferer", "requestBody" extract its length characteristics; "catOutcome" encodes dummy variables; "txId", "bytesOut", "bytesIn", "destGeoLatitud", "destGeoLongitude", "srcPort", "destPort" are all numerical features and are used directly; "requestBody", "httpReferer", "requestContentType" these three features have too many missing values but still want to use the only data, so two matrices are formed respectively: missing value display 1, and non-missing value display 1, respectively; "requestUrlQuery", "requestUrl", "responseHeader",

Figure 12. Model fusion evaluation results (DNN-5)

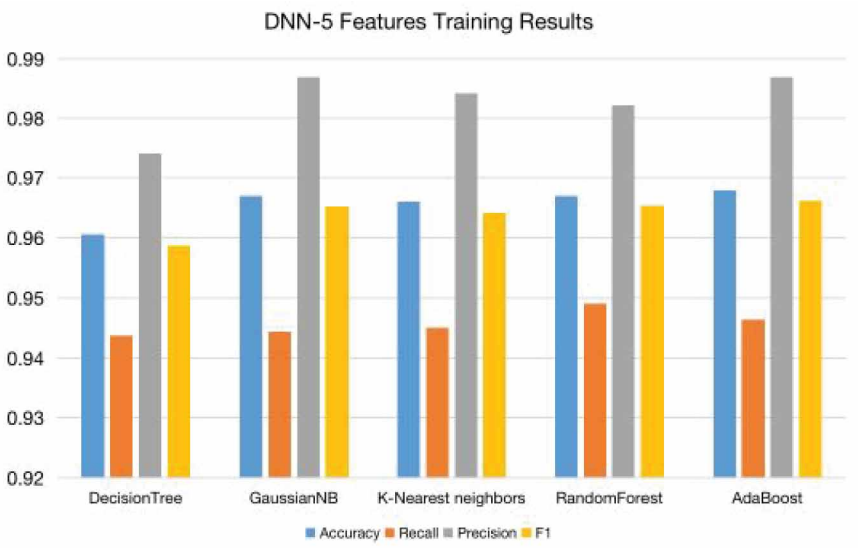
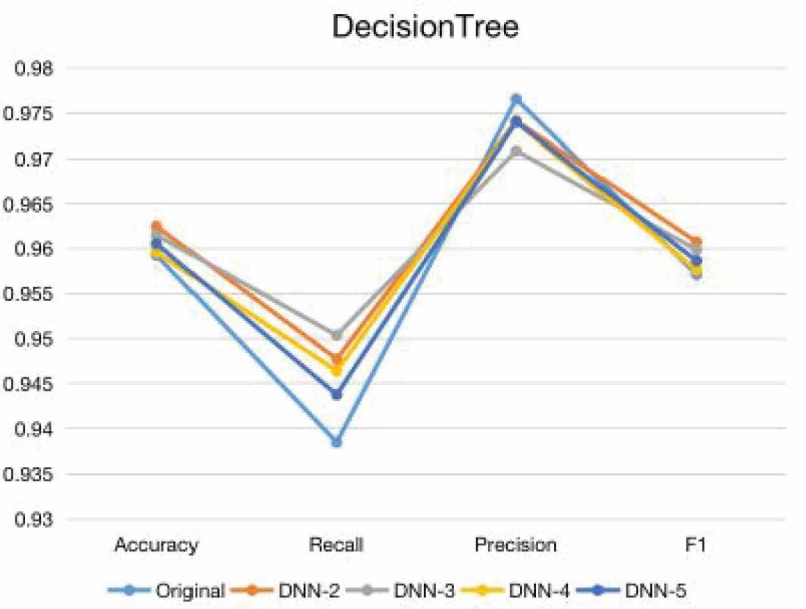


Figure 13. Comprehensive comparative analysis (Decision Tree)



"requestHeader" calculate the number of their special symbols; "startTime" time feature, first extract the pure digital value, and then set the judgment morningL: 0, afternoon: 1; Perform label encoding for "srcAddress", generate a dictionary and save it as a pkl file. The processed features are standardized (normalized), and then use the RFE method for feature selection, the final feature dimension is 41. Repeat the above B and C experiments. Finally got a higher evaluation result, F1 value is as high as 0.9867, but unfortunately the accuracy of DNN is only about 0.6.

