International Journal of Security and Its Application
Vol.11, No.1 (2017), pp.93-108
http://dx.doi.org/10.14257/ijsia.2017.11.1.09

Network Security Situation Prediction System Based on Neural Network and Big Data

* Bowen Zhu[1], Yonghong Chen [1], Yiqiao Cai [1], Hui Tian[1]and Tian Wang[1]

[1]*College of Computer Science and Technology, Huaqiao University, Xiamen, 361021, China*
1511314013@ hqu.edu.cn

Abstract

In today's big data era, traditional methods are of low efficiency in handling network security matters, and most of the time they even don't work. The system studied in this paper, the network security situation analysis and prediction system based on neural network is designed and implemented on the Hadoop platform. By collecting distributed data and decreasing their dimensions, this system reduces the complexity of data to realize efficient processing of big data. We adopt the optimized K-Means clustering analysis algorithm to simplify the data, and we utilized the optimal association rules mining method to find threats and risks existing in the network. The above part is the network security situation analysis. On the basis of network security situation analysis, a new method based on time dimension is used to forecast the future network security situation. By blending part of predictive results and adjusting error, the system realizes security situation prediction of the whole network and a self-improving neural network, thus ensuring a higher accuracy rate. The experimental result we obtained is that the time we spend is just 12% of what consumed by the traditional method in the same amount of data. We can draw the following conclusions: 1) the system proposed in this paper can effectively save time of handling big data 2) as the amount of data increases, this system will not reduce the accuracy rate but gets 95% correct.

Keywords*: Hadoop; Big Data; Parallel Machine Learning; Network Security Situational Prediction*

1. Introduction

The rapid development of computer technology and the fast growth of user demand have brought a wider application of computer network. In the increasingly complex computer network environment and the dynamically changing situation [1], obtaining the current network security status and forecasting its future trend can provide guidance for network security administrators about safety operation and decision-making, thus improving the initiative of network defense. At the same time, in the era of big data, network security incidents emerge in an endless stream and traditional single defense or test equipment cannot meet the security requirements. Traditional network security situation analysis methods being able to integrate multiple safety factors to reflect the dynamic network security situation on the whole and making the prediction of security situation but still seem powerless to big data. Therefore, the network security situation analysis and prediction based on big data is a hotspot in the field of network security.

Hadoop[2] is a top-level project in the Apache foundation. It can carry out the distributed computing and parallel processing of massive data. Under the drive of the Internet Company, Hadoop becomes more and more applied and the entire ecosystem tends to be mature and perfect. The current version 2.0 has a qualitative leap than last

[1] Bowen Zhu is the corresponding author.

ISSN: 1738-9976 IJSIA

generation version. In July 2015 Apache officially released the latest stable version (Hadoop 2.7.1) and our system is implemented in this version.

2. Related Technology

In this chapter, we will introduce the research contents and related methods.

The problem to be solved in data dimension reduction technique is to find the part which can represent the most important feature of the whole data set in a series of high dimensional data, so as to generate a low dimensional data set for easy processing and reducing the complexity of the data. In the dimension reduction technique, the Principal Component Analysis (PCA) [3],Eigen Decomposition [4] and Singular Value Decomposition (SVD) are widely used. Our system chooses to use the Eigen Decomposition algorithm whose idea is to regard the data set as matrix A. The eigenvectors of the matrix AA^T or A^TA will be solved and the axis corresponding to the biggest eigenvector (the main feature) in the result is the direction of the maximum variance of the data. The feature of the data set is most obvious in this direction and the decomposition of the data set can get a low dimensional data set represented by the main feature.

Clustering analysis algorithm is an important research topic in data mining and machine learning. At the same time, it is the basis of the data processing algorithms. However, big data has brought great challenges to traditional machine learning and data mining. So our system uses the traditional clustering algorithm: K-Means algorithm to realize the distributed parallel processing. The ultimate goal of the algorithm is to partition the entire data set into clusters and to have relatively high similarity in the data within the clusters, but each cluster is separated from each other. The initial point selection by the clustering algorithm is not stable but random, thus causing instability of the clustering results. In order to solve the problem, we use the Particle Swarm Optimization (PSO) algorithm to the cluster center for overall optimization[5].

At the Frequent itemsets mining algorithm, the frequent item can be regarded as the degree of association of two or more features. In the data set, the most frequent co-occurrences the more related. When the co-occurrences number reaches a certain threshold, it is called frequent item. Frequent Pattern-growth algorithm [6] based on its subset is an optimal algorithm for mining association rules. In the network attacks, attackers often take a variety of means of attack. At the same time, the network often presents multiple attack features. Mining the relationship between them and carrying out analysis of the network security situation.

After the analysis of the network security situation, the network security situation prediction based on the time dimension [7] is carried out. Attacks always aim at various vulnerabilities existing in the network, counting up the probability of the emergence of each attack according to the analysis results and adding up a random value (indicated some unexpected situation in the network) , then compare with the threshold. If it exceeds the threshold, it is considered that the current network is in a state of danger.

Hadoop 1.0 is a lack of scalability, resource utilization, and fault tolerance *etc*. Apache upgrades the MapReduce in Hadoop 2.0 and builds an independent general system YARN (Yet another Resource Negotiator) for the unified management and deployment of resources. The difference between the two versions of the Hadoop is shown in Figure 1.

From the figure, we can see that the upgrade of Hadoop not only can be carried out offline MapReduce programming but also supports more programming framework. The resource management and schedule are separated which makes the cluster more efficient.

