P-DNN: An Effective Intrusion Detection Method based on Pruning Deep Neural Network

Mingjian Lei, Xiaoyong Li, Binsi Cai, Yunfeng Li, Limengwei Liu, Wenping Kong
Key Laboratory of Trustworthy Distributed Computing and Service (BUPT), Ministry of Education
Beijing University of Posts and Telecommunications
Beijing, China
Email: {chnleimingjian, lxyxjtu, cbsbupt, liyfbupt, mwdoublel, kwenping}@163.com

Abstract—Today, the scale of global Internet users continues to grow; the Internet has become the main driver of global economic growth; IoT technology is also constantly pushing the process of the Internet of Everything. However, the ever-changing cybersecurity situation is not optimistic and the people’s demand for secure network is also increasing. In this paper, for the biggest challenge of building anomaly-based Network Intrusion Detection System: building a high-performance intrusion detection classifier model, we first propose an effective intrusion detection method based on pruning deep neural network: P-DNN. Firstly, we train a deep neural network with complex structure and good intrusion detection performance. Secondly, through the pruning operation, only the connections with more important information in the weight are reserved, reducing the complexity of the model. Finally, retrain the deep neural network to find the best model. We use the KDD Cup 99 dataset to evaluate the effectiveness of the method and achieve exciting results. The model constructed by P-DNN achieves a detection rate of 0.9904 for known attacks and a detection rate of 0.1050 for unknown attacks. By comparing with related work, the model achieves the best intrusion detection performance: COST is reduced to 0.1875 and ACC is increased to 0.9317.

Index Terms—pruning, deep neural network, intrusion detection

I. INTRODUCTION

In 2018, the number of Internet users worldwide had reached 3.8 billion, accounting for 51% of the global total [1]. As of December 2018, 7 of the 10 companies with the highest market capitalization were Internet technology companies, namely Microsoft, Amazon, Apple, Alphabet, Facebook, Alibaba, and Tencent [1]. By 2020, NB-IoT will achieve deep coverage in China for indoor, traffic network and underground pipe network; the base station scale will reach 1.5 million [2]. In 2018, National Internet Emergency Center handled about 106,000 network security incidents and captured more than 100 million computer malware samples [3].

Today, the scale of global Internet users continues to grow; the Internet has become the main driver of global economic growth; IoT technology is also constantly pushing the process of the Internet of Everything. However, the ever-changing cybersecurity situation is not optimistic and the people’s demand for secure network is also increasing to protect systems, services and data from unexpected threats. Since it was first proposed by Heberlein in 1991 [4], the Network Intrusion Detection System (NIDS) has received continuous attention from academia and industry. According to the detected technology, NIDS can be divided into signature-based NIDS and anomaly-based NIDS. Signature-based NIDS maintains a signature database, and compares the signature of the traffic with the signature database to determine whether it is normal traffic or attack traffic. Anomaly-based NIDS models normal and abnormal network traffic, and distinguishes normal traffic and abnormal traffic by calculating the similarity between the traffic and the model. Signature-based NIDS (such as the well-known open source NIDS: Snort [5] and Suricata [6]) has high detection rate when detecting known attacks. However, it cannot detect unknown attacks (0-day attacks and variants of attacks) that are not included in the signature database. Anomaly-based NIDS can overcome this weakness. On the basis of achieving a high detection rate for known attacks, anomaly-based NIDS can detect unknown attacks, increasing the probability of detecting high-risk attacks. In summary, anomaly-based NIDS has a greater possibility to build a more secure network environment and meet people’s demand for a secure network.

Contributions. The contributions of this paper are summarized as follows:

1) The relationship between the importance of the information owned by the connection and the absolute value of the weight. Through comparative experiments of three pruning methods, we prove that in the deep neural network under the intrusion detection environment, the connections with a larger absolute value of the weight have more important information than the connections with a smaller absolute value of the weight.