Figure 14. Comprehensive comparative analysis (Gaussian Bayes)

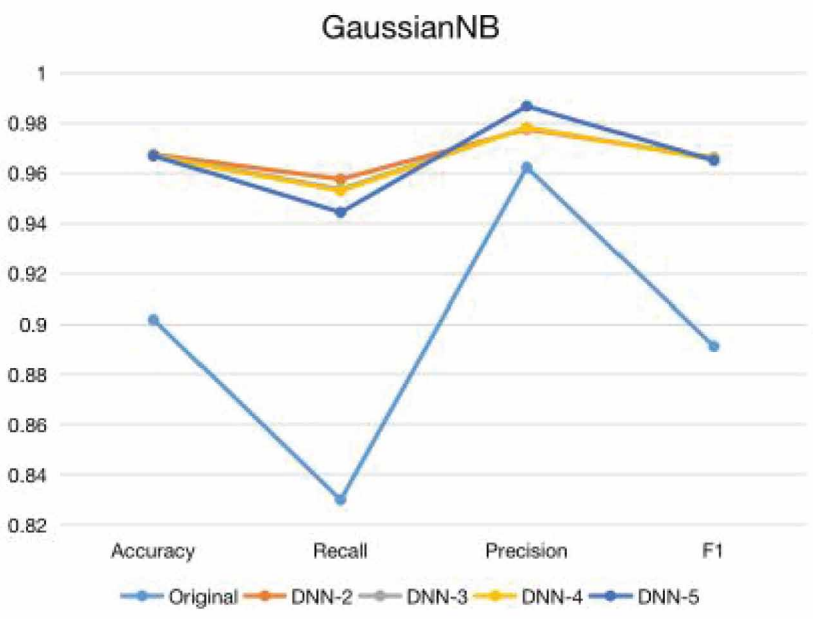
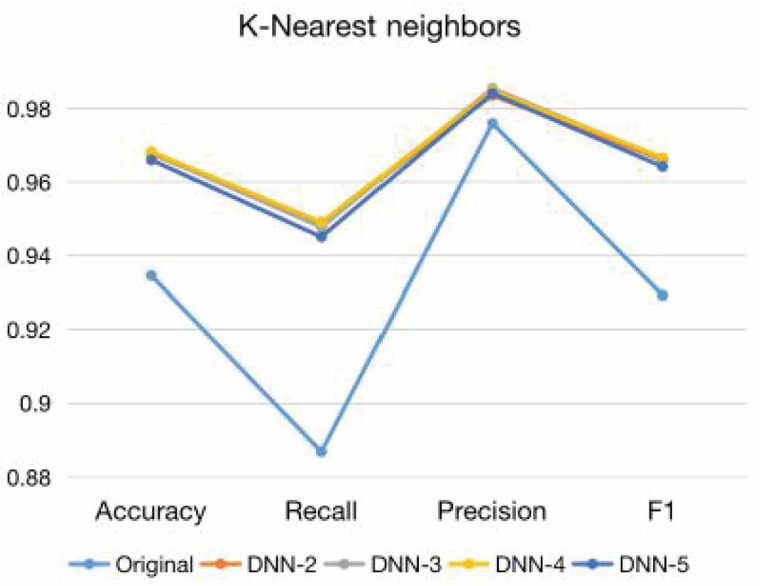


Figure 15. Comprehensive comparative analysis (KNN)



CONCLUSION

This paper uses real alert records, and uses traditional machine learning model like decision trees, Gaussian Bayes, KNN, random forest, AdaBoost, DNN, and the method of combining deep learning and machine learning to conduct alert false alert detection experiments. According to the above

Figure 16. Comprehensive comparative analysis (Random Forest)