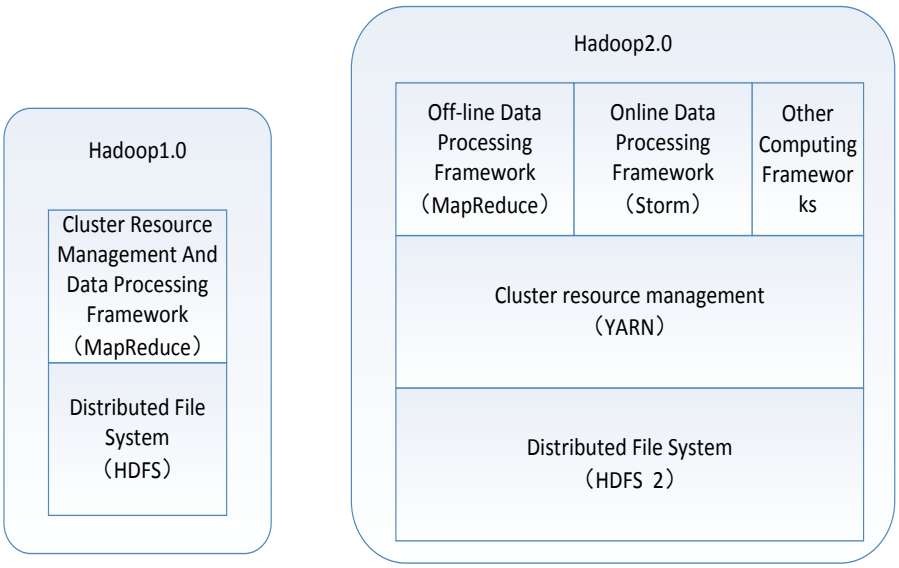

Figure 1. Hadoop Version Comparison

3. System Design

The neural network is very slow in the non-parallel processing system, so our system combines the characteristics of Hadoop distributed platform and the structure of BP neural network to design[8]. System structure as shown in Figure 2, this is a three layer of the horizontal structure of the neural network. It is composed of the input layer, hidden layer, and output layer, between each layer of network data flows in two directions.

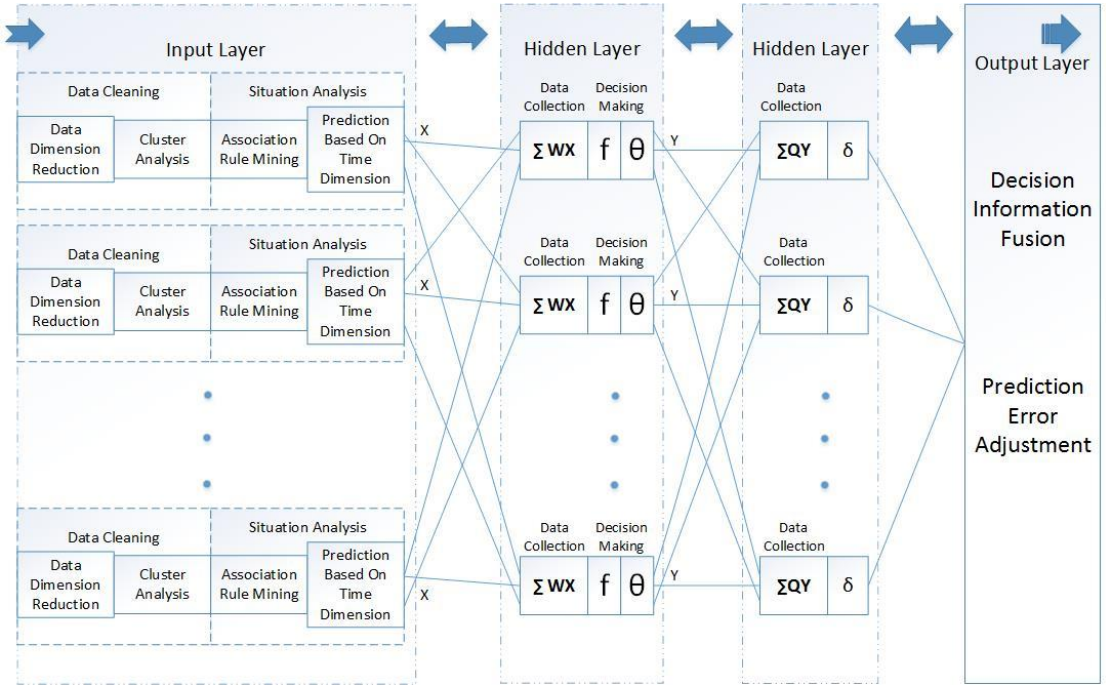

Figure 2. Network Security Situation Prediction System

3.1. Input Layer

The input layer is the data collection and event analysis unit of the system, it is composed of two parts: the data cleaning module and the network security situation analysis module.

The function of data cleaning module is to collect the data from the bottom of the system so that the data is conducive to the subsequent processing and analysis. In the process of data cleaning, the first thing is to reduce the dimension of data. In the era of big data, the amount of data collected is huge and the dimension of data is amazing. Therefore, Eigen Decomposition is used to reduce the dimension of data, collect the main features of the data, reduce the amount of data processed by the system and improve the system efficiency. The main feature of the data after dimension reduction are obvious, but the distribution of the data is messy and irregular, it is not conducive to the situation analysis. So the clustering analysis is continued to further strengthen the characteristics of the data and improve the accuracy of the data. PSO algorithm is used to optimize the K-Means algorithm which used in our system. Using PSO algorithm's global searching ability to make up for the k-means algorithm's disadvantages in clustering center selection in early operation will effectively avoid local optimum problems.