2) Application of pruning deep neural network in the field of intrusion detection. For the biggest challenge of building anomaly-based NIDS: building a high-performance intrusion detection classifier model, we first propose an effective intrusion detection method based on pruning deep neural network: P-DNN. We use the KDD Cup 99 dataset to evaluate the effectiveness of the method and achieve exciting results. The model constructed by P-DNN achieves a detection rate of 0.9904 for known attacks and a detection rate of 0.1050 for unknown attacks. By comparing with related work, the model achieves the best intrusion detection performance: COST is reduced to 0.1875 and ACC is increased to 0.9317.
II. RELATED WORK

In 2000, of the 24 entries in the KDD Cup 99 competition, the top three winners used some variants of the decision tree. The winner of the competition made use of an ensemble of $50 \times 10$ C5 decision trees, using cost-sensitive bagged boosting [7]. The runner-up first constructed a set of decision trees and then a problem-specific global optimization criterion was used to select an optimal subset of trees to give the final prediction [8]. The third-placed approach used two-layer decision trees. The first layer was trained on the connections which cannot be classified by security experts, and the second layer was built on the connections which cannot be classified by the first layer [9].

In 2001, Ramesh Agarwal et al. proposed a new framework, PNrule. The main idea is learning a rule-based model in two stages: First, find P-rules to predict the presence of a class and then find N-rules to predict the absence of the class. this strategy helps in overcoming the problem of small disjuncts often faced by other sequential covering based algorithms. Another key point in PNrule is the mechanism used for scoring. It allows to selectively tune the effect of each N-rule on a given P-rule. Experimental results showed that the PNrule framework held promise of performing well for real-world multiclassification problems with widely varying class distributions [10].

In 2003, Maheshkumar Sabhnani et al. studied the performance of classical machine learning algorithms on the KDD Cup 99 dataset. They found that some machine learning algorithms performed better for a given attack category. They then built a multi-classifier model in which a specific detection algorithm was associated with an attack category for which it was the most promising. The experimental results showed that significant improvements had been achieved in the detection of Probe, DoS, and U2R [11].

In 2004, Nahla Ben Amor et al. studied the application of Naive Bayes in intrusion detection and proved that even with a simple structure, Naive Bayes could provide very competitive results. In addition, they also compared Naive Bayes with the decision tree. The experimental results showed that the results of using naive Bayes or decision trees were slightly better than those of the KDD Cup 99 winner [12].

In 2005, Chi-Ho Tsang et al. proposed a multi-objective genetic fuzzy system called MOGFIDS for anomaly intrusion detection. The system extracted accurate and interpretable fuzzy rule-based knowledge from network data using an agent-based evolutionary computation framework. The experimental results showed that MOGFIDS achieved robust performance for classifying both intrusion attacks and normal network traffic. In addition, it could search for a reduced feature subset and obtain interpretable fuzzy systems [13].

In 2008, Jiong Zhang et al. used the random forest algorithm to construct the intrusion mode for the limitations of rule-based intrusion detection systems in detecting new attack traffic. By learning the training data, the random forest algorithm could automatically build patterns instead of manual coding rules. The experimental results showed that this method outperformed the winner of the KDD Cup 99 competition [14].

In 2012, Khaled Badran et al. presented a multi-dimensional multi-objective genetic programming feature extraction approach that maps the input feature space into a multi-dimensional decision space to maximize the discrimination between classes. A simple, normal-discriminant-function classifier was used for multi-category classification in the transformed decision space. They applied this approach to the KDD Cup 99 dataset and obtained results that are highly competitive with the KDD Cup 99 winner but with a significantly simpler classification framework [15].

In 2014, Saman Masarat et al. presented a new multi-step framework for intrusion detection systems. In the random feature selection step, features with a higher gain ratio are obtained by using Using Roulette Wheel based on Gain Ratio of features. In classifiers’ combination step, adding the fuzzy weighted combiner can tag weights to classifiers related to their cost and performance. Experimental results showed that this approach returned better results than other similar methods [16].

In 2019, R Vinayakumar et al. explored a deep learning model DNN and proposed a highly scalable and hybrid DNNs framework. In this framework, network attacks were detected and classified by learning the abstract and high-dimensional feature representation of the IDS data by passing them into many hidden layers. The experimental results showed that DNN performs well compared to the classic machine learning classifier on the KDD Cup 99 dataset [17].

Since the KDD Cup competition in 1999, researchers have explored many techniques for building intrusion detection classifier model using the KDD Cup 99 dataset. These have promoted more comprehensive development of research on intrusion detection. But there is still a lot of room for research on high-performance classifier models. On the basis of these investigations, we first propose an effective intrusion detection method based on pruning deep neural network: P-DNN, which aims to build a higher performance intrusion detection classifier model and promote the further development of intrusion detection research.

III. METHODOLOGY

The subsections below describe the proposed intrusion detection framework and the details of how P-DNN works in the framework.

