Received: 30 December 2019 | Revised: 7 February 2020 | Accepted: 11 February 2020

DOI: 10.1002/ett.3922

SPECIAL ISSUE ARTICLE

WILEY

An unknown protocol syntax analysis method based on convolutional neural network

Yichuan Wang | **Binbin Bai** | **Xinhong Hei** | **Lei Zhu** | **Wenjiang Ji**

College of Computer Science and Engineering, Xi'an University of Technology, Xi'an, China

Correspondence
Xinhong Hei, College of Computer Science and Engineering, Xi'an University of Technology, Xi'an, China.
Email:heixinhong@xaut.edu.cn

Funding information
Key Research and Development Program of Shaanxi Province, Grant/Award Numbers: 2017ZDXM-GY-098, 2019TD-014; National Key Research and Development Program of China, Grant/Award Number: 2018YFB1201500; National Natural Science Founds of China, Grant/Award Numbers: 61602374, 61602376, 61702411, 61773313; National Natural Science Founds of Shaanxi, Grant/Award Numbers: 2016JQ6041, 2017JQ6020; Science Technology Project of Shaanxi Education Department, Grant/Award Numbers: 16JK1552, 16JK1573

Abstract

In recent years, a large number of botnets and dark networks rely on command and control channels of unknown protocol formats for communication, and with the development of Internet of Things technology, this problem becomes more prominent. The syntax analysis of the unknown protocol is helpful to measure the boundary of Botnet in the environment of Internet of things, so as to protect the network security. Based on the analysis of the characteristics of the current bitstream protocol data format, this article proposes an unknown protocol syntax analysis method based on convolutional neural network (CNN). First, the protocol data are preprocessed, and then the image is transformed. Next, the converted image is input to the convolution layer for convolution. After convolution, the data are flattened. Then the flattened data are put into the fully connected neural network. Finally, the unknown protocol is analyzed and predicted. The experimental results show that compared with the traditional feature extraction combine frequent item algorithm (CFI) and other neural network deep neural networks, CNN is 15% more accurate than CFI in the analysis of unknown protocol syntax, and it can accurately analyze and identify the unknown protocol.

1 | INTRODUCTION

With the rapid development of technology such as the Internet of Things and the mobile Internet, the concept of the Internet of Things is gradually deeply rooted in the hearts of the people, and a large number of devices are directly connected to the intelligent terminals of users.[1] The Internet of Things[2] can be divided into three levels: application layer, network layer, and perceptual layer. As the medium between the perceptual layer and the application layer, the network layer undertakes the important task of collecting all kinds of information data at the bottom of the Internet of Things equipment and transmitting it to the cloud computing platform for aggregation and processing, so as to realize the real-time synchronization of the data. This process will lead to a large number of sensitive data uploaded to the cloud server through different ways. The application layer as an information business management platform is related to the core trade secrets of the enterprise. And the security monitoring of the application layer in the traditional Internet enterprises has been more mature. The operation and maintenance of the Internet of Things enterprises usually invest a lot of resources to maintain the security and stability of the server. Because of its difficult compatibility and cost control, the network layer is often ignored in the security architecture of Internet of Things enterprises.

Due to the gradual popularity of the Internet of Things, the corresponding network security problems are also gradually prominent.[3] A large number of devices accessing the network from different places will increase the network attack area, especially most of the Internet of things devices have limited energy and resources. And because most devices do not have traditional IT hardware protocols, they cannot run standard encryption, authorization and access control algorithms. Therefore, they are particularly vulnerable to targeted denial of service attacks,[4] including: (i) physical tampering and stealing data, code, and keys; (ii) data integrity is destroyed by falsifying identity; (iii) wiretapping is carried out by sharing wireless channel; and (iv) false nodes are used to maliciously interfere with the communication link between IoT devices.

Network protocol[5] is the communication channel between computers and the basis of network interconnection. In recent years, the frequency of Botnet, dark network, and illegal trading is increasing sharply. As a bridge of these methods, protocol can capture the lifeblood of these illegal means.Because the means of unknown network protocol is almost 0 in the existing technology, it is urgent to analyze the unknown network protocol. According to the ITRC, so far in 2018, there have been more than 1100 data leakage incidents,[6] with a total of 561 700 000 exposure records. The global average cost of data leakage research in 2018, sponsored by the Ponemon.

Institute and IBM security agencies,[7] is now $3.9 million, up 6% from 2017. Although ransomware topped the list of cyber threats in 2017, Wannacry and NotPetya were particularly notable. Blackmail attacks decreased in 2018. According to Kaspersky's ransomware and Malious Cryptomakers 2016-2018 report, ransomware infection has decreased by nearly 30% in the past 12 months, and cryptocurrency mining has increased by 44.5% over the same period.

While the number of blackmailing software is decreasing, the complexity is increasing as the network criminals upgrade the means of attack.[2] The number of new racketeering software variants increased by 46% over last year, which means that the blackmailing software is still a threat to many companies, especially two of the most hot targets in the area of health care and finance.

However, even if you do a good job in network security, it is hard to avoid mistakes. In order to reduce the risk of data loss caused by extortion software attacks, organizations such as enterprises should focus on implementing data protection strategies, which include not only automatic backup but also easy recovery.

With the development of the Internet of Things and the popularization of network applications, the above mentioned unsafe hidden dangers are more prominent. How to analyze software network behavior and communication protocol by protocol reverse analysis is a hot spot in the field of network security at present, and also an important part of protocol security analysis.

Protocol reverse parsing is to use the idea of reverse analysis to analyze the protocol format used by the target network application and the program semantic information corresponding to each protocol field.[8] It has very important application value in the field of network security such as vulnerability mining, network intrusion detection, network management, fingerprint generation, application session replay, and so on. Unknown protocol analysis includes syntax analysis, semantic analysis, and timing analysis. Syntax analysis refers to the extraction of protocol features to find out the format and content of protocol frames. The purpose of semantic analysis is to construct the logical model of protocol syntax, focusing on the internal logical relationship between the messages of the protocol. How the protocol interacts must follow certain syntax rules. Analyze what control information is sent, what action is done, and what response is made. Timing analysis refers to the analysis of the order in which protocol messages are sent, including the speed and time of transmission. As the basis of semantic analysis and timing analysis, grammar analysis is of self-evident importance, so our research goal is to analyze the protocol format by analyzing the syntax of unknown protocols.