In the network security situation analysis module, the frequent itemset mining algorithm is adopted to complete the overall analysis of the security situation on basis of association rules in the data. It can effectively overcome the one-sidedness of single detection. A preliminary situation prediction module based on the time dimension is added to the system on the basis of the above situation analysis. When analyzing the data of the decision-making module, it is considered to be dangerous, if the probability of all kinds of risks and threats goes beyond a threshold. According to the value of the output exceeding the threshold range, the value of the risk degree is quantified. The quantized value X is the output of the input layer.

3.2. Hidden Layer

The hidden layer is divided into two parts: data receiving module and decision module. Each input layer and the hidden layer are connected to each other, but there is no correlation between the hidden layer and the hidden layer. In the hidden layer, there is a numerical value W which indicates the weight of the connection strength between the hidden layer and the input layer. The data receiving module of the hidden layer receives the decision result of each input layer and calculates the $\sum WX$, then transfers the results to the judgment module. Through the operation of the input value, the core decision method of the judgment module will draw a local security situation to make a further network security situation prediction. The predicted value Y is the output of the hidden layer.

3.3 Output Layer

The output layer and the hidden layer also have a weight value Q. The output layer not only receives the predictive value of the hidden layer and calculates the $\sum QY$ but also compares the calculated results with the threshold. Then make the final prediction of the local security situation. If it is beyond the threshold, it is considered to be dangerous, otherwise, it is safe.

3.4 Decision Fusion Prediction Module

Through sending all of the prediction results of the output layer to the information fusion prediction module, the entire network security prediction information is fused and the overall network security situation prediction is obtained.

3.5 Self Learning Error Adjustment

Just same as a traditional neural network, our system also needs to adjust the error. When the decision fusion prediction module concludes the overall security situation is dangerous, it indicates that the next phase of the network state is at least as dangerous and even more dangerous as it is now. So we need to modify the weights between the input layer and the hidden layer. At the same time, modifying the weights between the hidden layer and the output layer. The two weights are increased, so that more dangerous situations can be found. This also helps the security administrator to take the next step. On the contrary, the two values are reduced. The accuracy of prediction is improved by the self - learning error adjustment.

4. System Implementation

HDFS (Hadoop Distributed File System) and MapReduce are the core of Hadoop. The system structure of the whole Hadoop is mainly through the HDFS to realize the distributed storage bottom support. And it can realize the support of offline distributed parallel task processing program through the MapReduce computing framework. Through the YARN to carry out the system of resource allocation and data allocation to be processed, so as to achieve the dynamic balance of resources and giving full play to the effectiveness of the system.

4.1. Distributed Programming Framework

MapReduce is a kind of distributed computing and programming framework which is transparent to users. Developers do not have to care about the specific "map" "reduce" process and the task will facilitate disassembling pushed to each node in the cluster to complete the rapid processing of massive data. This reflects the performance of the Hadoop platform in the process of high efficiency, high scalability, and high tolerance.

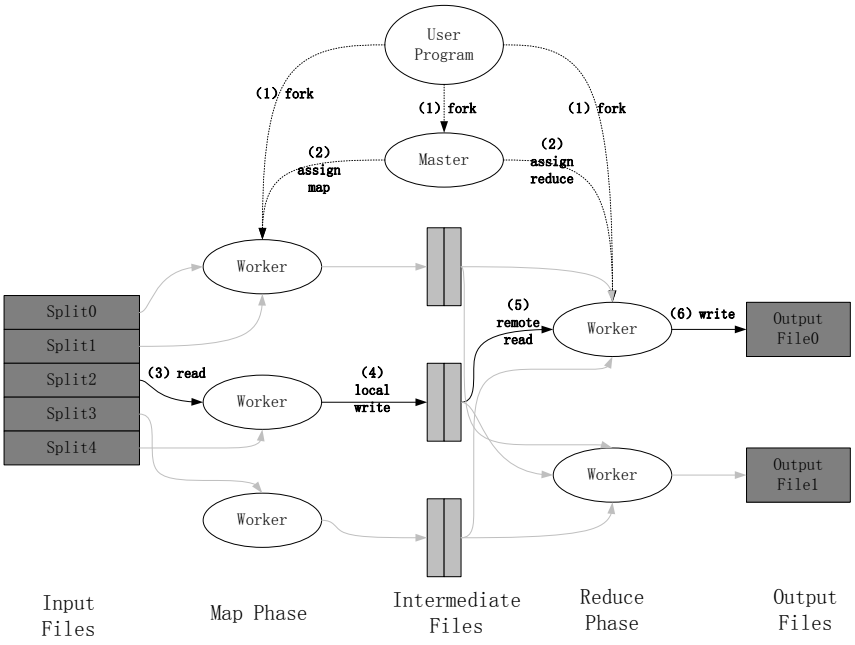

Figure 3. MapReduce Working Principle

MapReduce applications mainly include the following three parts: Map function, Reduce function, and Main function. The Map function in the distributed file system

(HDFS) is used to achieve massive data and tasks decomposition. The Reduce function is used to complete tasks and data of polymerization. The main function handles the processing program code and configuration files to drive the operation of the MapReduce program.

The key of MapReduce computing framework is Map and Reduce functions. The data processing process is described in Figure 3. Each step is explained as follows:

> ***Step1****: input files are divided into m part (m is a user – defined value), each one usually have 16MB to 64MB, such as shown in the figure on the left is divided into split0~4 (file blocks). Then use the fork to copy user process to the other machines in the cluster.*
> ***Step2****: user's program has two part, one called master, and the rest is called worker. Worker is responsible for scheduling. If there are free workers will be assigned Map task or Reduce task. Number of workers can be specified by the user.*
> ***Step3****: worker begin to implement the Map phase. Firstly, reading the file corresponding to the block of input data and map number, then split one-to-one correspondence. The map function produces intermediate key value pairs to be cached in memory.*
> ***Step4****: the middle of the key cache will be periodically written to the local disk.*
> ***Step5****: store the results of Map task to one of several local files (later each file will assign a Reduce task).*
> ***Step6****: Reduce phase traversal of all intermediate key-value pairs, then output the final result*

In the Map phase, data in the form of <key 1, value 1> as the input of the map function. When Data handled in the map function, it will be converted for the new <key 2, value 2 > on the output. Polymerizing the value of the output of Map in accordance with the key. The results of the aggregate will be as an input of the reduce function, then the output < key 3, value 3 > of the reduce function is the final result. Algorithm process is shown in Figure 4.