A. Intrusion detection framework

As shown in Figure 1, in the proposed intrusion detection framework, in order to protect demilitarized zone (DMZ) and internal network, the anomaly-based NIDS plays the following role by using port mirroring technology on the key switches: First, detect and alert in real time before the intrusion causes damage. Second, when the intrusion occurs, dynamic defense is performed through linkage with the firewall. Third, after the invasion, the forensic analysis is performed through the logs. Among it, we use P-DNN to complete the biggest
challenge of building an anomaly-based NIDS: building a high-performance intrusion detection classifier model.

B. Why choose the KDD Cup 99 dataset

In 1998, The Defense Advanced Research Projects Agency conducted an intrusion detection assessment project at the MIT Lincoln Laboratory to investigate and evaluate intrusion detection research. They built a network environment that simulates the US Air Force LAN and simulated various user types, various network traffic, and attack methods. Then the DARPA 1998 dataset [18] was constructed with 7 weeks of training data and 2 weeks of testing data. Subsequently, Professor Sal Stolfo and Professor Wenke Lee used data mining technology to perform feature analysis and data preprocessing on this dataset to form a new dataset, namely the well-known KDD Cup 99 dataset [19]. In 2011, Vegard Engen et al. stated that “despite the criticisms, researchers continue to use the data due to a lack of better publicly available alternatives [20].” In 2018, Serhat PEKER et al. investigated the application of neural network in network intrusion detection in the past decade and found that 19 of the 43 articles surveyed used the KDD Cup 99 dataset (44.2%, the largest proportion) [21]. In 2018, Hindy et al. investigated the most cited NIDS researches in the past decade and found that 44 of the 69 articles surveyed used the KDD Cup 99 dataset (63.8%, the largest proportion) [22].

The KDD Cup 99 dataset has been criticized by some network intrusion detection researchers [23] [24] [20], but the authority of the birth and the recognition of many related researchers indicate that it is still the most suitable dataset for evaluating the effectiveness of network intrusion detection methods. This is why we chose the KDD Cup 99 dataset.

C. Selection of data

As shown in Table I, the training dataset includes 494021 instances (kddcup.data_10_percent_corrected [19]), and the testing dataset includes 311029 instances (corrected [19]). The five data types in the table are as follows:

- Normal: normal network connections.
- Probe: attackers attempt to collect information about a computer network to circumvent its security controls.
- DoS: attackers prevent certain services from processing legitimate requests because memory resources are exhausted.
- U2R: attackers attempt to exploit certain vulnerabilities to gain root access to the system after getting normal access.
- R2L: attackers exploit certain vulnerabilities to get access as local users of the computer through a remote connection.

D. Pre-processing

Data pre-processing includes deduplication, character digitization, normalization and oversampling.

1) Deduplication. As shown in Table II, after the training dataset is deduplicated, the number of instances is reduced from 494021 to 145585. Deduplication preserves non-repeating instances and provides baseline data for subsequent oversampling operation.

2) Character digitization. The classifier based on deep neural network only uses numerical data for calculation. Therefore, two tasks need to be completed: First, convert non-numeric...
features of the dataset instance to numbers. Second, convert the attack type of the dataset instance to the number corresponding to its category. As shown in Table III, there are a total of 3 protocol types, 70 service types, 11 connection states, and 5 attack types converted into corresponding digital identifiers.

### Table III

<table>
<thead>
<tr>
<th>Character digitization</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Protocol</td>
<td>tcp, udp, icmp, ucp</td>
</tr>
<tr>
<td>Service</td>
<td>aol, auth, bgp, courier, esnet, ns, slf, daytime, discard, domain, domain_u, echo, . . .</td>
</tr>
<tr>
<td>Connection</td>
<td>oth, r2l, rstr, rsico, rstr, s0, s1, s2, s3, se, sf, sh</td>
</tr>
<tr>
<td>Normal</td>
<td>Normal network connections</td>
</tr>
<tr>
<td>Probe</td>
<td>ipsweep, mscan, nmap, portsweep, saint, satan</td>
</tr>
<tr>
<td>DoS</td>
<td>apache2, back, land, mailbomb, neptune, pod, processtable, snarf, teardrop, udpstorm</td>
</tr>
<tr>
<td>U2R</td>
<td>buffer_overflow, htpd, loadmodule, perl, ps, rootkit, sguil, xterm</td>
</tr>
<tr>
<td>R2L</td>
<td>ftp_write, guess_passwd, imap, multihop, named, phf, sendmail, smuggetattack, snmpguess, spy, warezclient, warezmaster, worm, xlock, xstock</td>
</tr>
</tbody>
</table>