Because many botnets are extending to the field of Internet of Things, many IoT devices have become the building nodes of botnets. These botnet nodes communicate by using unknown protocol and through unknown Command and Control channel.[9] Based on the above research status, combined with the purpose of analytic network protocol, this article proposes a method of network protocol syntax analysis based on convolutional neural network (CNN) to solve the problem of reverse analysis of unknown network protocol. CNN focuses on the characteristics of data. By inputting the data containing the characteristics and adjusting the parameters, the neural network can recognize the corresponding characteristics, and then classify and predict the protocol. Our method helps to measure the boundary of the botnet in the Internet environment and find the nodes in it to defend and protect the security.

The rest of this article is organized as follows. The first part introduces the development of unknown protocols in network security. The second part introduces our work in the analysis of unknown protocols. In the third part, we propose

a new protocol format analysis algorithm. In the fourth part, we analyze the performance of the new algorithm from many aspects and compare it with other algorithms. Finally, let us summarize our work.

2 | RELATED WORK

The identification technology of unknown bitstream protocol is one of the research areas of protocol reverse engineering. Protocol reverse engineering is to capture a large number of network packets and use technical analysis to find the types of protocols they belong to. In the early years, the traditional protocol reverse engineering technology is mainly based on the manual analysis of independent network packets to achieve the description of protocol characteristics. With the development of protocol reverse engineering, there are some protocol analysis tools, which gradually make protocol reverse engineering separate from the completely manual analysis method. The common protocol parsing tools are mainly divided into commercial parsing software and free parsing tools, of which tcpdump[10,11] and Wireshark are the most widely used free parsing tools.

At present, in the development process of protocol reverse technology automation,[12] the technology of protocol recognition mainly includes the recognition method based on pattern matching, the recognition method based on data mining, and the recognition method based on finite state machine (FSM). They analyze data from different perspectives and mine feature information to identify unknown protocols.

Protocol reverse based on feature recognition mainly studies the extraction of protocol features. Researchers improve the traditional feature recognition algorithm, and then apply the improved algorithm to the network data environment.[13] Or they propose a new efficient matching algorithm suitable for network data environment. Due to the similarity of the gene sequence and the protocol data sequence in the comparison, the target sequence can be correctly extracted from the original data. But the unknown protocol method based on the pattern recognition mainly aims at the protocol of the application layer, and the bit stream data is rarely analyzed.

Data mining techniques are applied to mainstream techniques of protocol feature recognition, including clustering and association analysis. Unlike the idea of pattern matching, data mining focuses on discovering implicit relationships between data. Clustering analysis is to cluster by calculating the attributes such as editing distance. It is considered that the central point of each class is the sequence of protocol features, which can represent the characteristics of this class, so as to realize the recognition of the protocol. Clustering analysis[14] mostly focuses on the analysis of protocol semantics. Association analysis is to analyze by setting the minimum support and minimum confidence, filter the nonfeature string with support, and then calculate the implied relationship between the string and the string with confidence. Association analysis can correlate, belong to the relationship chain of the protocol, but the selection of features is an important prerequisite for the quality of the relationship chain and the speed of mining, so the association rules need to cooperate with other techniques to realize the recognition of the protocol. Although the data mining method can well mine the feature information of the data, it cannot simply rely on the data mining technology to identify the unknown protocol because of the complexity of the algorithm and the large overhead.

The unknown protocol identification technology based on FSM is mainly used in high-level protocol analysis.[15] Using the advantages of FSM in state transition description, the protocol features can be displayed intuitively and correctly, and then the state transition probability can be calculated by using the idea of statistics, and the state transition of the protocol can also be described in an orderly manner. At the same time, the FSM method can be combined with machine learning and keyword matching technology, and the feature information can be trained by keyword to identify the protocol.[16,17] However, this technology cannot analyze the protocol features alone, so it needs to cooperate with pattern recognition technology or data mining technology to identify the accurate protocol features. At the same time, if the protocol has too much characteristic information, the establishment of FSM will be very complex.

According to the research of the traditional protocol identification method, the method based on the pattern matching needs to be improved to be applicable to the bit stream environment of the wireless network, and the method based on the data mining and the method based on the finite automaton are applied to the unknown protocol feature recognition of the wireless network as an auxiliary means. In this article, the identification method of the unknown protocol in the environment of the convolution neural network is proposed. The protocol characteristics, the identification, and analysis of the protocol can be fully realized for various forms of protocol data in the network environment.

In recent years, the method based on deep neural network (DNN)[18,19] has made great breakthroughs and achievements in the fields of speech recognition, natural language understanding, and one-view human motion recognition. The unknown protocol is used to identify the feature of automatic learning by using depth neural network.

CNN is the most important classification model in the field of deep learning. Its application in many fields exceeds the accuracy and performance that traditional pattern recognition and machine learning algorithms can achieve. In recent years, researchers have proposed many protocol recognition methods about CNN. In Reference 20, deep learning and neural network were used earlier to solve the problem of network protocol identification, and the first 1024B data of transmission control protocol (TCP) session was classified by artificial neural network. First, the data are transformed into one-dimensional vector, and then the vector is input into the artificial neural network model for training. The average recognition accuracy of the known protocol is 97.9%. In Reference 21, the influence of CNN trained by different optimizers on the performance of network protocol recognition is studied. The experimental results show that the performance of the optimizer based on gradient descent method is the best, but the overall recognition accuracy is only 77.81%. In Reference 22, only the application layer protocol based on TCP is considered. In References 23 and 24, CNN is applied to the field of malicious traffic classification for the first time. The effect of using all protocol level data and using only application-level protocol data is compared. The first 784B of data is transformed into a two-dimensional image of 28×28 and then input into CNN for classification. The classification results show that using all protocol level data as the research object achieves the best classification effect. Ma et al[25] proposed a CNN-based method for the identification of known and unknown traffic. In the experiment, 13 protocols are selected to form data sets, 10 of which are known protocols, and the other 3 are simulated data of unknown protocol traffic. The experiment is compared with the traditional machine learning algorithm support vector machine (SVM) and naive Bayesian classification model. The test results show that the recognition effect of the CNN model is better than that of the traditional machine learning method, but its method cannot accurately identify the various protocols contained in the unknown protocol traffic.

From the above data, we can see that the development of neural network is increasing day by day, but the theory of applying neural network to unknown protocol syntax analysis is almost zero. At present, the traditional method of analyzing the unknown bitstream protocol is manual, which is very labor-consuming and error prone. We need to propose an automatic and effective technical scheme. In the absence of any prior knowledge, we cannot know the three elements of the bitstream protocol, so the traditional data recognition method of the bitstream protocol cannot do it. Therefore, in order to solve the above problems, this article combines the syntax analysis of unknown protocol with neural network and proposes a new syntax analysis method of unknown protocol based on volume and neural network.