Map：<key1, value1> –> list<key2, value2>
Reduce: <key2, list (value2)> –> list<key3, value3>

Figure 4. MapReduce Coding process

The Map input parameters: <key1, value1> representation of data. Corresponding processing logic is a record data (such as text files in a row) will be the key-value to spread the map function. The Reduce input parameter is composed of a set of intermediate results of Map output.

The algorithms proposed in this paper to deal with the network security situation analysis of big data are in this form on the Hadoop platform to achieve parallel operation and data processing. They are processed in parallel by the way of Figure 3, which is simple and efficient.

4.2. Data Cleaning Module

Data cleaning is divided into data dimension reduction and clustering analysis. MapReduce distributed programming framework is used in the Hadoop platform to achieve these two processes. Big data is not only referred to a large amount of data but also referred to the data dimension is high, it is difficult to analyze. The dimension of the input data is very high, if the cluster analysis is conducted directly, it will consume a lot

of time. If there is a way to reduce the original dimension, the amount of operation will be reduced and the efficiency will be improved. Therefore, firstly we reduce the dimensionality of the data and then cluster analysis. In the process of big data, this is an effective way.

In the Eigen Decomposition algorithm, the eigenvectors and the eigenvalues of the covariance matrix are mainly required. For each eigenvalue and eigenvector can be derived according to the formula, so we can get the corresponding eigenmatrix. So as to realize the dimension reduction of the sample data. Its distributed implementation process is as follows.

Eigen Decomposition algorithm is a decomposition method that can be used to the diagonalization matrix (the number of rows equal to the number of columns). Usually, we do not deal with the data of the matrix because the number of rows is far greater than the number of columns. These data need to preprocess. The data preprocessing process is shown in Figure 5.

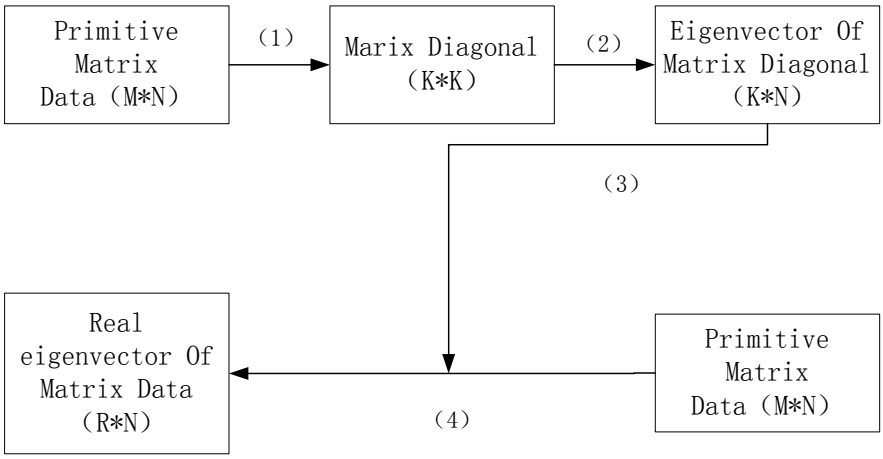

Figure 5. Data Preprocessing Process

***Step1**: the temporary diagonal matrix diag (K*K) is obtained by the transformation of the original matrix data (M*N).*
***Step2**: the eigenvalues and eigenvectors are obtained by the eigen decomposition of matrix diag (K*K).*
***Step3**: transform the eigenvector and the original matrix.*
***Step4**: get the matrix that needs to be processed.*

In K-Means clustering algorithm based on Particle Swarm Optimization (PSO-KM), every particle iteratively searches for K optimal cluster centers. The parallel implementation of the K-Means clustering algorithm optimized by PSO is shown in Figure 6.

***Step1**: all the data are scanned in the datasets, then select the k points as the initial cluster centers by PSO algorithm.*
***Step2**: all the data are clustered and the new cluster centers are found by computing the distance between each data to the cluster center, then the second stage is repeated until the condition is satisfied.*
***Step3**: all the data are divided and clustered according to the cluster center.*