3) Normalization. Data normalization is the process of scaling the value of each feature to a uniform range, thereby eliminating the bias caused by large numerical features, which is defined as (1).

$$x'_i = \frac{x_i}{\sqrt{x_1^2 + x_2^2 + \ldots + x_n^2}}$$  \hspace{1cm} (1)

where the vector $x(x_1, x_2, \ldots, x_n)$ represents the original value of the instance feature, and the vector $x'(x'_1, x'_2, \ldots, x'_n)$ represents the value of the instance feature after the normalization operation.

4) Oversampling. It is easy to see from Table II that the training dataset is highly unbalanced, which is one of the data distribution characteristics of the intrusion detection environment. According to analysis [24], the main reason for the poor performance of low-frequency data by the classifier trained by the KDD Cup 99 dataset is the imbalance of the dataset. In view of this situation, in order to improve the classification effect of low frequency data and the overall performance of the classifier, this paper integrates the oversampling technique [25] into the pre-processing operation. As shown in Figure 2, Probe, U2R, and R2L are oversampled with the number of instances of DOS as the baseline to construct a relatively balanced training dataset. In new dataset, the number of instances of Probe, U2R, and R2L has been expanded by 25 (54572/2130), 1049 (54572/52), and 54 (54572/999) times respectively.

![Fig. 2. Oversampling](image)

**E. Extending feature’s dimension**

A lot of research has shown an important conclusion: By training a larger and more complex neural network model, and then gradually pruning to get a smaller and simpler model, the results are better than those obtained by directly training such a small and simple model [26]. Our idea is that by expanding the original feature’s dimension of the dataset, the dimension of the input layer of the neural network becomes larger and the complexity of the model becomes higher. This operation prepares for the next pruning operation, which aims to reduce the complexity of the model. As shown in Figure 3, the feature’s dimension of the input data is expanded by repeating the original features in order.

![Fig. 3. Extending feature’s dimension.](image)

**F. Deep neural network**

The details of the proposed deep neural network (DNN) architecture are shown in Table IV. The key points are as follows:

### Table IV

<table>
<thead>
<tr>
<th>Layers</th>
<th>Type</th>
<th>units</th>
<th>Activation function</th>
<th>param</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-1</td>
<td>Input layer</td>
<td>512</td>
<td>ReLU</td>
<td>420352</td>
</tr>
<tr>
<td>1-2</td>
<td>Dropout=0.5</td>
<td>/</td>
<td>/</td>
<td>0</td>
</tr>
<tr>
<td>2-3</td>
<td>Full connected</td>
<td>256</td>
<td>ReLU</td>
<td>131328</td>
</tr>
<tr>
<td>3-4</td>
<td>Dropout=0.5</td>
<td>/</td>
<td>/</td>
<td>0</td>
</tr>
<tr>
<td>4-5</td>
<td>Full connected</td>
<td>128</td>
<td>ReLU</td>
<td>32896</td>
</tr>
<tr>
<td>5-6</td>
<td>Dropout=0.5</td>
<td>/</td>
<td>/</td>
<td>0</td>
</tr>
<tr>
<td>6-7</td>
<td>Full connected</td>
<td>64</td>
<td>ReLU</td>
<td>8256</td>
</tr>
<tr>
<td>7-8</td>
<td>Dropout=0.5</td>
<td>/</td>
<td>/</td>
<td>0</td>
</tr>
<tr>
<td>8-9</td>
<td>Full connected</td>
<td>32</td>
<td>ReLU</td>
<td>20800</td>
</tr>
<tr>
<td>9-10</td>
<td>Dropout=0.5</td>
<td>/</td>
<td>/</td>
<td>0</td>
</tr>
<tr>
<td>10-11</td>
<td>Full connected</td>
<td>5</td>
<td>Softmax</td>
<td>165</td>
</tr>
</tbody>
</table>

- Structure: The input layer includes 820 neurons. The five hidden layers include 512, 256, 128, 64, and 32 neurons respectively. The output layer includes 5 neurons.
- Connection mode: Full connection. Each neuron in the current layer is connected to all neurons in the next layer.
- Hidden layer activation function: ReLU. ReLU is a nonlinear activation function. Compared with the linear activation function, it can better express complex classification boundaries and more closely related to the signal excitation principle of neurons, which can improve the performance of the model [27]. In addition, ReLU can help to reduce the state of vanishing and error gradient issue [28].
- Output layer activation function: Softmax. In the five-category environment of this experiment, Softmax that solves the multi-classification problem is more suitable than Logistic that solves the two-category problem. In addition, we can get

![Diagram of DNN Architecture](image)
the probability distribution that the prediction result belongs
to a certain class through Softmax.