3 | SYNTAX ANALYSIS OF UNKNOWN PROTOCOL BASED ON CNN

Because in the protocol format, the top protocol format is distributed in the protocol header. It is easy to identify which protocol is by identifying the protocol header.[26] The protocol format of the same protocol is similar. If we transform the protocol data into images, this part of the characteristics of the same protocol is equivalent to the characteristics of the same kind of images, and the CNN has significant effect in image processing, so we can use the CNN to identify the unknown protocol.

With the increasing number of new network attacks, the attack mode of unknown protocol is endless.[27,28] In order to deal with the increasing number of new network attacks, we apply CNN to the protocol recognition system, and propose an unknown protocol recognition system model based on CNN.

The principle of the model is: first, the data package is analyzed in depth from the network traffic to get the data to be detected, which is processed by the preprocessing module to become the data to be detected. Then the data are convoluted, pooled, and flattened, then the processed data are put into the all connected network for training, and finally, our model is output. The specific flow chart is shown in Figure 1 .

3.1 | Data preprocessing

Part of the data used in this article comes from the data obtained by Wireshark packet capturing tool and the other part of the unknown protocol data is grabed at Jiangsu Province, Henan Province, Shaanxi Province, and Shandong Province, China.

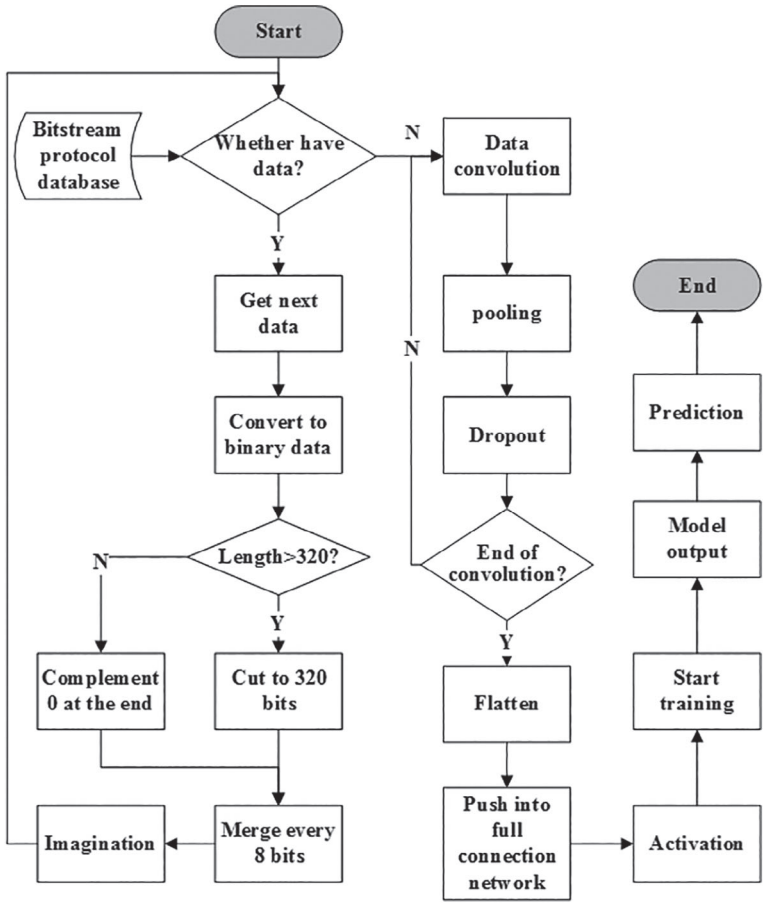


FIGURE 1 Unknown protocol syntax analysis flowchart

From the network communication, the protocol data are analyzed in depth to get the data to be detected, but usually these data cannot be directly input into the unknown protocol recognition model for training, so the data need to be preprocessed before the model training.[29] The first step of data preprocessing is to convert all data frames into binary data frames, which are between 0 and 1. Since the data obtained through Wireshark are hexadecimal data, we need to convert the data first. Then, in order to get the features that can be recognized by CNN, data frames need to be formatted. Because the protocol format is located at the head of the protocol frame, the protocol tail is not very helpful for protocol recognition, so we intercept the protocol data frame. By intercepting *nbytes* bits of the protocol, when the length of the protocol is less than *nbytes*, 0 is added at the end of the frame. For example, the data snippet we obtained from Wireshark is "|4c|cc|6a|4c|fb|16|14|14|." Let us first remove the redundant "|" symbol, and the result is "4ccc6a4cfb161414." It is then converted to binary and the result is "01001100110011000110101001001100111111011000101100001010000010100." Next, we determine whether the length of the binary is greater than 320, if it is greater than, the redundant content is deleted, and if it is less than zero at the end, it is equal to doing nothing. Since the string length is 64 bits, less than 320. Therefore, we make up a total of 256 zeros for the end of the string. The end result is "01001100110011000110101001001100 11111011000101100001010000010100000…0000." The detailed steps are shown in Table 1.

3.2 | Image conversion

Because convolution neural network has a good effect on image processing, and convolution operation is needed when using convolution neural network, convolution layer can only recognize image data of matrix type. So we need to transform the input data into images. When using binary to represent graphics, we usually use gray-scale image and color image to represent. The color image needs three-dimensional 8-bit binary representation. It is very difficult to transform a protocol data frame. When we use the gray-scale image to represent, because the gray-scale image is represented by one-dimensional 8-bit binary. We just need to put every 8 bits of binary together and convert them into a decimal number between 0 and 255. Each protocol will generate *nimage* image data between 0 and 255. For example, protocol data frame

TABLE 1 Data preprocessing steps

Steps number	Steps in detail
Input	Acquired data
Output	Data after data preprocessing.
Step 1.	Remove redundant information from data.
Step 2.	Convert the acquired data to binary data.
Step 3.	Determine whether the data length is greater than 320 bits.
Step 4.	If length is greater than 320 bits, skip to Step 5. Otherwise, skip to Step 6.
Step 5.	Delete data after 320 bits of data.
Step 6.	Fill in 0 after the data.
Step 7.	Save data.
Step 8.	Get the next data and skip to Step 1.

segments {010111011111000100010001}, we put them together every 8 bits, that is, {01011101,11110001,00010001}. And then convert them to decimal numbers, that is, {93,239,17}. {93,239,17} is the data after image.