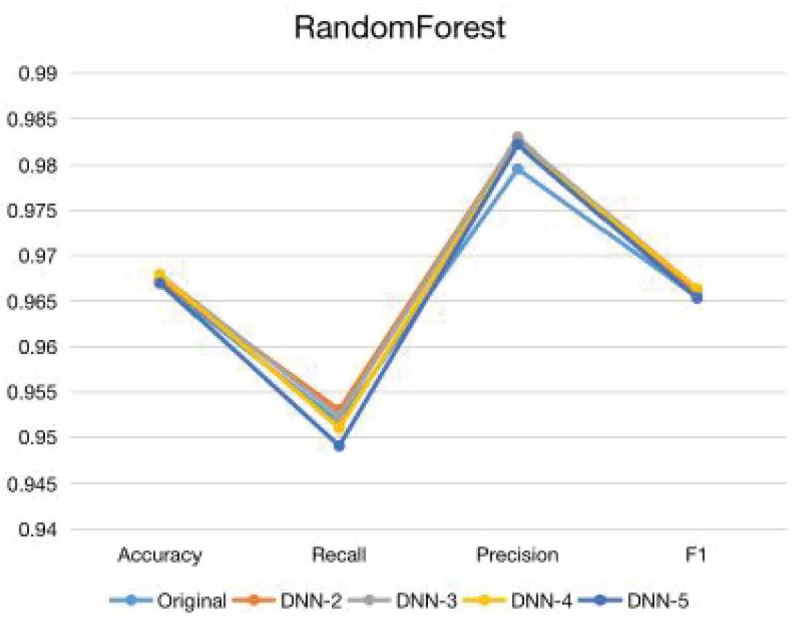
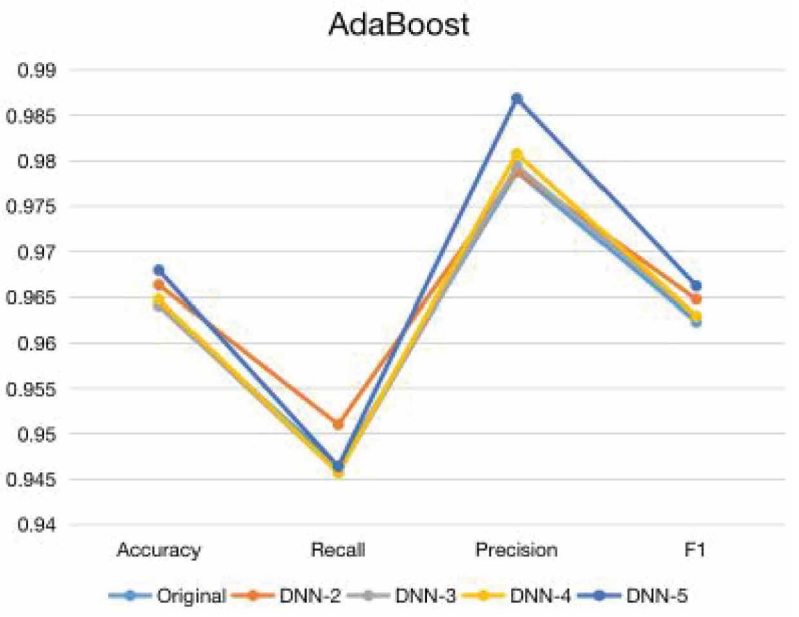


Figure 17. Comprehensive comparative analysis (AdaBoost)



experiments, it can be seen that machine learning and deep learning have relatively high accuracy in detecting false alerts; Using the features extracted from the intermediate hidden layer of deep learning, as training data, input into the traditional machine learning model for training, the authors found that compared with direct training, the classification detection effect has been significantly improved.

Therefore, this experiment proves that it is in the detection of false alerts. Deep learning can improve the classification effect of traditional machine learning models, so this article recommends using the features of DNN hidden layer output to train traditional machine learning ensemble model random forest, AdaBoost and other methods for false alert detection. Because the data is real alert data, the above method is also feasible in a real environment.

According to the above conclusions, in the process of solving such problems in the future, trying a variety of combinations of multiple models is an effective way of solving problems or improving efficiency. However, when I improved the feature engineering, the effect of the final classification detection was also improved, but because the feature dimension was greatly reduced (for guessing reasons), the deep learning model did not perform well, so I did not continue the combined training. The exploration of the specific essential reasons is also a direction worthy of continued research. The analysis of the alarm in the Internet of things to promote the development of the security of the Internet of animals can make people more trust the Internet of things, so as to accelerate its development.

Data Availability

The dataset is available at Flse-Alert-Detecton (https://github.com/asterism-q/Flse-Alert-Detecton).

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Funding Statement

This research was funded by NSFC (No.62072131), Key R&D Program of Guangdong Province (No. 2019B010136003), NSFC (61972106, 61702552), Science and Technology Projects in Guangzhou (No.202102010442), and National Key Research and Development Program of China (No.2019QY1406), Guangdong Higher Education Innovation Group (No.2020KCXTD007), Guangzhou Higher Education Innovation Group (No.202032854).

REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., & Isard, M. (2016). Tensorflow: A system for large-scale machine learning. OSDI, 16, 265-283.

Adat, V., & Gupta, B. B. (2018). Security in Internet of Things: Issues, challenges, taxonomy,and architecture. *Telecommunication Systems*, *67*(3), 423–441. doi:10.1007/s11235-017-0345-9

Anderson, J. P. (1980). *Computer security threat monitoring and surveillance. Technical Report*.