- Loss function: Categorical cross-entropy. Combining the
  output layer with Softmax as the activation function, we
  choose categorical cross-entropy as the loss function, which
  is defined as (2).

\[
    \text{loss}(pd, ed) = - \sum x pd(x) \log(ed(x)) \tag{2}
\]

where \( ed \) is true probability distribution, \( pd \) is predicted
probability distribution.

- Optimizer: Adam. It is designed to combine the advan-
tages of two recently popular methods: AdaGrad [29], which
works well with sparse gradients, and RMSProp [30], which
works well in on-line and non-stationary settings. Experiments
have shown that Adam performs better than other stochastic
optimization methods [31].

- Training algorithm: Back Propagation algorithm. The
  algorithm compares the error generated by the theoretical
  output with the actual output, and reversely adjusts the weight
  and bias of each layer connection to optimize the parameters
  of the whole network [32]. From a mathematical point of
  view, the training of neural network is the iterative process
  of each layer input under the nonlinear activation function.
  The process of this iteration is similar to the process of
  biological growth and evolution. This further explains why
  neural network can simulate part of human brain function and
  succeed in many fields.

- Dropout: This is a powerful technique to reduce overfitting
  and improve the generalization of neural network [33]. The
  key idea is that when the forward propagation, the activation
  value of the neuron stops working with a fixed probability, so
  that the neuron does not rely too much on some local features,
  and the generalization ability of the model is stronger.

\[G. \ Pruning\]

Pruning neural network has excellent performance in the
field of deep learning model compression [34]–[36]. The
field is dedicated to reducing the storage requirements of
the model while maintaining the accuracy of classical neural
network models (such as LeNet-5 [37], AlexNet [38], VGG-
16 [39]), and promoting the algorithm to be efficiently applied
to resource-constrained hardware platforms. Different from the
goal of deep learning model compression field, This paper is
dedicated to the application of pruning technology in the field
of intrusion detection to build a higher performance intrusion
detection classifier model, and promote the construction of a
more secure network environment. By predicting the parameters
of the neural network, Misha Denil et al. pointed out that there
are significant redundancy in the parameters used in the
neural network [40]. Brandon Reagen et al. also acknowledge
the fact that most neural network contain far more information
than is needed for precise reasoning [41]. Min Lin et al.
also showed that in the neural network, the fully connected
layer is easy to cause over-fitting, and the simplification of
the fully-connected layer contributes to the improvement of
precision [42]. As shown in Figure 4, the idea of pruning in
this paper is that by pruning the DNN, only the connections
with more important information in the weights are reserved,
and the complexity of the model is reduced, thereby improving
the intrusion detection performance of the model. Assuming
that there are 3 input neurons in a layer and 4 output neurons
in the next layer, and the weight matrix is \(3 \times 4\). The details
of the pruning are as follows:

- Step 1: Sort. The weights between each neuron in the
  DNN and all neurons in the next layer are sorted by absolute
  value.

- Step 2: Prune. According to the pruning rate \(P\) (represent-
ing the proportion of all pruned connections in all connections
of the DNN), the connections with smaller absolute value of
the weight are pruned (the weights are assigned to 0). The
position information of the pruned connections in the original
neural network is recorded during the pruning process.

- Step 3: Retrain. Retrain DNN. Using the position infor-
mation of the pruning, the weights of the pruned connections
are assigned 0 after each round of retraining.

- Step 4: Complete. Get the best model or go to Step 3.
IV. EXPERIMENTAL RESULTS

This section will first introduce the performance indicators used to evaluate the effectiveness of intrusion detection method. Then, the experimental results of this paper are presented and analyzed. Finally, compare our method with the excellent related work.