3.3 | Label settings

After image processing, label the sample according to the protocol category of the sample: set an array M with N (n is the number of protocol categories to be recognized, positive integer) elements, the $M[i] = 1$, and the rest values are 0, which means that the sample is the i protocol, as the label array y', for example: when $N = 4$, $i = 2$, the array $M = [0, 0, 1, 0]$ represents the third protocol among the four protocols to be recognized, and then the label and sample are stored in the training set one by one; repeat the operation of image and label setting, and the training set of CNN is generated. Similarly, the test set D can be generated. For example, when we need to represent TCP, UDP, ARP, and other protocols, we use [1,0,0,0],[0,1,0,0],[0,0,1,0],[0,0,0,1] four labels to represent Dataset $D = [[1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 1, 0], [0, 0, 0, 1]]$.

3.4 | CNN module

The first-stage convolution network is made up of one convolution layer and one maximum pool layer. The convolution kernel of the convolution layer has a size *kernel _size*, with the convolution kernel number $filters_1$, and the *strides*. The input of the first convolution layer is *input_ranges*, followed the maximum pool layer by a *pool_size*.

In this article, *dropout* regularization method[30] is used to prevent the model from over fitting and improve the generalization ability of the model. The activation function is the ReLU activation function. The purpose of activation function: in neural network, the function of activation function is to add some nonlinear factors to the neural network so that the neural network can better solve more complex problems.

The ReLU function is defined by the following formula:

$$ReLU(x) = \left\{ \begin{array}{l} x|if(x > 0) \\ 0|if(x \leq 0) \end{array} \right\}. \tag{1}$$

Advantages of ReLU activation function:

1. In the case of back propagation, the gradient can be avoided to disappear.
2. ReLU will make the output of some neurons to 0, which will cause the sparsity of the network, reduce the interdependence of parameters, and alleviate the occurrence of over fitting problem.
3. Compared with sigmoid activation function, tanh activation function has a simple derivation. When sigmoid and other functions are used to calculate the activation function (exponential operation), the amount of calculation is large. When backpropagation is used to calculate the error gradient, the derivation involves division, and the amount of calculation is relatively large. However, when ReLU activation function is used, the amount of calculation in the whole process is saved a lot.

The second convolution neural network is similar to the first one. The convolution kernel size is still same to convolution, but the number of convolution filters increases to $filters_2$.

Then through dropout's regularization method, randomly delete some redundant information to prevent model over fitting, so as to improve the generalization ability of the model.

Then enter the results into the flatten layer. Flatten layer is used to flatten the input, that is, to unidimensional the multidimensional input. Next, we will enter the full connection layer, the role of the full connection layer, which is to integrate the highly abstract features after the previous convolution, and then normalize them. Output a probability for all kinds of classification situations, and then the classifier can classify according to the probability obtained by the full connection.

Fully connected (FC) layers play the role of "Classifier" in the whole CNN. If the operations of volume layer, pool layer, and activation function layer are to map the original data to the hidden layer feature space, the full connection layer is to map the learned "distributed feature representation" to the sample marker space. In practice, the full connection layer can be realized by convolution operation: for the full connection layer of the front layer, the convolution kernel can be converted into the convolution of 1×1; for the full connection layer of the front layer, the convolution kernel can be converted into the $h \times w$ global convolution, and H and W are the height and width of the convolution result of the front layer, respectively.

Full connection is a matrix multiplication, equivalent to a feature space transformation, which can extract and integrate all the useful information in front. Coupled with the nonlinear mapping of the activation function, the multilayered fully connected layer can theoretically simulate any nonlinear transformation.

One of the functions of full connection is dimension transformation, especially it can change the high dimension to the low dimension, and keep the useful information. The other function of full connection is to embed the implicit meaning, mapping the original features to the hidden nodes. For the last layer of full connection, it is the display expression of classification. All connections in the same position of different channels are equivalent to convolution of 1×1. The total connection of N nodes can be approximated to the global average pooling (GAP) after convolution of N templates.

The full connection layer consists of two parts. First, the data of the upper layer are flattened, and then input to the full connection network. The full connection network has two layers. There are $nodes_1$ nodes in the first layer of fully connected network, and the activation function is still ReLU function. The last layer has $nodes_2$ nodes, and the activation function is softmax function.

The softmax function maps the output of multiple neurons into the (0,1) interval, which can be regarded as the probability that the current output belongs to each classification, so as to carry out multiclassification. The main application of this algorithm is multiclassification and mutual exclusion, that is, it only belongs to one of the classes. Unlike the activation functions of sigmoid class, the general activation functions can only be divided into two categories, so it can be understood that softmax is an extension of the activation functions of sigmoid class.

Sigmoid function,[31] also known as logistic function, is used for the output of hidden neurons, with a value of (0,1). It can map a real number between (0,1), and can be used for binary classification.

The sigmoid function is defined by the following formula:

$$S(x) = \frac{1}{1 + e^{(-x)}}, \tag{2}$$

where x is the measured value and S is the probability calculated by sigmoid. The sigmoid function image is shown in Figure 2.

When judging the classification of an object, it is difficult to achieve 100% assurance, and the probability needs to be calculated. It can be seen from the figure that the figure is a monotonically increasing image. We can refer to the line $y = 0.5$. Above 0.5, the probability that the measured value belongs to 1 is high. Below 0.5, the probability that the value belongs to 0 is high. The line $y = 0.5$ is a dividing line. It is impossible to determine which classification it belongs to. As x tends to be positive or negative infinity, the probability that the measured value belongs to 1 or 0 is higher.

Sigmoid maps a real data to an interval of $(0, 1)$ or $(-1, 1)$, which can be used for binary classification. Mapping the value of a function to $(0, 1)$ can explain the probability that belongs to this class. In addition, Sigmoid function is monotonically increasing and its reciprocal form is very simple, which is a more suitable function.

One disadvantage of sigmoid is that it can only do two categories. Softmax, by imitating sigmoid function, takes a k-dimensional real value vector $A(A1, A2, A3, A4)$ is mapped to a sequence $B(B1, B2, B3, B4)$ distributed between $(0, 1)$,

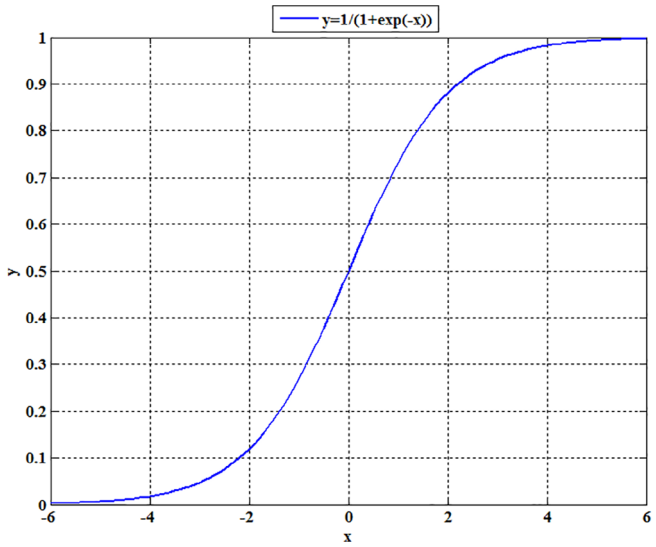


FIGURE 2 Sigmoid function

where *Bi* is a constant of $(0-1)$, then multiclassification tasks can be performed according to the size of *Bi*, such as taking the one dimension with the largest weight.