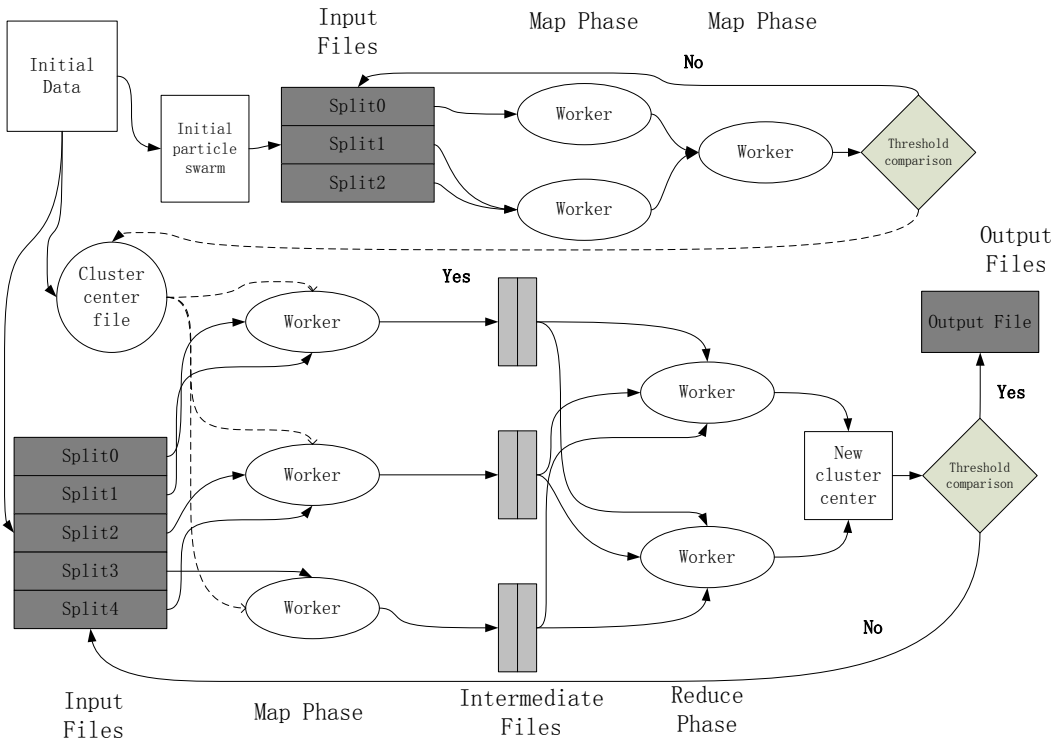

Figure 6. The Parallel Process of PSO Optimized K-Means Algorithm

4.3. Data Analysis Module

At present, the association rule algorithm has produced three algorithms: Apriori algorithm, FP-tree algorithm, and Eclat algorithm. (1) Apriori algorithm is the most influential algorithm for mining frequent itemsets of Boolean association rules. Its core is the recursive algorithm of mining frequent itemsets based on the two stage. However, with the increase of the data set, the computation cost of IO is greatly increased. (2) The inverted list idea is added to the Eclat algorithm, which can speed up the generation of frequent itemsets by converting the inverted list. The processing method of this algorithm is suitable for relational data. (3) The FP-tree algorithm is a method to mining frequent itemsets directly by the method of frequent pattern growth, which does not generate candidate patterns. This algorithm only needs to scan the data two times. In the face of big data, this is an absolute advantage. So our system uses this algorithm to mine the association rules in the MapReduce framework. The parallelization process is shown in Figure 7.

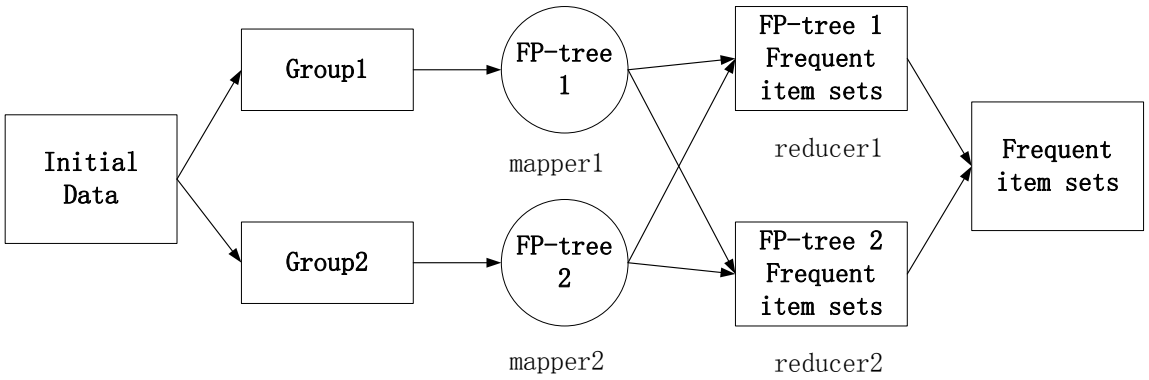

Figure 7. Parallel Frequent Pattern Mining Process

> ***Step1****: find out the one dimensional frequent itemset and coding in the initial data.*
> ***Step2****: group the initial data according to the one dimensional frequent itemset.*
> ***Step3****: construct the FP-tree for each group.*
> ***Step4****: carry out frequent itemsets mining on each FP-tree which is built.*
> ***Step5****: integrate the frequent itemsets which mining out of each FP-tree to get the final overall frequent itemset.*

Preliminary security situation prediction is carried out in the results. The prediction algorithm based on time dimension is adopted in our system. The flow of the algorithm is shown below.

1. *Statistics each attack type and then generate the known exception Library (KEL).*
2. *Map: enter all data sets*
3. *Reduce: AE (i) is the output of each attack behavior and the number of statistics.*
4. *If(AE(i)!=kel) KEL=KEL+AE(i);*
   Else {count i++};
5. *HAE=f(AE);*
6. *Map: enter new data sets*
7. *Reduce :< AE(i), number >*
8. *If(number > HAE(number)) think the current situation is dangerous*
   Else think the current situation is safe.
9. *Return current situation value X;*

In this algorithm, the first line, enter the experimental data and then count the existence of a security threat to generate a known exception library (KEL). This is the basis for making a preliminary decision. Second line and third line are the parallel module（input data and detect threats）based on the MapReduce framework. Where AE (i) represents the detection of the i kind of threat. In the fourth line, compare AE (i) with KEL if the AE (i) is known then count the number of appearances. If it is unknown then add it into KEL and count the number of appearances. In the fifth line, HAE represents a historical risk set. With the function of f () randomly select in the AE to generating the first generation of historical records. In the sixth and seventh lines, the new input data were processed to record the threat. Where number represents the number of times that AE (i) appears in this time period. In the eighth line, if the current threat is more than the number of times of the historical records, the current network situation is certainly dangerous. On the contrary, it is considered safe. The final output X is based on some of the data to make the situation prediction decision.