A. Performance indicators

In order to evaluate the effectiveness of our proposed method, we propose to use three evaluation indicators: COST, accuracy (ACC), detection rate (DR). Among them, COST is the most important, ACC is the second, and DR is the reference. As shown in Table V, we will explain the calculation details of the three evaluation indicators through an example of confusion matrix.

<table>
<thead>
<tr>
<th>Normal</th>
<th>Probe</th>
<th>DoS</th>
<th>U2R</th>
<th>R2L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>x_{00}</td>
<td>x_{01}</td>
<td>x_{02}</td>
<td>x_{03}</td>
</tr>
<tr>
<td>Probe</td>
<td>x_{10}</td>
<td>x_{11}</td>
<td>x_{12}</td>
<td>x_{13}</td>
</tr>
<tr>
<td>DoS</td>
<td>x_{20}</td>
<td>x_{21}</td>
<td>x_{22}</td>
<td>x_{23}</td>
</tr>
<tr>
<td>U2R</td>
<td>x_{30}</td>
<td>x_{31}</td>
<td>x_{32}</td>
<td>x_{33}</td>
</tr>
<tr>
<td>R2L</td>
<td>x_{40}</td>
<td>x_{41}</td>
<td>x_{42}</td>
<td>x_{43}</td>
</tr>
</tbody>
</table>

1) COST. As shown in (3), the smaller the COST value, the better the model. Where N represents the total number of instances tested. CM represents the confusion matrix, and CM(i, j) represents the number of instances originally belonging to class i that are classified as class j. C represents the cost matrix, and C(i, j) represents the cost that the instances originally belonging to class i are classified as class j.

\[
COST = \frac{1}{N} \sum_{i=0}^{4} \sum_{j=0}^{4} CM(i, j) \times C(i, j) \quad (3)
\]

<table>
<thead>
<tr>
<th>Normal</th>
<th>Probe</th>
<th>DoS</th>
<th>U2R</th>
<th>R2L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Normal</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Probe</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>DoS</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>U2R</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>R2L</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

As shown in Table VI, the official provided a cost matrix for evaluating entries in the KDD Cup 99 competition [19]. This is also the cost matrix used by COST. In the cost matrix, the magnitude of these cost values are proportional to the impact of the attack on the computing platform. Therefore, we think that COST is the most important evaluation indicator for analyzing the performance of a model from the perspective of detecting intrusion.

2) ACC. As shown in (4), the larger the ACC value, the better the model. Among them, x_{00}, x_{11}, x_{22}, x_{33} and x_{44} represent the number of instances belonging to Normal, Probe, DoS, U2R and R2L that are finally classified correctly.

\[
ACC = \frac{x_{00} + x_{11} + x_{22} + x_{33} + x_{44}}{\sum(X)} \quad (4)
\]

Although ACC is a widely used classifier performance evaluation indicator. But when the data is unbalanced (as shown in Table I, the testing dataset is extremely unbalanced), the ACC will be misleading to the researcher. Assuming that the testing dataset with 90 class A instances and 10 class B instances are classified. If the model predicts all instances as class A, the ACC is as high as 0.9. However, the model does not predict any class B instances, which is undoubtedly an unsuccessful classifier. Therefore, we think that ACC is the second most important indicator for analyzing model performance from the perspective of data classification.

3) DR. As shown in (5)-(9), the larger the DR value, the better the model. Among them, the DR is represented by the proportion of the number of correctly classified instances to the total number of instances of this type.

\[
DR_{Normal} = \frac{x_{00}}{\sum x_{0i}} \quad (5)
\]

\[
DR_{Probe} = \frac{x_{11}}{\sum x_{1i}} \quad (6)
\]

\[
DR_{DoS} = \frac{x_{22}}{\sum x_{2i}} \quad (7)
\]

\[
DR_{U2R} = \frac{x_{33}}{\sum x_{3i}} \quad (8)
\]

\[
DR_{R2L} = \frac{x_{44}}{\sum x_{4i}} \quad (9)
\]

DR shows the model’s ability to detect a certain type of data in detail. Therefore, we think that DR can be used as a reference indicator to analyze the performance of the model.

B. Details of experimental results

1) The relationship between the importance of the information owned by the connection and the absolute value of the weight. After sorting the weights between each neuron in the DNN and all the neurons in the next layer in absolute order, in the process of reducing the same complexity of the model, we set up three pruning methods to compare, where the pruning rate P represents the proportion of pruned connections in all connections of the DNN. Method 1: According to the value P, prune the connections with a larger absolute value of the weight. Method 2: In the order of absolute values from small to large, every other connection, prune x connections (x=1, P=0.5; x=2, P=0.667; ...), reserving some connections with a smaller absolute value of the weight and some connections with a larger absolute value of the weight. Method 3: According to the value P, prune the connections with a smaller absolute value of the weight.