The softmax function is described as follows.

$$\sigma(z)_j = \frac{e^z j}{\sum_{k=1}^{K} e^z k}. \tag{3}$$

The reason why softmax is chosen is largely because the exponent is used in softmax, which can make the big value bigger, the small one smaller, increase the contrast of differentiation, and improve the learning efficiency. The second reason is that softmax is continuous and differentiable, which eliminates inflection point. This feature is very necessary in gradient descent method of machine learning.

3.5 | Result output module

At last, we set up the model, the loss function is set as the categorical_ crossentropy loss function, the optimization method is selected as stochastic gradient descent (SGD), the initial learning rate is set as *learn_rate*, and the index to measure the model is chosen as accuracy. The amount of data selected during training is *epochs* epochs, and tensorboard is used as the callback function.

The cross entropy loss function is used to evaluate the difference between the probability distribution and the real distribution. It describes the distance between the actual output (probability) and the expected output (probability), that is, the smaller the value of cross entropy is, the closer the two probability distributions are.

Cross entropy loss function: set y as the expected output of the model and a as the actual output of the neural network, where $a = \sigma(z)$ and $z = \sum W_j + X_j + b$, B is the offset, W is the weight, and X is the input value. Then the cross entropy loss function is defined as:

$$C = -\frac{1}{n}\sum_x [y \ln a + (1-y)\ln(1-a)]. \tag{4}$$

Because $Z(W_j)$ is continuous and has no breakpoint, then $a = \sigma(z)$ is also continuous and differentiable with a value of 0-1, so $\ln(a)$ and $\ln(1-a)$ are also continuous and differentiable, so $C(W_j)$ is a differentiable function.

So we derive formula (4) as follows:

$$\frac{\partial C}{\partial W_j} = \frac{1}{n}\sum_x X_j(\sigma(z) - y). \tag{5}$$

It can be seen from formula (5) that there is no $\sigma'(z)$ term in the derivative, and the updating of the weight is affected by the $\sigma(z) - y$ term, that is, by the error. Therefore, when the error is large, the updating of the weight is fast, and when

the error is small, the updating of the weight is slow. So it can overcome the problem that the weight of variance cost function updating is too slow. Therefore, using cross entropy, loss function can quickly locate the position of the weight, reduce the training time, and ensure that all the weights will not be lost, thus making the experiment more stable. And it has the following properties.

Property:

a. Nonnegativity (so our goal is to minimize the cost function.)
b. When the real output A is close to the expected output y, the cost function is close to 0. (for example, when $y = 0$, $A \to 0$; $y = 1, A \to 1$, the cost function is close to 0).

In machine learning, we hope that the prediction data that the model learns from the training data will be distributed closer to the real data, so we often use the cross entropy loss function to calculate the binary loss function.

When we use neural network to analyze protocol syntax, we usually need a large-scale training set. When we use batch gradient descent, the calculation will be very large. At this time, we use random gradient descent method instead of batch gradient descent method.

SGD algorithm is to randomly select a group from the samples, update once according to the gradient after training, then extract another group, and then update once. In the case of large sample size and large sample size, it may not need to train all the samples to get a model with acceptable loss value.

SGD algorithm can analyze the results quickly when the sample size is large. Moreover, the time complexity of SGD is basically stable at $O(KNP)$, where K is the number of iterations and P is the average number of nonzero features of each sample. Using the random gradient descent method, although the accuracy will decline and may take many detours, the overall trend is toward the minimum loss value, which will save a lot of time and the algorithm is faster.

Next, we use Wireshark data to predict the prediction results. We put the remaining pieces of data into the test set to test and output their labels and accuracy. When we convert the labels, because the similarity is stored in the model prediction, and the label with the highest similarity can be identified as the protocol label. Therefore, we only need to find the location with the highest similarity and find its location. The protocol type represented is *OK*. Finally, the results are compared with ours. We can identify the unknown protocol, and then output the predicted protocol type and similarity.

4 | EXPERIMENTAL CONFIGURATION AND RESULT ANALYSIS

Although the UDP protocol includes DNS, OICQ, SSDP, and TCP, but for the protocol data frame, the header of each protocol must be the format of the protocol, so we can separate the upper protocol and the lower protocol for training and identification. Moreover, if the upper layer message data is put together with the lower layer message, it will increase the difficulty of setting the label. Therefore, this article treats UDP separately from the upper protocol.

In this article, the experimental environment is Windows 7 system, the programming language is python, the platform is visual studio code, and the neural network learning framework is keras framework. 240 000 known data used in this article are from the data obtained by Wireshark packet capturing tool, and the remaining 7029 unknown protocols are grabed at Jiangsu Province, Henan Province, Shaanxi Province, and Shandong Province, China.

In this article, we set $nbytes = 320$, so the $nimage = nbyte/8 = 40$. In this article, eight kinds of protocols are selected for recognition, so $N = 8$. The test dataset D=[[1,0,0,0,0,0,0,0], [0,1,0,0,0,0,0,0], [0,0,1,0,0,0,0,0], [0,0,0,1,0,0,0,0], [0,0,0,0,1,0,0,0], [0,0,0,0,0,1,0,0], [0,0,0,0,0,0,1,0], [0,0,0,0,0,0,0,1]] include eight kinds of tags, respectively, corresponding to ARP-like protocol, DNS-like protocol, HTTP-like protocol, ICMP-like protocol, OICQ-like protocol, SSDP-like protocol, TCP-like protocol, and UDP-like protocol.In CNN module, we set $kernel\_size = 3$, $filters_1 = 64$, $strides = 1$, $input\_ranges = 5 \times 8 \times 240000$ and $pool\_size = 2$. *dropout* regularization method is necessary, we set $dropout = 0.25$. The number of the second convolution neural network filters $filters_2 = 128$. There are $nodes_1 = 128$ nodes in the first layer of fully connected network, and the last layer has $nodes_2 = 8$ nodes. In Result output module, the learning rate $learn\_rate = 0.1$, and the epochs $epochs = 100$.