Chen, X., Tao, H., & Zeng, X. (2019). Research on attack scene assoc-iation rule mining method based on alert attributes clustering. *Advanced Engineering Sciences*, *51*(3), 144–150.

Cohen, W. (1995). Fast effective ru-le induction. In *Proceedings of the 12th international conference on machine learning*. Tahoe City, CA: Morgan Kaufmann.

Dahiya, A., & Gupta, B. B. (2021). A reputation score policy and Bayesian game theory based incentivized mechanism for DDoS attacks mitigation and cyber defense. *Future Generation Computer Systems*, *117*, 193–204. doi:10.1016/j.future.2020.11.027

Dong, L., Li, Z., & Jie, L. (2009). Research on the method of reducing false positives with periodicity. *Journal of Chinese Computer Systems*, *30*(7), 1336–1340.

Esposito, C., Ficco, M., & Gupta, B. B. (2021). Blockchain-based authentication and authorization for smart city applications. *Information Processing & Management*, *58*(2), 102468. doi:10.1016/j.ipm.2020.102468

Hellerstein, J. L., Ma, S., & Perng, C. S. (2002). Discovering actionable p-atterns in event data. *IBM Systems Journal*, *41*(3), 459–475. doi:10.1147/sj.413.0475

Koike, H., & Ohno, K. (2004). Snort View: Visualization System of Snort Logs. *Proc. of Workshop on Visualization for Computer Security*, 143-147.

Kumar, Gupta, & Tripathi. (2021). An ensemble learning and fog-cloud architecture-driven cyber-attack detection framework for IoMT networks. *Computer Communications, 166*.

Letteri, I., Della Penna, G., & De Gasperis, G. (2019). Security in the internet of things: Botnet detection in software-defined networks by deep learning techniques. *International Journal of High Performance Computing and Networking*, *15*(3/4), 170–182. doi:10.1504/IJHPCN.2019.106095

Li, M., Sun, Y., Lu, H., Maharjan, S., & Tian, Z. (2020, July). Deep Reinforcement Learning for Partially Observable Data Poisoning Attack in Crowdsensing Systems. *IEEE Internet of Things Journal*, *7*(7), 6266–6278. doi:10.1109/JIOT.2019.2962914

Li, S., Jiang, L., Wu, X., Han, W., Zhao, D., & Wang, Z. (2021). A Weighted Network Community Detection Algorithm Based on Deep Learning. *Applied Mathematics and Computation*, *401*, 126012. doi:10.1016/j.amc.2021.126012

Li, S., Li, Y., Han, W., Du, X., Guizani, M., & Tian, Z. (2021). Malicious mining code detection based on ensemble learning in cloud computing environment. *Simulation Modelling Practice and Theory*, *113*, 102391. doi:10.1016/j.simpat.2021.102391

Li, S., Zhang, Q., Wu, X., Han, W., & Tian, Z. (2021). Attribution classification method of APT malware in IoT using machine learning techniques. *Security and Communication Networks*, *2021*, 9396141. doi:10.1155/2021/9396141

Li, S., Zhao, D., Wu, X., Tian, Z., Li, A., & Wang, Z. (2020). Functional immunization of n-etworks based on message passing. *Applied Mathematics and Computation*, *366*, 124728. doi:10.1016/j.amc.2019.124728

Long, J. (2006). Daniel Schwartz. Disti-nguishing false from true alerts in Snort by data mining patterns of alerts. *Sara Stoecklin Proceedings of SPIE*, *6241*, 99–108.

Luo, C., Tan, Z., Min, G., Gan, J., Shi, W., & Tian, Z. (2021). A Novel Web Attack Detection Systemfor Internet of Things via Ensemble Classification. *IEEE Transactions on Industrial Informatics*, *17*(8), 5810–5818. Advance online publication. doi:10.1109/TII.2020.3038761

Mishra, A., Gupta, N., & Gupta, B. B. (2021). Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller. *Telecommunication Systems*, *77*(1), 1–16. doi:10.1007/s11235-020-00747-w

Nagisetty, A., & Gupta, G. P. (2019). Framework for detection of malicious activities in IoT networks using Keras deep learning library. *3rd International Conference on Computing Methodologies and Communication (ICCMC)*. doi:10.1109/ICCMC.2019.8819688

NURBOL. (2010). Research on Anomaly Detection Based on Data Mining and Multistage Intrusion Alert Correlation. College of Computer Science and Technology, Jilin University.