As shown in Figure 5, in contrast, the performance of the model is best under the effect of the pruning method 3 represented by the red line. The model achieves a lower COST, which is more stable and reduces the possibility of losing important information of the model caused by pruning and can...
not be retrained to recover. The second is pruning method 2, and the last is pruning method 3. This proves that in the DNN under the intrusion detection environment, the connection with a larger absolute weight value has more important information than the connection with a smaller absolute weight value. By adopting the P-DNN combined with the pruning method 3 which only the connections with more important information in the weight were reserved and reduce the complexity of the model, we found the best intrusion detection classifier model.

2) Best model. As shown in Figure 6, during the change of the pruning rate $P$ in steps of 0.001, we found the best model when $P$ equals 0.885. The confusion matrix and performance evaluation indicators of the model are shown in Table VII. From the perspective of detecting intrusion, the COST of the model is 0.1875. From the perspective of data classification, ACC of the model is 0.9317. From the perspective of evaluation reference indicator $DR_{normal}$, $DR_{probe}$, $DR_{DoS}$, $DR_{U2R}$ and $DR_{R2L}$ are 0.964, 0.886, 0.968, 0.272 and 0.313 respectively.

As shown in Table IX, the model performs well on known attacks (the attack instances exist in the training dataset and testing dataset) with a detection rate as high as 0.9904. The model performed poorly on unknown attacks (the attack instances only exist in the testing dataset) with a detection rate of only 0.1050. But the model exhibits high value for detecting the 0-day attack and variants of attacks. We think that the main reasons for poor performance include the following two points: First, it is difficult to detect application layer attacks, such as mailbomb, through the connection features of the network layer. Second, the features of the dataset are not sufficient to distinguish between the normal connections and the connections of certain attack; as shown in Table VIII, in the testing dataset, there is case where the snmpgetattack instance and the normal instance have exactly the same features.
## C. Comparison

In order to locate our research, we have made a comparison as shown in Table X. The comparison included recent research using the KDD Cup 99 dataset and achieving satisfactory results. Evaluation indicators include COST, ACC, and DR. Regrettably, due to the different indicators used by different researchers, only incomplete evaluation data can be collected in some research. However, we still include these research in the scope of comparison because of their excellent research results.

As can be seen from Table X, the model constructed by P-DNN is superior in performance to all other related work in the table. From the perspective of detecting intrusion, COST is greatly reduced to 0.1875, and from the perspective of data classification, ACC is slightly increased to 0.9317. These reflect the excellent intrusion detection performance of the model. Maheshkumar Sabhnani et al. stated that all the classic machine learning algorithms tested on the KDD Cup 99 dataset offered an acceptable level of detection performance only for DoS and PROBE attacks and demonstrated poor performance on the U2R and R2L [43]. However, from the perspective of the reference indicator detection rate, compared with other research works, the method we proposed can not only get satisfactory results on DR\textsubscript{Normal}, DR\textsubscript{Probe}, and DR\textsubscript{DoS} but also make DR\textsubscript{U2R} and DR\textsubscript{R2L} achieve some improvement.

## V. Conclusions

We first propose an effective intrusion detection method based on pruning deep neural network: P-DNN. Firstly we train a deep neural network with complex structure and good detection performance through extending feature’s dimension. Then, through comparative experiments of three pruning methods, it is proved that in the deep neural network under the intrusion detection environment, the connections with a larger absolute value of the weight have more important information than the connections with a smaller absolute value of the weight. On this basis, through the pruning operation, the weights of the deep neural network with a smaller absolute value are assigned to 0, which only reserve the connections with more important information in the weight, reducing the complexity of the model. Finally, retrain the remaining connections with a larger absolute value of the weight to find the best model. We use the KDD Cup 99 dataset to evaluate the effectiveness of the method and achieve exciting results. The model constructed by P-DNN achieves a detection rate of 0.9904 for known attacks and a detection rate of 0.1050 for unknown attacks. At the same time, we explain two main reasons why the model performs poorly on unknown attacks: First, it is difficult to detect application layer attacks through the connection features of the network layer. Second, the features of the dataset are not sufficient to distinguish between