TABLE 2 Protocol dataset

Protocol type	Total number of data frames (Pieces)	Total data frame size (KB)
ARP-like protocol	30 000	2639
DNS-like protocol	30 000	2560
HTTP-like protocol	30 000	2600
ICMP-like protocol	30 000	2567
OICQ-like protocol	30 000	2545
SSDP-like protocol	30 000	2763
TCP-like protocol	30 000	2595
UDP-like protocol	30 000	2565
Train Set	240 000	20 834

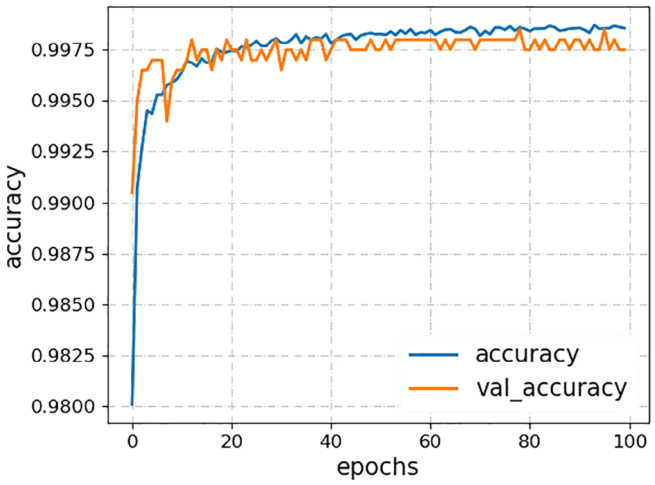


FIGURE 3 Accuracy comparison between training set and test set

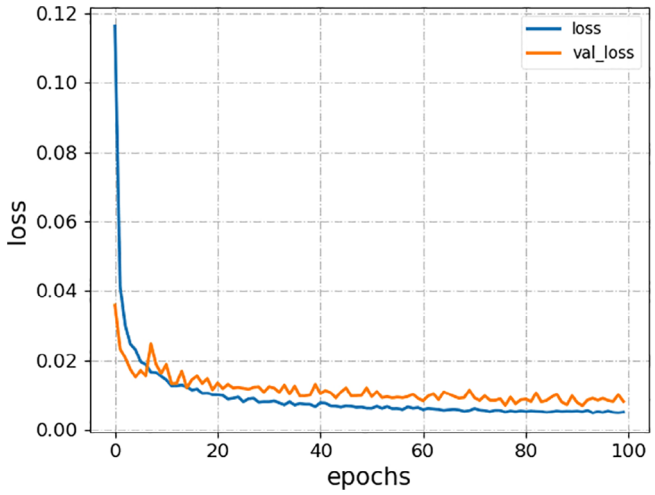


FIGURE 4 Loss comparison between training set and test set

The total amount of data used in this article is as shown in Table 2. Put all protocols together to get the train data set, randomly disorder the sequence of the train data set, and then take the first 192 000 pieces of the scrambled sequence for training and the last 48 000 pieces for testing.

After training the protocol and testing 6029 unknown protocols, the results are shown in Figures 3 and 4. It can be seen from the figure that the CNN method is still very good for the recognition of unknown protocols, with the recognition rate above 99%.

Because every unknown protocol format is fixed, it is in the head of bitstream protocol frame. After the protocol data are converted into image, each protocol format will be placed in the upper part of the image, and the characteristics of

FIGURE 5 Training set accuracy comparison

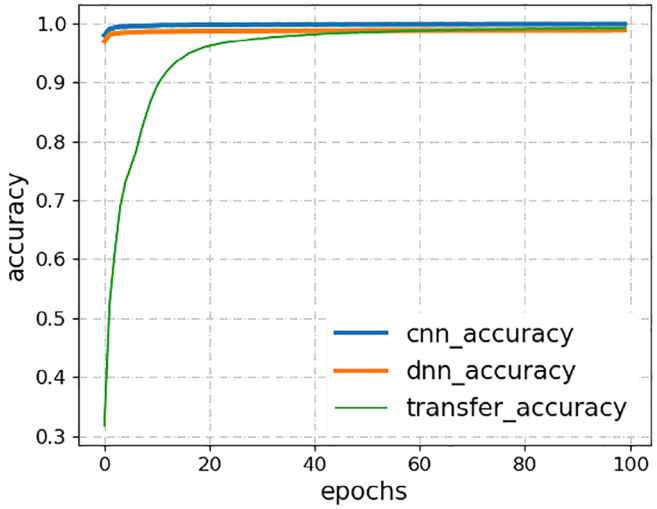
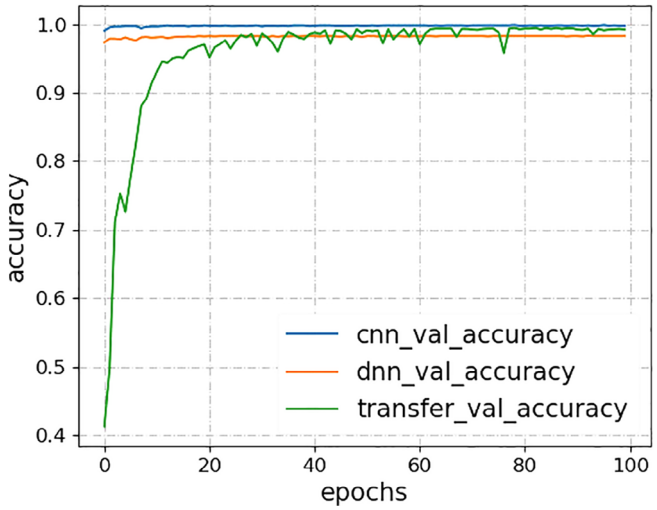


FIGURE 6 Test set accuracy comparison



the image will be very obvious. Volume and neural network are very effective for image processing, so using volume and neural network to analyze the protocol syntax will quickly converge.

During the experiment, we also compared the model with other models, including DNN algorithm and transfer learning algorithm. The experimental results including the comparison of training sets are shown in Figures 5 and 6. From the figure, we can see that the performance difference between CNN and DNN is not much, CNN is about 2% higher than DNN, and the accuracy of migration learning is obviously lower than CNN and DNN in the early stage, while In later tests, CNN and DNN were not stable.

In order to compare CNN and DNN more clearly, we compare the accuracy of CNN and DNN separately. The results are shown in Figure 7. From the figure, we can see that the accuracy of CNN is slightly better than that of DNN, about 2% higher than that of DNN.

Then, we also compare the accuracy of CNN with that of DNN and combine frequent item (CFI) algorithm using feature extraction. We take five kinds of data for testing. The accuracy results of protocol recognition are shown in Figure 8. From this, we can see that using CNN to analyze unknown protocol syntax is about 15% higher than that of CFI algorithm on average and about 2% higher than that of DNN.