4.4. Decision Module

Self-learning decision-making module is based on neural network. A complete Back Propagation Neural Network [9] is composed of data cleaning module, data analysis module and a decision module. Each neuron in the neural network has plasticity, each neuron receives the output from each neuron in the previous layer, and fuses all of these inputs to make a comprehensive decision. Each decision will affect the adjustment of the threshold and weight of each neuron, which makes the decision errorless and makes the whole network more suitable for the current environment.

The neural network is a parallel structure with parallel processing capability, which makes it very slow in a non-parallel processing system, so the whole system needs to be implemented on Hadoop platform. The distributed storage of knowledge makes the neural network has a strong ability of fault tolerance. If anyone or a few neurons appear problem, it does not affect the whole system which is necessary when dealing with big data. Through training and learning to improve the ability of error feedback and adjustment so as to find a better relationship between the input and output. So that the decision-making error is getting smaller and making the decision result more accurate. The process of neural network algorithm is shown in Figure 8.

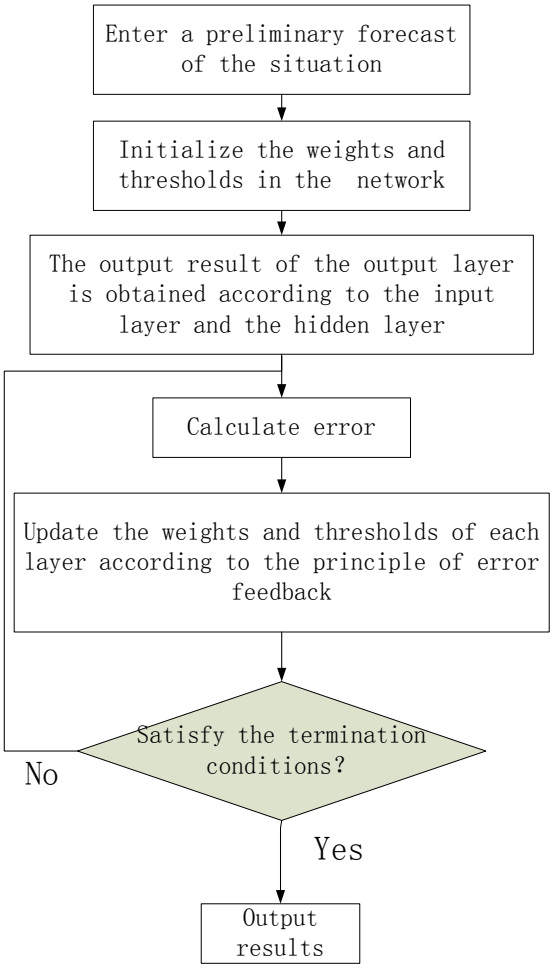

Figure 8. The Process of Neural Network Algorithm

The programming model based on MapReduce is shown below. In the Map phase, the output is calculated based on the input. Compare the actual output with the expected output to get the network learning error. The change of weight and threshold in the network is calculated based on the obtained learning error. In the Reduce phase, the output of the Map is used as input. Find the weights and thresholds of each neuron in each layer, then update them.

```
Step1: Map Phase
Input: entire dataset
Output; the corresponding weights and thresholds are corresponding to the values that need to be updated, such as this form <w, Δw>
{
        Calculate the output of the input layer;
        Calculate the output of the hidden layer;
        Calculate the output of the output layer;
        Calculate the learning error according to the actual output and the desired output;
        Update the weights and thresholds according to the learning error;
}

Step2: Reduce Phase
Input: input is the output of Map <w, Δ w>
Output: weights, thresholds and their corresponding updated values
{
        Find the weights and thresholds (w) which need to be updated;
        Add value Δ w;
        Calculate the value after the change, and then output it;
}
```

5. Experimental Results and Analysis

Our experimental data is used in the Lincoln Laboratory of Massachusetts Institute of Technology in 1989 to simulate the U.S. Air Force LAN environment, and thus obtain the network traffic test data set called KDD CUP 99. CAIDA data sets were obtained from the Center for Applied Internet Data Analysis. This dataset contains anonymized passive traffic traces from CAIDA's passive monitors in 2015. It contains traffic traces from the 'equinix-chicago' high-speed monitor.

In the KDD CUP 99 data set contains 4 types of intrusions, a total of 39 attacks. Professor Stolfo Sal from Columbia University and Professor Lee Wenke from North Carolina State University used data mining techniques to analyze the data set and data preprocessing. So the KDD CUP 99 data has been carried out various attacks and abnormal markers. Using this data set can easily and quickly verify the detection rate and precision. Our first experiment is to calculate the system's detection rate for each kind of intrusion behavior. A variety of intrusion detection rate as shown in Table 1.

Table . A Variety of Intrusion Detection Rate

Detection Rate of All Intrusion Types				
Type	**PROBE**	**DOS**	**U2R**	**R2L**
Detection rate	96.40%	94.98%	94.63%	94.91%

Detection Rate For Each Attack				
PROBE				
ipsweep	**mscan**	**nmap**	**portsweep**	**saint**
96.84%	96.61%	95.86%	95.86%	96.84%
satan				
94.08%				
DOS				
apache2	**back**	**land**	**mailbomb**	**neptune**
94.08%	94.15%	96.42%	94.39%	95.86%
pod	**processtable**	**smurf**	**teardrop**	**udpstorm**
94.24%	93.78%	96.84%	94.24%	95.86%
U2R				
buffer_overflow	**http tunnel**	**Load module**	**perl**	**ps**
94.43%	95.86%	94.46%	94.41%	94.48%
rootkit	**sqlattack**	**xterm**		
94.49%	94.62%	94.66%		
R2L				
ftp_write	**guess_passwd**	**imap**	**multihop**	**named**
94.39%	94.41%	94.38%	96.84%	96.84%
phf	**sendmail**	**snmpgetattack**	**snmpguess**	**spy**
95.86%	94.39%	94.15%	95.86	94.15%
warezclient	**warezmaster**	**worm**	**xlock**	
96.38%	94.45%	95.01%	94.33%	

It can be seen from the table that our system has a very good detection rate. The experimental results show that the average detection rate is 95%. This is a very good experimental result.