Pietraszek, T., & Tanner, A. (2005). Data mining and machine learning Towards reducing false positives in intrusion detection. *Information Security Technical Report*, *10*(3), 169–183. doi:10.1016/j.istr.2005.07.001

Sahoo, S. R., & Gupta, B. B. (2021). Multiple features based approach for automatic fake news detection on social networks using deep learning. *Applied Soft Computing*, *100*, 106983. doi:10.1016/j.asoc.2020.106983

Sarivougioukas, J., & Vagelatos, A. (2020, July). Modeling deep learning neural networks With denotational mathematics in ubiHealth environment. *International Journal of Software Science and Computational Intelligence*, *12*(3), 14–27. doi:10.4018/IJSSCI.2020070102

Sedik, A., Hammad, M., Abd El-Samie, F. E., Gupta, B. B., & Abd El-Latif, A. A. (2021). Efficient deep learning approach for augmented detection of Coronavirus disease. *Neural Computing & Applications*, 1–18. doi:10.1007/s00521-020-05410-8 PMID:33487885

Shafiq, M., Tian, Z., Bashir, A., Du, X., & Guizani, M. (2020). IoT malicious traffic identification using wrapper-based feature selection mechanisms. *Computers & Security*, *94*, 101863. doi:10.1016/j.cose.2020.101863

Shafiq, M., Tian, Z., Bashir, A. K., Du, X., & Guizani, M. (2021). CorrAUC: A M-alicious BotIoT Traffic Detection Method in IoT Network Using Machi-ne Learning Techniques. *IEEE Internet of Things Journal*, *8*(5), 3242–3254. Advance online publication. doi:10.1109/JIOT.2020.3002255

Tan, P.-N. (2006). *Introduction to Data Mining.* People Post Press.

Tian, Z., Luo, C., Qiu, J., Du, X., & Guizani, M. (2020). A Distributed Deep Learning System for Web Attack Detection on Edge Devices. *IEEE Transactions on Industrial Informatics*, *16*(3), 1963–1971. doi:10.1109/TII.2019.2938778

Viinikka, J., & Herv'e, D. (2006). Time Series Modeling for IDS Alert Management. *Proceedings of the ACM Symposium on Information, Computer and Communications Security (ASIACCS'06)*.

Vinayakumar, R., Soman, K. P., Velan, K. S., & Ganorkar, S. (2017, September). Evaluating shallow and deep networks for ransomware detection and classification. In *Advances in Computing, Communications and Informatics(ICACCI), 2017 International Conference on* (pp. 259-265). IEEE.

Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017, September). Applying convolutional neural network for n-etwork intrusion detection. In *Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on* (pp.1222-1228). IEEE.

Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2018). Evaluating deep learning approaches to characterize and classify malicious URLs. *Journal of Intelligent & Fuzzy Systems*, *34*(3), 1333–1343. doi:10.3233/JIFS-169429

Yang, Li, Wu, Lu, & Han. (2017). A Novel Solution for Malicious Code Detection and Family Clustering based on Machine Learning. *IEEE Access, 7*(1), 148853-148860.

Shudong Li received M.S. degree in applied mathematics from Tongji University (China) in June 2005 and his Ph.D. degree from Beijing University of Posts and Telecommunications (China) in July 2012. From 2013-2018, He was the postdoc of National University of Defense Technology China. Now, he is associate professor in Cyberspace Institute of Advanced Technology, Guangzhou University, China. His current research includes Big Data and its security, malware identification, information security and cryptography, the robustness of complex networks. His awards include 2015 First prize of scientific and technological progress in Hunan province, China, and 2018 First prize of Chans Chinese information processing science and technology, China.

Danyi Qin is a Master's candidate and now studying at Guangzhou University.

Xiaobo Wu received her B.S. degree in computer science from Yantai Normal College (China) in June 2002, M.S. degree in computer science and technology from Dalian University of Technology (China) in June 2008. Her current research interest includes protocols analysis, information security and cryptography.

Li Juan graduated from Shantou University with a master's degree in computer software and theory in 2005. At present, I am engaged in the research of network security, threat monitoring and network security technology.

Baohui Li received his Ph.D. degree from Beijing University of Posts and Telecommunications (China) in July 2016. From 2016, He was associate professor in Chinese Information Technology Security Evaluation Center, China. His current research includes Cloud computing and its security, malware identification.

Weihong Han received her M.S. degree in National University of Defense Technology, China, in 1997 and her Ph.D. degree in National University of Defense Technology in 2000. She is Member of Cyber Security Association of China. From 2002 to 2017, she worked at National University of Defense Technology. Now, she is a professor in Cyberspace Institute of Advanced Technology, Guangzhou University, China. Her current research interest is computer networks and network security.