Through the comparison between the CNN method proposed in this article and the recognition technologies of NDPI and LibProtoident algorithms, the validity of this method is verified. The recognition rate of different algorithms is shown in Figure 9 .

From Figure 9, it can be seen that the algorithm in this article takes the original network data as the input, extracts the features through CNN. It has a good classification effect for most protocols and can perform a good syntax analysis

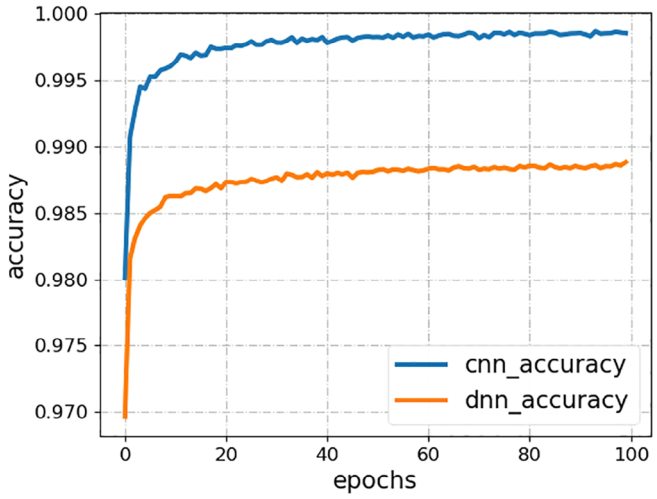


FIGURE 7 Detailed comparison figure of test set accuracy

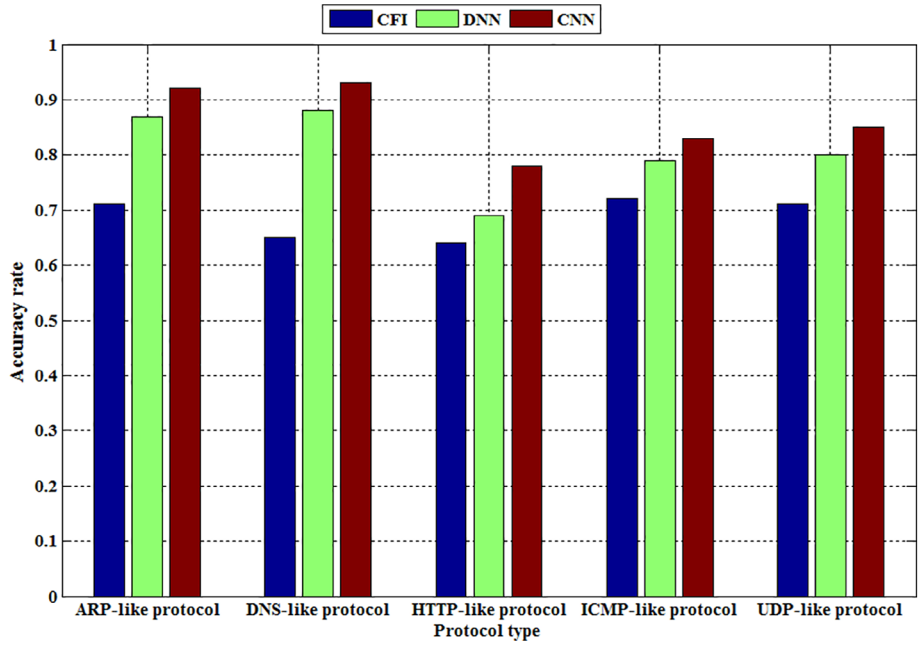


FIGURE 8 Protocol identification accuracy

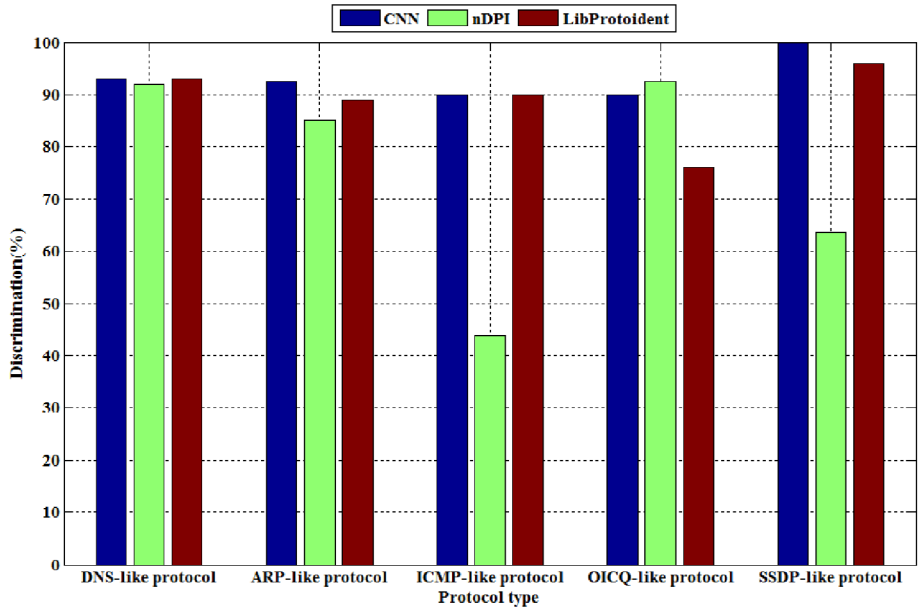


FIGURE 9 Recognition results of different algorithms

for unknown protocols. Compared with the other two traffic recognition methods, it has a certain improvement in the recognition rate, which proves the effectiveness of the algorithm in this article.

5 | CONCLUSIONS

Based on the analysis of the characteristics of the current bitstream protocol data format, this article proposes a method of unknown protocol syntax analysis and recognition based on CNN. For the captured protocol, the protocol data are preprocessed, including data cutting and splicing and data format conversion. Then the image is transformed. Next, the converted image is input to the convolution layer for convolution. After convolution, the data are pooled and flattened. Then the flattened data are put into the fully connected neural network. Finally, the unknown protocol is analyzed and predicted. Compared with the traditional clustering algorithm and other machine learning algorithms, we find that CNN is more accurate than CFI in unknown protocol syntax analysis. After calculation, the convolution neural network method is 15% higher than CFI clustering algorithm, and it can accurately analyze and identify the unknown protocol.