In order to further verify the detection rate of our system, we used the KDD CUP 99 test set to carry out another five experiments. The first group contains four types of abnormal data (Dos, Probe, R2L, U2R) and the normal data, the second group contains Dos attacks and the normal data, the third group contains Probe attacks and the normal data, the fourth group contains R2L attacks and the normal data, the fifth group containsU2R attacks and normal data. The normal data is greater than 75% in the total test data.

In order to make the assessment accurate and effective, the Error Detection Rate, the Correct Detection Rate, and Omission Detection Rate are measured in three aspects.

1) **Error Detection Rate**: the normal data is judged to be abnormal data.
2) **Correct Detection Rate**: detect the abnormal data of the data set correctly.
3) **Omission Detection Rate**: the abnormal data is judged to be normal data.

The experimental results are shown in Figure 9.

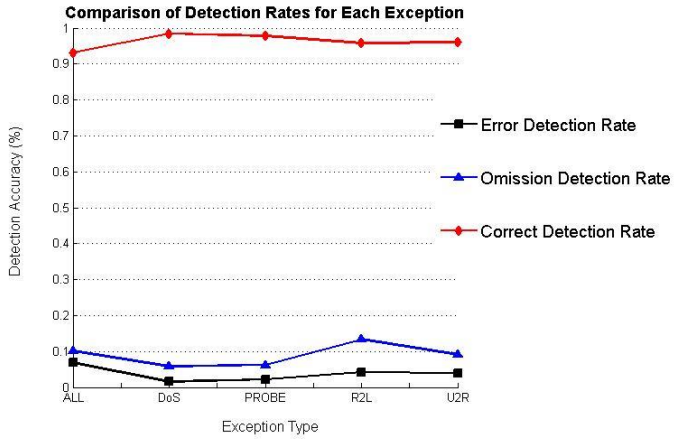

Figure 9. Comparison of Detection Rates for Each Exception

From the figure, we can see that the Correct Detection Rate is very high. Error Detection Rate is very low. But for the R2L Omission Detection Rate is a little high. The characteristics of each type of intrusion are different, so the detection effect will be different. However, the overall performance of the system is good.

KDD CUP 99 data set is much smaller than the CAIDA data set, In order to compare the running time of the system, we use the CAIDA data set to test the time of the parallel platform. The experimental results are shown in Figure 10.

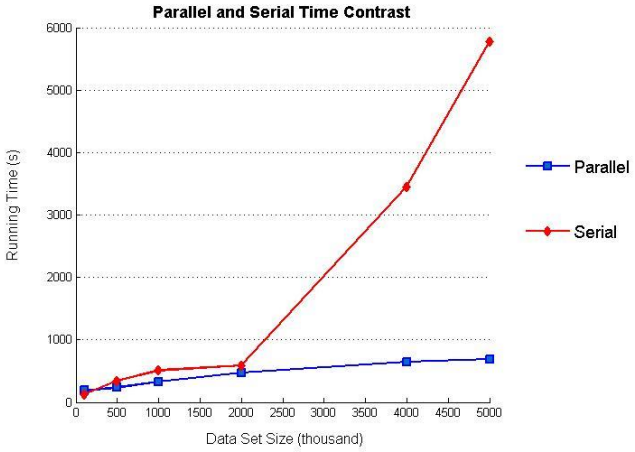

Figure 10. Parallel and Serial Time Contrast

As can be seen from the figure, the parallel implementation of the system consumes less time and is more efficient. So in the face of big data, parallel processing is a very good method.

In order to further prove the experimental results of this method, we carried out four sets of comparative experiments. In the same experimental environment, we used four different algorithms for the same data set. The experimental results are shown in Figure 11

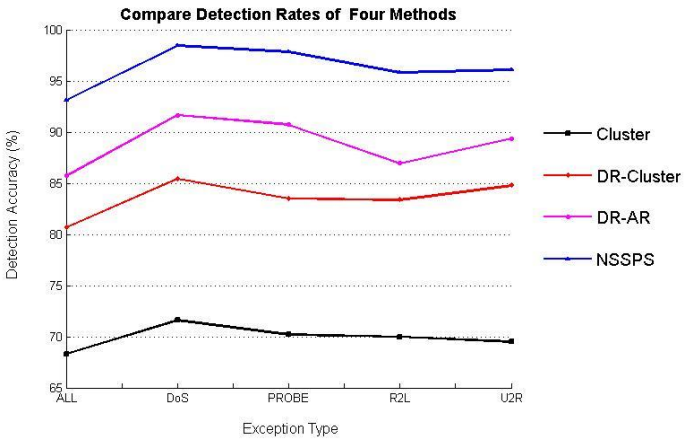

Figure 11. Compare Detection Rates of Four Methods

In the first method, we only use the parallel clustering algorithm for the analysis of the experimental data. It can be seen from the figure that the experimental results are very poor. Because the data dimension is high, which affects the accuracy of clustering. Firstly, we reduce the dimension of the data in the second method and then cluster the data. Detection rate has been greatly improved. After the data is reduced, the effect of the experiment is very good. So in the third method, we reduce the dimension of the data for association rules analysis. The fourth is the method proposed in this paper. Firstly, the data is cleaned and then the security situation is analyzed. Using the neural network to adjust the results of the preliminary analysis and get the final results. It can be seen from the figure that the detection rate of our system is the highest.