### Table IX

<table>
<thead>
<tr>
<th>Attack</th>
<th>Detected</th>
<th>Total</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probe</td>
<td>303</td>
<td>306</td>
<td>0.9902</td>
</tr>
<tr>
<td>nmap</td>
<td>84</td>
<td>84</td>
<td>0.9718</td>
</tr>
<tr>
<td>portsweeep</td>
<td>344</td>
<td>354</td>
<td>0.9718</td>
</tr>
<tr>
<td>satan</td>
<td>1633</td>
<td>1633</td>
<td>0.9945</td>
</tr>
<tr>
<td>DoS</td>
<td>back</td>
<td>0</td>
<td>0.9942</td>
</tr>
<tr>
<td></td>
<td>land</td>
<td>0</td>
<td>0.9942</td>
</tr>
<tr>
<td></td>
<td>Neptune</td>
<td>57943</td>
<td>58001</td>
</tr>
<tr>
<td></td>
<td>pod</td>
<td>44</td>
<td>87</td>
</tr>
<tr>
<td></td>
<td>smurf</td>
<td>163996</td>
<td>16401</td>
</tr>
<tr>
<td>U2R</td>
<td>buffer_overflow</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>loadmodule</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>perl</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>rootkit</td>
<td>6</td>
<td>13</td>
</tr>
<tr>
<td>R2L</td>
<td>ftp_binary</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>guess_password</td>
<td>3726</td>
<td>4367</td>
</tr>
<tr>
<td></td>
<td>imap</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>multihop</td>
<td>4</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>phf</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>warzone</td>
<td>1315</td>
<td>1602</td>
</tr>
</tbody>
</table>

### Table X

<table>
<thead>
<tr>
<th>Attack</th>
<th>Detected</th>
<th>Total</th>
<th>DR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Known attacks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mscan</td>
<td>591</td>
<td>1053</td>
<td>0.5613</td>
</tr>
<tr>
<td>sain</td>
<td>734</td>
<td>736</td>
<td>0.9973</td>
</tr>
<tr>
<td>R2L</td>
<td>apache2</td>
<td>169</td>
<td>794</td>
</tr>
<tr>
<td></td>
<td>mailbomb</td>
<td>0</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>processable</td>
<td>427</td>
<td>759</td>
</tr>
<tr>
<td></td>
<td>udpstorm</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Unknown attacks</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attack</td>
<td>Detected</td>
<td>Total</td>
<td>DR</td>
</tr>
<tr>
<td>mscan</td>
<td>591</td>
<td>1053</td>
<td>0.5613</td>
</tr>
<tr>
<td>sain</td>
<td>734</td>
<td>736</td>
<td>0.9973</td>
</tr>
<tr>
<td>R2L</td>
<td>apache2</td>
<td>169</td>
<td>794</td>
</tr>
<tr>
<td></td>
<td>mailbomb</td>
<td>0</td>
<td>5000</td>
</tr>
<tr>
<td></td>
<td>processable</td>
<td>427</td>
<td>759</td>
</tr>
<tr>
<td></td>
<td>udpstorm</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

## Multi-Objective Genetic Programming [15] 

- 0.995 0.780 0.970 0.114 0.036 0.9240 0.2431

## Tree Classifier + Fuzzy Ensemble [16] 

- 0.995 0.764 0.942 0.089 0.243 0.9129 0.1797

## DNN [17] 

- 0.995 0.764 0.942 0.089 0.243 0.9129 0.1797

## P-DNN 

- 0.995 0.764 0.942 0.089 0.243 0.9129 0.1797
the normal connections and the connections of certain attack. By comparing with related work, the model built by P-DNN achieves the best intrusion detection performance: From the perspective of detecting intrusion, COST is greatly reduced to 0.1875; from the perspective of data classification, ACC is slightly increased to 0.9317; from the perspective of detecting intrusion, it also achieves some improvement in DR_{Normal}, DR_{Probe}, and DR_{DoS}, and it also achieves some improvement in DR_{U2R} and DR_{R2L}.

ACKNOWLEDGMENT

This work was supported by NSFC-General Technology Fundamental Research Joint Fund (No. U1836215), and the National Key Research and Development Program of China (No. 2016QY03D0605).

AVAILABILITY

Codes are available at: https://github.com/BydRay/P-DNN

REFERENCES