ACKNOWLEDGMENTS

This research work is supported by the National Key R&D Program of China (2018YFB1201500), National Natural Science Founds of China (61602376, 61773313, 61602374, 61702411), National Natural Science Founds of Shaanxi (2017JQ6020, 2016JQ6041), Key Research and Development Program of Shaanxi Province (2017ZDXM-GY-098, 2019TD-014), and Science Technology Project of Shaanxi Education Department (16JK1573, 16JK1552).

ORCID

Binbin Bai https://orcid.org/0000-0002-5448-4907

REFERENCES

1. Perera C, Zaslavsky AB, Christen P, et al. context aware computing for the internet of things: a survey. *IEEE Commun Surv Tutor*. 2014;16(1):414-454.
2. He D, Chan S, Guizani M. Security in the internet of things supported by mobile edge computing. *IEEE Commun Mag*. 2018;56(8):56-61.
3. Garg S, Singh A, Kaur K, et al. Edge computing-based security framework for big data analytics in VANETs. *IEEE Netw*. 2019;33(2):72-81.
4. Yu W, Liang F, He X, et al. A survey on the edge computing for the internet of things. *IEEE Access*. 2018;6:6900-6919.
5. Wright CV, Monrose F, Masson GM. On inferring application protocol behaviors in encrypted network traffic. *J Mach Learn Res*. 2006;6(4):2745-2769.
6. Sija B D, Goo Y H, Shim K S, et al. Protocol reverse engineering methods for undocumented ethernet and wireless protocols; survey. Paper presented at: Proceedings of the Symposium of the Korean Institute of Communications and Information Sciences; 2017.
7. Duchene J, Le Guernic C, Alata E, et al. State of the art of network protocol reverse engineering tools. *J Comput Virol Hacking Tech*. 2018;14(1):53-68.
8. Lin R, Li O, Li Q, et al. Unknown network protocol classification method based on semi-supervised learning. Paper presented at: Proceedings of the 2015 IEEE International Conference on Computer and Communications (ICCC); 2015:300-308; IEEE.
9. Lin H, Yan Z, Chen Y, et al. A survey on network security-related data collection technologies. *IEEE Access*. 2018;6:18345-18365.
10. Fan Y, Zhu Y, Yuan L. Automatic reverse engineering of unknown security protocols from network traces. Paper presented at: Proceedings of the 2018 IEEE 4th International Conference on Computer and Communications (ICCC), 2018:1139-1148; IEEE.
11. Tang H, Xiao B, Li W, Wang G. Pixel convolutional neural network for multi-focus image fusion. *Inf Sci*. 2018;433:125-141.
12. Takabi H, Joshi J, Ahn G, et al. Security and privacy challenges in cloud computing environments. *IEEE Symp Sec Priv*. 2010;8(6):24-31.
13. Brüsch A, Nguyen N, Schürmann D, Sigg S, Wolf L. Security Properties of Gait for Mobile Device Pairing. *IEEE T Mobile Comput*. 2020;19(3):697-710.
14. Hei X, Bai B, Wang Y, et al. Feature extraction optimization for bitstream communication protocol format reverse analysis. Paper presented at: Proceedings of the 2019 18th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/13th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE). 2019:662-669; IEEE.
15. Shen Z, Lee PP, Shu J, et al. Encoding-aware data placement for efficient degraded reads in xor-coded storage systems: algorithms and evaluation. *IEEE Trans Parall Distrib Syst*. 2018;29(12):2757-2770.
16. Cheng Y, Wang F, Jiang H, et al. A communication-reduced and computation-balanced framework for fast graph computation. *Frontiers Comput Sci China*. 2018;12(5):887-907.
17. Lin B, Guo W, Xiong N, et al. A pretreatment workflow scheduling approach for big data applications in multicloud environments. *IEEE Trans Netw Serv Manag*. 2016;13(3):581-594.

18. Wiatowski T, Bolcskei H. A mathematical theory of deep convolutional neural networks for feature extraction. *IEEE Trans Inf Theory*. 2018;64(3):1845-1866.
19. Sabokrou M, Fayyaz M, Fathy M, et al. Deep-anomaly: fully convolutional neural network for fast anomaly detection in crowded scenes. *Comput Vis Image Underst*. 2018;172:88-97.
20. Wang Z. The applications of deep learning on traffic identification. *BlackHat USA*. 2015;24(11):1-10.
21. JAIN A V. Network traffic identification with convolutional neural networks. Paper presented at: Proceedings of the IEEE 16th International Conference on Dependable, Autonomic and Secure Computing, 16th International Conference on Pervasive Intelligence and Computing,4th International Conference on Big Data Intelligence and Computing and Cyber Science and Technology Congress. Piscataway; 2018:1001-1007; IEEE.
22. Li X, Ding Q, Sun JQ. Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliab Eng Syst Saf*. 2018;172:1-11.
23. Paoletti ME, Haut JM, Plaza J, et al. A new deep convolutional neural network for fast hyperspectral image classification. *ISPRS J Photogramm Remote Sens*. 2018;145:120-147.
24. Xiao B, Wang K, Bi X, Li W, Han J. 2D-LBP: an enhanced local binary feature for texture image classification. *IEEE Trans Circuits Syst Video Tech*. Sept. 2019;29(9):2796-2808.
25. Jianqiang Z, Xiaolin G, Xuejun Z. Deep convolution neural networks for twitter sentiment analysis. *IEEE Access*. 2018;6:23253-23260.
26. Nogueira R, Lotufo RD, Machado RC, et al. Fingerprint liveness detection using convolutional neural networks. *IEEE Trans Inf Forens Sec*. 2016;11(6):1206-1213.
27. Hezaveh Y, Levasseur LP, Marshall PJ, et al. Fast automated analysis of strong gravitational lenses with convolutional neural networks. *Nature*. 2017;548(7669):555-557.
28. Zou J, Dong L, Wu W. New algorithms for the unbalanced generalised birthday problem. *IET Inf Secur*. 2018;12(6):527-533.
29. Wang J, Zhang X, Lin Y, et al. Event-triggered dissipative control for networked stochastic systems under non-uniform sampling. *Inf Sci*. 2018;447:216-228.
30. Hu H, Tang B, Gong X, et al. Intelligent fault diagnosis of the high-speed train with big data based on deep neural networks. *IEEE Trans Ind Inform*. 2017;13(4):2106-2116.
31. Wang P, Li W, Gao Z, et al. Depth pooling based large-scale 3-d action recognition with convolutional neural networks. *IEEE Trans Multimedia*. 2018;20(5):1051-1061.

How to cite this article: Wang Y, Bai B, Hei X, Zhu L, Ji W. An unknown protocol syntax analysis method based on convolutional neural network. *Trans Emerging Tel Tech*. 2020;e3922. https://doi.org/10.1002/ett.3922