In order to verify the speedup of the parallel algorithm, we select 1-5 nodes as the experimental environment and the test is carried out. We select the largest amount of data in "Parallel and Serial Time Contrast" Experiment as the test data set. The calculation formula for the acceleration ratio is as follow R=T1/T2. T1: Time of serial consumption in a single machine environment, T2: Time consumed in a parallel environment. Finally, we get the results as shown in Figure 12.

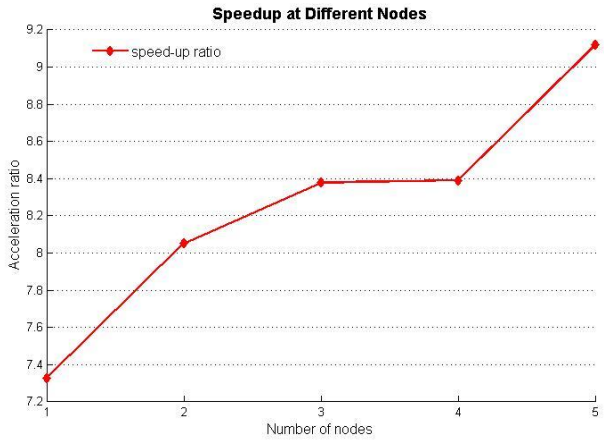

Figure 12. Speedup at Different Nodes

From the figure, we can see that with the increase of the number of nodes the speedup of the algorithm also increased. However, when the number of nodes is

from 3 to 4, the growth rate becomes slow. The reason for this situation is that with the increase of the number of nodes the communication between the nodes increased. Communication between nodes will consume a number of resources and spend the amount of time.

From our experiments, we can see that the system proposed in this paper has a very good effect on Correct Detection Rate, Error Detection Rate, and time efficiency.

6. Summary and Outlook

The network security situation prediction system based on big data and the neural network is realized in this paper. Our system is optimized and improved for a variety of algorithms to realize the network security situation analysis and prediction. We realize the data distributed storage and processing by integrating the Hadoop platform and the neural network. It overcomes the shortcomings of traditional network security situation prediction through the self-learning ability of error feedback. By using the parallel ability of Hadoop platform, it overcomes the bottleneck of the whole system and improves the ability to process big data. The results of our experiments are also very good. The experimental results show that our system has a high speed and high accuracy in dealing with big data.

Although the system has realized the data off-line analysis through the MapReduce framework. In the situation of network security, sometimes it is needed to handle the data collected and make decisions in real time. Therefore, the system can be extended to the Storm or Spark platform to make up the deficiencies of the system in the field of real-time processing. Most of the data needed to be processed in real time exist in the form of data flow, so the storage and processing of the stream data are another problems to be solved.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (NO.61370007, 61572206, U1405254), and the Program for New Century Excellent Talents in Fujian Province (2014FJ-NCET-ZR06) and by the Postgraduate Scientific Research Innovation Ability Training Plan Funding Projects of Huaqiao University(1511314013)

References

[1] Y. B. Leau, S. Manickam, and Y. W. Chong, "Network Security Situation Assessment: A Review and Discussion," Lecture Notes in Electrical Engineering, vol. 339, **(2015)** pp. 407-414.
[2] T. White, Hadoop: The Definitive Guide: Yahoo! Press, **(2010)**.
[3] K. K. Vasan and B. Surendiran, "Dimensionality reduction using Principal Component Analysis for network intrusion detection," Perspectives in Science, vol. 8, **(2016)** pp. 510-512.
[4] Y. Liu, Z. L. Sun, Y. P. Wang, and L. Shang, "An eigen decomposition based rank parameter selection approach for the NRSFM algorithm," Neurocomputing, vol. 198, **(2016)** pp. 109–113.
[5] A. Karami and M. Guerrero-Zapata, "A fuzzy anomaly detection system based on hybrid PSO-Kmeans algorithm in content-centric networks," Neurocomputing, vol. 149, **(2015)** pp. 1253-1269.
[6] R. Anil, "Mahout in Action," **(2010)**.
[7] Y. Liu, D. Feng, Y. Lian, K. Chen, and D. Wu, "Network situation prediction method based on spatial-time dimension analysis," **(2014)**.
[8] J. Cao, H. Cui, H. Shi, and L. Jiao, "Big Data: A Parallel Particle Swarm Optimization-Back-Propagation Neural Network Algorithm Based on MapReduce," Plos One, vol. 11, **(2016)**.
[9] J. Zhang, L. Xue, H. Rong, J. Wang, and F. U. Xiaodong, "Data Fusion Based on the GA-BP Neural Networks in WSNs," Journal of Shanxi University, **(2015)**.

Authors

Bowen Zhu, born in 1993. Graduate student. His main research direction includes network security and intelligence algorithm.

Yonghong Chen, Professor. Mainly engaged in computer network and information security research, including Internet of things and security, cloud computing and security, intrusion detection, digital watermarking, big data security

Yiqiao Cai, Ph.D. Mainly engaged in intelligent algorithms and their applications, data mining and other aspects of research.

Hui Tian, Ph.D**.** Mainly engaged in network and information security, cloud computing security, multimedia content security, digital forensics, information hiding and covert communications, intelligent computing

Tian Wang, Ph.D. Mainly engaged in mobile computing, networking, cloud computing and other fields of application and research.

